Adaptive Server Anywhere
SQL 参考手册

部件号：32070-01-0802-01
上次修改时间：2002 年 10 月

版权所有 © 1989-2003 Sybase, Inc. Portions 版权所有 © 2001-2002 iAnywhere Solutions, Inc. 保留所有权利。

未经 Sybase, Inc. 事先书面许可，不得以任何形式、任何手段 （电子的、机械的、手工的、光学的或其它方式）复制、传播或翻译本书的任何部分。

Sybase、 SYBASE（徽标）、 AccelaTrade、 ADA Workbench、 Adaptable Windowing Environment、 Adaptive Component Architecture、 Adaptive Server、 Adaptive Server Anywhere、 Adaptive Server Enterprise、 Adaptive Server Enterprise Monitor、 Adaptive Server Enterprise Replication、 Adaptive Server Everywhere、 Adaptive Server IQ、 Adaptive Warehouse、 AnswerBase、 Anywhere Studio、 Application Manager、 AppModeler、 APT Workbench、 APT-Build、 APT-Edit、 APT-Execute、 APT-FORMS、 APT-Library、 APT-Translator、 ASEP、 Backup Server、 BayCam、 Bit-Wise、 BizTracker、 Certified PowerBuilder Developer、 Certified SYBASE Professional、 Certified SYBASE Professional（徽标）、 ClearConnect、 Client Services、 Client-Library、 CodeBank、 Column Design、 ComponentPack、 Connection Manager、 Convoy/DM、 Copernicus、 CSP、 Data Pipeline、 Data Workbench、 DataArchitect、 Database Analyzer、 DataExpress、 DataServer、 DataWindow、 DB-Library、 dbQueue、 Developers Workbench、 Direct Connect Anywhere、 DirectConnect、 Distribution Director、 Dynamo、 e-ADK、 E-Anywhere、 e-Biz Integrator、 E-Whatever、 EC-GATEWAY、 ECMAP、 ECRTP、 eFulfillment Accelerator、 Electronic Case Management、 Embedded SQL、 EMS、 Enterprise Application Studio、 Enterprise Client/Server、 Enterprise Connect、 Enterprise Data Studio、 Enterprise Manager、 Enterprise SQL Server Manager、 Enterprise Work Architecture、 Enterprise Work Designer、 Enterprise Work Modeler、 eProcurement Accelerator、 eremote、 Everything Works Better When Everything Works Together、 EWA、 Financial Fusion、 Financial Fusion Server、 First Impression、 Formula One、 Gateway Manager、 GeoPoint、 iAnywhere、 iAnywhere Solutions、 ImpactNow、 Industry Warehouse Studio、 InfoMaker、 Information Anywhere、 Information Everywhere、 InformationConnect、 InstaHelp、 Intellidex、 InternetBuilder、 iremote、 iScript、 Jaguar CTS、 jConnect for JDBC、 KnowledgeBase、 Logical Memory Manager、 MainframeConnect、 Maintenance Express、 MAP、 MDI Access Server、 MDI Database Gateway、 media.splash、 MetaWorks、 MethodSet、 ML Query、 MobiCATS、 MySupport、 Net-Gateway、 Net-Library、 New Era of Networks、 Next Generation Learning、 Next Generation Learning Studio、 O DEVICE、 OASiS、 OASiS（徽标）、 ObjectConnect、 ObjectCycle、 OmniConnect、 OmniSQL Access Module、 OmniSQL Toolkit、 Open Biz、 Open Business Interchange、 Open Client、 Open Client/Server、 Open Client/Server Interfaces、 Open ClientConnect、 Open Gateway、 Open Server、 Open ServerConnect、 Open Solutions、 Optima++、 Partnerships that Work、 PB-Gen、 PC APT Execute、 PC DB-Net、 PC Net Library、 PhysicalArchitect、 Pocket PowerBuilder、 PocketBuilder、 Power Through Knowledge、 Power++、 power.stop、 PowerAMC、 PowerBuilder、 PowerBuilder Foundation Class Library、 PowerDesigner、 PowerDimensions、 PowerDynamo、 Powering the New Economy、 PowerJ、 PowerScript、 PowerSite、 PowerSocket、 Powersoft、 Powersoft Portfolio、 Powersoft Professional、 PowerStage、 PowerStudio、 PowerTips、 PowerWare Desktop、 PowerWare Enterprise、 ProcessAnalyst、 Rapport、 Relational Beans、 Replication Agent、 Replication Driver、 Replication Server、 Replication Server Manager、 Replication Toolkit、 Report Workbench、 Report-Execute、 Resource Manager、 RW-DisplayLib、 RW-Library、 S Designor、 S-Designor、 S.W.I.F.T. Message Format Libraries、 SAFE、 SAFE/PRO、 SDF、 Secure SQL Server、 Secure SQL Toolset、 Security Guardian、 SKILS、 smart.partners、 smart.parts、 smart.script、 SQL Advantage、 SQL Anywhere、 SQL Anywhere Studio、 SQL Code Checker、 SQL Debug、 SQL Edit、 SQL Edit/TPU、 SQL Everywhere、 SQL Modeler、 SQL Remote、 SQL Server、 SQL Server Manager、 SQL Server SNMP SubAgent、 SQL Server/CFT、 SQL Server/DBM、 SQL SMART、 SQL Station、 SQL Toolset、 SQLJ、 Stage III Engineering、 Startup.Com、 STEP、 SupportNow、 Sybase Central、 Sybase Client/Server Interfaces、 Sybase Development Framework、 Sybase Financial Server、 Sybase Gateways、 Sybase Learning Connection、 Sybase MPP、 Sybase SQL Desktop、 Sybase SQL Lifecycle、 Sybase SQL Workgroup、 Sybase Synergy Program、 Sybase User Workbench、 Sybase Virtual Server Architecture、 SybaseWare、 Syber Financial、 SyberAssist、 SybMD、 SyBooks、 System 10、 System 11、 System XI（徽标）、 SystemTools、 Tabular Data Stream、 The Enterprise Client/Server Company、 The Extensible Software Platform、 The Future Is Wide Open、 The Learning Connection、 The Model For Client/Server Solutions、 The Online Information Center、 The Power of One、 TradeForce、 Transact-SQL、 Translation Toolkit、 Turning Imagination Into Reality、 UltraLite、 UNIBOM、 Unilib、 Uninull、 Unisep、 Unistring、 URK Runtime Kit for UniCode、 Viewer、 Visual Components、 VisualSpeller、 VisualWriter、 VQL、 Warehouse Control Center、 Warehouse Studio、 Warehouse WORKS、 WarehouseArchitect、 Watcom、 Watcom SQL、 Watcom SQL Server、 Web Deployment Kit、 Web.PB、 Web.SQL、 WebSights、 WebViewer、 WorkGroup SQL Server、 XA-Library、 XA-Server 以及 XP Server 是 Sybase, Inc. 或其子公司的商标。

所有其它商标均归各自的所有者所有。

目录

关于本手册 xiii

第一部分

SQL 1

1 **SQL 语言元素 3**

关键字 4
标识符 8
字符串 10
运算符 12
表达式 17
搜索条件 26
特殊值 37
变量 43
注释 53
NULL 值 55

2 **SQL 数据类型 59**

字符数据类型 60
数值数据类型 64
货币数据类型 72
Bit 数据类型 74
日期和时间数据类型 75
二进制数据类型 83
域 86
Java 类数据类型 88
数据类型转换 94
Java / SQL 数据类型转换 97
2000 年兼容性问题 100

3 **SQL 函数 105**
函数类型 106
按字母顺序排列的函数列表 119

4 **SQL 语句 235**
使用 SQL 语句参考 236
ALLOCATE DESCRIPTOR 语句 [ESQL] 240
ALTER DATABASE 语句 242
ALTER DBSPACE 语句 246
ALTER EVENT 语句 248
ALTER FUNCTION 语句 250
ALTER PROCEDURE 语句 252
ALTER PUBLICATION 语句 254
ALTER REMOTE MESSAGE TYPE 语句 [SQL Remote] 256
ALTER SERVER 语句 258
ALTER SYNCHRONIZATION DEFINITION 语句 （不建议使用）..260
ALTER SYNCHRONIZATION SITE 语句 [MobiLink] （不建议使用） 263
ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink] 265
ALTER SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） 267
ALTER SYNCHRONIZATION USER 语句 [MobiLink] 270
ALTER TABLE 语句 272
ALTER TRIGGER 语句 279
ALTER VIEW 语句 281
ALTER WRITEFILE 语句 283
BACKUP 语句 285
BEGIN 语句 288
BEGIN TRANSACTION 语句 291
CALL 语句 294
CASE 语句 296
CHECKPOINT 语句 299
CLEAR 语句 [Interactive SQL] 300
CLOSE 语句 [ESQL] [SP] 301
COMMENT 语句 303
COMMIT 语句 305
CONFIGURE 语句 [Interactive SQL] 307
CONNECT 语句 [ESQL] [Interactive SQL] 308

CREATE COMPRESSED DATABASE 语句 312
CREATE DATABASE 语句....314
CREATE DBSPACE 语句....319
CREATE DECRYPTED FILE 语句 321
CREATE DOMAIN 语句 322
CREATE ENCRYPTED FILE 语句 324
CREATE EVENT 语句 326
CREATE EXISTING TABLE 语句....332
CREATE EXTERNLOGIN 语句 335
CREATE FUNCTION 语句....337
CREATE INDEX 语句....341
CREATE MESSAGE 语句 [T-SQL] 345
CREATE PROCEDURE 语句 347
CREATE PROCEDURE 语句 [T-SQL]....355
CREATE PUBLICATION 语句....357
CREATE REMOTE MESSAGE TYPE 语句 [SQL Remote] 360
CREATE SCHEMA 语句....362
CREATE SERVER 语句 364
CREATE STATISTICS 语句....366
CREATE SUBSCRIPTION 语句 [SQL Remote]....367
CREATE SYNCHRONIZATION DEFINITION 语句 [MobiLink] （不建议使用）....369
CREATE SYNCHRONIZATION SITE 语句 [MobiLink] （不建议使用） 372
CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]..375
CREATE SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用）....377
CREATE SYNCHRONIZATION USER 语句 [MobiLink] 380
CREATE TABLE 语句 395
CREATE TRIGGER 语句 408
CREATE TRIGGER 语句 [SQL Remote]....412
CREATE TRIGGER 语句 [T-SQL]....415
CREATE VARIABLE 语句....417
CREATE VIEW 语句 418
CREATE WRITEFILE 语句....420
DEALLOCATE 语句 422
DEALLOCATE DESCRIPTOR 语句 [ESQL]....423
声明部分 [ESQL]....424

DECLARE 语句 425
DECLARE CURSOR 语句 [ESQL] [SP] 426
DECLARE CURSOR 语句 [T-SQL] 431
DECLARE LOCAL TEMPORARY TABLE 语句 433
DELETE 语句 435
DELETE （定位）语句 [ESQL] [SP] 438
DESCRIBE 语句 [ESQL] 440
DISCONNECT 语句 [ESQL] [Interactive SQL] 444
DROP 语句 445
DROP DATABASE 语句 447
DROP CONNECTION 语句 448
DROP EXTERNLOGIN 语句 449
DROP PUBLICATION 语句 450
DROP REMOTE MESSAGE TYPE 语句 [SQL Remote] 451
DROP SERVER 语句 452
DROP STATEMENT 语句 [ESQL] 453
DROP STATISTICS 语句 454
DROP SUBSCRIPTION 语句 [SQL Remote] 456
DROP SYNCHRONIZATION DEFINITION 语句 [MobiLink] （不建议使用） 457
DROP SYNCHRONIZATION SITE 语句 [MobiLink] （不建议使用） 459
DROP SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink] 460
DROP SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） 461
DROP SYNCHRONIZATION USER 语句 [MobiLink] 463
DROP VARIABLE 语句 464
EXECUTE 语句 [ESQL] 465
EXECUTE 语句 [SP] 467
EXECUTE 语句 [T-SQL] 469
EXIT 语句 [Interactive SQL] 471
EXPLAIN 语句 [ESQL] 473
FETCH 语句 [ESQL] [SP] 475
FOR 语句 480
FORWARD TO 语句 482
FROM 子句 484
GET DATA 语句 [ESQL] 489
GET DESCRIPTOR 语句 [ESQL] 492
GET OPTION 语句 [ESQL] 494

GOTO 语句 [T-SQL] 496
GRANT 语句 497
GRANT CONSOLIDATE 语句 [SQL Remote] 502
GRANT PUBLISH 语句 [SQL Remote] 504
GRANT REMOTE 语句 [SQL Remote] 505
GRANT REMOTE DBA 语句 [SQL Remote] 508
HELP 语句 [Interactive SQL] 509
IF 语句 510
IF 语句 [T-SQL] 512
INCLUDE 语句 [ESQL] 514
INPUT 语句 [Interactive SQL] 515
INSERT 语句 520
INSTALL 语句 524
LEAVE 语句 527
LOAD STATISTICS 语句 529
LOAD TABLE 语句 530
LOCK TABLE 语句 537
LOOP 语句 539
MESSAGE 语句 541
OPEN 语句 [ESQL] [SP] 543
OUTPUT 语句 [Interactive SQL] 546
PARAMETERS 语句 [Interactive SQL] 551
PASSTHROUGH 语句 [SQL Remote] 552
PREPARE 语句 [ESQL] 553
PREPARE TO COMMIT 语句 556
PRINT 语句 [T-SQL] 557
PUT 语句 [ESQL] 558
RAISERROR 语句 [T-SQL] 560
READ 语句 [Interactive SQL] 562
READTEXT 语句 [T-SQL] 564
RELEASE SAVEPOINT 语句 565
REMOTE RESET 语句 [SQL Remote] 566
REMOVE 语句 567
REORGANIZE TABLE 语句 568
RESIGNAL 语句 570
RESTORE DATABASE 语句 571
RESUME 语句 573
RETURN 语句 574

REVOKE 语句 576
REVOKE CONSOLIDATE 语句 [SQL Remote] 578
REVOKE PUBLISH 语句 [SQL Remote] 579
REVOKE REMOTE 语句 [SQL Remote] 581
REVOKE REMOTE DBA 语句 [SQL Remote] 582
ROLLBACK 语句 583
ROLLBACK TO SAVEPOINT 语句 584
ROLLBACK TRIGGER 语句 585
SAVEPOINT 语句 586
SELECT 语句 587
SET 语句 592
SET 语句 [T-SQL] 594
SET CONNECTION 语句 [Interactive SQL] [ESQL] 598
SET DESCRIPTOR 语句 [ESQL] 599
SET OPTION 语句 601
SET OPTION 语句 [Interactive SQL] 605
SET REMOTE OPTION 语句 [SQL Remote] 606
SET SQLCA 语句 [ESQL] 608
SETUSER 语句 610
SIGNAL 语句 612
START DATABASE 语句 614
START ENGINE 语句 [Interactive SQL] 616
START JAVA 语句 617
START LOGGING 语句 [Interactive SQL] 618
START SUBSCRIPTION 语句 [SQL Remote] 619
START SYNCHRONIZATION DELETE 语句 [MobiLink] 621
STOP DATABASE 语句 623
STOP ENGINE 语句 625
STOP JAVA 语句 626
STOP LOGGING 语句 [Interactive SQL] 627
STOP SUBSCRIPTION 语句 [SQL Remote] 628
STOP SYNCHRONIZATION DELETE 语句 [MobiLink] 629
SYNCHRONIZE SUBSCRIPTION 语句 [SQL Remote] 630
SYSTEM 语句 [Interactive SQL] 632
TRIGGER EVENT 语句 633
TRUNCATE TABLE 语句 634
UNION 操作 636
UNLOAD 语句 638

UNLOAD TABLE 语句 640
UPDATE 语句 642
UPDATE （定位）语句 [ESQL] [SP] 647
UPDATE 语句 [SQL Remote] 649
VALIDATE INDEX 语句 652
VALIDATE TABLE 语句 653
WAITFOR 语句 655
WHENEVER 语句 [ESQL] 657
WHILE 语句 [T-SQL] 659
WRITETEXT 语句 [T-SQL] 660

第二部分

系统对象 661

5 系统表 663

DUMMY 系统表 664
SYSARTICLE 系统表 665
SYSARTICLECOL 系统表 666
SYSATTRIBUTE 系统表 667
SYSATTRIBUTENAME 系统表 668
SYSCAPABILITY 系统表 669
SYSCAPABILITYNAME 系统表 670
SYSCOLLATION 系统表 671
SYSCOLLATIONMAPPINGS 系统表 672
SYSCOLPERM 系统表 673
SYSCOLSTAT 系统表 674
SYSCOLUMN 系统表 676
SYSDOMAIN 系统表 679
SYSEVENT 系统表 680
SYSEVENTTYPE 系统表 682
SYSEXTENT 系统表 683
SYSEXTERNLOGINS 系统表 684
SYSFILE 系统表 685
SYSFKCOL 系统表 686
SYSFOREIGNKEY 系统表 687
SYSGROUP 系统表 689
SYSINDEX 系统表 690

SYSINFO 系统表 692
SYSIXCOL 系统表 694
SYSJAR 系统表 695
SYSJARCOMPONENT 系统表 696
SYSJAVACLASS 系统表 697
SYSLOGIN 系统表 699
SYSOPTBLOCK 系统表 700
SYSOPTION 系统表 701
SYSOPTJOINSTRATEGY 系统表 702
SYSOPTORDER 系统表 703
SYSOPTQUANTIFIER 系统表 704
SYSOPTREQUEST 系统表 705
SYSOPTREWRITE 系统表 706
SYSOPTSTAT 系统表 707
SYSPROCEDURE 系统表 708
SYSPROCPARM 系统表 710
SYSPROCPERM 系统表 712
SYSPUBLICATION 系统表 713
SYSREMOTEOPTION 系统表 714
SYSREMOTEOPTIONTYPE 系统表 715
SYSREMOTETYPE 系统表 716
SYSREMOTEUSER 系统表 717
SYSSCHEDULE 系统表 719
SYSSERVERS 系统表 721
SYSSQLSERVERTYPE 系统表 722
SYSSUBSCRIPTION 系统表 723
SYSSYNC 系统表 724
SYSTABLE 系统表 726
SYSTABLEPERM 系统表 729
SYSTRIGGER 系统表 731
SYSTYPEMAP 系统表 734
SYSUSERMESSAGES 系统表 735
SYSUSERPERM 系统表 736
SYSUSERTYPE 系统表 738
其它系统表 740

6 **系统视图 743**
简介 744
Transact-SQL 兼容性视图 750

7 **系统过程和函数 753**
系统过程概述 754
系统和分类存储过程 755
系统扩展存储过程 803
Adaptive Server Enterprise 系统和分类过程 814

索引 817

关于本手册

主题

本手册提供了 Adaptive Server Anywhere 所使用的 SQL 语言的完整参考。还介绍了 Adaptive Server Anywhere 系统表和过程。

其它手册就如何执行特定的任务提供了目的性更强的相关内容，而本手册仅提供了可用的 SQL 语法和系统对象的完整列表。

读者

本手册面向所有的 Adaptive Server Anywhere 用户。它包含对 MobiLink 和 SQL Remote 用户特别重要的内容。您可以将它与文档集中的其它手册结合使用。

SQL Anywhere Studio 文档

本手册是 SQL Anywhere 文档集的一部分。本节介绍文档集中包含的手册及使用方法。

SQL Anywhere Studio 文档集

SQL Anywhere Studio 文档集包括以下手册：

- **SQL Anywhere Studio 介绍** 此手册概述了 SQL Anywhere Studio 数据库管理和同步技术。手册中包括一些教程，分别介绍了 SQL Anywhere Studio 的各个组件。
- **SQL Anywhere Studio 的新功能** 此手册面向该软件早期版本的用户。列出了本产品的此版本和以前版本的新功能，并介绍了升级步骤。
- **Adaptive Server Anywhere 入门手册** 此手册面向对关系数据库或 Adaptive Server Anywhere 不熟悉的人员。它提供了 Adaptive Server Anywhere 数据库管理系统的入门知识，并介绍了有关设计、构建和使用数据库的内容。
- **Adaptive Server Anywhere 数据库管理指南** 此手册包含有关运行、管理和配置数据库的内容。
- **Adaptive Server Anywhere SQL 用户指南** 此手册介绍了如何设计和创建数据库；如何导入、导出和修改数据；如何检索数据以及如何创建存储过程和触发器。

- **Adaptive Server Anywhere SQL 参考手册** 此手册提供了有关 Adaptive Server Anywhere 所使用的 SQL 语言的完整参考。还介绍了 Adaptive Server Anywhere 系统表和过程。

- **Adaptive Server Anywhere 编程指南** 此手册介绍了如何使用 C、C++ 和 Java 编程语言来建立和部署数据库应用程序。 Visual Basic 和 PowerBuilder 等工具的用户可以使用那些工具所提供的编程接口。

- **Adaptive Server Anywhere 错误消息** 此手册提供了 Adaptive Server Anywhere 错误消息以及诊断信息的完整列表。

- **Adaptive Server Anywhere C2 安全补充说明** Adaptive Server Anywhere 7.0 荣获了美国政府授予的 TCSEC （可信计算机系统评估标准） C2 安全等级。那些希望以符合 C2 认证环境的方式运行 Adaptive Server Anywhere 当前版本的用户可能会对此手册的内容感兴趣。这本书*未* 包括该产品获得认证后新增的安全功能部分。

- **MobiLink 同步技术用户指南** 此手册介绍了用于移动计算的 MobiLink 数据同步系统的各个方面。 MobiLink 数据同步系统支持在单个 Oracle、 Sybase、 Microsoft 或 IBM 数据库与多个 Adaptive Server Anywhere 或 UltraLite 数据库之间共享数据。

- **SQL Remote 用户指南** 此手册介绍了用于移动计算的 SQL Remote 数据复制系统的各个方面。 SQL Remote 数据复制系统支持使用电子邮件或文件传输等间接链接，在单个 Adaptive Server Anywhere 或 Adaptive Server Enterprise 数据库与多个 Adaptive Server Anywhere 数据库之间共享数据。

- **UltraLite 用户指南** 此手册介绍了如何利用 Adaptive Server Anywhere 数据库的 UltraLite 部署技术，创建用于小型设备 （例如手持式管理器）的数据库应用程序。

- **UltraLite User's Guide for PenRight! MobileBuilder** 此手册面向 PenRight! MobileBuilder 开发工具的用户。手册中介绍了如何在 MobileBuilder 编程环境中使用 UltraLite 技术。

- **SQL Anywhere Studio 帮助** 此手册仅以联机方式提供。手册中包含 Sybase Central、 Interactive SQL 和其它图形化工具的上下文相关帮助。

除了本文档集外， SQL Modeler 和 InfoMaker 还有自己的联机文档。

文档格式

SQL Anywhere Studio 提供了下列格式的文档：

- **联机手册** 联机手册包含完整的 SQL Anywhere Studio 文档，其中包括印刷手册和 SQL Anywhere 工具的上下文相关帮助。每次发布产品的维护版本时都会对联机手册进行更新，因此这些手册是最全和最新的文档来源。

  若要在 Windows 操作系统中访问此联机手册，请选择 " 开始 " → " 程序 " → "Sybase SQL Anywhere 8" → " 联机手册 "。可以使用左窗格中的 HTML 帮助目录、索引和搜索功能以及右窗格中的链接和菜单来导航联机手册。

  若要在 UNIX 操作系统中访问联机手册，请在命令提示符下运行下面的命令：

  ```
  dbbooks
  ```

- **可打印手册** 以一组 PDF 文件的形式提供的 SQL Anywhere 手册，可以用 Adobe Acrobat Reader 来查看。

  在光盘上的 `pdf_docs` 目录中可以找到这些 PDF 文件。运行安装程序时可以选择安装这些文件。

- **印刷手册** SQL Anywhere Studio 的包装盒中提供了以下手册：

  - *SQL Anywhere Studio 介绍*。
  - *Adaptive Server Anywhere 入门手册*。
  - *SQL Anywhere Studio 快速参考手册*。此手册只有印刷版本。

  SQL Anywhere 整套文档集可在 Sybase 销售部门或 Sybase 网上商店 e-Shop 购买。 e-Shop 的网址是：http://e-shop.sybase.com/cgi-bin/eshop.storefront/。

文档约定

这一节列出了在本文档中使用的印刷和图形方面的约定。

语法约定

在 SQL 语法说明中使用了以下约定：

- **关键字** 所有 SQL 关键字的显示方式与以下示例中的 ALTER TABLE 类似：

ALTER TABLE [*owner*.]*table-name*

- **占位符** 必须替换为相应的标识符或表达式的项，其显示方式与以下示例中的单词 *owner* 和 *table-name* 类似。

  ALTER TABLE [*owner*.]*table-name*

- **重复项** 对于重复项列表，只显示列表中的一个元素，后跟省略号 （三个句点），如以下示例中的 *column-constraint*：

  ADD *column-definition* [*column-constraint, …*]

  允许指定一个或多个列表元素。如果指定了多个元素，必须用逗号将它们隔开。

- **可选部分** 语句的可选部分放在方括号内。

  RELEASE SAVEPOINT [*savepoint-name*]

  上面的方括号表示 *savepoint-name* 是可选的。不要键入方括号。

- **选项** 如果不必选择项列表中的项或者只能选择一个，则用垂直条分隔这些项，并将列表放在方括号内。

  [**ASC** | **DESC**]

  例如，可以从 ASC、 DESC 中任选一个，或者一个也不选。不要键入方括号。

- **二选一的选项** 如果必须而且只能选择一个选项，则将可供选择的选项放在大括号内。

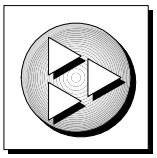
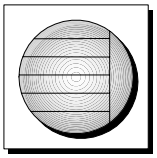
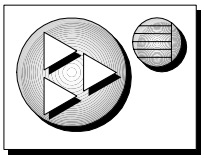
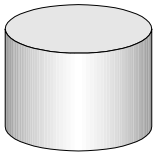
  [**QUOTES** { **ON** | **OFF** }]

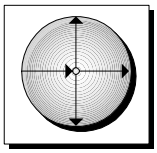
  如果选择了 QUOTES 选项，则必须提供 ON 或者 OFF。不要键入方括号和大括号。

- **一个或多个选项** 如果选择多个选项，则用逗号将选项分开。

  { **CONNECT**, **DBA**, **RESOURCE** }

图标 本文档中使用了下列图标：

图标	含义
	客户端应用程序。
	数据库服务器，例如 Sybase Adaptive Server Anywhere 或 Adaptive Server Enterprise。
	UltraLite 应用程序和数据库服务器。在 UltraLite 中，数据库服务器和应用程序是同一进程的组成部分。
	数据库。在某些高级别的示意图中，可以使用此图标表示数据库和管理数据库的数据库服务器。
	用于复制或同步的中间件。它们帮助在数据库之间共享数据。 MobiLink Synchronization Server、SQL Remote Message Agent 以及用于复制服务器的复制代理 (Log Transfer Manager) 都是中间件。

图标	含义
	Sybase Replication Server。
	编程接口。

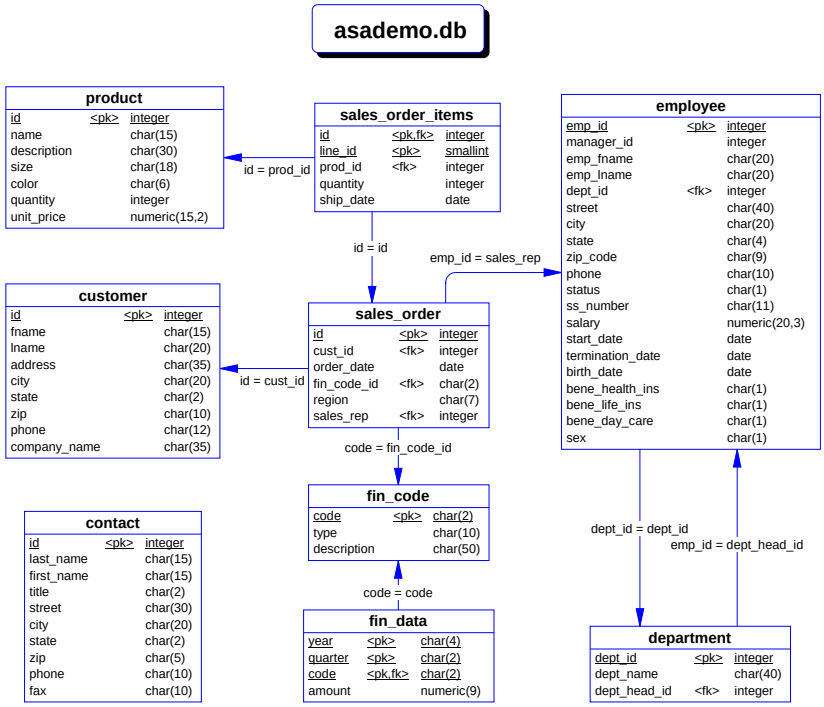
Adaptive Server Anywhere 示例数据库

文档中的许多示例都使用了 Adaptive Server Anywhere 示例数据库。

该示例数据库保存在名为 `asademo.db` 的文件中，此文件位于 SQL Anywhere 目录下。

该示例数据库描绘了一家小公司。它包含该公司的内部信息（雇员、部门和财务）以及产品信息和销售信息（销售订单、客户和联系方式）。数据库中的所有信息都是虚构的。

下图显示了该示例数据库中的表以及这些表之间的关系。



查找更多信息并提供反馈

我们欢迎您就本文档提出意见、建议或其它反馈信息。

您可以通过专为讨论 SQL Anywhere 技术而建立的新闻组来提供有关本文档和软件的反馈信息。在 `forums.sybase.com` 新闻服务器上可以找到这些新闻组。

包括下面这些新闻组：

- sybase.public.sqlanywhere.general。
- sybase.public.sqlanywhere.linux。
- sybase.public.sqlanywhere.mobilink。

- sybase.public.sqlanywhere.product_futures_discussion。
- sybase.public.sqlanywhere.replication。
- sybase.public.sqlanywhere.ultralite。

新闻组免责声明

iAnywhere Solutions 没有义务为其新闻组提供解决方案、信息或建议；除提供系统操作员监控服务和确保新闻组的运行和可用性外，iAnywhere Solutions 没有义务提供任何其它服务。

如果时间允许， iAnywhere Solutions 技术顾问和其他员工也会为新闻组服务提供帮助。他们是在自愿的基础上提供帮助，所以可以无法定期提供解决方案和信息。他们可以提供多少帮助取决于他们的工作量。

第一部分　SQL

这部分介绍 Adaptive Server Anywhere 所使用的 SQL 语言，包括数据类型、函数和语句。

第 1 章

SQL 语言元素

关于本章　　本章介绍 SQL 语言的元素和约定。

关键字

每个 SQL 语句都包含一个或多个关键字。 SQL 的关键字不区分大小写，但在这些手册中，关键字都以大写字母显示。

例如，在下面的语句中， SELECT 和 FROM 是关键字：

```
SELECT *
FROM employee
```

下列语句等同于上一语句：

```
Select *
From employee
select * from employee
sELECT * FRoM employee
```

有些关键字只有用双引号引起来才能用作标识符。这些关键字被称为保留字。有些关键字 （如 DBA）则不需要加双引号，也不是保留字。

保留字

SQL 中的一些关键字也是**保留字**。要在 SQL 语句中使用保留字作为标识符，必须用双引号将它引起来。 SQL 语句中出现的很多关键字都是保留字，但并非全部都是。例如，必须使用以下语法来检索名为 SELECT 的表的内容。

```
SELECT *
FROM "SELECT"
```

由于 SQL 的关键字不区分大小写，以下各保留字可以以大写、小写或二者的任意组合形式出现。与以下某保留字仅在大小写形式上有所不同的所有字符串都是保留字。

如果使用嵌入式 SQL，可以利用 DB library 函数 **SQL_needs_quotes** 确定一个字符串是否需要加上引号。如果一个字符串是保留字，或者包含标识符中通常不允许的字符，则该字符串必须加上引号。

Adaptive Server Anywhere 中有以下 SQL 保留字：

保留字	保留字	保留字	保留字
add	all	alter	and
any	as	asc	backup
begin	between	bigint	binary
bit	bottom	break	by
call	capability	cascade	case
cast	char	char_convert	character
check	checkpoint	close	comment
commit	connect	constraint	contains
continue	convert	create	cross
cube	current	cursor	date
dbspace	deallocate	dec	decimal
declare	default	delete	deleting
desc	distinct	do	double
drop	dynamic	else	elseif
encrypted	end	endif	escape
exception	exec	execute	existing
exists	externlogin	fetch	first
float	for	foreign	forward
from	full	goto	grant
group	having	holdlock	identified
if	in	index	inner
inout	insensitive	insert	inserting
install	instead	int	integer

保留字	保留字	保留字	保留字
integrated	into	iq	is
isolation	join	key	left
like	lock	login	long
match	membership	message	mode
modify	natural	new	no
noholdlock	not	notify	null
numeric	of	off	on
open	option	options	or
order	others	out	outer
over	passthrough	precision	prepare
primary	print	privileges	proc
procedure	publication	raiserror	readtext
real	reference	references	release
remote	remove	rename	reorganize
resource	restore	restrict	return
revoke	right	rollback	rollup
save	savepoint	schedule	scroll
select	sensitive	session	set
setuser	share	smallint	some
sqlcode	sqlstate	start	stop
subtrans	subtransaction	synchronize	syntax_error
table	temporary	then	time
timestamp	tinyint	to	top
tran	trigger	truncate	tsequal

保留字	保留字	保留字	保留字
union	unique	unknown	unsigned
update	updating	user	using
validate	values	varbinary	varchar
variable	varying	view	wait
waitfor	when	where	while
with	with_lparen	work	writetext

标识符

作用

标识符是数据库中的对象 （如用户 ID、表和列）的名称。

说明

当以下任一条件成立时，标识符需要用双引号引起来或用方括号括起来：

- 标识符包含空格。
- 标识符的首字符不是字母字符 （定义如下）。
- 标识符包含保留字。
- 标识符包含字母和数字以外的其它字符。

字母字符包括字母表中的字母，以及下划线 (_)、 at 符号 (@)、井号 (#) 和美元符号 ($)。数据库归类序列指出了哪些字符被视为字母字符或数字字符。

如果 QUOTED_IDENTIFIER 数据库选项被设置为 OFF，则双引号可用于界定 SQL 字符串，但不能用于界定标识符。不过，不管 QUOTED_IDENTIFIER 如何设置，方括号都始终可以用于界定标识符。

对于 Open Client 和 JDBC 连接， QUOTED_IDENTIFIER 选项的缺省设置为 OFF，其它情况下缺省为 ON。

在标识符中表示引号的方法可以是在一个引号后加上另一个引号。

标识符的最大长度为 128 个字节。

示例

以下各种情况均为有效标识符。

```
Surname
"Surname"
[Surname]
SomeBigName
"Client Number"
"With one double quotation " mark"
```

另请参阅

有关保留字的完整列表，请参阅 "" 保留字 " 第 4 页 "。

有关 QUOTED_IDENTIFIER 选项的信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > QUOTED_IDENTIFIER 选项 [compatibility]"。

字符串

字符串具有以下类型：

- 文字字符串
- 具有 CHAR 或 VARCHAR 数据类型的表达式。

具有 CHAR 数据类型的表达式可以是内置函数或用户定义的函数，或其它多种可用表达式之一。

有关表达式的更多信息，请参阅 " 表达式 " 第 17 页。

文字字符串是用撇号 （' 单引号 '）围起来的任何字符序列。字符数据类型的 SQL 变量可以保存字符串。下面是一个文字字符串的简单例子：

```
'This is a string.'
```

字符串中的特殊字符

在字符串中用转义序列表示特殊字符，如下所示：

- 要在字符串中表示撇号，需在一行中使用两个撇号。例如，

  ```
  'John''s database'
  ```

- 要表示换行字符，需使用后跟 n 的反斜线 （\n）。例如，

  ```
  'First line:\nSecond line:'
  ```

- 要表示反斜线字符，需在一行中使用两个反斜线 （\\）。例如，

  ```
  'c:\\temp'
  ```

- 十六进制转义序列可用于任何字符 （不论是否为可打印字符）。十六进制转义序列由反斜线后跟一个 x 再跟两个十六进制数字构成 （例如， \x6d 表示字母 m）。例如，

  ```
  '\x00\x01\x02\x03'
  ```

兼容性

为与 Adaptive Server Enterprise 兼容，可将 QUOTED_IDENTIFIER 数据库选项设置为 OFF。在这种设置下，仍可以用双引号来标记字符串的开始和结束。该选项的缺省设置为 ON。

有关 QUOTED_IDENTIFIER 选项的信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > QUOTED_IDENTIFIER 选项 [compatibility]"。

运算符

本节将介绍算术、字符串以及位运算符。有关比较运算符的信息，请参阅 "" 搜索条件 " 第 26 页 " 一节。

正常的运算优先级顺序仍适用。圆括号中的表达式先进行运算，然后是乘除，继而加减。字符串连接在加减之后进行。

有关更多信息，请参阅 "" 运算符优先级 " 第 16 页 "。

比较运算符

比较条件的语法如下：

expression compare expression

其中 *compare* 是比较运算符。以下是可用的比较运算符：

运算符	说明
=	等于
>	大于
<	小于
>=	大于或等于
<=	小于或等于
!=	不等于
<>	不等于
!>	不大于
!<	不小于

大小写敏感性
除非创建的数据库区分大小写，否则所有字符串比较运算都*不区分大小写*。

兼容性

- **尾随空白**　Adaptive Server Enterprise 在执行比较运算时忽略字符数据中的任何尾随空白。比较字符串时， Adaptive Server Anywhere 的行为受 -b 命令行开关控制，该开关可在创建数据库时设置。

  有关空白填充的更多信息，请参阅 "ASA 数据库管理指南 > 数据库管理实用程序 > Initialization 实用程序选项 "。

- **大小写敏感性**　缺省情况下，创建 Adaptive Server Anywhere 数据库时不区分大小写，而创建 Adaptive Server Enterprise 数据库时区分大小写。执行比较时不仅注意所在的数据库，还注意大小写。创建数据库时可以用 -c 命令行开关控制 Adaptive Server Anywhere 数据库的大小写敏感性。

  有关字符串比较时大小写敏感性的更多信息，请参阅 "ASA 数据库管理指南 > 数据库管理实用程序 > Initialization 实用程序选项 "。

逻辑运算符

可以使用 AND、 OR 和 NOT 来将搜索条件组合起来。

使用 AND 将条件组合起来的方式如下：

condition1 **AND** *condition2*

如果两个条件都为 TRUE，则组合条件为 TRUE ；如果其中某一条件为 FALSE，则组合条件为 FALSE ；其它情况下为 UNKNOWN。

使用 OR 将条件组合起来的方式如下：

condition1 **OR** *condition2*

如果其中某一为 TRUE，则组合条件为 TRUE ；如果两个条件都为 FALSE，则组合条件为 FALSE ；其它情况下为 UNKNOWN。

NOT 运算符的语法如下：

NOT *condition*

如果 *condition* 为 FALSE，则 NOT 条件为 TRUE ； 如果 *condition* 为 TRUE，则为 FALSE ； 如果 *condition* 为 UNKNOWN，则为 UNKNOWN。

IS 运算符提供了测试逻辑值的方法。 IS 运算符的语法如下：

expression **IS** [**NOT**] *truth-value*

如果 *expression* 的值为提供的 *truth-value* （必须是 TRUE、 FALSE、 UNKNOWN 或 NULL 其中之一），则条件为 TRUE。否则，条件值为 FALSE。

有关更多信息，请参阅 "" 三值逻辑 " 第 33 页 "。

兼容性

- 逻辑运算符在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 之间兼容。

算术运算符

expression + expression 加法。如果其中某一表达式值为 NULL，则结果为 NULL。

expression - expression 减法。如果其中某一表达式值为 NULL，则结果为 NULL。

-expression 非。如果表达式值为 NULL，则结果为 NULL。

expression * expression 乘法。如果其中某一表达式值为 NULL，则结果为 NULL。

expression / expression 除法。如果其中某一表达式值为 NULL，或第二个表达式为 0，则结果为 NULL。

expression % expression 两个整数相除之后，模取整数余数。例如，21 % 11 = 10，因为 21 除以 11 等于 1，余数为 10。

兼容性

- **模** 仅当 PERCENT_AS_COMMENT 选项被设置为 OFF 时，才能在 Adaptive Server Anywhere 中使用 % 运算符。缺省值为 ON。

字符串运算符

expression || expression　字符串连接 （两个竖直条）。如其中某一字符串为 NULL，则连接时将其视为空字符串。

expression + expression　字符串连接运算的另一种表达方式。使用 + 连接运算符时，必须确保操作数已被显式设置为字符数据类型，而不是依赖隐式数据转换。

例如，以下查询返回整数值 **579**：

```
SELECT 123 + 456
```

而以下查询则返回字符串 **123456**：

```
SELECT '123' + '456'
```

可以使用 CAST 或 CONVERT 函数来显式转换数据类型。

兼容性

- Adaptive Server Enterprise 不支持 **||** 连接运算符。

标准和兼容性

- **SQL/92**　|| 运算符是 SQL/92 的字符串连接运算符。
- **Sybase**　Adaptive Server Enterprise 支持 + 运算符。

位运算符

在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中，以下运算符都只适用于整数数据类型。

运算符	说明
&	位运算符 " 与 "
|	逐位或
^	逐位异或
~	逐位非

位运算符 &、| 和 ~ 与逻辑运算符 AND、 OR 和 NOT 不能互换。

示例

例如，以下语句选择设置了正确位的行。

```
SELECT *
FROM tableA
WHERE (options & 0x0101) <> 0
```

连接运算符

除了在 FROM 子句中使用表表达式的 SQL/92 连接语法外， Adaptive Server Anywhere 还支持 Transact-SQL 外连接运算符 *= 和 =*。

运算符优先级

表达式中的运算符优先级如下。列表顶部的运算符先于底部的运算符进行运算。

1. 一元运算符 （需要单操作数的运算符）
2. **.**'（Java 引用运算符）
3. **&**, **|** , **^**, **~**
4. ***, /, %***
5. **+**, **-**
6. **||**
7. **not**
8. **and**
9. **or**

在一个表达式中使用一个以上的运算符时，建议使用圆括号显式设置运算顺序，而不是依赖 Adaptive Server Enterprise 和 Adaptive Server Anywhere 之间相同的运算符优先级。

表达式

语法

expression:
case-expression
| *constant*
| [*correlation-name*.]*column-name* [*java-ref*]
| - *expression*
| *expression operator expression*
| (*expression*)
| *function-name* (*expression*, …)
| *if-expression*
| [*java-package-name*.] *java-class-name java-ref*
| *special value*
| (*subquery*)
| *variable-name* [*java-ref*]

参数

case-expression:
CASE *expression*
WHEN *expression*
THEN *expression,…*
[**ELSE** *expression*]
END

alternative form of case-expression:
CASE
WHEN *search-condition*
THEN *expression,…*
[**ELSE** *expression*]
END

constant:
integer | *number* | *string* | *host-variable*

special-value:
CURRENT { **DATE** | **TIME** | **TIMESTAMP** }
| **NULL**
| **SQLCODE**
| **SQLSTATE**
| **USER**

if-expression:
IF *condition*
THEN *expression*
[**ELSE** *expression*]
ENDIF

java-ref:
.*field-name* [*java-ref*]
| >> *field-name* [*java-ref*]
| **.***method-name* ([*expression*,…]) [*java-ref*]
| >> *method-name* ([*expression*,…]) [*java-ref*]

operator:
{ **+** | **-** | ***** | **/** | **||** | **%** }

用法

任何地方。

授权

必须连接到数据库。

副作用

无。

另请参阅

" 表达式中的常量 " 第 19 页

" 特殊值 " 第 37 页

" 表达式中的列名称 " 第 19 页

"SQL 函数 " 第 105 页

" 表达式中的子查询 " 第 19 页

" 搜索条件 " 第 26 页

"SQL 数据类型 " 第 59 页

" 变量 " 第 43 页

"CASE 表达式 " 第 20 页

说明

表达式由若干不同种类的元素构成。以下几节将就此进行论述。

有关函数的信息，请参阅 ""SQL 函数 " 第 105 页 "。有关变量的信息，请参阅 "" 变量 " 第 43 页 "。

标准和兼容性

- Adaptive Server Enterprise 不支持 IF 条件。
- Adaptive Server Enterprise 目前不支持 Java 表达式。
- 有关其它差异，请参阅以下章节中对各类表达式的单独说明。

表达式中的常量

常量是数字或字符串。字符串常量用撇号 （' 单引号 '）围起来。在一行中使用两个撇号来表示字符串中的撇号。

表达式中的列名称

列名称是一个跟在可选相关名后面的标识符。（相关名通常是表名。有关相关名的更多信息，请参阅 ""FROM 子句 " 第 484 页 "。）如果列名称含有字母、数字和下划线以外的其它字符，必须用引号 （"）将它引起来。例如，以下的字符是有效的列名称：

```
employee.name
address
"date hired"
"salary"."date paid"
```

有关标识符的更多信息，请参阅 "" 标识符 " 第 8 页 "。

表达式中的子查询

子查询是一个嵌入在另一个 SELECT、 INSERT、 UPDATE 或 DELETE 语句中或者其它子查询中的 SELECT 语句。

SELECT 语句必须用圆括号括起来，而且必须包含一个且只能包含一个选择列表项。用作表达式时，子查询通常只能返回一个值。

子查询可以在能够使用列名称的任何地方使用。例如，子查询可以在另一个 SELECT 语句的选择列表中使用。

有关子查询的其它用法，请参阅 "" 搜索条件中的子查询 " 第 27 页 "。

IF 表达式

IF 表达式的语法如下：

IF *condition*
THEN *expression1*
[**ELSE** *expression2*]
ENDIF

该表达式的返回方式如下：

- 如果 *condition* 为 TRUE，则 IF 表达式返回 *expression1*。
- 如果 *condition* 为 FALSE，则 IF 表达式返回 *expression2*。
- 如果 *condition* 为 FALSE，且无 *expression2*，则 IF 表达式返回 NULL。
- 如果 *condition* 为 UNKNOWN，则 IF 表达式返回 NULL。

有关 TRUE、 FALSE 和 UNKNOWN 条件的更多信息，请参阅 ""NULL 值 " 第 55 页 " 和 "" 搜索条件 " 第 26 页 "。

IF 语句不同于 IF 表达式
不要混淆 IF 表达式的语法和 IF 语句的语法。

有关 IF 语句的信息，请参阅 ""IF 语句 " 第 510 页 "。

CASE 表达式

CASE 表达式提供条件 SQL 表达式。 Case 表达式可以在能够使用表达式的任何地方使用。

CASE 表达式的语法如下：

CASE *expression*
WHEN *expression*
THEN *expression, …*
[**ELSE** *expression*]
END

如果 CASE 语句后的表达式等于 WHEN 语句后的表达式，则返回 THEN 语句后的表达式。否则，返回 ELSE 语句后的表达式 （如果它存在）。

例如，以下代码用 CASE 表达式作为 SELECT 语句的选择列列表中的第二列。

```
SELECT id,
   ( CASE name
      WHEN 'Tee Shirt' then 'Shirt'
      WHEN 'Sweatshirt' then 'Shirt'
      WHEN 'Baseball Cap' then 'Hat'
      ELSE 'Unknown'
   END ) as Type
FROM "DBA".Product
```

可以替换的语法如下：

CASE
WHEN *search-condition*
THEN *expression, …*
[**ELSE** *expression*]
END

如果满足 WHEN 语句后的搜索条件，则返回 THEN 语句后的表达式。否则，返回 ELSE 语句后的表达式 （如果它存在）。

例如，以下语句用 CASE 表达式作为 SELECT 语句的选择列列表中的第三列，以将字符串与搜索条件相关联。

```
SELECT id, name,
   ( CASE
      WHEN name='Tee Shirt' then 'Sale'
      WHEN quantity >= 50  then 'Big Sale'
      ELSE 'Regular price'
   END ) as Type
FROM "DBA".Product
```

用于简写 CASE 表达式的 NULLIF 函数

NULLIF 函数提供了一种以短格式写出某些 CASE 语句的方法。NULLIF 语法如下：

NULLIF (*expression-1*, *expression-2*)

NULLIF 比较两个表达式的值。如果第一个表达式的值与第二个表达式的值相等， NULLIF 返回 NULL。如果它们不相等， NULLIF 则返回第一个表达式。

CASE 语句不同于 CASE 表达式
不要混淆 CASE 表达式的语法和 CASE 语句的语法。

有关 CASE 语句的信息，请参阅 ""CASE 语句 " 第 296 页 "。

Java 表达式

以下几种 Java 表达式可用作 SQL 表达式：

- **Java 字段** 在任何需要表达式的地方，都可以调用已安装的 Java 类的任何字段。表达式的数据类型是根据 ""Java 到 SQL 的数据类型转换 " 第 97 页 " 中的表由 Java 字段数据类型进行转换得来的。实例字段和类字段都可用作表达式。
- **Java 方法** 在任何需要表达式的地方，都可以调用已安装的 Java 类的任何方法。表达式数据类型是根据 ""Java 到 SQL 的数据类型转换 " 第 97 页 " 中的表由 Java 方法的返回类型进行转换得来的。实例方法和类方法都可用作表达式。
- **Java 对象** NEW 运算符是 SQL 语言的扩展，它可使 SQL 语言与 Java 语法更加类似。

  NEW SQL 运算符与 Java 代码中的 **new** 关键字执行同样的操作：调用 Java 类的构造函数方法。 NEW 表达式的数据类型是一个 Java 类，具体说来就是所构建的 Java 类。

  以下表达式调用 String 类 （java.lang 包的成员）的构造函数方法。

  ```
  NEW java.lang.String( 'This argument is optional' )
  ```

  该表达式返回对新创建的 String 对象的引用，它可以传递给类型 *java.lang.String* 的变量或列。

  所调用的构造函数方法决定了参数的数目和类型。

  构造函数方法被调用的类必须首先安装到数据库。

有关类和实例的字段及方法的更多信息，请参阅 "ASA 编程指南 > 介绍数据库中的 Java > Java 讲座 "。

引用字段和方法

在从 Java 代码中引用 Java 字段或方法时，可以使用点 (.) 运算符。例如，要调用 DriverManager 类的 getConnection 方法，需使用如下语句：

```
conn = DriverManager.getConnection( temp.toString() ,
_props )
```

有两种方法可以从 SQL 语句中引用 Java 字段或方法。可以使用点运算符或 >> 运算符。

点运算符的优点是它看起来像 Java 代码；缺点是，在 SQL 中，点用于表示所有者、表和列的层次，因此阅读起来可能会产生混淆。

使用点运算符，用以下语句从 SQL 中调用名为 Employee 的对象的 name 方法：

```
select Employee.name ...
```

上述表达式可以引用 *Employee* 表的 *name* 列。

运算符 >> 的含义很明确，但看起来不像 Java 程序设计员所期望的那样。

表达式的兼容性

下列表介绍了 Adaptive Server Enterprise 与 Adaptive Server Anywhere 之间表达式和常量的兼容性。这些表只是一个指南，而且，" 两者都支持 " 的标记并不意味着在任何情况下出于任何目的表达式均以相同的方式执行。有关详细说明，请参考 Adaptive Server Enterprise 文档和 Adaptive Server Anywhere 文档中各表达式的相关内容。

下表中， **expr** 表示表达式， **op** 表示运算符。

表达式	支持情况
常量	两者都支持
列名称	两者都支持
变量名称	两者都支持
函数 (expr)	两者都支持

表达式	支持情况
- expr	两者都支持
expr op expr	两者都支持
(expr)	两者都支持
（子查询）	两者都支持
if 表达式	只有 Adaptive Server Anywhere

常量	支持情况
整数	两者都支持
数字	两者都支持
' 字符串 '	两者都支持
特殊常量	两者都支持
主机变量	Adaptive Server Anywhere

分隔字符串的缺省解释

缺省情况下， Adaptive Server Enterprise 和 Adaptive Server Anywhere 使用不同的方法来表示分隔字符串，即，用撇号 （单引号）将字符串围起来和用引号 （双引号）将字符串引起来。

Adaptive Server Anywhere 遵循 SQL/92 约定，用撇号围起来的字符串是常量表达式，用引号 （双引号）引起来的字符串是界定标识符 （数据库对象的名称）。 Adaptive Server Enterprise 则遵循如下约定：用引号引起来的字符串是常量，而在缺省情况下不允许界定标识符并将其按字符串处理。

quoted_identifier 选项

Adaptive Server Enterprise 和 Adaptive Server Anywhere 都提供了一个 **quoted_identifier** 选项，通过该选项，可以更改对分隔字符串的解释。缺省情况下， **quoted_identifier** 选项在 Adaptive Server Enterprise 中被设置为 OFF，在 Adaptive Server Anywhere 中被设置为 ON。

如果 **quoted_identifier** 选项被设置为 Off，则 SQL 保留字不能用作标识符。

有关保留字的完整列表，请参阅 "" 保留字 " 第 4 页 "。

设置该选项

虽然大多数 Adaptive Server Enterprise 连接选项不支持 Transact-SQL SET 语句，但 **quoted_identifier** 选项支持该语句。

在 Adaptive Server Anywhere 或 Adaptive Server Enterprise 中，以下语句将 **quoted_identifier** 选项的设置更改为 ON：

```
SET quoted_identifier ON
```

quoted_identifier 选项被设置为 ON 时， Adaptive Server Enterprise 允许用引号分隔表、视图和列的名称。在 Adaptive Server Enterprise 中不能分隔其它对象名称。

在 Adaptive Server Anywhere 或 Adaptive Server Enterprise 中，以下语句将 **quoted_identifier** 选项的设置更改为 OFF：

```
SET quoted_identifier OFF
```

分隔字符串的兼容解释

在 Adaptive Server Enterprise 和 Adaptive Server Anywhere 中，只要各个 DBMS 中的 **quoted_identifier** 选项都设为相同的值，就可以选用 SQL/92 或缺省 Transact-SQL 约定。

示例

如果选择在 **quoted_identifier** 选项为 ON （Adaptive Server Anywhere 中的缺省设置）时进行操作，则涉及 SQL 保留字 **user** 的以下语句对两个 DBMS 均有效。

```
CREATE TABLE "user" (
   col1 char(5)
) ;
INSERT "user" ( col1 )
VALUES ( 'abcde' ) ;
```

如果选择在 **quoted_identifier** 选项为 Off （Adaptive Server Enterprise 中的缺省设置）时操作，则以下语句对两个 DBMS 均有效。

```
SELECT *
FROM employee
WHERE emp_lname = "Chin"
```

搜索条件

作用

用于指定 WHERE 子句、 HAVING 子句、 CHECK 子句、表连接中 ON 短语，或者 IF 表达式的搜索条件。

语法

search-condition:
search-condition:
expression compare expression
| *expression compare* { [**ANY** | **SOME**] | **ALL** } (*subquery*)
| *expression* **IS** [**NOT**] **NULL**
| *expression* [**NOT**] **BETWEEN** *expression* **AND** *expression*
| *expression* [**NOT**] **LIKE** *expression* [**ESCAPE** *expression*]
| *expression* [**NOT**] **IN** ({ *expression*
| *subquery*
| *value-expr1* , *value-expr2* [,*value-expr3*] … })
| **EXISTS** (*subquery*)
| **NOT** *condition*
| *search-condition* **AND** *search-condition*
| *search-condition* **OR** *search-condition*
| (*search-condition*)
| (*search-condition* , *estimate*)
| *search-condition* **IS** [**NOT**] { **TRUE** | **FALSE** | **UNKNOWN** }
| *trigger-operation*

参数

compare:
= | **>** | **<** | **>=** | **<=** | **<>** | **!=** | **!<** | **!>**

trigger-operation:
INSERTING | **DELETING**
| **UPDATING(** *column-name-string* **)** | **UPDATE(** *column-name* **)**

用法

任何地方。

权限

必须连接到数据库。

副作用

无。

另请参阅

" 表达式 " 第 17 页

说明

搜索条件用于从表中选择行的子集或者在控制语句 （如 IF 语句）中确定控制流。

在 SQL 中，每个条件的值都为 TRUE、 FALSE 或 UNKNOWN 中的一个。这叫做三值逻辑。如果所比较的值中某一个为 NULL，则比较结果为 UNKNOWN。有关三值逻辑中逻辑运算表，请参阅 "" 三值逻辑 " 第 33 页 " 一节。

当且仅当条件的结果为 TRUE 时，行才满足搜索条件。条件为 UNKNOWN 或 FALSE 时，行不满足搜索条件。有关 NULL 的更多信息，请参阅 ""NULL 值 " 第 55 页 "。

子查询构成了一类很重要的表达式，该表达式可用在很多搜索条件中。有关在搜索条件中使用子查询的信息，请参阅 "" 搜索条件中的子查询 " 第 27 页 "。

以下几节论述了不同类型的搜索条件。

搜索条件中的子查询

在 SQL 语句中可以使用列名称的任何地方 （包括在表达式中），都可以使用只返回一列并且零行或一行的子查询。

例如，只要子查询返回的行数不超过一行，表达式就可以在比较条件下与子查询进行比较 （请参阅 "" 比较运算符 " 第 12 页 "）。如果子查询（必须含有一列）返回一行，则该行的值将与表达式进行比较。如果子查询未返回行，其值为 NULL。

在 IN、 ANY 和 ALL 条件下可使用只返回一列及任意行数的子查询。在 EXISTS 条件下可使用返回任意数目的列和行的子查询。以下几节将就这些条件进行论述。

ALL 或 ANY 条件

ANY 条件的语法为

expression compare **ANY (** *subquery* **)**

其中 *compare* 是比较运算符。

例如，含有一个相等运算符的 ANY 条件，

expression = **ANY (** *subquery* **)**

如果 *expression* 等于子查询结果中的任何一个值，则为 TRUE ；如果表达式不为 NULL 而且不等于子查询中的任何一列，则为 FALSE。如果 *expression* 的值为 NULL，则 ANY 条件为 UNKNOWN，除非子查询的结果中没有行，在这种情况下，该条件始终为 FALSE。

可以使用关键字 **SOME** 代替 **ANY**。

ALL 条件的语法为

expression compare **ALL** (*subquery*)

其中 *compare* 是比较运算符。

兼容性

- ANY 和 ALL 子查询在 Adaptive Server Enterprise 和 Adaptive Server Anywhere 之间兼容。但只有 Adaptive Server Anywhere 才支持 SOME 作为 ANY 的同义词。

BETWEEN 条件

BETWEEN 条件的语法如下：

expr [**NOT**] **BETWEEN** *start-expr* **AND** *end-expr*

BETWEEN 条件的值可以是 TRUE、 FALSE 或 UNKNOWN。无关键字 NOT 时，如果 *expr* 介于 *start-expr* 和 *end-expr* 之间，条件的值为 TRUE。 NOT 关键字使条件的含义相反，但保留 UNKNOWN 不变。

BETWEEN 条件相当于两个不等式的组合：

[**NOT**] (*expr* >= *start-expr* **AND** *expr* <= *end-expr*)

兼容性

- BETWEEN 条件在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 之间兼容。

LIKE 条件

LIKE 条件的语法如下：

expr [**NOT**] **LIKE** *pattern* [**ESCAPE** *escape-expr*]

LIKE 条件的值可以是 TRUE、 FALSE 或 UNKNOWN。

无关键字 NOT 时，如果 *expression* 与 *pattern* 匹配，条件的值可以是 TRUE。如果 *expression* 或 *pattern* 的值为 NULL，则条件为 UNKNOWN。关键字 NOT 使条件的含义相反，但保留 UNKNOWN 不变。

此模式可包含任意数目的通配符。通配符为：

通配符	匹配项
_（下划线）	任意一个字符
%（百分号）	包含零个或多个字符的任意字符串
[]	指定范围或集合中的任何单个字符
[^]	*不*在指定范围或集合中的任何单个字符

所有其它字符都必须完全匹配。

例如，搜索条件

```
... name LIKE 'a%b_'
```

为 TRUE （对任何 name 以字母 a 开头并以字母 b 作为倒数第二个字符的行）。

如果指定一个 *escape-expr*，它的值必须为单个字符。字符可以位于百分号、下划线、左方括号或 *pattern* 中的其它转义字符之前，以防止特殊字符具有特殊的含义。以这种方式转义之后，百分号匹配百分号，下划线匹配下划线。

支持所有长度小于或等于 126 个字符的模式。但不支持长度大于 254 个字符的模式。支持某些长度介于 127 和 254 个字符之间的模式，具体情况取决于这些模式的内容。

搜索集合中的单个字符

通过在方括号中列出字符，可以指定要查找的集合中的单个字符。例如，用以下条件可找到字符串 *smith* 和 *smyth* ：

```
LIKE 'sm[iy]th'
```

搜索一定范围内的字符

通过在方括号内给出范围的两端 （用连字符分隔），可以指定要查找的字符范围。例如，用以下条件可找到字符串 *bough* 和 *rough*，但找不到 *tough*：

```
LIKE '[a-r]ough'
```

字符串范围 [a-z] 可以理解为 " 大于或等于 a 且小于或等于 z"，其中大于和小于操作在数据库归类中执行。有关归类中的字符串排序的信息，请参阅 "ASA 数据库管理指南 > 国际语言和字符集 "。

范围的下端必须位于上端之前。例如，包含表达式 [z-a] 的 LIKE 条件不返回行，因为没有字符匹配范围 [z-a]。

除非创建的数据库区分大小写，否则字符范围就不区分大小写。例如，用以下条件可找到字符串 *Bough* 、 *rough* 和 *TOUGH*：

```
LIKE '[a-z]ough'
```

如果创建的数据库区分大小写，则搜索条件也区分大小写。要在一个区分大小写的数据库中执行不区分大小写的搜索，必须包括大写和小写字符。例如，用以下条件可找到字符串 *Bough* 、 *rough* 和 *TOUGH*：

```
LIKE '[a-zA-Z][oO][uU][gG][hH]'
```

将范围搜索和组搜索组合起来

可以在方括号中将范围表达和集合表达组合起来。例如，用以下条件可找到字符串 *bough* 、 *rough* 和 *tough*：

```
... LIKE '[a-rt]ough'
```

中括号 *[a-mpqs-z]* 被解释为 " 正好一个字符，该字符或者介于 *a* 和 *m* 之间 （包括 *a* 和 *m*），或者是 *p*，或者是 *q*，或者介于 *s* 和 *z* 之间 （包括 *s* 和 *z*） "。

搜索不在某范围内的字符

脱字符 (^) 用于指定排除在搜索之外的字符范围。例如，用以下条件可找到字符串 *tough*，但找不到 *rough* 或 *bough*：

```
... LIKE '[^a-r]ough'
```

脱字符否定中括号内的所有其余内容。例如，中括号 *[^a-mpqs-z]* 被解释为 " 刚好一个字符，该字符不在 *a* 和 *m* （包括 *a* 和 *m* ）之间，不是 *p*，不是 *q*，也不在 *s* 和 *z* （包括 *s* 和 *z* ）之间 "。

范围和组的特殊情况

方括号中的任何单个字符都表示该字母。例如，*[a]* 只匹配字符 *a*，*[^]* 只匹配脱字符，*[%]* 只匹配百分号字符（在此环境中百分号字符不用作通配符），而 *[_]* 只匹配下划线字符。同样，*[[]* 只匹配字符 *[* 。

其它特殊情况如下：

- 表达式 *[a-]* 匹配字符 *a* 或 - 。
- 表达式 *[]* 永不匹配，并且始终不返回行。
- 表达式 *[* 或 *[abp-q* 是非法格式，会给出语法错误。
- 方括号内不能使用通配符。表达式 *[a%b]* 查找 *a* 、*%* 或 *b* 中的一个。
- 如果不作为中括号内的首字符，脱字符不能用于否定范围。表达式 *[a^b]* 查找 *a* 、^ 或 *b* 中的一个。

搜索带尾随空白的模式

在搜索含有尾随空白的模式时，Adaptive Server Anywhere 只将此模式与包含空白的值进行匹配，并不填充字符串空白。例如，搜索模式 '90 '、 '90[ ]' 和 '90_' 匹配值 '90 '，但并不匹配值 '90'，即使测试的值位于宽为三个或三个字符以上的 char 或 varchar 列中。

兼容性

- 只有 Adaptive Server Anywhere 支持 ESCAPE 子句。

IN 条件

IN 条件的语法如下：

expression [**NOT**] **IN** { **(** *subquery* **)** | **(** *expression* **)** | **(** *value-expr*, … **)** }

无 NOT 关键字时，如果 *expression* 等于任一列出的值，则 IN 条件为 TRUE；如果 *expression* 值为 NULL，则 IN 条件为 UNKNOWN；其它情况下为 FALSE。参数 *value-expr1*、*value-expr2* 和 *value-expr3* 是具有单值的表达式，单值可以是一个字符串、数字、日期或其它 SQL 数据类型。关键字 NOT 使条件的含义相反，但保留 UNKNOWN 不变。

兼容性

- IN 条件在 Adaptive Server Enterprise 和 Adaptive Server Anywhere 之间兼容。

EXISTS 条件

EXISTS 条件的语法如下：

EXISTS(*subquery* **)**

如果子查询结果至少包含一行，则 EXISTS 条件为 TRUE ；如果子查询结果一行也不包含，则条件为 FALSE。 EXISTS 条件不能为 UNKNOWN。

兼容性

- EXISTS 条件在 Adaptive Server Enterprise 和 Adaptive Server Anywhere 之间兼容。

IS NULL 条件

IS NULL 条件的语法如下：

expression **IS** [**NOT**] **NULL**

无 NOT 关键字时，如果表达式值为 NULL，则 IS NULL 条件为 TRUE，其它情况下为 FALSE。 NOT 关键字使条件的含义相反。

兼容性

- IS NULL 条件在 Adaptive Server Enterprise 和 Adaptive Server Anywhere 之间兼容。

真值条件

真值条件的语法如下：

IS [**NOT**] *truth-value*

无 NOT 关键字时，如果 *condition* 的值为提供的 *truth-value* （必须是 TRUE、 FALSE 或 UNKNOWN 中的一个），则条件为 TRUE。否则，条件值为 FALSE。关键字 NOT 使条件的含义相反，但保留 UNKNOWN 不变。

兼容性

- 供应商扩展。 Adaptive Server Anywhere 不支持真值条件。

触发器操作条件

触发器操作条件的语法如下：

trigger-operation:
INSERTING | **DELETING**
| **UPDATING(** *column-name-string* **)** | **UPDATE(** *column-name* **)**

触发条件只能用在触发器中，根据激发触发器的操作种类来完成操作。

UPDATING 的参数是一个带引号的字符串 （例如， UPDATING('mycolumn')）。 UPDATE 的参数是一个标识符 （例如，UPDATE(mycolumn)）。两种版本可以互操作，包括这两种版本是为了与其他供应商的 DBMS 的 SQL 方言兼容。

如果您提供一个 UPDATING 或 UPDATE 函数，则还必须在 CREATE TRIGGER 语句中提供一个 REFERENCING 子句以避免出现语法错误。

示例

下面的触发器显示一条消息，说明哪个操作导致激发触发器。

```
CREATE TRIGGER tr BEFORE INSERT, UPDATE, DELETE
ON sample_table
REFERENCING OLD AS t1old
FOR EACH ROW
BEGIN
    DECLARE msg varchar(255);
    SET msg = 'This trigger was fired by an ';
    IF INSERTING THEN
        SET msg = msg || 'insert'
    ELSEIF DELETING THEN
        set msg = msg || 'delete'
    ELSEIF UPDATING THEN
        set msg = msg || 'update'
    END IF;
    MESSAGE msg TO CLIENT
END
```

三值逻辑

下表显示了 SQL 中的 AND、 OR、 NOT 和 IS 逻辑运算符如何以三值逻辑的方式操作。

AND 运算符

AND	TRUE	FALSE	UNKNOWN
TRUE	TRUE	FALSE	UNKNOWN
FALSE	FALSE	FALSE	FALSE
UNKNOWN	UNKNOWN	FALSE	UNKNOWN

OR 运算符

OR	TRUE	FALSE	UNKNOWN
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	UNKNOWN
UNKNOWN	TRUE	UNKNOWN	UNKNOWN

NOT 运算符

TRUE	FALSE	UNKNOWN
FALSE	TRUE	UNKNOWN

IS 运算符

IS	TRUE	FALSE	UNKNOWN
TRUE	TRUE	FALSE	FALSE
FALSE	FALSE	TRUE	FALSE
UNKNOWN	FALSE	FALSE	TRUE

显式选择性估计

Adaptive Server Anywhere 利用统计信息确定执行各语句的最有效策略。Adaptive Server Anywhere 自动收集和更新这些统计信息。这些统计信息永久存储在数据库的系统表 SYSCOLSTAT 中。处理一个语句时所收集的统计信息对于搜索后续语句的有效执行方式很有用。

有时，统计信息可能变得不准确，或者相关的统计信息不可用。最有可能发生这种情况的时候是：添加、更新或删除了大量数据后，只执行了很少的查询。

在这种情况下，您可能需要执行 CREATE STATISTICS 或 DROP STATISTICS。

然而，在极少数情况下，这些措施可能无效。这种情况下，有时可以通过提供显式选择性估计来提高性能。

对一个潜在执行计划中的各表，优化程序必须估计那些将成为结果集的一部分的行的数量。如果您知道某条件的成功率与优化程序估计的不一样，您可以在搜索条件中显式地提供用户的估计。

估计是一个百分比。它可以是正整数，也可以包含小数值。

注意：尽量避免在不断使用的语句中提供显式估计。如果数据改变，显式估计可能会变得不准确并强制优化程序选择低效率的计划。

可以通过将数据库选项 USER_ESTIMATES 设置为 OFF 来禁用用户估计。 USER_ESTIMATES 的缺省值为 OVERRIDE-MAGIC，这表示只有当优化程序对条件使用 MAGIC （缺省）选择性值时才可以使用用户提供的选择性估计。当优化程序不能准确判定一个谓词的选择性时，它的最后一个手段就是利用 MAGIC 值。

有关禁用用户定义的选择性估计的更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > USER_ESTIMATES 选项 [database]"。

有关统计的更多信息，请参阅 "ASA SQL 用户指南 > 查询优化与执行 > 优化程序估计值 "。

示例

- 以下查询估计 1% 的 **ship_date** 值会晚于 2001/06/30：

```
SELECT  ship_date
  FROM  sales_order_items
WHERE ( ship_date > '2001/06/30', 1 )
ORDER BY ship_date DESC
```

- 以下查询估计 0.5% 的行会满足条件：

```
SELECT *
  FROM customer c, sales_order o
WHERE (c.id = o.cust_id, 0.5)
```

小数值为连接 （特别是为大表）提供了更准确的用户估计。

兼容性

- Adaptive Server Enterprise 不支持显式估计。

特殊值

特殊值可用在表达式中，也可在创建表时用作列缺省值。

CURRENT DATABASE 特殊值

作用　CURRENT DATABASE 返回当前数据库的名称。

数据类型　STRING

另请参阅　" 表达式 " 第 17 页

CURRENT DATE 特殊值

作用　CURRENT DATE 返回当前年月日。

数据类型　DATE

另请参阅　" 表达式 " 第 17 页

"TIME 数据类型 [日期和时间]" 第 81 页

CURRENT PUBLISHER 特殊值

作用　CURRENT PUBLISHER 返回一个字符串，其中包含 SQL Remote 复制数据库的发布者用户 ID。

数据类型　STRING

CURRENT PUBLISHER 可在具有字符数据类型的列中用作缺省值。

另请参阅　" 表达式 " 第 17 页

"SQL Remote 用户指南 > 适用于 Adaptive Server Anywhere 的 SQL Remote 设计 "

CURRENT TIME 特殊值

作用 返回当前小时、分钟、秒和秒的小数值。

数据类型 TIME

说明 秒的小数值存储到 6 个小数位。当前时间的准确性受系统时钟准确性的限制。

另请参阅 " 表达式 " 第 17 页

"TIME 数据类型 [日期和时间]" 第 81 页

CURRENT TIMESTAMP 特殊值

作用 组合 CURRENT DATE 和 CURRENT TIME，以构成包含年月日、小时、分钟、秒和秒的小数值的 TIMESTAMP 值。与 CURRENT TIME 一样，秒的小数值的准确性受系统时钟的限制。

秒的小数值存储到 3 个小数位。其准确性受系统时钟准确性的限制。

数据类型 TIMESTAMP

另请参阅 " 表达式 " 第 17 页

"TIMESTAMP 数据类型 [日期和时间]" 第 82 页

CURRENT USER 特殊值

作用 CURRENT USER 返回一个包含当前连接的用户 ID 的字符串。

数据类型 STRING

CURRENT USER 可在具有字符数据类型的列中用作缺省值。

说明 在 UPDATE 时，不更改具有 CURRENT USER 缺省值的列。

另请参阅 " 表达式 " 第 17 页

CURRENT UTC TIMESTAMP 特殊值

作用 组合根据服务器时区调整值调整后的 CURRENT DATE 和 CURRENT TIME，以构成包含年月日、小时、分钟、秒和秒的小数值的协调通用时间 (UTC) TIMESTAMP 值。此功能允许以一致的时间参照输入数据，而不考虑是在哪个时区输入数据的。

数据类型 TIMESTAMP

另请参阅 "TIMESTAMP 数据类型 [日期和时间]" 第 82 页

"UTC TIMESTAMP 特殊值 " 第 41 页

"CURRENT TIMESTAMP 特殊值 " 第 38 页

"ASA 数据库管理指南 > 数据库选项 > TRUNCATE_TIMESTAMP_VALUES 选项 [database]"

LAST USER 特殊值

作用 返回上次修改行的用户的名称。

数据类型 String。

LAST USER 可在具有字符数据类型的列中用作缺省值。

说明 在 INSERT 时，此常量与 CURRENT USER 的效果相同。在 UPDATE 时，如未显式修改具有 LAST USER 缺省值的列，该列会变为当前用户的名称。

与 DEFAULT TIMESTAMP 组合时， LAST USER 缺省值可用于记录（在单独的列中）用户和上次更改行的日期和时间。

另请参阅 "CURRENT USER 特殊值 " 第 38 页

"CURRENT TIMESTAMP 特殊值 " 第 38 页

"CREATE TABLE 语句 " 第 395 页

SQLCODE 特殊值

作用 返回当前 SQLCODE 值。

数据类型 String。

说明 在每个语句后都设置了 SQLCODE 值。您可以检查 SQLCODE，以查看语句是否成功。

另请参阅 " 表达式 " 第 17 页

"ASA 错误消息 > 数据库错误消息 "。

SQLSTATE 特殊值

作用 返回当前 SQLSTATE 值

数据类型 STRING

说明 在每个语句后都设置了 SQLSTATE 值。您可以检查 SQLSTATE，以查看语句是否成功。

另请参阅 " 表达式 " 第 17 页

"ASA 错误消息 > 数据库错误消息 "

TIMESTAMP 特殊值

作用 TIMESTAMP 表示上次是在何时修改表中各行的。当用 DEFAULT TIMESTAMP 声明列时，会提供一个缺省的插入值，每当更新行时，该值都用当前日期和时间更新。

数据类型 TIMESTAMP

说明 用 DEFAULT TIMESTAMP 声明的列包含唯一值，以便应用程序能够检测到几乎同时发生的行更新。如果当前时间戳的值与上一个值相同，它将增长 DEFAULT_TIMESTAMP_INCREMENT 选项的值。

在 Adaptive Server Anywhere 中，您可以根据 DEFAULT_TIMESTAMP_INCREMENT 选项自动截断时间戳的值。这对于与其它数据库软件 （它们记录的时间戳值的精确性要差一些）兼容很有用。

全局变量 @@dbts 返回一个 TIMESTAMP 值，该值表示上次使用 DEFAULT TIMESTAMP 为列生成的值。

另请参阅

"TIMESTAMP 数据类型 [日期和时间]" 第 82 页

"CURRENT UTC TIMESTAMP 特殊值 " 第 39 页

"ASA 数据库管理指南 > 数据库选项 > DEFAULT_TIMESTAMP_INCREMENT 选项 [database]"

"ASA 数据库管理指南 > 数据库选项 > TRUNCATE_TIMESTAMP_VALUES 选项 [database]"

USER 特殊值

作用

USER 返回包含当前连接的用户 ID 的字符串。

数据类型

STRING

USER 可在具有字符数据类型的列中用作缺省值。

说明

在 UPDATE 时，不更改具有 USER 缺省值的列。

另请参阅

" 表达式 " 第 17 页

"CURRENT USER 特殊值 " 第 38 页

"LAST USER 特殊值 " 第 39 页

UTC TIMESTAMP 特殊值

作用

UTC TIMESTAMP 表示上次修改表内各行时的协调通用时间 (UTC)。

特殊值

数据类型 TIMESTAMP

另请参阅 "TIMESTAMP 数据类型 [日期和时间]" 第 82 页

"CURRENT UTC TIMESTAMP 特殊值 " 第 39 页

"TIMESTAMP 特殊值 " 第 40 页

"ASA 数据库管理指南 > 数据库选项 > DEFAULT_TIMESTAMP_INCREMENT 选项 [database]"

"ASA 数据库管理指南 > 数据库选项 > TRUNCATE_TIMESTAMP_VALUES 选项 [database]"

变量

Adaptive Server Anywhere 支持三个级别的变量：

- **局部变量**　这些是使用 DECLARE 语句在过程或批处理中的复合语句内定义的变量。它们只存在于复合语句内。
- **连接级变量**　这些是使用 CREATE VARIABLE 语句定义的变量。它们属于当前连接，当断开现有连接或使用 DROP VARIABLE 语句时，它们就会消失。
- **全局变量**　这些是系统提供的具有系统提供值的变量。所有全局变量的名称都以两个 @ 符号开头。例如，全局变量 **@@version** 所具有的值是数据库服务器的当前版本号。用户不能定义全局变量。

本地和连接级变量由用户声明，并且可在 SQL 语句的过程或批处理中用于保存信息。而全局变量是系统提供的变量，这些变量可以提供系统提供的值。

另请参阅

"TIMESTAMP 数据类型 [日期和时间]" 第 82 页

"CREATE VARIABLE 语句 " 第 417 页

局部变量

局部变量是使用 DECLARE 语句声明的，它们只能用在复合语句（即用关键字 BEGIN 和 END 括起来的语句）内。这类变量最初被设置为 NULL。局部变量的值可以用 SET 语句进行设置，或用带有 INTO 子句的 SELECT 语句指派。

DECLARE 语句的语法如下：

```
DECLARE variable-name data-type
```

只要从复合语句内调用过程，局部变量就可以作为参数传递给该过程。

示例

- 下面的批处理语句说明了如何使用局部变量。

```
BEGIN
  DECLARE local_var INT;
  SET local_var = 10;
  MESSAGE 'local_var = ', local_var TO CLIENT;
END
```

如果从 Interactive SQL 中运行此批处理语句， Interactive SQL 的 " 消息 " 窗格中会给出消息 local_var = 10。

- 变量 *local_var* 在声明此变量的复合语句外部不存在。下面的批处理语句无效，会给出错误未找到列。

```
-- This batch is invalid.
BEGIN
  DECLARE local_var INT;
  SET local_var = 10;
END;
MESSAGE 'local_var = ', local_var TO CLIENT;
```

- 下例说明了如何用带有 INTO 子句的 SELECT 语句设置局部变量的值：

```
BEGIN
  DECLARE local_var INT;
  SELECT 10 INTO local_var;
  MESSAGE 'local_var = ', local_var TO CLIENT;
END
```

如果从 Interactive SQL 中运行此批处理语句，服务器窗口中会给出消息 local_var = 10。

兼容性

- **名称** Adaptive Server Enterprise 和 Adaptive Server Anywhere 都支持局部变量。在 Adaptive Server Enterprise 中，所有变量都必须用一个 @ 符号作为前缀。而在 Adaptive Server Anywhere 中， @ 前缀是可选的。要写出兼容的 SQL，在所有变量前都需加前缀 @。

- **作用域** 局部变量的作用域在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中不同。 Adaptive Server Anywhere 支持在批处理语句内使用 DECLARE 语句声明局部变量。然而，如果在复合语句内执行 DECLARE 语句，作用域仅限于该复合语句。

- **声明**　在 Adaptive Server Anywhere 中，每个 DECLARE 语句只能声明一个变量。而在 Adaptive Server Enterprise 中，一个语句可以声明多个变量。

有关批处理和局部变量作用域的更多信息，请参阅 "ASA SQL 用户指南 > Transact-SQL 兼容性 > Transact-SQL 过程中的变量 "。

连接级变量

连接级变量是用 CREATE VARIABLE 语句声明的。连接级变量可作为参数传递给过程。

CREATE VARIABLE 语句的语法如下：

```
CREATE VARIABLE variable-name data-type
```

创建的变量最初设置为 NULL。连接级变量的值可用与局部变量相同的方法设置：使用 SET 语句或用带有 INTO 子句的 SELECT 语句。

连接级变量一直存在到连接终止，或者直到用 DROP VARIABLE 语句显式删除变量。以下语句会删除变量 **con_var**：

```
DROP VARIABLE con_var
```

示例

- 下面的 SQL 批处理语句说明如何使用连接级变量。

  ```
  CREATE VARIABLE con_var INT;
  SET con_var = 10;
  MESSAGE 'con_var = ', con_var TO CLIENT;
  ```

  如果从 Interactive SQL 中运行此批处理语句，服务器窗口中会给出消息 con_var = 10。

兼容性

- Adaptive Server Enterprise 不支持连接级变量。

全局变量

全局变量的值由数据库服务器设置。例如，全局变量 **@@version** 所具有的值是数据库服务器的当前版本号。

全局变量的名称前有两个 @ 符号，这是其与本地和连接级变量的不同之处。例如， **@@error** 和 **@@rowcount** 都是全局变量。用户不能创建全局变量，也不能直接更新全局变量的值。

有些全局变量 （如 **@@identity**）保存连接特定的信息，因此具有连接特定的值。有些变量 （如 **@@connections**）则具有所有连接共有的值。

全局变量和特殊常量

特殊常量 （例如 CURRENT DATE、 CURRENT TIME、 USER 和 SQLSTATE）类似于全局变量。

以下语句检索版本全局变量的值。

```
SELECT @@version
```

在过程和触发器中，全局变量可以选入变量列表。以下过程用参数 *ver* 返回服务器版本号。

```
CREATE PROCEDURE VersionProc (OUT ver
            VARCHAR(100))
BEGIN
   SELECT @@version
   INTO ver;
END
```

在嵌入式 SQL 中，全局变量可以选入主机变量列表。

全局变量列表

下表列出了 Adaptive Server Anywhere 中可用的全局变量。

变量名	含义
@@*dbts*	一个 TIMESTAMP 类型的值，表示上次生成的值，用于用 DEFAULT TIMESTAMP 定义的所有列。
@@*error*	常用于检查最近执行的语句的错误状态 （成功或失败）。如果上一事务成功，则包含 0，否则包含上次系统生成的错误号。如果发生错误，像 "if @@error != 0 return" 这样的语句会导致退出。每个 SQL 语句都会重置 @@error，因此在成功状态不确定的语句后必须立即执行状态检查。

变量名	含义
@@fetch_status	包含来自上一 fetch 语句的状态信息。 @@fetch_status 可包含以下值： • 0 fetch 语句已成功完成。 • -1 fetch 语句引起错误。 • -2 结果集中没有其它数据。 此功能与 @@sqlstatus 相同，只不过它返回的值不同。这是为了与 Microsoft SQL Server 兼容。
@@identity	上次用 INSERT 或 SELECT INTO 语句插入到任何 IDENTITY 或 DEFAULT AUTOINCREMENT 列中的值。 有关说明，请参阅 "@@identity 全局变量 " 第 51 页。
@@isolation	当前隔离级别。 @@isolation 采用活动级别的值。
@@procid	当前执行的过程的存储过程 ID。
@@rowcount	受上一语句影响的行数。应在该语句后立即检查 @@rowcount 的值。 插入、更新和删除会将 @@rowcount 设置为受影响的行数。 在带有游标的情况下， @@rowcount 表示从游标结果集返回给客户端的累计行数 （截止到上次读取请求）。与在 Adaptive Server Enterprise 中不同，任何不影响行的语句 （如 IF 语句）都不将 @@rowcount 重置为零。
@@servername	当前数据库服务器的名称。
@@sqlstatus	包含来自上一 fetch 语句的状态信息。 @@sqlstatus 可包含以下值： • 0 fetch 语句已成功完成。 • 1 fetch 语句引起错误。 • 2 结果集中没有其它数据。
@@version	Adaptive Server Anywhere 当前版本的版本号。

兼容性

以下列表包含 Adaptive Server Anywhere 中支持的所有 Adaptive Server Enterprise 全局变量。 Adaptive Server Anywhere 不支持的 Adaptive Server Enterprise 全局变量没有在表中列出。与上表相反，此列表包含所有返回值的全局变量，包括值固定为 NULL、 1、 -1 或 0 以及可能无含义的全局变量。

全局变量	返回值
@@char_convert	返回 0。
@@client_csname	在 Adaptive Server Enterprise 中，返回客户端的字符集名称。如果客户端字符集从未初始化，则设置为 NULL ；否则包含最近使用的字符集的名称。在 Adaptive Server Anywhere 中返回 NULL。
@@client_csid	在 Adaptive Server Enterprise 中，返回客户端的字符集 ID。如果客户端字符集从未初始化，则设置为 -1 ；否则包含 syscharsets 中的最近使用的客户端字符集 ID。在 Adaptive Server Anywhere 中返回 -1。
@@connections	自服务器上次启动后的登录数
@@cpu_busy	在 Adaptive Server Enterprise 中，返回自 Adaptive Server Enterprise 上次启动后 CPU 花在 Adaptive Server Enterprise 工作上的时间 （以时钟周期为单位）。在 Adaptive Server Anywhere 中返回 0。
@@error	常用于检查最近执行的语句的错误状态 （成功或失败）。如果上一事务成功，则包含 0，否则包含上次系统生成的错误号。如果发生错误，像 `"if @@error != 0 return"` 这样的语句会导致退出。每个语句（包括 PRINT 语句或 IF 测试）都重置 @@error，因此在成功状态不确定的语句后必须立即执行状态检查。
@@identity	上次用 INSERT 或 SELECT INTO 语句插入到 IDENTITY 列中的值。 有关说明，请参阅 ""@@identity 全局变量 " 第 51 页 "。
@@idle	在 Adaptive Server Enterprise 中，返回自 Adaptive Server Enterprise 上次启动后的空闲时间 （以时钟周期为单位）。在 Adaptive Server Anywhere 中返回 0。

全局变量	返回值
@@io_busy	在 Adaptive Server Enterprise 中，返回自 Adaptive Server Enterprise 上次启动后花在输入和输出操作上的时间 （以时钟周期为单位）。在 Adaptive Server Anywhere 中返回 0。
@@isolation	当前连接的隔离级别。在 Adaptive Server Enterprise 中， @@isolation 采用活动级别的值。
@@langid	在 Adaptive Server Enterprise 中，定义当前正在使用的语言的本地语言 ID。在 Adaptive Server Anywhere 中返回 0。
@@language	在 Adaptive Server Enterprise 中，定义当前正在使用的语言的名称。在 Adaptive Server Anywhere 中返回 "English"。
@@maxcharlen	在 Adaptive Server Enterprise 中，返回 Adaptive Server Enterprise 的缺省字符集中的最大字符长度 （以字节为单位）。在 Adaptive Server Anywhere 中返回 1。
@@max_ connections	对于个人服务器，返回可同时连接到服务器的最大数目 10。 对于网络服务器，返回活动客户端 （不是数据库连接，因为各客户端可支持多个连接）的最大数目。 对于 Adaptive Server Enterprise，返回与服务器的最大连接数。
@@ncharsize	在 Adaptive Server Enterprise 中，返回本国字符的平均长度 （以字节为单位）。在 Adaptive Server Anywhere 中返回 1。
@@nestlevel	在 Adaptive Server Enterprise 中，返回当前执行的嵌套级别 （最初为 0）。每次一个存储过程或触发器调用另一个存储过程或触发器，嵌套级别都会递增。在 Adaptive Server Anywhere 中返回 -1。
@@pack_received	在 Adaptive Server Enterprise 中，返回 Adaptive Server Enterprise 自上次启动后读取的输入数据包数。在 Adaptive Server Anywhere 中返回 0。

全局变量	返回值
@@pack_sent	在 Adaptive Server Enterprise 中，返回 Adaptive Server Enterprise 自上次启动后写入的输出数据包数。在 Adaptive Server Anywhere 中返回 0。
@@packet_errors	在 Adaptive Server Enterprise 中，返回在 Adaptive Server Enterprise 发送和接收数据包过程中发生的错误数。在 Adaptive Server Anywhere 中返回 0。
@@procid	当前执行的过程的存储过程 ID。
@@rowcount	返回受上一命令影响的行数。在 Adaptive Server Enterprise 中，任何不返回行的命令 （如 *IF* 语句）都将 @@rowcount 重置为零；而在 Adaptive Server Anywhere 中，这类语句不重置 @@rowcount。在带有游标的情况下， @@rowcount 表示从游标结果集返回给客户端的累计行数 （截止到上次读取请求）。
@@servername	本地 Adaptive Server Enterprise 或 Adaptive Server Anywhere 服务器的名称。
@@spid	在 Adaptive Server Enterprise 中，返回当前进程的服务器进程 ID 号。在 Adaptive Server Anywhere 中，返回当前连接的连接句柄。该值与 sa_conn_info 过程显示的值相同。
@@sqlstatus	包含上次读取语句所产生的状态信息。 @@sqlstatus 可以包含下列值 • 0 读取语句成功完成。 • 1 读取语句导致错误。 • 2 结果集中没有其它数据。
@@textsize	返回 SET TEXTSIZE 选项的当前值，它指定了用选择语句返回的文本和图形数据的最大长度 （以字节为单位）。缺省设置为 32765，这是用 READTEXT 可返回的最大字节字符串。也可用 SET 语句设置该值。
@@thresh_hysteresis	在 Adaptive Server Enterprise 中，返回激活一个阈值所需的可用空间变化。在 Adaptive Server Anywhere 中返回 0。

全局变量	返回值
@@timeticks	在 Adaptive Server Enterprise 中，返回每个时钟周期的微秒数。每个时钟周期的时间量与机器相关。在 Adaptive Server Anywhere 中返回 0。
@@total_errors	在 Adaptive Server Enterprise 中，返回在 Adaptive Server Enterprise 读取或写入过程中发生的错误数。在 Adaptive Server Anywhere 中返回 0。
@@total_read	在 Adaptive Server Enterprise 中，返回 Adaptive Server Enterprise 自上次启动后读取的磁盘数。在 Adaptive Server Anywhere 中返回 0。
@@total_write	在 Adaptive Server Enterprise 中，返回 Adaptive Server Enterprise 自上次启动后写入的磁盘数。在 Adaptive Server Anywhere 中返回 0。
@@tranchained	Transact-SQL 程序的当前事务模式。 @@tranchained 返回 0 (unchained) 或 1 (chained)。
@@trancount	事务的嵌套级别。批处理中的各 BEGIN TRANSACTION 都增加事务计数。
@@transtate	在 Adaptive Server Enterprise 中，返回语句执行后事务的当前状态。在 Adaptive Server Anywhere 中返回 -1。
@@version	有关 Adaptive Server Enterprise 或 Adaptive Server Anywhere 当前版本的信息。

@@identity 全局变量

@@identity 变量保存插入到 IDENTITY 列或 DEFAULT AUTOINCREMENT 列中的最新值，如果最新插入是在不含这些列的表中进行的，则为零。

每次向表中插入一行时， @@identity 的值都会重置。如果一个语句插入多行，则 @@identity 反映上次插入的行的 IDENTITY 值。如果受影响的表不包含 IDENTITY 列，则 @@identity 设置为 0。

INSERT 或 SELECT INTO 语句的失败，或者包含 INSERT 或 SELECT INTO 语句的事务的回退，都不会影响 @@identity 的值。 @@identity 保留上次插入到 IDENTITY 列中的值，即使插入该值的语句未能提交。

@@identity 和触发器

当插入导致了参照完整性操作或者触发了触发器时， *@@identity* 的行为与堆栈一样。例如，如果一个向表 *T1* （具有标识列或自动增量列）的插入触发了向表 *T2* （也具有标识列或自动增量列）中插入一行的触发器，那么返回给执行此插入的应用程序或过程的值就是插入到 *T1* 的值。在触发器内， *@@identity* 在插入到 *T2* 前具有 *T1* 值，插入后具有 *T2* 值。如果触发器需要访问这两个值，它可以将其复制到局部变量。

注释

注释用于在 SQL 语句或语句块中附加说明性文字。数据库服务器不执行注释。

Adaptive Server Anywhere 中有若干种注释指示符。

- **-- （双连字符）** 数据库服务器忽略行上的任何其余字符。这是 SQL/92 注释指示符。
- **// （双斜线）** 双斜线与双连字符的含义相同。
- **/* … */ （斜线加星号）** 忽略两个注释标记间的任何字符。两个注释标记可以在同一行上或者在不同的行上。可以嵌套以此风格指示的注释。这种注释风格也称为 "C 样式 " 注释。
- **% （百分号）** 如果 PERCENT_AS_COMMENT 选项设置为 ON，则百分号与双连字符的含义相同。建议不要将 % 用作注释指示符。

兼容性

- 双连字符和斜线加星号注释风格与 Adaptive Server Enterprise 兼容。

示例

- 下例说明了如何使用双划线注释：

```
CREATE FUNCTION fullname (firstname CHAR(30),
     lastname CHAR(30))
RETURNS CHAR(61)
-- fullname concatenates the firstname and
lastname
-- arguments with a single space between.
BEGIN
  DECLARE name CHAR(61);
  SET name = firstname || ' ' || lastname;
  RETURN ( name );
END
```

- 下例说明如何使用 C 样式注释：

```
/*
  Lists the names and employee IDs of employees
  who work in the sales department.
*/
CREATE VIEW SalesEmployee AS
SELECT emp_id, emp_lname, emp_fname
FROM "DBA".employee
WHERE dept_id = 200
```

NULL 值

作用　指定未知或不适用的值。

语法　**NULL**

用法　任何地方。

权限　必须连接到数据库。

副作用　无。

另请参阅　" 表达式 " 第 17 页

" 搜索条件 " 第 26 页

说明　NULL 值是一个特殊值，它不同于任何数据类型的任何有效值。然而，NULL 值在任何数据类型中都是合法值。 NULL 值用于表示缺少或不适用的信息。请注意，下面是使用 NULL 的两种截然不同的情况：

情况	说明
缺少	字段具有值，但该值未知。
不适用	字段不适用于特定的行。

SQL 允许在创建列时带有 NOT NULL 限制。这意味着那些列不能包含 NULL 值。

NULL 值将三值逻辑的概念引入 SQL。 NULL 值通过任何比较运算符与任何值 （包括 NULL 值）进行比较时都是 "UNKNOWN"。返回 TRUE 的唯一搜索条件是 IS NULL 谓词。在 SQL 中，只有 WHERE 子句中的搜索条件的值为 TRUE 时才选择行；不选择其值为 UNKNOWN 或 FALSE 的行。

IS [NOT] *truth-value* 子句 （其中 *truth-value* 是 TRUE、 FALSE 或 UNKNOWN 中的一个）可用于选择涉及 NULL 值的行。有关该子句的说明，请参阅 "" 搜索条件 " 第 26 页 "。

以下示例中，列 **Salary** 包含 NULL。

条件	真值	是否选择？
Salary = NULL	UNKNOWN	NO
Salary <> NULL	UNKNOWN	NO
NOT (Salary = NULL)	UNKNOWN	NO
NOT (Salary <> NULL)	UNKNOWN	NO
Salary = 1000	UNKNOWN	NO
Salary IS NULL	TRUE	YES
Salary IS NOT NULL	FALSE	NO
Salary = *expression* IS UNKNOWN	TRUE	YES

同样的规则也适用于比较两个不同表中的列。因此，将两个表连接到一起不会选择其所比较的任何一列含有 NULL 值的行。

用于数字表达式时， NULL 还有一个有趣的属性。*任何*涉及 NULL 值的数字表达式的结果都为 NULL。这意味着，如果将 NULL 加到一个数字上，结果为 NULL 而不是数字。如果希望 NULL 被视作 0，必须使用 **ISNULL(** *expression*, **0)** 函数 （请参阅 ""SQL 函数 " 第 105 页 "）。

公式化 SQL 查询时的很多常见错误是由 NULL 的行为引起的。必须小心避免这些问题。有关组合搜索条件时三值逻辑效果的说明，请参阅 "" 搜索条件 " 第 26 页 "。

标准和兼容性

- **SQL/92** 初级功能。
- **Sybase** 在有些环境中， Adaptive Server Enterprise 将 NULL 按值处理，而 Adaptive Server Anywhere 不是这样。例如，在 Adaptive Server Anywhere 中，使用以下 WHERE 子句的查询结果中并不包括列 *c1* 中为 NULL 的行，因为条件的值为 UNKNOWN：

  ```
  WHERE NOT( C1 = NULL )
  ```

  而在 Adaptive Server Enterprise 中，条件的值为 TRUE，因此返回这些行。为了兼容，应该使用 IS NULL 而不是比较运算符。

Adaptive Server Anywhere 中的唯一索引可以包含保存 NULL 和其它相同值的行。而 Adaptive Server Enterprise 中的唯一索引不允许这样的条目。

如果使用 jConnect，则 TDS_EMPTY_STRING_IS_NULL 选项将控制返回的空字符串是作为 NULL 字符串，还是作为包含一个空白字符的字符串。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > TDS_EMPTY_STRING_IS_NULL 选项 [database]"。

示例

- 以下 INSERT 语句将一个 NULL 值插入到表 *Borrowed_book* 的 *date_returned* 列中。

```
INSERT
INTO Borrowed_book
( date_borrowed, date_returned, book )
VALUES ( CURRENT DATE, NULL, '1234' )
```

第 2 章

SQL 数据类型

关于本章 本章将介绍 Adaptive Server Anywhere 支持的数据类型。

字符数据类型

作用

存储字母、数字和符号的字符串。

说明

Adaptive Server Anywhere 将 CHAR、 VARCHAR 和 LONG VARCHAR 列全部视为同一类型。长度不超过 254 个字符的值存储为短字符串，并且前面有一个长度字节。任何长度超过 255 个字节的值都被视为长字符串。第 255 个字节之后的字符与包含长字符串值的行分开存储。

有几种函数 （请参阅 "SQL 函数 " 第 105 页）将忽略任何字符串中第 255 个字符之后的部分。它们是 **soundex**、 **similar** 和所有日期函数。另外，任何涉及长字符串到数字转换的算法只对前 255 个字符有效。碰到上述限制的可能性非常小。

所有其他函数和所有其他运算符都对长字符串的全部长度起作用。

字符集和代码页

字符数据按照从应用程序传递而来的准确的二进制表示形式被放在数据库中。这通常意味着字符数据是以当前**代码页**的二进制表示形式存储在数据库中的。代码页是 IBM 兼容的个人计算机所使用的字符集表示形式。在操作系统的文档中可以找到有关代码页的文档。

对于前 128 个字符，大多数代码页都是相同的。如果您使用代码页的高位特殊字符 （重音国际语言字符），则必须当心您的数据库。具体说来，如果您将数据库复制到使用另一种代码页的计算机上，则从数据库中检索的那些特殊字符将采用原来的代码页表示形式。对于新代码页，它们在屏幕上将显示为错误字符。

如果您有两个客户端，它们使用同一多用户服务器，但运行不同的代码页，也会出现此问题。一个客户端插入或更新的数据在另一个客户端看来可能是不正确的。

如果跨平台使用数据库，也会出现此问题。 PowerBuilder 和许多其他的 Windows 应用程序以标准 ANSI 字符集的形式将数据插入到数据库中。如果非 Windows 应用程序试图使用这些数据，它们将不能正确地显示或更新扩展字符。

此问题十分复杂。如果您的某些应用程序使用代码页的高位扩展字符，请确保使用数据库的所有客户端和所有计算机都使用相同或兼容的代码页。

注意

不允许小于 1 的数据类型长度。

兼容性

- Adaptive Server Enterprise 不支持 CHAR 的替换形式 CHARACTER (n)。
- Adaptive Server Anywhere 不支持 Adaptive Server Enterprise 提供的 NCHAR 和 NVARCHAR 数据类型。

CHAR 数据类型 [字符]

作用

最大长度为 *max-length* 字节的字符数据。

语法

{ **CHAR** | **CHARACTER** } [**(** *max-length* **)**]

用法

max-length 的缺省值为 1。

对于最大长度为 254 字节的字符串，存储要求是将字符串中的字节数再加上一个字节。对于更长的字符串，需要更多开销。

多字节字符的字符串可以保存为 CHAR 数据类型，但 *max-length* 是以字节为单位的，而不是以字符为单位。

参数

max-length　以字节为单位的字符串的最大长度。允许的最大大小为 32767。

标准和兼容性

- **SQL/92**　与 SQL/92 兼容。
- **Sybase**　与 Adaptive Server Enterprise 兼容。在 Adaptive Server Enterprise 中， CHAR 数据类型的存储要求始终为 *max-length*。Adaptive Server Enterprise 的最大 *max-length* 为 255。
- **其它数据库系统**　在许多其它数据库管理系统中，与 Adaptive Server Anywhere 不同， CHAR 数据类型会导致用空格填充整个字符串长度。这意味着无论实际字符串的长度是多少， CHAR 数据类型都需要 *max-length* 字节的存储。

另请参阅

"CHARACTER VARYING 数据类型 [字符]" 第 62 页

"LONG VARCHAR 数据类型 [字符]" 第 62 页

CHARACTER VARYING 数据类型 [字符]

作用 与 CHAR 相同。

语法 { **VARCHAR** | **CHARACTER VARYING** } [**(** *max-length* **)**]

用法 *max-length* 的缺省值为 1。

对于最大长度为 254 字节的字符串，存储要求是字符串中的字节数再加上一个字节。对于更长的字符串，需要更多开销。

多字节字符的字符串可以保存为 CHAR 数据类型，但要注意 *max-length* 是以字节为单位的，而不是以字符为单位。

参数 **max-length** 字符的最大长度，以字节为单位。允许的最大大小为 32767。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **Sybase** 与 Adaptive Server Enterprise 兼容。 Adaptive Server Enterprise 的最大 *max-length* 为 255。

另请参阅 "CHAR 数据类型 [字符]" 第 61 页

"LONG VARCHAR 数据类型 [字符]" 第 62 页

LONG VARCHAR 数据类型 [字符]

作用 任意长度的字符数据。

语法 **LONG VARCHAR**

用法 任意长度的字符串。最大大小受数据库文件的最大大小限制 （当前为 2 GB）。

除了字符串本身的长度外，还有一些额外的存储开销。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅　"CHAR 数据类型 [字符]" 第 61 页

"CHARACTER VARYING 数据类型 [字符]" 第 62 页

TEXT 数据类型 [字符]

作用　这是一个域。它是作为允许 NULL 的 LONG VARCHAR 来实现的。

语法　**TEXT**

用法　任意长度的字符串。用法与 LONG VARCHAR 相同。

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅　"LONG VARCHAR 数据类型 [字符]" 第 62 页

数值数据类型

作用

存储数值数据。

注意

- NUMERIC 和 DECIMAL 数据类型，以及各种 INTEGER 数据类型，有时被称为**精确**数值数据类型，与之相对的是**近似**数值数据类型 FLOAT、 DOUBLE 和 REAL。

  精确数值数据类型是那些可以指定精度值和小数位数值的数据类型，而近似数值数据类型是以预定义方式存储的。*只有精确数值数据肯定会在算术运算后精确到指定的最小有效位。*

- 在 5.5 版之前，长度超过四个字节的十六进制常量被视为字符串常量，其它十六进制常量则被视为整数。新的缺省行为是将它们视为二进制类型的常量。若要使用传统的行为，请将 TSQL_HEX_CONSTANTS 数据库选项设置为 "OFF"。

- 不允许小于 1 的数据类型长度和精度。

兼容性

- 只有小数位数为 0 的 NUMERIC 数据类型可以用于 Transact-SQL *identity* 列。

- 应该避免 NUMERIC 和 DECIMAL 数据类型的缺省精度和小数位数设置，因为它们在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 之间是不同的。在 Adaptive Server Anywhere 中，缺省精度为 30，缺省小数位数为 6。在 Adaptive Server Enterprise 中，缺省精度为 18，缺省小数位数为 0。

- FLOAT (*p*) 数据类型是 REAL 或 DOUBLE 数据类型的同义词，具体取决于 *p* 的值。对于 Adaptive Server Enterprise， REAL 用于 p 小于或等于 15 的数值，DOUBLE 用于 *p* 大于 15 的数值。对于 Adaptive Server Anywhere，分界值与平台相关，但在所有平台上分界值都大于 15。

有关通过设置数据库选项更改缺省值的信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > PRECISION 选项 [database]" 和 "ASA 数据库管理指南 > 数据库选项 > SCALE 选项 [database]"。

BIGINT 数据类型 [数值]

作用　需要 8 个字节存储的整数。

语法　[**UNSIGNED**] **BIGINT**

用法　BIGINT 数据类型是一种精确数值数据类型：其精度在算术运算后不变。

BIGINT 值需要 8 个字节存储。

带符号的 BIGINT 值的范围为 -2^{63} 到 $2^{63} - 1$，或 -9223372036854775808 到 9223372036854775807。

无符号的 BIGINT 值的范围为 0 到 $2^{64} - 1$，或 0 到 18446744073709551615。

缺省情况下，数据类型是带符号的。

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

另请参阅　"INT 或 INTEGER 数据类型 [数值]" 第 68 页

"TINYINT 数据类型 [数值]" 第 70 页

"SMALLINT 数据类型 [数值]" 第 70 页

DECIMAL 数据类型 [数值]

作用　总位数为 *precision* 小数点后位数为 *scale* 的十进制数字。

语法　{ **DECIMAL** | **DEC** } [**(** *precision* [, *scale*] **)**]

用法　DECIMAL 数据类型是精确数值数据类型，其精度在算术运算后保留到最小有效位。

十进制数字所需的存储可以按下面的公式来计算

```
2 + int( (before + 1)/2 ) + int( (after + 1)/2 )
```

函数 **int** 接受其参数的整数部分， **before** 和 **after** 是小数点前后的有效位数。存储基于的是所存储的值，而不是列中允许的最大精度和小数位数。

参数

precision 指定表达式中的位数的整数表达式。缺省设置是 30。

scale 指定小数点后位数的整数表达式。缺省设置是 6。

通过设置数据库选项可以更改缺省值。有关信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > PRECISION 选项 [database]" 和 "ASA 数据库管理指南 > 数据库选项 > SCALE 选项 [database]"。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"FLOAT 数据类型 [数值]" 第 67 页

"REAL 数据类型 [数值]" 第 69 页

"DOUBLE 数据类型 [数值]" 第 66 页

DOUBLE 数据类型 [数值]

作用

双精度浮点数。

语法

DOUBLE [**PRECISION**]

用法

DOUBLE 数据类型保存双精度浮点数。它是一种近似数值数据类型，在算术运算后容易产生舍入误差。 DOUBLE 值的近似特性意味着在比较 DOUBLE 值时通常应该避免使用相等运算符的查询。

DOUBLE 值需要 8 个字节存储。

值的范围为 2.22507385850721e-308 到 1.79769313486231e+308。保存为 DOUBLE 数据类型的值可精确到 15 个有效位，但超过第十五位可能会产生舍入误差。

标准和兼容性

- **SQL/92**　与 SQL/92 兼容。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅

"FLOAT 数据类型 [数值]" 第 67 页

"REAL 数据类型 [数值]" 第 69 页

"DECIMAL 数据类型 [数值]" 第 65 页

FLOAT 数据类型 [数值]

作用

浮点数，可以是单精度或双精度。

语法

FLOAT [**(** *precision* **)**]

用法

当使用 FLOAT (*precision*) 数据类型创建列时，所有平台上的列肯定会将值至少保存到指定的最小精度。相反， REAL 和 DOUBLE 不保证与平台无关的最小精度。

如果没有提供 *precision*， FLOAT 数据类型是单精度浮点数，等同于 REAL 数据类型，需要 4 个字节存储。

如果提供了 *precision*，则 FLOAT 数据类型是单精度或双精度，具体取决于指定的精度值。 REAL 和 DOUBLE 之间的分界值与平台相关。单精度 FLOAT 数据类型需要 4 个字节存储，双精度 FLOAT 数据类型则需要 8 个字节。

FLOAT 数据类型是一种近似数值数据类型。它在算术运算后容易产生舍入误差。 FLOAT 值的近似特性意味着在比较 FLOAT 值时通常应该避免使用相等运算符的查询。

参数

precision　指定小数点后位数的整数表达式。

标准和兼容性

- **SQL/92**　与 SQL/92 兼容。
- **Sybase**　可以使用 "ASA 数据库管理指南 > 数据库选项 > FLOAT_AS_DOUBLE 选项 [compatibility]" 优化 FLOAT 数据类型的行为，以便与 Adaptive Server Enterprise 兼容。

另请参阅

"DECIMAL 数据类型 [数值]" 第 65 页

"REAL 数据类型 [数值]" 第 69 页

"DOUBLE 数据类型 [数值]" 第 66 页

INT 或 INTEGER 数据类型 [数值]

作用 需要 4 个字节存储的整数。

语法 [**UNSIGNED**] { **INT** | **INTEGER** }

用法 INTEGER 数据类型是精确数值数据类型，其精度在算术运算后不变。

如果您指定 UNSIGNED，则整数永远不能被指定为负数。缺省情况下，数据类型是带符号的。

带符号的整数的范围为 -2^{31} 到 $2^{31} - 1$，或 -2147483648 到 2147483647。

无符号的整数的范围为 0 到 $2^{32} - 1$，或 0 到 4294967295。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。 UNSIGNED 关键字是供应商扩展。
- **Sybase** 带符号的数据类型与 Adaptive Server Enterprise 兼容。 Adaptive Server Enterprise 不支持 UNSIGNED 数据类型。

另请参阅

"BIGINT 数据类型 [数值]" 第 65 页

"TINYINT 数据类型 [数值]" 第 70 页

"SMALLINT 数据类型 [数值]" 第 70 页

数值数据类型 [数值]

作用 与 DECIMAL 相同。

语法 **NUMERIC** [**(** *precision* [, *scale*] **)**]

用法

NUMERIC 数据类型是一种精确数值数据类型，其精度在算术运算后保留到最小有效位。

存储十进制数字所需的字节数可以按下面的公式来计算

```
2 + int( (before+1)/2 ) + int( (after+1)/2 )
```

函数 **int** 接受其参数的整数部分， **before** 和 **after** 是小数点前后的有效位数。存储基于的是所存储的值，而不是列中允许的最大精度和小数位数。

参数

precision　指定表达式中的位数的整数表达式。缺省值是 30。

scale　指定小数点后位数的整数表达式。缺省值是 6。

通过设置数据库选项可以更改缺省值。有关信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > PRECISION 选项 [database]" 和 "ASA 数据库管理指南 > 数据库选项 > SCALE 选项 [database]"。

标准和兼容性

- **SQL/92**　如果 SCALE 选项设置为零，则与 SQL/92 兼容。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅

"FLOAT 数据类型 [数值]" 第 67 页

"REAL 数据类型 [数值]" 第 69 页

"DOUBLE 数据类型 [数值]" 第 66 页

REAL 数据类型 [数值]

作用

以 4 个字节存储的单精度浮点数。

语法

REAL

用法

REAL 数据类型是一种近似数值数据类型，在算术运算后容易产生舍入误差。

值的范围为 1.175495e-38 到 3.402823e+38。保存为 REAL 数据类型的值可精确到 6 个有效位，但超过第六位容易产生舍入误差。

REAL 值的近似特性意味着在比较 REAL 值时通常应该避免使用相等运算符的查询。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

SMALLINT 数据类型 [数值]

作用 需要 2 个字节存储的整数。

语法 **[UNSIGNED] SMALLINT**

用法 SMALLINT 数据类型是一种精确数值数据类型，其精度在算术运算后不变。它需要 2 个字节存储。

带符号的 SMALLINT 值的范围为 -2^{15} 到 $2^{15} - 1$，或 -32768 到 32767。

无符号的 SMALLINT 值的范围为 0 到 $2^{16} - 1$，或 0 到 65535。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。 UNSIGNED 关键字是供应商扩展。
- **Sybase** 带符号的数据类型与 Adaptive Server Enterprise 兼容。Adaptive Server Enterprise 不支持 UNSIGNED 数据类型。

另请参阅 "INT 或 INTEGER 数据类型 [数值]" 第 68 页

"TINYINT 数据类型 [数值]" 第 70 页

"BIGINT 数据类型 [数值]" 第 65 页

TINYINT 数据类型 [数值]

作用 需要 1 个字节存储的无符号整数。

语法 **[UNSIGNED] TINYINT**

用法

TINYINT 数据类型是一种精确数值数据类型；其精度在算术运算后不变。

可以将 TINYINT 显式指定为 UNSIGNED，但是该类型始终是无符号的，因此 UNSIGNED 修饰符无效。

TINYINT 值的范围为 0 到 2^8 - 1，或 0 到 255。

在嵌入式 SQL 中，不应该将 TINYINT 列读入定义为 *char* 或 *unsigned char* 的变量，因为这样做的结果是尝试将列的值转换为字符串，然后将第一个字节分配给程序中的变量。相反，应该将 TINYINT 列读入 2 个字节或 4 个字节的整数列。另外，若要从用 C 语言编写的应用程序向数据库中发送 TINYINT 值，变量 C 的类型应该为整数。

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅

"BIGINT 数据类型 [数值]" 第 65 页

"TINYINT 数据类型 [数值]" 第 70 页

"SMALLINT 数据类型 [数值]" 第 70 页

货币数据类型

作用 存储货币数据。

MONEY 数据类型 [货币]

作用 该数据类型便于存储货币数据，并保证了与 Adaptive Server Enterprise MONEY 数据类型的兼容性。

语法 **MONEY**

用法 MONEY 数据类型是一个域，它是作为允许 NULL 的 NUMERIC(19,4) 来实现的。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 中的货币数据类型是作为域来实现的，提供它的目的主要是为了与 Adaptive Server Enterprise 兼容。

另请参阅 "SMALLMONEY 数据类型 [货币]" 第 72 页

SMALLMONEY 数据类型 [货币]

作用 该数据类型便于存储不太大的货币数据，并提供了与 Adaptive Server Enterprise SMALLMONEY 数据类型的兼容性。

语法 **SMALLMONEY**

用法 SMALLMONEY 数据类型在 Adaptive Server Anywhere 中是一个域，是作为允许 NULL 的 NUMERIC(10,4) 来实现的。

标准和兼容性

- **SQL/92.** 供应商扩展。
- **Sybase.** Adaptive Server Anywhere 中的货币数据类型是作为域来实现的，提供它的目的主要是为了与 Adaptive Server Enterprise 兼容。

另请参阅

"MONEY 数据类型 [货币]" 第 72 页

Bit 数据类型

作用　存储布尔值。

语法　**BIT**

用法　缺省情况下， BIT 数据类型的列不允许 NULL。此行为与其它数据类型不同。如果愿意，您可以显式允许 NULL。

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

日期和时间数据类型

作用　　存储日期和时间。

向数据库发送日期和时间

可以使用以下方式之一向数据库发送日期和时间：

- 使用任何接口，以字符串形式
- 使用 ODBC，以 TIMESTAMP 结构形式
- 使用嵌入式 SQL，以 SQLDATETIME 结构形式

将时间以字符串 （对于 TIME 数据类型）或部分字符串 （对于 TIMESTAMP 或 DATE 数据类型）的形式发送到数据库时，小时、分钟和秒必须按照 *hh*:*mm*:*ss.sss* 格式用冒号分开，但是它们可以出现在字符串中的任何位置。以下是用于指定时间的有效、明确的字符串：

```
21:35 -- 24 hour clock if no am or pm specified
10:00pm -- pm specified, so interpreted as 12 hour
clock
10:00 -- 10:00am in the absence of pm
10:23:32.234 -- seconds and fractions of a second
included
```

将日期以字符串形式发送到数据库时，其到日期数据类型的转换是自动发生的。可以用以下两种方式之一提供字符串：

- 以 *yyyy/mm/dd* 或 *yyyy-mm-dd* 格式的字符串形式，其格式由数据库明确解释
- 以按照 DATE_ORDER 数据库选项解释的字符串形式

字符串到日期 / 时间转换的 Transact-SQL 兼容性

将字符串转换为日期和时间数据类型时， Adaptive Server Anywhere 和 Adaptive Server Enterprise 之间的行为有些差异。

如果将只包含时间值 （不包含日期）的字符串转换为日期 / 时间数据类型， Adaptive Server Enterprise 使用缺省日期 1990 年 1 月 1 日，而 Adaptive Server Anywhere 使用当前日期。

如果时间的小数部分不超过 3 位，根据值的前面是句点还是冒号，Adaptive Server Enterprise 以不同的方式对值进行解释。如果前面是冒号，则值表示秒的千分位。如果前面是句点，则一位表示十分位，两位表示百分位，三位表示千分位。而 Adaptive Server Anywhere 以同一种方式解释值，不管分隔符是什么。

示例

Adaptive Server Enterprise 按下面所显示的方式转换值。每一对的第二行与第一行的不同之处在于使用的是冒号而不是句点。

```
12:34:56.7 to 12:34:56.700
12:34:56:7 to 12:34:56.007
12.34.56.78 to 12:34:56.780
12.34.56:78 to 12:34:56.078
12:34:56.789 to 12:34:56.789
12:34:56:789 to 12:34:56.789
```

在两种情况下， Adaptive Server Anywhere 都以 Adaptive Server Enterprise 转换前面是句点的值的方式来转换毫秒值：

```
12:34:56.7 to 12:34:56.700
12:34:56:7 to 12:34:56.700
12.34.56.78 to 12:34:56.780
12.34.56:78 to 12:34:56.780
12:34:56.789 to 12:34:56.789
12:34:56:789 to 12:34:56.789
```

从数据库中检索日期和时间

可以使用下列方式之一从数据库中检索日期和时间：

- 使用任何接口，以字符串形式
- 使用 ODBC，以 TIMESTAMP 结构形式
- 使用嵌入式 SQL，以 SQLDATETIME 结构形式

以字符串形式检索日期或时间时，所检索的数据采用数据库选项 DATE_FORMAT、 TIME_FORMAT 和 TIMESTAMP_FORMAT 指定的格式。有关这些选项的说明，请参阅 "SET OPTION 语句 " 第 601 页。

有关用于处理日期和时间的函数的信息，请参阅 " 日期和时间函数 " 第 107 页。日期允许下列算术运算符：

- **timestamp + integer**　向日期或时间戳中添加指定的天数。
- **timestamp - integer**　从日期或时间戳中减去指定的天数。
- **date - date**　计算两个日期或时间戳之间的天数。
- **date + time**　创建组合给定日期和时间的时间戳。

比较数据库中的日期和时间

缺省情况下，作为 DATE 存储的值没有任何小时或分钟值，因此比较日期非常简单。

如果将 TRUNCATE_DATE_VALUES 选项设置为 "OFF"，则 DATE 数据类型还包含时间，这使日期的比较变得复杂。如果在将日期输入数据库时没有指定时间，则时间缺省为 0:00 或 12:00am （午夜）。带有此选项设置的任何日期比较操作，不仅比较日期本身，还比较时间。数据库日期值 "1999-05-23 10:00" 与常量 "1999-05-23" 不相等。可以使用 DATEFORMAT 函数或某个其它日期函数来比较日期和时间字段的各部分。例如，

```
DATEFORMAT(invoice_date,'yyyy/mm/dd') = '1999/05/23'
```

如果数据库列只需要日期，客户程序应该确保在将数据输入到数据库时不指定时间。这样，只包含日期字符串的比较会按预期的方式工作。

如果希望*以字符串形式*比较日期和字符串，则在比较前必须使用 DATEFORMAT 函数或 CAST 函数将日期转换为字符串。

使用明确的日期和时间

yyyy/mm/dd 或 *yyyy-mm-dd* 格式的日期总会被明确地识别为日期，与 DATE_ORDER 设置无关。可以使用其它字符代替 "**/**" 或 "**-**" 作为分隔符；例如， "**?**"、空格字符或 "**,**"。在不同的用户可能采用不同的 DATE_ORDER 设置的任何环境中都应该使用此格式。例如，在存储过程中，使用明确的日期格式可以防止按照用户的 DATE_ORDER 设置错误地解释日期。

另外， *hh*:*mm*:*ss.sss* 格式的字符串被明确地解释为时间。

对于日期和时间的组合，任何明确的日期和任何明确的时间都会产生一个明确的日期时间值。此外，格式

```
YYYY-MM-DD HH.MM.SS.SSS
```

是明确的日期时间值。句点只能在与日期组合的时间中使用。

在其它环境中，可以使用更灵活的日期格式。Adaptive Server Anywhere 可以将各种字符串解释为日期。具体的解释取决于数据库选项 DATE_ORDER 的设置。 DATE_ORDER 数据库选项可以具有值 *MDY*、*YMD* 或 *DMY*（请参阅 "SET OPTION 语句 " 第 601 页）。例如，以下语句将 DATE_ORDER 选项设置为 *DMY*：

```
SET OPTION DATE_ORDER = 'DMY' ;
```

缺省的 DATE_ORDER 设置为 "YMD"。每当建立连接时， ODBC 驱动程序都将 DATE_ORDER 选项设置为 "YMD"。仍然可以使用 SET TEMPORARY OPTION 语句更改该值。

数据库选项 DATE_ORDER 决定数据库将字符串 10/11/12 解释为 2010 年 11 月 12 日、2012 年 10 月 11 日还是 2012 年 11 月 10 日。日期字符串的年、月、日应该用某个字符 （/、 - 或空格）分开，并且应按照 DATE_ORDER 选项指定的顺序显示。

可以提供 2 位数或 4 位数的年份。选项 NEAREST_CENTURY 的值影响 2 位数年份的解释：对于小于 NEAREST_CENTURY 的值解释为 2000，对于所有其它值则解释为 1900。该选项的缺省值为 50。因此，缺省情况下将 50 解释为 1950，将 49 解释为 2049。

月份可以是月份的名称或编号。小时和分钟用冒号分开，但可以出现在字符串中的任何位置。

注意

- 我们建议应始终使用 4 位数格式指定年份。

  有关 2000 年兼容性问题的更多信息，请参阅 "2000 年兼容性问题 " 第 100 页。

- 对于 DATE_ORDER 的相应设置，下列字符串都是有效的日期：

  ```
  99-05-23 21:35
  99/5/23
  1999/05/23
  May 23 1999
  23-May-1999
  Tuesday May 23, 1999 10:00pm
  ```

- 如果字符串只包含部分日期说明，则使用缺省值填写日期。使用下列缺省值：

  - **年**　今年
  - **月**　没有缺省值
  - **日**　1（对月份字段很有用；例如， 1999 年 5 月解释为日期 1999-05-01 00:00）
  - **小时、分钟、秒、小数**　0

DATE 数据类型 [日期和时间]

作用

日历日期，如年、月、日。

语法

DATE

用法

年份可以从 0001 年到 9999 年。 Adaptive Server Anywhere 中的最小日期和时间是 0001-01-01 00:00:00。

由于历史的原因，如果 TRUNCATE_DATE_VALUES 选项设置为 "OFF"，则 DATE 列还可以包含小时和分钟。对于任何包含小时和分钟的对象，建议使用 TIMESTAMP 数据类型。

应用程序检索 DATE 值的格式由 DATE_FORMAT 设置控制。例如，表示 2003 年 7 月 19 日的日期值可能以 2003/07/19、 Jul 19, 2003 或其它某个可能的值的形式返回给应用程序。

数据库服务器将字符串解释为日期的方式由 DATE_ORDER 选项控制。例如，根据 DATE_ORDER 设置的不同，应用程序为 DATE 值提供的值 02/05/2002 在数据库中可能被解释为 5 月 2 日或 2 月 5 日。

DATE 值需要 4 个字节存储。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ASA 数据库管理指南 > 数据库选项 > DATE_FORMAT 选项 [compatibility]"

"ASA 数据库管理指南 > 数据库选项 > DATE_ORDER 选项 [compatibility]"

"DATETIME 数据类型 [日期和时间]" 第 80 页

"SMALLDATETIME 数据类型 [日期和时间]" 第 81 页

"TIMESTAMP 数据类型 [日期和时间]" 第 82 页

"ASA 数据库管理指南 > 数据库选项 > TRUNCATE_DATE_VALUES 选项 [database]"

DATETIME 数据类型 [日期和时间]

作用

作为 TIMESTAMP 实现的域。

语法

```
DATETIME
```

用法

提供 DATETIME 的目的主要是为了与 Adaptive Server Enterprise 兼容。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。有关特例，请参阅 " 字符串到日期 / 时间转换的 Transact-SQL 兼容性 " 第 75 页。

另请参阅

"DATE 数据类型 [日期和时间]" 第 79 页

"SMALLDATETIME 数据类型 [日期和时间]" 第 81 页

"TIMESTAMP 数据类型 [日期和时间]" 第 82 页

SMALLDATETIME 数据类型 [日期和时间]

作用　作为 TIMESTAMP 实现的域。

语法　**SMALLDATETIME**

用法　提供 SMALLDATETIME 的目的主要是为了与 Adaptive Server Enterprise 兼容。

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。有关特例，请参阅 " 字符串到日期 / 时间转换的 Transact-SQL 兼容性 " 第 75 页。

另请参阅

"DATE 数据类型 [日期和时间]" 第 79 页

"DATETIME 数据类型 [日期和时间]" 第 80 页

"TIMESTAMP 数据类型 [日期和时间]" 第 82 页

TIME 数据类型 [日期和时间]

作用　某天的时间，包括小时、分钟、秒和秒的小数。

语法　**TIME**

用法　小数保存到六个小数位。一个 TIME 值需要 8 个字节存储。（ODBC 标准将 TIME 数据类型限制为精确到秒。因此，在依赖于精度比秒更高的 WHERE 子句比较中，不要使用 TIME 数据类型。）

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

另请参阅

"TIMESTAMP 数据类型 [日期和时间]" 第 82 页

TIMESTAMP 数据类型 [日期和时间]

作用

时间点，包括年、月、日、小时、分钟、秒和秒的小数。

语法

TIMESTAMP

用法

小数保存到六个小数位。一个 TIMESTAMP 值需要 8 个字节存储。

尽管 TIMESTAMP 数据类型的可能日期范围与 DATE 类型相同 （从 0001 年到 9999 年），但 TIMESTAMP 日期类型的有用范围是从 1600-02-28 23:59:59 到 7911-01-01 00:00:00。早于或晚于此范围的 TIMESTAMP 的时间部分可能不完整。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"TIME 数据类型 [日期和时间]" 第 81 页

二进制数据类型

作用　存储二进制数据，包括图像和数据库不能解释的其它信息。

BINARY 数据类型 [二进制]

作用　具有指定最大长度 （以字节为单位）的二进制数据。

语法　**BINARY** [(*max-length*)]

用法　缺省 *max-length* 为 1。

允许的最大大小是 32767。BINARY 数据类型与 CHAR 数据类型相同，但用在比较中除外。 BINARY 值被精确地比较，而 CHAR 值以数据库的归类序列进行比较。

参数　*max-length* 指定表达式最大长度的整数表达式。

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　Adaptive Server Enterprise 支持的 *max-length* 达 255。

另请参阅　"LONG BINARY 数据类型 [BINARY]" 第 83 页

"VARBINARY 数据类型 [BINARY]" 第 85 页

LONG BINARY 数据类型 [BINARY]

作用　任意长度的二进制数据。

语法　**LONG BINARY**

用法　最大大小受数据库文件的最大大小限制。

有关限制的更多信息，请参阅 "ASA 数据库管理指南 > 物理限制 > 大小和数量限制 "。

标准和兼容性

- **SQL/92**　供应商扩展。

- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "BINARY 数据类型 [二进制]" 第 83 页

"VARBINARY 数据类型 [BINARY]" 第 85 页

IMAGE 数据类型 [BINARY]

作用 允许 NULL 的 LONG BINARY 数据。

语法 **IMAGE**

用法 IMAGE 在 Adaptive Server Anywhere 中是一个域，是作为允许 NULL 的 LONG BINARY 来实现的。提供它的目的主要是为了与 Adaptive Server Enterprise 兼容。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

UNIQUEIDENTIFIER 数据类型 [Binary]

作用 UUID （又称 GUID）值的存储。

语法 **UNIQUEIDENTIFIER**

用法 UNIQUEIDENTIFIER 数据类型是二进制 (16)，存储 UUID（Universally Unique Identifier，通用唯一标识符）或 GUID （Globally Unique Identifier，全局唯一标识符）值。

UUID 和 GUID 可用来唯一地标识表中的行。生成这些值以保证一台计算机上生成的值不会匹配在另一计算机上生成的 UUID 或 GUID。这些值可用作复制环境中的键。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅　"ASA SQL 用户指南 > 确保数据完整性 > NEWID 缺省值 "

"NEWID 函数 [杂类]" 第 185 页

"UUIDTOSTR 函数 [字符串]" 第 227 页

"STRTOUUID 函数 [字符串]" 第 217 页

VARBINARY 数据类型 [BINARY]

作用　与 BINARY 相同。

语法　**VARBINARY** [**(** *max-length* **)**]

用法　可变长度的二进制字符串。 *max-length* 的缺省值为 1。

标准和兼容性

- **SQL/92**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅　"BINARY 数据类型 [二进制]" 第 83 页

"LONG BINARY 数据类型 [BINARY]" 第 83 页

域

作用

域是内置数据类型的别名，包括适用的精度值和小数位数值，还可以包括 DEFAULT 值和 CHECK 条件。 Adaptive Server Anywhere 中预先定义了一些域 （如货币数据类型），您也可以添加更多自己的域。

域 （也被称为**用户定义的数据类型**）允许整个数据库中的列都被自动定义成相同的数据类型，有相同的 NULL 或 NOT NULL 条件、相同的 DEFAULT 设置和相同的 CHECK 条件。域有利于整个数据库的一致性，并且可以消除一些错误类型。

简单的域

域是通过 CREATE DOMAIN 语句来创建的。有关完整的语法说明，请参阅 "CREATE DOMAIN 语句 " 第 322 页。

下面的语句创建了一个名为 **street_address** 的数据类型，它是一个有 35 个字符的字符串。

```
CREATE DOMAIN street_address CHAR( 35 )
```

可以使用 CREATE DATATYPE 替代 CREATE DOMAIN，但不建议这样做，因为 CREATE DOMAIN 是 SQL/3 标准草案中使用的语法。

创建数据类型时需要资源权限。创建了某个数据类型后，执行 CREATE DOMAIN 语句的用户 ID 就是该数据类型的所有者。任何用户都可以使用该数据类型。与其它数据库对象不同，所有者名称从不作为数据类型名称的前缀。

在定义列时， **street_address** 数据类型的使用方式可能与任何其它数据类型完全相同。例如，在以下有两列的表中，第二列是 **street_address** 列：

```
CREATE TABLE twocol (
   id INT,
   street street_address
)
```

域可被其所有者或 DBA 使用 DROP DOMAIN 语句加以删除：

```
DROP DOMAIN street_address
```

只有当数据类型没有在数据库中的任何表中使用时，才能执行该语句。如果尝试删除正在使用中的域，会显示消息 " 表 'SYSUSERTYPE' 中行的主键正在另一个表中被引用 "。

域的约束和缺省值

在域中可以内置许多与列关联的属性，如允许 NULL 值、具有 DEFAULT 值等。以这样的数据类型 （域）定义的任何列都自动继承 NULL 设置、 CHECK 条件和 DEFAULT 值。这样就可以以一种类似的含义在整个数据库的列中内置一致性。

例如，示例数据库中的很多主键列是持有 ID 号的整数列。以下语句创建了对这些列可能有用的数据类型：

```
CREATE DOMAIN id INT
NOT NULL
DEFAULT AUTOINCREMENT
CHECK( @col > 0 )
```

以数据类型 **id** 创建的任何列都不能包含 NULL，它缺省为自动递增值，而且必须包含正数。在 *@col* 变量中，可以使用任何标识符取代 *col*。

在需要时，可以通过显式提供列的属性，替换数据类型的属性。一个以数据类型 **id** 创建的、显式允许 NULL 值的列确实允许 NULL，与 **id** 数据类型的设置无关。

兼容性

- **命名约束和缺省值**　在 Adaptive Server Anywhere 中，创建的域具有基本数据类型、可选的 NULL 或 NOT NULL 条件，一个缺省值和一个 CHECK 条件。不支持命名约束和命名缺省值。
- **创建数据类型**　在 Adaptive Server Anywhere 中，可以使用 **sp_addtype** 系统过程添加域，也可以使用 CREATE DOMAIN 语句。在 Adaptive Server Enterprise 中，必须使用 **sp_addtype**。

Java 类数据类型

作用

安装在数据库中的任何公共 Java 类都可作为 SQL 数据类型使用。 Java 类数据类型提供了在数据库中使用的抽象数据类型。

标准和用户定义的 Java 类

数据库中的 Java 类分为以下几类：

- **标准类** 标准 Java 类是那些属于 Sun Microsystems Java 开发 工具包 (JDK) 的类。标准类集的一个子集作为内置类安装在所有支持 Java 的数据库中。
- **用户定义的类** 具有 DBA 权限的用户可以在数据库中安装已编译的 Java 类。

不是所有的标准 Java 类都受支持

Adaptive Server Anywhere 支持 JDK 版本 1.1.8 和 1.3，但并非每个版本的所有标准 Java 类都受支持。这是有意安排的。有些类因为安全原因不受支持。有些类不受支持是因为它们可用于用户界面编程。最后，只有以单词 **java** 开头的包中的类被认为是百分之百纯粹的 Java，所以任何以其他单词开头（如 **sun** 或 **javax**）的包不受支持。

受支持的定义

当一个 Java 类被列为**受支持**时，这意味着它的所有方法在 Sybase VM 中都已实现。如果类被列为**部分受支持**，则它的一些方法还没有实现。

即使某个类受支持，并非受支持的 Java 类的所有方法肯定都有效。当受支持的类的方法依赖于不受支持的类时，其方法会失败。

不受支持时的错误条件

当类不受支持导致方法失败时，对于没有实现的方法，错误条件通常是一个未处理的异常：java.lang.UnsatisfiedLinkError。

受支持的 Java 包

本节列出了在支持 Java 的数据库中可用作 SQL 数据类型的内置类的包。有关可能不受支持或部分受支持的包中的任何类信息，请参阅 " 不受支持的 Java 包和类 " 第 90 页和 " 部分受支持的包和类 " 第 90 页。

在此没有列出的包必须安装到数据库中，才能用作数据类型。

- java.beans

- java.io。对文件访问进行管理的类仅受某些 Windows 操作系统支持，并且仅当 JAVA_INPUT_OUTPUT 选项设置为 "ON" 时才有效。请参阅 "ASA 数据库管理指南 > 数据库选项 > JAVA_INPUT_OUTPUT 选项 [database]"。
- java.lang
- java.lang.reflect
- java.lang.Thread
- java.math
- java.net
- java.net.PlainDatagramSocketImpl
- java.rmi
- java.rmi.dgc
- java.rmi.registry
- java.rmi.server
- java.security
- java.security.acl
- java.security.interfaces
- java.SQL。有关 JDBC 2.0 功能支持的详细信息，请参阅 "ASA 编程指南 > 使用 JDBC 访问数据 > 数据库功能中的 JDBC"。
- java.text
- java.util
- java.util.zip

不受支持的 Java 包和类

Sybase VM 不支持以下包中的类：

- java.applet
- java.awt
- java.awt.datatransfer
- java.awt.event
- java.awt.image
- 所有以 sun 作前缀的包。例如， sun.audio。

部分受支持的包和类

以下类是*部分受支持的*。它们有一些不受支持的内部方法：

- java.lang.ClassLoader
- java.lang.Compiler
- java.lang.Runtime (exec/load/loadlibrary)
- java.io.File
- java.io.FileDescriptor
- java.io.FileInputStream
- java.io.FileOutputStream
- java.io.RandomAccessFile
- java.util.zip.Deflater
- java.util.zip.Inflater

用户定义的 Java 类

具有 DBA 权限的用户可以将 Java 类安装到数据库中。任何安装到数据库中的公共类都可用作数据类型。

使用 JDK 准备类

Java 开发工具包 (JDK) 提供了准备安装到数据库中的类所需的工具。

❖ **使用 JDK 准备类的安装：**

1. 使用文本编辑器在数据库外编写一个 Java 类，并将其存储为 Java 源代码文件 （一般以 ".java" 作扩展名）。

   例如，可以创建一个名为 `MyFirstClass.java` 的文件。

2. 使用 `javac` 编译器编译 Java 类，以生成一个 Java 类文件 （以 ".class" 作扩展名）。

   例如，要编译文件 `MyFirstClass.java`，请在命令提示符处键入以下命令：

   ```
   javac MyFirstClass.java
   ```

安装类

有了已编译的 Java 类文件后，就可以将该文件安装到数据库中。可以使用 Sybase Central 或 Interactive SQL 方便地完成此项工作。

❖ **使用 Sybase Central 安装类：**

1. 从 Sybase Central 作为具有 DBA 权限的用户 ID 连接到数据库。
2. 打开 "Java 对象 " 文件夹。
3. 双击 " 添加 Java 类 "。按照向导中的指示来安装类。

❖ **使用 Interactive SQL 安装类：**

1. 从 Interactive SQL 作为具有 DBA 权限的用户 ID 连接到数据库。
2. 通过输入以下命令来安装类：

```
INSTALL JAVA NEW FROM FILE filename
```

有关安装类的更多信息，请参阅 "INSTALL 语句 " 第 524 页。

Java 类数据类型的大小写敏感性

Java 标识符 （包括数据类型）都是区分大小写的：Java **int** 数据类型不能写成 **INT**。 SQL 标识符 （包括数据类型）不区分大小写。 **int** 数据类型也可以写成 **Int** 或大小写字符的任意组合。

当 Java 类用作 SQL 数据类型时，数据类型总是区分大小写的。这是 SQL 标识符不区分大小写规则的特例。

示例

如果安装一个名为 Demo 的类，以下语句是*不*等效的：

```
CREATE TABLE t1 (
   id INT PRIMARY KEY
   demo_column Demo )
CREATE TABLE t1 (
   id INT PRIMARY KEY
   demo_column DEMO )
```

使用类作为数据类型

使用 Java 类数据类型创建表

可以创建一个其列基于 Java 类数据类型的表，就像使用任何其它数据类型那样。例如，如果 *MyClass* 是安装到数据库中的 Java 类，可以按如下方法使用该类创建表：

```
CREATE TABLE mytable (
   id INT NOT NULL PRIMARY KEY,
   mycol MyClass )
```

在该语句中， *MyClass* 数据类型作为一个 SQL 数据类型使用，但区分大小写，因为 Java 语言区分大小写。

插入 Java 对象

可以使用 INSERT 语句在表中插入 Java 对象，就像插入任何其它行那样。因为每一行都是类的一个单独实例，所以必须使用关键字 "NEW" 来创建实例。例如，

```
INSERT INTO t2
VALUES ( 1, NEW MyClass() )
```

此例中， MyClass() 是 Java 类名称，不是 SQL 数据类型，所以必须以正确的大小写输入。

此例中， MyClass() 没有参数，所以每行都是用缺省的构造函数创建的。一般而言，您可以提供类参数，在每行放置不同的值。

使用子类

如果安装一个类 （比如 MySubClass），它是 MyClass 的子类，这时可将 MySubClass 的实例插入数据类型为 MyClass 的列中。

数据类型转换

类型转换可以自动发生，也可以使用 CAST 或 CONVERT 函数显式请求类型转换。

如果字符串在数值表达式中使用或者作为需要数值参数的函数参数来使用，该字符串将被转换为数字。

如果数字在字符串表达式中使用或者作为字符串函数参数使用，该数字被转换为字符串后再使用。

所有日期常量都被指定为字符串。字符串被自动转换为日期后再使用。

在某些情况下不适合进行自动数据库转换。

```
'12/31/90' + 5
'a' > 0
```

在这里，自动数据类型转换失败。可用 CAST 或 CONVERT 函数强制进行类型转换。有关 CAST 和 CONVERT 函数的信息，请参阅 " 数据类型转换函数 " 第 106 页。

也可使用以下函数强制进行类型转换（请参阅 "SQL 函数 " 第 105 页）。

- **DATE(value)** 将表达式转换为日期，并删除任何小时、分钟或秒。可能会报告转换错误。
- **STRING(value)** 与 CAST(value AS LONG VARCHAR) 等效。
- 有关 STRING 函数的详细信息，参见 "STRING 函数 [字符串]" 第 217 页。
- **VALUE+0.0** 与 CAST(value AS DECIMAL) 等效。

有关 CAST 函数的详细信息，参见 "CAST 函数 [数据类型转换]" 第 125 页。

使用比较运算符的转换

在不同数据类型的参数间执行比较时 （如 "="），必须转换其中的一个参数或两个，使比较在同一个数据类型下执行。有时显式转换参数更适合。

Adaptive Server Anywhere 使用以下规则进行比较：

1. 如果参数的数据类型有公用的超类型，则转换为公用超类型然后进行比较。超类型是以下每个列表中的最后那个数据类型。

   - BIT → TINYINT → UNSIGNED SMALLINT → UNSIGNED INTEGER → UNSIGNED BIGINT → NUMERIC
   - SMALLINT → INTEGER → BIGINT → NUMERIC
   - REAL → DOUBLE
   - CHAR → LONG VARCHAR
   - BINARY → LONG BINARY
   - DATE → TIMESTAMP
   - TIME → TIMESTAMP

   例如，如果两个参数的数据类型为 BIT 和 TINYINT，则它们都转换为 NUMERIC。

2. 如果规则 1 不适用，且参数之一为 DATE 或 TIMESTAMP 数据类型，则转换为 TIMESTAMP 然后进行比较。

   例如，如果两个参数的数据类型为 REAL 和 DATE，则它们都转换为 TIMESTAMP。

3. 如果规则 1 和规则 2 均不适用，且一个参数的数据类型为 CHARACTER，而另一个的数据类型为 BINARY，则转换为 BINARY 然后进行比较。

4. 如果规则 1 到 3 都不适用，且一个参数的数据类型为 NUMERIC，而另一个的数据类型为 FLOAT，则转换为 DOUBLE 然后进行比较。

5. 如果没有一条规则适用，则转换为 NUMERIC 然后进行比较。

例如，如果两个参数的数据类型为 REAL 和 CHAR，则它们都转换为 NUMERIC。

注意

- 您可以不遵守上述规则，而将参数显式转换为其它类型。例如，如果要按 CHAR 来比较 DATE 和 CHAR，则需要将 DATE 显式转换为 CHAR。
- 规则 2 和规则 5 都可能会导致转换失败。

Java / SQL 数据类型转换

当一个 Java 类字段或方法在 SQL 语句中被调用时， Java 对象返回 Java 数据类型。它必须转换为 SQL 数据类型以便在 SQL 语句中 （例如在比较中）使用。

同样，当一个 SQL 值被指派给 Java 类字段或者作为一个参数提供 Java 类方法时，该 SQL 值必须转换为 Java 数据类型。

Java 到 SQL 和 SQL 到 Java 数据类型转换是按照 JDBC 标准执行的。以下各表描述了这些转换。

Java 到 SQL 的数据类型转换

Java 类型	SQL 类型
String	CHAR
String	VARCHAR
String	TEXT
java.math.BigDecimal	NUMERIC
java.math.BigDecimal	MONEY
java.math.BigDecimal	SMALLMONEY
boolean	BIT
byte	TINYINT
Short	SMALLINT
Int	INTEGER
long	INTEGER
float	REAL
double	DOUBLE
byte[ ]	VARBINARY

Java 类型	SQL 类型
byte[ ]	IMAGE
java.SQL.Date	DATE
java.SQL.Time	TIME
java.SQL.Timestamp	TIMESTAMP
java.lang.Double	DOUBLE
java.lang.Float	REAL
java.lang.Integer	INTEGER
java.lang.Long	INTEGER
void	this*

* 方法返回对象本身

SQL 到 Java 的数据类型转换

SQL 类型	Java 类型
CHAR	String
VARCHAR	String
TEXT	String
NUMERIC	java.math.BigDecimal
DECIMAL	java.math.BigDecimal
MONEY	java.math.BigDecimal
SMALLMONEY	java.math.BigDecimal
BIT	boolean
TINYINT	byte
SMALLINT	short

SQL 类型	Java 类型
INTEGER	int
BIGINT	long
REAL	float
FLOAT	double
DOUBLE	double
BINARY	byte[ ]
VARBINARY	byte[ ]
LONG VARBINARY	byte[ ]
IMAGE	byte[ ]
DATE	java.SQL.Date
TIME	java.SQL.Time
TIMESTAMP	java.SQL.Timestamp

2000 年兼容性问题

关于日期处理的问题，特别是围绕 2000 年的年份值处理，曾是计算机业界普遍关心的一个话题。

本节探讨了 Adaptive Server Anywhere 的 2000 年兼容性问题。本节还解释了 Adaptive Server Anywhere 在内部如何对日期值进行处理，以及 Adaptive Server Anywhere 如何处理不明确的日期信息，比如两位数年份字符串值的转换。

Sybase Adaptive Server Anywhere 及其旧版本的用户可以放心，日期的内部处理和存储方式没有受到从二十世纪过渡到二十一世纪的负面影响。

请看以下 Adaptive Server Anywhere 对 2000 年兼容性问题的处理措施：

- Adaptive Server Anywhere 总是对任何合法的日期算术和逻辑运算返回正确的值，不论计算出的值是否跨越不同的世纪。
- 在任何时候， Adaptive Server Anywhere 的内部日期存储都显式包括年份值的世纪部分。
- Adaptive Server Anywhere 的操作不受任何返回值 （包括当前日期）的影响。
- 日期值总能以完整的世纪格式输出。

文档的其它部分对本节中概括的许多与日期相关的主题有更详尽的介绍。

日期的存储方式

包含年份值的日期以下列数据类型之一在内部使用和存储在 Adaptive Server Anywhere 数据库中：

数据类型	包含	存储为	可能的值的范围
DATE	日历日期 （年、月、日）	4 个字节	0001-01-01 到 9999-12-31

数据类型	包含	存储为	可能的值的范围
TIMESTAMP	时间戳（年、月、日、小时、分钟、秒和精确到六个小数位的秒的小数）	8 个字节	0001-01-01 到 9999-12-31（TIMESTAMP 时间部分中早于 1600-02-28 23:59:59 和晚于 7911-01-01 00:00:00 的精度被删除）

有关 Adaptive Server Anywhere 日期和时间数据类型的更多信息，请参阅 " 日期和时间数据类型 " 第 75 页。

发送和检索日期值

日期值在 Adaptive Server Anywhere 中存储为 DATE 或 TIMESTAMP 数据类型，但它们是通过以下方法之一传递给 Adaptive Server Anywhere 或从 Adaptive Server Anywhere 中检索的：

- 以字符串的形式，使用任何 Adaptive Server Anywhere 编程接口。
- 以 TIMESTAMP 结构的形式，使用 ODBC。
- 以 SQLDATETIME 结构的形式，使用嵌入式 SQL。

如果包含日期值的字符串以以下格式传递：*yyyy-mm-dd*（"-" 破折号分隔符是允许的几种字符之一），则该字符串被认为是明确的，并自动转换为 DATE 或 TIMESTAMP 数据类型，不存在被错误解释的可能。

有关详细信息

通过设置 DATE_FORMAT 数据库选项，可以使用 *yyyy-mm-dd* 以外的日期格式。有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > DATE_FORMAT 选项 [compatibility]"。

有关明确的日期格式的更多信息，请参阅 " 使用明确的日期和时间 " 第 78 页。

有关 ODBC TIMESTAMP 结构的更多信息，请参阅 Microsoft 开放式数据库连接 SDK 或 " 向数据库发送日期和时间 " 第 75 页。

在 C 程序的开发中使用时，嵌入式 SQL SQLDATETIME 结构的年份值是一个 16 位带符号的整数。

有关 SQLDATETIME 数据类型的更多信息，请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > 嵌入式 SQL 数据类型 "。

闰年

2000 年是闰年，二月多了一天。 Adaptive Server Anywhere 使用全球通用的算法来确定哪些年是闰年。使用这种算法，当年份可被 4 整除时则为闰年，但年份为世纪年时 （如 1900 年）除外，此时年份只有在可被 400 整除时才是闰年。

Adaptive Server Anywhere 能正确处理所有的闰年。例如，以下 SQL 语句产生的返回值为 "Tuesday"：

```
SELECT DAYNAME('2000-02-29')
```

Adaptive Server Anywhere 接受 2000 年 2 月 29 日 （闰年）作为日期，并用这个日期来确定星期几。

但是， Adaptive Server Anywhere 会拒绝以下语句：

```
SELECT DAYNAME('2001-02-29')
```

这个语句会导致错误 （不能将 '2001-02-29' 转换为日期），因为 2001 年不存在 2 月 29 日。

不明确的字符串到日期的转换

当需要日期值时， Adaptive Server Anywhere 自动将字符串转换为日期，即使年份在字符串中是仅用两位数表示的。

如果年份值中的世纪部分被省略，转换方法取决于 NEAREST_CENTURY 数据库选项。

NEAREST_CENTURY 数据库选项是一个数值，充当 19YY 日期值和 20YY 日期值之间的断点。

小于 NEAREST_CENTURY 值的两位数年份值转换为 "20yy"，而大于或等于该值的年份转换为 "19yy"。

如果没有设置该选项，则采用缺省设置 "50"。因此，两位数的年份字符串被理解为是指从 1950 到 2049 之间的年份。

此 NEAREST_CENTURY 选项是在 SQL Anywhere 5.5 版中引入的。在 5.5 版中，缺省设置为 "0"。

不明确的日期转换示例

以下语句创建了一个表，可用来阐释 Adaptive Server Anywhere 中的不明确日期信息的转换。

```
CREATE TABLE T1 (C1 DATE);
```

表 T1 包含一列 C1，其数据类型为 DATE。

以下语句将日期值插入列 C1 中。 Adaptive Server Anywhere 自动转换含有不明确年份值的字符串，这种字符串用两位数表示年份，但没有指示出世纪。

```
INSERT INTO T1 VALUES('00-01-01');
```

缺省情况下， NEAREST_CENTURY 选项设置为 "50"，因此 Adaptive Server Anywhere 将上述字符串转换为日期 2000-01-01。以下语句校验此插入的结果。

```
SELECT * FROM T1;
```

用以下语句更改 NEAREST_CENTURY 选项会改变转换过程。

```
SET OPTION NEAREST_CENTURY = 0;
```

当 NEAREST_CENTURY 选项设置为 "0" 时，用同样的语句执行先前的插入会创建不同的日期值：

```
INSERT INTO T1 VALUES('00-01-01');
```

上述语句现在将插入日期显示为 1900-01-01。使用以下语句校验结果。

```
SELECT * FROM T1;
```

日期到字符串的转换

Adaptive Server Anywhere 提供了几个将 Adaptive Server Anywhere 日期和时间值转换为各种字符串和其它表达式的函数。在将日期值转换为字符串时，年份部分有可能减少为一个表示年份的两位数字，从而丢失日期中的世纪部分。

错误的世纪值

请看以下语句，该语句在没有数据库错误发生的情况下，错误地将一个表示日期 2000 年 1 月 1 日的字符串转换为表示 1900 年 1 月 1 日的字符串。

```
SELECT DATEFORMAT (
         DATEFORMAT('2000-01-01', 'Mmm dd/yy' ),
         'yyyy-Mmm-dd' )
   AS Wrong_year;
```

Adaptive Server Anywhere 将明确的日期字符串 2000-01-01 自动转换为正确的日期值。但是，内部 （即嵌套） DATEFORMAT 函数的 'Mmm dd/yy' 格式化功能在日期转换回字符串并传递给外部 DATEFORMAT 函数时会删除日期的世纪部分。

因为在此情况下数据库选项 NEAREST_CENTURY 设置为 "0"，所以外部 DATEFORMAT 函数将用两位数年份值表示日期的字符串转换为介于 1900 和 1999 之间的年份。

有关日期和时间函数的更多信息，请参阅 " 日期和时间函数 " 第 107 页。

第 3 章

SQL 函数

关于本章

函数用于从数据库返回信息。在任何可以使用表达式的地方都可以使用函数。

NULL 参数
除非另外说明，任何接收 NULL 作为参数的函数都返回 NULL。

本章对函数按类型进行了分组，然后提供了按函数的字母顺序排序的列表。

函数类型

本节将按类型对可用的函数进行分组。

集合函数

集合函数对数据库中的一组行上的数据进行汇总。这些组是使用 SELECT 语句的 GROUP BY 子句构成的。只能在 SELECT 语句的选择列表和 HAVING 和 ORDER BY 子句中使用集合函数。

函数列表

以下是可用的集合函数：

- "AVG 函数 [集合]" 第 123 页
- "COUNT 函数 [集合]" 第 135 页
- "LIST 函数 [集合]" 第 174 页
- "MAX 函数 [集合]" 第 179 页
- "MIN 函数 [集合]" 第 180 页
- "SUM 函数 [集合]" 第 220 页

数据类型转换函数

这些函数用于将参数从一种数据类型转换为另一种数据类型，或者测试它们是否可以转换。

兼容性

- Adaptive Server Enterprise 目前不支持 Adaptive Server Anywhere **cast** 函数。

函数列表

以下是可用的数据类型转换函数：

- "CAST 函数 [数据类型转换]" 第 125 页
- "CONVERT 函数 [数据类型转换]" 第 132 页
- "HEXTOINT 函数 [数据类型转换]" 第 164 页

- "INTTOHEX 函数 [数据类型转换]" 第 169 页
- "ISDATE 函数 [数据类型转换]" 第 170 页
- "ISNUMERIC 函数 [杂类]" 第 171 页

日期和时间函数

日期和时间函数对日期和时间数据类型执行操作，或者返回日期或时间信息。

在本章中，使用术语 " **日期时间** " 来表示日期或时间或者时间戳。具体的数据类型 DATETIME 表示为 DATETIME。

有关日期时间数据类型的更多信息，请参阅 " 日期和时间数据类型 " 第 75 页。

函数列表

以下是可用的日期和时间函数：

- "DATE 函数 [日期和时间]" 第 139 页
- "DATEADD 函数 [日期和时间]" 第 139 页
- "DATEDIFF 函数 [日期和时间]" 第 140 页
- "DATEFORMAT 函数 [日期和时间]" 第 142 页
- "DATENAME 函数 [日期和时间]" 第 143 页
- "DATEPART 函数 [日期和时间]" 第 143 页
- "DATETIME 函数 [日期和时间]" 第 144 页
- "DAY 函数 [日期和时间]" 第 144 页
- "DAYNAME 函数 [日期和时间]" 第 145 页
- "DAYS 函数 [日期和时间]" 第 145 页
- "DOW 函数 [日期和时间]" 第 150 页
- "GETDATE 函数 [日期和时间]" 第 161 页

- "HOUR 函数 [日期和时间]" 第 165 页
- "HOURS 函数 [日期和时间]" 第 165 页
- "MINUTE 函数 [日期和时间]" 第 181 页
- "MINUTES 函数 [日期和时间]" 第 181 页
- "MONTH 函数 [日期和时间]" 第 183 页
- "MONTHNAME 函数 [日期和时间]" 第 184 页
- "MONTHS 函数 [日期和时间]" 第 184 页
- "NOW 函数 [日期和时间]" 第 189 页
- "QUARTER 函数 [日期和时间]" 第 197 页
- "SECOND 函数 [日期和时间]" 第 205 页
- "SECONDS 函数 [日期和时间]" 第 206 页
- "TODAY 函数 [日期和时间]" 第 222 页
- "WEEKS 函数 [日期和时间]" 第 229 页
- "YEARS 函数 [日期和时间]" 第 231 页
- "YMD 函数 [日期和时间]" 第 232 页

日期部分

很多日期函数使用从**日期部分**生成的日期。下表显示了允许的日期部分值。

日期部分	缩写	值
Year	yy	1-9999
季度	qq	1-4
Month	mm	1-12

日期部分	缩写	值
Week	wk	1-54. 一周从星期日开始。
Day	dd	1-31
Dayofyear	dy	1-366
Weekday	dw	1-7 （星期日 = 1, …, 星期六 = 7）
Hour	hh	0-23
Minute	mi	0-59
Second	ss	0-59
Millisecond	ms	0-999
Calyearofweek	cyr	整数。一周开始的年份。包含一年头几天的周可能在前一年已经开始，具体取决于这一年从星期几开始。从星期一到星期四开始的年没有属于前一年的天，但从星期五到星期日开始的年在这一年的第一个星期一开始第一周。
Calweekofyear	cwk	1-54. 一年内包含指定日期的周的顺序号。
Caldayofweek	cdw	1-7. （星期日 = 1, …, 星期六 = 7）

Java 和 SQL 用户定义的函数

在 Adaptive Server Anywhere 中有两种创建用户定义函数的机制。您可以使用 SQL 语言编写函数，也可以使用 Java。

SQL 中用户定义的函数

您可以使用 "CREATE FUNCTION 语句 " 第 337 页 在 SQL 中实现自己的函数。 CREATE FUNCTION 语句内的 RETURN 语句确定函数的数据类型。

创建了 SQL 用户定义的函数后，可以在任何使用相同数据类型的内置函数的地方使用该函数。

有关创建 SQL 函数的更多信息，请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 "。

Java 中用户定义的函数

尽管 SQL 函数很有用，但 Java 类提供了更强大而灵活的用于实现用户定义的函数的方法，而且还有一个额外的优点：如果您愿意，可以将它们从数据库服务器移动到客户端应用程序。

已安装的 Java 类的任何**类方法**都可以作为用户定义的函数在任何使用相同数据类型的内置函数的地方使用。

实例方法与类的特定实例相关，所以其行为与标准用户定义的函数不同。

有关创建 Java 类和类方法的更多信息，请参阅 "ASA 编程指南 > 介绍数据库中的 Java > Java 讲座 "。

杂类函数

杂类函数对算术、字符串或日期 / 时间表达式 （包括其它函数的返回值）执行运算。

函数列表

以下是可用的杂类函数：

- "ARGN 函数 [杂类]" 第 120 页
- "COALESCE 函数 [杂类]" 第 129 页
- "ESTIMATE 函数 [杂类]" 第 151 页
- "ESTIMATE_SOURCE 函数 [杂类]" 第 152 页
- "EXPERIENCE_ESTIMATE 函数 [杂类]" 第 157 页
- "EXPLANATION 函数 [杂类]" 第 157 页
- "GET_IDENTITY 函数 [杂类]" 第 159 页
- "GRAPHICAL_PLAN 函数 [杂类]" 第 161 页
- "GRAPHICAL_ULPLAN 函数 [杂类]" 第 163 页
- "GREATER 函数 [杂类]" 第 164 页
- "IDENTITY 函数 [杂类]" 第 166 页

- "IFNULL 函数 [杂类]" 第 167 页
- "INDEX_ESTIMATE 函数 [杂类]" 第 168 页
- "ISNULL 函数 [数据类型转换]" 第 170 页
- "LESSER 函数 [杂类]" 第 174 页
- "LONG_ULPLAN 函数 [杂类]" 第 177 页
- "NEWID 函数 [杂类]" 第 185 页
- "NULLIF 函数 [杂类]" 第 189 页
- "NUMBER 函数 [杂类]" 第 190 页
- "PLAN 函数 [杂类]" 第 193 页
- "REWRITE 函数 [杂类]" 第 202 页
- "SHORT_ULPLAN 函数 [杂类]" 第 207 页
- "SQLDIALECT 函数 [杂类]" 第 215 页
- "TRACEBACK 函数 [杂类]" 第 222 页
- "TRANSACTSQL 函数 [杂类]" 第 223 页
- "VAREXISTS 函数 [杂类]" 第 228 页
- "WATCOMSQL 函数 [杂类]" 第 229 页

数字函数

数字函数执行数字数据类型的数学运算或者返回数字信息。

函数列表

以下是可用的数字函数：

- "ABS 函数 [数字]" 第 119 页
- "ACOS 函数 [数字]" 第 119 页

- "ASIN 函数 [数字]" 第 121 页
- "ATAN 函数 [数字]" 第 122 页
- "ATN2 函数 [数字]" 第 122 页
- "CEILING 函数 [数字]" 第 126 页
- "COS 函数 [数字]" 第 134 页
- "COT 函数 [数字]" 第 135 页
- "DEGREES 函数 [数字]" 第 148 页
- "EXP 函数 [数字]" 第 156 页
- "FLOOR 函数 [数字]" 第 159 页
- "LOG 函数 [数字]" 第 176 页
- "LOG10 函数 [数字]" 第 177 页
- "MOD 函数 [数字]" 第 183 页
- "PI 函数 [数字]" 第 192 页
- "POWER 函数 [数字]" 第 194 页
- "RADIANS 函数 [数字]" 第 198 页
- "RAND 函数 [数字]" 第 198 页
- "REMAINDER 函数 [数字]" 第 199 页
- "ROUND 函数 [数字]" 第 204 页
- "SIGN 函数 [数字]" 第 208 页
- "SIN 函数 [数字]" 第 209 页
- "SQRT 函数 [数字]" 第 215 页
- "TAN 函数 [数字]" 第 221 页

- "TRUNCATE 函数 [数字]" 第 224 页
- "TRUNCNUM 函数 [数字]" 第 225 页

字符串函数

字符串函数执行字符串的转换、抽取或处理操作，或者返回有关字符串的信息。

当使用多字节字符集时，请仔细检查所使用的函数是否返回有关字符或字节的信息。

函数列表

以下是可用的字符串函数：

- "ASCII 函数 [字符串]" 第 121 页
- "BYTE_LENGTH 函数 [字符串]" 第 124 页
- "BYTE_SUBSTR 函数 [字符串]" 第 125 页
- "CHAR 函数 [字符串]" 第 127 页
- "CHARINDEX 函数 [字符串]" 第 127 页
- "CHAR_LENGTH 函数 [字符串]" 第 128 页
- "COMPARE 函数 [字符串]" 第 129 页
- "CSCONVERT 函数 [字符串]" 第 136 页
- "DIFFERENCE 函数 [字符串]" 第 149 页
- "INSERTSTR 函数 [字符串]" 第 168 页
- "LCASE 函数 [字符串]" 第 172 页
- "LEFT 函数 [字符串]" 第 172 页
- "LENGTH 函数 [字符串]" 第 173 页
- "LOCATE 函数 [字符串]" 第 175 页

- "LOWER 函数 [字符串]" 第 178 页
- "LTRIM 函数 [字符串]" 第 179 页
- "PATINDEX 函数 [字符串]" 第 191 页
- "REPEAT 函数 [字符串]" 第 199 页
- "REPLACE 函数 [字符串]" 第 200 页
- "REPLICATE 函数 [字符串]" 第 201 页
- "RIGHT 函数 [字符串]" 第 204 页
- "RTRIM 函数 [字符串]" 第 205 页
- "SIMILAR 函数 [字符串]" 第 209 页
- "SORTKEY 函数 [字符串]" 第 210 页
- "SOUNDEX 函数 [字符串]" 第 213 页
- "SPACE 函数 [字符串]" 第 214 页
- "STR 函数 [字符串]" 第 216 页
- "STRING 函数 [字符串]" 第 217 页
- "STRTOUUID 函数 [字符串]" 第 217 页
- "STUFF 函数 [字符串]" 第 218 页
- "SUBSTRING 函数 [字符串]" 第 219 页
- "TRIM 函数 [字符串]" 第 223 页
- "UCASE 函数 [字符串]" 第 226 页
- "UPPER 函数 [字符串]" 第 226 页
- "UUIDTOSTR 函数 [字符串]" 第 227 页

系统函数

系统函数用于返回系统信息。

函数列表

以下是可用的系统函数：

- "CONNECTION_PROPERTY 函数 [系统]" 第 131 页
- "DATALENGTH 函数 [系统]" 第 138 页
- "DB_ID 函数 [系统]" 第 147 页
- "DB_NAME 函数 [系统]" 第 147 页
- "DB_PROPERTY 函数 [系统]" 第 148 页
- "EVENT_CONDITION 函数 [系统]" 第 153 页
- "EVENT_CONDITION_NAME 函数 [系统]" 第 155 页
- "EVENT_PARAMETER 函数 [系统]" 第 155 页
- "NEXT_CONNECTION 函数 [系统]" 第 187 页
- "NEXT_DATABASE 函数 [系统]" 第 188 页
- "PROPERTY 函数 [系统]" 第 195 页
- "PROPERTY_DESCRIPTION 函数 [系统]" 第 195 页
- "PROPERTY_NAME 函数 [系统]" 第 196 页
- "PROPERTY_NUMBER 函数 [系统]" 第 197 页

兼容性

下表显示了 Adaptive Server Enterprise 系统函数以及它们在 Adaptive Server Anywhere 中的状态：

作用	状态
Col_length	已实现
Col_name	已实现

作用	状态
Curunreservedpgs	未实现
Data_pgs	未实现
Datalength	已实现
Db_id	已实现
Db_name	已实现
Host_id	未实现
Host_name	未实现
Index_col	已实现
Lct_admin	未实现
Object_id	已实现
Object_name	已实现
Proc_role	总是返回 0
Reserved_pgs	未实现
Rowcnt	未实现
Show_role	总是返回 NULL
Suser_id	已实现
Suser_name	已实现
Tsequal	已实现
Used_pgs	未实现
User_id	已实现
User_name	已实现
Valid_name	未实现
Valid_user	未实现

注意

- 某些系统函数在 Adaptive Server Anywhere 中是作为存储过程来实现的。

- *db_id*、 *db_name* 和 *datalength* 函数是作为内置函数来实现的。

下表描述了已实现的系统函数。

系统函数	说明
Col_length(*table-name*, *column-name* **)**	返回定义的列长度
Col_name(*table-id*, *column-id* [, *database-id*] **)**	返回列名称
Datalength(*expression* **)**	返回表达式的长度 （以字节为单位）
Db_id([*database-name*] **)**	返回数据库 ID 号
Db_name([*database-id*] **)**	返回数据库名
Index_col (*table-name*, *index-id*, *key_#* [, *userid*] **)**	返回索引列的名称
Object_id (*object-name* **)**	返回对象 ID
Object_name (*object-id* [, *database-id*] **)**	返回对象名称
Suser_id([*user-name*] **)**	返回整数用户标识号
Suser_name([*user-id*] **)**	返回用户 ID （在 Adaptive Server Enterprise 中返回服务器用户名）
Tsequal (*timestamp*, *timestamp2* **)**	比较时间戳的值以防止更新自被选取后又被修改的行
User_id([*user-name*] **)**	返回整数用户标识号。不返回 Adaptive Server Anywhere 用户 ID
User_name([*user-id*] **)**	返回用户 ID （在 Adaptive Server Enterprise 中返回用户名）

文本和图像函数

文本和图像函数对文本和图像数据类型进行操作。 Adaptive Server Anywhere 仅支持 *textptr* 文本和图像函数。

兼容性

- Adaptive Server Anywhere 目前不支持 Adaptive Server Enterprise **textvalid** 函数。

函数列表

以下是可用的文本和图像函数：

- "TEXTPTR 函数 [文本和图像]" 第 221 页

按字母顺序排列的函数列表

列出了每个函数，并在它旁边指出函数类型 （数字、字符等等）。

有关给定类型的所有函数的链接，请参阅 "" 函数类型 " 第 106 页 "。

ABS 函数 [数字]

作用 返回数字表达式的绝对值。

语法 **ABS (** *numeric-expression* **)**

参数 **numeric expression** 要返回其绝对值的数字。

示例 下面的语句返回值 66。

```
SELECT ABS( -66 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 核心 SQL 外部的 SQL/ 基础功能。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

ACOS 函数 [数字]

作用

- 返回数字表达式的反余弦值 （以弧度表示）。

语法 **ACOS (** *numeric-expression* **)**

参数 **numeric-expression** 角度的余弦值。

示例 下面的语句返回值 1.023945。

```
SELECT ACOS( 0.52 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"ASIN 函数 [数字]" 第 121 页

"ATAN 函数 [数字]" 第 122 页

"ATN2 函数 [数字]" 第 122 页

"COS 函数 [数字]" 第 134 页

ARGN 函数 [杂类]

作用

从参数列表中返回所选参数。

语法

ARGN (*integer-expression*, *expression* [, …] **)**

参数

integer expression 表达式列表中的参数位置。

expression 传递给函数的任意数据类型的表达式。提供的所有表达式必须具有相同的数据类型。

示例

下面的语句返回值 6。

```
SELECT ARGN( 6, 1,2,3,4,5,6 )
```

用法

使用整数表达式的值作为 n，从其余参数列表中返回第 n 个参数 （从 1 开始）。虽然表达式可以是任意数据类型，但它们必须具有相同的数据类型。整数表达式必须是从 1 到列表中的表达式数目，否则返回 NULL。多个表达式之间用逗号分隔。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

ASCII 函数 [字符串]

作用　返回字符串表达式中第一个字节的整数 ASCII 值。

语法　**ASCII (** *string-expression* **)**

参数　**string-expression**　字符串。

示例　下面的语句返回值 90。

```
SELECT ASCII( 'Z' )
```

用法　如果字符串为空，则 ASCII 返回零。文字字符串必须用引号括起来。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

ASIN 函数 [数字]

作用　返回一个数字的反正弦值 （以弧度表示）。

语法　**ASIN (** *numeric-expression* **)**

参数　**numeric-expression**　角度的正弦值。

示例　下面的语句返回值 .546850。

```
SELECT ASIN( 0.52 )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅　"ACOS 函数 [数字]" 第 119 页

"ATAN 函数 [数字]" 第 122 页

"ATN2 函数 [数字]" 第 122 页

"SIN 函数 [数字]" 第 209 页

ATAN 函数 [数字]

作用 返回一个数字的反正切值 （以弧度表示）。

语法 **ATAN (** *numeric-expression* **)**

参数 **numeric-expression** 角度的正切值。

示例 下面的语句返回值 **0.479519**。

```
SELECT ATAN( 0.52 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"ACOS 函数 [数字]" 第 119 页

"ASIN 函数 [数字]" 第 121 页

"ATN2 函数 [数字]" 第 122 页

"TAN 函数 [数字]" 第 221 页

ATN2 函数 [数字]

作用 返回两个数字的比率的反正切值 （以弧度表示）。

语法 { **ATN2** | **ATAN2** } **(** *numeric-expression1*, *numeric-expression2* **)**

参数 **numeric-expression1** 要计算其反正切值的比率中的分子。

numeric-expression2 要计算其反正切值的比率中的分母。

示例

下面的语句返回值 .008666。

```
SELECT ATAN2( 0.52, 060 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** ATN2 与 Adaptive Server Enterprise 兼容。Adaptive Server Enterprise 不支持 ATAN2。

另请参阅

"ACOS 函数 [数字]" 第 119 页

"ASIN 函数 [数字]" 第 121 页

"ATAN 函数 [数字]" 第 122 页

"TAN 函数 [数字]" 第 221 页

AVG 函数 [集合]

作用

计算一组行、数字表达式或一组唯一值的平均值。

语法

AVG (*numeric-expression* | **DISTINCT** *column-name* **)**

参数

numeric-expression 要在一组行上计算其平均值的表达式。

DISTINCT column-name 计算 *column-name* 中的唯一值的平均值。其用途有限，但出于完整性考虑将其包括在内。

示例

下面的语句返回值 49988.6。

```
SELECT AVG( salary ) FROM employee
```

用法

该平均值不包括 *numeric expression* 的值为 NULL 的行。对于不包含任何行的组，返回 NULL 值。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。

- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"SUM 函数 [集合]" 第 220 页

"COUNT 函数 [集合]" 第 135 页

BYTE_LENGTH 函数 [字符串]

作用

返回字符串中的字节数。

语法

BYTE_LENGTH (*string-expression* **)**

参数

string-expression 要计算其长度的字符串。

示例

下面的语句返回值 12。

```
SELECT BYTE_LENGTH( 'Test Message' )
```

用法

返回的长度中包括尾随空格字符。

NULL 字符串的返回值为 NULL。

如果字符串在多字节字符集中，则 BYTE_LENGTH 值不同于 CHAR_LENGTH 返回的字符数。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"CHAR_LENGTH 函数 [字符串]" 第 128 页

"DATALENGTH 函数 [系统]" 第 138 页

"LENGTH 函数 [字符串]" 第 173 页

BYTE_SUBSTR 函数 [字符串]

作用　返回字符串的子串。子串是用字节而不是字符来计算的。

语法　**BYTE_SUBSTR (** *string-expression*, *start* [, *length*] **)**

参数　**string- expression**　从中获取子串的字符串。

start　表示子串开始位置的整数表达式。正整数表示从字符串开始处开始，第一个字符位置为 1。负整数表示子串从字符串结尾处开始，最后一个字符的位置为 -1。

length　表示子串长度的整数表达式。正长度表示从开始位置*开始*　选取的字节数。负长度表示到开始位置*结束* 选取的字节数。

示例　下面的语句返回值 *age* 。

```
SELECT BYTE_SUBSTR( 'Test Message',-1,-3 )
```

用法　如果指定 *length*，则子串被限定为该字节数。 *start* 和 *length* 可以是正数也可以是负数。负起始位置表示从字符串结尾处开始计算而不是从开始处计算的字节数。正 *length* 表示子串在起始位置的右侧 *length* 字节数处结束，而负 *length* 表示子串在起始位置的左侧 *length* 字节数处结束并在 *start* 指定的位置处结束。使用适当的负数和正数组合，可以从字符串的开始处或结束处获取子串。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

另请参阅　"SUBSTRING 函数 [字符串]" 第 219 页

CAST 函数 [数据类型转换]

作用　返回转换为提供的数据类型的表达式的值。

语法　**CAST (** *expression* **AS** *data type* **)**

参数 **expression** 要转换的表达式。

data type 目标数据类型。

示例 下面的函数确保字符串被用作日期：

```
CAST( '2000-10-31' AS DATE )
```

计算表达式 1 + 2 的值，并将结果转换为单字符字符串。

```
CAST( 1 + 2 AS CHAR )
```

可以使用 **CAST** 函数缩短字符串：

```
CAST( Surname AS CHAR(10) )
```

用法 如果未指定字符串类型的长度，数据库服务器将选择适当的长度。如果没有为十进制转换指定精度和小数位数，则数据库服务器将选择适当的值。

标准和兼容性

- **SQL/92** 此函数与 SQL/92 兼容。
- **SQL/99** 核心特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "CONVERT 函数 [数据类型转换]" 第 132 页

CEILING 函数 [数字]

作用

- 返回一个数字的上限 （不小于的最小整数）。

语法 **CEILING (** *numeric-expression* **)**

参数 **numeric expression** 要计算其上限的数字。

示例 下面的语句返回值 60。

```
SELECT CEILING( 59.84567 )
```

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "FLOOR 函数 [数字]" 第 159 页

CHAR 函数 [字符串]

作用 返回数字所表示 ASCII 码值的字符。

语法 **CHAR (** *integer-expression* **)**

参数 **integer expression** 要转换成 ASCII 字符的数字。该数字必须在 0 到 255 （包括 0 和 255）之间。

示例 下面的语句返回值 Y。

```
SELECT CHAR( 89 )
```

用法 根据二进制排序顺序，返回的字符与当前数据库的字符集中提供的数字表达式相对应。

对于值大于 255 或小于 0 的整数表达式， CHAR 返回 NULL。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

CHARINDEX 函数 [字符串]

作用 返回一个字符串在另一个字符串中的位置。

语法 **CHARINDEX (** *string-expression1, string-expression2* **)**

参数 **string expression1** 您要搜索的字符串。

string expression2 被搜索的字符串。

示例

语句

```
SELECT emp_lname, emp_fname
FROM employee
WHERE CHARINDEX('K', emp_lname ) = 1
```

返回下列值：

emp_lname	Emp_fname
Klobucher	James
Kuo	Felicia
Kelly	Moira

用法

被搜索的字符串中第一个字符的位置是 1。

如果被搜索的字符串中含有另一个字符串的多个实例，则 CHARINDEX 返回第一个实例的位置。

如果被搜索的字符串中不含另一个字符串，则 CHARINDEX 返回 0。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"SUBSTRING 函数 [字符串]" 第 219 页

CHAR_LENGTH 函数 [字符串]

作用

返回字符串中的字符数。

语法

CHAR_LENGTH (*string-expression* **)**

参数

string-expression 要计算其长度的字符串。

用法

返回的长度中包括尾随空格字符。

NULL 字符串的返回值为 NULL。

如果字符串在多字节字符集中，则 CHAR_LENGTH 值不同于 BYTE_LENGTH 返回的字节数。

示例 下面的语句返回值 **8**。

```
SELECT CHAR_LENGTH( 'Chemical' )
```

标准和兼容性

- **SQL/92** 此函数与 SQL/92 兼容。
- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "BYTE_LENGTH 函数 [字符串]" 第 124 页

COALESCE 函数 [杂类]

作用 返回列表中的第一个非空表达式。

语法 **COALESCE (** *expression*, *expression* [, ...] **)**

参数 **expression** 任意表达式。

示例 下面的语句返回值 34。

```
SELECT COALESCE( NULL, 34, 13, 0 )
```

标准和兼容性

- **SQL/92** SQL/92。
- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

COMPARE 函数 [字符串]

作用 用于根据可选归类规则直接比较两个字符串。

语法 **COMPARE (** *string-expression-1*, *string-expression-2* [, *collation-name* | , *collation-id*] **)**

参数

string-expression-1 第一个字符串表达式。

string-expression-2 第二个字符串表达式。

字符串表达式只能包含以数据库的字符集编码的字符。

collation-name 一个字符串或字符变量，用于指定要使用的排序顺序的名称。有关有效归类名的列表，请参阅 ""SORTKEY 函数 [字符串]" 第 210 页 "。

collation-id 用于指定要使用的排序顺序的变量或整数常量。只能对内部归类使用 collation-id。有关更多信息，请参阅 ""SORTKEY 函数 [字符串]" 第 210 页 "。

如果未指定归类名或 ID，则缺省值为 " 缺省 Unicode 多语言 "。

用法

COMPARE 函数根据您选择的归类规则返回下列值：

值	含义
1	string-expression-1 大于 string-expression-2
0	string-expression-1 等于 string-expression-2
-1	string-expression-1 小于 string-expression-2

COMPARE 函数不将空字符串和仅包含空格的字符串视为相等，即使数据库启用了空白填充。 COMPARE 使用 SORTKEY 函数生成用于比较的归类关键字。因此，空字符串、含有一个空格的字符串和含有两个空格的字符串在比较时不会相等。

如果 *string-expression-1* 或 *string-expression-2* 中有一个为空，则结果为空。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"SORTKEY 函数 [字符串]" 第 210 页

CONNECTION_PROPERTY 函数 [系统]

作用 以字符串形式返回给定连接属性的值。

语法 **CONNECTION_PROPERTY (** { *integer-expression* | *string-expression* } … [, *integer-expression*] **)**

参数 **integer-expression** 大多数情况下，以第一个参数提供字符串表达式更方便。如果您确实提供了整数表达式，则它将是连接属性 ID。这可以用 PROPERTY_NUMBER 函数来确定。

string-expression 连接属性名称。必须指定属性 ID 或属性名称两者中的一个。

有关连接属性的列表，请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "。

integer-expression 当前数据库连接的连接 ID。如果省略此参数，则使用当前连接。

示例 下面的语句返回所维护的 Prepared 语句的数目。

```
SELECT connection_property( 'PrepStmt' )
```

用法 如果省略第二个参数，则使用当前连接。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "

"PROPERTY_NUMBER 函数 [系统]" 第 197 页

CONVERT 函数 [数据类型转换]

作用　返回转换成提供的数据类型的表达式。

语法　**CONVERT (** *data type*, *expression* [, *format-style*] **)**

参数

data type　表达式将转换成的数据类型。

expression　要转换的表达式。

format-style　对于将字符串转换为日期或时间数据类型以及相反的转换过程， *format-style* 是描述要使用的日期格式字符串的样式代码。*format-style* 参数的值具有下列含义：

不含世纪 (yy)	含世纪 (yyyy)	输出
-	0 或 100	Mmm dd yyyy hh:nn:ss:sss AM （或 PM）
1	101	mm/dd/yy[yy]
2	102	[yy]yy.mm.dd
3	103	dd/mm/yy[yy]
4	104	dd.mm.yy[yy]
5	105	dd-mm-yy[yy]
6	106	dd Mmm yy[yy]
7	107	Mmm dd, yy[yy]
8	108	hh:nn:ss
-	9 或 109	Mmm dd yyyy hh:nn:ss:sssAM （或 PM）
10	110	mm-dd-yy[yy]
11	111	[yy]yy/mm/dd
12	112	[yy]yymmdd
13	113	dd Mmm yyy hh:nn:ss:sss （24 小时制，欧洲缺省时间 + 毫秒， 4 位数年份）

不含世纪 (yy)	含世纪 (yyyy)	输出
14	114	hh:nn:ss:sss （24 小时制）
20	120	yyyy-mm-dd hh:nn:ss:sss （24 小时制，ODBC 规范， 4 位数年份）
21	121	yyyy-mm-dd hh:nn:ss.sss （24 小时制，ODBC 规范加毫秒， 4 位数年份）

如果未提供 *format-style* 参数，则使用样式代码 0。

有关每种输出符号 （如 Mmm）生成的样式的描述，请参阅 "ASA 数据库管理指南 > 数据库选项 > DATE_FORMAT 选项 [compatibility]"。

示例

下列语句说明了格式样式的用法：

```
SELECT CONVERT( CHAR( 20 ), order_date, 104 )
FROM sales_order
```

order_date
16.03.2000
20.03.2000
23.03.2000
25.03.2000
…

```
SELECT CONVERT( CHAR( 20 ), order_date, 7 )
FROM sales_order
```

order_date
Mar 16, 00
Mar 20, 00
Mar 23, 00

order_date
Mar 25, 00
…

下面的语句说明了到整数的转换，并返回值 5：

```
SELECT CONVERT( integer, 5.2 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"CAST 函数 [数据类型转换]" 第 125 页

COS 函数 [数字]

作用

- 返回一个数字的余弦值。

语法

COS (*numeric-expression* **)**

参数

numeric expression 以弧度表示的角度。

示例

语句

```
SELECT COS( 0.52 )
```

返回值 0.86781。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"ACOS 函数 [数字]" 第 119 页

"COT 函数 [数字]" 第 135 页

"SIN 函数 [数字]" 第 209 页

"TAN 函数 [数字]" 第 221 页

COT 函数 [数字]

作用

- 返回一个数字的余切值。

语法

COT (*numeric-expression* **)**

参数

numeric-expression　角度 （以弧度为单位）。

示例

下面的语句返回值 1.74653。

```
SELECT COT( 0.52 )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅

"COS 函数 [数字]" 第 134 页

"SIN 函数 [数字]" 第 209 页

"TAN 函数 [数字]" 第 221 页

COUNT 函数 [集合]

作用

根据指定的参数计算组中的行数。

语法

COUNT (* | *expression* | **DISTINCT** { *expression* | *column-name* } **)**

参数

*****　返回每组中的行数。

expression　返回每个 *expression 不为空值*的组中的行数。

DISTINCT expression 或 column-name 返回表达式中不同值的数目或名称为 *column-name* 的列的数目。计数中不包括值为 NULL 的行。

示例 下面的语句返回每个唯一的城市和含有该城市值的行数：

```
SELECT city , Count(*)
FROM employee
GROUP BY city
```

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "AVG 函数 [集合]" 第 123 页

"SUM 函数 [集合]" 第 220 页

CSCONVERT 函数 [字符串]

作用 在不同字符集之间转换字符串。

语法

CSCONVERT (*string-expression, 'target-charset'*
[*, 'source-charset'*] **)**

参数

string-expression 字符串。

target-charset 目标字符集。 *Target-charset* 可以是下列情况之一：

- **os_charset** 操作系统使用的字符集。
- **db_charset** 数据库使用的字符集。
- **任何其它受支持的字符集标签** 您可以指定 Adaptive Server Anywhere 支持的任何字符集标签。有关更多信息，请参阅 "ASA 数据库管理指南 > 国际语言和字符集 > 字符集标签 "。

source-charset 原字符串表达式使用的字符集。缺省设置是 *db_charset*。 *Source-charset-name* 可以是下列情况之一：

- **os_charset** 操作系统使用的字符集。
- **db_charset** 数据库使用的字符集。
- **任何其它受支持的字符集标签** 您可以指定 Adaptive Server Anywhere 支持的任何字符集标签。有关更多信息，请参阅 "ASA 数据库管理指南 > 国际语言和字符集 > 字符集标签 "。

示例 1 下面的代码段将 mytext 列从繁体中文字符集转换为简体中文字符集：

```
SELECT
   CSCONVERT (mytext, 'cp936', 'cp950')
   FROM mytable
```

示例 2 下面的代码段将 mytext 列从数据库字符集转换为简体中文字符集：

```
SELECT
   CSCONVERT (mytext, 'cp936')
   FROM mytable
```

示例 3 如果文件名存储在数据库中，则它是用数据库的字符集存储的。如果服务器从其名称存储在一个数据库中的文件 （例如，外部存储过程）中进行读取或向该文件写入，则必须将该文件名显式转换为操作系统的字符集，然后才能访问该文件。存储在数据库中的和由客户端检索的文件名会自动转换为客户端的字符集，因此不需要显式转换。

下面的代码段将 filename 列从数据库字符集转换为操作系统字符集：

```
SELECT
   CSCONVERT (filename, 'os_charset')
   FROM mytable
```

示例 4 有一个表包含一个文件名列表。一个外部存储过程将此表中的一个文件名用作参数，直接从该文件读取信息。在不需要字符集转换时可以使用下面的代码段：

```
SELECT
   MYFUNC( filename )
   FROM mytable
```

其中 mytable 是一个包含一个 filename 列的表。但是，如果您需要将 filename 转换为操作系统的字符集，则应使用下面的语句。

```
SELECT
   MYFUNC( csconvert( filename, 'os_charset' ) )
   FROM mytable
```

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"ASA 数据库管理指南 > 国际语言和字符集 > 启动使用字符集转换的数据库服务器 "

DATALENGTH 函数 [系统]

作用

返回表达式结果的基础存储的长度 （以字节为单位）。

语法

DATALENGTH (*expression* **)**

参数

expression 表达式通常是列名称。如果表达式是字符串常量，则它必须用引号括起来。

用法

DATALENGTH 的返回值如下：

数据类型	DATALENGTH
SMALLINT	2
INTEGER	4
DOUBLE	8
CHAR	数据的长度
BINARY	数据的长度

示例

下面的语句返回值 **27**，这是 *company_name* 列中最长的字符串。

```
SELECT following MAX( DATALENGTH( company_name ) )
FROM customer
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

DATE 函数 [日期和时间]

作用 将表达式转换为日期，并删除任何小时、分钟或秒。

语法 **DATE (** *expression* **)**

参数 **expression** 要转换成日期格式的值。表达式通常是字符串。

示例 下面的语句返回值 1999-01-02 作为日期。

```
SELECT DATE( '1999-01-02 21:20:53' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

DATEADD 函数 [日期和时间]

作用 返回通过将若干日期部分添加到日期中而产生的日期。

语法 **DATEADD (** *date-part*, *numeric-expression*, *date-expression* **)**

date-part :
year | quarter | month | week | day | hour | minute | second | millisecond

参数 **date-part** 要添加到日期中的日期部分。

有关日期部分的更多信息，请参阅 "" 日期部分 " 第 108 页 "。

numeric-expression 要添加到日期中的日期部分的数目。
numeric_expression 可以是任意数字类型，但它们的值将被截断成整数。

date-expression 要修改的日期。

示例

- 下面的语句返回值：1995-11-02 00:00:00.0。

```
SELECT dateadd( month, 102, '1987/05/02' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

DATEDIFF 函数 [日期和时间]

作用

返回两个日期之间的间隔。

语法

DATEDIFF (*date-part*, *date-expression1*, *date-expression2* **)**

date-part :
year | **quarter** | **month** | **week** | **day** | **hour** | **minute** | **second** | **millisecond**

参数

date-part 指定要测量其间隔的日期部分。

有关日期部分的更多信息，请参阅 "" 日期部分 " 第 108 页 "。

date-expression1 某一间隔的起始日期。从 *date-expression2* 中减去该值，返回两个参数之间 *date-parts* 的数目。

date-expression2 某一间隔的结束日期。从该值中减去 *Date-expression1*，返回两个参数之间 *date-parts* 的数目。

示例

下面的语句返回 1：

```
SELECT datediff( hour, '4:00AM', '5:50AM' )
```

下面的语句返回 102：

```
SELECT datediff( month, '1987/05/02', '1995/11/15' )
```

下面的语句返回 0：

```
SELECT datediff( day, '00:00', '23:59' )
```

下面的语句返回 4：

```
SELECT datediff( day, '1999/07/19 00:00', '1999/07/23
23:59' )
```

下面的语句返回 0：

```
SELECT datediff( month, '1999/07/19', '1999/07/23' )
```

下面的语句返回 1：

```
SELECT datediff( month, '1999/07/19', '1999/08/23' )
```

用法

此函数计算两个指定日期之间日期部分的数目。结果为日期部分中等于（date2 - date1）的有符号的整数值。

当结果不是日期部分的偶数倍时， DATEDIFF 将被截断而不是被舍入。

当使用 **day** 作为日期部分时， DATEDIFF 返回两个指定的时间之间（包括第二个日期但不包括第一个日期）的午夜数目。

当使用 **month** 作为日期部分时， DATEDIFF 返回两个日期之间 （包括第二个日期但不包括第一个日期）月中第一天的数目。

当使用 **week** 作为日期部分时， DATEDIFF 返回两个日期 （包括第二个日期但不包括第一个日期）之间星期日的数目。

对于更小的时间单位存在溢出值：

- **milliseconds** 24 天
- **seconds** 68 年
- **minutes** 4083 年
- **others** 没有溢出限制

如果超出这些限制，此函数将返回溢出错误。

标准和兼容性

- **SQL/92.** Transact-SQL 扩展。
- **SQL/99.** Transact-SQL 扩展。
- **Sybase.** 与 Adaptive Server Enterprise 兼容。

DATEFORMAT 函数 [日期和时间]

作用

以指定的格式返回表示日期表达式的字符串。

语法

DATEFORMAT (*datetime-expression*, *string-expression* **)**

参数

datetime-expression 要转换的日期时间。

string-expression 转换后的日期格式。

有关日期格式说明的信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > DATE_FORMAT 选项 [compatibility]"。

示例

下面的语句返回值 Jan 01, 1989。

```
SELECT DATEFORMAT( '1989-01-01', 'Mmm dd, yyyy' )
```

用法

任何允许的日期格式都可用于字符串表达式。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

千年虫问题

使用 DATEFORMAT 函数可能会产生一个仅用两位数表示年份的字符串。这会导致千年虫问题，即使没有发生任何错误。

有关千年虫问题的更多信息，请参阅 "2000 年兼容性问题 " 第 100 页。

另请参阅 "ASA 数据库管理指南 > 数据库选项 > DATE_FORMAT 选项 [compatibility]"

DATENAME 函数 [日期和时间]

作用 以字符串形式返回日期时间值中指定部分的名称 （如月份 "June"）。

语法 **DATENAME (** *date-part*, *date-expression* **)**

参数 **date-part** 要指定的日期部分。

有关允许的日期部分的完整列表，请参阅 "" 日期部分 " 第 108 页 "。

date-expression 要返回其日期部分名称的日期。日期必须包含请求的 *date-part*。

示例 下面的语句返回值 May。

```
SELECT datename( month , '1987/05/02' )
```

用法 DATENAME 返回一个字符串，即使结果是数字 （如表示日期的 23）。

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

DATEPART 函数 [日期和时间]

作用 返回日期时间值中一部分的值。

语法 **DATEPART (** *date-part*, *date-expression* **)**

参数 **date-part** 要返回的日期部分。

有关允许的日期部分的完整列表，请参阅 "" 日期部分 " 第 108 页 "。

date-expression 要返回其日期部分的日期。日期必须包含 *date-part* 字段。

示例 下面的语句返回值 5。

```
SELECT datepart( month , '1987/05/02' )
```

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

DATETIME 函数 [日期和时间]

作用 将表达式转换成时间戳。

语法 **DATETIME (** *expression* **)**

参数 **expression** 要转换的 *expression*。通常是字符串。试图转换数字值时会返回错误。

示例 下面的语句返回值为 1998-09-09 12:12:12.000 的时间戳。

```
SELECT DATETIME( '1998-09-09 12:12:12.000' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

DAY 函数 [日期和时间]

作用 返回 1 到 31 之间的一个整数，该整数对应于日期的每月的几号。

语法 **DAY (** *date-expression* **)**

参数 **date-expression** 日期。

示例 下面的语句返回值 12。

```
SELECT DAY( '2001-09-12' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

DAYNAME 函数 [日期和时间]

作用 返回日期中的星期几的名称。

语法 **DAYNAME(** *date-expression* **)**

参数 **date-expression** 日期。

示例 下面的语句返回值 Saturday。

```
SELECT DAYNAME ( '1987/05/02' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

DAYS 函数 [日期和时间]

作用 给定一个日期，此函数返回自 0000-02-29 以来的天数。

给定两个日期，此函数返回它们之间的整数天数。对于此用途，建议改用 "DATEDIFF 函数 [日期和时间]" 第 140 页。

给定一个日期和整数，此函数将整数天数添加到指定日期中。对于此用途，建议改用 "DATEADD 函数 [日期和时间]" 第 139 页。

语法 1 返回整数。语法 2 返回时间戳。

DAYS 忽略小时、分钟和秒。

语法 1 **DAYS (** [*datetime-expression*,] *datetime-expression* **)**

语法 2 **DAYS (** *datetime-expression*, *integer-expression* **)**

参数 **datetime-expression** 日期和时间。

integer-expression 要添加到 *datetime-expression* 的天数。如果 *integer-expression* 是负数，则从时间戳中减去相应的天数。如果提供整数表达式，必须将 *datetime-expression* 类型显式转换为日期或时间戳。

有关转换数据类型的信息，请参阅 ""CAST 函数 [数据类型转换]" 第 125 页 "。

示例 下面的语句返回整数 729 889。

```
SELECT DAYS( '1998-07-13 06:07:12' )
```

下面的语句返回当前儒略日。

```
SELECT DAYS( CURRENT DATE ) + 1721119
```

下列语句返回整数值 -366，表示第二个日期比第一个日期早 366 天。建议使用第二个示例 (DATEDIFF)。

```
SELECT DAYS( '1998-07-13 06:07:12',
             '1997-07-12 10:07:12' )
SELECT DATEDIFF( day,
   '1998-07-13 06:07:12',
   '1997-07-12 10:07:12' )
```

下列语句返回时间戳 1999-07-14 00:00:00.0。建议使用第二个示例 (DATEADD)。

```
SELECT DAYS( CAST('1998-07-13' AS DATE ), 366 )
SELECT DATEADD( day, 366, '1998-07-13' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

DB_ID 函数 [系统]

作用 返回数据库 ID 号。

语法 **DB_ID (** [*database-name*] **)**

参数 **database-name** 包含数据库名的字符串。如果未提供 *database-name*，则返回当前数据库的 ID 号。

示例 如果 *asademo* 是唯一正在运行的数据库，则下面的语句返回值 0：

```
SELECT DB_ID( 'asademo' )
```

对唯一正在运行的数据库执行时，下面的语句返回值 0。

```
SELECT DB_ID()
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

DB_NAME 函数 [系统]

作用 返回具有给定 ID 号的数据库的名称。

语法 **DB_NAME (** [*database-id*] **)**

参数 **database-id** 数据库的 ID。 *database-id* 必须是数字表达式。

示例 对作为服务器上的唯一数据库的示例数据库执行时，该语句返回数据库名 *asademo*。

```
SELECT DB_NAME( 0 )
```

用法 如果未提供数据库 ID，则返回当前数据库的名称。

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

DB_PROPERTY 函数 [系统]

作用 返回给定属性的值。

语法

DB_PROPERTY ({ *property_id* | *property_name* }
… [, { *database_id* | *database_name* }] **)**

参数

property_id 数据库属性 ID。

property_name 数据库属性名称。

database_id 数据库 ID 号，由 DB_ID 返回。通常使用数据库名。

database_name 数据库的名称，由 DB_NAME 返回。

示例 下面的语句返回当前数据库的页大小 （以字节为单位）。

```
SELECT DB_PROPERTY( 'PAGESIZE' )
```

用法 返回字符串。如果省略第二个参数，则使用当前数据库。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"DB_ID 函数 [系统]" 第 147 页

"DB_NAME 函数 [系统]" 第 147 页

"ASA 数据库管理指南 > 数据库性能和连接属性 > 数据库级属性 "

DEGREES 函数 [数字]

作用

- 将数字从弧度转换为度数。

语法 **DEGREES (** *numeric-expression* **)**

参数 **numeric-expression** 以弧度表示的角度。

示例 下面的语句返回值 29.793805。

```
SELECT DEGREES( 0.52 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

DIFFERENCE 函数 [字符串]

作用 返回两个字符串表达式之间 SOUNDEX 值的差。

语法 **DIFFERENCE (** *string-expression-1*, *string-expression-2* **)**

参数 **string-expression-1** 第一个 SOUNDEX 参数。

string-expression-2 第二个 SOUNDEX 参数。

示例 下面的语句返回值 3。

```
SELECT DIFFERENCE( 'test', 'chest' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "SOUNDEX 函数 [字符串]" 第 213 页

DOW 函数 [日期和时间]

作用 返回 1 到 7 之间的一个数字，表示日期中的星期几，如星期日 =1、星期一 =2，依此类推。

语法 **DOW (** *date-expression* **)**

参数 **date- expression** 日期。

示例 下面的语句返回值 5。

```
SELECT DOW( '1998-07-09' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

ERRORMSG 函数 [杂类]

作用 提供当前错误或者指定的 SQLSTATE 或 SQLCODE 值的错误消息。

语法 **ERRORMSG (** [*sqlstate* | *sqlcode*] **)**

sqlstate: string

sqlcode: integer

参数 **sqlstate** 要返回其错误消息的 SQLSTATE 值。

sqlcode 要返回其错误消息的 SQLCODE 值。

返回值 包含错误消息的字符串。如果未提供参数，则提供当前状态的错误消息。进行任何替代，如表名和列名。

如果提供了参数，则返回提供的 SQLSTATE 或 SQLCODE 的错误消息，不进行替代。表名和列名以占位符形式 (%1) 提供。

示例 下面的语句返回 SQLCODE -813 的错误消息。

```
select errormsg( -813 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ASA 错误消息 > 数据库错误消息 > 错误消息按 SQLSTATE 进行索引 "

"ASA 错误消息 > 数据库错误消息 > 错误消息按 Adaptive Server Anywhere SQLCODE 进行索引 "

ESTIMATE 函数 [杂类]

作用

根据指定的参数，为查询优化程序提供选择性估计。

语法

ESTIMATE (*column-name* [, *value* [, *relation-string*]] **)**

参数

column-name 在估计中使用的列。

value 要与列进行比较的值。

relation-string 用于比较的比较运算符，用单引号括起来；缺省值为 '='。

示例

下面的语句返回估计大于 200 的 *emp_id* 值的百分比。精度值取决于您对数据库执行的操作。

```
SELECT FIRST ESTIMATE( emp_id, 200, '>' )
   FROM employee
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"INDEX_ESTIMATE 函数 [杂类]" 第 168 页

"ESTIMATE_SOURCE 函数 [杂类]" 第 152 页

ESTIMATE_SOURCE 函数 [杂类]

作用 提供查询优化程序使用的选择性估计源。

语法 **ESTIMATE_SOURCE (** *column-name* [, *value* [, *relation-string*]] **)**

参数 **column-name** 要调查的列的名称。

value 要与列进行比较的值。这是可选的。

relation-string 用于比较的比较运算符，用单引号括起来。缺省值为相等运算符 (=)。

返回值 选择性估计源可以是下列情况之一：

- **Statistics** 。当您指定了一个值，并且有可用来估计列中的值的平均选择性的存储统计时，使用此来源。仅当值的选择性是统计中存储的有效数字时，统计才可用。目前，如果值至少出现在 1% 的行中，则被认为是有效的。
- **Column** 。类似于 Statistics，不同的是值的选择性出现在不到 1% 的行中。这种情况下，使用的选择性是统计中存储的、出现在不到 1% 的行中的所有值的平均值。
- **Guess** 。当没有相关的索引可以使用，并且没有为列收集统计时，返回此来源。这种情况下，使用内部推测。
- **Column-column** 。当使用的估计是连接的选择性时，返回此来源。这种情况下，估计等于连接结果集中的行数除以两个表的笛卡尔乘积中的行数。
- **Index** 。当没有可用于估计选择性的统计，但存在可供探测以估计选择性的索引时，使用此来源。
- **User** 。当存在用户提供的估计而且 USER_ESTIMATES 数据库选项未设置为 DISABLED 时，返回此来源。

  有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > USER_ESTIMATES 选项 [database]"。

- **Computed** 。当优化程序根据其它信息计算统计时，返回此来源。例如， Adaptive Server Anywhere 不维护多个列的统计，因此，如果需要多个列等式（如 x=5 和 y=10）的估计，并且存在列 x 和 y 的统计，则优化程序通过将每一列的估计选择性相乘来创建估计。
- **Always** 。当测试根据定义为真时，使用此来源。例如，如果值是 1=1。
- **Combined** 。当优化程序使用上述多个来源并且合并使用时，使用此来源。
- **Bounded** 。它可以限定其它来源之一。这表示 Adaptive Server Anywhere 已对估计设置了上限和 / 或下限。优化程序这样做是为了将估计保持在逻辑范围内。例如，它确保估计不大于 100%，或者选择性不小于一行。

示例

下面的语句返回值 Index，这表示查询优化程序已探测索引来估计选择性。

```
SELECT FIRST ESTIMATE_SOURCE( emp_id, 200, '>' )
FROM employee
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ESTIMATE 函数 [杂类]" 第 151 页

"INDEX_ESTIMATE 函数 [杂类]" 第 168 页

EVENT_CONDITION 函数 [系统]

作用

指定何时触发事件处理程序。

语法

EVENT_CONDITION (*condition-name* **)**

参数

condition-name 触发事件的条件。可能的值已经在数据库中预设，并且不区分大小写。每个条件仅对某些特定的事件类型有效。下表列出了一些条件以及它们所适用的事件：

条件名称	单位	适用于 …	注释
DBFreePercent	无	DBDiskSpace	
DBFreeSpace	MB	DBDiskSpace	
DBSize	MB	GrowDB	
ErrorNumber	无	RAISERROR	
IdleTime	秒	ServerIdle	
Interval	秒	全部	自上次执行处理程序以来的时间
LogFreePercent	无	LogDiskSpace	
LogFreeSpace	MB	LogDiskSpace	
LogSize	MB	GrowLog	
RemainingValues	整数	GlobalAutoincrement	剩余值的数目
TempFreePercent	无	TempDiskSpace	
TempFreeSpace	MB	TempDiskSpace	
TempSize	MB	GrowTemp	

示例

下列事件定义使用了 **event_condition** 函数：

```
create event LogNotifier
type LogDiskSpace
where event_condition( 'LogFreePercent' ) < 50
handler
begin
   message 'LogNotifier message'
end
```

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "CREATE EVENT 语句 " 第 326 页

EVENT_CONDITION_NAME 函数 [系统]

作用 可用于列出 EVENT_CONDITION 的可能的参数。

语法 **EVENT_CONDITION_NAME (** *integer* **)**

参数 **integer** 必须大于或等于零。

用法 可以使用 EVENT_CONDITION_NAME 获得所有 EVENT_CONDITION 参数的列表，方法是在整数中循环，直到函数返回 NULL。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "CREATE EVENT 语句 " 第 326 页

EVENT_PARAMETER 函数 [系统]

作用 提供事件处理程序的上下文信息。

语法 **EVENT_PARAMETER (** *context-name* **)**

context-name:
'ConnectionID'
| **'User'**
| **'EventName'**
| **'Executions'**
| **'NumActive'**
| **'TableName'**
| *condition-name*

参数

context-name 其中一个预设字符串。这些字符串不区分大小写，它们传递下列信息：

- **ConnectionId** 连接 ID，由 connection_property('id') 返回
- **User** 导致事件被触发的用户的用户 ID。
- **EventName** 已触发的事件的名称。
- **Executions** 事件处理程序执行的次数。
- **NumActive** 事件处理程序的活动实例数。如果想限制事件处理程序以便在任何给定的时间仅执行一个实例，则这很有用。
- **TableName** 表名，与 RemainingValues 一起使用。

此外，可以从 EVENT_PARAMETER 函数访问 EVENT_CONDITION 函数的任何有效的 *condition-name* 参数。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"EVENT_CONDITION 函数 [系统]" 第 153 页

"CREATE EVENT 语句 " 第 326 页

EXP 函数 [数字]

作用

返回指数函数，即 e 的指定数字次乘方。

语法

EXP (*numeric-expression* **)**

参数

numeric-expression 指数。

示例

下面的语句返回值 3269017.372。

```
SELECT EXP( 15 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

EXPERIENCE_ESTIMATE 函数 [杂类]

作用 此函数与 ESTIMATE 函数相同，不过它总是查询频率表。

语法 **EXPERIENCE_ESTIMATE (** *column-name* [, *value* [, *relation-string*]] **)**

参数 **column-name** 要调查的列的名称。

value 要与列进行比较的值。

relation-string 用于比较的比较运算符，用单引号括起来；缺省值为 '='。

示例 下面的语句返回 NULL。

```
SELECT DISTINCT EXPERIENCE_ESTIMATE( emp_id, 200, '>' )
FROM employee
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "ESTIMATE 函数 [杂类]" 第 151 页

EXPLANATION 函数 [杂类]

作用 以字符串的形式返回 SQL 语句的简单计划优化策略。

语法 **EXPLANATION (** *string-expression* [*cursor-type*], [*update-status*] **)**

参数

string-expression SQL 语句，通常是 SELECT 语句，但也可以是 UPDATE 或 DELETE。

cursor-type 字符串。**Cursor-type** 可以是 **asensitive**（缺省）、**insensitive**、**sensitive** 或 **keyset-driven**。

update-status 一个字符串参数，它接受下列值之一，这些值指示优化程序如何处理给定的游标：

值	说明
READ-ONLY	游标是只读的。
READ-WRITE（缺省）	可以读取游标，也可以向游标写入。
FOR UPDATE	可以读取游标，也可以向游标写入。这与 READ-WRITE 完全相同。

示例

下面的语句将 SELECT 语句作为字符串参数传递，并返回查询的执行计划。

```
SELECT EXPLANATION( 'SELECT * FROM department WHERE
dept_id > 100' )
```

这些信息可以帮助决定要添加哪些索引或如何构建数据库，以获得更好的性能。

下面的语句返回一个字符串，该字符串包含基于 'select * from department where ….' 语句的 INSENSITIVE 游标的文本计划的短格式。

```
SELECT EXPLANATION( 'SELECT * FROM department WHERE
dept_id > 100', 'insensitive', 'read-only' )
```

在 Interactive SQL 中，可以在 " 结果 " 窗格中的 " 计划 " 选项卡上查看任意 SQL 语句的计划。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"PLAN 函数 [杂类]" 第 193 页

"GRAPHICAL_PLAN 函数 [杂类]" 第 161 页

"GRAPHICAL_ULPLAN 函数 [杂类]" 第 163 页

"LONG_ULPLAN 函数 [杂类]" 第 177 页

"SHORT_ULPLAN 函数 [杂类]" 第 207 页

FLOOR 函数 [数字]

作用 返回一个数字的下限 （不大于的最大整数）。

语法 **FLOOR (** *numeric-expression* **)**

参数 **numeric- expression** 数字，通常是浮点数。

示例

值	下限 （值）
123	123
123.45	123
-123.45	-124

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"CEILING 函数 [数字]" 第 126 页

GET_IDENTITY 函数 [杂类]

作用 将值分配给自动增量列。这可以作为使用自动增量生成编号的替代方法。

语法

GET_IDENTITY ([*owner*.] *table-name* [, *num_to_alloc*],… **)**

参数

num_to_allocate 缺省值为 1。

示例

下面的语句调用三次 GET_IDENTITY 函数：

```
SELECT GET_IDENTITY('T1'),
   GET_IDENTITY('T2',10),
   GET_IDENTITY('T3',5)
```

用法

使用自动增量或全局自动增量仍是生成 ID 的最有效方法，但提供了此函数作为替代方法。此函数假设表定义了自动增量列。缺省情况下，它返回为表的自动增量列生成的下一个可用值，并保留该值以便没有其它连接使用它。

如果未找到表，此函数返回错误，如果表没有自动增量列，则返回 NULL。如果存在多个自动增量列，此函数使用它找到的第一个自动增量列。

如果 *num_to_alloc* 大于 1，此函数还保留剩余值。下一次分配将使用当前数字加上 *num_to_alloc* 的值。这使应用程序可以不用频繁地执行 *get_identity*。

执行 *get_identity* 之后不需要 COMMIT，所以可以使用用于插入行的同一连接调用 *get_identity*。如果需要几个表的 ID 值，可以在一个 SELECT 中包括多个 *get_identity* 调用 （如上例所示）以获取这些值。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"CREATE TABLE 语句 " 第 395 页

"ALTER TABLE 语句 " 第 272 页

"NUMBER 函数 [杂类]" 第 190 页

GETDATE 函数 [日期和时间]

作用　返回当前日期和时间。

语法　**GETDATE ()**

示例　下面的语句返回系统日期和时间。

```
SELECT GETDATE( )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

GRAPHICAL_PLAN 函数 [杂类]

作用　以字符串的形式返回 XML 格式的 SQL 语句的计划优化策略。

语法　**GRAPHICAL_PLAN (** *string-expression* [*statistics-level*], [*cursor-type*], [*update-status*] **)**

参数　**string-expression**　SQL 语句，通常是 SELECT 语句，但也可以是 UPDATE 或 DELETE。

statistics-level　一个整数。 *Statistics-level* 可以是下列值之一：

值	说明
0	仅包括优化程序估计。（缺省值）
1	包括从执行中获得的实际摘要统计信息。
2	包括详细的实际统计信息。

cursor-type　字符串。 **Cursor-type** 可以是 **asensitive**（缺省）、**insensitive**、 **sensitive** 或 **keyset-driven**。

update-status 一个字符串参数，它可接受下列值之一，这些值指示优化程序如何处理给定的游标：

值	说明
READ-ONLY	游标是只读的。
READ-WRITE（缺省）	可以读取游标，也可以向游标写入。
FOR UPDATE	可以读取游标，也可以向游标写入。它与 READ-WRITE 完全相同。

示例

下面的 Interactive SQL 示例将 SELECT 语句作为字符串参数传递，并返回查询的执行计划。它将计划保存在 `plan.xml` 文件中。

```
SELECT GRAPHICAL_PLAN( 'SELECT * FROM department WHERE
dept_id > 100' );
OUTPUT TO plan.xml FORMAT FIXED
```

下面的语句返回一个字符串，该字符串包含基于 'SELECT * FROM department WHERE ….' 查询的键集驱动的可更新游标的图形计划。通过 'SELECT * FROM department WHERE ….' 这一查询返回一个字符串，该字符串包含一个键集驱动的可更新游标的图形计划。它还使服务器利用实际的执行统计信息连同优化程序使用的估计统计信息为计划加批注。

```
SELECT GRAPHICAL_PLAN( 'SELECT * FROM department WHERE
dept_id > 100', 2, 'keyset-driven', 'for update' )
```

在 Interactive SQL 中，可以在 " 结果 " 窗格中的 " 计划 " 选项卡上查看任意 SQL 语句的计划。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"PLAN 函数 [杂类]" 第 193 页

"EXPLANATION 函数 [杂类]" 第 157 页

"GRAPHICAL_ULPLAN 函数 [杂类]" 第 163 页

"LONG_ULPLAN 函数 [杂类]" 第 177 页

"SHORT_ULPLAN 函数 [杂类]" 第 207 页

GRAPHICAL_ULPLAN 函数 [杂类]

作用

以字符串的形式返回 XML 格式的 SQL 语句的 UltraLite 计划优化策略。UltraLite 计划不包括统计信息。

对于某些查询， UltraLite 执行计划可能不同于为 Adaptive Server Anywhere 选择的计划。

语法

GRAPHICAL_ULPLAN (*string-expression* **)**

参数

string-expression SQL 语句，通常是 SELECT 语句，但也可以是 UPDATE 或 DELETE。

示例

下面的 Interactive SQL 示例将 SELECT 语句作为字符串参数传递，并返回查询的执行计划。它将计划保存在 `plan.xml` 文件中。

```
SELECT GRAPHICAL_ULPLAN( 'select * from department
where dept_id > 100' );
OUTPUT TO ulplan.xml FORMAT FIXED
```

若要显示该计划，请在 Interactive SQL 中打开 `ulplan.xml` 文件。

另一种方法是，可以在 Interactive SQL 的 "UltraLite 计划 " 选项卡上查看任意 SQL 语句的计划：选择 " 文件 " → " 保存 " 并将文件类型更改为 xml。若要更改显示的计划的类型，请选择"工具"→"选项"并打开"计划"选项卡。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"PLAN 函数 [杂类]" 第 193 页

"EXPLANATION 函数 [杂类]" 第 157 页

"GRAPHICAL_PLAN 函数 [杂类]" 第 161 页

"LONG_ULPLAN 函数 [杂类]" 第 177 页

"SHORT_ULPLAN 函数 [杂类]" 第 207 页

GREATER 函数 [杂类]

作用 返回两个参数值中较大的那个。如果两个参数相等，则返回第一个。

语法 **GREATER (** *expression1*, *expression2* **)**

示例 下面的语句返回值 10。

```
SELECT GREATER( 10,5 ) FROM dummy
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "LESSER 函数 [杂类]" 第 174 页

HEXTOINT 函数 [数据类型转换]

作用 返回十六进制字符串的等效十进制整数。

语法 **HEXTOINT (** *hexadecimal-string* **)**

参数 **hexadecimal-string** 要转换成整数的字符串。

示例 下面的语句返回值 420。

```
SELECT HEXTOINT ( '1A4' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅　"INTTOHEX 函数 [数据类型转换]" 第 169 页

HOUR 函数 [日期和时间]

作用　返回 0 到 23 之间的一个数字，该数字对应于日期时间的小时部分。

语法　**HOUR (** *datetime-expression* **)**

参数　**datetime-expression**　日期时间。

示例　下面的语句返回值 21：

```
SELECT HOUR( '1998-07-09 21:12:13' )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

HOURS 函数 [日期和时间]

作用　给定两个时间戳，此函数返回两个时间戳之间的整数小时数。对于此用途，建议改用 "DATEDIFF 函数 [日期和时间]" 第 140 页。

给定一个日期，此函数返回自 0000-02-29 00:00:00 以来的小时数。

给定一个日期和整数，此函数将整数小时数添加到指定的时间戳中。对于此用途，建议改用 "DATEADD 函数 [日期和时间]" 第 139 页。

语法 1 返回整数。语法 2 返回时间戳。

语法 1　**HOURS (** [*datetime-expression*,] *datetime-expression* **)**

语法 2　**HOURS (** *datetime-expression*, *integer-expression* **)**

参数　**datetime-expression**　日期和时间。

integer-expression 要添加到 *datetime-expression* 中的小时数。如果 *integer-expression* 是负数，则从日期时间中减去相应的小时数。如果提供整数表达式，必须将 *datetime-expression* 类型显式转换为日期时间数据类型。

有关转换数据类型的信息，请参阅 ""CAST 函数 [数据类型转换]" 第 125 页 "。

示例

下列语句返回值 4，表示第二个时间戳比第一个时间戳晚四个小时。建议使用第二个示例 (DATEDIFF)。

```
SELECT HOURS( '1999-07-13 06:07:12',
   '1999-07-13 10:07:12' )
SELECT DATEDIFF( hour,
   '1999-07-13 06:07:12',
   '1999-07-13 10:07:12' )
```

下面的语句返回值 17,517,342。

```
SELECT HOURS( '1998-07-13 06:07:12' )
```

下列语句返回日期时间 1999-05-13 02:05:07.0。建议使用第二个示例 (DATEADD)。

```
SELECT HOURS(
   CAST( '1999-05-12 21:05:07' AS DATETIME ), 5)
SELECT DATEADD( hour, 5, '1999-05-12 21:05:07' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

IDENTITY 函数 [杂类]

作用

为查询中的每个连续行生成从 1 开始的整数值。它的实现方式与 NUMBER 函数相同。

语法

IDENTITY (*expression* **)**

参数 **expression** 表达式。对表达式进行分析，但在执行此函数时忽略表达式。

示例 下面的语句返回按顺序排列的雇员列表。

```
SELECT IDENTITY( 10 ), emp_lname FROM employee
```

用法 有关如何使用 IDENTITY 函数的说明，请参阅 ""NUMBER 函数 [杂类]" 第 190 页 "。

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 其行为类似于 Adaptive Server Enterprise 的行为。

另请参阅 "NUMBER 函数 [杂类]" 第 190 页

IFNULL 函数 [杂类]

作用 如果第一个表达式为 NULL 值，则返回第二个表达式的值。如果第一个表达式不为 NULL，则返回第三个表达式的值。如果第一个表达式不为 NULL，并且没有第三个表达式，则返回 NULL。

语法 **IFNULL (** *expression-1*, *expression-2* [, *expression-3*] **)**

参数 **expression-1** 要计算其值的表达式。它的值将确定返回 expression-2 还是 expression-3。

expression-2 返回值 （如果 *expression-1* 为 NULL）。

expression-3 返回值 （如果 *expression-1* 不为 NULL）。

示例 下面的语句返回值 -66：

```
SELECT IFNULL( NULL, -66 )
```

下面的语句返回 NULL，因为第一个表达式不为 NULL，并且没有第三个表达式：

```
SELECT IFNULL( -66, -66 )
```

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

INDEX_ESTIMATE 函数 [杂类]

作用

此函数与 ESTIMATE 函数相同，不过它总是仅查询索引。

语法

INDEX_ESTIMATE(*column-name*, *number* [, *relation-string*] **)**

参数

column-name 在估计中使用的列的名称。

number 如果指定了 *number*，则此函数以 REAL 形式返回查询优化程序使用的百分比估计。

relation-string *relation-string* 必须是用单引号括起来的比较运算符；缺省值为 '='。

示例

下面的语句返回值 81.304607。

```
SELECT FIRST ESTIMATE( emp_id, 200, '>' )
FROM employee
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ESTIMATE 函数 [杂类]" 第 151 页

"ESTIMATE_SOURCE 函数 [杂类]" 第 152 页

INSERTSTR 函数 [字符串]

作用

将一个字符串插入到另一个字符串中的指定位置。

语法

INSERTSTR (*integer-expression*, *string-expression-1*, *string-expression-2* **)**

参数 **integer expression** 要在其后插入字符串的位置。使用零可将字符串插入到开头。

string-expression-1 要在其中插入其它字符串的字符串。

string-expression-2 要插入的字符串。

示例 下面的语句返回值 backoffice。

```
SELECT INSERTSTR( 0, 'office ', 'back' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "STUFF 函数 [字符串]" 第 218 页

INTTOHEX 函数 [数据类型转换]

作用 返回包含整数的等效十六进制值的字符串。

语法 **INTTOHEX (** *integer-expression* **)**

参数 **integer expression** 要转换成十六进制的整数。

示例 下面的语句返回值 9c：

```
SELECT INTTOHEX( 156 )
```

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "HEXTOINT 函数 [数据类型转换]" 第 164 页

ISDATE 函数 [数据类型转换]

作用

测试字符串参数是否可以转换为日期。如果可以转换，则此函数返回 1；否则返回 0。如果参数为空，则返回 0。

语法

ISDATE (*string* **)**

示例

下面的示例从外部文件导入数据，导出包含无效值的行，并将其余的行复制到永久表中。

```
create global temporary table MyData(
      person         varchar(100),
      birth_date     varchar(30),
      height_in_cms  varchar(10)
   ) on commit preserve rows;
   load table MyData from 'exported.dat';
   unload
      select *
      from MyData
      where isdate(birth_date)=0 or
isnumeric(height_in_cms)=0
   to 'badrows.dat';
   insert into PermData
      select person,birthdate,height_in_cms
      from MyData
      where isdate(birth_date)=1 and
isnumeric(height_in_cms)=1;
   commit;
   drop table MyData;
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

ISNULL 函数 [数据类型转换]

作用

- 返回参数列表中的第一个非 NULL 表达式。

语法

ISNULL (*expression*, *expression* [, …] **)**

参数 **expression** 要测试是否为 NULL 的表达式。

必须给此函数至少传递两个表达式。

示例 下面的语句返回值 -66。

```
SELECT ISNULL( NULL ,-66, 55, 45, NULL, 16 )
```

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容，但 Adaptive Server Enterprise 只允许两个表达式。

另请参阅 "COALESCE 函数 [杂类]" 第 129 页

ISNUMERIC 函数 [杂类]

作用 测试字符串参数是否可以转换为数字。如果可以转换，则此函数返回 1；否则返回 0。如果参数为空，则返回 0。

语法 **ISNUMERIC (** *string* **)**

示例 下面的示例从外部文件导入数据，导出包含无效值的行，并将其余的行复制到永久表中。

```
create global temporary table MyData(
      person         varchar(100),
      birth_date     varchar(30),
      height_in_cms  varchar(10)
   ) on commit preserve rows;
   load table MyData from 'exported.dat';
   unload
      select *
      from MyData
      where isdate(birth_date)=0 or
isnumeric(height_in_cms)=0
   to 'badrows.dat';
   insert into PermData
      select person,birthdate,height_in_cms
```

```
      from MyData
      where isdate(birth_date)=1 and
isnumeric(height_in_cms)=1;
   commit;
   drop table MyData;
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

LCASE 函数 [字符串]

作用 将字符串中的所有字符转换成小写字母。

语法 **LCASE (** *string-expression* **)**

参数 **string-expression** 要转换成小写字母的字符串。

示例 下面的语句返回值 lower case。

```
SELECT LCASE( 'LOWER CasE' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Enterprise 不支持 LCASE； 使用 LOWER 可以获得同样的功能。

另请参阅

"LOWER 函数 [字符串]" 第 178 页

"UCASE 函数 [字符串]" 第 226 页

"UPPER 函数 [字符串]" 第 226 页

LEFT 函数 [字符串]

作用 从字符串开始处返回一定数量的字符。

语法 **LEFT (** *string-expression*, *integer-expression* **)**

参数 **string-expression** 字符串。

integer expression 要返回的字符数。

示例 下面的语句返回值 choco。

```
SELECT LEFT( 'chocolate', 5 )
```

用法 如果字符串包含多字节字符，并且使用了适当的归类，则返回的字节数可能大于指定的字符数。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "RIGHT 函数 [字符串]" 第 204 页

"ASA 数据库管理指南 > 国际语言和字符集 "

LENGTH 函数 [字符串]

作用 返回指定字符串中的字符数。

语法 **LENGTH (** *string-expression* **)**

参数 **string-expression** 字符串。

示例 下面的语句返回值 9。

```
SELECT LENGTH( 'chocolate' )
```

用法 如果字符串包含多字节字符，并且使用了适当的归类，则 LENGTH 返回字符数而不是字节数。如果字符串是 BINARY 数据类型，则 LENGTH 函数的行为与 BYTE_LENGTH 的行为相同。

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "BYTE_LENGTH 函数 [字符串]" 第 124 页

"ASA 数据库管理指南 > 国际语言和字符集 "

LESSER 函数 [杂类]

作用 返回两个参数值中较小的那个。如果两个参数相等，则返回第一个。

语法 **LESSER (** *expression1*, *expression2* **)**

示例 下面的语句返回值 5。

```
SELECT LESSER( 10,5 ) FROM dummy
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "GREATER 函数 [杂类]" 第 164 页

LIST 函数 [集合]

作用 返回以逗号分隔的值列表

语法 **LIST (** { *string-expression* | **DISTINCT** *column-name* } [, *delimiter-string*] **)**

参数 **string-expression** 字符串，通常是列名称。对于每一行，表达式的值被添加到逗号分隔的结果中。

DISTINCT column-name 在查询中使用的列的名称。对于该列中的每个唯一值，该值被添加到逗号分隔的结果中。

delimiter-string 此可选参数指定列表项的分隔符字符串。缺省设置是逗号。如果提供 NULL 值或空字符串，则没有分隔符。 *delimiter-string* 应该是常量。

示例

下面的语句返回值 48 Kennedy Court, 54 School Street。

```
SELECT LIST( street ) FROM employee
WHERE emp_fname = 'Thomas'
```

用法

NULL 值不添加到列表中。 List(X) 返回组中每一行的 X 的所有非 NULL 值的连接 （带分隔符）。如果组中没有任何一行具有明确的 X 值，则 LIST(X) 返回空字符串。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

LOCATE 函数 [字符串]

作用

返回一个字符串在另一个字符串中的位置。

语法

LOCATE (*string-expression-1*, *string-expression-2* [, *integer-expression*] **)**

参数

string-expression-1 要在其中进行搜索的字符串。

string-expression-2 要搜索的字符串。该字符串的长度不应超过 255 个字节。

integer-expression 字符串中搜索的起始字符位置。第一个字符在位置 1。如果起始偏移是负值，则定位函数返回最后一个匹配字符串偏移而非第一个。负的偏移指示从搜索中排除字符串尾的多长一部分。排除的字节数计算为 （-1 × 偏移）-1。

示例

下面的语句返回值 8。

```
SELECT LOCATE( 'office party this week - rsvp as soon
as possible', 'party', 2 )
```

下面的语句：

```
BEGIN
   declare str long varchar;
   declare pos int;
   set str = 'c:\test\functions\locate.sql';
   set pos = locate( str, '\', -1 );
   select str, pos,
      substr( str, 1, pos -1 ) as path,
      substr( str, pos +1 ) as filename;
END
```

返回下面的输出：

strpospathfilename c:\test\functions\locate.sql18c:\test\functionslocate.sql

用法

如果指定了 *integer-expression*，则在字符串中的该偏移处开始搜索。

第一个字符串可以是长字符串 （长于 255 个字节），但第二个字符串的长度不应超过 255 个字节。如果第二个参数是长字符串，此函数返回 NULL 值。如果未找到字符串，则返回 0。搜索零长度字符串将返回 1。如果有某个参数为 NULL，则结果为 NULL。

如果使用多字节字符，并具有适当的归类，则起始位置和返回值可能不同于*字节* 位置。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

LOG 函数 [数字]

作用

- 返回一个数字的自然对数。

语法

LOG (*numeric-expression* **)**

参数

numeric-expression 数字。

示例

下面的语句返回值 3.912023。

```
SELECT LOG( 50 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "LOG10 函数 [数字]" 第 177 页

LOG10 函数 [数字]

作用

- 返回一个数字的以 10 为底的对数。

语法 **LOG10 (** *numeric-expression* **)**

参数 **numeric-expression** 数字。

示例 下面的语句返回值 1.698970。

```
SELECT LOG10( 50 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "LOG 函数 [数字]" 第 176 页

LONG_ULPLAN 函数 [杂类]

作用 以字符串形式返回 SQL 语句的 UltraLite 计划优化策略的长说明。此说明与 PLAN 函数返回的说明相同。

对于某些查询， UltraLite 执行计划可能不同于为 Adaptive Server Anywhere 选择的计划。

语法 **LONG_ULPLAN (** *string-expression* **)**

参数 **string-expression** SQL 语句，通常是 SELECT 语句，但也可以是 UPDATE 或 DELETE。

示例 下面的语句将 SELECT 语句作为字符串参数传递，并返回查询的执行计划。

```
SELECT LONG_ULPLAN( 'select * from department where
dept_id > 100' )
```

该信息可以帮助决定要添加的索引或如何构建数据库，以获得更好的性能。

在 Interactive SQL 中，可以在 " 结果 " 窗格中的 "UltraLite 计划 " 选项卡上查看任意 SQL 语句的计划。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"PLAN 函数 [杂类]" 第 193 页

"EXPLANATION 函数 [杂类]" 第 157 页

"GRAPHICAL_PLAN 函数 [杂类]" 第 161 页

"GRAPHICAL_ULPLAN 函数 [杂类]" 第 163 页

"SHORT_ULPLAN 函数 [杂类]" 第 207 页

LOWER 函数 [字符串]

作用 将字符串中的所有字符转换成小写字母。

语法 **LOWER (** *string-expression* **)**

参数 **string-expression** 要转换的字符串。

示例 下面的语句返回值 lower case。

```
SELECT LOWER( 'LOWER CasE' )
```

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"LCASE 函数 [字符串]" 第 172 页

"UCASE 函数 [字符串]" 第 226 页

"UPPER 函数 [字符串]" 第 226 页

LTRIM 函数 [字符串]

作用 剪裁字符串中的前导空白。

语法 **LTRIM (** *string-expression* **)**

参数 **string-expression** 要剪裁的字符串。

示例 下面的语句返回值 Test Message，其中的所有前导空白都已删除。

```
SELECT LTRIM( '     Test Message' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"RTRIM 函数 [字符串]" 第 205 页

"TRIM 函数 [字符串]" 第 223 页

MAX 函数 [集合]

作用 返回在每一组行中找到的最大 *expression* 值。

语法

MAX (*expression*
| DISTINCT *column name* **)**

参数

expression 要计算其最大值的表达式。这通常是列名称。

DISTINCT column-name 与 MAX(*expression*) 返回相同的值，为了完整起见，才将它列了出来。

示例

下面的语句返回值 138948.000，该值表示雇员表中的最高薪水。

```
SELECT MAX( salary )
FROM employee
```

用法

忽略 *expression* 为 NULL 的行。对于不包含任何行的组返回 NULL。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"MIN 函数 [集合]" 第 180 页

MIN 函数 [集合]

作用

返回在每一组行中找到的最小表达式值。

语法

MIN (*expression*
| DISTINCT *column name* **)**

参数

expression 要计算其最小值的表达式。这通常是列名称。

DISTINCT column-name 与 MIN(*expression*) 返回相同的值，为了完整起见，才将它列了出来。

示例

下面的语句返回值 24903.000，该值表示雇员表中的最低薪水。

```
SELECT MIN( salary )
FROM employee
```

用法

忽略 *expression* 为 NULL 的行。对于不包含任何行的组返回 NULL。

标准和兼容性

- **SQL/92**　与 SQL/92 兼容。
- **SQL/99**　核心特性。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅　"MAX 函数 [集合]" 第 179 页

MINUTE 函数 [日期和时间]

作用　返回 0 到 59 之间的一个数字，该数字对应于日期时间值的分钟部分。

语法　**MINUTE (** *datetime-expression* **)**

参数　**datetime-expression**　日期时间值。

示例　下面的语句返回值 22。

```
SELECT MINUTE( '1998-07-13 12:22:34' )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

MINUTES 函数 [日期和时间]

作用　给定两个时间戳，此函数返回二者之间的整数分钟数。对于此用途，建议改用 "DATEDIFF 函数 [日期和时间]" 第 140 页。

给定一个日期，此函数返回自 0000-02-29 00:00:00 以来的分钟数。

给定一个日期和整数，此函数将整数分钟数添加到指定的时间戳中。请改用 "DATEADD 函数 [日期和时间]" 第 139 页。

语法 1 返回整数。语法 2 返回时间戳。

语法 1　**MINUTES (** [*datetime-expression*,] *datetime-expression* **)**

语法 2

MINUTES (*datetime-expression*, *integer-expression* **)**

参数

datetime-expression 日期和时间。

integer-expression 要添加到 *datetime-expression* 中的分钟数。如果 *integer-expression* 是负数，则从日期时间值中减去相应的分钟数。如果提供整数表达式，必须将 *datetime-expression* 类型显式转换为日期时间数据类型。

有关转换数据类型的信息，请参阅 ""CAST 函数 [数据类型转换]" 第 125 页 "。

示例

下列语句返回值 240，表示第二个时间戳比第一个时间戳晚 240 秒。建议使用第二个示例 (DATEDIFF)。

```
SELECT MINUTES( '1999-07-13 06:07:12',
   '1999-07-13 10:07:12' )
SELECT DATEDIFF( minute,
   '1999-07-13 06:07:12',
   '1999-07-13 10:07:12' )
```

下面的语句返回值 1,051,040,527。

```
SELECT MINUTES( '1998-07-13 06:07:12' )
```

下列语句返回时间戳 1999-05-12 21:10:07.0。建议使用第二个示例 (DATEADD)。

```
SELECT MINUTES( CAST( '1999-05-12 21:05:07'
AS DATETIME ), 5)
SELECT DATEADD( minute, 5, '1999-05-12 21:05:07' )
```

用法

因为此函数返回整数，所以，当使用语法 1 并且时间戳大于或等于 4083-03-23 02:08:00 时可能会发生溢出。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

MOD 函数 [数字]

作用 返回一个整数除以另一个整数之后产生的余数。

语法 **MOD (** *dividend*, *divisor* **)**

参数 **dividend** 被除数，即除法的分子。

divisor 除数，即除法的分母。

示例 下面的语句返回值 2。

```
SELECT MOD( 5, 3 )
```

用法 被除数为负数时结果为负数或零。除数的符号则没有影响。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 核心 SQL 外部的 SQL/ 基础功能。
- **Sybase** 不受 Adaptive Server Enterprise 支持。 % 运算符在 Adaptive Server Enterprise 中作为模运算符使用。

另请参阅 "REMAINDER 函数 [数字]" 第 199 页

MONTH 函数 [日期和时间]

作用

- 返回 1 到 12 之间的一个数字，该数字对应于给定日期的月份。

语法 **MONTH (** *date-expression* **)**

参数 **date-expression** 日期时间值。

示例 下面的语句返回值 7。

```
SELECT MONTH( '1998-07-13' )
```

标准和兼容性

- **SQL/92.** 供应商扩展。
- **SQL/99.** 供应商扩展。
- **Sybase.** 不受 Adaptive Server Enterprise 支持。

MONTHNAME 函数 [日期和时间]

作用 返回日期中的月份名称。

语法 **MONTHNAME (** *date-expression* **)**

参数 **date-expression** 日期时间值。

示例 下面的语句返回值 September。

```
SELECT MONTHNAME( '1998-09-05' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

MONTHS 函数 [日期和时间]

作用 给定两个日期，此函数返回二者之间的整数月数。对于此用途，建议改用 "DATEDIFF 函数 [日期和时间]" 第 140 页。

给定一个日期，此函数返回自 0000-02 以来的月数。

给定一个日期和整数，此函数将整数月数添加到指定的日期中。对于此用途，建议改用 "DATEADD 函数 [日期和时间]" 第 139 页。

语法 1 返回整数。语法 2 返回时间戳。

语法 1 **MONTHS (** [*datetime-expression*,] *datetime-expression* **)**

语法 2 **MONTHS (** *datetime-expression*, *integer-expression* **)**

参数

datetime-expression 日期和时间。

integer-expression 要添加到 *datetime-expression* 中的月数。如果 *integer-expression* 是负数，则从日期时间值中减去相应的月数。如果提供整数表达式，必须将 *datetime-expression* 类型显式转换为日期时间数据类型。

有关转换数据类型的信息，请参阅 ""CAST 函数 [数据类型转换]" 第 125 页 "。

示例

下列语句返回值 2，表示第二个日期比第一个日期晚两个月。建议使用第二个示例 (DATEDIFF)。

```
SELECT MONTHS( '1999-07-13 06:07:12',
   '1999-09-13 10:07:12' )
SELECT DATEDIFF( month,
   '1999-07-13 06:07:12',
   '1999-09-13 10:07:12' )
```

下面的语句返回值 23,982。

```
SELECT MONTHS( '1998-07-13 06:07:12' )
```

下列语句返回时间戳 1999-10-12 21:05:07.0。建议使用第二个示例 (DATEADD)。

```
SELECT MONTHS( CAST( '1999-05-12 21:05:07'
AS DATETIME ), 5)
SELECT DATEADD( month, 5, '1999-05-12 21:05:07' )
```

用法

MONTHS 的值是根据两个日期之间每个月的 1 号的数目计算出的。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

NEWID 函数 [杂类]

作用

生成 UUID （通用唯一标识符）值。 UUID 与 GUID （全局唯一标识符）相同。

语法

NEWID( )

参数

没有任何与 NEWID() 关联的参数。

示例

下面的语句创建一个拥有两个列的 *mytab* 表。列 *pk* 的数据类型是唯一标识符，它以 newid() 函数作为缺省值。列 *c1* 的数据类型是整数。

CREATE TABLE mytab(pk uniqueidentifier primary key default newid(), c1 int)

如果您执行以下语句，

```
SELECT newid()
```

则以字符串形式返回唯一标识符。例如，其值可以是 0xd3749fe09cf446e399913bc6434f1f08。您可以使用 UUIDTOSTR() 函数将此字符串转换为可读的格式。

用法

NEWID() 函数生成一个唯一标识符值。它可以用在一个列的 DEFAULT 子句中。

UUID 可用来唯一地标识表中的行。生成的值的情况是：在一台计算机上生成的值与另一台计算机上生成的值不相同。因此，这些值也可以在复制和同步环境中用作关键字。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ASA SQL 用户指南 > 确保数据完整性 > NEWID 缺省值 "

"STRTOUUID 函数 [字符串]" 第 217 页

"UUIDTOSTR 函数 [字符串]" 第 227 页

"UNIQUEIDENTIFIER 数据类型 [Binary]" 第 84 页

NEXT_CONNECTION 函数 [系统]

作用　返回一个连接的标识号。

语法　**NEXT_CONNECTION (** [*connection-id*] [, *database-id*] **)**

参数

connection-id　一个整数，通常从前一个 NEXT_CONNECTION 调用返回。如果 *connection-id* 为 NULL，则 NEXT_CONNECTION 返回第一个连接 ID。

database-id　一个整数，该属性表示当前服务器上的某个数据库。如果未提供数据库 ID，则使用当前数据库。如果提供 NULL，则 NEXT_CONNECTION 返回下一个连接，与数据库无关。

示例　下面的语句返回当前数据库上第一个连接的标识符。标识符是类似 569851433 的整数值。

```
SELECT NEXT_CONNECTION( NULL )
```

下面的语句返回类似 1661140050 的值。

```
SELECT NEXT_CONNECTION( 569851433 )
```

下列调用返回当前数据库上 *connection-id* 之后的连接。

```
NEXT_CONNECTION( connection-id )
```

下列调用返回 *connection-id* 之后的连接 （与数据库无关）。

```
NEXT_CONNECTION( connection-id, NULL )
```

下列调用返回指定数据库上 *connection-id* 之后的连接。

```
NEXT_CONNECTION( connection-id, database-id )
```

下列调用返回第一个连接 （与数据库无关）。

```
NEXT_CONNECTION(NULL, NULL)
```

下列调用返回指定数据库上的第一个连接。

```
NEXT_CONNECTION( NULL, database-id )
```

用法

NEXT_CONNECTION 可用于枚举数据库的连接。若要获取第一个连接，请传递 NULL；若要获取每个后续连接，请传递前一个返回值。当没有其它的连接时，此函数返回 NULL。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

NEXT_DATABASE 函数 [系统]

作用

返回数据库的标识号。

语法

NEXT_DATABASE ({ **NULL** | *database-id* } **)**

参数

database-id 一个整数，指定数据库的 ID 号。

示例

下面的语句返回值 0，这是第一个数据库值。

```
SELECT NEXT_DATABASE( NULL )
```

下面的语句返回 NULL，表示服务器上没有其它的数据库。

```
SELECT NEXT_DATABASE( 0 )
```

用法

NEXT_DATABASE 可用于枚举数据库服务器上运行的数据库。若要获取第一个数据库，请传递 NULL；若要获得每个后续数据库，请传递前一个返回值。当没有其它的数据库时，此函数返回 NULL。

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

NOW 函数 [日期和时间]

作用　返回当前日期和时间。这是 CURRENT TIMESTAMP 的传统语法。

语法　**NOW (*)**

示例　下面的语句返回当前日期和时间。

```
SELECT NOW(*)
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

NULLIF 函数 [杂类]

作用　通过比较表达式提供缩写的 CASE 表达式。

语法　**NULLIF (** *expression-1*, *expression-2* **)**

参数　**expression-1**　要比较的表达式。

expression-2　要比较的表达式。

示例　下面的语句返回值 a：

```
SELECT NULLIF( 'a', 'b' )
```

下面的语句返回 NULL。

```
SELECT NULLIF( 'a', 'a' )
```

用法　NULLIF 比较两个表达式的值。

如果第一个表达式的值与第二个表达式的值相等， NULLIF 返回 NULL。

如果第一个表达式的值不等于第二个表达式的值，或者第二个表达式为 NULL，则 NULLIF 返回第一个表达式。

NULLIF 函数提供了编写某些 CASE 表达式的简便方法。

标准和兼容性

- **SQL/92** 初级功能。
- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"CASE 表达式 " 第 20 页

NUMBER 函数 [杂类]

作用

为查询结果中的每个连续行生成从 1 开始的编号。 NUMBER 主要是在选择列表中使用。

语法

NUMBER (*)

示例

下面的语句返回按顺序排列的部门列表。

```
SELECT NUMBER( * ), dept_name
FROM department
WHERE dept_id > 5
ORDER BY dept_name
```

用法

可以在选择列表中使用 NUMBER(*)，以在结果集中提供按顺序编号的行。 NUMBER(*) 返回每个结果行的 ANSI 行号值。这意味着 NUMBER 返回的值可能是正数或负数，具体取决于应用程序在结果集中的滚动方式。对于不敏感游标， NUMBER(*) 值总是正数，因为整个结果集是在 OPEN 状态下实现的。

此外，行号可能受某些游标类型更改的影响。对于不敏感游标和滚动游标，该值是固定的。如果存在并发更新，则该值对于动态游标和敏感游标可能会改变。

如果在 DELETE 语句、 WHERE 子句、 HAVING 子句、 ORDER BY 子句、子查询、涉及聚合、任何约束、 GROUP BY、 DISTINCT 的查询、包含 UNION ALL 的查询或派生表中使用 NUMBER，会产生语法错误。

NUMBER(*) 可在视图中使用 （受上述限制的约束），但在查询或外部视图中，对应于涉及 NUMBER(*) 的表达式的视图列最多可以被引用一次，而且视图在左外连接或完全外连接中不能以提供 Null 的表的形式出现。

在嵌入式 SQL 中，当您使用的游标引用包含 NUMBER(*) 函数的查询时，应小心。具体说来，当使用相对于游标结尾处的位置 （偏移为负的绝对位置）来定位数据库游标时，此函数返回负数。

在 UPDATE 语句的 SET 子句中，可以在赋值符号的右侧使用 NUMBER。例如， SET x = NUMBER(*)。

从 SELECT 语句中使用 INSERT （请参阅 "INSERT 语句 " 第 520 页）时，还可以使用 NUMBER 生成主键，尽管生成顺序主键的首选机制是使用 AUTOINCREMENT。

有关 AUTOINCREMENT 的信息，请参阅 "CREATE TABLE 语句 " 第 395 页。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

行为更改
在版本 8 中 NUMBER 函数的行为已经更改。有关更多信息，请参阅 "SQL Anywhere Studio 的新功能 > 8.0 版中的新增功能 > Adaptive Server Anywhere 行为更改 "。

- **Sybase** 不受 Adaptive Server Enterprise 支持。

PATINDEX 函数 [字符串]

作用 返回一个整数，该整数表示模式第一次在字符串中出现的起始位置。

语法 **PATINDEX ('%***pattern***%'**, *string_expression* **)**

参数 **pattern** 要搜索的模式。如果省略前导百分号通配符，那么，如果模式出现在字符串起始处， PATINDEX 返回 1 ； 否则返回零。

模式与 LIKE 比较运算使用相同的通配符。这些通配符是：

通配符	匹配项
_（下划线）	任意一个字符
%（百分号）	包含零个或多个字符的任意字符串
[]	指定范围或集合中的任何单个字符
[^]	不在指定范围或集合中的任何单个字符

string-expression 要在其中搜索模式的字符串。

示例

下面的语句返回值 2。

```
SELECT PATINDEX( '%hoco%', 'chocolate' )
```

下面的语句返回值 11。

```
SELECT PATINDEX ('%4_5_', '0a1A 2a3A 4a5A')
```

用法

PATINDEX 返回模式第一次出现的起始位置。如果未找到模式，则返回 0。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容，但不支持 USING 子句。

另请参阅

"LIKE 条件 " 第 28 页

"LOCATE 函数 [字符串]" 第 175 页

PI 函数 [数字]

作用

- 返回数字值 PI。

语法

PI (*)

示例　下面的语句返回值 3.141592653…

```
SELECT PI( * )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　Adaptive Server Enterprise 支持 PI() 函数，但不支持 PI(*)。

PLAN 函数 [杂类]

作用　以字符串形式返回 SQL 语句的详细计划优化策略。

语法　**PLAN (** *string-expression,* [*cursor-type*], [*update-status*] **)**

参数

string-expression　SQL 语句，通常是 SELECT 语句，但也可以是 UPDATE 或 DELETE。

cursor-type　字符串。**Cursor-type** 可以是 **asensitive**（缺省值）、**insensitive**、**sensitive** 或 **keyset-driven**。

update-status　字符串参数，它采用下列值之一来指示优化程序如何处理给定的游标：

值	说明
READ-ONLY	游标是只读的。
READ-WRITE (default)	可以读取游标，也可以向游标写入。
FOR UPDATE	可以读取游标，也可以向游标写入。这与 READ-WRITE 完全相同。

示例　下面的语句将 SELECT 语句作为字符串参数传递，并返回查询的执行计划。

```
SELECT PLAN( 'SELECT * FROM department WHERE dept_id >
100' )
```

该信息可以帮助决定要添加的索引或如何构建数据库，以获得更好的性能。

下面的语句返回一个字符串，该字符串包含基于 'select * from department where ….' 语句的 INSENSITIVE 游标的文本计划。

```
SELECT PLAN( 'SELECT * FROM department WHERE dept_id >
100', 'insensitive', 'read-only' )
```

在 Interactive SQL 中，可以在 " 结果 " 窗格中的 " 计划 " 选项卡上查看任意 SQL 语句的计划。

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"EXPLANATION 函数 [杂类]" 第 157 页

"GRAPHICAL_PLAN 函数 [杂类]" 第 161 页

"GRAPHICAL_ULPLAN 函数 [杂类]" 第 163 页

"LONG_ULPLAN 函数 [杂类]" 第 177 页

"SHORT_ULPLAN 函数 [杂类]" 第 207 页

POWER 函数 [数字]

作用

以一个数字为底数以另一个数字为指数计算乘方值。

语法

POWER (*numeric-expression-1*, *numeric-expression-2* **)**

参数

numeric-expression-1 底数。

numeric-expression-2 指数。

示例

下面的语句返回值 64。

```
SELECT Power( 2, 6 )
```

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

PROPERTY_DESCRIPTION 函数 [系统]

作用 返回有关属性的说明。

语法 **PROPERTY_DESCRIPTION (** { *property-id* | *property-name* } **)**

参数 **property-id** 一个整数，该整数表示数据库属性的属性号。通过 PROPERTY_NUMBER 函数可以确定此数字。当在一组属性间循环时，常使用 *property-id*。

property-name 给出数据库属性名称的字符串。

示例 下面的语句返回索引插入的说明性编号。

```
SELECT PROPERTY_DESCRIPTION( 'IndAdd' )
```

用法 每个属性都有编号和名称，但编号在不同的版本间可能有变化，因此不应用作给定属性的可靠标识符。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 "

PROPERTY 函数 [系统]

作用 以字符串形式返回指定的服务器级属性值。

语法 **PROPERTY (** { *property-id* | *property-name* } **)**

参数 **property-id** 一个整数，该属性表示服务器级属性的属性号。通过 PROPERTY_NUMBER 函数可以确定此数字。当在一组属性间循环时，常使用 *property-id*。

property-name 给出数据库属性名称的字符串。

示例 下面的语句返回当前数据库服务器的名称：

```
SELECT PROPERTY( 'Name' )
```

用法 每个属性都有编号和名称，但编号在不同的版本间可能有变化，因此不应用作给定属性的可靠标识符。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 服务器级属性 "

PROPERTY_NAME 函数 [系统]

作用 返回具有所提供属性号的属性的名称。

语法 **PROPERTY_NAME (** *property-id* **)**

参数 **property-id** 数据库属性的属性号。

示例 下面的语句返回与属性号 126 关联的属性。它指向的实际属性因版本而异。

```
SELECT PROPERTY_NAME( 126 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 数据库属性 "

PROPERTY_NUMBER 函数 [系统]

作用 返回具有所提供属性名称的属性的属性号。

语法 **PROPERTY_NUMBER (** *property-name* **)**

参数 **property-name** 属性名称。

示例 下面的语句返回一个整数值。实际的值因版本而异。

```
SELECT PROPERTY_NUMBER( 'PAGESIZE' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 数据库属性 "

QUARTER 函数 [日期和时间]

作用 从提供的日期表达式中返回一个表示某季度的数字。

语法 **QUARTER(** *date-expression* **)**

参数 **date- expression** 日期。

示例 下面的语句返回值 2。

```
SELECT QUARTER ( '1987/05/02' )
```

用法 有如下这些季度：

季度	期限 （含起止日期）
1	1 月 1 日到 3 月 31 日
2	4 月 1 日到 6 月 30 日

季度	期限 （含起止日期）
3	7 月 1 日到 9 月 30 日
4	10 月 1 日到 12 月 31 日

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

RADIANS 函数 [数字]

作用

- 将数字由角度转换成弧度。

语法

RADIANS (*numeric-expression* **)**

参数

numeric-expression 数字 （以角度为单位）。该角度将被转换成弧度。

示例

下面的语句返回近似值 .5236。

```
SELECT RADIANS( 30 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

RAND 函数 [数字]

作用

返回 0 到 1 之间的一个随机数 （带可选的种子值）。

语法

RAND ([*integer-expression*] **)**

参数

integer expression 用于创建随机数的可选种子值。该参数可用于创建可重复的随机数序列。

示例 下面的语句返回近似值 0.0554504。

```
SELECT RAND( 4 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

REMAINDER 函数 [数字]

作用 返回一个整数除以另一个整数之后产生的余数。

语法 **REMAINDER (** *dividend*, *divisor* **)**

参数 **dividend** 被除数，即除法的分子。

divisor 除数，即除法的分母。

示例 下面的语句返回值 2。

```
SELECT REMAINDER( 5, 3 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。 %（模）运算符和除法运算符可用于产生余数。

另请参阅 "MOD 函数 [数字]" 第 183 页

REPEAT 函数 [字符串]

作用 以指定的次数连接字符串。

语法 **REPEAT (** *string-expression*, *integer-expression* **)**

参数 **string-expression** 要重复的字符串。

integer-expression 字符串的重复次数。如果 *integer-expression* 是负数，则返回空字符串。

示例 下面的语句返回值 repeatrepeatrepeat。

```
SELECT REPEAT( 'repeat', 3 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持，但 REPLICATE 可提供同样的功能。

另请参阅 "REPLICATE 函数 [字符串]" 第 201 页

REPLACE 函数 [字符串]

作用 用另一个子串替换在各个位置出现的某个子串。

语法 **REPLACE (** *original-string*, *search-string*, *replace-string* **)**

参数 如果有某个参数为 NULL，此函数返回 NULL。

original-string 要被搜索的字符串。这可以是任意长度。

search-string 要搜索并被 *replace-string* 替换的字符串。该字符串的长度不应超过 255 个字节。如果 *search-string* 是空字符串，则按原样返回原始字符串。

replace-string 该字符串用于替换 *search-string*。它可以是任意长度。如果 *replacement-string* 是空字符串，则删除出现的所有 *search-string*。

示例 下面的语句返回值 xx.def.xx.ghi。

```
SELECT REPLACE( 'abc.def.abc.ghi', 'abc', 'xx' )
```

下面的语句生成包含 ALTER PROCEDURE 语句的结果集，这些语句在执行时会修复引用已被重命名的表的存储过程。（表名必须唯一才能使用。）

```
SELECT REPLACE(
   replace(proc_defn,'OldTableName','NewTableName'),
   'create procedure',
   'alter procedure')
FROM SYS.SYSPROCEDURE
WHERE proc_defn LIKE '%OldTableName%'
```

对于 LIST 函数，使用逗号以外的分隔符：

```
SELECT REPLACE( list( table_id ), ',', '--')
FROM  SYS.SYSTABLE
WHERE table_id <= 5
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "SUBSTRING 函数 [字符串]" 第 219 页

REPLICATE 函数 [字符串]

作用 以指定的次数连接字符串。

语法 **REPLICATE (** *string-expression*, *integer-expression* **)**

参数 **string-expression** 要重复的字符串。

integer-expression 字符串的重复次数。

示例 下面的语句返回值 repeatrepeatrepeat。

```
SELECT REPLICATE( 'repeat', 3 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅 "REPEAT 函数 [字符串]" 第 199 页

REWRITE 函数 [杂类]

作用

返回重写的 SELECT、 UPDATE 或 DELETE 语句。

语法

REWRITE (*select-statement* [**, 'ANSI'**] **)**

示例

在下例中，对查询执行了两个重写优化。第一个优化解除了雇员与 sales_order 表之间连接中的子查询嵌套。第二个优化消除雇员与 sales_order 之间的主键 - 外键连接，从而简化了查询。此重写优化的一部分是用连接谓词 s.sales_rep IS NOT NULL 替换谓词 e.emp_id=s.sales_rep。

```
SELECT REWRITE( 'SELECT s.id, s.order_date
    FROM sales_order s
    WHERE  EXISTS(SELECT *
       FROM employee e
          WHERE e.emp_id = s.sales_rep)' ) FROM dummy
```

查询返回包含重写查询的单列结果集：

```
'SELECT s.id, s.order_date FROM sales_order s WHERE
s.sales_rep IS NOT NULL'
```

下一个 REWRITE 示例使用 ANSI 参数。

```
SELECT REWRITE( 'SELECT DISTINCT s.id, s.order_date,
e.emp_fname, e.emp_id
    FROM sales_order s, employee e
         WHERE e.emp_id *= s.sales_rep', 'ANSI' ) FROM
dummy
```

结果是语句的 ANSI 等效项。这种情况下， Transact-SQL 外连接转换为 ANSI 外连接。查询返回单列结果集：

```
'SELECT DISTINCT s.id, s.order_date, e.emp_id,
e.emp_fname FROM employee as e LEFT OUTER JOIN
sales_order as s ON e.emp_id = s.sales_rep'
```

用法

使用 REWRITE 函数时可以不带 ANSI 参数，这样可以帮助理解优化程序是如何生成给定查询的访问计划的。具体说来，可以发现 Adaptive Server Anywhere 是如何重写语句的 WHERE、 ON 和 HAVING 子句中的条件，然后确定是否存在可用于缩短请求执行时间的索引。

REWRITE 返回的语句可能与原始语句的语义不匹配。这是因为一些重写优化引入了不能直接转换成 SQL 的内部机制。例如，服务器使用行标识符执行重复项消除就不能转换成 SQL。

REWRITE() 函数重写的查询不是用来执行的。它是一种工具，用于通过显示重写阶段后传递给优化程序的内容来分析性能问题。

有些重写优化不在 REWRITE 的输出中反映出来。它们包括 LIKE 优化、最小或最大函数的优化、上限 / 下限消除以及判定包容。

如果指定了 ANSI，则 REWRITE 返回语句的 ANSI 等效项。这种情况下，仅应用下列重写优化：

- Transact-SQL 外连接被重写为 ANSI SQL 外连接。
- 消除重复的相关名。
- KEY 和 NATURAL 连接被重写为 ANSI SQL 连接。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ASA SQL 用户指南 > 查询优化与执行 > 语义查询转换 "

"ASA 数据库管理指南 > 数据库选项 > EXTENDED_JOIN_SYNTAX 选项 [database]"

"ASA SQL 用户指南 > 连接：从多个表检索数据 > Transact-SQL 外连接（*= 或 =*） "

"ASA SQL 用户指南 > 连接：从多个表检索数据 > 键连接 "

"ASA SQL 用户指南 > 连接：从多个表检索数据 > 自然连接 "

"ASA SQL 用户指南 > 连接：从多个表检索数据 > 连接中的重复相关名（星形连接） "

RIGHT 函数 [字符串]

作用

- 返回字符串中最右边的字符。

语法

RIGHT (*string-expression*, *integer-expression* **)**

参数

string-expression 从左边截断的字符串。

integer-expression 要返回的从字符串结尾处开始计数的字符数。

示例

下面的语句返回值 olate。

```
SELECT RIGHT( 'chocolate', 5 )
```

用法

如果字符串包含多字节字符，并且使用了适当的归类，则返回的字节数可能大于指定的字符数。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"LEFT 函数 [字符串]" 第 172 页

"ASA 数据库管理指南 > 国际语言和字符集 "

ROUND 函数 [数字]

作用

将 *numeric-expression* 舍入到指定的整数表达式中小数点后的位数。

语法

ROUND (*numeric-expression*, *integer-expression* **)**

参数

numeric-expression 传递给函数的要舍入的数字。

integer-expression 正整数指定小数点右边舍入到的有效位数。负数表达式指定小数点左边舍入到的有效位数。

示例

下面的语句返回值 123.200。

```
SELECT ROUND( 123.234, 1 )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅　"TRUNCNUM 函数 [数字]" 第 225 页

RTRIM 函数 [字符串]

作用

- 返回尾随空白已删除的字符串。

语法　**RTRIM (** *string-expression* **)**

参数　**string-expression**　要剪裁的字符串。

示例　下面的语句返回字符串 Test Message，其中的所有尾随空白都已删除。

```
SELECT RTRIM( 'Test Message     ' )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅　"LTRIM 函数 [字符串]" 第 179 页

SECOND 函数 [日期和时间]

作用　返回 0 到 59 之间的一个数字，该数字对应于给定日期时间值的秒部分。

语法　**SECOND (** *datetime-expression* **)**

参数　**datetime-expression**　日期时间值。

示例　下面的语句返回值 21。

```
SELECT SECOND( '1998-07-13:21:21:25' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

SECONDS 函数 [日期和时间]

作用

给定两个时间戳，此函数返回二者之间的整数秒数。对于此用途，建议改用 "DATEDIFF 函数 [日期和时间]" 第 140 页。

给定一个日期，此函数返回自 0000-02-29 00:00:00 以来的秒数。

给定一个日期和整数，此函数将整数秒数添加到指定的时间戳中。对于此用途，建议改用 "DATEADD 函数 [日期和时间]" 第 139 页。

语法 1 返回一个 bigint。语法 2 返回时间戳。

语法 1

SECONDS ([*datetime-expression*,] *datetime-expression* **)**

语法 2

SECONDS (*datetime-expression*, *integer-expression* **)**

参数

datetime-expression 日期和时间。

integer-expression 要添加到 *datetime-expression* 中的秒数。如果 *integer-expression* 是负数，则从日期时间值中减去相应的秒数。如果提供整数表达式，必须将 *datetime-expression* 类型显式转换为日期时间数据类型。

有关转换数据类型的信息，请参阅 ""CAST 函数 [数据类型转换]" 第 125 页 "。

示例

下列语句返回值 14 400，表示第二个时间戳比第一个时间戳晚 14 400 秒。

```
SELECT SECONDS( '1999-07-13 06:07:12',
   '1999-07-13 10:07:12' )
SELECT DATEDIFF( second,
   '1999-07-13 06:07:12',
   '1999-07-13 10:07:12' )
```

下面的语句返回值 63,062,431,632。

```
SELECT SECONDS( '1998-07-13 06:07:12' )
```

下列语句返回日期时间 1999-05-12 21:05:12.0。

```
SELECT SECONDS( CAST( '1999-05-12 21:05:07'
AS TIMESTAMP ), 5)
SELECT DATEADD( second, '1999-05-12 21:05:07' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

SHORT_ULPLAN 函数 [杂类]

作用

以字符串形式返回 SQL 语句的 UltraLite 计划优化策略的简短说明。此说明与 EXPLANATION 函数返回的说明相同。

对于某些查询， UltraLite 执行计划可能不同于为 Adaptive Server Anywhere 选择的计划。

语法

SHORT_ULPLAN (*string-expression* **)**

参数

string-expression SQL 语句，通常是 SELECT 语句，但也可以是 UPDATE 或 DELETE。

示例

下面的语句将 SELECT 语句作为字符串参数传递，并返回查询的执行计划。

```
SELECT EXPLANATION( 'select * from department where
dept_id > 100' )
```

该信息可以帮助决定要添加的索引或如何构建数据库，以获得更好的性能。

在 Interactive SQL 中，可以在 " 结果 " 窗格中的 "UltraLite 计划 " 选项卡上查看任意 SQL 语句的计划。

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"PLAN 函数 [杂类]" 第 193 页

"EXPLANATION 函数 [杂类]" 第 157 页

"GRAPHICAL_PLAN 函数 [杂类]" 第 161 页

"GRAPHICAL_ULPLAN 函数 [杂类]" 第 163 页

"LONG_ULPLAN 函数 [杂类]" 第 177 页

SIGN 函数 [数字]

作用

返回一个数字的符号。

语法

SIGN (*numeric-expression* **)**

参数

numeric-expression 要返回其符号的数字。

示例

下面的语句返回值 -1

```
SELECT SIGN( -550 )
```

返回值

对于负数， SIGN 返回 -1。

对于零， SIGN 返回 0。

对于正数， SIGN 返回 1。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

SIMILAR 函数 [字符串]

作用　返回一个数字，以表示两个字符串之间的相似性。

语法　**SIMILAR (** *string-expression-1*, *string-expression-2* **)**

参数　**string-expression-1**　要比较的第一个字符串。

string-expression-2　要比较的第二个字符串。

示例　下面的语句返回值 75。

```
SELECT SIMILAR( 'toast', 'coast' )
```

这表示两个值有 75% 相似。

用法　此函数返回 0 和 100 之间的一个整数，以表示两个字符串之间的相似性。结果可以被解释成两个字符串之间匹配字符的百分比。 100 表示两个字符串相同。

此函数可用于校正名称 （如客户）列表。某些客户可能用稍微不同的名称多次添加到列表中。将表与自身连接，生成一个关于所有大于 90% 但小于 100% 的相似性的报告。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

SIN 函数 [数字]

作用

- 返回一个数字的正弦值。

语法　**SIN (** *numeric-expression* **)**

参数　**numeric-expression**　角度 （以弧度为单位）。

示例　下面的语句返回值 .496880。

```
SELECT SIN( 0.52 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"ASIN 函数 [数字]" 第 121 页

"COS 函数 [数字]" 第 134 页

"COT 函数 [数字]" 第 135 页

"TAN 函数 [数字]" 第 221 页

SORTKEY 函数 [字符串]

作用

生成可用于根据可选归类规则对字符串进行排序的值。

语法

SORTKEY (*string-expression* [, *collation-name* | , *collation-id*] **)**

参数

string-expression 字符串表达式只能包含以数据库的字符集编码的字符。

如果 *string-expression* 是空字符串，则 SORTKEY 返回零长度的二进制值。如果 *string-expression* 为 Null，则 SORTKEY 返回 Null 值。空字符串的排序顺序值不同于数据库列中的 Null 字符串。

SORTKEY 可以处理的最大字符串长度是 254 个字节。任何多出的部分都将被忽略。

collation-name 一个字符串或字符变量，用于指定要使用的排序顺序的名称。

collation-id 一个变量、整数常量或字符串，用于指定要使用的排序顺序的 ID 号。

如果未指定归类，则缺省值为 " 缺省 Unicode 多语言 "。

下面是 *collation-name* 和 *collation-id* 的有效值：

说明	归类名	归类 ID
缺省 Unicode 多语言	default	0
CP 850 备选：无重音	altnoacc	39
CP 850 备选：小写字母优先	altdict	45
CP 850 西欧：不分大小写优先	altnocsp	46
CP 850 斯堪的纳维亚文字典	scandict	47
CP 850 斯堪的纳维亚：不分大小写优先	scannocp	48
GB 拼音	gbpinyin	无
二进制排序	binary	50
拉丁文 -1 英语、法语、德语字典	dict	51
拉丁文 -1 英语、法语、德语，不分大小写	nocase	52
拉丁文 -1 英语、法语、德语，不分大小写优先	nocasep	53
拉丁文 -1 英语、法语、德语，无重音	noaccent	54
拉丁文 -1 西班牙语字典	espdict	55
拉丁文 -1 西班牙语，不分大小写	espnocs	56
拉丁文 -1 西班牙语，无重音	espnoac	57
ISO 8859-5 俄语字典	rusdict	58
ISO 8859-5 俄语，不分大小写	rusnocs	59
ISO 8859-5 西里尔字母字典	cyrdict	63
ISO 8859-5 西里尔字母，不分大小写	cyrnocs	64
ISO 8859-7 希腊文字典	elldict	65
ISO 8859-2 匈牙利文字典	hundict	69
ISO 8859-2 匈牙利文，无重音	hunnoac	70

说明	归类名	归类 ID
ISO 8859-2 匈牙利文，不分大小写	hunnocs	71
ISO 8859-5 土耳其文字典	turdict	72
ISO 8859-5 土耳其文，无重音	turnoac	73
ISO 8859-5 土耳其文，不分大小写	turnocs	74
CP 874 (TIS 620) 皇家泰语字典	thaidict	257
ISO 14651 排序标准	14651	258
Shift-JIS 二进制顺序	sjisbin	259
Unicode UTF-8 二进制排序	utf8bin	260
EUC JIS 二进制顺序	eucjisbin	261
GB2312 二进制顺序	gb2312bin	262
CP932 MS 二进制顺序	cp932bin	263
Big5 二进制顺序	big5bin	264
EUC KSC 二进制顺序	euckscbin	265

归类表有两种类型：内部和外部。内部表包含在 dll 中，外表位于单独的文件中。 *collation-id* 不能用于外表。

您还可以定义自己的归类表。为此，可以在 `.ust` 文件中创建您自己的归类表，并将其复制到预装的 `.ust` 文件所在的同一文件夹中：`.../charsets/unicode/`。该文件名可以用作 *collation-name*。

有关 `.ust` 文件的更多信息，请参阅 Adaptive Server Enterprise 文档。

示例

下列语句按排序顺序返回排序关键字值：拉丁文 -1，英语、法语、德语字典。

```
SELECT SORTKEY('coop', 'dict')
```

SORTKEY('coop', 'dict')
0x08890997099709b30008000800080008

```
SELECT SORTKEY ( 'C?-op', 51 )
```

SORTKEY('C?op', 51)
0x08890997099709b3002000470002000800080008000 1fffd002d

用法

SORTKEY 函数生成的值可用于根据预定义的排序顺序行为对结果进行排序。这样，您就可以处理超出 Adaptive Server Anywhere 归类限制的字符排序顺序行为。返回值是一个二进制值，包含有关从 SORTKEY 函数保持的输入字符串的已编码排序顺序信息。

例如，可以将 SORTKEY 返回的值存储在包含源字符串的列中。当您想按所需的顺序检索字符数据时， SELECT 语句只需要在包含 SORTKEY 运行结果的列上包括 ORDER BY 子句。

SORTKEY 函数保证它为一组给定的排序顺序标准返回的值适用于对 varbinary 数据类型执行的二进制比较。

SORTKEY 的输入最多可以为每个输入字符生成 6 个字节的排序顺序信息。 SORTKEY 的输出是 varbinary 类型，最大长度为 (254 * 6) 个字节。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容，但 Adaptive Server Enterprise 不允许使用自定义排序顺序。

另请参阅

"COMPARE 函数 [字符串]" 第 129 页

"ASA 数据库管理指南 > 国际语言和字符集 > 国际语言和字符集简介 "

SOUNDEX 函数 [字符串]

作用

返回表示字符串声音的数字。

语法

SOUNDEX (*string-expression* **)**

参数

string-expression 字符串。

示例 下面的语句返回两个数字，表示每个名称的声音。每个参数的 SOUNDEX 值都是 3827。

```
SELECT SOUNDEX( 'Smith' ), SOUNDEX( 'Smythe' )
```

用法 字符串的 SOUNDEX 函数值基于第一个字母和随后三个除 H、 Y 和 W 之外的辅音。两个重复的字母按一个字母计算。例如，

```
SOUNDEX( 'apples' )
```

基于字母 A、 P、 L 和 S。

SOUNDEX 函数忽略多字节字符。

尽管不很完美，但 SOUNDEX 对于那些听起来相似和以相同字母开头的单词，通常返回相同的数字。

SOUNDEX 函数对英语单词效果最好。对于其它语言，它的用处不大。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容，但 Adaptive Server Enterprise 返回 CHAR(4) 结果，而 Adaptive Server Anywhere 返回整数。

SPACE 函数 [字符串]

作用

- 返回指定数目的空格。

语法 **SPACE (** *integer-expression* **)**

参数 **integer expression** 要返回的空格数。

示例 下面的语句返回包含 10 个空格的字符串。

```
SELECT SPACE( 10 )
```

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

SQLDIALECT 函数 [杂类]

作用 返回 "Watcom-SQL" 或 "Transact-SQL"，表示语句的 SQL 方言。

语法 **SQLDIALECT (** *sql-statement-string* **)**

示例 下面的语句返回字符串 Transact-SQL。

```
SELECT SQLDIALECT( 'SELECT employeeName = emp_lname
FROM employee' ) FROM dummy
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "TRANSACTSQL 函数 [杂类]" 第 223 页

"WATCOMSQL 函数 [杂类]" 第 229 页

SQRT 函数 [数字]

作用

- 返回一个数字的平方根。

语法 **SQRT (** *numeric-expression* **)**

参数 **numeric-expression** 要计算其平方根的数字。

示例 下面的语句返回值 3。

```
SELECT SQRT( 9 )
```

标准和兼容性

- **SQL/92.** 供应商扩展。
- **SQL/99.** 供应商扩展。
- **Sybase.** 与 Adaptive Server Enterprise 兼容。

STR 函数 [字符串]

作用

返回一个数字的等效字符串。

语法

STR (*numeric_expression* [, *length* [, *decimal*]] **)**

参数

numeric-expression 任何近似数字 （浮点、实数或双精度）表达式。

length 要返回的字符数 （包括小数点、小数点右侧和左侧的所有位以及空白）。缺省值为 10。

decimal 要返回的小数位数。缺省值为 0。

示例

下面的语句返回的字符串共有 10 个字符，其中 1235 前面有 6 个空格：

```
SELECT STR( 1234.56 )
```

下面的语句返回结果 1234.6：

```
SELECT STR( 1234.56, 6, 1 )
```

用法

如果数字的整数部分超过了指定的长度，则返回具有指定长度但只包含星号的字符串。例如，下面的语句返回 ***

```
SELECT STR( 1234.56, 3 )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

STRING 函数 [字符串]

作用　将一个或多个字符串连接为一个大字符串。

语法　**STRING (** *string-expression* [, …] **)**

参数　**string-expression**　字符串。

如果只提供一个参数，则它被转换为单个表达式。如果提供多个参数，则它们被连接为单个字符串。

示例　下面的语句返回的值是 *testing123*。

```
SELECT STRING( 'testing', NULL, 123 )
```

用法　数字或日期参数在连接前转换为字符串。 STRING 函数还可用于将任何单个表达式转换为字符串，方法是将该表达式作为唯一的参数提供。

如果所有参数都为 NULL，则 STRING 返回 NULL。如果任何参数为非 NULL，则任何 NULL 参数都会被视为空字符串。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

STRTOUUID 函数 [字符串]

作用　将字符串值转换为唯一标识符 (UUID 或 GUID) 值。

语法　**STRTOUUID(** *string-expression* **)**

参数　**string-expression**　字符串，格式为 *xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx*

示例

```
CREATE TABLE
    T (pk uniqueidentifier primary key, c1 int);
INSERT INTO T (pk, c1)
VALUES  (STRTOUUID
    ('12345678-1234-5678-9012-123456789012'), 1);
```

用法 将一个格式为 *xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx*（其中 *x* 是十六进制数字）的字符串转换为一个唯一标识符值。如果该字符串不是有效的 UUID 字符串，则返回 NULL。

此函数用于在 Adaptive Server Anywhere 数据库中输入 UUID 值。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "UUIDTOSTR 函数 [字符串]" 第 227 页

"NEWID 函数 [杂类]" 第 185 页

STUFF 函数 [字符串]

作用 从一个字符串中删除一些字符，并用另一个字符串替换它们。

语法 **STUFF (** *string-expression1*, *start*, *length*, *string-expression2* **)**

参数 **string-expression1** 要用 STUFF 函数修改的字符串。

start 字符位置，从此处开始删除字符。字符串中第一个字符的位置是 1。

length 要删除的字符数。

string-expression2 要插入的字符串。若要使用 STUFF 删除字符串的一部分，请使用 NULL 作为替换字符串。

示例 下面的语句返回值 chocolate pie。

```
SELECT STUFF( 'chocolate cake', 11, 4, 'pie' )
```

标准和兼容性

- **SQL/92.** 供应商扩展。
- **SQL/99.** 供应商扩展。
- **Sybase.** 与 Adaptive Server Enterprise 兼容。

另请参阅 "INSERTSTR 函数 [字符串]" 第 168 页

SUBSTRING 函数 [字符串]

作用 返回字符串的子串。

语法 { **SUBSTRING** | **SUBSTR** }**(** *string-expression*, *start* [, *length*] **)**

参数 **string-expression** 从中返回子串的字符串。

start 要返回的子串的起始位置 （以字符为单位）。负起始位置指定从字符串结尾处 （而不是开始处）开始的字符数。字符串中第一个字符的位置是 1。

length 要返回的子串的长度 （以字符为单位）。正 *length* 表示子串在起始位置的右侧 *length* 字符数处结束，而负 *length* 表示子串在起始位置的左侧结束 *length* 字符数处结束。

示例 下面的语句返回 *back*：

```
SELECT SUBSTRING( 'back yard',1 ,4 )
```

下面的语句返回 *yard*：

```
SELECT SUBSTRING( 'back yard', -1 , -4 )
```

用法 如果指定了 *length*，则子串不应超过该长度。如果未指定长度，则返回字符串的剩余部分，从 *start* 位置开始。

start 和 *length* 都可以为负数。使用适当的负数和正数组合，可以从字符串的开始处或结束处获取子串。

如果 *string-expression* 是二进制数据类型，则 SUBSTRING 函数的行为与 BYTE_SUBSTR 相同。

标准和兼容性

- **SQL/92** 初级功能。
- **SQL/99** 核心特性。
- **Sybase** SUBSTRING 与 Adaptive Server Enterprise 兼容。Adaptive Server Enterprise 不支持 SUBSTR。

另请参阅

"BYTE_SUBSTR 函数 [字符串]" 第 125 页

SUM 函数 [集合]

作用

返回每一组行的指定表达式总数。

语法

SUM (*expression* | **DISTINCT** *column-name* **)**

参数

expression 要总计的对象。这通常是列名称。

DISTINCT column-name 其用途有限，才将它列了出来。

示例

下面的语句返回值 3749146。

```
SELECT SUM( salary )
FROM Employee
```

用法

不包括指定表达式为 NULL 的行。

对于不包含任何行的组返回 NULL。

标准和兼容性

- **SQL/92** 与 SQL/92 兼容。
- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"COUNT 函数 [集合]" 第 135 页

"AVG 函数 [集合]" 第 123 页

TAN 函数 [数字]

作用

- 返回一个数字的正切值。

语法

TAN (*numeric-expression* **)**

参数

numeric-expression　角度 （以弧度为单位）。

示例

下面的语句返回值 .572561。

```
SELECT TAN( 0.52 )
```

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　与 Adaptive Server Enterprise 兼容。

另请参阅

"COS 函数 [数字]" 第 134 页

"SIN 函数 [数字]" 第 209 页

TEXTPTR 函数 [文本和图像]

作用

返回指向指定文本列的第一页的 16 字节二进制指针。

语法

TEXTPTR (*column-name* **)**

参数

column-name　文本列的名称。

示例

使用 TEXTPTR 定位与作者的 blurbs 表中的 au_id 486-29-1786 关联的文本列 copy。

文本指针放置在局部变量 @val 中，并作为 readtext 命令的参数提供，此命令返回从第二个字节处开始 （偏移 1）的 5 个字节。

```
DECLARE @val VARBINARY(16)
SELECT @val = TEXTPTR(copy)
FROM blurbs
WHERE au_id = "486-29-1786"
READTEXT blurbs.copy @val 1 5
```

用法 包括此函数是为了与 Transact-SQL 兼容。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

TODAY 函数 [日期和时间]

作用 返回当前日期。这是 CURRENT DATE 的传统语法。

语法 **TODAY (*)**

示例 下列语句根据系统时钟返回当前日期。

```
SELECT TODAY( * )
SELECT CURRENT DATE
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

TRACEBACK 函数 [杂类]

作用 返回字符串，其中包含在最近的异常 （错误）发生时执行的过程和触发器的跟踪信息。

语法 **TRACEBACK (*)**

示例 使用 traceback 函数的方法是，在过程执行中发生错误后输入以下语句：

```
SELECT TRACEBACK ( * )
```

用法 这对于调试过程和触发器很有用。

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

TRANSACTSQL 函数 [杂类]

作用 接受 Watcom-SQL 语句并用 Transact-SQL 方言重写。

语法 **TRANSACTSQL(** *sql-statement-string* **)**

示例 下面的语句返回字符串 'select EmployeeName = empl_name from employee'。

```
SELECT TRANSACTSQL( 'SELECT empl_name as EmployeeName
FROM employee' ) FROM dummy
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅 "SQLDIALECT 函数 [杂类]" 第 215 页

"WATCOMSQL 函数 [杂类]" 第 229 页

TRIM 函数 [字符串]

作用

- 删除字符串中的前导和尾随空白。

语法 **TRIM (** *string-expression* **)**

参数 **string-expression** 要剪裁的字符串。

示例 下面的语句返回没有前导和尾随空白的值 *chocolate* 。

```
SELECT TRIM( '   chocolate   ' )
```

标准和兼容性

- **SQL/92** 初级功能。
- **SQL/99** 核心特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"LTRIM 函数 [字符串]" 第 179 页

"RTRIM 函数 [字符串]" 第 205 页

TRUNCATE 函数 [数字]

作用 在小数点后的指定位数截断数字。此函数已被弃用，赞成使用 TRUNCNUM 函数。

语法 "**TRUNCATE**" **(** *numeric-expression*, *integer-expression* **)**

参数 **numeric-expression** 要截断的数字。

integer-expression 正整数指定小数点右边舍入到的有效位数。负数表达式指定小数点左边舍入到的有效位数。

示例 下面的语句返回值 600。

```
SELECT "TRUNCATE"( 655, -2 )
```

下面的语句返回值 655.340。

```
SELECT "TRUNCATE"( 655.348, 2 )
```

用法 此函数与 TRUNCNUM 相同。建议使用 TRUNCNUM，因为它不会导致关键字冲突。

必须使用引号，因为关键字与 TRUNCATE TABLE 语句冲突。仅当 QUOTED_IDENTIFIER 选项被设置为 OFF 时，使用 TRUNCATE 时才不需要引号。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ASA 数据库管理指南 > 数据库选项 > QUOTED_IDENTIFIER 选项 [compatibility]"

"TRUNCNUM 函数 [数字]" 第 225 页

TRUNCNUM 函数 [数字]

作用

在小数点后的指定位数截断数字。

语法

TRUNCNUM (*numeric-expression*, *integer-expression* **)**

参数

numeric-expression 要截断的数字。

integer-expression 正整数指定小数点右边舍入到的有效位数。负数表达式指定小数点左边舍入到的有效位数。

示例

下面的语句返回值 600。

```
SELECT TRUNCNUM( 655, -2 )
```

下面的语句返回值 655.340。

```
SELECT TRUNCNUM( 655.348, 2 )
```

用法

此函数与 TRUNCATE 相同，但不会导致关键字冲突。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"ROUND 函数 [数字]" 第 204 页

"TRUNCATE 函数 [数字]" 第 224 页

UCASE 函数 [字符串]

作用 将字符串中的所有字符转换成大写字母。

语法 **UCASE (** *string-expression* **)**

参数 **string-expression** 要转换成大写字母的字符串。

示例 下面的语句返回值 CHOCOLATE。

```
SELECT UCASE( 'ChocoLate' )
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Enterprise 不支持 UCASE，但 UPPER 可以以兼容的方式提供同样的功能。

另请参阅 "UPPER 函数 [字符串]" 第 226 页

"LCASE 函数 [字符串]" 第 172 页

UPPER 函数 [字符串]

作用 将字符串中的所有字符转换成大写字母。

语法 **UPPER (** *string-expression* **)**

参数 **string-expression** 要转换成大写字母的字符串。

示例 下面的语句返回值 CHOCOLATE。

```
SELECT UPPER( 'ChocoLate' )
```

标准和兼容性

- **SQL/92** 此函数与 SQL/92 兼容。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"UCASE 函数 [字符串]" 第 226 页

"LCASE 函数 [字符串]" 第 172 页

"LOWER 函数 [字符串]" 第 178 页

UUIDTOSTR 函数 [字符串]

作用

将唯一标识符值 （UUID，又称 GUID）转换为字符串值。

语法

UUIDTOSTR(*uuid-expression* **)**

参数

uuid-expression　唯一标识符值。

示例

下面的语句创建一个拥有两个列的 *mytab* 表。列 *pk* 的数据类型是唯一标识符，列 *c1* 的数据类型是整数。该语句在列 *c1* 中插入两行 （值分别为 1 和 2）。

```
CREATE TABLE mytab(
    pk uniqueidentifier primary key default newid(),
    c1 int )
INSERT INTO mytab( c1 ) values ( 1 )
INSERT INTO mytab( c1 ) values ( 2 )
```

执行下面的 SELECT 语句时将返回新创建的表中的所有数据。

```
SELECT *
FROM mytab
```

您将看到一个包含两列、两行的表。列 *pk* 的显示值将是二进制值。

若要将唯一标识符值转换为可读的格式，请执行下面 的命令：

```
SELECT uuidtostr(pk),c1
FROM mytab
```

用法

将唯一标识符转换为格式为 *xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx* （其中的 x 是十六进制数字）的字符串值。如果此二进制值不是有效的标识符，则返回 NULL。

如果您想查看 UUID 值，则可使用此函数。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"NEWID 函数 [杂类]" 第 185 页

"STRTOUUID 函数 [字符串]" 第 217 页

VAREXISTS 函数 [杂类]

作用

如果已经使用给定的名称创建或声明了一个用户定义的变量，则返回 1。如果没有创建这样的变量，则返回 0。

语法

VAREXISTS (*variable-name-string* **)**

参数

variable-name-string *name to be tested, as a* 字符串。

示例

下面的 IF 语句使用名称 **start_time** 创建一个变量 （如果尚未创建或声明变量）。接着可以安全地使用此变量。

```
IF VAREXISTS('start_time') = 0 THEN
    CREATE VARIABLE start_time TIMESTAMP;
END IF;
SET start_time = current timestamp;
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

另请参阅

"CREATE VARIABLE 语句 " 第 417 页

"DECLARE 语句 " 第 425 页

"IF 语句 " 第 510 页

WATCOMSQL 函数 [杂类]

作用

接受 Transact-SQL 语句并以 Watcom-SQL 方言重写。当将现有的 Adaptive Server Enterprise 存储过程转换成 Watcom SQL 语法时，此函数很有用。

语法

WATCOMSQL(*sql-statement-string* **)**

示例

下面的语句返回字符串 'select empl_name as EmployeeName from employee'。

```
SELECT WATCOMSQL( 'SELECT EmployeeName=empl_name FROM
employee' ) FROM dummy
```

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

另请参阅

"SQLDIALECT 函数 [杂类]" 第 215 页

"TRANSACTSQL 函数 [杂类]" 第 223 页

WEEKS 函数 [日期和时间]

作用

给定两个日期，此函数返回二者之间的整数周数。对于此用途，建议改用 "DATEDIFF 函数 [日期和时间]" 第 140 页。

给定一个日期，此函数返回自 0000-02-29 以来的周数。

给定一个日期和整数，此函数将整数周数添加到指定的日期中。对于此用途，建议改用 "DATEADD 函数 [日期和时间]" 第 139 页。

语法 1 返回整数。语法 2 返回时间戳。

语法 1

WEEKS ([*datetime-expression*,] *datetime-expression* **)**

语法 2

WEEKS (*datetime-expression*, *integer-expression* **)**

参数

datetime-expression 日期和时间。

integer-expression 要添加到 *datetime-expression* 中的周数。如果 *integer-expression* 是负数，则从日期时间值中减去相应的周数。如果提供整数表达式，必须将 *datetime-expression* 类型显式转换为日期时间数据类型。

有关转换数据类型的信息，请参阅 ""CAST 函数 [数据类型转换]" 第 125 页 "。

示例

下列语句返回值 8，表示第二个日期比第一个日期晚八周。建议使用第二种形式 (DATEDIFF)。

```
SELECT WEEKS( '1999-07-13 06:07:12',
   '1999-09-13 10:07:12' )
SELECT DATEDIFF( week,
   '1999-07-13 06:07:12',
   '1999-09-13 10:07:12' )
```

下面的语句返回值 104 270。

```
SELECT WEEKS( '1998-07-13 06:07:12' )
```

下列语句返回时间戳 1999-06-16 21:05:07.0。建议使用第二种形式 (DATEADD)。

```
SELECT WEEKS( CAST( '1999-05-12 21:05:07'
AS TIMESTAMP ), 5)
SELECT DATEADD( week, '1999-05-12 21:05:07' )
```

用法

周中两个日期的差别就是两个日期之间星期日的数目。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

YEARS 函数 [日期和时间]

作用

给定两个日期，此函数返回二者之间的整数年数。对于此用途，建议改用 "DATEDIFF 函数 [日期和时间]" 第 140 页。

给定一个日期，此函数返回年份。对于此用途，建议改用 "DATEPART 函数 [日期和时间]" 第 143 页。

给定一个日期和整数，此函数将整数年数添加到指定的日期中。对于此用途，建议改用 "DATEADD 函数 [日期和时间]" 第 139 页。

语法 1 返回整数。语法 2 返回时间戳。

语法 1

YEARS ([*datetime-expression*,] *datetime-expression* **)**

语法 2

YEARS (*datetime-expression*, *integer-expression* **)**

参数

datetime-expression　日期和时间。

integer-expression　要添加到 *datetime-expression* 中的年数。如果 *integer-expression* 是负数，则从日期时间值中减去相应的年数。如果提供整数表达式，必须将 *datetime-expression* 类型显式转换为日期时间数据类型。

有关转换数据类型的信息，请参阅 ""CAST 函数 [数据类型转换]" 第 125 页 "。

示例

下面两个语句都返回 -4

```
SELECT YEARS( '1998-07-13 06:07:12',
              '1994-03-13 08:07:13' )
SELECT DATEDIFF( year,
   '1998-07-13 06:07:12',
   '1994-03-13 08:07:13' )
```

下列语句返回 1998。

```
SELECT YEARS( '1998-07-13 06:07:12' )
SELECT DATEPART( year, '1998-07-13 06:07:12' )
```

下列语句返回 300 年后的特定日期。

```
SELECT YEARS(
   CAST( '1998-07-13 06:07:12' AS TIMESTAMP ),
   300 )
SELECT DATEADD( year, 300,
   '1998-07-13 06:07:12' )
```

用法 YEARS 值是从两个日期之间年中第一天的数目计算出的。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

YMD 函数 [日期和时间]

作用 返回对应于给定年、月、日的日期值。值可以是从 -32768 到 32767 的小整数。

语法 **YMD (** *integer-expression*, *integer-expression*, *integer-expression* **)**

参数 **integer expression** 年。

integer expression 表示月份的数字。如果月份超出 1-12 的范围，则相应地调整年份。

integer expression 表示日的数字。日可以是任意整数，将相应地调整日期。

示例 下面的语句返回值 1998-06-12。

```
SELECT YMD( 1998, 06, 12 )
```

如果值超出它们的正常范围，则相应地调整日期。例如，下面的语句返回值 2000-03-01。

```
SELECT YMD( 1999, 15, 1 )
```

标准和兼容性

- **SQL/92.** 供应商扩展。
- **SQL/99.** 供应商扩展。
- **Sybase.** 与 Adaptive Server Enterprise 兼容

第 4 章

SQL 语句

关于本章

本章提供 Adaptive Server Anywhere 用户可用的 SQL 语句的详细说明，包括只能在 Embedded SQL 或 Interactive SQL 中使用的一些语句。

本章包括一个按字母顺序排序的 SQL 语句列表。

使用 SQL 语句参考

本节描述记录 SQL 语句时使用的一些约定。

SQL 语法中的常见元素

本节列出了在许多 SQL 语句的语法中出现的语言元素。

有关此处描述的元素的更多信息，请参阅 "" 标识符 " 第 8 页 "、 ""SQL 数据类型 " 第 59 页 "、 "" 搜索条件 " 第 26 页 "、 "" 表达式 " 第 17 页 " 或 "" 字符串 " 第 10 页 "。

- **column-name** 表示列名称的标识符。
- **condition** 取值为 TRUE、 FALSE 或 UNKNOWN 的表达式。
- **connection-name** 表示活动连接名称的字符串。
- **data-type** 存储数据类型。
- **expression** 表达式。
- **filename** 包含文件名的字符串。
- **hostvar** C 语言变量，声明为前面有一个冒号的主机变量。有关更多信息，请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > 使用主机变量 "。
- **indicator-variable** 类型为 **short int** 的另一个主机变量，其紧跟在标准主机变量之后。它的前面也必须有一个冒号。指示符变量用于向数据库传入和从数据库传出 NULL 值。
- **number** 任何数字序列，后跟一个可选的小数部分，前面有一个可选的负号。数字后面还可以有一个 E 和一个指数。例如，

```
42
-4.038
.001
3.4e10
1e-10
```

- **owner**　表示拥有数据库对象的用户 ID 的标识符。
- **role-name**　表示外键角色名称的标识符。
- **savepoint-name**　表示保存点名称的标识符。
- **search-condition**　取值为 TRUE、 FALSE 或 UNKNOWN 的条件。
- **special-value**　" 特殊值 " 第 37 页中描述的特殊值之一。
- **statement-label**　表示循环或复合语句标签的标识符。
- **table-list**　表名列表，可以包含相关名。

  有关更多信息，请参阅 "FROM 子句 " 第 484 页。
- **table-name**　表示表名的标识符。
- **userid**　表示用户名的标识符。
- **variable-name**　表示变量名的标识符。

语法约定

在 SQL 语法说明中，使用了以下约定：

- **关键字**　：所有 SQL 关键字的显示方式与以下示例中的 ALTER TABLE 类似：

  ALTER TABLE [*owner*.]*table-name*
- **占位符**　：必须替换为相应的标识符或表达式的项，其显示方式与以下示例中的 *owner* 和 *table-name* 类似。

  ALTER TABLE [*owner*.]*table-name*
- **重复项**　：对于重复项列表，只显示列表中的一个元素，后跟省略号 （三个句点），如以下示例中的 *column-constraint*：

  ADD *column-definition* [*column-constraint, …*]

允许指定一个或多个列表元素。如果指定了多个元素，则必须用逗号将它们隔开。

- **可选部分** ：语句的可选部分放在方括号内。

  RELEASE SAVEPOINT [*savepoint-name*]

  上面的方括号表示 *savepoint-name* 是可选的。不应键入方括号。

- **选项** ：如果不必选择项目列表的项目或者只能选中一个，则用垂直条分隔这些项目，并将列表放在方括号内。

  [**ASC** | **DESC**]

  例如，可以从 ASC、 DESC 中任选一个，或者一个也不选。不应键入方括号。

- **二选一选项** ：如果必须明确选择一个选项，则将替换选项放在大括号内。

  [**QUOTES** { **ON** | **OFF** }]

  如果选择了 QUOTES 选项，则必须提供 ON 和 OFF 两者之一。不应键入方括号和大括号。

- **一个或多个选项** ：如果选择多个选项，则用逗号将选项隔开。

  { **CONNECT**, **DBA**, **RESOURCE** }

语句适用性指示符

某些语句标题的后面有一个指示符 （括在方括号内），用于指示语句的使用环境。这些指示符如下：

- **[ESQL]** 表示语句用于嵌入式 SQL 中。
- **[Interactive SQL]** 表示语句只能用于 Interactive SQL 中。
- **[SP]** 表示语句用于存储过程、触发器或批处理语句中。

- **[T-SQL]**　表示实现语句的目的是为了与 Adaptive Server Enterprise 兼容。在某些情况下，语句不能用于非 Transact-SQL 格式的存储过程。在其它情况下，如果不存在 Transact-SQL 兼容问题，建议使用与 SQL/92 标准接近的替换语句。
- **[MobiLink]**　表示语句仅用于 MobiLink 客户端。
- **[SQL Remote]**　表示语句只能用于 SQL Remote。

如果使用两组括号，表示语句可用于两种环境。例如， [ESQL][SP] 表示语句既可以用于嵌入式 SQL，也可以用于存储过程。

ALLOCATE DESCRIPTOR 语句 [ESQL]

说明

此语句用于为 SQL 描述符区域 (SQLDA) 分配空间。

语法

ALLOCATE DESCRIPTOR *descriptor-name*
[**WITH MAX** { *integer* | *hostvar* }]

descriptor-name : string

参数

WITH MAX 子句 允许您指定描述符区内的变量数。缺省大小为 1。在执行读取操作或任何访问描述符区内的数据的语句之前，仍必须调用 **fill_sqlda** 为实际的数据项分配空间。

用法

为描述符区 (SQLDA) 分配空间。使用此语句前，必须在 C 代码中声明以下内容：

```
struct sqlda * descriptor_name
```

权限

无。

副作用

无。

另请参阅

"DEALLOCATE DESCRIPTOR 语句 [ESQL]" 第 423 页

"ASA 编程指南 > 嵌入式 SQL 编程 > SQL 描述符区域 (SQLDA)"

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** 受 Open Client/Open Server 支持。

示例

下面的示例程序包括 ALLOCATE DESCRIPTOR 语句用法的一个示例。

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
EXEC SQL INCLUDE SQLCA;
#include <sqldef.h>
EXEC SQL BEGIN DECLARE SECTION;
int      x;
short    type;
```

```
int      numcols;
char     string[100];
a_SQL_statement_number  stmt = 0;
EXEC SQL END DECLARE SECTION;
int main(int argc, char * argv[]){
   struct sqlda *      sqlda1;
   if( !db_init( &sqlca ) ) {
      return 1;
   }
   db_string_connect( &sqlca,
   "UID=DBA;PWD=SQL;DBF=d:\\DB Files\\sample.db");
   EXEC SQL ALLOCATE DESCRIPTOR sqlda1 WITH MAX 25;
   EXEC SQL PREPARE :stmt FROM
      'SELECT * FROM employee';
   EXEC SQL DECLARE curs CURSOR FOR :stmt;
   EXEC SQL OPEN curs;
   EXEC SQL DESCRIBE :stmt into sqlda1;
   EXEC SQL GET DESCRIPTOR sqlda1 :numcols=COUNT;
   // how many columns?
   if( numcols > 25 ) {
      // reallocate if necessary
      EXEC SQL DEALLOCATE DESCRIPTOR sqlda1;
      EXEC SQL ALLOCATE DESCRIPTOR sqlda1
         WITH MAX :numcols;
      EXEC SQL DESCRIBE :stmt into sqlda1;
   }
   type = DT_STRING; // change the type to string
   EXEC SQL SET DESCRIPTOR sqlda1 VALUE 2 TYPE = :type;
   fill_sqlda( sqlda1 );
   // allocate space for the variables
   EXEC SQL FETCH ABSOLUTE 1 curs
      USING DESCRIPTOR sqlda1;
   EXEC SQL GET DESCRIPTOR sqlda1
      VALUE 2 :string = DATA;
   printf("name = %s", string );
   EXEC SQL DEALLOCATE DESCRIPTOR sqlda1;
   EXEC SQL CLOSE curs;
   EXEC SQL DROP STATEMENT :stmt;
   db_string_disconnect( &sqlca, " );
   db_fini( &sqlca );
   return 0;
}
```

ALTER DATABASE 语句

说明

此语句用于升级用软件的早期版本创建的数据库，或者向任何数据库中添加 Java 或 JConnect 支持。

语法 1

ALTER DATABASE
[**UPGRADE** [**JAVA** { **ON** | **OFF** | **JDK** { **'1.1.8'** | **'1.3'** } }]
[**JCONNECT** { **ON** | **OFF** }]
| **REMOVE JAVA**]

语法 2

ALTER DATABASE
{ **CALIBRATE** [**SERVER**]
| **CALIBRATE DBSPACE** *dbspace-name*
| **CALIBRATE TEMPORARY DBSPACE**
| **RESTORE DEFAULT CALIBRATION**
}

语法 3

ALTER DATABASE *dbfile*
MODIFY [**TRANSACTION**] **LOG**
{ { **OFF** | **ON** } { *log-name* | *log-name* **MIRROR** *mirror-name* | **MIRROR** *mirror-name* } }
[**KEY** *key* [**ALGORITHM** *algorithm*]]

参数

JAVA 子句 控制升级数据库中的 Java 支持。

- 指定 **JAVA ON** 通过向系统表中添加缺省 Sybase 运行时 Java 类的条目，启用数据库中的 Java 支持。缺省类是 JDK 1.3 类。
- 指定 **JAVA OFF** 以禁止添加 Java 支持。设置 **JAVA OFF** 不会从数据库中移除 Java 支持。
- 指定 **JAVA JDK '1.1.8'** 或 **JAVA JDK '1.3'** 以显式安装对指定 JDK 版本的支持。可以将数据库升级到更高的 JDK 版本，但不能降到更低的版本。

  对于 JDK 1.1.8，上述类保存在 SQL Anywhere 目录下的 `java\1.1\classes.zip` 中。对于 JDK 1.3，它们保存在 `java\1.3\rt.jar` 中。

缺省行为是 **JAVA OFF**。

如果添加数据库中的 Java，必须重新启动数据库才能使用它。

数据库中的 Java 是单独授权的组件。有关更多信息，请参阅 "ASA 编程指南 > 介绍数据库中的 Java"。

JCONNECT 子句 如果希望使用 Sybase jConnect JDBC 驱动程序访问系统目录信息，需要将 JCONNECT 指定为 ON。如果希望排除 JConnect 系统对象，将 JCONNECT 指定为 OFF。只要不访问系统目录信息，仍可以使用 JDBC。缺省为包括 JConnect 支持 (JCONNECT ON)。

设置 JCONNECT OFF 并不会从数据库中删除 JConnect 支持。

REMOVE JAVA 子句 从数据库中删除数据库中的 Java。该操作使数据库像是用 JAVA OFF 创建的一样。当发出此语句时，数据库中的 Java 一定不能正在被使用。执行此语句前必须从数据库中删除所有 Java 类。此语句不会删除引用 Java 对象的存储过程和触发器，这些对象的存在并不会在 ALTER DATABASE 语句中引发错误。

用法

语法 1 可以使用 ALTER DATABASE 语句取代升级实用程序来升级数据库。使用 ALTER DATABASE UPGRADE 后，应该关闭数据库。（升级实用程序会自动完成此操作。）

升级前备份
与任何软件一样，对于数据库，建议在升级前先进行备份。

ALTER DATABASE 可以用于升级用软件的早期版本创建的数据库。这也适用于维护版本，因此在软件的 7.0.3 版中，可以使用 ALTER DATABASE 语句，将比如用 7.0.2 版创建的数据库升级到 7.0.3 标准版。

通常情况下，数据库在次版本中的改动仅限于增加的数据库选项和不重要的系统表改动。

用于升级数据库时， ALTER DATABASE 进行以下改动：

- 将系统表升级为当前版本。
- 添加任何新的数据库选项。
- 删除并重新创建所有系统存储过程。

如果数据库是用软件的当前版本创建的，也可以使用 ALTER DATABASE 只添加数据库中的 Java 或 JConnect 特性。

有关添加 Java 支持的更多信息，请参阅 "ASA 编程指南 > 在数据库中使用 Java > 为数据库启用 Java"。有关向第 6 版数据库添加 JConnect 支持的更多信息，请参阅 "ASA 编程指南 > 使用 JDBC 访问数据 > 在数据库中安装 jConnect 系统对象 "。

不是所有特性都可用
要求在物理上重新组织数据库文件的特性并不能通过 ALTER DATABASE 提供。这样的特性包括索引增强和数据库存储更改。要想从这些增强中受益，必须卸载然后重装数据库。

有关更多信息，请参阅 "ASA SQL 用户指南 > 导入和导出数据 > 重建数据库 "。

语法 2 也可以使用 ALTER DATABASE 对优化程序使用的 I/O 成本模型进行重新校准。这将更新磁盘传送时间 (DTT) 模型，它是成本模型使用的磁盘 I/O 的数学模型。

在正常操作中，成本模型使用内置的缺省 DTT 模型。该缺省模型是根据典型的硬件和配置设计的。少数情况下需要使用专门的硬件 （如非标准磁盘驱动器），这时如果出现了性能问题，用基于具体设置的模型覆盖缺省模型也许会有帮助。但是，通常建议不要改变缺省模型。

当重新校准 I/O 成本模型时，服务器不能用于其它用途。另外，计算机上的所有其它活动必须是空闲的。重新校准服务器是一种资源开销巨大的操作，可能需要一些时间完成。

当使用 CALIBRATE [SERVER] 参数时，将校准除临时 dbspace 外的所有 dbspace。使用 CALIBRATE TEMPORARY DBSPACE 可校准临时 dbspace。使用 CALIBRATE DBSPACE *dbspace-name* 可校准单个 dbspace。使用 RESTORE DEFAULT CALIBRATION 可恢复缺省 DTT 模型。

语法 3 您可以使用 ALTER DATABASE 语句来更改事务日志及镜像与数据库文件关联的名称。这些更改与事务日志 (dblog) 实用程序所做的更改相同。您可以在连接着该 utility 数据库或另一数据库 （具体是哪个数据库取决于 -gu 选项的设置）时执行此语句。如果您要更改某个加密的数据库的事务日志或镜像日志，则必须指定一个密钥和加密算法。

权限

必须具有 DBA 权限，并且必须是唯一的数据库连接。

对于 REMOVE JAVA，当发出此语句时，数据库中的 Java 一定不能正在被使用。

在 Windows CE 上不受支持。

数据库中的 Java 是单独授权的组件。

副作用

自动提交

另请参阅

"CREATE DATABASE 语句 " 第 314 页

"ASA 数据库管理指南 > 数据库管理实用程序 > Upgrade 实用程序 "

"CREATE STATISTICS 语句 " 第 366 页

"ASA 数据库管理指南 > 数据库管理实用程序 > Transaction Log 实用程序 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的示例将升级一个数据库以启用 Java 操作。

```
ALTER DATABASE UPGRADE
JAVA ON
```

下面的示例将与 `asademo.db` 关联的事务日志文件名设置为 `newdemo.log`。

```
ALTER DATABASE 'asademo.db'
MODIFY LOG ON 'newdemo.log'
```

ALTER DBSPACE 语句

说明

使用此语句可为 dbspace 或事务日志预分配空间，也可以在重命名或移动数据库文件时使用此语句。

语法

ALTER DBSPACE { *dbspace-name* | **TRANSLOG** }
{ **ADD** *number [* **PAGES** | **KB** | **MB** | **GB** | **TB** *]*
| **RENAME** *filename-string* }

参数

TRANSLOG 您提供一个专用的 dbspace 名称 TRANSLOG 为事务日志预分配磁盘空间。如果估计事务日志增长速度很快，则预分配可以提高性能。例如，当处理许多二进制大对象 (BLOB)（如位图）时，可能需要使用此特性。

ADD 子句 带 ADD 子句的 ALTER DBSPACE 为 dbspace 预分配磁盘空间。它使相应的数据库文件扩展指定的大小，以页、千字节 (**KB**)、兆字节 (**MB**)、千兆字节 (**GB**) 或兆兆字节 (**TB**) 为单位。如果您不指定单位， **PAGES** 为缺省单位。创建数据库后，数据库的页面大小是固定的。

如果未预分配空间，则在需要空间时，数据库文件每次扩展 256 KB。预分配空间可以提高装入大量数据时的性能，而且还可以使数据库文件在文件系统中保持更好的连接性。

RENAME 子句 如果重命名主文件以外的数据库文件或将其移到不同的目录或设备，可以使用带 RENAME 子句的 ALTER DBSPACE，以确保启动数据库时 Adaptive Server Anywhere 能够找到新文件。

对主 dbspace SYSTEM 使用带 RENAME 的 ALTER DBSPACE 无效。

用法

每个数据库保存在一个或多个文件中。 dbspace 是一个附加文件，它具有与每个数据库文件关联的逻辑名称，用于保存比主数据库文件自己所能容纳的数据更多的数据。 ALTER DBSPACE 修改主 dbspace（也称为根文件）或附加 dbspace。数据库的 dbspace 名称保存在 SYSFILE 系统表中。主数据库文件的 dbspace 名称为 SYSTEM。

当启动多文件数据库时，启动行或 ODBC 数据源描述向 Adaptive Server Anywhere 说明了主数据库文件的位置。主数据库文件中保存着系统表。Adaptive Server Anywhere 在这些系统表中查找其它 dbspace 的位置，然后逐个打开它们。

权限 必须拥有 DBA 权限。必须是唯一的数据库连接。

副作用 自动提交。

另请参阅 "CREATE DBSPACE 语句 " 第 319 页

"ASA SQL 用户指南 > 使用数据库对象 > 使用数据库 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例 下面的示例将 SYSTEM dbspace 的大小增加 200 页。

```
ALTER DBSPACE system
ADD 200
```

下面的示例将 SYSTEM dbspace 的大小增加 400 兆字节。

```
ALTER DBSPACE system
ADD 400 MB
```

下面的示例更改与 *system_2* dbspace 关联的文件名：

```
ALTER DBSPACE system_2
RENAME 'e:\db\dbspace2.db'
```

ALTER EVENT 语句

说明

此语句用于更改事件的定义或者与之关联的、自动执行预定义操作的处理程序。另外，使用此语句还可以变更调度操作的定义。

语法

ALTER EVENT *event-name*
[**DELETE TYPE** | **TYPE** *event-type*]
{ **WHERE** { *trigger-condition* | **NULL** }
| { **ADD** | **MODIFY** | **DELETE** } **SCHEDULE** *schedule-spec*
}
[**ENABLE** | **DISABLE**]
[[**MODIFY**] **HANDLER** *compound-statement* | **DELETE HANDLER** }

event-type :
BackupEnd | **"Connect"**
| **ConnectFailed** | **DatabaseStart**
| **DBDiskSpace** | **"Disconnect"**
| **GlobalAutoincrement** | **GrowDB**
| **GrowLog** | **GrowTemp**
| **LogDiskSpace** | **"RAISERROR"**
| **ServerIdle** | **TempDiskSpace**

trigger-condition :
event_condition(*condition-name* **)** { **=** | **<** | **>** | **!=** | **<=** | **>=** } *value*

schedule-spec :
[*schedule-name*]
{ **START TIME** *start-time* | **BETWEEN** *start-time* **AND** *end-time* }
[**EVERY** *period* { **HOURS** | **MINUTES** | **SECONDS** }]
[**ON** { **(** *day-of-week*, … **)** | **(** *day-of-month*, … **)** }]
[**START DATE** *start-date*]

event-name | *schedule-name* : *identifier*

day-of-week : *string*

value | *period* | *day-of-month* : *integer*

start-time | *end-time* : *time*

start-date : *date*

参数

DELETE TYPE 子句 删除事件与事件类型的关联。

ADD | MODIFY | DELETE SCHEDULE 子句 更改调度的定义。在任一 ALTER EVENT 语句中只能变更一个调度。

WHERE 子句 WHERE NULL 选项删除条件。

有关大多数参数的说明，请参阅 ""CREATE EVENT 语句 " 第 326 页 "。

用法

此语句允许您变更用 CREATE EVENT 创建的事件定义。可能的用法包括：

- 在开发期间可以使用 ALTER EVENT 更改事件处理程序。
- 在开发阶段，可能需要定义和测试没有触发条件或调度的事件处理程序，然后在事件处理程序完成后，使用 ALTER EVENT 添加执行条件。
- 可能需要通过禁用事件暂时禁用事件处理程序。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"BEGIN 语句 " 第 288 页

"CREATE EVENT 语句 " 第 326 页

ALTER FUNCTION 语句

说明

此语句用于修改函数。必须在 ALTER FUNCTION 语句中包括整个新函数。

语法 1

ALTER FUNCTION [*owner*.]*function-name*
function-definition

function-definition:
同 CREATE FUNCTION 语法中函数名之后的部分

语法 2

ALTER FUNCTION [*owner*.]*function-name* **SET HIDDEN**

用法

语法 1 除第一个单词外， ALTER FUNCTION 语句的语法与 CREATE FUNCTION 语句的语法相同。两个语法版本的 CREATE FUNCTION 语句都可以如此变更。

系统维护函数的现有权限，因此不必重新分配。如果执行了 DROP FUNCTION 和 CREATE FUNCTION，则必须重新分配执行权限。

语法 2 可以使用 SET HIDDEN 使关联的函数的定义变成乱码，使之不可读。可以卸载该函数，然后将其重装到其他数据库中。

此设置是不可逆的。
如果您还将再次需要起始源，则必须将它保存在数据库外。

如果使用 SET HIDDEN，则使用存储过程调试程序进行调试将不会显示函数定义，也无法通过过程分析获得函数定义。

权限

必须是函数的所有者或者是 DBA。

副作用

自动提交。

另请参阅

"CREATE FUNCTION 语句 " 第 337 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 隐藏过程、函数、触发器和视图的内容 "

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

ALTER PROCEDURE 语句

说明

此语句用于修改过程，或启用和禁用 Sybase Replication Server 的复制过程。必须在 ALTER PROCEDURE 语句中包括整个新过程。

语法 1

ALTER PROCEDURE [*owner*.]*procedure-name*
procedure-definition

procedure-definition:
同 CREATE PROCEDURE 语法中函数名之后的部分

语法 2

ALTER PROCEDURE [*owner*.]*procedure-name*
REPLICATE { **ON** | **OFF** }

语法 3

ALTER PROCEDURE [*owner*.]*procedure-name* **SET HIDDEN**

用法

语法 1 除第一个单词外， ALTER PROCEDURE 语句的语法与 CREATE PROCEDURE 语句的语法相同。两个语法版本的 CREATE PROCEDURE 语句都可以如此变更。

系统维护过程的现有权限，因此不必重新分配。如果执行了 DROP PROCEDURE 和 CREATE PROCEDURE，则必须重新分配执行权限。

语法 2 如果要使用 Sybase Replication Server 将一个过程复制到其它站点，则必须对该过程设置 REPLICATE ON。

ALTER PROCEDURE 语句的语法 2 与 Adaptive Server Enterprise 中 *sp_setreplicate* 或 *sp_setrepproc 'table'* 系统过程具有相同的效果。

语法 3 可以使用 SET HIDDEN 使关联的过程的定义变成乱码，使之不可读。可以卸载该过程，然后将其重装到其他数据库中。

此设置是不可逆的。
如果您还将再次需要起始源，则必须将它保存在数据库外。

如果使用 SET HIDDEN，则使用存储过程调试程序进行调试将不会显示过程定义，也无法通过过程分析获得过程定义。

语法 2 不可与语法 1 合并使用。语法 3 不可与语法 1 合并使用，也不可与语法 2 合并使用。

权限

必须是过程的所有者或者是 DBA。

副作用

自动提交。

另请参阅

"CREATE PROCEDURE 语句 " 第 347 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 隐藏过程、函数、触发器和视图的内容 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

ALTER PUBLICATION 语句

说明

此语句可用于更改发布。在 MobiLink 中，发布可以标识 Adaptive Server Anywhere 远程数据库中的同步数据。在 SQL Remote 中，发布标识统一数据库和远程数据库中的复制数据。

语法

ALTER PUBLICATION [*owner*.]*publication-name alterpub-clause, …*

alterpub-clause:
  ADD TABLE *article-description*
| **MODIFY TABLE** *article-description*
| { **DELETE** | **DROP** } **TABLE** [*owner*.]*table-name*
| **RENAME** *publication-name*

owner, *publication-name*, *table-name* : *identifier*

article-description :
  table-name [**(** *column-name*, … **)**]
  [**WHERE** *search-condition*]
  [**SUBSCRIBE BY** *expression*]

用法

此语句仅适用于 MobiLink 和 SQL Remote。

ALTER PUBLICATION 语句变更数据库中的发布。一个表中的发布构成称为**项目**。通过添加、修改或删除项目，或者通过重命名发布，可以对发布进行更改。如果修改某个项目，必须输入对该修改项目的完整说明。

应使用 ALTER SYNCHRONIZATION SUBSCRIPTION 语句或 CREATE SYNCHRONIZATION SUBSCRIPTION 语句中的 ADD OPTION 子句为 MobiLink 发布设置选项。

权限

必须具有 DBA 权限，或者是发布的所有者。要求可以对语句中涉及的所有表进行独占访问。

副作用

自动提交。

另请参阅

"CREATE PUBLICATION 语句 " 第 357 页

"DROP PUBLICATION 语句 " 第 450 页

"ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 265 页

"CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 375 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_add_article 过程 "

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_add_article_col 过程 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

下列语句将 *customer* 表添加到 *pub_contact* 发布中。

```
ALTER PUBLICATION pub_contact
   ADD TABLE customer
```

ALTER REMOTE MESSAGE TYPE 语句 [SQL Remote]

说明

对于一个已创建的消息类型，此语句用于更改发布者的消息系统，或者更改给定消息系统的发布者地址。

语法

ALTER REMOTE MESSAGE TYPE *message-system*
ADDRESS *address*

message-system: **FILE** | **FTP** | **MAPI** | **SMTP** | **VIM**

address: *string*

参数

message-system SQL Remote 支持的消息系统之一。它必须是下面的一个值：FILE | FTP | MAPI | SMTP | VIM

address 一个字符串，包含指定消息系统的有效地址。

用法

此语句更改给定消息类型的发布者地址。

消息代理通过受支持的消息链接之一，从数据库发送出去的消息。抽取实用程序在远程数据库中执行 GRANT CONSOLIDATE 语句时使用该地址。

在指定的消息系统中，地址是发布者的地址。如果是电子邮件系统，则地址字符串必须是有效的电子邮件地址。如果是文件共享系统，则地址字符串是 SQLREMOTE 环境变量指定的目录的子目录，如果未设置该变量，则是当前目录的子目录。在远程数据库中，可以覆盖 GRANT CONSOLIDATE 语句中的该设置。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"CREATE REMOTE MESSAGE TYPE 语句 [SQL Remote]" 第 360 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_remote_type 过程 "

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99**　供应商扩展。

示例　　下面的语句将 FILE 消息链接的发布者地址更改为 **new_addr**。

```
CREATE REMOTE MESSAGE TYPE file
ADDRESS 'new_addr'
```

ALTER SERVER 语句

说明

此语句用于修改远程服务器的属性。

语法

ALTER SERVER *server-name*
[**CLASS** '*server-class*']
[**USING** '*connection-info*']
[**CAPABILITY** '*cap-name*' { **ON** | **OFF** }]

server-class :
ASAJDBC | **ASEJDBC**
| **ASAODBC** | **ASEODBC**
| **DB2ODBC** | **MSSODBC**
| **ORAODBC** | **ODBC**

connection-info :
machine-name:*port-number*[/*dbname*] | *data-source-name*

参数

CLASS 子句 指定 CLASS 子句以更改服务器类。

有关服务器类和如何配置服务器的更多信息，请参阅 "ASA SQL 用户指南 > 用于进行远程数据访问的服务器类 "。

USING 子句 指定 USING 子句以更改服务器连接信息。有关 *connection-info* 的更多信息，请参阅 ""CREATE SERVER 语句 " 第 364 页 "。

CAPABILITY 子句 CAPABILITY 子句将服务器功能设置为 ON 或 OFF。服务器功能存储在系统表 *syscapability* 中。这些功能的名称存储在系统表 *syscapabilityname* 中。直到与远程服务器建立连接时，该远程服务器在 *syscapability* 表中才包含条目。首次连接时， Adaptive Server Anywhere 向服务器询问它的功能，然后填充 *syscapability* 表。对于后面的连接，它从该表中获取服务器的功能。

通常情况下，不需要变更服务器的功能。可能需要变更属于 ODBC 类的通用服务器的功能。

用法

ALTER SERVER 语句修改服务器的属性。直到下次连接到远程服务器时，这些更改才生效。

权限

必须有 RESOURCE 权限。

副作用 自动提交。

另请参阅

"CREATE SERVER 语句 " 第 364 页

"DROP SERVER 语句 " 第 452 页

"ASA SQL 用户指南 > 用于进行远程数据访问的服务器类 "

"ASA SQL 用户指南 > 访问远程数据 > 远程数据访问疑难解答 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Open Client/Open Server 支持。

示例 下面的示例更改名为 *ase_prod* 的 Adaptive Server 的服务器类，使该服务器与 Adaptive Server Anywhere 的连接基于 ODBC。其数据源名称为 `ase_prod`。

```
ALTER SERVER ase_prod
CLASS 'ASEODBC'
USING 'ase_prod'
```

- 下面的示例更改服务器 *infodc* 的功能。

```
ALTER SERVER infodc
CAPABILITY 'insert select' OFF
```

ALTER SYNCHRONIZATION DEFINITION 语句（不建议使用）

说明

此语句变更 Adaptive Server Anywhere 远程数据库中的 MobiLink 同步定义。不建议使用此命令。应当使用 ALTER PUBLICATION 或 ALTER SYNCHRONIZATION SUBSCRIPTION 取代它。

语法

ALTER SYNCHRONIZATION DEFINITION *sync-def-name*
[**SITE** *sync-site-name*,]
[**TYPE** *sync-type*,]
[**ADDRESS** *network-parameters*,]
[**ADD OPTION** *parameter*=*value*, …]
[**MODIFY OPTION** *parameter*=*value*, …]
[**DELETE OPTION** *parameter*, …
| **DELETE ALL OPTION**,]
[**RENAME** *new-sync-def-name*,]
[**ADD TABLE** *article-description*, …]
[**MODIFY TABLE** *article-description*, …]
[**DELETE TABLE** *table-name*, …]

network-parameters: string

article-description:
table-name [**(** *column-name*, … **)**]
[**WHERE** *search-condition*]

value:
string | *integer*

参数

SITE 子句 在 MobiLink 安装中唯一标识此远程数据库的名称。

TYPE 子句 此子句指定同步方法。缺省值是 TCP/IP。也可以选择使用 HTTP 或 HTTPS。

ADDRESS 子句 此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

ADD OPTION、 MODIFY OPTION、 DELETE OPTION 和 DELETE ALL OPTION 子句 这些子句可用于添加、修改、删除选项或删除所有选项。您只需在每个子句中指定一个参数。可以多次指定这些子句以影响多个选项。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

有关选项的列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

输入选项时要小心
数据库引擎将接受您输入的任何选项，不检查它们的有效性。因此，如果拼错了选项，数据库引擎检测不到错误，直到运行 `dbmlsync` 命令执行同步时才会给出错误消息。

RENAME 子句 使用该子句可重命名同步定义。

ADD TABLE、 MODIFY TABLE 和 DELETE TABLE 子句 您以前在 CREATE SYNCHRONIZATION DEFINITION 语句中已指定要上载到统一数据库的内容。使用这些子句可对这些指定进行更改。

用法

在 Adaptive Server Anywhere 版本 7 数据库中，创建将用作 MobiLink 客户端的同步定义。每个定义指定在 MobiLink 设置中唯一标识该 MobiLink 客户端站点的逻辑名称。另外，每个站点指定如何联系 MobiLink 同步服务器以及远程数据库中的哪些数据将与统一数据库同步。

使用 ALTER SYNCHRONIZATION DEFINITION 语句可变更同步定义。

例如，如果希望更改选项，将内存设置为 3 MB，可以按如下所示变更同步定义：

```
ALTER SYNCHRONIZATION DEFINITION mysharedtables
   OPTION memory='3m'
```

权限

必须拥有 DBA 权限。要求可以对语句中涉及的所有表进行独占访问。

副作用

自动提交

另请参阅

"ALTER PUBLICATION 语句 " 第 254 页

"ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 265 页

"CREATE SYNCHRONIZATION DEFINITION 语句 [MobiLink]（不建议使用）" 第 369 页

"DROP SYNCHRONIZATION DEFINITION 语句 [MobiLink]（不建议使用）" 第 457 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例

变更 *mysharedtables* 同步定义。切换到不同的远程数据库并设置不同的选项。另外，表 *Books* 也上载到统一数据库，而表 *People* 中只有两列上载到统一数据库。

```
ALTER SYNCHRONIZATION DEFINITION mysharedtables
   SITE 'new_sync_site',
   ADD OPTION verbose='on',
   MODIFY OPTION memory='3m',
   ADD TABLE Books,
   MODIFY TABLE People(fname, lname);
```

ALTER SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）

说明

此语句用于在 MobiLink 参考数据库中更改站点，在使用 `mlxtract` 实用程序抽取 Adaptive Server Anywhere 远程数据库时将用到此功能。不建议使用此命令。应当使用 ALTER PUBLICATION 或 ALTER SYNCHRONIZATION SUBSCRIPTION 取代它。

语法

ALTER SYNCHRONIZATION SITE *sync-site-name*
[**RENAME** *new-sync-site-name*,]
[**TYPE** *sync-type*,]
[**ADDRESS** *network-parameters*,]
[**ADD OPTION** *parameter*=*value*, …]
[**MODIFY OPTION** *parameter*=*value*, …]
[**DELETE OPTION** *parameter*, …| **DELETE ALL OPTION**]

network-parameters: string

value: *string* | *integer*

参数

SITE 子句 在 MobiLink 安装中唯一标识此远程数据库的名称。

TYPE 子句 此子句指定同步方法。缺省值为 **tcpip**。也可以选择使用 **http** 或 **https**。

ADDRESS 子句 此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

ADD OPTION、 MODIFY OPTION、 DELETE OPTION 和 DELETE ALL OPTION 子句 这些子句可用于添加、修改、删除选项或删除所有选项。您只需在每个子句中指定一个参数。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

用法

仅当通过从 Adaptive Server Anywhere 第 7 版参考数据库中抽取的方式创建 Adaptive Server Anywhere 远程数据库时，才使用同步模板和同步站点。

每个远程数据库从同步站点中创建并存储在参考数据库中。每个同步站点都基于单个同步模板，不过许多站点可以使用单个模板。

使用 ALTER SYNCHRONIZATION SITE 语句可修改同步站点的属性。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"ALTER PUBLICATION 语句 " 第 254 页

"ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 265 页

"CREATE SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）" 第 372 页

"DROP SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）" 第 459 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例

将 *mytemplate* 部署到一个不同的远程数据库并设置不同于模板中的选项。

```
ALTER SYNCHRONIZATION SITE USING mytemplate
   SITE 'new_sync_site',
   ADD OPTION verbose='on'
   MODIFY OPTION memory='3m';
```

ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]

说明

在 Adaptive Server Anywhere 远程数据库中使用此语句可变更 MobiLink 用户对一发布所做预订的属性。

语法

ALTER SYNCHRONIZATION SUBSCRIPTION
TO *publication-name*
[**FOR** *ml_username*, …]
[**TYPE** *sync-type*]
[**ADDRESS** *network-parameters*]
[**ADD OPTION** *option=value*, …]
[**MODIFY OPTION** *option=value*, …]
[**DELETE** { **ALL OPTION** | **OPTION** *option=value*, … }]

ml_username: identifier

network-parameters: *string*

sync-type: **http** | **https** | **tcpip** | **ActiveSync**

value: *string* | *integer*

参数

TO 子句 指定发布名称。

FOR 子句 另指定一个 MobiLink 用户 ID。

省略此子句将更改发布的缺省预订。如果预订此发布的 MobiLink 用户没有设置新值覆盖这些缺省值，将继承这些缺省值。此功能在从参考数据库中抽取远程数据库时非常有用。

TYPE 子句 此子句指定同步中使用的通信协议。缺省协议为 **tcpip**。

ADDRESS 子句 此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

ADD OPTION、 MODIFY OPTION、 DELETE OPTION 和 DELETE ALL OPTION 子句 这些子句可用于添加、修改、删除选项或删除所有选项。您只需在每个子句中指定一个参数。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

用法

此语句可用于在 MobiLink 远程数据库或参考数据库中更改同步预订。

权限

必须拥有 DBA 权限。要求可以对发布中涉及的所有表进行独占访问。

副作用

自动提交。

另请参阅

"CREATE PUBLICATION 语句 " 第 357 页

"CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 8.0 版。

示例

为销售发布创建包含缺省预订值的缺省预订 （通过省略 FOR 子句）。指出 MobiLink 同步服务器的地址并指定只有 Certicom 根证书受信任。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
   TO sales_publication
   ADDRESS 'host=test.internal;port=2439;
      security=ecc_tls'
   OPTION memory='2m';
```

为 MobiLink 用户 *ml_user1* 预订销售发布。将内存选项设置为 3 MB，而不是缺省发布中指定的值。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
   TO sales_publication
   FOR 'ml_user1'
   OPTION memory='3m';
```

ALTER SYNCHRONIZATION TEMPLATE 语句 [MobiLink]（不建议使用）

说明

此语句用于变更 MobiLink 参考数据库中的模板，在使用 `mlxtract` 命令行实用程序抽取 Adaptive Server Anywhere 远程数据库时将用到此功能。不建议使用此命令。请使用同步发布和预订命令。

语法

ALTER SYNCHRONIZATION TEMPLATE *sync-template-name*,
[**TYPE** *sync-type*,]
[**ADDRESS** *network-parameters*,]
[**ADD OPTION** *parameter*=*value*, …]
[**MODIFY OPTION** *parameter*=*value*, …]
[**DELETE OPTION** *parameter*, …| **DELETE ALL OPTION**,]
[**RENAME** *new-sync-def-name*,]
[**ADD TABLE** *article-description*, …]
[**MODIFY TABLE** *article-description*, …]
[**DELETE TABLE** *table-name*, …]

article-description:
table-name [**(** *column-name*, … **)**]
[**WHERE** *search-condition*]

value:
string | *integer*

参数

TYPE 子句　此子句指定同步方法。缺省值为 **tcpip**。也可以选择使用 **http** 或 **https**。

ADDRESS 子句　此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

ADD OPTION、 MODIFY OPTION、 DELETE OPTION 和 DELETE ALL OPTION 子句　这些子句可用于添加、修改、删除选项或删除所有选项。您只需在每个子句中指定一个参数。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

RENAME 子句 使用该子句可重命名同步定义。

ADD TABLE、 MODIFY TABLE 和 DELETE TABLE 子句 您以前在 CREATE SYNCHRONIZATION DEFINITION 语句中已指定要上载到统一数据库的内容。使用这些子句可对这些指定进行更改。

用法

仅当通过从 Adaptive Server Anywhere 第 7 版参考数据库中抽取的方式创建 Adaptive Server Anywhere 远程数据库时，才使用同步模板和同步站点。

每个远程数据库从同步站点中创建并存储在参考数据库中。每个同步站点都基于单个同步模板，不过许多站点可以使用单个模板。

使用 ALTER SYNCHRONIZATION TEMPLATE 语句可修改同步模板的属性。

权限

必须拥有 DBA 权限。要求可以对语句中涉及的所有表进行独占访问。

副作用

自动提交

另请参阅

"ALTER PUBLICATION 语句 " 第 254 页

"ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 265 页

"CREATE SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） " 第 377 页

"DROP SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） " 第 461 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例

变更 *mytemplate* 同步模板。设置不同的选项。指定表 *Books* 也上载到统一数据库。 *People* 表中只有两个指定的列上载到统一数据库。

```
ALTER SYNCHRONIZATION TEMPLATE mytemplate
   ADD OPTION verbose='on',
   MODIFY OPTION memory='3m',
   ADD TABLE Books,
   MODIFY TABLE (People(fname,lname));
```

ALTER SYNCHRONIZATION USER 语句 [MobiLink]

说明

此语句可用于在 Adaptive Server Anywhere 远程数据库中更改 MobiLink 用户的属性。

语法

ALTER SYNCHRONIZATION USER *ml_username*
[**TYPE** *sync-type*]
[**ADDRESS** *network-parameters*]
[**ADD OPTION** *option*=*value*, …]
[**MODIFY OPTION** *option*=*value*, …]
[**DELETE** { **ALL OPTION** | **OPTION** *option* }]

ml_username: identifier

network-parameters: *string*

sync-type: **http** | **https** | **tcpip** | **ActiveSync**

value: *string* | *integer*

参数

TYPE 子句 此子句指定同步中使用的通信协议。

ADDRESS 子句 此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

ADD OPTION、 MODIFY OPTION、 DELETE OPTION 和 DELETE ALL OPTION 子句 这些子句可用于添加、修改、删除选项或删除所有选项。您只需在每个子句中指定一个参数。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

用法

此语句用于变更 MobiLink 远程数据库中同步用户的属性。

权限

必须拥有 DBA 权限。要求可以对发布中涉及的所有表进行独占访问。

副作用

自动提交。

另请参阅

"ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 265 页

"CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页

"CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 375 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 8.0 版。

ALTER TABLE 语句

说明

此语句用于修改表定义或使表能够参与 Replication Server 复制。

语法 1

ALTER TABLE [*owner*.]*table-name altertable-clause*

altertable-clause:
ADD *column-definition* [*column-constraint* …]
| **ADD** *table-constraint*
| { **ADD PCTFREE** *percent-free-space* | **PCTFREE DEFAULT** }
| **MODIFY** *column-definition*
| **MODIFY** *column-name* **DEFAULT** *default-value*
| **ALTER** *column-name* **SET DEFAULT** *default-value*
| **ALTER** *column-name* **DROP DEFAULT**
| **ALTER** *column-name* **SET COMPUTE (** *expression* **)**
| **ALTER** *column-name* **DROP COMPUTE**
| **MODIFY** *column-name* [**NOT**] **NULL**
| **MODIFY** *column-name* **CHECK NULL**
| **MODIFY** *column-name* **CHECK (** *condition* **)**
| { **DELETE** | **DROP** } *column-name*
| { **DELETE** | **DROP** } **CHECK**
| { **DELETE** | **DROP** } **UNIQUE (** *column-name*, … **)**
| { **DELETE** | **DROP** } **PRIMARY KEY**
| { **DELETE** | **DROP** } **FOREIGN KEY** *role-name*
| **RENAME** *new-table-name*
| **RENAME** *column-name* **TO** *new-column-name*

column-definition :
column-name data-type [**NOT NULL**] [**DEFAULT** *default-value*]

column-constraint :
{ **UNIQUE**
| **PRIMARY KEY**
| **REFERENCES** *table-name*
[**(** *column-name* **)**] [*actions*] [**CLUSTERED**]}
| **CHECK (** *condition* **)**
| **COMPUTE (** *expression* **)**

default-value :
special-value
| *string*
| *global variable*
| [**-**] *number*
| **(** *constant-expression* **)**
| *built-in-function***(** *constant-expression* **)**

| **AUTOINCREMENT**
| **CURRENT DATABASE**
| **CURRENT REMOTE USER**
| **CURRENT UTC TIMESTAMP**
| **GLOBAL AUTOINCREMENT** [**(** *partition-size* **)**]
| **NULL**
| **TIMESTAMP**
| **UTC TIMESTAMP**
| **LAST USER**

special-value:
CURRENT { **DATE** | **TIME** | **TIMESTAMP** | **UTC TIMESTAMP** | **USER** | **PUBLISHER** }
| **USER**

table-constraint :
UNIQUE (*column-name*, … **)**
| **PRIMARY KEY** [**CLUSTERED**] **(** *column-name*, … **)**
| **CHECK (** *condition* **)**
| *foreign-key-constraint*

foreign-key-constraint :
[**NOT NULL**] **FOREIGN KEY** [*role-name*] [**(***column-name*, … **)**]
… **REFERENCES** *table-name* [**(***column-name*, … **)**] [**CLUSTERED**]
… [*actions*] [**CHECK ON COMMIT**]

actions :
[**ON UPDATE** *action*] [**ON DELETE** *action*]

action :
CASCADE
| **SET NULL**
| **SET DEFAULT**
| **RESTRICT**

percent-free-space : integer

语法 2　**ALTER TABLE** [*owner*.]*table-name* **REPLICATE** { **ON** | **OFF** }

参数　**ADD column-definition**　向表中添加新列。要指定 NOT NULL，表必须为空。

如果列有缺省值，新列的所有行将用该缺省值来填充。

NULL 值
Adaptive Server Anywhere 优化可以包含 NULL 的列的创建过程。可以包含 NULL 的第一列分配供八个这样的列使用的空间，然后将这八列全部初始化为 NULL。（这不需要额外的存储空间。）因此，后面添加的七列不需要对表行进行更改。

然后，添加第九列时，为另外八个这样的列分配空间，并修改表的每一行以分配额外的空间。因此，八列中取七列的添加方法运行得非常快。

ADD table-constraint 为表添加约束。有关表约束的完整解释，请参阅 ""CREATE TABLE 语句 " 第 395 页 "。

如果指定 PRIMARY KEY，表一定不能已具有 CREATE TABLE 语句或另一 ALTER TABLE 语句创建的主键。

PCTFREE 指定希望为每个表页保留的可用空间的百分比。如果数据更新时行大小增加，将占用可用空间。如果表页中没有可用空间，则该页上的行大小每次增加时，行都需要在多个表页中拆分，从而导致行碎片并可能引起性能下降。

percent-free-space 值是 0 到 100 之间的整数。前者指定每页上没有剩余的可用空间：每页都完全填满。高值会将每行单独插入到页中。如果未设置 PCTFREE，或者如果指定 DEFAULT，则每页保留 200 字节。

如果设置了 PCTFREE，所有表页的后续插入都使用新值，但是已插入的行不受影响。值一直保持到它被更改或表被删除。

PCTFREE 说明可用于基表、全局临时表或局部临时表。除局部临时表外， PCTFREE 值存储在 SYSATTRIBUTE 系统表中。

有关更多信息，请参阅 ""SYSATTRIBUTE 系统表 " 第 667 页 "。

MODIFY column-definition 更改表中现有列的长度或数据类型。如果指定了 NOT NULL，则 NOT NULL 约束将添加到指定的列。否则，不更改列的 NOT NULL 约束。如果有必要，修改列中的数据将转换为新的数据类型。如果发生转换错误，操作将失败，而表保留不变。

不能修改列以使它成为计算列。只能添加或删除计算列。

删除索引、约束或键
如果列包含在唯一性约束、外键或主键中，则必须删除约束或键后才能修改列。如果删除主键，引用表的所有外键也将被删除。

不能通过 MODIFY 修改表约束或列约束。要更改约束，必须通过 DELETE 删除旧约束并通过 ADD 添加新约束。

MODIFY column-name DEFAULT default-value　更改表中现有列的缺省值。要删除列的缺省值，请指定 DEFAULT NULL。修改缺省值不会更改表中的任何现有值。

ALTER column-name SET DEFAULT default-value　更改表中现有列的缺省值。也可以使用 MODIFY 子句完成此任务，但是 ALTER 与 SQL/92 兼容，而 MODIFY 不兼容。修改缺省值不会更改表中的任何现有值。

ALTER column-name DROP DEFAULT　删除表中现有列的缺省值。也可以使用 MODIFY 子句完成此任务，但是 ALTER 与 SQL/92 兼容，而 MODIFY 不兼容。删除缺省值不会更改表中的任何现有值。

ALTER column-name SET COMPUTE (expression)　更改与计算列关联的表达式。当执行语句时，重新计算列中的值，如果新表达式无效，则语句执行失败。

ALTER column-name DROP COMPUTE　将列从计算列更改为非计算列。此语句不更改表中的任何现有值。

MODIFY column-name [NOT] NULL　更改列的 NOT NULL 约束，以在列中允许或禁止使用 NULL 值。

MODIFY column-name CHECK NULL　删除列的检查约束。此语句不能用于用 5.0 版之前的版本创建的数据库。

MODIFY column-name CHECK (condition)　用指定的 CHECK 条件替换列的现有 CHECK 条件。此语句不能用于用 5.0 版之前的版本创建的数据库。

DELETE column-name　从表中删除列。如果列包含在任何索引、唯一性约束、外键或主键中，则必须删除索引、约束或键后才能删除列。这不会删除引用列的 CHECK 约束。

DELETE CHECK 删除表的所有检查约束。这包括表检查约束和列检查约束。

DELETE UNIQUE (column-name, …) 删除该表的唯一性约束。还删除引用该唯一性约束的任何外键 （而不是主键）。

DELETE PRIMARY KEY 删除该表的主键约束。还删除引用该表的主键的所有外键。

DELETE FOREIGN KEY role-name 删除该表的具有给定角色名称的外键约束。

RENAME new-table-name 将表名更改为 *new-table-name*。注意必须修改任何使用旧表名的应用程序。被自动分配旧表名的外键的名称不更改。

RENAME column-name TO new-column-name 将列名更改为 *new-column-name*。注意需要修改任何使用旧列名的应用程序。

用法

语法 1 ALTER TABLE 语句更改以前创建的表中的表属性 （列定义、约束）。注意语法允许使用变更子句列表，但是在一个 ALTER TABLE 语句中，只能添加、修改或删除一个表约束或列约束。在同一语句中不能同时添加和修改表。

对于局部临时表不能使用 ALTER TABLE。

只要语句影响了当前正由其他连接使用的表，都会禁止 ALTER TABLE。 ALTER TABLE 可能很耗时，当正在处理语句时，服务器不会处理引用表的请求。

有关使用 CLUSTERED 选项的更多信息，请参阅 "ASA SQL 用户指南 > 使用数据库对象 > 使用聚簇索引 "。

5.0 版之前，所有的表约束和列约束都保存在单个表约束中。因此，对于这些数据库，各列约束不能用 MODIFY column-name CHECK NULL 子句删除，也不能用 MODIFY column-name CHECK (condition) 子句替换。要使用这些语句，应该删除整个表约束，并使用 MODIFY column-name CHECK (condition) 子句添加回约束。这时可以使用 MODIFY CHECK。

语法 2 当表有 REPLICATE ON 时，表的所有更改都发送到 Replication Server 进行复制。 Replication Server 中的复制定义用于决定将哪些表更改发送到其他站点。本节的其余部分描述语法 1。

权限

必须是以下之一：

- 表的所有者。
- 具有 DBA 权限的用户。
- 被授予表的 ALTER 权限的用户。
- ALTER TABLE 要求能够排它访问表。

除非已引用临时表的所有用户都断开了连接，否则无法变更全局临时表。

副作用

自动提交。

MODIFY 和 DELETE (DROP) 选项关闭当前连接的所有游标。

ALTER TABLE 操作开始时执行检查点。

一旦变更了列或表，引用变更列的任何存储过程、视图或其他项不再有效。

另请参阅

"CREATE TABLE 语句 " 第 395 页

"DROP 语句 " 第 445 页

"SQL 数据类型 " 第 59 页

"ASA 编程指南 > 在数据库中使用 Java > 使用具有 Java 类的计算列 "

"ASA SQL 用户指南 > 使用数据库对象 > 变更表 "

" 特殊值 " 第 37 页

标准和兼容性

- **SQL/92** 中间级特性。 MODIFY 与 SQL/92 不兼容。
- **SQL/99** ADD COLUMN 是核心特性。其他子句是供应商扩展或特殊的指定 SQL/99 扩展的实现。

- **Sybase** Adaptive Server Enterprise 支持一些子句。

示例

下面的示例在 *employee* 表中添加一个新列，以显示雇员的办公地点。

```
ALTER TABLE employee
ADD office CHAR(20) DEFAULT 'Boston'
```

下面的示例从 *employee* 表中删除办公室列。

```
ALTER TABLE employee
DELETE office
```

客户表中的地址列目前可以保存最多 35 个字符。要使它可以保存最多 50 个字符，输入下面的命令。

```
ALTER TABLE customer
MODIFY address CHAR(50)
```

下面的示例在客户表中添加一列，它为每个客户指定一个销售联系人。

```
ALTER TABLE customer
ADD sales_contact INTEGER
REFERENCES employee (emp_id)
ON UPDATE CASCADE
ON DELETE SET NULL
```

此外键通过级联更新构造并在删除时设置为空。如果雇员更改他们的雇员 ID，列将更新以反映这一更改。如果雇员离开公司并且删除了他们的雇员 ID，则列设置为 NULL。

ALTER TRIGGER 语句

说明

此语句用于用修改的版本替换触发器定义。

在 ALTER TRIGGER 语句中必须包含整个新的触发器定义。

语法 1

ALTER TRIGGER *trigger-name trigger-definition*

trigger-definition :
同 CREATE TRIGGER 语法中触发器名后的部分

语法 2

ALTER TRIGGER *trigger-name* **ON** [*owner.*] *table-name* **SET HIDDEN**

用法

语法 1 除第一个单词外， ALTER TRIGGER 语句的语法与 CREATE TRIGGER 语句的语法相同。有关 *trigger-definition* 的信息，请参阅 ""CREATE TRIGGER 语句 " 第 408 页 和 "CREATE TRIGGER 语句 [T-SQL]" 第 415 页 "。

可以使用 CREATE TRIGGER 语法的 Transact-SQL 形式或 Watcom-SQL 形式。

语法 2 可以使用 SET HIDDEN 使关联的触发器的语法变成乱码，使之不可读。可以卸载该触发器，然后将其重装到其他数据库中。

此设置是不可逆的。
如果您还将再次需要起始源，则必须将它保存在数据库外。

如果使用 SET HIDDEN，则使用存储过程调试程序进行调试将不会显示触发器定义，也无法通过过程分析获得触发器定义。

权限

必须是定义触发器的表的所有者，或者是 DBA，或者有表的 ALTER 权限以及有 RESOURCE 权限。

副作用

自动提交。

另请参阅

"CREATE TRIGGER 语句 " 第 408 页

"CREATE TRIGGER 语句 [T-SQL]" 第 415 页

"DROP 语句 " 第 445 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 隐藏过程、函数、触发器和视图的内容 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

ALTER VIEW 语句

说明

此语句用于用修改的版本替换视图定义。在 ALTER VIEW 语句中必须包含整个新的视图定义。

语法 1

ALTER VIEW
[*owner*.]*view-name* [**(** *column-name*, … **)**] **AS** *select-without-order-by*
[**WITH CHECK OPTION**]

语法 2

ALTER VIEW
[*owner*.]*view-name* **SET HIDDEN**

用法

语法 1　除第一个单词外， ALTER VIEW 语句的语法与 CREATE VIEW 语句的语法相同。 ALTER VIEW 语句用它的内容替换 CREATE VIEW 语句的整个内容。维护视图的现有权限，不必重新分配。如果使用 DROP VIEW 后再执行 CREATE VIEW，而不是 ALTER VIEW，则视图的权限必须重新分配。

语法 2　可以使用 SET HIDDEN 使关联的视图的语法变成乱码，使之不可读。可以卸载该视图，然后将其重装到其他数据库中。

此设置是不可逆的。
如果您还将再次需要起始源，则必须将它保存在数据库外。

如果使用 SET HIDDEN，则使用存储过程调试程序进行调试将不会显示视图定义，也无法通过过程分析获得视图定义。

有关关键字和选项的信息，请参阅 ""CREATE VIEW 语句 " 第 418 页 "。

权限

必须是视图的所有者或者有 DBA 权限。

副作用

自动提交。

从内存中卸载所有的过程和触发器，使引用视图的任何过程或触发器都反映新的视图定义。如果定期变更视图，则过程和触发器的卸载和装载将影响性能。

另请参阅

"CREATE VIEW 语句 " 第 418 页

"DROP 语句 " 第 445 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 隐藏过程、函数、触发器和视图的内容 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

ALTER WRITEFILE 语句

说明

此语句用于更改写文件引用的只读数据库文件的名称。

语法

ALTER WRITEFILE *write-file-name*
REFERENCES *db-file-name* [**KEY** *key*]

write-file-name | *db-file-name* : *string*

用法

ALTER WRITEFILE 语句更改写文件引用的只读数据库文件的名称。如果将数据库文件从一个目录移动到另一个目录，可以使用此语句将写文件指向新位置。

数据库文件的路径名相对于启动时数据库服务器的当前目录。

有关字符串中转义反斜线字符的信息，请参阅 "" 字符串 " 第 10 页 "。

权限

执行此语句所需的权限在服务器命令行上用 -gu 选项来设置。缺省设置为要求具有 DBA 权限。

如果要更改高度加密数据库的写文件，需要指定 KEY 值。

在 Windows CE 上不受支持。

副作用

自动提交。

另请参阅

"CREATE WRITEFILE 语句 " 第 420 页

"ASA 数据库管理指南 > 数据库管理实用程序 > Write File 实用程序 "

"ASA 数据库管理指南 > 使用数据库文件 > 使用写文件 "

"ASA 数据库管理指南 > 使用数据库文件 > 使用 utility 数据库 "

"ASA 数据库管理指南 > 连接参数和通信参数 > Encryption Key 连接参数 [DBKEY]"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的语句更改现有写文件 `c:\readwrite.wrt` 以指向数据库文件 `h:\readonly.db`。

```
ALTER WRITEFILE 'c:\\readwrite.wrt'
REFERENCES 'h:\\readonly.db'
```

BACKUP 语句

说明　此语句用于备份数据库和事务日志。

语法 1（映像备份）

BACKUP DATABASE
DIRECTORY *backup-directory*
[**WAIT BEFORE START**]
[**WAIT AFTER END**]
[**DBFILE ONLY**]
[**TRANSACTION LOG ONLY**]
[**TRANSACTION LOG RENAME** [**MATCH**]]
[**TRANSACTION LOG TRUNCATE**]

backup-directory :　string

语法 2（档案备份）

BACKUP DATABASE TO *archive-root*
[**ATTENDED** { **ON** | **OFF** }]
[**WITH COMMENT** *comment string*]

archive-root :　string

comment-string :　string

参数

backup-directory　那些文件在磁盘上的目标位置，相对于启动时服务器的当前目录。如果目录尚不存在，则创建该目录。如果指定空字符串作为目录，则不必先复制日志就可以重命名或截断它。

WAIT BEFORE START 子句　该子句确保数据库的备份副本不包含任何恢复所需的信息。具体说来，它确保每个连接的回退日志为空。

如果使用该子句得到了一个备份，那么可以以只读模式启动该备份数据库并校验它。通过启用备份数据库校验，客户可以避免额外复制数据库。

WAIT AFTER END 子句　如果要重命名或截断事务日志，可以使用该子句。此语句确保在重命名或截断日志前完成所有事务。如果使用该子句，则备份必须等待其他连接提交或者在结束前回退任何打开的事务。

MATCH 关键字　如果提供 MATCH 关键字，则事务日志的备份副本被赋予 `YYMMDDnn.log` 形式的名称。这使同一个语句可以执行多次，而不会覆盖旧数据。

archive-root　档案文件的文件名或磁带驱动器设备名。

要备份到磁带，必须指定磁带驱动器的设备名。例如，在 Windows NT 或 NetWare 上，第一个磁带驱动器是 `\\.\tape0`。

反斜线 (\ ) 是 SQL 字符串中的转义字符，因此每个反斜线必须是两道。有关转义字符和字符串的更多信息，请参阅 "" 字符串 " 第 10 页 "。

WITH COMMENT 在档案文件和备份历史文件中记录注释。

ATTENDED 该子句仅适用于备份到磁带设备时。 ATTENDED ON （缺省值）表示有人可以监视磁带驱动器的状态，必要时在驱动器中放入新磁带。如果磁带驱动器要求干预，一条消息将发送到发出 BACKUP 语句的应用程序。然后数据库服务器等待驱动器就绪。这在某些情况下会发生，比如在需要新磁带时。

如果指定 ATTENDED OFF 并需要新磁带或者驱动器没有就绪，则不发送消息并且给出错误。

每个 BACKUP 操作 （不论是映像还是档案）都更新名为 `backup.syb` 的历史文件。该文件存储在数据库服务器可执行文件所在的同一目录中。

用法

第一个语法是映像备份，第二个语法是档案备份。

语法 1 映像备份创建每个数据库文件的副本，采用的方式与 Backup 实用程序 (`dbbackup`) 相同。但是， BACKUP 语句在服务器上备份，而 Backup 实用程序从客户机备份。

（可选）只能保存数据库文件或事务日志。备份完成后也可以重命名或截断日志。

或者，可以指定空字符串作为目录，这样不必事先复制日志就可以重命名或截断它。这在需要考虑空间大小的复制环境中尤其有用。此功能与事务日志大小的事件处理程序一起使用时，可在日志达到给定大小时重命名它，与 DELETE_OLD_LOGS 选项一起使用时可删除不再需要的日志。

要从映像备份恢复，请将保存的文件复制回原来的位置并重新应用事务日志，详见 "ASA 数据库管理指南 > 备份和数据恢复 " 一章中的介绍。

语法 2 档案备份创建单个文件来保存所有必需的备份信息。目标可以是文件名或磁带驱动器设备名。比 NetWare 5 早的 NetWare 版本不支持到磁带的档案备份。

一个给定的磁带上只能有一个备份。文件 backup.syb 记录在给定服务器上执行的 BACKUP 和 RESTORE 操作。

磁带在备份结束时弹出。

每个磁带上只能有一个档案，但是一个档案可以跨转多个磁带。要从档案备份恢复数据库，使用 RESTORE DATABASE 语句。

权限

必须拥有 DBA 权限。

副作用

导致检查点。

另请参阅

"RESTORE DATABASE 语句 " 第 571 页

"ASA 数据库管理指南 > 备份和数据恢复 "

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 与 Adaptive Server Enterprise 不兼容。
- **Windows CE** 在 Windows CE 平台上仅支持 BACKUP DATABASE DIRECTORY 语法 （上面的语法 1）。

示例

将当前数据库和事务日志备份到文件，重命名现有事务日志。创建映像备份。

```
BACKUP DATABASE
DIRECTORY 'd:\\temp\\backup'
TRANSACTION LOG RENAME
```

重命名事务日志的选项在仍需要旧事务日志的复制环境中特别有用。

将当前数据库和事务日志备份到磁带：

```
BACKUP DATABASE
TO '\\\\.\\tape0'
```

不进行复制即重命名日志：

```
BACKUP DATABASE DIRECTORY''
TRANSACTION LOG ONLY
TRANSACTION LOG RENAME
```

BEGIN 语句

说明

此语句用于将 SQL 语句组合在一起。

语法

[*statement-label* :]
BEGIN [[**NOT**] **ATOMIC**]
[*local-declaration*; …]
statement-list
[**EXCEPTION** [*exception-case* …]]
END [*statement-label*]

local-declaration :
variable-declaration
| *cursor-declaration*
| *exception-declaration*
| *temporary-table-declaration*

variable-declaration :
DECLARE *variable-name data-type*

exception-declaration :
DECLARE *exception-name* **EXCEPTION**
FOR SQLSTATE [**VALUE**] *string*

exception-case :
WHEN *exception-name* [, …] **THEN** *statement-list*
| **WHEN OTHERS THEN** *statement-list*

参数

local-declaration 紧接在 BEGIN 后，复合语句可以有仅存在于复合语句中的对象的本地声明。复合语句可以有变量、游标、临时表或异常的本地声明。本地声明可由该复合语句中或嵌套的任何复合语句中的任何语句引用。本地声明对于从复合语句中调用的其他过程不可见。

statement-label 如果指定了结尾 *statement-label*，则必须与起始 *statement-label* 匹配。 LEAVE 语句可用于在复合语句后的第一个语句处继续执行。复合语句是过程或触发器的主体，它具有与过程或触发器同名的隐式标签。

有关复合语句和异常处理的完整说明，请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 "。

用法

过程或触发器的主体是复合语句。过程或触发器内的控制语句也可以使用复合语句。

复合语句允许一个或多个 SQL 语句组合在一起，按一个单元处理。复合语句以关键字 BEGIN 开始，以关键字 END 结束。

权限

无。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"DECLARE LOCAL TEMPORARY TABLE 语句 " 第 433 页

"LEAVE 语句 " 第 527 页

"SIGNAL 语句 " 第 612 页

"RESIGNAL 语句 " 第 570 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

标准和兼容性

- **SQL/92**　持久存储模块特性。
- **SQL/99**　持久存储模块特性。
- **Sybase**　受 Adaptive Server Enterprise 支持。这并不意味着支持复合语句中的所有语句。

  在 Transact-SQL 中不要求 BEGIN 和 END 关键字。

  在 Transact-SQL 中， BEGIN 和 END 用于将一组语句组合成一个复合语句，以便控制语句（如 IF … ELSE，它只影响单个 SQL 语句的执行）可以影响整个组的执行。 Adaptive Server Enterprise 不支持 ATOMIC 关键字。

  在 Transact-SQL 中， DECLARE 语句不必紧跟在 BEGIN 关键字后，并且声明的游标或变量在复合语句的持续时间内存在。为了兼容，应该在复合语句的开头声明变量。

示例

过程或触发器的主体是复合语句。

```
CREATE PROCEDURE TopCustomer (OUT TopCompany CHAR(35),
OUT TopValue INT)
BEGIN
   DECLARE err_notfound EXCEPTION FOR
      SQLSTATE '02000';
   DECLARE curThisCust CURSOR FOR
      SELECT company_name, CAST(
            sum(sales_order_items.quantity *
            product.unit_price) AS INTEGER) VALUE
      FROM customer
            LEFT OUTER JOIN sales_order
            LEFT OUTER JOIN sales_order_items
            LEFT OUTER JOIN product
      GROUP BY company_name;
   DECLARE ThisValue INT;
   DECLARE ThisCompany CHAR(35);
   SET TopValue = 0;
   OPEN curThisCust;
   CustomerLoop:
   LOOP
      FETCH NEXT curThisCust
         INTO ThisCompany, ThisValue;
      IF SQLSTATE = err_notfound THEN
         LEAVE CustomerLoop;
      END IF;
      IF ThisValue > TopValue THEN
         SET TopValue = ThisValue;
         SET TopCompany = ThisCompany;
      END IF;
   END LOOP CustomerLoop;
   CLOSE curThisCust;
END
```

BEGIN TRANSACTION 语句

说明

此语句用于开始用户定义的事务。

语法

BEGIN TRAN[**SACTION**] [*transaction-name*]

用法

可选参数 *transaction-name* 是分配给该事务的名称。它必须是有效的标识符。仅在最外层的嵌套 BEGIN/COMMIT 语句对或 BEGIN/ROLLBACK 语句对中使用事务名称。

在事务内部执行时， BEGIN TRANSACTION 语句将事务嵌套级别增加一级。通过 COMMIT 语句可减少嵌套级别。当事务被嵌套时，只有最外层的 COMMIT 使数据库更改永久生效。

Adaptive Server Enterprise 和 Adaptive Server Anywhere 都有两种事务模式。

缺省的 Adaptive Server Enterprise 事务模式称为非链接模式，它单独提交每个语句，除非执行显式 BEGIN TRANSACTION 语句启动事务。相反，与 ISO SQL/92 兼容的链接模式仅当执行显式 COMMIT 时或执行自动提交的语句 （如数据定义语句）时才提交事务。

通过设置 CHAINED 数据库选项可以控制模式。 Adaptive Server Anywhere 中的 ODBC 和嵌入式 SQL 连接的缺省设置是 ON，这种情况下， Adaptive Server Anywhere 在链接模式下运行。（ODBC 用户还应该检查 AutoCommit ODBC 设置）。 TDS 连接的缺省值是 OFF。

在非链接模式下，事务在任何数据检索或修改语句前隐式启动。这些语句包括：DELETE、 INSERT、 OPEN、 FETCH、 SELECT 和 UPDATE。仍必须用 COMMIT 或 ROLLBACK 语句显式结束事务。

不能在事务中变更 CHAINED 选项。

注意
当调用存储过程时，应该确保它在要求的事务模式下正确运行。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > CHAINED 选项 [compatibility]"。

当前嵌套级别保存在全局变量 **@@trancount** 中。执行第一个 BEGIN TRANSACTION 语句前， **@@trancount** 变量的值为零，并且只有当 **@@trancount** 等于 1 时执行的 COMMIT 才永久地更改数据库。

没有事务名称或保存点名称的 ROLLBACK 语句总是将语句回退到最外层的 BEGIN TRANSACTION （显式或隐式）语句，并且取消整个事务。

权限

无。

副作用

无。

另请参阅

"COMMIT 语句 " 第 305 页

"ASA 数据库管理指南 > 数据库选项 > ISOLATION_LEVEL 选项 [compatibility]"

"ROLLBACK 语句 " 第 583 页

"SAVEPOINT 语句 " 第 586 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

示例

下面的批处理语句报告 *@@trancount* 的连续值：0、 1、 2、 1 和 0。这些值显示在服务器窗口上。

```
PRINT @@trancount
BEGIN TRANSACTION
PRINT @@trancount
BEGIN TRANSACTION
PRINT @@trancount
COMMIT
PRINT @@trancount
COMMIT
PRINT @@trancount
```

应该只是依靠 **@@trancount** 的值来跟踪已发出的显式 BEGIN TRANSACTION 语句的数量。

当 Adaptive Server Enterprise 隐式启动事务时， @@trancount 变量设置为 1。当隐式启动事务时， Adaptive Server Anywhere 不将 @@trancount 变量值设置为 1。因此，在任何 BEGIN TRANSACTION 语句前 （甚至有当前事务）， Adaptive Server Anywhere **@@trancount** 变量的值为零，而在 Adaptive Server Enterprise （在 **chained** 模式下）中，它的值为 1。

对于以 BEGIN TRANSACTION 语句开始的事务，在第一个 BEGIN TRANSACTION 语句后， Adaptive Server Anywhere 和 Adaptive Server Enterprise 中的 **@@trancount** 值均为 1。如果用不同的语句隐式启动事务，然后执行 BEGIN TRANSACTION 语句，则在执行 BEGIN TRANSACTION 语句后， Adaptive Server Anywhere 中的 **@@trancount** 值为 1，在 Adaptive Server Enterprise 中的值为 2。

CALL 语句

说明

此语句用于调用过程。

语法 1

[*variable* =] **CALL** *procedure-name* **(** [*expression*, …] **)**

语法 2

[*variable* =] **CALL** *procedure-name* **(** [*parameter-name* = *expression*, …] **)**

用法

CALL 语句调用以前用 CREATE PROCEDURE 语句创建的过程。当过程完成后，将复制回任何 INOUT 或 OUT 参数值。

通过位置或者通过使用关键字格式，可以指定参数列表。通过位置，参数与过程的参数列表中相对应的参数匹配。通过关键字，参数与指定的参数匹配。

在 CREATE PROCEDURE 语句中可以为过程参数分配缺省值，缺少的参数被分配缺省值。如果没有设置缺省值，也没有提供参数，则会给出错误。

在过程内部，当过程返回结果集时， CALL 语句可以用在 DECLARE 语句中 （请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 从过程返回结果 "）。

使用 RETURN 语句，过程可以返回整数值 （比方说作为状态指示符）。使用等号作为赋值运算符，可以将该返回值保存在变量中：

```
CREATE VARIABLE returnval INT;
returnval = CALL proc_integer ( arg1 = val1, ... )
```

有关返回非整数值的信息，请参阅 ""CREATE FUNCTION 语句 " 第 337 页 "。

权限

必须是过程的所有者，有过程的 EXECUTE 权限或者有 DBA 权限。

副作用

无。

另请参阅

"CREATE PROCEDURE 语句 " 第 347 页

"GRANT 语句 " 第 497 页

"EXECUTE 语句 [T-SQL]" 第 469 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。有关支持的替代选项，请参阅 ""EXECUTE 语句 [T-SQL]" 第 469 页 "。

示例

调用 *sp_customer_list* 过程。该过程没有参数，并返回结果集。

```
CALL sp_customer_list()
```

下面的 Interactive SQL 示例创建一个返回客户 （提供了该客户的 ID）所下的订单数的过程，创建一个保存结果的变量，调用此过程并显示结果。

```
CREATE PROCEDURE OrderCount (IN customer_ID INT, OUT
Orders INT)
BEGIN
   SELECT COUNT("DBA".sales_order.id)
   INTO Orders
   FROM "DBA".customer
   KEY LEFT OUTER JOIN "DBA".sales_order
   WHERE "DBA".customer.id = customer_ID;
END
go
 -- Create a variable to hold the result
CREATE VARIABLE Orders INT
go
-- Call the procedure, FOR customer 101
CALL OrderCount ( 101, Orders)
go
--  Display the result
SELECT Orders FROM DUMMY
go
```

CASE 语句

说明

此语句用于根据多种情况选择执行路径。

语法 1

CASE *value-expression*
WHEN [*constant* | **NULL**] **THEN** *statement-list* …
[**WHEN** [*constant* | **NULL**] **THEN** *statement-list*] …
[**ELSE** *statement-list*]
END CASE

语法 2

CASE
WHEN [*search-condition* | **NULL**] **THEN** *statement-list* …
[**WHEN** [*search-condition* | **NULL**] **THEN** *statement-list*] …
[**ELSE** *statement-list*]
END CASE

用法

语法 1 CASE 语句是控制语句，它允许您根据表达式的值选择要执行的 SQL 语句列表。 *value-expression* 是具有单值的表达式，它的值可以是字符串、数字、日期或其他 SQL 数据类型。如果 *value-expression* 的值存在匹配的 WHEN 子句，则执行 WHEN 子句中的 *statement-list*。如果没有合适的 WHEN 子句而有 ELSE 子句，则执行 ELSE 子句中的 *statement-list*。执行在 END CASE 后的第一个语句重新开始。

如果 *value-expression* 可以为空，使用 ISNULL 函数用不同的表达式替换 NULL *value-expression* 。

有关 ISNULL 函数的更多信息，请参阅 ""ISNULL 函数 [数据类型转换]" 第 170 页 "。

语法 2 在这种格式下，执行 CASE 语句中第一个满足 *search-condition* 的语句。如果不满足 *search-conditions*，则执行 ELSE 子句。

如果表达式可以为 NULL，对第一个 *search-condition* 使用以下语法：

```
WHEN search-condition IS NULL THEN statement-list
```

有关 NULL 值的更多信息，请参阅 "ASA SQL 用户指南 > 查询：从表中选择数据 > 未知值：NULL"。

CASE 语句不同于 CASE 表达式
不要混淆 CASE 语句与 CASE 表达式的语法。

有关 CASE 表达式的信息，请参阅 ""CASE 表达式 " 第 20 页 "。

权限

无。

副作用

无。

另请参阅

"BEGIN 语句 " 第 288 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

标准和兼容性

- **SQL/92** 持久存储模块特性。 Adaptive Server Anywhere 支持允许使用 WHEN NULL 的 CASE 语句。这是 SQL/92 标准的供应商扩展。
- **SQL/99** 持久存储模块特性。 Adaptive Server Anywhere 支持允许使用 WHEN NULL 的 CASE 语句。这是 SQL/92 标准的供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的过程使用 CASE 语句，将示例数据库的产品表中列出的产品分为衬衣、帽子、短裤、未知几大类。

```
CREATE PROCEDURE ProductType (IN product_id INT, OUT
type CHAR(10))
BEGIN
   DECLARE prod_name CHAR(20);
   SELECT name INTO prod_name FROM "DBA"."product"
   WHERE id = product_id;
   CASE prod_name
   WHEN 'Tee Shirt' THEN
      SET type = 'Shirt'
   WHEN 'Sweatshirt' THEN
      SET type = 'Shirt'
   WHEN 'Baseball Cap' THEN
      SET type = 'Hat'
   WHEN 'Visor' THEN
      SET type = 'Hat'
   WHEN 'Shorts' THEN
```

```
      SET type = 'Shorts'
   ELSE
      SET type = 'UNKNOWN'
   END CASE;
END
```

下面的示例使用语法 2 生成关于示例数据库中产品数量的消息。

```
CREATE PROCEDURE StockLevel (IN product_id INT)
BEGIN
   DECLARE qty INT;
   SELECT quantity INTO qty FROM product
   WHERE id = product_id;
   CASE
   WHEN qty < 30 THEN
      MESSAGE 'Order Stock' TO CLIENT;
   WHEN qty > 100 THEN
      MESSAGE 'Overstocked' TO CLIENT;
   ELSE
      MESSAGE 'Sufficient stock on hand' TO CLIENT;
   END CASE;
END
```

CHECKPOINT 语句

说明

此语句用于对数据库执行检查点操作。

语法

CHECKPOINT

用法

CHECKPOINT 语句强制数据库服务器执行检查点操作。数据库服务器也会根据内部算法自动执行检查点操作。应用程序通常不需要发出 CHECKPOINT 语句。

有关检查点的完整说明，请参阅 "ASA 数据库管理指南 > 备份和数据恢复 "。

权限

对网络数据库执行检查点操作需要有 DBA 权限。

对个人数据库服务器执行检查点操作不需要任何权限。

副作用

无。

另请参阅

"ASA 数据库管理指南 > 数据库选项 > CHECKPOINT_TIME 选项 [database]"

"ASA 数据库管理指南 > 数据库选项 > RECOVERY_TIME 选项 [database]"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

CLEAR 语句 [Interactive SQL]

说明

此语句用于清除 Interactive SQL 窗格。

语法

CLEAR

用法

CLEAR 语句用于清除 "SQL 语句 " 窗格、 " 消息 " 窗格以及 " 结果 " 窗格中的 " 结果 "、 " 消息 "、 " 计划 " 和 "UltraLite 计划 " 选项卡。

权限

无。

副作用

关闭与清除的数据关联的游标。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用

CLOSE 语句 [ESQL] [SP]

说明　此语句用于关闭游标。

语法　**CLOSE** *cursor-name*

cursor-name : identifier | hostvar

用法　此语句关闭指定的游标。

权限　游标在此前必须已打开。

副作用　无。

另请参阅　"OPEN 语句 [ESQL] [SP]" 第 543 页

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"PREPARE 语句 [ESQL]" 第 553 页

标准和兼容性

- **SQL/92**　初级特性。
- **SQL/99**　核心特性。
- **Sybase**　受 Adaptive Server Enterprise 支持。

示例　下面的示例关闭嵌入式 SQL 中的游标。

```
EXEC SQL CLOSE employee_cursor;
EXEC SQL CLOSE :cursor_var;
```

下面的过程使用游标。

```
CREATE PROCEDURE TopCustomer (OUT TopCompany CHAR(35),
OUT TopValue INT)
BEGIN
   DECLARE err_notfound EXCEPTION
      FOR SQLSTATE '02000';
   DECLARE curThisCust CURSOR FOR
   SELECT company_name, CAST(
sum(sales_order_items.quantity *
   product.unit_price) AS INTEGER) VALUE
   FROM customer
```

```
   LEFT OUTER JOIN sales_order
   LEFT OUTER JOIN sales_order_items
   LEFT OUTER JOIN product
   GROUP BY company_name;
   DECLARE ThisValue INT;
   DECLARE ThisCompany CHAR(35);
   SET TopValue = 0;
   OPEN curThisCust;
   CustomerLoop:
   LOOP
      FETCH NEXT curThisCust
      INTO ThisCompany, ThisValue;
         IF SQLSTATE = err_notfound THEN
            LEAVE CustomerLoop;
         END IF;
         IF ThisValue > TopValue THEN
            SET TopValue = ThisValue;
            SET TopCompany = ThisCompany;
         END IF;
      END LOOP CustomerLoop;
   CLOSE curThisCust;
END
```

COMMENT 语句

说明

此语句用于在系统表中存储关于数据库对象的注释。

语法

COMMENT ON
{
COLUMN [*owner*.]*table-name*.*column-name*
| **EVENT** *event-name*
| **FOREIGN KEY** [*owner*.]*table-name*.*role-name*
| **INDEX** [[*owner*.] *table*.]*index-name*
| **JAVA CLASS** *java-class-name*
| **JAVA JAR** *java-jar-name*
| **LOGIN** *integrated_login_id*
| **PROCEDURE** [*owner*.]*procedure-name*
| **TABLE** [*owner*.]*table-name*
| **TRIGGER** [[*owner*.]*tablename*.]*trigger-name*
| **USER** *userid*
| **VIEW** [*owner*.]*view-name*
}
IS *comment*

comment : string | **NULL**

用法

几个系统表中有一个名为 " 注释 " 的列，在该列中可以将注释与数据库项关联 （SYSUSERPERM、 SYSTABLE、 SYSCOLUMN、 SYSINDEX、 SYSLOGIN、 SYSFOREIGNKEY、 SYSPROCEDURE、 SYSTRIGGER）。COMMENT ON 语句允许在这些系统表中设置 " 注释 " 列。通过将注释设置成 NULL 可以删除注释。

对于索引或触发器的注释，其所有者是定义该索引或触发器的表的所有者。

权限

必须是要注释的数据库对象的所有者，或者必须有 DBA 权限。

副作用

自动提交。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

示例

下面的示例显示如何添加和删除注释。

向雇员表添加注释。

```
COMMENT
ON TABLE employee
IS 'Employee information'
```

从雇员表中删除注释。

```
COMMENT
ON TABLE employee
IS NULL
```

COMMIT 语句

说明

此语句用于使数据库更改永久生效，或用于终止用户定义的事务。

语法 1

COMMIT [**WORK**]

语法 2

COMMIT TRAN[**SACTION**] [*transaction-name*]

参数

transaction-name　指派给该事务的可选名称。它必须是有效的标识符。只应在最外层的嵌套 BEGIN/COMMIT 语句对或 BEGIN/ROLLBACK 语句对中使用事务名称。

有关 Adaptive Server Enterprise 和 Adaptive Server Anywhere 中的事务嵌套的更多信息，请参阅 ""BEGIN TRANSACTION 语句 " 第 291 页 "。有关保存点的更多信息，请参阅 ""SAVEPOINT 语句 " 第 586 页 "。

可以使用一组选项来控制 COMMIT 语句的行为细节。有关信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > COOPERATIVE_COMMIT_TIMEOUT 选项 [database]"、 "ASA 数据库管理指南 > 数据库选项 > COOPERATIVE_COMMITS 选项 [database]"、"ASA 数据库管理指南 > 数据库选项 > DELAYED_COMMITS 选项 [database]" 和 "ASA 数据库管理指南 > 数据库选项 > DELAYED_COMMIT_TIMEOUT 选项 [database]"。可以使用 Commit 连接属性返回当前连接的 Commit 数。

用法

语法 1　COMMIT 语句结束事务，并使在该事务期间所做的所有更改在数据库中永久生效。

数据定义语句全都自动执行提交。有关信息，请参阅每个 SQL 语句的 " 副作用 " 清单。

如果数据库服务器检测到任何无效的外键， COMMIT 语句将失败。这产生的结果是：不可能通过任何无效外键结束事务。通常，在每个数据处理操作中会检查外键的完整性。但是，如果数据库选项 WAIT_FOR_COMMIT 设置为 ON，或者使用 CHECK ON COMMIT 子句定义了具体的外键，则数据库服务器会将完整性检查推迟到执行 COMMIT 语句后。

语法 2 可以使用 BEGIN TRANSACTION 和 COMMIT TRANSACTION 语句对来构造事务嵌套。嵌套事务类似于保存点。当在一组嵌套事务的最外层执行时，此语句使数据库更改永久生效。当在事务内部执行时，COMMIT TRANSACTION 语句使事务嵌套级别减少一级。当事务被嵌套时，只有最外层的 COMMIT 使数据库更改永久生效。

权限

无。

副作用

关闭打开为 WITH HOLD 的游标之外的所有其他游标。

删除此连接上临时声明的表的所有行，除非这些表是用 ON COMMIT PRESERVE ROWS 声明的。

另请参阅

"BEGIN TRANSACTION 语句 " 第 291 页

"PREPARE TO COMMIT 语句 " 第 556 页

"ROLLBACK 语句 " 第 583 页

标准和兼容性

- **SQL/92** 语法 1 是初级特性。语法 2 是 Transact-SQL 扩展。
- **SQL/99** 语法 1 是核心特性。语法 2 是 Transact-SQL 扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

示例

下面的语句提交当前事务：

```
COMMIT
```

下面的 Transact-SQL 批处理语句报告 *@@trancount* 的连续值：0、 1、 2、 1、 0。

```
PRINT @@trancount
BEGIN TRANSACTION
PRINT @@trancount
BEGIN TRANSACTION
PRINT @@trancount
COMMIT TRANSACTION
PRINT @@trancount
COMMIT TRANSACTION
PRINT @@trancount
go
```

CONFIGURE 语句 [Interactive SQL]

说明

此语句用于打开 "Interactive SQL 选项 " 对话框。

语法

CONFIGURE

用法

CONFIGURE 语句激活 "Interactive SQL 选项 " 对话框。此窗口显示所有 Interactive SQL 选项的当前设置。它不显示数据库选项，也不允许对它们进行修改。

在此对话框中可以配置 Interactive SQL 设置。如果选择了 " 设为永久 "，这些选项将写入数据库的 SYSOPTION 表，而且数据库服务器会执行自动的 COMMIT。如果未选择 " 设为永久 "，而是单击了 " 确定 "，则这些选项只是暂时设置的，它们仅对当前的数据库连接有效。

权限

无。

副作用

无。

另请参阅

"SET OPTION 语句 " 第 601 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

CONNECT 语句 [ESQL] [Interactive SQL]

说明

此语句用于建立到数据库的连接。

语法 1

CONNECT
[**TO** *engine-name*]
[**DATABASE** *database-name*]
[**AS** *connection-name*]
[**USER**] *userid* **IDENTIFIED BY** *password*

engine-name, *database-name*, *connection-name*, *userid*, *password* :
{ *identifier* | *string* | *hostvar* }

语法 2

CONNECT USING *connect-string*

connect-string : { *identifier* | *string* | *hostvar* }

参数

AS 子句 通过指定 AS 子句可对连接进行命名（可选）。这允许建立到同一个数据库的多个连接，或者到同一个或不同的多个数据库服务器的多个连接，所有连接都是同时发生的。每个连接都有自己的关联事务。事务之间有时甚至会发生锁定冲突。例如，当试图从两个不同的连接修改同一数据库中的同一记录时，就会出现这种情况。

语法 2 *connect-string* 是格式为 **keyword**=*value* 的参数设置的列表，必须用单引号将它括起来。

有关连接字符串的更多信息，请参阅 "ASA 数据库管理指南 > 连接到数据库 > 连接参数 "。

用法

CONNECT 语句建立数据库连接，该数据库在由 *engine-name* 标识的服务器上运行的 *database-name* 来标识。

嵌入式 SQL 行为 在嵌入式 SQL 中，如果未指定 *engine-name*，将采用缺省的本地数据库服务器（启动的第一个数据库服务器）。如果未指定 *database-name*，则采用给定服务器上的第一个数据库。

WHENEVER 语句、 SET SQLCA 和一些 DECLARE 语句不生成代码，因此它们在源文件中可以出现在 CONNECT 语句之前。在其它情况下，在执行一条成功的 CONNECT 语句之前不能执行任何语句。

用户 ID 和口令用于对所有动态 SQL 语句执行权限检查。

可以在不显式指定口令的情况下连接，方法是对口令使用主机变量，并将该主机变量的值设置成空指针。

如果以具有 DBA 权限的用户 ID 连接，则不用指定口令就可以连接到其它用户 ID。（dbtran 的输出要求此功能。）

有关连接算法的详细说明，请参阅 "ASA 数据库管理指南 > 连接到数据库 > 对连接进行疑难解答 "。

Interactive SQL 行为 如果在 CONNECT 语句中未指定任何数据库或服务器， Interactive SQL 将保持与当前数据库 （而不是与缺省的服务器和数据库）的连接。如果指定了数据库名称但未指定服务器名称， Interactive SQL 会尝试连接到当前服务器上的指定数据库。如果指定了服务器名称但未指定数据库名称， Interactive SQL 会连接到指定服务器上的缺省数据库。

例如，如果与数据库连接时执行以下批处理语句，则将在同一个数据库中创建两个表。

```
CREATE TABLE t1( c1 int )
go
CONNECT DBA IDENTIFIED BY SQL
go
CREATE TABLE t2 (c1 int )
go
```

在执行一条成功的 CONNECT 语句之前，不能执行任何其它数据库语句。

在用户界面中，如果未指定口令或用户 ID 和口令，系统将提示用户键入缺少的信息。

在以命令提示符模式或批处理模式运行的 Interactive SQL 中，如果不带 AS 子句执行 CONNECT，将打开一个未命名的连接。如果已有其它未命名的连接打开，则以前的那个连接将自动关闭。否则，在运行 CONNECT 时不关闭现有的连接。

对多个连接的管理是通过当前连接的概念来实现的。在成功执行连接语句后，新连接变成当前连接。要切换到不同的连接，请使用 SET CONNECTION 语句。 DISCONNECT 语句用于删除连接。

在 Interactive SQL 中，连接信息 （包括数据库名称、用户 ID 和数据库服务器）出现在 "SQL 语句 " 窗格上方的标题栏中。如果未连接到数据库，标题栏中将显示 " 未连接 "。

权限

无。

副作用

无。

另请参阅

"GRANT 语句 " 第 497 页

"DISCONNECT 语句 [ESQL] [Interactive SQL]" 第 444 页

"SET CONNECTION 语句 [Interactive SQL] [ESQL]" 第 598 页

"SETUSER 语句 " 第 610 页

"ASA 数据库管理指南 > 连接参数和通信参数 > 连接参数 "

标准和兼容性

- **SQL/92** 语法 1 是完全的 SQL 特性。语法 2 是供应商扩展。
- **SQL/99** 语法 1 是核心 SQL 外部的 SQL/ 基础特性。语法 2 是供应商扩展。
- **Sybase** Open Client 嵌入式 SQL 支持另一种 CONNECT 语句语法。

示例

以下是 CONNECT 在嵌入式 SQL 中的用法示例。

```
EXEC SQL CONNECT AS :conn_name
USER :userid IDENTIFIED BY :password;
EXEC SQL CONNECT USER "DBA" IDENTIFIED BY "SQL";
```

从 Interactive SQL 连接到数据库。 Interactive SQL 提示您输入用户 ID 和口令。

```
CONNECT
```

以 DBA 身份从 Interactive SQL 连接到缺省数据库。 Interactive SQL 提示您输入口令。

```
CONNECT USER "DBA"
```

以 DBA 身份从 Interactive SQL 连接到示例数据库。

```
CONNECT
TO asademo
USER DBA
IDENTIFIED BY SQL
```

使用连接字符串从 Interactive SQL 连接到示例数据库。

```
CONNECT
USING 'UID=DBA;PWD=SQL;DBN=asademo'
```

连接到示例数据库后，数据库名称、用户 ID 以及服务器名称就出现在标题栏中：*asademo (DBA) on asademo*。

CREATE COMPRESSED DATABASE 语句

说明

此语句用于用现有的数据库文件创建压缩数据库，或用于展开压缩数据库。

语法

CREATE [**COMPRESSED** | **EXPANDED**] **DATABASE** *new-db-file-name*
FROM *old-db-file-name* [**KEY** *key*]

用法

用未压缩数据库文件创建压缩数据库文件，或者将压缩数据库文件恢复成解压缩数据库文件。

任何相对路径都被解析成相对于服务器的当前工作路径。

不能对主数据库文件以外的任何其它文件使用此语句。

权限

- 执行此语句所需的权限在服务器命令行上用 -gu 选项来设置。缺省设置为要求具有 DBA 权限。
- 运行服务器时所使用的操作系统帐户必须对创建文件的目录具有写权限。
- 旧的数据库文件当前不能正在运行。
- 在 Windows CE 上不受支持。
- 如果要创建高度加密的数据库的压缩数据库，必须指定密钥。

副作用

创建操作系统文件。

另请参阅

"ASA 数据库管理指南 > 数据库管理实用程序 > Compression 实用程序 "

"ASA 数据库管理指南 > 数据库管理实用程序 > Uncompression 实用程序 "

"ASA 数据库管理指南 > 连接参数和通信参数 > Encryption Key 连接参数 [DBKEY]"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的语句在 C:\ 目录中创建一个名为 compress.db 的压缩数据库文件，它的源文件是服务器当前工作目录中名为 full.db 的数据库文件。

```
CREATE COMPRESSED DATABASE 'C:\\compress.db'
FROM 'full.db'
```

下面的语句在 C:\ 目录中创建一个名为 full.db 的解压缩数据库文件，它的源文件是服务器当前工作目录中名为 compress.db 的压缩数据库文件。

```
CREATE EXPANDED DATABASE 'C:\\full.db'
FROM 'compress.db'
```

CREATE DATABASE 语句

说明

此语句用于创建数据库。数据库存储为操作系统文件。

语法

CREATE DATABASE *db-file-name*
[[**TRANSACTION**] { **LOG OFF** | **LOG ON** } [*log-file-name-string*]
[**MIRROR** *mirror-file-name-string*]]
[**CASE** { **RESPECT** | **IGNORE** }]
[**PAGE SIZE** *page-size*]
[**COLLATION** *collation-label*]
[**ENCRYPTED** { **ON** | **OFF** | *key-spec* }]
[**BLANK PADDING** { **ON** | **OFF** }]
[**ASE** [**COMPATIBLE**]]
[**JAVA** { **ON** | **OFF** | **JDK** { **'1.1.8'** | **'1.3'** } }]
[**JCONNECT** { **ON** | **OFF** }]
]

page-size :
1024 | **2048** | **4096** | **8192** | **16384** | **32768**

collation-label : string

key-spec:
[**ON**] **KEY** *key* [**ALGORITHM** { **'AES'** | **'MDSR'** }]

参数

File name 文件名（*db-file-name-string*、*log-file-name-string* 和 *mirror-file-name-string*）是包含操作系统文件名的字符串。作为字符串，它们必须用单引号引起来。

- 如果指定路径，则任何后跟 **n** 或 **x** 的反斜线字符 (\) 都必须是两个。将它们转义后可防止按照 SQL 中的字符串规则将它们解释成换行符 **(\n)** 或十六进制数 **(\x)**。

  更安全的做法是始终对反斜线字符进行转义。例如，

  ```
  CREATE DATABASE 'c:\\sybase\\my_db.db'
  LOG ON 'e:\\logdrive\\my_db.log'
  ```

- 如果指定相对路径或者未指定路径，数据库文件将在相对于服务器工作目录的位置上创建。如果没有指定日志文件的路径，则日志文件在数据库文件所在的同一目录中创建。

- 如果未提供文件扩展名，则创建的文件带不同的扩展名：对数据库带 .db，对事务日志带 .log，对镜像日志带 .mlg。

TRANSACTION LOG 子句 事务日志是数据库服务器用来记录所有数据库更改的文件。事务日志在备份和恢复（请参阅 "ASA 数据库管理指南 > 备份和数据恢复 > 事务日志 "）以及在数据复制中起着非常重要的作用。

MIRROR 子句 事务日志镜像是事务日志的相同副本，通常在单独的设备上维护，以便更好地保护数据。缺省情况下， Adaptive Server Anywhere 不使用镜像事务日志。如果确实希望使用事务日志镜像，可以通过此选项提供一个文件名。

CASE 子句 对于用 CASE RESPECT 创建的数据库而言，在比较和字符串操作中，所有的值都区分大小写。

提供此选项是为了与 ISO/ANSI SQL 标准兼容。此选项的缺省值是 CASE IGNORE，即所有的比较都不区分大小写。如果创建了区分大小写的数据库，所有的口令也区分大小写。数据库中的用户 ID 和其它标识符不区分大小写，即使是在区分大小写的数据库中。

PAGE SIZE 子句 数据库的页面大小可以是 1024、 2048、 4096、 8192、 16384 或 32768 个字节。缺省的页面大小是 2048 个字节。大型数据库通常可从较大的页面大小中获益，但也将负担与较大的页面大小相关的额外开销。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库管理实用程序 > Information 实用程序选项 "。

例如，

```
CREATE DATABASE 'c:\\sybase\\my_db.db'
PAGE SIZE 4096
```

页面大小限制
页面大小不能超出当前服务器所使用的页面大小。服务器页面大小可从启动的第一个数据库集中获取，或者在服务器命令行上使用 -gp 选项来设置。

COLLATION 子句 用于数据库中所有字符串比较的归类序列。

有关归类序列的更多信息，请参阅 "ASA 数据库管理指南 > 国际语言和字符集 "。

ENCRYPTED 子句 加密后，存储在物理数据库文件中的数据将无法阅读。有两个级别的加密：

简单加密相当于打乱数据。虽然这些数据无法阅读，但了解加密技术的人可能会破解数据。简单加密是通过指定不带 KEY 子句的 ENCRYPTED 子句来实现的。

通过使用 128 位算法和安全密钥，可以实现高度加密。数据无法阅读，而且如果没有密钥，基本上无法破解数据。要创建高度加密的数据库，请指定带 KEY 子句的 ENCRYPTED 子句。像大多数口令一样，最好选择不容易猜出来的 KEY 值。建议选择满足以下条件的 KEY 值：长度至少为 16 个字符，包含大小写混合以及包含数字、字母和特殊字符。

每当希望启动数据库时，都需要此密钥。

与 ENCRYPTED 和 KEY 子句一起使用 ALGORITHM 子句可以指定加密算法。可以选择 AES 或 MDSR 算法。如果使用了 ENCRYPTED 子句但未指定任何算法，则缺省值为 AES。

注意
保护好密钥！请务必将密钥的副本保存在一个安全的位置。如果遗失了密钥，数据库将彻底无法访问，这种情况下没有任何恢复办法。

BLANK PADDING 子句 如果指定了 BLANK PADDING ON，比较时将忽略尾随空白。例如，在用 BLANK PADDING ON 创建的数据库中，以下两个字符串

```
'Smith'
'Smith   '
```

被视作相等。

提供此选项是为了与 ISO/ANSI SQL 标准兼容，后者在比较中忽略尾随空白。缺省是比较空白 （即 BLANK PADDING OFF）。

ASE COMPATIBLE 子句 不会创建 SYS.SYSCOLUMNS 和 SYS.SYSINDEXES 视图。缺省情况下，创建这些视图是为了与 Watcom SQL（该软件的版本 4 和更早版本）中可用的系统表兼容。这些视图与 Sybase Adaptive Server Enterprise 兼容性视图 *dbo.syscolumns* 和 *dbo.sysindexes* 冲突。

JCONNECT 子句 如果希望用 Sybase jConnect JDBC 驱动程序访问系统目录信息，需要安装 jConnect 支持。如果希望排除 jConnect 系统对象，请指定 JCONNECT OFF。如果不访问系统信息，仍然可以使用 JDBC。

JAVA 子句 缺省行为是 **JAVA OFF**。

要在数据库中使用 Java，必须将 Sybase 运行时 Java 类的条目安装到系统表中。如果显式指定 **JAVA JDK '1.1.8'** 或 **JAVA JDK '1.3'**，则将安装指定 JDK 版本的条目。对于 JDK 1.1.8，上述类保存在 SQL Anywhere 目录下的 `java\1.1\classes.zip` 中。对于 JDK 1.3，这些类保存在 `java\1.3\rt.jar` 中。缺省值是 JDK 1.3 类。

数据库中的 Java 是单独授权的组件。有关更多信息，请参阅 "ASA 编程指南 > 介绍数据库中的 Java"。

用法

创建具有所提供的名称和属性的数据库文件。

权限

执行此语句所需的权限在服务器命令行上用 -gu 选项来设置。缺省设置为要求具有 DBA 权限。

运行服务器时所使用的帐户必须对创建文件的目录具有写权限。

在 Windows CE 上不受支持。

副作用

创建操作系统文件。

另请参阅

"ALTER DATABASE 语句 " 第 242 页

"DROP DATABASE 语句 " 第 447 页

"ASA 数据库管理指南 > 数据库管理实用程序 > Initialization 实用程序 "

"ASA 数据库管理指南 > 连接参数和通信参数 > Encryption Key 连接参数 [DBKEY]"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Enterprise 提供了 CREATE DATABASE 语句，但带有不同的选项。

示例

下面的语句在 C:\ 目录中创建名为 mydb.db 的数据库文件。

```
CREATE DATABASE 'C:\\mydb'
TRANSACTION LOG ON
CASE IGNORE
PAGE SIZE 1024
COLLATION '437'
ENCRYPTED OFF
BLANK PADDING OFF
JAVA JDK '1.3'
JCONNECT OFF
```

下面的语句创建一个不含 Sybase 运行时 Java 类的数据库。除了涉及 Java 类或对象的操作，所有的数据库操作都将正常执行。

```
CREATE DATABASE 'C:\\nojava'
JAVA OFF
```

CREATE DBSPACE 语句

说明

此语句用于定义新的数据库空间并创建关联的数据库文件。

语法

CREATE DBSPACE *dbspace-name* **AS** *filename*

参数

dbspace-name 数据库文件的内部名称。 *filename* 参数是数据库文件的实际名称，有时需要包括路径。

filename 如果 *filename* 没有带显式目录，则在主数据库文件所在的同一目录中进行创建。任何相对目录都相对于主数据库文件。 *filename* 相对于数据库服务器。在使用 NetWare 数据库服务器时，如果指定了绝对目录，则 *filename* 应该使用卷名 （而不是驱动器号）。

用法

CREATE DBSPACE 语句创建新的数据库文件。创建数据库时，它仅包含一个文件。所有创建的表和索引都放在这个文件中。 CREATE DBSPACE 在数据库中添加新文件。该文件可以位于和主文件不同的磁盘驱动器上，这意味着数据库的大小可以超过一个物理设备。

对于每个数据库，除了主文件外，最多还可以有 12 个 dbspace。

每个表应完全包含在一个数据库文件中。 CREATE TABLE 语句的 IN 子句指定将表放在哪一个 dbspace 中。缺省情况下，表放在主数据库文件中。

权限

必须拥有 DBA 权限。

副作用

自动提交。自动执行检查点操作。

另请参阅

"DROP 语句 " 第 445 页

"ASA 数据库管理指南 > 使用数据库文件 > 使用附加 dbspace"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

创建名为 *library* 的 dbspace 来保存 *LibraryBooks* 表及其索引。

```
CREATE DBSPACE library
AS 'e:\\dbfiles\\library.db';
CREATE TABLE LibraryBooks (
  title char(100),
  author char(50),
  isbn char(30),
) IN library;
```

CREATE DECRYPTED FILE 语句

说明

此语句用于解密高度加密的数据库。

语法

CREATE DECRYPTED FILE *newfile*
FROM *oldfile***KEY**
key

参数

FROM　列出加密文件的文件名。

KEY　列出访问加密文件所需的密钥。

用法

此语句将加密的数据库、事务日志文件或 dbspace 解密并创建新的解密文件。原始文件必须用加密密钥高度加密。得到的文件是加密文件的相同副本，但没有加密，因此不需要加密密钥。

如果使用此语句对数据库进行解密，则相应的事务日志文件 （以及任何 dbspace）也必须被解密才能使用数据库。

如果需要恢复的数据库是解密的，则其事务日志必须也被解密，并且仍需要恢复新数据库。

在上述过程中，事务日志文件的名称保持不变。因此，如果数据库和事务日志文件被重命名，则需要对得到的数据库运行 dblog -t 命令。

如果您想为一个现有数据库加密，则需要使用 CREATE ENCRYPTED FILE 语句，或者卸载数据库然后联合使用 -an 选项与 -ek 或 -ep 来重装数据库。使用这种方法还可以更改现有的加密密钥。

权限

- 必须是具有 DBA 权限的用户。

副作用

无。

示例

下面的示例对 *contacts* 数据库进行解密，并新建一个名为 *contacts2* 的解密数据库。

```
CREATE DECRYPTED FILE 'contacts2.db'
FROM 'contacts.db'
KEY 'Sd8f6654*Mnn'
```

CREATE DOMAIN 语句

说明

此语句用于在数据库中创建域。

语法

CREATE { **DOMAIN** | **DATATYPE** } [**AS**] *domain-name data-type*
[[**NOT**] **NULL**]
[**DEFAULT** *default-value*]
[**CHECK (** *condition* **)**]

domain-name : identifier

data-type : built-in data type, with precision and scale

参数

DOMAIN | DATATYPE 建议使用 CREATE DOMAIN 而不是 CREATE DATATYPE，因为 CREATE DOMAIN 是 ANSI/ISO SQL3 术语。

NULL 缺省情况下，域允许 NULL，除非 **allow_nulls_by_default** 选项设置为 OFF。这个时候，缺省情况下新域不允许 NULL。在域中创建的列的为空性取决于该域的定义设置，而不是引用该列时的 **allow_nulls_by_default** 选项设置。任何在列定义中显式设置的 NULL 或 NOT NULL 都取代域设置。

CHECK 子句 创建 CHECK 条件时，可以在条件中使用一个带 @ 符号前缀的变量名。如果使用该数据类型定义列，这样的变量将被列名称替换。这使 CHECK 条件可以在数据类型上定义并由任何名称的列使用。

用法

域是内部数据类型的别名，对有些数据类型，包括精度值和小数位数。它们提高了方便程度并有利于数据库的一致性。

域是数据库中的对象。其名称必须遵守标识符规则。与内部数据类型的名称一样，域名称始终不区分大小写。

创建数据类型的用户自动成为该数据类型的所有者。在 CREATE DATATYPE 语句中不能指定所有者。域名称必须是唯一的，所有用户都可以访问这个数据类型且不必将所有者作为前缀。

域可以有 CHECK 条件和 DEFAULT 值，您可以指示数据类型是否允许 NULL 值。在数据类型上创建的任何列都将继承这些条件和值。在列上显式指定的任何条件或值都取代为数据类型指定的条件或值。

要从数据库中删除数据类型，请使用 DROP 语句。必须是数据类型的所有者或者具有 DBA 权限才能删除域。

权限

必须有 RESOURCE 权限。

副作用

自动提交。

另请参阅

"DROP 语句 " 第 445 页

"SQL 数据类型 " 第 59 页

标准和兼容性

- **SQL/92**　中间级特性。
- **SQL/99**　核心 SQL 外部的 SQL/ 基础特性。
- **Sybase**　不受 Adaptive Server Enterprise 支持。 Transact-SQL 使用 *sp_addtype* 系统过程以及 CREATE DEFAULT 和 CREATE RULE 语句提供了类似的功能。

示例

下面的语句创建名为 **address** 的数据类型，它包含 35 个字符长的字符串并且可以为 NULL。

```
CREATE DOMAIN address CHAR( 35 ) NULL
```

下面的语句创建名为 **id** 的数据类型，它不允许 NULL 并且在缺省情况下自动递增。

```
CREATE DOMAIN id INT
NOT NULL
DEFAULT AUTOINCREMENT
```

CREATE ENCRYPTED FILE 语句

说明

此语句加密高度加密的数据库、事务日志文件或 dbspace。

语法

CREATE ENCRYPTED FILE *newfile*
FROM *oldfile***KEY**
key
ALGORITHM *algorithm*

参数

FROM 列出未加密文件的文件名。

KEY 列出赋予加密文件的密钥。

ALGORITHM 可以是 *AES*（缺省值）或 *MDSR*。仅在 Win32 平台上支持 *MDSR*。

用法

此语句接受未加密的数据库、事务日志文件或 dbspace 并创建新的加密文件。一定不要加密原文件。结果文件是与原文件完全一样的副本，只是它已用指定的算法和密钥加密。

如果使用此语句对数据库进行加密，则必须使用相同的算法和密钥加密相应的事务日志文件 （以及任何 dbspace），这样才能使用数据库。您不能将加密的文件和未加密的文件混在一起，也不能将使用不同算法或使用不同密钥加密的文件混在一起。

如果需要恢复的数据库是加密的，则其事务日志必须也被加密，并且仍需要恢复新数据库。

在上述过程中，事务日志文件的名称保持不变。因此，如果数据库和事务日志文件被重命名，则需要对得到的数据库运行 dblog -t 命令。

通过卸载数据库然后联合使用 -an 选项与 -ek 或 -ep 重装数据库，可以加密一个现有数据库或更改现有的加密密钥。也可以联合使用 CREATE ENCRYPTED FILE 语句与 CREATE DECRYPTED FILE 语句来更改加密密钥。

权限

- 必须是具有 DBA 权限的用户。

副作用

无。

示例 1

下面的示例对 *contacts* 数据库进行解密，并新建一个名为 *contacts2* 的解密数据库。

```
CREATE ENCRYPTED FILE 'contacts2.db'
FROM 'contacts.db'
KEY 'Sd8f6654*Mnn'
```

示例 2

下面的示例加密 *contacts* 数据库和 *contacts* 日志文件，并重命名这两个文件。您需要运行 dblog -ek abcd -t contacts2.log contacts.db，因为虽然日志已被重命名，但数据库文件仍指向旧的日志。

```
CREATE ENCRYPTED FILE 'contacts2.db'
FROM 'contacts.db'
KEY 'Sd8f6654*Mnn'
CREATE ENCRYPTED FILE 'contacts2.log'
FROM 'contacts.db'
KEY 'Te9g7765*Noo'
```

示例 3

下面的示例加密 *contacts* 数据库和 *contacts* 日志文件，让原日志文件名保持不变。在这种情况下，您不需要运行 dblog，因为文件的名称未改变。

```
CREATE ENCRYPTED FILE 'newpath\contacts.db'
FROM 'contacts.db'
KEY 'Sd8f6654*Mnn'
CREATE ENCRYPTED FILE 'newpath\contacts.log'
FROM 'contacts.log'
KEY 'Sd8f6654*Mnn'
```

示例 4

下面的示例更改了 *contacts* 数据库的加密密钥。

```
CREATE DECRYPTED FILE 'temp.db'
FROM 'contacts.db'
KEY 'oldkey'
del contacts.db
CREATE ENCRYPTED FILE 'contacts.db'
FROM 'temp.db'
KEY 'newkey'
del temp.db
```

CREATE EVENT 语句

说明

此语句用于定义事件及其自动执行预定义操作的关联处理程序。另外，还用于定义调度操作。

语法

CREATE EVENT *event-name*
[**TYPE** *event-type*
[**WHERE** *trigger-condition* [**AND** *trigger-condition*] …]
| **SCHEDULE** *schedule-spec*, …]
[**ENABLE** | **DISABLE**]
[**AT** { **CONSOLIDATED** | **REMOTE** | **ALL** }]
[**HANDLER**
BEGIN
…
END]

event-type :
BackupEnd | **"Connect"**
| **ConnectFailed** | **DatabaseStart**
| **DBDiskSpace** | **"Disconnect"**
| **GlobalAutoincrement** | **GrowDB**
| **GrowLog** | **GrowTemp**
| **LogDiskSpace** | **"RAISERROR"**
| **ServerIdle** | **TempDiskSpace**

trigger-condition :
event_condition(*condition-name* **)** { **=** | **<** | **>** | **!=** | **<=** | **>=** } *value*

schedule-spec :
[*schedule-name*]
{ **START TIME** *start-time* | **BETWEEN** *start-time* **AND** *end-time* }
[**EVERY** *period* { **HOURS** | **MINUTES** | **SECONDS** }]
[**ON** { **(** *day-of-week*, … **)** | **(** *day-of-month*, … **)** }]
[**START DATE** *start-date*]

event-name | *schedule-name* : *identifier*

day-of-week : *string*

day-of-month | *value* | *period* : *integer*

start-time | *end-time* : *time*

start-date : *date*

参数

CREATE EVENT 子句　事件名称是一个标识符。事件都有创建者，即创建事件的用户。事件处理程序只能在创建者的许可下执行。这与存储过程的执行是一样的。您不能创建由其他用户所拥有的事件。

TYPE 子句　可以指定带可选 WHERE 子句的 TYPE 子句，或者指定 SCHEDULE。

event-type 是所列出的一组系统定义事件类型中的一种。事件类型不区分大小写。要指定该 *event-type* 触发事件的条件，请使用 WHERE 子句。

- **DiskSpace 事件类型**　如果数据库包含 DiskSpace 类型之一的事件处理程序，则数据库服务器每隔 30 秒对每一个与相关文件关联的设备上的可用空间进行检查。

  如果数据库包含位于不同驱动器上的多个 dbspace，则 **DBDiskSpace** 检查每个驱动器并根据最低的可用空间执行操作。

  LogDiskSpace 事件类型检查事务日志和任何镜像事务日志的位置，并根据最小的可用空间生成报告。

  在 Windows CE 或非常早的 Windows 95 版本上不支持磁盘空间事件类型。

- **Globalautoincrement 事件类型**　此事件在表的 GLOBAL AUTOINCREMENT 缺省值离它的范围阈值仅差一个百分点时触发。处理程序的典型操作可能是为 GLOBAL_DATABASE_ID 选项请求新值。

  对这种事件类型，可以使用以 **RemainingValues** 为参数的 *event_condition* 函数。

- **ServerIdle 事件类型**　如果数据库包含 **ServerIdle** 类型的事件处理程序，则服务器每隔 30 秒检查一次服务器的活动。

WHERE 子句　此触发条件定义事件在什么情况下触发。例如，如果希望在含有事务日志的磁盘上有 80% 以上已满时采取行动，请使用以下触发条件：

```
...
WHERE event_condition( 'LogDiskSpacePercentFree' ) <
20
...
```

event_condition 函数的参数必须对事件类型有效。

可以使用多个 AND 条件来构成 WHERE 子句，但不能使用 OR 条件或其它条件。

有关有效参数的信息，请参阅 ""EVENT_CONDITION 函数 [系统]" 第 153 页 "。

SCHEDULE 子句 此子句指定调度的操作何时发生。时间序列充当事件处理程序中定义的关联操作的一组触发条件。

对给定事件及其关联的处理程序，可以定义多个调度。这样可以实现复杂的调度。有多个调度时必须提供调度名称，而仅提供一个调度时，这是可选的。

如果调度事件的定义中含有 EVERY 或 ON，则调度事件将反复出现。如果未使用这些保留字，则事件最多执行一次。如果试图创建启动时间已过去的非反复出现的调度事件，就会产生错误。当非反复出现的调度事件过去后，将删除其调度，但不删除事件处理程序。

调度事件的时间从调度创建时开始计算，并在事件处理程序完成执行时重新计算。计算下一次事件时间的方法是，检查事件的调度并找到即将到来的下一次调度时间。如果指示事件处理程序在 9:00 和 5:00 之间每隔一小时运行一次，并且它需要 65 分钟来执行，则它将分别在 9:00、11:00、 1:00、 3:00 和 5:00 时运行。如果希望重叠执行，必须创建多个事件。

以下是调度定义的子子句：

- **START TIME** 在调度了事件的每一天中的最初调度时间。如果指定了 START DATE，则 START TIME 引用该日期。如果未指定 START DATE，则 START TIME 在当天 （除非该时间已经过去）和随后的每一天 （如果调度中含有 EVERY 或 ON）。
- **BETWEEN** … **AND** 显示一天中的一段时间，在这段时间之外没有调度的时间。如果指定了 START DATE，则调度的时间直到该日期才发生。

- **EVERY** 连续调度事件之间的间隔。调度的事件仅在当天的 START TIME 之后或在 BETWEEN … AND 指定的范围内发生。
- **ON** 调度事件发生日的列表。如果指定了 EVERY，则缺省为每天。事件发生日可指定为周内某日或月内某日。

  周内某日是星期一、星期二等等。还可以使用星期周内某日的简写形式，例如 Mon （星期一）。注意，如果您想让使用非英语运行的服务器识别周内某日的名称，则必须使用其英语全名，例如 Monday （星期一）。

  月内某日是从 0 到 31 的整数。 0 值表示任何月的最后一天。
- **START DATE** 调度事件开始发生的日期。缺省为当前日期。

每当调度事件的处理程序完成后，即开始计算下一个调度时间和日期。

1. 如果使用了 EVERY 子句，请查看下一个调度时间是否在当天，以及是否在 BETWEEN … AND 范围之内。如果是，那就是下一个调度时间。
2. 如果下一个调度时间不在当天，请查找下一个执行事件的日期。
3. 查找指定日期的 START TIME，或 BETWEEN … AND 范围的开始时间。

ENABLE | DISABLE 缺省情况下，事件处理程序被启用。指定 DISABLE 后，即使是在调度时间内或满足触发条件时，事件处理程序将不会执行。 TRIGGER EVENT 语句*不* 会使禁用的事件处理程序执行。

AT 子句 如果要在 SQL Remote 设置中的远程或统一数据库中执行事件，可以使用此子句限制在哪一个数据库中处理事件。缺省情况下，所有数据库都可以执行事件。

HANDLER 子句 每个事件都有一个处理程序。

用法

主要有两种使用事件的方式：

- **调度操作** 数据库服务器按时间调度执行一组操作。可以使用此功能调度备份、有效性检查、报告表填充查询等。

- **事件处理操作** 数据库服务器在预定义的事件发生时执行一组操作。可以处理的事件包括磁盘空间限制 （当磁盘使用超过了指定的百分比后）、服务器何时空闲等。

事件定义包括两个截然不同的方面。触发条件可以是一个事件，如磁盘使用超出了定义的阈值。调度是一组时间，其中每个时间充当一个触发条件。当满足触发条件时，执行事件处理程序。事件处理程序包含一个或多个在复合语句 (BEGIN… END) 内指定的操作。

如果未提供触发条件或调度说明，则只有显式 TRIGGER EVENT 语句可以触发事件。在开发中，可能需要使用 TRIGGER EVENT 来开发和测试事件处理程序，并在测试完成后添加调度或 WHERE 子句。

事件错误记录到数据库服务器主控台。

触发事件处理程序后，服务器通过 *event_parameter* 函数，使上下文信息（比如导致事件被触发的连接 ID）可供事件处理程序使用。有关 event_parameter 的更多信息，请参阅 ""EVENT_PARAMETER 函数 [系统]" 第 155 页 "。

权限

必须拥有 DBA 权限。

在事件所有者的许可下，事件处理程序在一个单独的连接上执行。要使用 DBA 以外的权限执行，可以从事件处理程序内部调用过程：过程用其所有者的权限来执行。这个单独的连接不计入个人数据库服务器只能有 10 个连接的限制中。

副作用

自动提交。

如果在执行期间未检测到任何错误，则将提交事件处理程序的操作。如果检测到错误，将回退。

另请参阅

"BEGIN 语句 " 第 288 页

"ALTER EVENT 语句 " 第 248 页

"COMMENT 语句 " 第 303 页

"DROP 语句 " 第 445 页

"TRIGGER EVENT 语句 " 第 633 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

指示数据库服务器在每天凌晨 1 点，使用在 Windows NT 计算机上找到的第一个磁带驱动器执行自动磁带备份。

```
CREATE EVENT DailyBackup
SCHEDULE daily_backup
START TIME '1:00AM' EVERY 24 HOURS
HANDLER
   BEGIN
      BACKUP DATABASE TO '\\\\.\\tape0'
      ATTENDED OFF
   END
```

指示数据库服务器从星期一到星期五，在每天上午 8 点到下午 6 点，每隔一小时执行一次自动事务日志备份。

```
CREATE EVENT HourlyLogBackup
SCHEDULE hourly_log_backup
BETWEEN '8:00AM' AND '8:00PM'
EVERY 1 HOURS ON

('Monday','Tuesday','Wednesday','Thursday','Friday')
HANDLER
   BEGIN
      BACKUP DATABASE TO 'c:\\database\\backup'
      TRANSACTION LOG ONLY
      TRANSACTION LOG RENAME
   END
```

有关更多示例，请参阅 "ASA 数据库管理指南 > 使用调度和事件自动完成任务 > 定义事件的触发条件 "。

CREATE EXISTING TABLE 语句

说明

此语句用于创建新的代理表。代理表表示远程服务器上的现有对象。

语法

CREATE EXISTING TABLE [*owner*.]*table-name*
[**(***column-definition*, …**)**]
AT *location-string*

column-definition :
column-name data-type [**NOT NULL**]

location-string :
remote-server-name.[*db-name*].[*owner*].*object-name*
| *remote-server-name*;[*db-name*];[*owner*];*object-name*

参数

AT 子句 AT 子句指定远程对象的位置。 AT 子句支持将分号 (;) 用作分隔符。如果分号出现在表示位置的字符串中，则它用作字段分隔符。如果没有分号，则使用句号作为字段分隔符。这允许在数据库和所有者字段中使用文件名和扩展名。例如，下面的语句将表 `a1` 映射到 MS Access 文件 `mydbfile.mdb`：

```
CREATE EXISTING TABLE a1
AT 'access;d:\mydbfile.mdb;;a1'
```

用法

CREATE EXISTING TABLE 语句创建新的本地代理表，该代理表映射到外部位置的表。 CREATE EXISTING TABLE 语句是 CREATE TABLE 语句的变化形式。 EXISTING 关键字与 CREATE TABLE 一起使用时，指定一个表已存在于远程位置，并且其元数据将导入到 Adaptive Server Anywhere 中。以这种方式可将远程表建立为 Adaptive Server Anywhere 用户能够看得见的实体。 Adaptive Server Anywhere 在创建表之前校验它是否存在于外部位置。

如果对象不存在 （主机数据文件或远程服务器对象），此语句将被拒绝并伴随出现错误消息。

主机数据文件或远程服务器表的索引信息被抽取出来用于创建系统表 *sysindexes* 的行。这定义了服务器方面的索引和关键字，使查询优化程序可以考虑该表上可能存在的任何索引。

参照约束在适当的时候传递到远程位置。

如果未指定列定义， Adaptive Server Anywhere 会根据它从远程表中获得的元数据来派生列列表。如果指定了列定义， Adaptive Server Anywhere 会校验列定义。对列名称、数据类型、长度、标识属性和空值属性将作如下检查：

- 列名称必须完全匹配 （但忽略大小写）。
- CREATE EXISTING TABLE 语句中的数据类型必须匹配或者转换成远程位置的列的数据类型。例如，本地列的数据类型定义为货币，而远程列的数据类型为数字。
- 检查每列的 NULL 属性。如果本地列的 NULL 属性与远程列的 NULL 属性不同，则会发出警告信息，但语句不中止。
- 检查每列的长度。如果 char、 varchar、二进制、 varbinary、十进制和数字类型的列的长度不匹配，则会发出警告信息，但命令不中止。

  可以选择在 CREATE EXISTING 语句中仅包含实际远程列列表的子集。

权限

必须有 RESOURCE 权限。若要为其他用户创建表，必须有 DBA 权限。

在 Windows CE 上不受支持。

副作用

自动提交。

另请参阅

CREATE TABLE 语句

"ASA SQL 用户指南 > 访问远程数据 > 指定代理表位置 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Open Client/Open Server 支持。

示例

在远程服务器 *server_a* 上，为 *blurbs* 表创建名为 *blurbs* 的代理表。

```
CREATE EXISTING TABLE blurbs
( author_id id not null,
copy text not null)
AT 'server_a.db1.joe.blurbs'
```

在远程服务器 *server_a* 上，为 *blurbs* 表创建名为 *blurbs* 的代理表。Adaptive Server Anywhere 根据它从远程表中获得的元数据派生列列表。

```
CREATE EXISTING TABLE blurbs
AT 'server_a.db1.joe.blurbs'
```

在 Adaptive Server Anywhere 远程服务器 *asademo* 上，为 *employee* 表创建名为 *rda_employee* 的代理表。

```
CREATE EXISTING TABLE rda_employee
AT 'asademo..DBA.employee'
```

CREATE EXTERNLOGIN 语句

说明 此语句用于指派与远程服务器通信时使用的替代登录名和口令。

语法

CREATE EXTERNLOGIN *login-name*
TO *remote-server*
REMOTE LOGIN *remote-user*
[**IDENTIFIED BY** *remote-password*]

参数

login-name 指定本地用户的登录名。如果使用集成登录， *login-name* 是 Windows 用户 ID 映射到的数据库用户。

TO 子句 TO 子句指定远程服务器的名称。

REMOTE LOGIN 子句 REMOTE LOGIN 子句在远程服务器上为本地用户 *login-name* 指定用户帐户。

IDENTIFIED BY 子句 IDENTIFIED BY 子句指定 *remote-user* 的 *remote-password*。 *remote-user* 和 *remote-password* 组合必须在远程服务器上有效。

用法 缺省情况下， Adaptive Server Anywhere 每次代表其客户端连接到远程服务器时都使用客户端的名称和口令。 CREATE EXTERNLOGIN 指派与远程服务器通信时使用的替代登录名和口令。

口令以加密形式存储在内部。 *remote-server* 必须通过 *SYSERVERS* 表中的条目被本地服务器知道。有关更多信息，请参阅 "CREATE SERVER 语句 " 第 364 页。

具有自动口令失效功能的站点应该就定期更新外部登录口令作出计划。

CREATE EXTERNLOGIN 不能从事务内部使用。

权限 只有登录名和 DBA 帐户可以添加或修改登录名的外部登录。

在 Windows CE 上不受支持。

副作用 自动提交。

另请参阅 "DROP EXTERNLOGIN 语句 " 第 449 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Open Client/Open Server 支持。

示例

连接到服务器 **sybase1** 时，将名为 **DBA** 的本地用户映射到用户 **sa**，口令为 **Plankton**。

```
CREATE EXTERNLOGIN DBA
TO sybase1
REMOTE LOGIN sa
IDENTIFIED BY Plankton
```

CREATE FUNCTION 语句

说明

此语句用于在数据库中创建新函数。

语法

CREATE FUNCTION [*owner*.]*function-name* **(** [*parameter*, …] **)**
RETURNS *data-typeroutine-characteristics*

{ *compound-statement* | *external-name* }

parameter :
[**IN**] *parameter-name data-type*

routine-characteristics
ON EXCEPTION RESUME | [**NOT**] **DETERMINISTIC**

external-name:
EXTERNAL NAME *library-call*
| **EXTERNAL NAME** *java-call* **LANGUAGE JAVA**

library-call :
'[*operating-system*:]*function-name@library.dll*; …'

operating-system :
Windows95 | **WindowsNT** | **NetWare** | **UNIX**

java-call :
'[*package-name*.]*class-name.method-name method-signature*'

method-signature :
([*field-descriptor*, …] **)** *return-descriptor*

field-descriptor | *return-descriptor* :
Z | **B** | **S** | **I** | **J** | **F** | **D** | **C** | **V** | **[***descriptor* | **L***class-name***;**

参数

CREATE FUNCTION 子句　参数名必须符合数据库标识符规则。它们必须有有效的 SQL 数据类型，而且必须有关键字 IN 作为前缀，以表明参数是为函数提供值的表达式。

EXTERNAL NAME 子句　使用 EXTERNAL NAME 子句的函数是关于外部库中的函数调用的包装。使用 EXTERNAL NAME 的函数的 RETURNS 子句后可以没有其它子句。

有关外部库调用的信息，请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 从过程调用外部库 "。

EXTERNAL NAME LANGUAGE JAVA 子句 使用带 LANGUAGE JAVA 子句的 EXTERNAL NAME 的函数是关于 Java 方法的包装。

有关调用 Java 过程的信息，请参阅 ""CREATE PROCEDURE 语句 " 第 347 页 "。

ON EXCEPTION RESUME 子句 使用类似于 Transact-SQL 的错误处理。有关更多信息，请参阅 "CREATE PROCEDURE 语句 " 第 347 页。

NOT DETERMINISTIC 子句 被指定为 NOT DETERMINISTIC 的函数每次在查询中被调用时都重新求值。不是以这种方式指定的函数的结果可以缓存起来以便提高性能。每次在查询求值过程中使用相同的参数调用函数时，都会重用缓存的结果。

如果函数有副作用，比如修改基础数据等，应将它们声明为 NOT DETERMINISTIC。例如，一个生成主键值并用在 INSERT …SELECT 语句中的函数应该声明为 NOT DETERMINISTIC：

```
CREATE FUNCTION keygen( increment INTEGER )
RETURNS INTEGER
NOT DETERMINISTIC
BEGIN
  DECLARE keyval INTEGER;
  UPDATE counter SET x = x + increment;
  SELECT counter.x INTO keyval FROM counter;
  RETURN keyval
END
INSERT INTO new_table
SELECT keygen(1), ...
FROM old_table
```

如果函数对给定的输入参数总是返回相同的值，则可以将它们声明为 DETERMINISTIC。在本软件的将来版本中，可能使用此声明来实现对同一输入可能返回不同值的函数的优化，而这些优化在目前可能是不安全的。

用法

CREATE FUNCTION 语句在数据库中创建用户定义的函数。通过指定所有者名称，可以为其他用户创建函数。只要权限允许，可以像其它非集合函数那样使用用户定义的函数。

Adaptive Server Anywhere 将所有用户定义的函数按等幂处理：函数对相同的参数返回一致的结果，而且没有副作用。即，服务器假定使用相同的参数对同一函数的两个连续调用返回相同的结果，对查询的语义没有任何多余的副作用。

权限

必须有 RESOURCE 权限。

外部函数 （包括 Java 函数）必须有 DBA 权限。

副作用

自动提交。

另请参阅

"ALTER FUNCTION 语句 " 第 250 页

"DROP 语句 " 第 445 页

"BEGIN 语句 " 第 288 页

"CREATE PROCEDURE 语句 " 第 347 页

"RETURN 语句 " 第 574 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的函数将 *firstname* 字符串和 *lastname* 字符串串联在一起。

```
CREATE FUNCTION fullname (
   firstname CHAR(30),
   lastname CHAR(30) )
RETURNS CHAR(61)
BEGIN
   DECLARE name CHAR(61);
   SET name = firstname || ' ' || lastname;
   RETURN (name);
END
```

下面的示例说明了 **fullname** 功能的使用。

从两个提供的字符串中返回完整的名称：

```
SELECT fullname ('joe','smith')
```

fullname('joe', 'smith')
joe smith

列出所有雇员的姓名：

```
SELECT fullname (emp_fname, emp_lname)
FROM employee
```

fullname (emp_fname, emp_lname)
Fran Whitney
Matthew Cobb
Philip Chin
Julie Jordan
…

CREATE INDEX 语句

说明

此语句用于创建指定表的索引。索引可以提高数据库的性能。

语法

CREATE [**UNIQUE**] [**CLUSTERED**] **INDEX** *index-name*
　　ON [*owner*.]*table-name*
　　(*column-name* [**ASC** | **DESC**], … **)**
　　[{ **IN** | **ON** } *dbspace-name*]

参数

CLUSTERED 关键字　CLUSTERED 属性使表行按照与索引对应的大概键顺序来存储。当服务器尝试保留键顺序时，不保证实施总体群集。

如果存在聚簇索引， LOAD TABLE 语句便按照索引键的顺序在表中插入行，而 INSERT 语句将按照键顺序的定义尝试将新行添加到包含相邻行的表页上。

有关更多信息，请参阅 "ASA SQL 用户指南 > 使用数据库对象 > 使用聚簇索引 "。

UNIQUE 关键字　UNIQUE 属性确保表中不会有两行的值在所有索引列中相同。每个索引键必须是唯一的，或者至少在一列中包含 NULL。

表的唯一约束和唯一索引是有区别的。唯一索引的列可以为 NULL，而唯一约束的列不能为 NULL。外键可以引用主键或者具有唯一约束的列，但不能引用具有唯一索引的列，因为唯一索引可以包括多个 NULL 实例。

ASC | DESC 选项　除非显式指定降序 (DESC)，否则列以升序 (increasing) 排序。不管索引是升序排列还是降序排列，在执行升序或降序 ORDER BY 操作时都会使用索引。但是，如果通过混合的升序和降序属性来执行 ORDER BY，则仅当索引是用同样的升序和降序属性创建的时才使用索引。

IN | ON 子句　缺省情况下，索引放在它的表所在的同一个数据库文件中。通过指定用来存放索引的 dbspace 名称，可以将索引放在单独的数据库文件中。此特性主要用于帮助大型数据库克服文件大小限制。

有关限制的更多信息，请参阅 "ASA 数据库管理指南 > 物理限制 > 大小和数量限制 "。

用法

CREATE INDEX 语句在指定表的指定列上创建排序索引。索引自动用于改进向数据库发出的查询的性能，以及通过 ORDER BY 子句对查询排序。索引一旦创建，在 SQL 语句中就不会再引用它，除非要校验 (VALIDATE INDEX) 或删除 (DROP INDEX) 它。

不能创建视图索引。

- **索引所有权** 在 CREATE INDEX 语句中无法指定索引所有者。索引总是归表的所有者所拥有。索引名称对每个所有者必须是唯一的。
- **不存在视图索引** 无法为视图创建索引。
- **索引名称空间** 每个索引的名称对给定的表必须是唯一的。
- **排它表使用** 只要 CREATE INDEX 影响了当前正由其它连接使用的表，就会禁止该语句。 CREATE INDEX 可能很耗时，服务器在处理该语句时不会处理引用同一个表的请求。
- **自动创建的索引** Adaptive Server Anywhere 自动创建主键和唯一约束的索引。这些自动创建的索引保存在表所在的同一数据库文件中。

权限

必须是表的所有者，或者有 DBA 权限或 REFERENCES 权限。

表必须是基表或全局临时表。

副作用

自动提交。

另请参阅

"DROP 语句 " 第 445 页

"ASA SQL 用户指南 > 查询优化与执行 > 索引 "

"ASA SQL 用户指南 > 查询优化与执行 > 索引的类型 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Enterprise 的 CREATE INDEX 语句比 Adaptive Server Anywhere 的复杂。虽然在 Adaptive Server Anywhere 中允许使用 Adaptive Server Enterprise 语法，但一些子句和关键字会被忽略。

- Adaptive Server Enterprise 11.5 的完整语法如下：

```
CREATE [ UNIQUE ] [ CLUSTERED | NONCLUSTERED ]
    INDEX index-name
    ON [ [ database.]owner.]table_name
          (column_name [, column_name], …)
     [ WITH {
       {  FILLFACTOR | MAX_ROWS_PER_PAGE } = x,
          CONSUMERS = x,
       … IGNORE_DUP_KEY,
       … SORTED_DATA,
          [ IGNORE_DUP_ROW | ALLOW_DUP_ROW ]
       } ]
     [ ON segment_name ]
```

Adaptive Server Enterprise 索引可以是聚簇的或非聚簇的。聚簇索引在数据检索上几乎总是比非聚簇索引快。每个表只能有一个聚簇索引。

Adaptive Server Anywhere 不支持聚簇索引。 Adaptive Server Anywhere 允许使用 CLUSTERED 和 NONCLUSTERED 关键字，但不采取任何操作。

Adaptive Server Anywhere 还允许使用 （通过忽略）以下关键字：

- FILLFACTOR
- IGNORE_DUP_KEY
- SORTED_DATA
- IGNORE_DUP_ROW
- ALLOW_DUP_ROW

在 Adaptive Server Enterprise 和 Adaptive Server Anywhere 中，索引的物理放置方式是不同的。 Adaptive Server Anywhere 支持 **ON** *segment-name* 子句，但 *segment-name* 引用 dbspace。

Adaptive Server Anywhere 中的唯一索引允许包含 NULL，其他方面都相同。 Adaptive Server Enterprise 中的唯一索引允许最多一个空值的行，不允许重复的行。

对于 Adaptive Server Anywhere 和 Enterprise，给定表的索引名称必须都是唯一的。

示例

为 *employee* 表创建一个有两列的索引。

```
CREATE INDEX employee_name_index
ON employee
( emp_lname, emp_fname )
```

为 *sales_order_items* 表的 *prod_id* 列创建索引。

```
CREATE INDEX item_prod
ON sales_order_items
( prod_id )
```

CREATE MESSAGE 语句 [T-SQL]

说明

此语句用于将用户定义的消息添加到 SYSUSERMESSAGES 系统表，供 PRINT 和 RAISERROR 语句使用。

语法

CREATE MESSAGE *message-number* **AS** *message-text*

message-number :　*integer*

message-text :　*string*

参数

message_number　要添加的消息的消息号。用户定义消息的消息号必须为 20000 或更大。

message_text　要添加的消息文本。最大长度为 255 个字节。 PRINT 和 RAISERROR 识别消息文本中的占位符。一条消息可以按任何顺序包含多达 20 个不同的占位符。消息文本发送到客户端时，消息后面的任何参数的格式化内容将取代这些占位符。

将消息翻译成一种具有不同语法结构的语言时，占位符进行编号以允许参数重排序。参数占位符的形式为 "%nn!"：百分号 (%)，后跟一个 1 到 20 之间的整数，然后是一个感叹号 (!)，其中整数代表参数在参数列表中的位置。 "%1!" 是第一个参数， "%2!" 是第二个参数，依此类推。

没有与 **sp_addmessage** 的 *language* 参数相对应的参数。

用法

CREATE MESSAGE 将消息号与消息字符串关联。在 PRINT 和 RAISERROR 语句中可以使用消息号。

要删除消息，请参阅 ""DROP 语句 " 第 445 页 "。

权限

必须有 RESOURCE 权限。

副作用

自动提交。

另请参阅

"PRINT 语句 [T-SQL]" 第 557 页

"RAISERROR 语句 [T-SQL]" 第 560 页

标准和兼容性

- **SQL/92**　供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 在 Adaptive Server Enterprise 中由 **sp_addmessage** 过程提供 CREATE MESSAGE 的特性。

CREATE PROCEDURE 语句

说明　　此语句用于在数据库中创建过程。

语法 1

CREATE PROCEDURE [*owner*.]*procedure-name* **(** [*parameter*, …] **)**
{ [**RESULT (** *result-column*, … **)**]
[**ON EXCEPTION RESUME**]
compound-statement
| **AT** *location-string*
| **EXTERNAL NAME** *library-call*
| [**DYNAMIC RESULT SETS** *integer-expression*]
[**EXTERNAL NAME** *java-call* **LANGUAGE JAVA**]
}

语法 2

CREATE PROCEDURE [*owner*.]*procedure-name* **(** [*parameter*, …] **)**
compound-statement

parameter :
parameter_mode parameter-name data-type [**DEFAULT** *expression*]
| **SQLCODE**
| **SQLSTATE**

parameter_mode : **IN** | **OUT** | **INOUT**

result-column : *column-name data-type*

library-call :
'[*operating-system*:]*function-name@library.dll*; …'

operating-system :
Windows95 | **WindowsNT** | **NetWare** | **UNIX**

java-call :
'[*package-name*.]*class-name.method-name method-signature*'

method-signature :
([*field-descriptor*, …] **)** *return-descriptor*

field-descriptor | *return-descriptor* :
Z | **B** | **S** | **I** | **J** | **F** | **D** | **C** | **V** | **[***descriptor* | **L***class-name***;**

参数

CREATE PROCEDURE 子句 参数名必须符合其它数据库标识符 （如列名）的规则。它们必须是有效的 SQL 数据类型 （请参阅 ""SQL 数据类型 " 第 59 页 "），并且必须使用关键字 IN、 OUT 或 INOUT 之一作为前缀。这些关键字具有以下含义：

- **IN** ： 此参数是一个为过程提供值的表达式。
- **OUT** ： 此参数是一个可由过程赋值的变量。
- **INOUT** ： 此参数是一个为过程提供值的变量，并且可由过程赋值。

使用 CALL 语句执行过程时，不需要指定所有的参数。如果在 CREATE PROCEDURE 语句中提供了缺省值，缺少的参数会被分配缺省值。如果 CALL 语句中没有提供参数并且没有设置缺省值，则发生错误。

SQLSTATE 和 SQLCODE 是特殊参数，它们在过程结束时输出 SQLSTATE 或 SQLCODE 值 （它们是 OUT 参数）。无论是否指定 SQLSTATE 和 SQLCODE 参数，总是可以在过程调用后立即检查 SQLSTATE 和 SQLCODE 特殊值以测试过程的返回状态。

SQLSTATE 和 SQLCODE 特殊值在执行下一个 SQL 语句时被修改。将 SQLSTATE 或 SQLCODE 作为过程参数提供可允许返回代码存储在变量中。

RESULT 子句 RESULT 子句声明结果集中列的数量和类型。 RESULT 关键字后面括在括号内的列表定义结果的列名和类型。描述 CALL 语句时，嵌入式 SQL DESCRIBE 或 ODBC **SQLDescribeCol** 返回此信息。 ""SQL 数据类型 " 第 59 页 " 中列出了允许的数据类型。

有关从过程中返回结果集的更多信息，请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 从过程返回结果 "。

因执行的方式不同，有些过程包含多个结果集，并且列数也不同。例如，下面的过程在有些情况下返回两列，在有些情况下返回一列。

```
CREATE PROCEDURE names( IN formal char(1))
BEGIN
   IF formal = 'n' THEN
      SELECT emp_fname
      FROM employee
```

```
    ELSE
       SELECT emp_lname,emp_fname
       FROM employee
    END IF
END
```

包含可变结果集的过程必须不带 RESULT 子句写出，或者用 Transact-SQL 写出。它们的使用受以下限制：

- **嵌入式 SQL**　打开结果集的游标之后，但在返回任何行之前，必须用 DESCRIBE 描述过程调用，才能获得正确的结果集形状。需要 DESCRIBE 语句的 **CURSOR** *cursor-name* 子句。
- **ODBC**　ODBC 应用程序可以使用可变结果集过程。结果集的正确描述由 ODBC 驱动程序完成。
- **Open Client 应用程序**　Open Client 应用程序可以使用可变结果集过程。

如果过程仅返回一个结果集，则应使用 RESULT 子句。该子句可防止 ODBC 和 Open Client 应用程序在游标打开后重新描述结果集。

为处理多个结果集， ODBC 必须描述当前正在执行的游标，而不是过程的已定义结果集。因此， ODBC 不会始终按过程定义的 RESULT 子句中的定义来描述列名。为避免这种问题，请在生成结果集的 SELECT 语句中使用列的别名。

ON EXCEPTION RESUME 子句　该子句使类似 Transact-SQL 的错误处理能够在 Watcom-SQL 语法过程中使用。

如果使用 ON EXCEPTION RESUME，则过程会根据 ON_TSQL_ERROR 选项的设置采取不同的操作。如果 ON_TSQL_ERROR 设置为 CONDITIONAL （缺省设置），则当下一条语句处理错误，继续执行；否则将退出。

错误处理语句包括以下这些：

- IF
- SELECT @variable =
- CASE

- LOOP
- LEAVE
- CONTINUE
- CALL
- EXECUTE
- SIGNAL
- RESIGNAL
- DECLARE
- SET VARIABLE

在 ON EXCEPTION RESUME 子句中不要使用显式错误处理代码。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > ON_TSQL_ERROR 选项 [compatibility]"。

EXTERNAL NAME 子句 使用 EXTERNAL NAME 子句的过程是关于外部库调用的包装。使用 EXTERNAL NAME 的存储过程在参数列表后可以不使用其它子句。

有关外部库调用的信息，请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 从过程调用外部库 "。

AT location-string 子句 在当前数据库中为 *location-string* 指定的远程过程创建代理存储过程。 AT 子句支持分号 (;) 作为 *location-string* 中的字段分隔符。如果没有分号，则使用句号作为字段分隔符。这允许在数据库和所有者字段中使用文件名和扩展名。

例如，下面的语句创建代理过程 (*remotewho*)，此过程对 *bostonase* 服务器的 *master* 数据库调用 *dbo.sp_who* 过程：

```
CREATE PROCEDURE remotewho ()
AT 'bostonase.master.dbo.sp_who
```

远程过程在输出变量中最多只能返回 254 个字符。

有关远程服务器的信息，请参阅 ""CREATE SERVER 语句 " 第 364 页 "。有关使用远程过程的信息，请参阅 "ASA SQL 用户指南 > 访问远程数据 > 使用远程过程调用 (RPC)"。

DYNAMIC RESULT SETS 子句　该子句与属于 Java 方法包装的过程一起使用。如果不提供 DYNAMIC RESULT SETS 子句，则假定此方法不返回结果集。

EXTERNAL NAME LANGUAGE JAVA 子句　使用带 LANGUAGE JAVA 子句的 EXTERNAL NAME 的过程是关于 Java 方法的包装。

如果参数数目比方法签名中指出的数目少，则差值必须等于 DYNAMIC RESULT SETS 中指定的数目，并且方法签名中超出过程参数列表中的参数的每个参数必须具有方法签名 **[Ljava/SQL/ResultSet;**。

Java 方法签名是参数类型和返回值类型的压缩字符表示形式。

field-descriptor 和 *return-descriptor* 有以下含义：

字段类型	Java 数据类型
B	byte
C	char
D	double
F	float
I	int
J	long
L*class-name***;**	类 *class-name* 的实例。类名必须是完全限定的，而且名称中的任何点都必须由 /. 代替。例如， **java/lang/String**
S	short
V	void
Z	Boolean
[	数组的每个维度都使用一个

例如，

```
double some_method(
  boolean a,
  int b,
  java.math.BigDecimal c,
  byte [][] d,
  java.SQL.ResultSet[] rs ) {
}
```

有如下签名：

```
'(ZILjava/math/BigDecimal;[[B[Ljava/SQL/ResultSet;)D'
```

有关更多信息，请参阅 "ASA 编程指南 > 在数据库中使用 Java > 从 Java 方法返回结果集 "。

用法

CREATE PROCEDURE 语句在数据库中创建过程。有 DBA 权限的用户可以通过指定 **owner** 为其他用户创建过程。过程由 CALL 语句激活。

权限

必须有 RESOURCE 权限。

引用外部过程或者为其他用户创建过程必须有 DBA 权限。

副作用

自动提交。

另请参阅

"BEGIN 语句 " 第 288 页

"CALL 语句 " 第 294 页

"CREATE FUNCTION 语句 " 第 337 页

"CREATE PROCEDURE 语句 [T-SQL]" 第 355 页

"DROP 语句 " 第 445 页

"EXECUTE 语句 [SP]" 第 467 页

"GRANT 语句 " 第 497 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

标准和兼容性

- **SQL/92** 持久存储模块特性。

- **SQL/99** 持久存储模块特性。
- **Sybase** Transact-SQL CREATE PROCEDURE 语句与此不同。
- **SQLJ** 提议的 SQLJ1 标准中指定了 Java 结果集的语法扩展。

示例

下面的过程使用 case 语句对查询结果归类。

```
CREATE PROCEDURE ProductType (IN product_id INT, OUT
type CHAR(10))
BEGIN
   DECLARE prod_name CHAR(20);
   SELECT name INTO prod_name FROM "DBA"."product"
   WHERE id = product_id;
   CASE prod_name
   WHEN 'Tee Shirt' THEN
      SET type = 'Shirt'
   WHEN 'Sweatshirt' THEN
      SET type = 'Shirt'
   WHEN 'Baseball Cap' THEN
      SET type = 'Hat'
   WHEN 'Visor' THEN
      SET type = 'Hat'
   WHEN 'Shorts' THEN
      SET type = 'Shorts'
   ELSE
      SET type = 'UNKNOWN'
   END CASE;
END
```

下面的过程使用游标并循环通过游标的行以返回单值。

```
CREATE PROCEDURE TopCustomer (OUT TopCompany CHAR(35),
OUT TopValue INT)
BEGIN
   DECLARE err_notfound EXCEPTION
   FOR SQLSTATE '02000';
   DECLARE curThisCust CURSOR FOR
   SELECT company_name, CAST(
sum(sales_order_items.quantity *
   product.unit_price) AS INTEGER) VALUE
   FROM customer
   LEFT OUTER JOIN sales_order
   LEFT OUTER JOIN sales_order_items
   LEFT OUTER JOIN product
   GROUP BY company_name;
```

```
   DECLARE ThisValue INT;
   DECLARE ThisCompany CHAR(35);
   SET TopValue = 0;
   OPEN curThisCust;
   CustomerLoop:
   LOOP
      FETCH NEXT curThisCust
      INTO ThisCompany, ThisValue;
      IF SQLSTATE = err_notfound THEN
         LEAVE CustomerLoop;
      END IF;
      IF ThisValue > TopValue THEN
         SET TopValue = ThisValue;
         SET TopCompany = ThisCompany;
         END IF;
   END LOOP CustomerLoop;
   CLOSE curThisCust;
END
```

CREATE PROCEDURE 语句 [T-SQL]

说明

此语句用于在数据库中创建新过程，采用与 Adaptive Server Enterprisese 兼容的方式。

语法 1

Adaptive Server Anywhere 支持 Transact-SQL CREATE PROCEDURE 语句的下列子集。

CREATE PROCEDURE [*owner*.]*procedure_name*
[[**(**] *@parameter_name data-type* [= *default*] [**OUTPUT**], … [**)**]]
[**WITH RECOMPILE**] **AS** *statement-list*

用法

下面列出 Transact-SQL 与 Adaptive Server Anywhere 语句 (Watcom-SQL) 之间的差异，以帮助用这两种方言编写代码的人员。

- **带 @ 前缀的变量名** "@" 符表示 Transact-SQL 变量名，而 Watcom-SQL 变量可以是任何有效的标识符，也可以带 @ 前缀。
- **输入和输出参数** Watcom-SQL 过程参数指定为 IN、 OUT 或 INOUT，而 Transact-SQL 过程参数缺省为 INPUT 参数，也可以指定为 OUTPUT。在 Adaptive Server Anywhere 中声明为 INOUT 或 OUT 的参数在 Transact-SQL 中应声明为 OUTPUT。
- **参数缺省值** Watcom-SQL 使用关键字 DEFAULT 为过程参数提供缺省值，而 Transact-SQL 使用等号 (=) 提供缺省值。
- **返回结果集** Watcom-SQL 使用 RESULT 子句指定返回的结果集。在 Transact-SQL 过程中，第一个查询的列名或别名返回到调用环境。

  下面的 Transact-SQL 过程阐释了结果集如何从 Transact-SQL 存储过程返回：

```
CREATE PROCEDURE showdept @deptname varchar(30)
AS
SELECT employee.emp_lname, employee.emp_fname
FROM department, employee
WHERE department.dept_name = @deptname
AND department.dept_id = employee.dept_id
```

  下面是相应的 Watcom-SQL 过程：

```
CREATE PROCEDURE showdept(in deptname
     varchar(30) )
RESULT ( lastname char(20), firstname char(20))
ON EXCEPTION RESUME
BEGIN
SELECT employee.emp_lname, employee.emp_fname
FROM department, employee
WHERE department.dept_name = deptname
AND department.dept_id = employee.dept_id
END
```

- **过程主体** Transact-SQL 过程的主体是一系列以关键字 AS 开始的 Transact-SQL 语句。 Watcom-SQL 过程的主体是关键字 BEGIN 和 END 之间的复合语句。

权限

必须有 RESOURCE 权限。

副作用

自动提交。

另请参阅

"CREATE PROCEDURE 语句 " 第 347 页

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** Anywhere 支持 Adaptive Server Enterprise CREATE PROCEDURE 语句语法的子集。

  如果提供了 Transact-SQL WITH RECOMPILE 可选子句，它将被忽略。数据库启动后，当过程第一次执行时， Adaptive Server Anywhere 总是重新编译它们，并存储编译的过程直到数据库停止。

  不支持过程组。

CREATE PUBLICATION 语句

说明

此语句可用于创建发布。在 MobiLink 中，发布标识 UltraLite 或 Adaptive Server Anywhere 远程数据库中的同步数据。在 SQL Remote 中，发布标识统一数据库和远程数据库中的复制数据。

语法

CREATE PUBLICATION [*owner*.]*publication-name*
(TABLE *article-description*, … **)**

owner, *publication-name* : *identifier*

article-description:
table-name [**(** *column-name*, … **)**]
[**WHERE** *search-condition*]
[**SUBSCRIBE BY** *expression*]

参数

项目说明 发布是由项目构建的。每个项目都是一个表或表的一部分。项目可以是表的垂直分区 （表列的子集）、水平分区 （表行的子集）或垂直与水平分区。

WHERE 子句 WHERE 子句定义项目中包括的表行子集的一种方法。如果同一子集将由发布的所有预订者接收，则 WHERE 子句很有用。

SUBSCRIBE BY 子句 在 SQL Remote 中，定义项目中包括的表行子集的一种方法是使用 SUBSCRIBE BY 子句。该子句允许许多不同的预订者在单个发布定义中接收表的不同行。 MobiLink 同步期间忽略该子句。

在项目定义中，可以组合使用 WHERE 子句与 SUBSCRIB BY 子句，但 SUBSCRIBE BY 子句仅用于 SQL Remote。

用法

此语句仅适用于 MobiLink 和 SQL Remote。

CREATE PUBLICATION 语句在数据库中创建发布。通过指定所有者名称，可以为其他用户创建发布。

在 MobiLink 中，发布在 Adaptive Server Anywhere 远程数据库中是必不可少，但在 UltraLite 数据库中却是可选的。这些发布以及对它们的预订决定了哪些数据将上载到 MobiLink 同步服务器。可以通过直接创建发

布和预订来构造远程数据库。或者，也可以在充当远程数据库模板的 Adaptive Server Anywhere 参考数据库中创建发布和预订，然后使用 MobiLink 抽取实用程序构造远程数据库。

应使用 ALTER SYNCHRONIZATION SUBSCRIPTION 语句或 CREATE SYNCHRONIZATION SUBSCRIPTION 语句中的 ADD OPTION 子句为 MobiLink 发布设置选项。

在 SQL Remote 中，发布是一个双向操作，因为在统一数据库和远程数据库中都可以输入数据。在 SQL Remote 安装中，任何统一数据库和所有远程数据库必须定义相同的发布。从统一数据库中运行 SQL Remote 抽取实用程序时，远程数据库中会自动执行正确的 CREATE PUBLICATION 语句。

权限

必须拥有 DBA 权限。要求可以对语句中涉及的所有表进行独占访问。

副作用

自动提交。

另请参阅

"ALTER PUBLICATION 语句 " 第 254 页

"DROP PUBLICATION 语句 " 第 450 页

"ALTER SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 265 页

"CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 375 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_create_publication 过程 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

下面的语句发布两个表中的所有列和行。

```
CREATE PUBLICATION pub_contact (
   TABLE contact,
   TABLE company
)
```

下面的语句仅发布一个表中的一些列。

```
CREATE PUBLICATION pub_customer (
   TABLE customer ( id, company_name, city )
)
```

下面的语句通过包括一个测试 customer 表的状态列的 WHERE 子句，仅发布活动客户行。

```
CREATE PUBLICATION pub_customer (
   TABLE customer ( id, company_name, city, state )
   WHERE status = 'active'
)
```

下面的语句通过提供预订者值，仅发布一些行。此方法只能用于 SQL Remote。

```
CREATE PUBLICATION pub_customer (
   TABLE customer ( id, company_name, city, state )
   SUBSCRIBE BY state
)
```

创建 SQL Remote 预订时，按如下所示使用预订者值。

```
CREATE SUBSCRIPTION TO pub_customer ( 'NY' )
   FOR jsmith
```

CREATE REMOTE MESSAGE TYPE 语句 [SQL Remote]

说明

此语句用于标识消息链接和从数据库返回外发消息的地址。

语法

CREATE REMOTE MESSAGE TYPE *message-system*
ADDRESS *address*

message-system: **FILE** | **FTP** | **MAPI** | **SMTP** | **VIM**

address: *string*

参数

message-system 支持的消息系统之一。

address 指定消息系统的地址。

用法

消息代理使用支持的消息链接之一，从数据库发送外发消息。只要远程数据库是用抽取实用程序创建的，为使用指定链接的用户返回的消息都将发送到指定的地址。消息代理仅当有链接的远程用户时才启动那些链接。

在指定的消息系统中，地址是发布者的地址。如果是电子邮件系统，则地址字符串必须是有效的电子邮件地址。如果是文件共享系统，地址字符串是 SQLREMOTE 环境变量中设置的目录的子目录，或者是当前目录的子目录（如果未设置目录）。在远程数据库中，可以覆盖 GRANT CONSOLIDATE 语句中的该设置。

初始化实用程序自动创建不带地址的消息类型。与其它 CREATE 语句不同，如果类型已存在， CREATE REMOTE MESSAGE TYPE 语句不给出错误消息，而是变更类型。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"GRANT PUBLISH 语句 [SQL Remote]" 第 504 页

"GRANT REMOTE 语句 [SQL Remote]" 第 505 页

"GRANT CONSOLIDATE 语句 [SQL Remote]" 第 502 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_remote_type 过程 "

"SQL Remote 用户指南 > SQL Remote 管理 > 使用消息类型 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

使用抽取实用程序抽取远程数据库时，下面的语句设置所有文件消息系统的消息接收者，以将消息发送回 company 子目录。

此语句还指示 dbremote 在 company 子目录中查找进来的消息。

```
CREATE REMOTE MESSAGE TYPE file
ADDRESS 'company'
```

CREATE SCHEMA 语句

说明

此语句用于创建数据库用户的表、视图和权限的集合。

语法

CREATE SCHEMA AUTHORIZATION *userid*
[
create-table-statement
| *create-view-statement*
| *grant-statement*
], …

用法

CREATE SCHEMA 语句创建模式。模式是表、视图及其关联的权限的集合。

userid 必须是当前连接的用户 ID。无法为其他用户创建模式。

如果 CREATE SCHEMA 语句中包含的任何语句失败，则回滚整个 CREATE SCHEMA 语句。

CREATE SCHEMA 语句只是一种将各 CREATE 和 GRANT 语句合成为一个操作的方法。数据库中没有创建任何 SCHEMA 数据库对象，要删除对象，必须使用各 DROP TABLE 或 DROP VIEW 语句。要撤消权限，必须对每个授予的权限使用 REVOKE 语句。

各 CREATE 或 GRANT 语句不用语句分隔符分隔。语句分隔符标记 CREATE SCHEMA 语句本身的结束。

必须排序各 CREATE 或 GRANT 语句，以便在授予对象权限之前创建它们。

尽管目前可以为用户创建多个模式，但不建议这样做，将来的版本可能也不支持这样做。

权限

必须有 RESOURCE 权限。

副作用

自动提交。

另请参阅

"CREATE TABLE 语句 " 第 395 页

"CREATE VIEW 语句 " 第 418 页

"GRANT 语句 " 第 497 页

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** Adaptive Server Anywhere 不支持在 CREATE SCHEMA 语句中使用 REVOKE 语句，也不允许在 Transact-SQL 批处理或过程中使用它。

示例

以下 CREATE SCHEMA 语句创建由两个表组成的模式。此语句必须由具有 RESOURCE 权限的用户 ID *sample_user* 执行。如果创建表 *t2* 的语句失败，则两个表都不创建。

```
CREATE SCHEMA AUTHORIZATION sample_user
CREATE TABLE t1 ( id1 INT PRIMARY KEY )
CREATE TABLE t2 ( id2 INT PRIMARY KEY );
```

下列 CREATE SCHEMA 语句中的语句分隔符位于第一个 CREATE TABLE 语句之后。由于语句分隔符标记 CREATE SCHEMA 语句的结束，因此数据库服务器将该示例解释为两个批处理语句。因此，如果创建表 **t2** 的语句失败，仍会创建表 **t1**。

```
CREATE SCHEMA AUTHORIZATION sample_user
CREATE TABLE t1 ( id1 INT PRIMARY KEY );
CREATE TABLE t2 ( id2 INT PRIMARY KEY );
```

CREATE SERVER 语句

说明

此语句用于将服务器添加到 SYSSERVERS 系统表中。

语法

CREATE SERVER *server-name*
CLASS '*server-class*'
USING '*connection-info*'
[**READ ONLY**]

server-class :
ASAJDBC | **ASEJDBC**
| **ASAODBC** | **ASEODBC**
| **DB2ODBC** | **MSSODBC**
| **ORAODBC** | **ODBC**

connection-info :
{ *machine-name*:*port-number* [*/dbname*] | *data-source-name* }

参数

USING 子句 如果使用基于 JDBC 的服务器类，则 USING 子句的形式为 *hostname:portnumber [/dbname]*，其中：

- **hostname** 是运行远程服务器的计算机
- **portnumber** 是远程服务器监听的 TCP/IP 端口号。 Adaptive Server Anywhere 的缺省端口号为 2638。
- **dbname** 对于 Adaptive Server Anywhere 远程服务器，如果不指定 *dbname*，则使用缺省数据库。对于 Adaptive Server Enterprise，缺省值为 **master** 数据库， *dbname* 的替代方法是通过某种其他的方式使用另一个数据库 （例如，在 FORWARD TO 语句中）。

如果使用基于 ODBC 的服务器类，则 USING 子句为 *data-source-name*。data-source-name 是 ODBC 数据源名称。

READ ONLY READ ONLY 子句指定远程服务器是只读数据源。Adaptive Server Anywhere 拒绝任何更新请求。

用法

CREATE SERVER 语句从 Adaptive Server Anywhere 目录中定义远程服务器。

有关服务器类和如何配置服务器的更多信息，请参阅 "ASA SQL 用户指南 > 用于进行远程数据访问的服务器类 "。

权限

必须有 RESOURCE 权限。

在 Windows CE 上不受支持。

副作用

自动提交。

另请参阅

"ALTER SERVER 语句 " 第 258 页

"DROP SERVER 语句 " 第 452 页

"ASA SQL 用户指南 > 用于进行远程数据访问的服务器类 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Open Client/Open Server 支持。

示例

下面的示例创建一个名为 *testasa* 的 Adaptive Server Anywhere 远程服务器，它位于 `apple` 计算机上，并监听端口号 2638，使用：

```
CREATE SERVER testasa
CLASS 'asajdbc'
USING 'apple:2638'
```

下面的示例为基于 JDBC 的 Adaptive Server 创建一个名为 *ase_prod* 的远程服务器。它的计算机名称是 banana，端口号是 3025。

```
CREATE SERVER ase_prod
CLASS 'asejdbc'
USING 'banana:3025'
```

下面的示例为 Oracle 服务器创建一个名为 oracle723 的远程服务器。它的 ODBC 数据源名称是 `oracle723`。

```
CREATE SERVER oracle723
CLASS 'oraodbc'
USING 'oracle723'
```

CREATE STATISTICS 语句

说明

仅在极少情况下使用此语句。它显式地重新创建优化程序使用的统计。

语法

CREATE STATISTICS *table-name* [**(** *column-list* **)**]

用法

此语句重新创建 Adaptive Server Anywhere 用于优化数据库查询的统计。这些统计分析数据库中指定表的数据分布。 CREATE STATISTICS 执行整个表的有序扫描，因此它的运行时间很长。

在极少数情况下，数据库查询的变化非常大，数据分布不一致或数据不停地变化，这时可针对表或列运行 CREATE STATISTICS 以提高性能。这会导致表或列的有序扫描，可能时会使用索引。

CREATE STATISTICS 覆盖现有统计。执行此语句前不必删除统计。

无论表的大小和 MIN_TABLE_SIZE_FOR_HISTOGRAM 的设置如何，CREATE STATISTICS 都为指定的表创建直方图。

也可以使用 LOAD TABLE 语句创建统计。

权限

必须拥有 DBA 权限。

副作用

查询计划可能会更改。

另请参阅

"DROP STATISTICS 语句 " 第 454 页

"ALTER DATABASE 语句 " 第 242 页

"ASA 数据库管理指南 > 数据库选项 > MIN_TABLE_SIZE_FOR_HISTOGRAM 选项 [database]"

"LOAD TABLE 语句 " 第 530 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

CREATE SUBSCRIPTION 语句 [SQL Remote]

说明

此语句用于为用户创建发布预订。

语法

CREATE SUBSCRIPTION
    TO *publication-name* [**(** *subscription-value* **)**]
    FOR *subscriber-id*

publication-name: *identifier*

subscription-value, *subscriber-id*: *string*

subscriber-id: *string*

用法

在 SQL Remote 安装中，数据组织到 **publications** 中以便复制。要接收 SQL Remote 消息，必须为具有 REMOTE 权限的用户 ID 创建 **subscription**。

如果预订中提供了字符串，则它与发布中的每个 SUBSCRIBE BY 表达式进行匹配。预订者接收表达式的值等于所提供的字符串的所有行。

在 SQL Remote 中，发布和预订是一种双向关系。如果在统一数据库中为远程用户创建发布预订，则在远程数据库中也应为统一数据库创建预订。抽取实用程序自动执行此过程。

参数

publication-name　用户预订的发布的名称。这可以包括发布的所有者。

subscription-value　与发布的预订表达式进行比较的字符串。预订者接收预订表达式与预订值相匹配的所有行。

subscriber-id　发布预订者的用户 ID。必须授予此用户 REMOTE 权限。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"DROP SUBSCRIPTION 语句 [SQL Remote]" 第 456 页

"GRANT REMOTE 语句 [SQL Remote]" 第 505 页

"SYNCHRONIZE SUBSCRIPTION 语句 [SQL Remote]" 第 630 页

"START SUBSCRIPTION 语句 [SQL Remote]" 第 619 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_subscription 过程 "

示例

- 下面的语句为用户 **p_chin** 创建发布 **pub_sales** 的预订。预订者接收预订表达式的值为 **Eastern** 的所有行。

```
CREATE SUBSCRIPTION
TO pub_sales ( 'Eastern' )
FOR p_chin
```

CREATE SYNCHRONIZATION DEFINITION 语句 [MobiLink]（不建议使用）

说明

此语句用于指定如何向 MobiLink 同步服务器注册，以及如何标识将从远程数据库上载到统一数据库中的内容。不建议使用此命令。您应当使用 ALTER PUBLICATION 或 ALTER SYNCHRONIZATION SUBSCRIPTION 取代它。

语法

CREATE SYNCHRONIZATION DEFINITION *sync-def-name*
    SITE *ml_username*
    [**TYPE** *sync-type*]
    ADDRESS *network-parameters*
    [**OPTION** *parameter***=***value***,** …]
    (**TABLE** *article-description***,** … **)**

ml_username: identifier

network-parameters: *string*

sync-type: **http** | **https** | **tcpip**

value: *string* | *integer*

article-description:
    table-name [**(** *column-name***,** … **)**]
        [**WHERE** *search-condition*]

参数

SITE 子句　在 MobiLink 安装中唯一标识此远程数据库的名称。

TYPE 子句　此子句指定同步方法。缺省值为 **tcpip**。也可以选择使用 **http** 或 **https**。

ADDRESS 子句　此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关可用参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

OPTION 子句　OPTION 子句允许您设置选项。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

用法

在 Adaptive Server Anywhere 版本 7 数据库中，创建将用作 MobiLink 客户端的同步定义。每个定义指定在 MobiLink 设置中唯一标识该 MobiLink 客户端站点的逻辑名称。另外，每个站点指定如何联系 MobiLink 同步服务器以及远程数据库中的哪些数据将与统一数据库同步。

使用此语句，可以设置选项并选择同步方法。另外，通过列出每个表，后跟列列表和 WHERE 子句，可以指定希望上载的内容。 WHERE 子句是定义同步中将包括的表行子集的一种方法。

权限

必须拥有 DBA 权限。要求可以对语句中涉及的所有表进行独占访问。

副作用

自动提交。

另请参阅

"CREATE PUBLICATION 语句 " 第 357 页

"CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 375 页

"ALTER SYNCHRONIZATION DEFINITION 语句 （不建议使用） " 第 260 页

"DROP SYNCHRONIZATION DEFINITION 语句 [MobiLink] （不建议使用） " 第 457 页

"CREATE SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） " 第 377 页

"CREATE SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）" 第 372 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例

使用 TCP/IP 同步，将 *pet_id* 为 2 的 *Pets* 表以及 *People* 表中的 *person_id*、*fname* 和 *lname* 列从远程数据库上载到统一数据库：

```
CREATE SYNCHRONIZATION DEFINITION mysharedtables
   SITE "demo_sync_site"
   TYPE 'tcpip'
   ADDRESS 'host=localhost;port=2439;'
   OPTION memory='2m', dir='c:\db\logs'
   (table People( person_id, fname, lname ),
      table Pets WHERE pet_id=2);
```

CREATE SYNCHRONIZATION SITE 语句 [MobiLink] （不建议使用）

说明

此语句可用于在 MobiLink 参考数据库中创建站点，在使用 `mlxtract` 实用程序抽取 Adaptive Server Anywhere 远程数据库时将用到此功能。不建议使用此命令。您应当使用 CREATE SYNCHRONIZATION SUBSCRIPTION、 CREATE PUBLICATION 和 CREATE SYNCHRONIZATION USER 取代它。

语法

CREATE SYNCHRONIZATION SITE *ml_username*
    USING *sync-template-name*
    [**TYPE** *sync-type*]
    [**ADDRESS** *network-parameters*]
    [**OPTION** *option***=***value*, …]

ml_username: identifier

sync-template-name: *name*

network-parameters: *string*

sync-type: **http** | **https** | **tcpip**

value: *string* | *integer*

参数

SITE 子句 在 MobiLink 安装中唯一标识此远程数据库的名称。

TYPE 子句 此子句指定同步方法。缺省值为 **tcpip**。也可以选择使用 **http** 或 **https**。

ADDRESS 子句 此子句指定网络参数，包括 MobiLink 同步服务器的位置。此子句的值覆盖同步模板中的值。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

OPTION 子句 OPTION 子句允许设置以下选项。使用以逗号分隔的列表设置适当的选项。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

用法

仅当通过从 Adaptive Server Anywhere 第 7 版参考数据库中抽取的方式创建 Adaptive Server Anywhere 远程数据库时，才使用同步模板和同步站点。

每个远程数据库从同步站点中创建并存储在参考数据库中。每个同步站点都基于单个同步模板，不过许多站点可以使用单个模板。

许多站点共用的信息应该存储在同步模板中，而不是在每个站点中复制。

抽取过程不仅创建远程数据库，还使用相应的同步模板和站点的属性创建同步定义。此信息必须包含该 MobiLink 站点在 MobiLink 系统内的唯一名称。另外，此信息还应指定如何联系 MobiLink 同步服务器以及同步远程数据库中的哪些数据。

此语句用于为每个远程数据库创建单独的同步站点。此语句还允许覆盖模板设置。例如，可以选择与模板中指定的选项不同的选项。

权限

必须拥有 DBA 权限。

副作用

自动提交

另请参阅

"CREATE PUBLICATION 语句 " 第 357 页

"CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 375 页

"ALTER SYNCHRONIZATION SITE 语句 [MobiLink] （不建议使用） " 第 263 页

"DROP SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）" 第 459 页

"CREATE SYNCHRONIZATION DEFINITION 语句 [MobiLink]（不建议使用） " 第 369 页

"CREATE SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） " 第 377 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例

将模板 *mytemplate* 部署到站点 *demo_sync_site*。连接到模板中指定的服务器以外的 MobiLink 同步服务器，并将内存选项设置为 3 MB。

```
CREATE SYNCHRONIZATION SITE 'demo_sync_site'
   USING mytemplate
   ADDRESS 'host=test.internal;port=2439;'
   OPTION memory='3m';
```

使用 Certicom 传输层安全性将模板 *mytemplate* 部署到站点 *demo_sync_site*。

```
CREATE SYNCHRONIZATION SITE 'demo_sync_site'
   USING mytemplate2
   ADDRESS 'host=test.internal;port=2439;
      security=ecc_tls'
```

CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]

说明

此语句可用于在 Adaptive Server Anywhere 数据库中为 MobiLink 用户预订发布。

语法

CREATE SYNCHRONIZATION SUBSCRIPTION
TO *publication-name*
[**FOR** *ml_username*, …]
[**TYPE** *sync-type*]
[**ADDRESS** *network-parameters*]
[**OPTION** *option*=*value*, …]

ml_username: identifier

network-parameters: *string*

sync-type: **http** | **https** | **tcpip** | **ActiveSync**

value: *string* | *integer*

参数

TO 子句 指定发布名称。

FOR 子句 指定一个或多个 MobiLink 用户 ID。

省略此子句将为发布创建缺省预订。如果预订此发布的 MobiLink 用户没有设置新值覆盖这些缺省值，将继承这些缺省值。此功能在从参考数据库中抽取远程数据库时非常有用。

TYPE 子句 此子句指定同步中使用的通信协议。缺省协议为 **tcpip**。

ADDRESS 子句 此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

OPTION 子句 此子句允许为预订设置扩展选项。如果没有提供 FOR 子句，扩展选项将充当发布的缺省值，并被为同步用户设置的任何扩展选项所覆盖。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

用法

在 MobiLink 远程或参考数据库中，此语句用于创建同步预订。

权限

必须拥有 DBA 权限。要求可以对发布中涉及的所有表进行独占访问。

副作用

自动提交。

另请参阅

"CREATE PUBLICATION 语句 " 第 357 页

"CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 8.0 版。

示例

为销售发布创建包含缺省预订值的缺省预订 （通过省略 FOR 子句）。指出 MobiLink 同步服务器的地址并指定只有 Certicom 根证书受信任。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
   TO sales_publication
   ADDRESS 'host=test.internal;port=2439;
      security=ecc_tls'
   OPTION memory='2m';
```

为 MobiLink 用户 *ml_user1* 预订销售发布。将内存选项设置为 3 MB，而不是缺省发布中指定的值。

```
CREATE SYNCHRONIZATION SUBSCRIPTION
   TO sales_publication
   FOR ml_user1
   OPTION memory='3m';
```

CREATE SYNCHRONIZATION TEMPLATE 语句 [MobiLink]（不建议使用）

说明

此语句可用于在 MobiLink 参考数据库中创建模板，在使用 `mlxtract` 实用程序抽取 Adaptive Server Anywhere 远程数据库时将用到此功能。不建议使用此命令。您应当使用 CREATE SYNCHRONIZATION SUBSCRIPTION 和 CREATE PUBLICATION 取代它。

语法

CREATE SYNCHRONIZATION TEMPLATE *sync-template-name*
[**TYPE** *sync-type*]
ADDRESS *network-parameters*
[**OPTION** *option*=*value*]
(**TABLE** *article-description*, … **)**

network-parameters: string

article-description:
table-name [**(** *column-name*, … **)**]
[**WHERE** *search-condition*]

value:
string | *integer*

参数

TYPE 子句　此子句指定同步方法。缺省值为 **tcpip**。也可以选择使用 **http** 或 **https**。

ADDRESS 子句　此子句指定网络参数，包括 MobiLink 同步服务器的位置。

有关网络参数的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

OPTION 子句　OPTION 子句允许您设置选项。每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。

有关选项的完整列表，请参阅 ""CREATE SYNCHRONIZATION USER 语句 [MobiLink]" 第 380 页 "。

用法

仅当通过从 Adaptive Server Anywhere 第 7 版参考数据库中抽取的方式创建 Adaptive Server Anywhere 远程数据库时，才使用同步模板和同步站点。

每个远程数据库从同步站点中创建并存储在参考数据库中。每个同步站点都基于单个同步模板，不过许多站点可以使用单个模板。

许多站点共用的信息应该存储在同步模板中，而不是在每个站点中复制。

抽取过程不仅创建远程数据库，还使用相应的同步模板和站点的属性创建同步定义。此信息必须包含该 MobiLink 站点在 MobiLink 系统内的唯一名称。另外，此信息还应指定如何联系 MobiLink 同步服务器以及同步远程数据库中的哪些数据。

此语句用于为各种类型的远程数据库创建模板。

权限

必须拥有 DBA 权限。要求可以对语句中涉及的所有表进行独占访问。

副作用

自动提交

另请参阅

"CREATE PUBLICATION 语句 " 第 357 页

"CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 375 页

"ALTER SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） " 第 267 页

"DROP SYNCHRONIZATION TEMPLATE 语句 [MobiLink] （不建议使用） " 第 461 页

"CREATE SYNCHRONIZATION DEFINITION 语句 [MobiLink] （不建议使用） " 第 369 页

"CREATE SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）" 第 372 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

- **Sybase**　Adaptive Server Anywhere 7.0 版。

示例

使用 TCP/IP 同步，将 *pet_id* 为 2 的 *Pets* 表以及 *People* 表中的 *person_id*、*fname* 和 *lname* 列从远程数据库上载到统一数据库：

```
CREATE SYNCHRONIZATION TEMPLATE mytemplate
   TYPE 'tcpip'
   ADDRESS 'host=localhost;port=2439;'
   OPTION memory='2m', dir='c:\db\logs'
   (table People( person_id, fname, lname ),
      table Pets WHERE pet_id=2);
```

CREATE SYNCHRONIZATION USER 语句 [MobiLink]

说明

此语句可用于在 Adaptive Server Anywhere 远程数据库中创建同步用户。

语法

CREATE SYNCHRONIZATION USER *ml_username*
[**TYPE** *sync-type*]
[**ADDRESS** *network-parameters*]
[**OPTION** *option*=*value*, …]

ml_username: *identifier*

sync-type: **tcpip** | **http** | **https** | **ActiveSync**

network-parameters: *string*

value: *string* | *integer*

参数

TYPE 子句 此子句指定同步中使用的通信协议。选项是 tcpip、 http、 https 和 ActiveSync。缺省协议为 tcpip。

如果指定的值存在冲突，则按照以下优先级方案使用协议：

- 通过 dbmlsync 扩展选项 CommunicationType 使用 -eu 选项指定的协议
- 通过 dbmlsync 扩展选项 CommunicationType 使用 -e 选项指定的协议
- 为预订指定的协议
- 为用户指定的协议
- 为发布指定的协议

ADDRESS 子句 此子句指定 *network-parameters*，格式为 *keyword*=*value*，以分号分隔。您提供什么设置取决于当前使用的通信协议 （TCP/IP、 HTTP、 HTTPS 或 ActiveSync）。

如果指定的值冲突，则按照下面的优先级方案使用网络参数：

- 通过 dbmlsync 扩展选项 CommunicationAddress 使用 -eu 选项指定的网络参数
- 通过 dbmlsync 扩展选项 CommunicationAddress 使用 -e 选项指定的网络参数
- 为预订指定的网络参数
- 为用户指定的网络参数
- 为预订指定的网络参数
- **TCP/IP 参数**　如果您指定 tcpip 协议，则还可以选择使用以下 *network-parameters*：
  - **client_port=nnnnn 或 client_port=nnnnn-mmmmm**　一组供通信使用的客户端端口。如果仅指定一个值，而端口范围的结束值比初始值大 100，那么端口的总数为 101。

    此选项对于防火墙内的客户端与防火墙外的 MobiLink 同步服务器进行的通信十分有用。
  - **host=hostname**　运行 MobiLink 同步服务器的机器的主机名或 IP 号。缺省值为 **localhost**。对于 Windows CE，缺省值是注册表文件夹 `Comm\Tcpip\Hosts\ppp_peer` 中 `ipaddr` 的值。它使 Windows CE 设备可以连接到 MobiLink 同步服务器，该服务器在 Windows CE 设备的底座连接到的台式计算机上执行。

    对于 Palm 计算平台，本地主机的缺省值是指设备。建议指定显式主机名或 IP 地址。
  - **port=portnumber**　套接字端口号。端口号必须是十进制数，并应与 MobiLink 同步服务器所监控端口相匹配。端口参数的缺省值为 2439，这是用于 MobiLink 同步服务器的 IANA 注册端口号。
  - **keep_alive=[0|1]**　在某些情况下，当基于 TCP/IP 的连接在同步过程中消失时， MobiLink 工作线程将不可用。这些工作线程正在等待来自 MobiLink 客户端的应答。如果所有工作线程都处于此状态，则 MobiLink 将无法处理同步。类似地，

如果连接消失， MobiLink 客户端会被阻塞。 MobiLink 同步期间所使用的基于 TCP/IP 的数据流在客户和服务器两端都接受一个参数来管理活动。缺省设置是 1 （开）。

- **network_name=name** 为使用 MobiLink 的自动拨号特性指定网络名。这样，您从 Pocket PC 2002 或 Windows 桌上型计算机进行连接时不必手动拨号。在与调度联合使用时，您的远程计算机可以在无人照管的情况下进行同步。在不使用调度的情况下使用此特性，您不必手动进行拨号连接即可运行 dbmlsync。该名称应当是您在 " 设置 " → " 连接 " → " 连接 "(Pocket PC) 或 " 网络和拨号连接 "(Windows) 中的下拉列表中指定的网络名。

  有关调度的更多信息，请参阅 "MobiLink 同步技术用户指南 > Adaptive Server Anywhere 客户端 > 调度同步 "。

- **network_connect_timeout=seconds** 在指定 network_name 时，还可以指定一个超时时间，经过该段时间后仍连接不上则拨号失败。此特性仅适用于 Pocket PC 2002。（在 Windows 系统中，可以通过配置连接配置文件来控制此特性。）缺省值是 120 秒。

- **network_leave_open={0|1}** 在指定 network_name 时，还可以指定在同步完成后让连接继续保持打开状态 (1)。缺省行为是关闭连接 (0)。

- **security=cipher(keyword=value;…)** 所有通过此连接进行的通信都将使用指定的密码组加密。 cipher 可以是 **ecc_tls** 或 **rsa_tls**。两者分别指 elliptic-curve 和 RSA 证书。为了向后兼容，还可以用 **certicom_tls** 指定 **ecc_tls**。

需要有可单独授权的选件
要使用 Certicom 技术，必须获取单独许可的 SQL Anywhere Studio 安全性选件，并且受出口法规约束。

有关安全性的更多信息，请参阅 "MobiLink 同步技术用户指南 > 传输层安全性 "。

支持以下安全关键字。

- **certificate_company** 如果指定此参数，则仅当服务器证书的组织字段与该值相匹配时， MobiLink 客户端才接受该证书。
- **certificate_unit** 如果指定此参数，则仅当服务器证书的组织单位字段与该值相匹配时， MobiLink 客户端才接受该证书。
- **certificate_name** 如果指定此参数，仅当服务器证书的公用名字段与该值相匹配时， MobiLink 客户端才接受该证书。
- **trusted_certificates** 通过 Certicom TLS 同步流发生同步时， MobiLink 同步服务器将它的证书连同签名实体的证书直到自签名根证书一起发送给客户端。

  客户端检查证书链是否有效，以及是否信任证书链中的根证书。此功能允许您指定受信任的根证书。缺省情况下， Sybase 证书是受信任的根证书。

- **HTTP 参数** 如果您指定 http 协议，则还可以指定以下 *network-parameters*：
  - **buffer_size=number** 固定内容长度消息的最大主体大小，以字节为单位。更改此选项将减小或增大为发送内容而分配的内存量。缺省值为 64 000，但在 UltraLite 和 Pocket PC 上是 1 000。
  - **client_port=nnnnn or client_port=nnnnn-mmmmm** 一组供通信使用的客户端端口。如果仅指定一个值，而端口范围的结束值比初始值大 100，那么端口的总数为 101。

    此选项对于防火墙内的客户端与防火墙外的 MobiLink 同步服务器进行的通信十分有用。
  - **host=hostname** 运行 MobiLink 同步服务器的机器的主机名或 IP 号。缺省值为 **localhost**。对于 Windows CE，缺省值是注册表文件夹 `Comm\Tcpip\Hosts\ppp_peer` 中 `ipaddr` 的值。它使 Windows CE 设备可以连接到 MobiLink 同步服务器，该服务器在 Windows CE 设备的底座连接到的台式计算机上执行。

对于 Palm 计算平台，本地主机的缺省值是指设备。建议指定显式主机名或 IP 地址。

- **keep_alive=[0|1]** 在某些情况下，当基于 TCP/IP 的连接在同步过程中消失时， MobiLink 工作线程将不可用。这些被阻塞的工作线程正在等待来自 MobiLink 客户端的应答。如果所有工作线程都处于此状态，则 MobiLink 将无法处理同步。类似地，如果连接消失， MobiLink 客户端会被阻塞。MobiLink 同步期间所使用的基于 TCP/IP 的数据流在客户和服务器两端都接受一个参数来管理活动。缺省设置是 1（开）。

- **network_name=name** 为使用 MobiLink 的自动拨号特性指定网络名。这样，您从 Pocket PC 2002 或 Windows 桌上型计算机进行连接时不必手动拨号。在与调度联合使用时，您的远程计算机可以在无人照管的情况下进行同步。在不使用调度的情况下使用此特性，您不必手动进行拨号连接即可运行 dbmlsync。该名称应当是您在 " 设置 " → " 连接 " → " 连接 "(Pocket PC) 或 " 网络和拨号连接 "(Windows) 中的下拉列表中指定的网络名。

  有关调度的更多信息，请参阅 "MobiLink 同步技术用户指南 > Adaptive Server Anywhere 客户端 > 调度同步 "。

- **network_connect_timeout=seconds** 在指定 network_name 时，还可以指定一个超时时间，经过该段时间后仍连接不上则拨号失败。此特性仅适用于 Pocket PC 2002。（在 Windows 系统中，可以通过配置连接配置文件来控制此特性。）缺省值是 120 秒。

- **network_leave_open={0|1}** 在指定 network_name 时，还可以指定在同步完成后让连接继续保持打开状态 (1)。缺省行为是关闭连接 (0)。

- **persistent={TRUE|FALSE}** TRUE 表示客户端对一个同步过程中的所有 HTTP 请求尝试使用同一 TCP/IP 连接。设置为 FALSE 则与中间代理兼容得较好。缺省值为 FALSE，但在 Palm 设备上是 TRUE。

*注意：*除在 Palm 设备上外，如果您直接与 MobiLink 连接，则只应将 persistent 设置为 TRUE。如果您是通过中间代理（例如代理或重定向程序）连接的，则持续连接可能会造成问题。

- **port=portnumber** 套接字端口号。端口号必须是十进制数，并应与 MobiLink 同步服务器所监控的端口相匹配。端口号的缺省值为 **80**，这是用于 HTTP 的 IANA 注册端口号。
- **proxy_host=proxy_hostname** 代理服务器的主机名或 IP 地址。缺省值为 **localhost**。
- **proxy_port=proxy_portnumber** 代理服务器的端口号。缺省值为 **80**。
- **security=cipher(keyword=value;…)** 所有通过此连接进行的通信都将使用指定的密码组加密。 cipher 可以是 **ecc_tls** 或 **rsa_tls**。两者分别指 elliptic-curve 和 RSA 证书。为了向后兼容，还可以用 **certicom_tls** 指定 **ecc_tls**。

需要有可单独授权的选件
要使用 Certicom 技术，必须获取单独许可的 SQL Anywhere Studio 安全性选件，并且受出口法规约束。

有关安全性的更多信息，请参阅 "MobiLink 同步技术用户指南 > 传输层安全性 "。

支持以下安全关键字。

- **certificate_company** 如果指定此参数，仅当服务器证书的组织字段与该值相匹配时， MobiLink 客户端才接受该证书。
- **certificate_unit** 如果指定此参数，仅当服务器证书的组织单位字段与该值相匹配时， MobiLink 客户端才接受该证书。
- **certificate_name** 如果指定此参数，仅当服务器证书的公用名字段与该值相匹配时， MobiLink 客户端才接受该证书。

- **trusted_certificates** 通过 Certicom TLS 同步流发生同步时， MobiLink 同步服务器将它的证书连同签名实体的证书直到自签名根证书一起发送给客户端。

  客户端检查证书链是否有效，以及是否信任证书链中的根证书。此功能允许您指定受信任的根证书。缺省情况下， Sybase 证书是受信任的根证书。

- **url_suffix= 后缀** 添加到每个 HTTP 请求第一行的 URL 的后缀。通过代理服务器进行同步时，可能必须加上后缀才能找到 MobiLink 同步服务器。缺省值为 **MobiLink**。

- **version=versionnumber** 指定要使用的 HTTP 版本的字符串。您可以选择 **1.0** 或 **1.1**。缺省值为 **1.1**。

- **HTTPS 参数** HTTPS 通信流使用 Certicom RSA 安全功能。

需要有可单独授权的选件
要使用 Certicom 技术，必须获取单独许可的 SQL Anywhere Studio 安全性选件，并且受出口法规约束。

有关安全性的更多信息，请参阅 "MobiLink 同步技术用户指南 > 传输层安全性 "。

如果您指定 HTTPS 协议，则还可以指定以下 *network-parameters*：

- **buffer_size=number** 固定内容长度消息的最大主体大小，以字节为单位。更改此选项将减小或增大为发送内容而分配的内存量。缺省值为 64 000，但在 UltraLite 和 Pocket PC 上是 1 000。

- **client_port=nnnnn or client_port=nnnnn-mmmmm** 一组供通信使用的客户端端口。如果仅指定一个值，而端口范围的结束值比初始值大 100，那么端口的总数为 101。

  此选项对于防火墙内的客户端与防火墙外的 MobiLink 同步服务器进行的通信十分有用。

- **host=hostname** 运行 MobiLink 同步服务器的机器的主机名或 IP 号。缺省值为 **localhost**。对于 Windows CE，缺省值是注册表文件夹 `Comm\Tcpip\Hosts\ppp_peer` 中 `ipaddr` 的

值。它使 Windows CE 设备可以连接到 MobiLink 同步服务器，该服务器在 Windows CE 设备的底座连接到的台式计算机上执行。

对于 Palm 计算平台，本地主机的缺省值是指设备。建议指定显式主机名或 IP 地址。

- **keep_alive=[0|1]** 在某些情况下，当基于 TCP/IP 的连接在同步过程中消失时， MobiLink 工作线程将不可用。这些工作线程正在等待来自 MobiLink 客户端的应答。如果所有工作线程都处于此状态，则 MobiLink 将无法处理同步。类似地，如果连接消失， MobiLink 客户端会被阻塞。 MobiLink 同步期间所使用的基于 TCP/IP 的数据流在客户和服务器两端都接受一个参数来管理活动。缺省设置是 1 （开）。

- **network_name=name** 为使用 MobiLink 的自动拨号特性指定网络名。这样，您从 Pocket PC 2002 或 Windows 桌上型计算机进行连接时不必手动拨号。在与调度联合使用时，您的远程计算机可以在无人照管的情况下进行同步。在不使用调度的情况下使用此特性，您不必手动进行拨号连接即可运行 dbmlsync。该名称应当是您在 " 设置 " → " 连接 " → " 连接 "(Pocket PC) 或 " 网络和拨号连接 "(Windows) 中的下拉列表中指定的网络名。

  有关调度的更多信息，请参阅 "MobiLink 同步技术用户指南 > Adaptive Server Anywhere 客户端 > 调度同步 "。

- **network_connect_timeout=seconds** 在指定 network_name 时，还可以指定一个超时时间，经过该段时间后仍连接不上则拨号失败。此特性仅适用于 Pocket PC 2002。（在 Windows 系统中，可以通过配置连接配置文件来控制此特性。）缺省值是 120 秒。

- **network_leave_open={0|1}** 在指定 network_name 时，还可以指定在同步完成后让连接继续保持打开状态 (1)。缺省行为是关闭连接 (0)。

- **persistent={TRUE|FALSE}** TRUE 表示客户端对一个同步过程中的所有 HTTPS 请求尝试使用同一 TCP/IP 连接。设置为 FALSE 则与中间代理兼容得较好。缺省值为 FALSE，但在 Palm 设备上是 TRUE。

*注意：*除在 Palm 设备上外，如果您直接与 MobiLink 连接，则只应将 persistent 设置为 TRUE。如果您是通过中间代理（例如代理或重定向程序）连接的，则持续连接可能会造成问题。

port=portnumber 套接字端口号。端口号必须是十进制数，并应与 MobiLink 同步服务器所监控的端口相匹配。端口参数的缺省值为 **443**，这是用于 HTTPS 的 IANA 注册端口号。

- **proxy_host=proxy_hostname** 代理服务器的主机名或 IP 地址。缺省值为 **localhost**。
- **proxy_port=proxy_portnumber** 代理服务器的端口号。缺省值为 **443**。
- **certificate_company** 如果指定此参数，仅当服务器证书的组织字段与该值相匹配时， MobiLink 客户端才接受该证书。
- **certificate_unit** 如果指定此参数，仅当服务器证书的组织单位字段与该值相匹配时，MobiLink 客户端才接受该证书。
- **certificate_name** 如果指定此参数，仅当服务器证书的公用名字段与该值相匹配时， MobiLink 客户端才接受该证书。
- **trusted_certificates** 通过 Certicom TLS 同步流发生同步时，MobiLink 同步服务器将它的证书连同签名实体的证书直到自签名根证书一起发送给客户端。

  客户端检查证书链是否有效，以及是否信任证书链中的根证书。此功能允许您指定受信任的根证书。缺省情况下，Sybase 证书是受信任的根证书。

  有关安全性的更多信息，请参阅 "MobiLink 同步技术用户指南 > 传输层安全性 "。
- **url_suffix= 后缀** 添加到每个 HTTPS 请求第一行的 URL 的后缀。通过代理服务器进行同步时，可能必须加上后缀才能找到 MobiLink 同步服务器。缺省值为 **MobiLink**。
- **version=versionnumber** 指定要使用的 HTTP 版本的字符串。您可以选择 **1.0** 或 **1.1**。缺省值为 **1.1**。

- **ActiveSync 参数** 在 ActiveSync 同步期间，将使用 ActiveSync 与驻留在台式计算机中的 ActiveSync 的 MobiLink 提供程序交换数据。 ActiveSync 参数描述了 ActiveSync 的 MobiLink 提供程序与 MobiLink 同步服务器之间的通信。

  ActiveSynche 的地址字符串采用以下格式：

  stream=*desktop-stream*;[*desktop-stream-params*]

  其中：

  - *desktop-stream* 是 ActiveSync 的 MobiLink 提供程序与 MobiLink 同步服务器之间使用的同步流。它可以是 **http**、**https** 或 **tcpip**。缺省设置为 **tcpip**。
  - *desktop-stream-params* 是上述列表中描述的 TCP/IP、 HTTP 或 HTTPS 参数。

    有关更多信息，请参阅 "MobiLink 同步技术用户指南 > 实用程序 > ActiveSync 提供程序的安装实用程序 "。

OPTION 子句 OPTION 子句可让您通过使用 *option=value* 以逗号分隔的列表形式设置选项。

每个选项的值不能包含字符 "**=**"、 "**,**" 或 "**;**"。数据库服务器接受您输入的任何选项而检查其是否有效。因此，如果选项的拼写不正确或输入的值无效，则直到您运行 `dbmlsync` 命令执行同步时，才会出现错误消息。

在各个预订或在 dbmlsync 命令行上可以覆盖为同步用户设置的选项。有关更多信息，请参阅 "MobiLink 同步技术用户指南 > MobiLink 同步服务器选项 > -e 扩展选项 "。

下表列出了可用的选项：

全称	简称	缺省	说明
ConflictRetries	cr	-1	因冲突导致下载失败后进行重试的次数。 -1 表示无限期重试。
DisablePolling	p	OFF	禁用事务日志扫描轮询。

全称	简称	缺省	说明
DownloadBufferSize	dbs	1 Mb（Windows CE 中为 32K）	用于缓冲下载流的缓冲大小。如果大小太小，容纳不下整个下载流，则 dbmlsync 在应用下载流之前，将使用一个临时文件存储它。有效设置为 *n*、 *n*K 和 *n*M，其中的 *n* 是零或正整数，单位是字节。如果设置为 0，则 dbmlsync 不缓冲下载流。如果设置大于 0 但小于 4K，则 dbmlsync 将发出警告信息，并自动使用 4K 内存来缓冲下载流。
ErrorLogSendLimit	el	32 kb	发生错误时，发送给 MobiLink 同步服务器的错误日志的最大大小。
FireTriggers	ft	ON	下载时触发触发器。
Increment	inc	无限制	大致按指定大小的大块发送上载流。一旦达到指定的大小，就在数据流中没有部分未完成事务的下一时刻发送大块。直到接收最后一个大块后， MobiLink 才向远程发送下载流。如果不指定增量大小，则在一次传送中发送整个上载流 （缺省设置）。
IgnoreHookErrors	eh	OFF	忽略处理同步挂接时发生的错误。
IgnoreScheduling	isc	OFF	忽略调度信息。
LockTables	lt	ON	设置为 OFF，将允许在同步过程中进行修改。
Memory	mem	1 Mb	用于汇集上载流的高速缓存的大小。如果需要更多的空间，一些内容会临时写入磁盘。
MobiLinkPwd	mp	NULL	设置 MobiLink 口令。
NewMobiLinkPwd	mn	NULL	设置新的 MobiLink 口令。

全称	简称	缺省	说明
OfflineDirectory	dir	NULL	包含脱机事务日志的目录的路径。
PollingPeriod	pp	1 分钟	事务日志扫描周期。
Schedule	sch	NULL	如果设置，则同步后保持备用模式并在指定的时间再次自动同步。下面是对如何设置调度的完整说明。
ScriptSiteName	sn	NULL	同步后执行的脚本的名称。
ScriptVersion	sv	缺省	通知 MobiLink 同步服务器使用指定模式版本的脚本。
SendColumnNames	scn	OFF	发送列名以便自动生成脚本。
SendTriggers	st	OFF	在上载时发送触发器动作。
StreamCompression	sc	MEDIUM	压缩上载流。选择 LOW、MEDIUM 或 HIGH。在更高级别的设置中，流更小但创建时间更长。
TableOrder	tor	NULL	指定上载流中表的顺序，以解决参照完整性问题。
Verbose	v	OFF	全部细节。这与设置命令行选项 -v+ 作用相同。
VerboseHooks	vs	OFF	记录与挂接脚本相关的消息。
VerboseMin	vm	OFF	记录最少量的信息。
VerboseOptions	vo	OFF	记录您指定的扩展选项的列表。
VerboseRowCounts	vn	OFF	记录已上载或下载的行数。
VerboseRowValues	vr	OFF	记录已上载或下载的行的值。
VerboseUpload	vu	OFF	记录关于上载流的信息。

有关如何使用这些选项优化同步的信息，请参阅 "MobiLink 同步技术用户指南 > Adaptive Server Anywhere 客户端 > 调优同步 "。

调度选项语法 调度选项语法与同步 SQL 语句和 dbmlsync 命令行中使用的语法相同。这两种情况下的语法如下。
*sch***=** { **EVERY**:*hhhh*:*mm* | *singleSchedule*,… }

其中：

hhhh : **00**… **9999**

singleSchedule : *day* @*hh*:*mm*[**AM** | **PM**] [**-***hh*:*mm*[**AM** | **PM**]]

day : **EVERYDAY** | **WEEKDAY** | **MONDAY** | **TUESDAY** | **WEDNESDAY** | **THURSDAY** | **FRIDAY** | **SATURDAY** | **SUNDAY** | *dayOfMonth*

hh : **00**… **24**

mm : **00**… **59**

dayOfMonth : **1**… **31**

如果存在关键字 EVERY，则立即发生同步，然后在指定的时间段后无限期重复同步。如果同步过程比指定周期长，则同步立即再次启动。重复周期为 0 与重复周期为 1 分钟的效果相同。

如果提供一个或多个单独的调度，则同步只在指定的日期和时间发生。

如果不指定 AM 或 PM，则假定为 24 小时时钟。 24:00 被解释为第二天的 00:00。

间隔可以指定为 *@hh:mm-hh:mm* （可以指定 AM 或 PM）。如果指定间隔，则只发生一次同步，在间隔时间内的随机时间开始。间隔提供了一个同步时间窗口，以便具有相同调度的多个远程数据库不会正好在同一时间同步，避免了同步服务器上发生拥塞。

间隔的结束时间总是被解释为位于开始时间后。如果时间间隔包含午夜，则它在第二天结束。如果 `dbmlsync` 在时间间隔的中间启动，则同步在结束时间前的随机时间发生。

EVERYDAY 是指一个星期内的全部七天，而 WEEKDAY 仅指从星期一到星期五。

周内某日是星期一、星期二等等。还可以使用星期周内某日的简写形式，例如 Mon （星期一）。如果您想让使用非英语运行的服务器识别周内某日的名称，则必须使用其英语全名，例如 Monday （星期一）。

如果在调度时间前一个同步仍未完成，则调度的同步在前一个同步完成时开始。

IgnoreScheduling 选项和 -is 命令行选项指示 dbmlsync 忽略调度，从而使同步立即执行。有关更多信息，请参阅 "MobiLink 同步技术用户指南 > MobiLink 同步服务器选项 > -is 选项 "。

有关调度的更多信息，请参阅 "MobiLink 同步技术用户指南 > Adaptive Server Anywhere 客户端 > 调度同步 "。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"ALTER SYNCHRONIZATION USER 语句 [MobiLink]" 第 270 页

"CREATE SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 375 页

"CREATE PUBLICATION 语句 " 第 357 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Open Client/Open Server 支持。

示例

下面的示例创建一个名为 SSinger 的用户，此用户通过 TCP/IP 协议并使用密码 Sam 与名为 mlserver.mycompany.com 的服务器计算机同步。在用户定义中使用密码是*不* 安全的。

```
CREATE SYNCHRONIZATION USER SSinger
TYPE http
ADDRESS 'host=mlserver.mycompany.com'
OPTION MobiLinkPwd='Sam'
```

下面创建一个称为 factory014 同步用户，该用户使 dbmlsync 处于停留状态，然后使用通过 Certicom 加密的 TCP/IP 在每 8 小时的间隔内随机进行同步。这种随机性有助于避免 MobiLink 服务器因同时进行多个同步操作而降低性能：

```
CREATE SYNCHRONIZATION USER factory014
TYPE tcpip
ADDRESS
'host=mycompany.manufacturing.mobilink1;security=certi
com_tls(certificate=mycompany_mobilink.crt;certificate
_password=thepassword)'
OPTION Schedule='EVERY:08:00'
```

下面创建一个名为 sales5322 同步用户，该用户将用于通过 HTTP 进行同步。在此示例中， MobiLink 同步服务器在公司防火墙内运行，同步请求是使用 Redirector （到一个 NSAPI Web 服务器的反向代理）重定向给该服务器的。

```
CREATE SYNCHRONIZATION USER sales5322
TYPE https
ADDRESS
'host=www.mycompany.com;port=80;url_suffix=mlredirect/
ml/'
```

CREATE TABLE 语句

说明　此语句用于在数据库中创建新表，也可以在远程服务器中创建表。

语法

CREATE [**GLOBAL TEMPORARY**] **TABLE** [*owner*.]*table-name*
({ *column-definition* | *table-constraint* | *pctfree* }, … **)**
[{ **IN** | **ON** } *dbspace-name*]
[**ON COMMIT** { **DELETE** | **PRESERVE** } **ROWS**]
[**AT** *location-string*]

column-definition :
column-name data-type [**NOT NULL**]
[**DEFAULT** *default-value*] [*column-constraint* …]

default-value :
special-value
| *string*
| *global variable*
| [**-**] *number*
| **(** *constant-expression* **)**
| *built-in-function***(** *constant-expression* **)**
| **AUTOINCREMENT**
| **CURRENT DATABASE**
| **CURRENT REMOTE USER**
| **CURRENT UTC TIMESTAMP**
| **GLOBAL AUTOINCREMENT** [**(** *partition-size* **)**]
| **NULL**
| **TIMESTAMP**
| **UTC TIMESTAMP**
| **LAST USER**

special-value:
CURRENT { **DATE** | **TIME** | **TIMESTAMP**
| **UTC TIMESTAMP** | **USER** | **PUBLISHER** }
| **USER**

column-constraint :
{ **UNIQUE**
| **PRIMARY KEY** [**CLUSTERED**]
| **REFERENCES** *table-name*
[**(** *column-name* **)**] [*actions*] [**CLUSTERED**]
}
| **CHECK (** *condition* **)**
| **COMPUTE (** *expression* **)**

table-constraint :
    UNIQUE (*column-name*, … **)**
  | **PRIMARY KEY** [**CLUSTERED**] **(** *column-name*, … **)**
  | **CHECK (** *condition* **)**
  | *foreign-key-constraint*

foreign-key-constraint :
  [**NOT NULL**] **FOREIGN KEY** [*role-name*] [**(***column-name*, … **)**]
  … **REFERENCES** *table-name* [**(***column-name*, … **)**] [**CLUSTERED**]
  … [*actions*] [**CHECK ON COMMIT**]

action :
  ON { **UPDATE** | **DELETE** }
  …{ **CASCADE** | **SET NULL** | **SET DEFAULT** | **RESTRICT** }

location-string :
    remote-server-name.[*db-name*].[*owner*].*object-name*
  | *remote-server-name*;[*db-name*];[*owner*];*object-name*

pctfree : **PCTFREE** *percent-free-space*

percent-free-space : integer

参数

PCTFREE 指定想为每个表页保留的可用空间百分比。如果数据更新时行大小增加，将占用可用空间。如果表页中没有可用空间，则该页上的行大小每次增加时，行都需要在多个表页中拆分，从而导致行碎片并可能引起性能下降。

percent-free-space 值是 0 到 100 之间的整数。前者指定每页上没有剩余的可用空间：每页都完全填满。高值会将每行单独插入到页中。如果没有设置 PCTFREE，则每页保留 200 字节。

PCTFREE 的值存储在 SYSATTRIBUTE 系统表中。

有关更多信息，请参阅 ""SYSATTRIBUTE 系统表 " 第 667 页 "。

IN 子句 IN 子句指定用来创建表的 dbspace。如果表是 GLOBAL TEMPORARY 表，则忽略 IN 子句。

有关 dbspace 的更多信息，请参阅 ""CREATE DBSPACE 语句 " 第 319 页 "。

ON COMMIT 子句 ON COMMIT 子句只能用于临时表。缺省情况下，临时表的行将在 COMMIT （提交）时被删除。

AT 子句 在 *location-string* 指定的不同服务器中创建远程表，还在当前数据库中创建映射到此远程表的代理表。 AT 子句支持分号 (;) 作为 *location-string* 中的字段分隔符。如果没有分号，则使用句号作为字段分隔符。这允许在数据库和所有者字段中使用文件名和扩展名。

例如，下面的语句将表 `a1` 映射到 MS Access 文件 `mydbfile.mdb`：

```
CREATE TABLE a1
AT 'access;d:\mydbfile.mdb;;a1'
```

有关远程服务器的信息，请参阅 ""CREATE SERVER 语句 " 第 364 页 "。有关代理表的信息，请参阅 ""CREATE EXISTING TABLE 语句 " 第 332 页 " 和 "ASA SQL 用户指南 > 访问远程数据 > 指定代理表位置 "。

忽略远程表的外键定义。引用远程表的本地表的外键定义也被忽略。如果远程服务器支持主键，则主键定义发送到此服务器。

忽略远程表的 COMPUTE 子句。

column-definition 定义表中的列。以下是部分列定义。

- **column-name** 列名是标识符。同一表中的两列不能同名。有关更多信息，请参阅 "" 标识符 " 第 8 页 "。
- **data-type** 有关数据类型的信息，请参阅 ""SQL 数据类型 " 第 59 页 "。
- **NOT NULL** 如果指定 NOT NULL，或者列具有唯一约束或主键约束，则任何行中的列不能为 NULL。
- **DEFAULT** 有关 *special-value* 的更多信息，请参阅 "" 特殊值 " 第 37 页 "。

  如果指定 DEFAULT，则它用作未指定列值的任何 INSERT 语句中的列值。如果不指定 DEFAULT，则等效于 DEFAULT NULL。

  一些缺省值需要详细说明：

  - **AUTOINCREMENT** 使用 AUTOINCREMENT 时，列必须是整型数据类型之一或是精确的数值类型。

插入到表中时，如果没有指定 AUTOINCREMENT 列的值，则生成一个比列中的任何其它值都大的唯一值。如果 INSERT 指定了列值，则使用该值；如果指定的值大于当前列中的最大值，该值将用作后续插入的起始值。

删除行不会减少 AUTOINCREMENT 计数器的值。由于删除行而产生的间隙只能由使用插入时的显式赋值填充。显式插入小于该列最大值的行后，后面没有显式赋值的行仍自动增加比前一个最大值大 1 的值。

用于每列的下一个值以整型存储。使用大于 (2^{31} -1) 的值可能会导致绕回到不正确的值，这种情况下不要使用 AUTOINCREMENT。

通过检查 @@identity 全局变量，可以找到最近插入的列值。

标识列是使用 AUTOINCREMENT 缺省值的替代方法，与 Transact-SQL 兼容。在 Adaptive Server Anywhere 中，标识列作为 AUTOINCREMENT 缺省值实现。有关信息，请参阅 "ASA SQL 用户指南 > Transact-SQL 兼容性 > 特殊 IDENTITY 列 "。

- **GLOBAL AUTOINCREMENT** 当 SQL Remote 复制或 MobiLink 同步环境中将使用多个数据库时，使用该缺省值。

  除了对域进行分区外，此缺省值类似于 AUTOINCREMENT。每个分区都包含相同数量的值。您为数据库的每个副本指派一个唯一全局数据库标识号。Adaptive Server Anywhere 只从使用数据库编号唯一标识的分区中提供数据库中的缺省值。

  可以在紧跟在 AUTOINCREMENT 关键字后面的圆括号内指定分区大小。分区大小可以为任意正整数，但分区大小的选择一般需要保证任何一个分区内的编号资源不被用尽，或极少用尽。

  如果列为 BIGINT 或 UNSIGNED BIGINT 类型，则缺省分区大小为 2^{32} = 4294967296；对于所有其它类型的列，缺省分区大小为 2^{16} = 65536。这些缺省值可能不合适，尤其当列不是 INT 或 BIGINT 类型时，因此最好显式指定分区大小。

使用此缺省值时，每个数据库中的公共选项 **Global_database_id** 的值必须设置为唯一的非负整数。该值唯一标识数据库，并指示从哪个分区分配缺省值。允许的值范围介于 *n p* + 1 到 (*n* + 1) *p* 之间，其中 *n* 是公共选项 **Global_database_id** 的值， *p* 是分区大小。例如，如果将分区大小定义为 1000 并将 **Global_database_id** 设置为 3，则范围介于 3001 到 4000 之间。

如果前一个值小于 (*n* + 1) *p*，则下一个缺省值将比列中的前一个最大值大 1。如果列不包含任何值，则第一个缺省值为 *n p* + 1。不在当前分区内的列的值不影响缺省列值，即小于 *pn* + 1 或大于 *p*(*n* + 1) 的列值不影响缺省列值。这些范围之外的值可能是在 MobiLink 同步过程中从其他数据库复制来的。

因为公共选项 **Global_database_id** 不能设置为负值，所以选择的值始终为正数。标识号的最大值只受列的数据类型及分区大小的限制。

如果公共选项 **Global_database_id** 设置为缺省值 2147483647，则在列中插入空值。如果不允许空值，试图插入行时将出错。例如，如果该列包含在表的主键中，则会出现这种情况。

当分区内的可用值用完时，也会产生空的缺省值。这种情况下，应将新的 **Global_database_id** 值分配给数据库，以允许从另一分区中选择缺省值。如果该列不允许使用空值，则插入空值的尝试将导致错误。要检测是不是未使用的值已经过少并处理此问题，您可以创建一个 **GlobalAutoincrement** 类型的事件。

在用版本 6 或更早版本的软件创建的数据库中，即使它们已升级，也不能使用 DEFAULT GLOBAL AUTOINCREMENT。

- **常量表达式**　在 DEFAULT 子句中允许使用不参考数据库对象的常量表达式，因此可以使用 GETDATE 或 DATEADD 这类函数。如果表达式不是函数或简单值，必须用圆括号括起来。

- **TIMESTAMP** 提供一种指示表中每行的上次修改时间的方法。当用 DEFAULT TIMESTAMP 声明列时，会提供一个缺省的插入值，每当更新行时，该值都用当前日期和时间更新。

  要提供插入时的缺省值，而在每次更新时不改变，请使用 DEFAULT CURRENT TIMESTAMP 而不是 DEFAULT TIMESTAMP。

  有关时间戳列的更多信息，请参阅 "ASA SQL 用户指南 > Transact-SQL 兼容性 > 特殊的 Transact-SQL timestamp 列和数据类型 "。

  用 DEFAULT TIMESTAMP 声明的列包含唯一值，以便应用程序能检测到同一行几乎同时发生的更新。如果当前时间戳的值与上一个值相同，它将增长 DEFAULT_TIMESTAMP_INCREMENT 选项的值。

  有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > DEFAULT_TIMESTAMP_INCREMENT 选项 [database]"。

  在 Adaptive Server Anywhere 中，您可以根据 DEFAULT_TIMESTAMP_INCREMENT 选项自动截断时间戳的值。这对于同其它记录精度较低的时间戳值的数据库软件保持兼容很有用。

  有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > TRUNCATE_TIMESTAMP_VALUES 选项 [database]"。

  全局变量 **@@dbts** 一个 TIMESTAMP 值，该值表示上次使用 DEFAULT TIMESTAMP 为列生成的值。有关更多信息，请参阅 "" 全局变量 " 第 45 页 "。

- **string** 有关更多信息，请参阅 "" 字符串 " 第 10 页 "。
- **global-variable** 有关更多信息，请参阅 "" 全局变量 " 第 45 页 "。

- **column-constraint** 列约束限制列能够保存的值。

table-constraint 表约束限制表的一列或多列能够保存的值。

约束 列约束和表约束有助于确保数据库的数据完整性。如果语句会导致违反约束，则该语句的执行不会完成。该语句在检测到错误前所做的任何更改都被撤消并报告错误。列约束是相应表约束的简化。

- 例如，下列语句是等效的：

```
CREATE TABLE Product (
   product_num INTEGER UNIQUE
)
CREATE TABLE Product (
   product_num INTEGER,
   UNIQUE ( product_num )
)
```

正常情况下使用列约束，除非约束引用表中的多列。这些情况下，必须使用表约束。

约束包括以下这些：

- **CHECK** 允许对任意条件进行校验。例如，检查约束可用于确保名为 **Sex** 的列只包含值 M 或 F。

  表中的任何行都不能违反约束。如果 INSERT 或 UPDATE 语句会导致行违反约束，则不允许该操作并且撤消语句的执行结果。

  仅当约束条件计算为 FALSE 时才拒绝更改，如果约束条件计算为 TRUE 或 UNKNOWN，则允许更改。

  有关 TRUE、 FALSE 和 UNKNOWN 条件的更多信息，请参阅 ""NULL 值 " 第 55 页 " 和 "" 搜索条件 " 第 26 页 "。

- **COMPUTE** COMPUTE 约束只是列约束。使用 COMPUTE 约束创建列时，此列在任何行中的值均为提供的表达式的值。对于应用程序而言，使用此约束创建的列为只读列：计算表达式时，数据库服务器将更改该值。

  任何试图更改计算列的值的 UPDATE 语句都会触发与该列关联的所有触发器。

  使用 Java 类数据类型设计数据库时， Compute 约束尤其有用。有关更多信息，请参阅 "ASA 编程指南 > 在数据库中使用 Java > 使用具有 Java 类的计算列 "。

- **UNIQUE** 标识唯一标识表中各行的一列或多列。表中任何两行的值在所有指定的列中不能相同。表可以有多个唯一约束。

  唯一约束与唯一索引是有区别的。唯一索引的列可以为 NULL，而唯一约束的列不能为 NULL。外键可以引用主键或者具有唯一约束的列，但不能引用具有唯一索引的列，因为唯一索引可以包括多个 NULL 实例。

  有关唯一索引的信息，请参阅 ""CREATE INDEX 语句 " 第 341 页 "。

- **PRIMARY KEY** 它与唯一约束相同，但表只能有一个主键约束。主键通常标识行的最佳标识符。例如，客户号可能是 customer 表的主键。

  主键包括的列不能为 NULL。表的每一行均有一个唯一主键值。表只能有一个 PRIMARY KEY。

  主键中的列顺序是在表中创建列时的顺序，而不是创建主键时列出它们的顺序。

  有关 CLUSTERED 选项和聚簇索引的更多信息，请参阅 "ASA SQL 用户指南 > 使用数据库对象 > 使用聚簇索引 "。

- **外键** 通过使用 REFERENCES 列约束 （仅限于单列）或 FOREIGN KEY 表约束可以实现外键约束。它限制一组列的值与主键值匹配，或者在少数情况下，与另一个表 （主表）的唯一约束值匹配。例如，外键约束可用于确保 invoice 表中的客户号与 customer 表中的客户号相对应。

  如果在 REFERENCES 列约束中指定 *column name*，则列必须在主表中，必须受唯一约束或主键约束的制约，而且此约束必须仅包含那一列。如果不指定 *column-name*，则外键引用主表的主键。

  有关 CLUSTERED 选项和聚簇索引的更多信息，请参阅 "ASA SQL 用户指南 > 使用数据库对象 > 使用聚簇索引 "。

  如果不显式定义外键列，则使用与主表中的相应列相同的数据类型创建外键列。这些自动创建的列不能属于外表的主键。因此，必须显式创建在同一表的外键和主键中都使用的列。

如果指定外键列名，则还必须指定主键列名，列名按照列表中的位置成对出现。如果在 FOREIGN KEY 表约束中未指定主表列名，则使用主键列。如果未指定外键列名，则外键列的列名与主表的列名相同。

如果多列外键中至少有一个值为 NULL，则该键的其它列中可以保存的值没有限制。

临时表不能有引用基表的外键，而基表不能有引用临时表的外键。

- **NOT NULL** 在外键列中禁止 NULL。外键中的 NULL 表示主表中没有与外表中的该行相对应的行。
- **role-name** 角色名是外键的名称。角色名的主要作用是区分同一表的两个外键。如果不指定角色名，则按如下方式分配角色名：

1. 如果没有与表名同名的角色名外键，则将表名指派为角色名。
2. 如果表名已被使用，则角色名为表名加上表的唯一 3 位零填充数字。

- **action** 参照完整性操作定义为维护数据库中的外键关系而采取的操作。每当更改主键值或从数据库表中删除主键值时，其它表中的相应外键值也应该以某种方式修改。可以指定 ON UPDATE 子句或 ON DELETE 子句或者两者，然后执行以下操作之一：
  - **CASCADE** 使用 ON UPDATE 时，更新相应的外键以匹配新主键值。使用 ON DELETE 时，从外键表中删除与已删除的主键匹配的行。
  - **SET NULL** 将所有与已更新或已删除的主键对应的外键值设置为 NULL。
  - **SET DEFAULT** 将与已更新或已删除的主键值匹配的外键值设置为每个外键列的 DEFAULT 子句中指定的值。
  - **RESTRICT** 如果在数据库的其它地方有相应的外键时试图更新或删除主键值，将发生错误。 RESTRICT 是缺省操作。

- **CHECK ON COMMIT** CHECK ON COMMIT 子句覆盖 WAIT_FOR_COMMIT 数据库选项，使数据库服务器在检查外键的 RESTRICT 操作前等待 COMMIT。 CHECK ON COMMIT 子句不推迟 CASCADE、 SET NULL 或 SET DEFAULT 操作。

  如果使用 CHECK ON COMMIT 但不指定任何操作，则表示针对 UPDATE 和 DELETE 使用 RESTRICT 操作。

用法

CREATE TABLE 语句创建新表。通过指定所有者名称，可为其他用户创建表。如果指定 GLOBAL TEMPORARY，则表为临时表。否则，表为基表。

与基表的定义一样，临时表的定义也存在于数据库中并一直保留在数据库中，直到被 DROP TABLE 语句显式删除。临时表中的行仅对插入行的连接是可见的。来自相同或不同应用程序的多个连接可同时使用同一临时表，而每个连接只能看到自己的行。连接结束时删除用于连接的临时表中的行。

权限

必须有 RESOURCE 权限。

必须有 DBA 权限才能为其他用户创建表。

Windows CE 不支持用于创建代理表的 AT 子句。

副作用

自动提交。

另请参阅

"ALTER TABLE 语句 " 第 272 页

"CREATE DBSPACE 语句 " 第 319 页

"CREATE EXISTING TABLE 语句 " 第 332 页

"DECLARE LOCAL TEMPORARY TABLE 语句 " 第 433 页

"DROP 语句 " 第 445 页

" 特殊值 " 第 37 页

"SQL 数据类型 " 第 59 页

"ASA SQL 用户指南 > 使用数据库对象 > 创建表 "

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。

  以下为供应商扩展：

  - { **IN** | **ON** } *dbspace-name* 子句。
  - **ON COMMIT** 子句
  - 一些缺省值。
- **Sybase** 受 Adaptive Server Enterprise 支持，有一些差异。
  - **临时表** 通过在 CREATE TABLE 语句中的表名前加上 " # " 符号，可以创建临时表。在 Adaptive Server Anywhere 中，这些是已声明的临时表，它们仅在当前连接中可用。有关信息，请参阅 "DECLARE LOCAL TEMPORARY TABLE 语句 " 第 433 页。
  - **物理放置** 在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中，表的物理放置方式不同。在 Adaptive Server Enterprise 中支持 **ON** *segment-name* 子句，在 Adaptive Server Anywhere 中也支持它，但 *segment-name* 引用 dbspace 名。
  - **约束** Adaptive Server Anywhere 不支持命名约束或命名缺省值，但确实支持允许在数据类型定义中封装约束和缺省值定义的域。它也支持 CREATE TABLE 语句中的显式缺省值和 CHECK 条件。
  - **NULL 缺省值** 缺省情况下， Adaptive Server Enterprise 中的列缺省为 NOT NULL，而 Adaptive Server Anywhere 中的缺省设置为 NULL。使用 ALLOW_NULLS_BY_DEFAULT 数据库选项可控制此设置。应显式指定 NULL 或 NOT NULL，以便可在 Adaptive Server Anywhere 与 Adaptive Server Enterprise 之间传递数据定义语句。

    有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > ALLOW_NULLS_BY_DEFAULT 选项 [compatibility]"。

示例

下面的示例为图书馆数据库创建用于保存图书信息的表。

```
CREATE TABLE library_books (
-- NOT NULL is assumed for primary key columns
isbn CHAR(20)      PRIMARY KEY,
copyright_date     DATE,
title              CHAR(100),
author             CHAR(50),
-- column(s) corresponding to primary key of room
-- are created automatically
FOREIGN KEY location REFERENCES room
)
```

下面的示例为图书馆数据库创建用于保存借出图书信息的表。*date_borrowed* 的缺省值指示在创建该条目当天借出的图书。*date_returned* 列在图书返回前一直为 NULL。

```
CREATE TABLE borrowed_book (
date_borrowed DATE NOT NULL DEFAULT CURRENT DATE,
date_returned       DATE,
book                CHAR(20)
                    REFERENCES library_books (isbn),
-- The check condition is UNKNOWN until
-- the book is returned, which is allowed
CHECK( date_returned >= date_borrowed )
)
```

下面的示例在销售数据库中创建保存订单和订单项信息的表。

```
CREATE TABLE Orders (
   order_num INTEGER NOT NULL PRIMARY KEY,
   date_ordered DATE,
   name CHAR(80)
);
CREATE TABLE Order_item (
   order_num        INTEGER NOT NULL,
   item_num         SMALLINT NOT NULL,
   PRIMARY KEY (order_num, item_num),
   -- When an order is deleted, delete all of its
   -- items.
   FOREIGN KEY (order_num)
   REFERENCES Orders (order_num)
   ON DELETE CASCADE
)
```

下面的示例在远程服务器 *SERVER_A* 上创建名为 *t1* 的表，并创建映射到此远程表的名为 *t1* 的代理表。

```
CREATE TABLE t1
( a  INT,
  b  CHAR(10))
AT 'SERVER_A.db1.joe.t1'
```

CREATE TRIGGER 语句

说明

此语句用于在表中创建新触发器。

语法

CREATE TRIGGER *trigger-name trigger-time trigger-event* [, *trigger-event,…*]
[**ORDER** *integer*] **ON** *table-name*
[**REFERENCING** [**OLD AS** *old-name*]
[**NEW AS** *new-name*]]
[**REMOTE AS** *remote-name*]]
[**FOR EACH** { **ROW** | **STATEMENT** }]
[**WHEN (** *search-condition* **)**]
compound-statement

trigger-time : **BEFORE** | **AFTER** | **RESOLVE**

trigger-event :
DELETE | **INSERT** | **UPDATE** | **UPDATE OF** *column-list*

参数

触发器事件 触发器可以由以下一个或多个事件触发：

- **DELETE** 每当删除关联表中的行时激活。
- **INSERT** 每当有新行插入到与触发器关联的表中时激活。
- **UPDATE** 每当更新关联表中的行时激活。
- **UPDATE OF column-list** 每当更新关联表中的行并修改 *column-list* 中的列时激活。

您可以为各个需要处理的事件分别编写不同的触发器，或者，如果您有一些共享的操作和一些依赖事件的操作，则可以创建一个用于所有事件的触发器，并可使用 IF 语句区分正在发生的操作。

有关更多信息，请参阅 "IF 语句 " 第 510 页。

trigger-time 可以定义行级触发器在插入、更新或删除之前 (BEFORE) 或之后 (AFTER) 执行。语句级触发器在语句后 (AFTER) 执行。RESOLVE 触发器时间用于 SQL Remote：它仅在行级 UPDATE 或 UPDATE OF column-lists 之前触发。

BEFORE UPDATE 在行上发生 UPDATE （更新）的任何时候触发，不论新值是否不同于旧值。 AFTER UPDATE 触发器仅在新值不同于旧值时才触发。

FOR EACH 子句　FOR EACH ROW 子句用于声明触发器为行级触发器。要将触发器声明为语句级触发器，可以使用 FOR EACH STATEMENT 子句或省略 FOR EACH 子句。为清楚起见，建议声明语句级触发器时，输入 FOR EACH STATEMENT 子句。

ORDER 子句　同时触发（之前、之后或解析）的同类型触发器（插入、更新或删除）可以使用 ORDER 子句确定触发器的触发顺序。

REFERENCING 子句　REFERENCING OLD 和 REFERENCING NEW 子句允许引用已删除和插入的行。在此子句中， UPDATE 按删除后加插入处理。

REFERENCING REMOTE 子句用于 SQL Remote。它允许引用 UPDATE 语句的 VERIFY 子句中的值。它仅能用于 RESOLVE UPDATE 或 RESOLVE UPDATE OF column-list 触发器。

REFERENCING OLD 和 REFERENCING NEW 的含义不同，具体取决于触发器是行级还是语句级触发器。对于行级触发器， REFERENCING OLD 子句允许引用更新或删除前行中的值， REFERENCING NEW 子句允许引用已插入或更新的值。在 BEFORE 和 AFTER 触发器中可以引用 OLD 和 NEW 行。 REFERENCING NEW 允许在插入或更新操作发生之前在 BEFORE 触发器中修改新行。

对于语句级触发器， REFERENCING OLD 和 REFERENCING NEW 子句引用保存旧行值和新行值的声明临时表。这些表的缺省名为 **deleted** 和 **inserted**。

WHEN 子句　触发器仅对搜索条件计算为真值的行触发。 WHEN 子句仅能用于行级触发器。

用法

CREATE TRIGGER 语句创建与数据库中的表关联的触发器，并在数据库中存储触发器。

触发器声明为行级触发器（此情况下，它在每行修改之前或之后执行）或语句级触发器（此情况下，它在整个触发器语句完成后执行）。

权限

必须有 RESOURCE 权限和表的 ALTER 权限，或者必须是表的所有者，或者有 DBA 权限。 CREATE TRIGGER 在表上放置表锁，因此要求排它使用表。

副作用

自动提交。

另请参阅

"BEGIN 语句 " 第 288 页

"CREATE PROCEDURE 语句 " 第 347 页

"CREATE TRIGGER 语句 [T-SQL]" 第 415 页

"DROP 语句 " 第 445 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

标准和兼容性

- **SQL/92** 持久存储模块特性。一些子句是供应商扩展。
- **SQL/99** 持久存储模块特性。一些子句是供应商扩展。
- **Sybase** 此语法不同于 Adaptive Server Enterprise 支持的语法。

示例

第一个示例创建行级触发器。当任命一个新的部门经理时，更新该部门中雇员的 **manager_id** 列。

```
CREATE TRIGGER tr_manager
BEFORE UPDATE OF dept_head_id
ON department
REFERENCING OLD AS old_dept NEW AS new_dept
FOR EACH ROW
BEGIN
   UPDATE employee
   SET employee.manager_id=new_dept.dept_head_id
   WHERE employee.dept_id=old_dept.dept_id
END
```

下一个示例更复杂，涉及语句级触发器。首先，按如下所示创建表：

```
CREATE TABLE "DBA"."t0"
(
   "id"     integer NOT NULL,
   "times"  timestamp NULL DEFAULT current timestamp,
   "remarks"       text NULL,
   PRIMARY KEY ("id")
)
```

然后，创建此表的语句级触发器：

```
create trigger DBA."insert-st" after insert order 4 on
DBA.t0
referencing new as new_name
for each statement
begin
  declare @id1 integer;
  declare @times1 timestamp;
  declare @remarks1 long varchar;
  declare @err_notfound exception for sqlstate value
'02000';
//declare a cursor for table new_name
  declare new1 cursor for
   select id,times,remarks from
      new_name;
  open new1;
 //Open the cursor, and get the value
  LoopGetRow:
  loop
      fetch next new1
   into @id1, @times1,@remarks1;
      if sqlstate = @err_notfound then
   leave LoopGetRow
      end if;
      //print the value or for other use
      Print (@remarks1);
  end loop LoopGetRow;
  close new1
end
```

CREATE TRIGGER 语句 [SQL Remote]

说明

此语句用于在数据库中创建新触发器。它是一种形式的触发器，专门设计用于 SQL Remote。

语法

CREATE TRIGGER *trigger-name trigger-time*
trigger-event, …
[**ORDER** *integer*] **ON** *table-name*
[**REFERENCING** [**OLD AS** *old-name*]
[**NEW AS** *new-name*]]
[**REMOTE AS** *remote-name*]]
[**FOR EACH** { **ROW** | **STATEMENT** }]
[**WHEN (** *search-condition* **)**]
[**IF UPDATE (** *column-name* **) THEN**
[{ **AND** | **OR** } **UPDATE (** *column-name* **)**] …]
compound-statement
[**ELSEIF UPDATE (** *column-name* **) THEN**
[{ **AND** | **OR** } **UPDATE (** *column-name* **)**] …
compound-statement
END IF]]

trigger-time:
BEFORE | **AFTER** | **RESOLVE**

trigger-event:
DELETE | **INSERT** | **UPDATE**
| **UPDATE OF** *column-name* [**,** *column-name***,** …]

参数

trigger-time 可以定义行级触发器在插入、更新或删除之前 (BEFORE) 或之后 (AFTER) 执行。语句级触发器在语句后 (AFTER) 执行。RESOLVE 触发器时间用于 SQL Remote：它仅在行级 UPDATE 或 UPDATE OF column-lists 之前触发。

BEFORE UPDATE 在行上发生 UPDATE （更新）的任何时候触发，不论新值是否不同于旧值。 AFTER UPDATE 触发器仅在新值不同于旧值时才触发。

触发器事件 触发器可以由以下一个或多个事件触发：

- **DELETE** 每当删除关联表中的行时激活。
- **INSERT** 每当有新行插入到与触发器关联的表中时激活。

- **UPDATE** 每当更新关联表中的行时激活。
- **UPDATE OF column-list** 每当更新关联表中的行并修改 *column-list* 中的列时激活。

用法

任何地方。

权限

必须有 RESOURCE 权限和表的 ALTER 权限，或者必须有 DBA 权限。CREATE TRIGGER 在表上放置表锁，因此要求排它使用表。

副作用

自动提交。

另请参阅

"UPDATE 语句 " 第 642 页

说明

CREATE TRIGGER 语句创建与数据库中的表关联的触发器，并且在数据库中存储触发器。

BEFORE UPDATE 触发器在行上发生更新的任何时候触发，不论新值是否不同于旧值。 AFTER UPDATE 触发器仅当新值不同于旧值时才触发。

行级和语句级触发器

触发器声明为行级触发器 （此情况下，它在每行修改之前或之后执行）或语句级触发器 （此情况下，它在整个触发器语句完成后执行）。

可以定义行级触发器在插入、更新或删除之前 (BEFORE) 或之后 (AFTER) 执行。语句级触发器在语句后 (AFTER) 执行。 RESOLVE 触发器事件用于 SQL Remote：它仅在行级 UPDATE 或 UPDATE OF column-lists 之前触发。

要将触发器声明为行级触发器，请使用 FOR EACH ROW 子句。要将触发器声明为语句级触发器，可以使用 FOR EACH STATEMENT 子句或省略 FOR EACH 子句。为清楚起见，建议声明语句级触发器时，输入 FOR EACH STATEMENT 子句。

触发顺序

同时触发 （之前、之后或解析）的同类型触发器可以使用 ORDER 子句确定触发器的触发顺序。

引用已删除和插入的值

REFERENCING OLD 和 REFERENCING NEW 子句允许引用已删除和插入的行。在此子句中， UPDATE 按删除后加插入处理。

REFERENCING REMOTE 子句用于 SQL Remote。它允许引用 UPDATE 语句的 VERIFY 子句中的值。它仅能用于 RESOLVE UPDATE 或 RESOLVE UPDATE OF column-list 触发器。

REFERENCING OLD 和 REFERENCING NEW 的含义不同，具体取决于触发器是行级还是语句级触发器。对于行级触发器， REFERENCING OLD 子句允许引用更新或删除前行中的值， REFERENCING NEW 子句允许引用已插入或更新的值。在 BEFORE 和 AFTER 触发器中可以引用 OLD 和 NEW 行。 REFERENCING NEW 允许在插入或更新操作发生之前在 BEFORE 触发器中修改新行。

对于语句级触发器， REFERENCING OLD 和 REFERENCING NEW 子句引用保存旧行值和新行值的声明临时表。这些表的缺省名为 **deleted** 和 **inserted**。

WHEN 子句使触发器仅对搜索条件计算为真值的行触发。

用同一值更新值

BEFORE UPDATE 在行上发生 UPDATE （更新）的任何时候触发，不论新值是否不同于旧值。 AFTER UPDATE 触发器仅在新值不同于旧值时才触发。

示例

- 当任命一个新的部门经理时，更新该部门中雇员的 **manager_id** 列。

```
CREATE TRIGGER
tr_manager BEFORE UPDATE OF dept_head_id ON
department
REFERENCING OLD AS old_dept
NEW AS new_dept
FOR EACH ROW
BEGIN
UPDATE employee
SET employee.manager_id=new_dept.dept_head_id
WHERE employee.dept_id=old_dept.dept_id
END
```

CREATE TRIGGER 语句 [T-SQL]

说明

此语句用于采用与 Adaptive Server Enterprise 兼容的方式，在数据库中创建新触发器。

语法 1

CREATE TRIGGER [*owner*.]*trigger_name*
ON [*owner*.]*table_name*
FOR { **INSERT**, **UPDATE**, **DELETE** }
AS *statement-list*

语法 2

CREATE TRIGGER [*owner*.]*trigger_name*
ON [*owner*.]*table_name*
FOR {**INSERT**, **UPDATE**}
AS
[**IF UPDATE (** *column_name* **)**
[{ **AND** | **OR** } **UPDATE (** *column_name* **)**] …]
statement-list
[**IF UPDATE (** *column_name* **)**
[{ **AND** | **OR**} **UPDATE (** *column_name* **)**] …]
statement-list

用法

删除或插入的行保存在两个临时表中。在触发器的 Transact-SQL 表中，与在 Adaptive Server Enterprise 中一样，可以使用表名 **deleted** 和 **inserted** 来访问它们。在 Watcom-SQL CREATE TRIGGER 语句中，可以使用 REFERENCING 子句来访问它们。

触发器名在数据库中必须是唯一的。

Transact-SQL 触发器在触发器语句后 (AFTER) 执行。

权限

必须有 RESOURCE 权限和表的 ALTER 权限，或者必须有 DBA 权限。

CREATE TRIGGER 锁定表中的所有行，因此要求排它使用表。

副作用

自动提交。

另请参阅

"CREATE TRIGGER 语句 " 第 408 页

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/92** Transact-SQL 扩展。

- **Sybase** Anywhere 支持 Adaptive Server Enterprise 语法的子集。

CREATE VARIABLE 语句

说明

此语句用于创建 SQL 变量。

语法

CREATE VARIABLE *identifier data-type*

用法

CREATE VARIABLE 语句创建指定数据类型的新变量。此变量包含 NULL 值，直到它被 SET VARIABLE 语句赋予了不同的值。

变量可以在 SQL 表达式中允许使用列名的任何地方使用。如果列名和变量名相同，则使用变量值。

变量属于当前连接，当从数据库断开连接或使用 DROP VARIABLE 语句时，它就会消失。变量对其它连接不可见。变量不受 COMMIT 或 ROLLBACK 语句影响。

在为嵌入式 SQL 程序中的 INSERT 或 UPDATE 语句创建文本或二进制大对象时，变量非常有用。

过程和触发器中的局部变量在复合语句中声明 （请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 使用复合语句 "）。

权限

无。

副作用

无。

另请参阅

"BEGIN 语句 " 第 288 页

"SQL 数据类型 " 第 59 页

"DROP VARIABLE 语句 " 第 464 页

"SET 语句 " 第 592 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

有关示例，请参阅 ""SET 语句 " 第 592 页 "。

CREATE VIEW 语句

说明

此语句用于创建数据库视图。视图用于以一种不同的角度提供数据，即使它不是以那种方式存储的。

语法

CREATE VIEW
[*owner*.]*view-name* [**(** *column-name*, … **)**]
AS *select-without-order-by*
[**WITH CHECK OPTION**]

参数

view-name *view-name* 是标识符。缺省所有者是当前用户 ID。

column-name 视图中的列名由 *column-name* 列表指定。如果没有指定列名列表，则视图列名来自选择列表项。为了使用选择列表项中的名称，每项必须是简单列名或者指定了别名 （请参阅 ""SELECT 语句 " 第 587 页 "）。此选择列表中的所有项的名称必须是唯一的。

AS 子句 视图基于的 SELECT 语句不能有 ORDER BY 子句。它可以有 GROUP BY 子句并且可以是 UNION。 SELECT 语句不能引用局部临时表。

WITH CHECK OPTION 子句 WITH CHECK OPTION 子句拒绝不满足 SELECT 语句定义的视图条件的任何视图更新和插入。

用法

CREATE VIEW 语句创建具有给定名称的视图。通过指定 **owner** 可以创建由其他用户拥有的视图。必须拥有 DBA 权限才能为其他用户创建的视图。

在 SELECT、 DELETE、 UPDATE 和 INSERT 语句中，可以使用视图名取代表名。但在数据库中，视图实际上不是以表的形式存在。每次使用时它们就会派生。视图来自于 CREATE VIEW 语句中指定的 SELECT 语句。视图中使用的表名应该由表所有者的用户 ID 限定。否则，一个不同的用户 ID 可能无法找到表或者可能得到错误的表。

视图可以更新，除非定义视图的 SELECT 语句包含 GROUP BY 子句、集合函数或涉及 UNION 操作。视图的更新会导致基表更新。

权限

必须有 RESOURCE 权限和视图定义中的表的 SELECT 权限。

副作用

自动提交。

另请参阅

"DROP 语句 " 第 445 页

"CREATE TABLE 语句 " 第 395 页

标准和兼容性

- **SQL/92**　初级特性。
- **SQL/99**　核心特性。
- **Sybase**　受 Adaptive Server Enterprise 支持。

示例

下面的示例创建仅显示男雇员信息的视图。此视图和基表具有同样的列名。

```
CREATE VIEW male_employee
AS SELECT *
FROM Employee
WHERE Sex = 'M'
```

下面的示例创建显示雇员和他们所属部门的视图。

```
CREATE VIEW emp_dept
AS SELECT emp_lname, emp_fname, dept_name
FROM Employee JOIN Department
ON Employee.dept_id = Department.dept_id
```

CREATE WRITEFILE 语句

说明

此语句用于创建数据库的写文件。

语法

CREATE WRITEFILE *write-file-name*
    FOR DATABASE *db-file-name* [**KEY** *key*]
    [**LOG OFF** | **LOG ON** [*log-file-name* [**MIRROR** *mirror-file-name*]]]

write-file-name | *db-file-name* | *log-file-name* | *mirror-file-name* : *string*

用法

用提供的名称和属性创建数据库的写文件。

文件名 (*write-file-name*, *db-file-name*, *log-file-name*, *mirror-file-name*) 是包含操作系统文件名的字符串。

有关字符串的信息，请参阅 "" 字符串 " 第 10 页 "。

如果不指定路径或相对路径，则在相对于服务器的当前工作目录的位置创建文件。

不能创建当前正在装载的数据库的写文件。

权限

执行此语句所需的权限在服务器命令行上用 -gu 选项来设置。缺省设置为要求具有 DBA 权限。

运行服务器时所使用的帐户必须对创建文件的目录具有写权限。

在 Windows CE 上不受支持。

如果要创建高度加密数据库的写文件，必须指定 KEY 值。

副作用

创建操作系统文件。

另请参阅

"CREATE DATABASE 语句 " 第 314 页

"ASA 数据库管理指南 > 数据库管理实用程序 > Write File 实用程序 "

"ASA 数据库管理指南 > 使用数据库文件 > 使用写文件 "

"ASA 数据库管理指南 > 连接参数和通信参数 > Encryption Key 连接参数 [DBKEY]"

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。

示例

下面的语句创建一个写文件。

```
CREATE WRITEFILE 'c:\\sybase\\my_db.wrt'
FOR DATABASE 'c:\\sybase\\my_db.db'
LOG ON 'e:\\logdrive\\my_db.log'
```

DEALLOCATE 语句

说明

此语句用于释放与某个游标关联的资源。

语法

DEALLOCATE [CURSOR] *cursor-name*

cursor-name : *identifier*

用法

释放与游标关联的所有内存，包括数据项、指示符变量和结构本身。

此选项在 Adaptive Server Anywhere 中没有任何作用。提供此选项是为了与 Adaptive Server Enterprise 和 Microsoft SQL Server 兼容。在 Adaptive Server Enterprise 中，要求使用 CURSOR 关键字。在 Microsoft SQL Server 中，不允许使用此关键字。 Adaptive Server Anywhere 对两种形式都识别。

权限

无。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

DEALLOCATE DESCRIPTOR 语句 [ESQL]

说明　此语句用于释放与 SQL 描述符区关联的内存。

语法　**DEALLOCATE DESCRIPTOR** *descriptor-name*

descriptor-name :　*string*

用法　释放与描述符区关联的所有内存，包括数据项、指示符变量和结构本身。

权限　无。

副作用　无。

另请参阅　"ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页

"ASA 编程指南 > 嵌入式 SQL 编程 > SQL 描述符区域 (SQLDA)"

"SET DESCRIPTOR 语句 [ESQL]" 第 599 页

标准和兼容性

- **SQL/92**　初级特性。
- **SQL/99**　核心特性。
- **Sybase**　受 Open Client/Open Server 支持。

示例　有关示例，请参阅 "ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页。

声明部分 [ESQL]

说明

此语句用于在嵌入式 SQL 程序中声明主机变量。主机变量用于同数据库交换数据。

语法

EXEC SQL BEGIN DECLARE SECTION;
C declarations
EXEC SQL END DECLARE SECTION;

用法

声明部分只是由 BEGIN DECLARE SECTION 和 END DECLARE SECTION 语句包围的 C 变量声明部分。声明部分使 SQL 处理器可以识别将用作主机变量的 C 变量。并非所有的 C 声明在声明部分内都有效。有关更多信息，请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > 使用主机变量 "。

权限

无。

另请参阅

"BEGIN 语句 " 第 288 页

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** 与 Adaptive Server Enterprise 兼容。

示例

```
EXEC SQL BEGIN DECLARE SECTION;
char *emp_lname, initials[5];
int dept;
EXEC SQL END DECLARE SECTION;
```

DECLARE 语句

说明

此语句用于在复合语句 (BEGIN … END) 中声明 SQL 变量。

语法

DECLARE *variable-name data-type*

用法

在过程、触发器或批处理主体中使用的变量可以用 DECLARE 语句声明。变量在声明它的复合语句的持续时间内存在。

Watcom-SQL 过程或触发器的主体是复合语句，变量必须紧跟在 BEGIN 后声明。在 Transact-SQL 过程或触发器中没有此限制。

标准和兼容性

- **SQL/92**　持久存储模块特性。
- **SQL/99**　持久存储模块特性。
- **Sybase**　受 Adaptive Server Enterprise 支持。
  - 为与 Adaptive Server Enterprise 兼容，变量名的前面必须有一个 @。
  - 在 Adaptive Server Enterprise 中，过程或触发器中声明的变量在过程或触发器的持续时间内存在。在 Adaptive Server Anywhere 中，如果在复合语句内声明某变量，该变量仅在此复合语句的持续时间内存在（不论它是在 Watcom-SQL 还是在 Transact-SQL 复合语句内声明的）。

示例

下面批处理语句阐释了 DECLARE 语句的使用并在服务器窗口上输出消息：

```
BEGIN
  DECLARE varname CHAR(61);
  SET varname = 'Test name';
  MESSAGE varname;
END
```

DECLARE CURSOR 语句 [ESQL] [SP]

说明

此语句用于声明游标。游标是处理查询结果的主要方法。

语法 1 [ESQL]

DECLARE *cursor-name*
[**UNIQUE**]
[**NO SCROLL**
| **DYNAMIC SCROLL**
| **SCROLL**
| **INSENSITIVE**
| **SENSITIVE**
]
CURSOR FOR
{ *select-statement*
| *statement-name*
[**FOR** { **UPDATE** [*cursor-concurrency*] | **READ ONLY** }]
| *call-statement* }

语法 2 [SP]

DECLARE *cursor-name*
[**NO SCROLL**
| **DYNAMIC SCROLL**
| **SCROLL**
| **INSENSITIVE**
| **SENSITIVE**
]
CURSOR FOR
{ *select -statement*
[**FOR** { **UPDATE** [*cursor-concurrency*] | **READ ONLY** }]
| *call-statement*
| **USING** *variable-name* }

cursor-name : *identifier*

statement-name : *identifier* | *hostvar*

variable-name : *identifier*

cursor-concurrency :
BY { **VALUES** | **TIMESTAMP** | **LOCK** }

参数

UNIQUE 当游标声明为 UNIQUE 时，查询被强制返回唯一标识每行所需的所有列。这通常意味着确保返回主键或唯一性表约束中的所有列。任何需要的但是在查询中未指定的列都添加到结果集中。

UNIQUE 游标上完成的 DESCRIBE 在指示符变量中设置以下附加标志：

- DT_KEY_COLUMN 此列是行键的一部分
- DT_HIDDEN_COLUMN 此列被添加到查询中，因为唯一标识行需要它。

NO SCROLL 声明为 NO SCROLL 的游标仅限于使用 FETCH NEXT 和 FETCH RELATIVE 0 查找操作在结果集中前进。

因为一旦游标离开行，行就不能返回，所以游标没有敏感性限制。因此，当请求 NO SCROLL 游标时， Adaptive Server Anywhere 提供效率最高的游标，即敏感性未定型游标。

有关更多信息，请参阅 "ASA 编程指南 > 在应用程序中使用 SQL > 敏感性未定型游标 "。

DYNAMIC SCROLL DYNAMIC SCROLL 是缺省游标类型。DYNAMIC SCROLL 游标可以使用 FETCH 语句的所有形式。

当请求 DYNAMIC SCROLL 游标时， Adaptive Server Anywhere 提供敏感性未定型游标。当使用游标时，在效率和一致性之间总存在平衡问题。敏感性未定型游标以一致性为代价提供高效性能。

有关更多信息，请参阅 "ASA 编程指南 > 在应用程序中使用 SQL > 敏感性未定型游标 "。

SCROLL 声明为 SCROLL 的游标可以使用 FETCH 语句的所有形式。当请求 SCROLL 游标时， Adaptive Server Anywhere 提供对值敏感的游标。

有关更多信息，请参阅 "ASA 编程指南 > 在应用程序中使用 SQL > 对值敏感的游标 "。

Adaptive Server Anywhere 必须以保证结果集成员资格的方式执行对值敏感的游标。 DYNAMIC SCROLL 游标的效率更高，应该使用它，除非要求 SCROLL 游标的一致行为。

INSENSITIVE 声明为 INSENSITIVE 的游标在打开时修复其成员资格；系统会使用所有原始行的副本创建一个临时表。 INSENSITIVE 游标中的 FETCHING 看不到其它任何 INSERT、 UPDATE 或 DELETE 语句的效果，也看不到另一个游标上的其它任何 PUT、 PDATE WHERE

CURRENT、 DELETE WHERE CURRENT 操作。它看不到同一游标上 PUT、 UPDATE WHERE CURRENT、 DELETE WHERE CURRENT 操作的效果。

有关更多信息，请参阅 "ASA 编程指南 > 在应用程序中使用 SQL > 不敏感游标 "。

SENSITIVE 声明为 SENSITIVE 的游标对结果集的成员资格或值的更改敏感。

有关更多信息，请参阅 "ASA 编程指南 > 在应用程序中使用 SQL > 敏感游标 "。

FOR statement-name 使用 PREPARE 语句为语句命名。只能为准备的 SELECT 或 CALL 声明游标。

FOR UPDATE | READ ONLY 声明为 FOR READ ONLY 的游标不能在 UPDATE （定位）或 DELETE （定位）操作中使用。 FOR UPDATE 是缺省设置。

为响应指定 FOR UPDATE 的游标的请求， Adaptive Server Anywhere 提供对值敏感的游标或敏感性未定型的游标。不敏感的游标或者敏感性未定型游标不能更新。

USING variable-name 仅用于存储过程。此变量是包含游标 SELECT 语句的字符串。当处理 DECLARE 时，此变量必须可用，因此必须是以下之一：

- 过程的参数。例如，

```
create function get_row_count(in qry long varchar)
returns int
begin
  declare crsr cursor using qry;
  declare rowcnt int;
  set rowcnt = 0;
  open crsr;
  lp: loop
    fetch crsr;
```

```
    if SQLCODE <> 0 then leave lp end if;
    set rowcnt = rowcnt + 1;
  end loop;
  return rowcnt;
end
```

- 此变量被赋值后，嵌套在另一个 BEGIN… END 内。例如，

```
create procedure get_table_name(
  in id_value int, out tabname char(128)
)
begin
  declare qry long varchar;
  set qry = 'select table_name from SYS.SYSTABLE '
||
            'where table_id=' || string(id_value);
  begin
    declare crsr cursor using qry;
    open crsr;
    fetch crsr into tabname;
    close crsr;
  end
end
```

用法

DECLARE CURSOR 语句用 SELECT 语句或 CALL 语句的指定名称声明游标。

权限

无。

副作用

无。

另请参阅

"PREPARE 语句 [ESQL]" 第 553 页

"OPEN 语句 [ESQL] [SP]" 第 543 页

"EXPLAIN 语句 [ESQL]" 第 473 页

"SELECT 语句 " 第 587 页

"CALL 语句 " 第 294 页

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。

- **Sybase** 受 Open Client/Open Server 支持。

示例

下面的示例阐释如何在嵌入式 SQL 中声明滚动游标：

```
EXEC SQL DECLARE cur_employee SCROLL CURSOR
FOR SELECT * FROM employee;
```

下面的示例阐释如何在嵌入式 SQL 中为准备的语句声明游标：

```
EXEC SQL PREPARE employee_statement
FROM 'SELECT emp_lname FROM employee';
EXEC SQL DECLARE cur_employee CURSOR
FOR employee_statement;
```

下面的示例阐释游标在存储过程中的使用：

```
BEGIN
  DECLARE cur_employee CURSOR FOR
      SELECT emp_lname
      FROM employee;
  DECLARE name CHAR(40);
  OPEN cur_employee;
  LOOP
    FETCH NEXT cur_employee INTO name;
    ...
  END LOOP
  CLOSE cur_employee;
END
```

DECLARE CURSOR 语句 [T-SQL]

说明

此语句用于以与 Adaptive Server Enterprise 兼容的方式声明游标。

语法

DECLARE *cursor-name*
CURSOR FOR *select-statement*
[**FOR** { **READ ONLY** | **UPDATE** }]

cursor-name : *identifier*

select-statement : *string*

用法

Adaptive Server Anywhere 支持 Adaptive Server Enterprise 中不支持的 DECLARE CURSOR 语法。有关全部 DECLARE CURSOR 语法的信息，请参阅 ""DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页 "。

本节说明 Adaptive Server Anywhere 和 Enterprise 之间 DECLARE CURSOR 风格的重叠部分。

权限

无。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

标准和兼容性

- **SQL/92** 初级特性。 FOR UPDATE 和 FOR READ ONLY 选项是 Transact-SQL 扩展。
- **SQL/92** 核心特性。 FOR UPDATE 和 FOR READ ONLY 选项是 Transact-SQL 扩展。
- **Sybase** Adaptive Server Enterprise DECLARE CURSOR 语句的一些特性在 Adaptive Server Anywhere 中不受支持。
  - Adaptive Server Enterprise 支持打开游标以更新 *select-statement* 中指定的表的列列表。这在 Adaptive Server Anywhere 中不受支持。
  - 在 Watcom-SQL 方言中，过程、触发器或批处理中的 DECLARE CURSOR 语句必须紧跟在 BEGIN 关键字后。Transact-SQL 方言没有此限制。

- 在 Adaptive Server Enterprise 中，当在过程、触发器或批处理中声明游标时，它在过程、触发器或批处理的持续时间内存在。在 Adaptive Server Anywhere 中，如果在复合语句中声明游标，则它仅在该复合语句的持续时间内存在（不论它是在 Watcom-SQ 还是在 Transact-SQL 复合语句中声明的）。
- 在 Adaptive Server Anywhere 中不支持 CURSOR *type*（UNIQUE、 NO SCROLL 等）和 CURSOR FOR *statement-name*。

DECLARE LOCAL TEMPORARY TABLE 语句

说明

此语句用于声明局部临时表。

语法

DECLARE LOCAL TEMPORARY TABLE *table-name*
({ *column-definition* [*column-constraint* …] | *table-constraint* | *pctfree* }, …
)
[**ON COMMIT** { **DELETE** | **PRESERVE** } **ROWS**]

pctfree : **PCTFREE** *percent-free-space*

percent-free-space : integer

参数

PCTFREE 指定想为每个表页保留的可用空间百分比。如果数据更新时行大小增加，将占用可用空间。如果表页中没有可用空间，则该页上的行大小每次增加时，行都需要在多个表页中拆分，从而导致行碎片并可能引起性能下降。

percent-free-space 值是 0 到 100 之间的整数。前者指定每页上没有剩余的可用空间：每页都完全填满。高值会将每行单独插入到页中。如果没有设置 PCTFREE，则每页保留 200 字节。

用法

DECLARE LOCAL TEMPORARY TABLE 语句声明临时表。有关 *column-definition*、 *column-constraint* 和 *table-constraint* 的定义，请参阅 ""CREATE TABLE 语句 " 第 395 页 "。

复合语句中声明的临时表存在于复合语句中。（请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 使用复合语句 "）。否则，声明的局部临时表在连接断开前一直存在。

缺省情况下，临时表的行将在 COMMIT （提交）时被删除。

权限

无。

副作用

无。

另请参阅

"CREATE TABLE 语句 " 第 395 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 使用复合语句 "

标准和兼容性

- **SQL/92** 符合 SQL/92 标准。

- **SQL/99** 核心 SQL 外部的 SQL/ 基础特性。
- **Sybase** Adaptive Server Enterprise 不支持 DECLARE TEMPORARY TABLE。

示例

下面的示例阐释如何在嵌入式 SQL 中声明临时表：

```
EXEC SQL DECLARE LOCAL TEMPORARY TABLE MyTable (
  number INT
   );
```

下面的示例阐释如何在存储过程中声明临时表：

```
BEGIN
  DECLARE LOCAL TEMPORARY TABLE TempTab (
    number INT
  );
  ...
END
```

DELETE 语句

说明

此语句用于从数据库中删除行。

语法

DELETE [**FIRST** | **TOP** *n*]
[**FROM**] [*owner*.]*table-name*
[**FROM** *table-list*]
[**WHERE** *search-condition*]

用法

DELETE 语句删除指定表中所有满足搜索条件的行。如果没有指定 WHERE 子句，则删除指定表中的所有行。

如果定义视图的 SELECT 语句的 FROM 子句中只有一个表并且不包含 GROUP BY 子句、集合函数或 UNION 操作，则可以在视图上使用 DELETE 语句。

DELETE 语句中可选的第二个 FROM 子句允许基于连接删除行。如果第二个 FROM 子句存在，则 WHERE 子句限定此第二个 FROM 子句的行。行从第一个 FROM 子句给出的表名中删除。

第二个 FROM 子句可以包含任意复杂的表达式，例如 KEY 和 NATURAL 连接。有关 FROM 子句和连接的完整说明，请参阅 ""FROM 子句 " 第 484 页 "。

下面的语句阐释 DELETE 语句（有两个使用相关名的 FROM 子句）的表名中潜在的不明确性：

```
DELETE
FROM table_1
FROM table_1 AS alias_1, table_2 AS alias_2
WHERE ...
```

table_1 表在第一个 FROM 子句中标识（没有相关名），但是在第二个 FROM 子句中有相关名。此情况下，第一个子句的 *table_1* 以第二个子句中的 *alias_1* 标识：此语句中只有一个 *table_1* 实例。

这是一般规则的特例，即一个表在同一语句中的两个实例，一次用相关名标识另一次没有相关名。

请看下面的示例：

```
DELETE
FROM table_1
FROM table_1 AS alias_1, table_1 AS alias_2
WHERE ...
```

此情况下，第二个 FROM 子句中有两个 *table_1* 实例。此语句会因语法错误而失败，因为第二个 FROM 子句中的 *table_1* 实例与第一个 FROM 子句中的 *table_1* 实例匹配，这是不明确的。

权限

必须有表的 DELETE 权限。

副作用

无。

另请参阅

"TRUNCATE TABLE 语句 " 第 634 页

"INSERT 语句 " 第 520 页

"INPUT 语句 [Interactive SQL]" 第 515 页

"FROM 子句 " 第 484 页

标准和兼容性

- **SQL/92** 初级遵从。在 FROM 子句中使用一个以上的表是供应商扩展。
- **SQL/99** 核心特性。在 FROM 子句中使用一个以上的表是供应商扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持 （包括供应商扩展）。

示例

将 ID 为 105 的雇员从数据库中删除。

```
DELETE
FROM employee
WHERE emp_id = 105
```

从 **fin_data** 表中删除 2000 年之前的所有数据。

```
DELETE
FROM fin_data
WHERE year < 2000
```

从 **sales_order_items** 表中删除所有订单，如果订单的发货日期早于 2001 年 1 月 1 日并且它们的地区是 "Central"。

```
DELETE
FROM sales_order_items
FROM sales_order
WHERE sales_order_items.id = sales_order.id
  and ship_date < '2001-01-01' and region ='Central'
```

DELETE （定位）语句 [ESQL] [SP]

说明

此语句用于从游标的当前位置删除数据。

语法

DELETE [**FROM** *table-spec*] **WHERE CURRENT OF** *cursor-name*

cursor-name : *identifier* | *hostvar*

table-spec : [*owner*.]*correlation-name*

owner : *identifier*

用法

这种形式的 DELETE 语句删除指定游标的当前行。当前行被定义为从游标最近读取的一行。

从中删除行的表由以下确定：

- 如果不包括 FROM 子句，则游标必须仅在一个表中。
- 如果游标用于连接查询 （包括使用包含连接的视图），则必须使用 FROM 子句。仅删除指定表的当前行。不影响连接中的其它表。
- 如果包括 FROM 子句但未指定表所有者，则 *table-spec* 首先与任何相关名进行匹配。
  - 如果相关名存在，则 *table-spec* 由相关名标识。
  - 如果相关名不存在，则 *table-spec* 必须明确标识为游标中的表名。
- 如果包括 FROM 子句并且指定了表的所有者，则 *table-spec* 必须明确标识为游标中的表名。
- 只要视图是可更新的，定位 DELETE 语句可以在视图中打开的游标上使用。

权限

必须有游标中使用的表的 DELETE 权限。

副作用

无。

另请参阅

"UPDATE 语句 " 第 642 页

"UPDATE （定位）语句 [ESQL] [SP]" 第 647 页

"INSERT 语句 " 第 520 页

"PUT 语句 [ESQL]" 第 558 页

标准和兼容性

- **SQL/92** 初级特性。如果 ANSI_UPDATE_CONSTRAINTS 选项设置为 OFF，可以更新的游标的范围可能包含供应商扩展。
- **SQL/99** 核心特性。如果 ANSI_UPDATE_CONSTRAINTS 选项设置为 OFF，可以更新的游标的范围可能包含供应商扩展。
- **Sybase** Open Client/Open Server 支持使用嵌入式 SQL。过程和触发器的使用仅受 Adaptive Server Anywhere 支持。

示例

下面的语句从数据库中删除当前行。

```
DELETE
WHERE CURRENT OF cur_employee
```

DESCRIBE 语句 [ESQL]

说明

此语句用于获取有关存储从数据库中检索的数据所需的主机变量的信息，以及向数据库传递数据所需的主机变量的信息。

语法

DESCRIBE
[**USER TYPES**]
[**ALL** | **BIND VARIABLES FOR** | **INPUT** | **OUTPUT**
| **SELECT LIST FOR**]
[**LONG NAMES** [*long-name-spec*] | **WITH VARIABLE RESULT**]
[**FOR**] { *statement-name* | **CURSOR** *cursor-name* }
INTO *sqlda-name*

long-name-spec :
OWNER.TABLE.COLUMN | **TABLE.COLUMN** | **COLUMN**

statement-name : identifier | *hostvar*

cursor-name : declared cursor

sqlda-name : identifier

参数

USER TYPES 带 USER TYPES 子句的 DESCRIBE 语句返回列的域信息。通常，当前一个 DESCRIBE 返回 DT_HAS_USERTYPE_INFO 的指示符时会做这样的 DESCRIBE。

返回的信息和不带 USER TYPES 关键字的 DESCRIBE 返回的信息是相同的，但 **sqlname** 字段保存的是域名而不是列名。

如果 DESCRIBE 使用 LONG NAMES 子句，则 **sqldata** 字段保存此信息。

ALL DESCRIBE ALL 允许用一个数据库服务器请求描述 INPUT 和 OUTPUT。这样可以提高性能。 INPUT 信息首先填充 SQLDA，然后是 OUTPUT 信息。 **sqld** 字段包含 INPUT 和 OUTPUT 变量的总数。在指示符变量中，为 INPUT 变量设置了 DT_DESCRIBE_INPUT 位，对 OUTPUT 变量则清除了此位。

INPUT 绑定变量是数据库执行语句时由应用程序提供的值。绑定变量可以被看成是语句的参数。 DESCRIBE INPUT 用绑定变量名填充 SQLDA 中的名称字段。 DESCRIBE INPUT 还将绑定变量的数目放在 SQLDA 的 **sqld** 字段中。

DESCRIBE 使用 SQLDA 中的指示符变量提供其它信息。DT_PROCEDURE_IN 和 DT_PROCEDURE_OUT 是描述 CALL 语句时在指示符变量中设置的位。 DT_PROCEDURE_IN 指示 IN 或 INOUT 参数， DT_PROCEDURE_OUT 指示 INOUT 或 OUT 参数。过程 RESULT 列将清除这两个位。描述 OUTPUT 后，这些位可以用于区分有结果集的语句 （需要使用 OPEN、 FETCH、 RESUME、 CLOSE）和没有结果集的语句 （需要使用 EXECUTE）。当绑定变量是 CALL 语句的参数时， DESCRIBE INPUT 仅相应地设置 DT_PROCEDURE_IN 和 DT_PROCEDURE_OUT ； 作为 CALL 语句参数的表达式中的绑定变量不会设置这些位。

OUTPUT　DESCRIBE OUTPUT 语句填充 SQLDA 中的每个选择列表项的数据类型和长度。名称字段也用选择列表项的名称填充。如果为选择列表项指定了别名，则名称就是该别名。否则，名称将从选择列表项中派生：如果项是简单列名，则使用它；否则将使用表达式的子字符串。DESCRIBE 还会将选择列表项的数目放在 SQLDA 的 **sqld** 字段中。

如果语句被描述为两个或更多个 SELECT 语句的 UNION，则为 DESCRIBE OUTPUT 返回的列名与为第一个 SELECT 语句返回的列名相同。

如果描述 CALL 语句，则 DESCRIBE OUTPUT 语句为过程中的每个 INOUT 或 OUT 参数填充 SQLDA 中的数据类型、长度和名称。DESCRIBE OUTPUT 还将 INOUT 或 OUT 参数的数目放在 SQLDA 的 **sqld** 字段中。

如果用结果集描述 CALL 语句，则 DESCRIBE OUTPUT 语句为过程定义中的每个 RESULT 列填充 SQLDA 中的数据类型、长度和名称。DESCRIBE OUTPUT 还会将结果列的数目放在 SQLDA 的 **sqld** 字段中。

LONG NAMES　提供 LONG NAMES 子句以检索语句或游标的列名。若无此子句，则列名长度有 29 个字符的限制；有此子句，则支持任意长度的名称。

如果使用了 LONG NAMES，长名称就会放到 SQLDA 的 SQLDATA 字段中，就象是从游标中读取一样。不填充其它字段 （SQLLEN、SQLTYPE 等）。必须将 SQLDA 设置成类似 FETCH SQLDA：它必须包含每列的一个条目，每个条目必须是字符串类型。如果存在指示符变量，会以通常的方式指示截断。

长名称的缺省说明为 **TABLE.COLUMN**。

WITH VARIABLE RESULT 此子句用于描述可能有不止一个结果集（列数或列类型不同）的过程。

如果使用了 WITH VARIABLE RESULT，则数据库服务器会在 DESCRIBE 语句后将 SQLCOUNT 值设置为以下值之一：

- **0** 结果集可以更改。在每个 OPEN 语句应该重新描述过程调用。
- **1** 结果集是固定的。不需要重新描述。

有关如何使用 SQLDA 结构的更多信息，请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > SQL 描述符区域 (SQLDA)"。

用法

DESCRIBE 语句设置指定的 SQLDA 以描述指定语句的 OUTPUT （等效于 SELECT LIST）或 INPUT (BIND VARIABLES)。

在 INPUT 的情况中， DESCRIBE BIND VARIABLES 不在 SQLDA 中设置数据类型：这需要由应用程序完成。 ALL 关键字允许在一个 SQLDA 中描述 INPUT 和 OUTPUT。

如果指定了语句名，则以前必须已经使用 PREPARE 语句用同样的语句名准备了此语句，并且以前必须已经分配了 SQLDA （请参阅 ""ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页 "）。

如果指定了游标名，则以前必须已经声明和打开过此游标。缺省操作是描述 OUTPUT。仅 SELECT 语句和 CALL 语句有 OUTPUT。将 SQLDA 的 **sqld** 字段设置为零后，对其它任何语句或非动态游标执行 DESCRIBE OUTPUT 都不会显示输出。

权限

无。

副作用

无。

另请参阅

"ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"OPEN 语句 [ESQL] [SP]" 第 543 页

"PREPARE 语句 [ESQL]" 第 553 页

标准和兼容性

- **SQL/92** 属于 SQL/92 标准。一些子句是供应商扩展。

- **SQL/99**　核心特性。一些子句是供应商扩展。
- **Sybase**　一些子句受 Open Client/Open Server 支持。

示例

下面的示例显示如何使用 DESCRIBE 语句：

```
sqlda = alloc_sqlda( 3 );
EXEC SQL DESCRIBE OUTPUT
  FOR employee_statement
  INTO sqlda;
if( sqlda->sqld  >  sqlda->sqln ) {
  actual_size = sqlda->sqld;
  free_sqlda( sqlda );
  sqlda = alloc_sqlda( actual_size );
  EXEC SQL DESCRIBE OUTPUT
    FOR employee_statement
    INTO sqlda;
}
```

DISCONNECT 语句 [ESQL] [Interactive SQL]

说明

此语句用于删除数据库的当前连接。

语法

DISCONNECT [*connection-name* | **CURRENT** | **ALL**]

connection-name : *identifier*, *string,* or *hostvar*

用法

DISCONNECT 语句删除数据库服务器连接并且释放它使用的所有资源。如果在 CONNECT 语句中指定了要删除的连接，则可以指定此名称。指定 ALL 会删除应用程序与所有数据库环境的连接。 CURRENT 是缺省设置，它删除当前连接。

在删除的连接上会执行隐式 ROLLBACK。

有关删除非当前连接的信息，请参阅 ""DROP CONNECTION 语句 " 第 448 页 "。

权限

无。

副作用

无。

另请参阅

"CONNECT 语句 [ESQL] [Interactive SQL]" 第 308 页

"SET CONNECTION 语句 [Interactive SQL] [ESQL]" 第 598 页

标准和兼容性

- **SQL/92** 中间级特性。
- **SQL/99** 核心 SQL 外部的 SQL/ 基础特性。
- **Sybase** 受 Open Client/Open Server 支持。

示例

下面的语句显示如何在嵌入式 SQL 中使用 DISCONNECT：

```
EXEC SQL DISCONNECT :conn_name
```

下面的语句显示如何从 Interactive SQL 中使用 DISCONNECT 断开所有连接：

```
DISCONNECT ALL
```

DROP 语句

说明

此语句用于从数据库中删除对象。

语法

DROP
　{ **DATATYPE** | **DOMAIN** } *datatype-name*
| **DBSPACE** *dbspace-name*
| **EVENT** *event-name*
| **FUNCTION** [*owner*.]*function-name*
| **INDEX** [[*owner*.]*table-name*.]*index-name*
| **MESSAGE** *msgnum*
| **PROCEDURE** [*owner*.]*procedure-name*
| **TABLE** [*owner*.]*table-name*
| **TRIGGER** [[*owner*.]*table-name*.]*trigger-name*
| **VIEW** [*owner*.]*view-name*

用法

DROP 语句删除指定数据库结构的定义。如果结构是 dbspace，则在删除此 dbspace 之前必须先删除 dbspace 中的所有表。如果结构是表，则表中的所有数据作为删除过程的一部分自动删除。另外，表的所有索引和键都由 DROP TABLE 语句删除。

只要 DROP TABLE、 DROP INDEX 和 DROP DBSPACE 语句影响到当前正由另一连接使用的表，就会禁止这些语句。

当过程或函数正由另一连接使用时，会禁止 DROP PROCEDURE 和 DROP FUNCTION。

如果表中使用了数据类型，则会禁止 DROP DATATYPE。要删除数据类型，必须更改域上定义的所有列的数据类型。建议使用 DROP DOMAIN 而不是 DROP DATATYPE，因为 DROP DOMAIN 是 ANSI/ISO SQL3 草案中使用的语法。

权限

任何拥有对象或 DBA 权限的用户都可以执行 DROP 语句。

对于 DROP DBSPACE，您必须是数据库的唯一连接。

具有表的 ALTER 权限的用户可以执行 DROP TRIGGER。

具有表的 REFERENCES 权限的用户可以执行 DROP INDEX。

除非引用全局临时表的所有用户都已经断开，否则不能删除全局临时表。

副作用

自动提交。在 Interactive SQL 的 " 结果 " 窗格中清除 " 结果 " 选项卡。DROP TABLE 和 DROP INDEX 关闭当前连接的所有游标。

局部临时表是个特例，删除它时不执行提交。

删除视图时，所有过程和触发器都从内存中卸载，因此引用此视图的任何过程或触发器都会反映此视图不存在的事实。如果按常规删除和创建视图，则卸载和装载过程和触发器会对性能造成负面影响。

另请参阅

"CREATE DATABASE 语句 " 第 314 页

"CREATE DOMAIN 语句 " 第 322 页

"CREATE INDEX 语句 " 第 341 页

"CREATE FUNCTION 语句 " 第 337 页

"CREATE PROCEDURE 语句 " 第 347 页

"CREATE TABLE 语句 " 第 395 页

"CREATE TRIGGER 语句 " 第 408 页

"CREATE VIEW 语句 " 第 418 页

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** 对于 Adaptive Server Enterprise 中存在的对象，受 Adaptive Server Enterprise 支持。

示例

- 从数据库中删除部门表。

```
DROP TABLE department
```

从数据库中删除 *emp_dept* 视图。

```
DROP VIEW emp_dept
```

DROP DATABASE 语句

说明

此语句用于删除与数据库关联的数据库文件。

语法

DROP DATABASE *database-name* [**KEY** *key*]

用法

DROP DATABASE 语句在物理上从磁盘中删除所有关联的数据库文件。如果数据库文件不存在或者所处的状态不适合要启动的数据库，则会发生错误。

DROP DATABASE 不能在存储过程中使用。

权限

必需的权限通过数据库服务器 -gu 选项来设置。缺省设置为要求具有 DBA 权限。

数据库必须没有被使用才能进行删除。

如果要删除高度加密的数据库，必须指定密钥。

在 Windows CE 上不受支持。

副作用

除了从磁盘上删除数据库文件外，任何关联的事务日志文件或事务日志镜像文件也被删除。

另请参阅

"CREATE DATABASE 语句 " 第 314 页

"ASA 数据库管理指南 > 连接参数和通信参数 > Encryption Key 连接参数 [DBKEY]"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

删除 `C:\temp` 目录中的数据库 `temp.db`。

```
DROP DATABASE 'c:\temp\temp.db'
```

DROP CONNECTION 语句

说明

此语句用于删除属于任何用户的数据库连接。

语法

DROP CONNECTION *connection-id*

用法

DROP CONNECTION 语句通过删除数据库的连接，将用户同数据库断开。

通过使用 **connection_property** 函数请求连接号，可以获取 *connection-id*。下面的语句返回当前连接的连接 ID：

```
SELECT connection_property( 'number' )
```

权限

必须拥有 DBA 权限。

副作用

无。

另请参阅

"CONNECT 语句 [ESQL] [Interactive SQL]" 第 308 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的语句删除 ID 号为 4 的连接。

```
DROP CONNECTION 4
```

DROP EXTERNLOGIN 语句

说明 此语句用于从 Adaptive Server Anywhere 目录中删除外部登录。

语法 **DROP EXTERNLOGIN** *login-name* **TO** *remote-server*

参数 **DROP 子句** 指定本地用户登录名

TO 子句 指定远程服务器的名称。该服务器的本地用户替代登录名和口令是删除的外部登录。

用法 DROP EXTERNLOGIN 从 Adaptive Server Anywhere 目录中删除外部登录。

权限 必须是 *login-name* 的所有者或者有 DBA 权限。

副作用 自动提交。

另请参阅 "CREATE EXTERNLOGIN 语句 " 第 335 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Open Client/Open Server 支持。

示例

```
DROP EXTERNLOGIN DBA TO sybase1
```

DROP PUBLICATION 语句

说明

此语句可用于删除发布。在 MobiLink 中，发布可以标识 Adaptive Server Anywhere 远程数据库中的同步数据。在 SQL Remote 中，发布标识统一数据库和远程数据库中的复制数据。

语法

DROP PUBLICATION [*owner*.]*publication-name*

owner, *publication-name* : *identifier*

用法

此语句仅适用于 MobiLink 和 SQL Remote。

权限

必须拥有 DBA 权限。

副作用

自动提交。删除发布的所有预订。

另请参阅

"ALTER PUBLICATION 语句 " 第 254 页

"CREATE PUBLICATION 语句 " 第 357 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_drop_publication 过程 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

下面的语句删除 *pub_contact* 发布。

```
DROP PUBLICATION pub_contact
```

DROP REMOTE MESSAGE TYPE 语句 [SQL Remote]

说明

此语句用于从数据库中删除消息类型定义。

语法

DROP REMOTE MESSAGE TYPE *message-system*

message-system: **FILE** | **FTP** | **MAPI** | **SMTP** | **VIM**

用法

此语句从数据库中删除消息类型。

权限

必须拥有 DBA 权限。若要能够删除类型，一定不能有被授予此类型的 REMOTE 或 CONSOLIDATE 权限的用户。

副作用

自动提交。

另请参阅

"CREATE REMOTE MESSAGE TYPE 语句 [SQL Remote]" 第 360 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_drop_remote_type 过程 "

"SQL Remote 用户指南 > SQL Remote 管理 > 使用消息类型 "。

示例

下面的语句从数据库中删除 FILE 消息类型。

```
DROP REMOTE MESSAGE TYPE file
```

DROP SERVER 语句

说明

此语句用于从 Adaptive Server Anywhere 目录中删除远程服务器。

语法

DROP SERVER *server-name*

用法

DROP SERVER 从 Adaptive Server Anywhere 目录中删除远程服务器。必须删除已经为远程服务器定义的所有代理表之后，此语句才会成功。

权限

只有 DBA 帐户可以删除远程服务器。

在 Windows CE 上不受支持。

副作用

自动提交。

另请参阅

"CREATE SERVER 语句 " 第 364 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 受 Open Client/Open Server 支持。

示例

```
DROP SERVER ase_prod
```

DROP STATEMENT 语句 [ESQL]

说明　此语句用于释放语句资源。

语法　**DROP STATEMENT** [*owner*.]*statement-name*

statement-name : *identifier* | *hostvar*

用法　DROP STATEMENT 语句释放指定的准备语句所使用的资源。这些资源由成功的 PREPARE 语句分配，通常情况下直到释放数据库连接时才释放。

权限　必须已经准备了此语句。

副作用　无。

另请参阅　"PREPARE 语句 [ESQL]" 第 553 页

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　Open Client/Open Server 中不受支持。

示例　以下是 DROP STATEMENT 的用法示例：

```
EXEC SQL DROP STATEMENT S1;
EXEC SQL DROP STATEMENT :stmt;
```

DROP STATISTICS 语句

说明

此语句用于删除指定列的所有优化程序统计信息。

语法

DROP STATISTICS [**ON**] [*owner.*]*table-name* [**(** *column-list* **)**]

用法

Adaptive Server Anywhere 优化程序使用统计信息来确定每个语句的最佳执行策略。 Adaptive Server Anywhere 自动收集和更新这些统计信息。这些统计信息永久存储在数据库的系统表 SYSCOLSTAT 中。处理一个语句时所收集的统计信息对于搜索后续语句的有效执行方式很有用。

有时，统计信息可能变得不准确，或者相关的统计信息不可用。最有可能发生这种情况的时候是：添加、更新或删除了大量数据后，只执行了很少的查询。

DROP STATISTICS 语句从指定列的 SYSCOLSTAT 系统表中删除所有内部统计数据。这种剧烈的动作导致优化程序无法访问基本的统计信息。没有了这些统计信息，优化程序就可能生成非常低效的数据访问计划，导致数据库性能降低。

仅当在确定问题时或者向数据库中重新装入与原数据有很大差别的数据时，才应该使用此语句。

版本 8 之前的数据库 DROP STATISTICS 语法对 Adaptive Server Anywhere 7 和更早版本的数据库无效。要删除那些数据库的统计信息，请使用此语法：

DROP OPTIMIZER STATISTICS

此语法删除数据库的所有统计信息。如果对版本 8 数据库使用此语法，什么也不会发生：统计信息不会被删除。此语法不建议使用。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"CREATE STATISTICS 语句 " 第 366 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

- **Sybase**　不受 Adaptive Server Enterprise 支持。

DROP SUBSCRIPTION 语句 [SQL Remote]

说明

此语句用于从发布中删除用户的预订。

语法

DROP SUBSCRIPTION TO *publication-name* [**(** *subscription-value* **)**]
FOR *subscriber-id*, …

subscription-value: *string*

subscriber-id: *string*

参数

publication-name 用户预订的发布的名称。这可以包括发布的所有者。

subscription-value 与发布的预订表达式进行比较的字符串。此值是必需的，因为用户可能有不止一个的发布预订。

subscriber-id 发布预订者的用户 ID。

用法

在当前数据库中，删除发布用户 ID 的 SQL Remote 预订。当发布中的数据更改时，此用户 ID 不再接收更新。

在 SQL Remote 中，发布和预订是一种双向关系。如果删除远程用户对统一数据库发布的预订，则还应该删除远程数据库的统一数据库预订，以防止远程数据库的更新发送到统一数据库。

权限

必须拥有 DBA 权限。

副作用

自动提交。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

另请参阅

"CREATE SUBSCRIPTION 语句 [SQL Remote]" 第 367 页

示例

下面的语句删除用户 ID **SamS** 对发布 **pub_contact** 的预订。

```
DROP SUBSCRIPTION TO pub_contact
FOR SamS
```

DROP SYNCHRONIZATION DEFINITION 语句 [MobiLink]（不建议使用）

说明

此语句可用于删除同步定义。不建议使用此命令。定义和站点已替换为同步发布和预订。

语法

DROP SYNCHRONIZATION DEFINITION *sync-def-name*

用法

同步定义在作为 MobiLink 客户端运行的 Adaptive Server Anywhere 第 7 版数据库中创建。每个定义指定在 MobiLink 设置中唯一标识该 MobiLink 客户端站点的逻辑名称。另外，每个站点指定如何联系 MobiLink 同步服务器以及远程数据库中的哪些数据将与统一数据库同步。

DROP SYNCHRONIZATION DEFINITION 语句用于删除同步定义。

不过请注意，一旦删除同步定义，未完成的更新 （插入、删除、更新和变更命令）就不会同步。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"DROP PUBLICATION 语句 " 第 450 页

"DROP SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 460 页

"CREATE SYNCHRONIZATION DEFINITION 语句 [MobiLink]（不建议使用） " 第 369 页

"ALTER SYNCHRONIZATION DEFINITION 语句 （不建议使用） " 第 260 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例

删除 *mysharedtables* 同步定义。

```
DROP SYNCHRONIZATION DEFINITION mysharedtables;
```

DROP SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）

说明 此语句可用于删除特定的同步站点。不建议使用此命令。定义和站点已替换为同步发布和预订。

语法 **DROP SYNCHRONIZATION SITE** *sync-site-name*

用法 仅当通过从 Adaptive Server Anywhere 第 7 版参考数据库中抽取的方式创建 Adaptive Server Anywhere 远程数据库时，才使用同步模板和同步站点。

每个远程数据库从同步站点中创建并存储在参考数据库中。每个同步站点都基于单个同步模板，不过许多站点可以使用单个模板。

DROP SYNCHRONIZATION SITE 语句用于删除特定的同步站点。

权限 必须拥有 DBA 权限。

副作用 自动提交。

另请参阅 "DROP PUBLICATION 语句 " 第 450 页

"DROP SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 460 页

"CREATE SYNCHRONIZATION SITE 语句 [MobiLink]（不建议使用）" 第 372 页

"ALTER SYNCHRONIZATION SITE 语句 [MobiLink] （不建议使用） " 第 263 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例 *mytemplate* 模板不再部署到 *new_sync_site* 远程数据库。

```
DROP SYNCHRONIZATION SITE new_sync_site;
```

DROP SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]

说明

此语句可用于在 MobiLink 远程数据库或 MobiLink 参考数据库中删除同步预订。还可以使用它为指定的发布删除包含缺省预订值的缺省预订。

语法

DROP SYNCHRONIZATION SUBSCRIPTION
TO *publication-name*
[**FOR** *ml_username*, …]

参数

TO 子句 指定发布名称。

FOR 子句 再指定一个 MobiLink 用户。

省略该子句将删除发布的缺省预订。已预订发布的 MobiLink 用户将缺省发布中的值按缺省值继承。

用法

删除 MobiLink 远程或参考数据库中的同步预订。

权限

必须拥有 DBA 权限。要求可以对发布中涉及的所有表进行独占访问。

副作用

自动提交。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 8.0 版。

示例

取消 MobiLink 用户 *ml_user1* 对销售发布的预订。

```
DROP SYNCHRONIZATION SUBSCRIPTION
   TO sales_publication
   FOR "ml_user1"
```

删除销售发布的缺省预订 （它包含缺省预订值）（通过省略 FOR 子句）。

```
DROP SYNCHRONIZATION SUBSCRIPTION
   TO sales_publication
```

DROP SYNCHRONIZATION TEMPLATE 语句 [MobiLink]（不建议使用）

说明

此语句可用于删除同步模板。不建议使用此命令。请使用同步发布和预订命令。

语法

DROP SYNCHRONIZATION TEMPLATE *sync-template-name*

用法

仅当通过从 Adaptive Server Anywhere 第 7 版参考数据库中抽取的方式创建 Adaptive Server Anywhere 远程数据库时，才使用同步模板和同步站点。

每个远程数据库从同步站点中创建并存储在参考数据库中。每个同步站点都基于单个同步模板，不过许多站点可以使用单个模板。

DROP SYNCHRONIZATION TEMPLATE 语句用于删除同步模板。

权限

必须拥有 DBA 权限。

副作用

自动提交。删除同步模板的同时将隐式删除使用该模板的所有站点。

另请参阅

"DROP PUBLICATION 语句 " 第 450 页

"DROP SYNCHRONIZATION SUBSCRIPTION 语句 [MobiLink]" 第 460 页

"CREATE SYNCHRONIZATION TEMPLATE 语句 [MobiLink]（不建议使用）" 第 377 页

"ALTER SYNCHRONIZATION TEMPLATE 语句 [MobiLink]（不建议使用）" 第 267 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 7.0 版。

示例

- 删除 *mytemplate* 同步模板。

```
DROP SYNCHRONIZATION TEMPLATE mytemplate;
```

DROP SYNCHRONIZATION USER 语句 [MobiLink]

说明 此语句可用于从 MobiLink 远程数据库删除同步用户。

语法 **DROP SYNCHRONIZATION USER** *ml_username*, …

ml_username: *identifier*

用法 从 MobiLink 远程数据库中删除一个或多个同步用户。

权限 必须拥有 DBA 权限。要求可以对发布中涉及的所有表进行独占访问。

副作用 所有与用户关联的预订也将删除。

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Anywhere 8.0 版。

示例 从数据库中删除 MobiLink 用户 *ml_user1*。

```
DROP SYNCHRONIZATION USER ml_user1
```

DROP VARIABLE 语句

说明

此语句用于消除 SQL 变量。

语法

DROP VARIABLE *identifier*

用法

DROP VARIABLE 语句删除以前用 CREATE VARIABLE 语句创建的 SQL 变量。释放数据库连接时，变量将自动删除。变量通常用于大对象，因此在使用后删除它们或者将它们设置为 NULL 可以释放重要资源（主要是磁盘空间）。

权限

无。

副作用

无。

另请参阅

"CREATE VARIABLE 语句 " 第 417 页

"SET 语句 " 第 592 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

EXECUTE 语句 [ESQL]

说明

此语句用于执行准备的 SQL 语句。

语法 1

EXECUTE *statement*
[**USING** { *hostvar-list* | **DESCRIPTOR** *sqlda-name* }]
[**INTO** { *into-hostvar-list* | **DESCRIPTOR** *into-sqlda-name* }]
[**ARRAY :***integer*]

statement : { *identifier* | *hostvar* | *string* }

sqlda-name : *identifier*

into-sqlda-name : *identifier*

语法 2

EXECUTE IMMEDIATE *statement*

statement : { *string* | *hostvar* }

参数

USING 子句 SELECT 语句或 CALL 语句的结果放在变量列表的变量中或放在指定的 SQLDA 所描述的程序数据区中。 OUTPUT （选择列表或参数）和主机变量列表或 SQLDA 描述符数组之间是一对一的对应关系。

INTO 子句 如果 EXECUTE INTO 与 INSERT 语句一起使用，则插入行在第二个描述符中返回。例如，当使用自动增量主键或生成主键值的 BEFORE INSERT 触发器时， EXECUTE 语句提供一种机制，立即重新读取该行并确定分配给该行的主键值。对自动增量键使用 **@@identity** 也可以获得同样的效果。

ARRAY 子句 可选的 ARRAY 子句与准备的 INSERT 语句一起使用可允许宽插入，即一次插入多行，这样可提高性能。整数值是插入的行数。 SQLDA 中，每个条目必须有一个变量 （行数 * 列数）。第一行放在 SQLDA 变量 0 到 (columns per row)-1 中，依此类推。

用法

EXECUTE 语句可用于任何可以准备的 SQL 语句。游标用于从数据库中返回许多行的 SELECT 语句或 CALL 语句 （请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > 在嵌入式 SQL 中使用游标 "）。

成功执行 INSERT、 UPDATE 或 DELETE 语句后， SQLCA (SQLCOUNT) 中的 *sqlerrd[2]* 字段由上述操作所影响的行数填充。

语法 1 执行指定的动态语句，这是先前已准备的语句。如果动态语句包含为请求（绑定变量）提供信息的主机变量占位符，则 *sqlda-name* 必须指定一个指向 SQLDA（它包含的描述符足够语句中出现的所有绑定变量使用）的 C 指针变量，或者必须在 *hostvar -list* 中提供绑定变量。

语法 2 PREPARE 和 EXECUTE 语句的短格式，不包含绑定变量或输出。立即执行字符串或主机变量中包含的 SQL 语句，并且在完成后将其删除。

权限

检查所执行的语句的权限。

副作用

无。

另请参阅

"EXECUTE 语句 [SP]" 第 467 页

"PREPARE 语句 [ESQL]" 第 553 页

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

标准和兼容性

- **SQL/92** 中间级特性。
- **SQL/99** 核心 SQL 外部的特性。
- **Sybase** 在 Open Client/Open Server 中受支持。

示例

执行 DELETE。

```
EXEC SQL EXECUTE IMMEDIATE
'DELETE FROM employee WHERE emp_id = 105';
```

执行准备的 DELETE 语句。

```
EXEC SQL PREPARE del_stmt FROM
'DELETE FROM employee WHERE emp_id = :a';
EXEC SQL EXECUTE del_stmt USING :employee_number;
```

执行准备的查询。

```
EXEC SQL PREPARE sel1 FROM
'SELECT emp_lname FROM employee WHERE emp_id = :a';
EXEC SQL EXECUTE sel1 USING :employee_number INTO
:emp_lname;
```

EXECUTE 语句 [SP]

说明 此语句用于使动态构造的语句能够从过程内执行。

语法 1 **EXECUTE IMMEDIATE** [**WITH QUOTES**] *string-expression*

语法 2 **EXECUTE (** *string-expression* **)**

参数 **WITH QUOTES** 当指定 WITH QUOTES 时，字符串表达式中的任何双引号都被假定为用于界定标识符。当没有指定 WITH QUOTES 时，字符串表达式中的双引号处理取决于 QUOTED_IDENTIFIER 选项的当前设置。如果使用传递到存储过程中的对象名构造要执行的语句，但该名称可能需要双引号并且当 QUOTED_IDENTIFIER 设置为 OFF 时可能调用该过程，这时 WITH QUOTES 很有用。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > QUOTED_IDENTIFIER 选项 [compatibility]"。

用法 EXECUTE 语句扩展了可以从过程和触发器内执行的语句的范围。它使您可以执行动态准备的语句，比如用传递到过程中的参数构造的语句。

此语句中的字符串必须用单引号括起来，并且此语句必须位于一行上。

EXECUTE IMMEDIATE 语句不支持返回结果集的语句和查询。

EXECUTE IMMEDIATE 执行的语句只能引用全局变量。

在 Transact-SQL 存储过程和触发器的内部只能使用语法 2。

权限 无。此语句用过程所有者的权限执行，而不是用调用过程的用户的权限执行。

副作用 无。但是，如果此语句是副作用为自动提交的数据定义语句，则会提交。

有关在过程中使用 EXECUTE IMMEDIATE 语句的更多信息，请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 在过程中使用 EXECUTE IMMEDIATE 语句 "。

另请参阅 "CREATE PROCEDURE 语句 " 第 347 页

"BEGIN 语句 " 第 288 页

"EXECUTE 语句 [ESQL]" 第 465 页

标准和兼容性

- **SQL/92** 中间级特性。
- **SQL/99** 核心 SQL 外部的 SQL/ 基础特性。
- **Sybase** 在 Open Client/Open Server 中受支持。

示例

下面的过程创建一个表，其中表名作为过程的参数提供。 EXECUTE IMMEDIATE 语句必须全部在一行上。

```
CREATE PROCEDURE CreateTableProc(
                  IN tablename char(30)
                  )
BEGIN
   EXECUTE IMMEDIATE 'CREATE TABLE ' || tablename ||
' ( column1 INT PRIMARY KEY)'
END
```

调用过程并创建名为 *mytable* 的表：

```
CALL CreateTableProc( 'mytable' )
```

EXECUTE 语句 [T-SQL]

说明

语法 1 用于激活过程，它是替代 CALL 语句的方法并且与 Adaptive Server Enterprise 兼容。*注意：*也可以*在* Transact-SQL 存储过程和触发器内执行语句。有关更多信息，请参阅 "EXECUTE 语句 [SP]" 第 467 页。语法 2 用于在 T-SQL 中执行准备的 SQL 语句。

语法 1

EXECUTE [*@return_status* =] [*creator*.]*procedure_name* [*argument*, …]

argument :
[*@parameter-name* =] *expression*
| [*@parameter-name* =] *@variable* [*output*]

语法 2

EXECUTE (*string-expression* **)**

用法

语法 1 执行存储过程，选择性地提供过程参数以及检索输出值和返回状态信息。

实现 EXECUTE 语句是为了与 Transact-SQL 兼容，但此语句可以在 Transact-SQL 或 Watcom-SQL 批处理和过程中使用。

使用语法 2，可以在 Transact-SQL 存储过程和触发器内执行语句。EXECUTE 语句扩展了可以从过程和触发器内执行的语句的范围。它使您可以执行动态准备的语句，比如用传递到过程中的参数构造的语句。此语句中的字符串必须用单引号括起来，并且此语句必须位于一行上。

权限

必须是过程的所有者，有过程的 EXECUTE 权限或者有 DBA 权限。

副作用

无。

另请参阅

"CALL 语句 " 第 294 页

"EXECUTE 语句 [ESQL]" 第 465 页

示例

下面的过程阐释了语法 1。

```
CREATE PROCEDURE p1( @var INTEGER = 54 )
AS
PRINT 'on input @var = %1!', @var
DECLARE @intvar integer
SELECT @intvar=123
SELECT @var=@intvar
PRINT 'on exit @var = %1!', @var
```

下面的语句执行过程，为参数提供输入值 23。如果是从 Open Client 或 JDBC 应用程序连接，则 PRINT 消息显示在客户端窗口上。如果是从 ODBC 或嵌入式 SQL 应用程序连接，则消息显示在数据库服务器窗口上。

```
EXECUTE p1 23
```

以下是另一种执行过程的方法，当有几个参数时该方法很有用。

```
EXECUTE p1 @var = 23
```

下面的语句执行过程，使用参数的缺省值

```
EXECUTE p1
```

下面的语句执行过程，并在用于检查返回状态的变量中存储返回值。

```
EXECUTE @status = p1 23
```

EXIT 语句 [Interactive SQL]

说明

此语句用于退出 Interactive SQL。

语法

{ **EXIT** | **QUIT** | **BYE** } [*return-code*]

return-code:number
| *hostvar*

用法

此语句关闭与数据库的连接，然后关闭 Interactive SQL 环境。关闭数据库连接前，如果 COMMIT_ON_EXIT 选项设置为 ON，则 Interactive SQL 自动执行 COMMIT 语句。如果该选项设置为 OFF，则 Interactive SQL 改为执行 ROLLBACK。缺省情况下， COMMIT_ON_EXIT 选项设置为 ON。

在批处理文件中可以使用可选的返回代码来指示 Interactive SQL 命令文件中的命令是成功还是失败。缺省返回代码是 0。

权限

无。

副作用

如果选项 COMMIT_ON_EXIT 设置为 ON （缺省），则此语句自动执行提交；否则执行回退。

在 Windows 操作系统中，提供的可选返回值是 ERRORLEVEL。

另请参阅

"SET OPTION 语句 " 第 601 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 在 Adaptive Server Enterprise 中不可用。

示例

如果表 *T* 有行，下面的示例将把 Interactive SQL 返回值设置为 1，如果 *T* 中没有行，则 设置为 0。

```
CREATE VARIABLE rowCount INT;
CREATE VARIABLE retcode INT;
SELECT COUNT(*) INTO rowCount FROM T;
IF( rowCount > 0 ) THEN
    SET retcode = 1;
ELSE
    SET retcode = 0;
END IF;
EXIT retcode;
```

下面的示例是无效的，因为 EXIST 语句只是一个 Interactive SQL 语句。它不能包含在 IF 语句或任何其它 SQL 块语句内。

```
// this example shows incorrect code
CREATE VARIABLE rowCount INT;
SELECT COUNT(*) INTO rowCount FROM T;
IF( rowCount > 0 ) THEN
    EXIT 1;    //  <-- not allowed
ELSE
    EXIT 0;    //  <-- not allowed
END IF;
```

下面的 Windows 批处理文件将在命令提示符下显示 Error = 1。

```
dbisql -c "dsn=ASA 8.0 Sample" EXIT 1
if errorlevel 1 echo "Errorlevel is 1"
```

EXPLAIN 语句 [ESQL]

说明

此语句用于检索用于特定游标的优化策略的文本说明。

语法

EXPLAIN PLAN FOR CURSOR *cursor-name*
{ **INTO** *hostvar* | **USING DESCRIPTOR** *sqlda-name* }

cursor-name : *identifier* or *hostvar*

sqlda-name : *identifier*

用法

EXPLAIN 语句检索指定游标的优化策略的文本表示形式。游标必须事先声明并且处于打开状态。

hostvar 或 *sqlda-name* 变量必须是字符串类型。优化字符串指定搜索表的顺序，还指定搜索所使用的索引 （如果有的话）。

根据查询的不同，该字符串可能很长，并且具有以下格式：

```
table (index), table (index), ...
```

如果为表提供了相关名，将出现相关名而不是表名。表名在列表中出现的顺序就是数据库服务器访问它们的顺序。每个表名后是一个括在括号内的索引名。这是将用于访问该表的索引。如果不使用任何索引 （将按顺序扫描表），将显示字母 "seq" 代替索引名。如果特定的 SQL SELECT 语句涉及子查询，将用冒号 (:) 分隔每个子查询的优化字符串。这些子查询部分将按照数据库服务器执行查询的顺序出现。

成功执行 EXPLAIN 语句后， SQLCA (SQLIOESTIMATE) 的 **sqlerrd[3]** 字段将由读取查询的所有行所需的输入 / 输出操作数估计值填充。

在 "ASA SQL 用户指南 > 监控和提高性能 " 中可以找到有关很多优化字符串示例的讨论。

权限

必须已经打开指定的游标。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"PREPARE 语句 [ESQL]" 第 553 页

"FETCH 语句 [ESQL] [SP]" 第 475 页

"CLOSE 语句 [ESQL] [SP]" 第 301 页

"OPEN 语句 [ESQL] [SP]" 第 543 页

"ASA 编程指南 > 嵌入式 SQL 编程 > 在嵌入式 SQL 中使用游标 "

"ASA 编程指南 > 嵌入式 SQL 编程 > SQL 通信区域 (SQLCA)"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的示例阐释了 EXPLAIN 的用法：

```
EXEC SQL BEGIN DECLARE SECTION;
char plan[300];
EXEC SQL END DECLARE SECTION;
EXEC SQL DECLARE employee_cursor CURSOR FOR
   SELECT emp_id, emp_lname
   FROM employee
   WHERE emp_lname like :pattern;
EXEC SQL OPEN employee_cursor;
EXEC SQL EXPLAIN PLAN FOR CURSOR employee_cursor INTO
:plan;
printf( "Optimization Strategy: '%s'.n", plan );
```

计划变量包含以下字符串：

```
'employee <seq>'
```

FETCH 语句 [ESQL] [SP]

说明

此语句用于重新定位游标，然后从游标中获取数据。

语法

FETCH *cursor-position cursor-name*
[**INTO** { *hostvar-list* | *variable-list* }
| **USING DESCRIPTOR** *sqlda-name*]
[**PURGE**]
[**BLOCK** *n*]
[**FOR UPDATE**]
[**ARRAY** *fetch-count*]
INTO *variable-list* [**FOR UPDATE**]

cursor-position :
NEXT | **PRIOR** | **FIRST** | **LAST**
| { **ABSOLUTE** | **RELATIVE** } *row-count*

row-count : *number* or *hostvar*

cursor-name : *identifier* or *hostvar*

hostvar-list : may contain indicator variables

variable-list : stored procedure variables

sqlda-name : *identifier*

fetch-count : *integer* or *hostvar*

参数

INTO　INTO 子句是可选的。如果未指定该子句，则 FETCH 语句仅定位游标。 *hostvar-list* 仅供嵌入式 SQL 使用。

cursor position　一个可选定位参数，它允许在读取行之前移动游标。如果读取操作中包括定位参数，而位置在允许的游标位置外，则会发出 SQLE_NOTFOUND 警告，并且 SQLCOUNT 字段指示从有效位置的偏移量。

OPEN 语句最初将游标定位在第一行的前面。

- **NEXT**　NEXT 是缺省定位参数，它使游标在读取行之前向前移动一行。
- **PRIOR**　使游标在读取前向后移动一行。

- **RELATIVE** RELATIVE 定位用于在读取前将游标沿任一方向移动指定的行数。正数表示向前移动，负数表示向后移动。因此，NEXT 等效于 RELATIVE 1， PRIOR 等效于 RELATIVE -1。RELATIVE 0 检索的行与该游标的上一读取语句检索的行相同。

- **ABSOLUTE** ABSOLUTE 定位参数用于转到具体某行。 0 表示第一行前面的位置 （请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 在过程和触发器中使用游标 "）。

  1 表示第一行，依此类推。负数用于指定游标末尾的绝对位置。 -1 表示游标的最后一行。

- **FIRST** ABSOLUTE 1 的缩写形式。

- **LAST** ABSOLUTE -1 的缩写形式。

游标定位问题
插入 DYNAMIC SCROLL 游标并对其进行某些更新会导致与游标定位有关的问题。除非 SELECT 语句中有 ORDER BY 子句，否则数据库服务器不将插入的行放在游标内可预知的位置。在某些情况下，插入的行直到关闭并重新打开游标后才显示出来。

如果在打开游标时必须创建一个临时表，则会发生这种情况 （请参阅 "ASA SQL 用户指南 > 监控和提高性能 > 在查询处理中使用工作表 "）。

UPDATE 语句可能使某行在游标中移动。如果游标有使用现有索引的 ORDER BY （未创建临时表），则会发生这种情况。

BLOCK 子句 客户端应用程序一次可读取多行。这类读取称为块读取、预读或多行读取。第一个读取导致从服务器发回若干行。客户端将这些行放入缓冲区，后面的读取从这些缓冲区中检索，而不对服务器发出新的请求。

BLOCK 子句仅供嵌入式 SQL 使用。它提示客户端和服务器：应用程序可读取多少行。特殊值 0 表示请求将发送到服务器，并且只返回一行（没有行块）。

如果未指定 BLOCK 子句，则使用 OPEN 上指定的值。有关更多信息，请参阅 "OPEN 语句 [ESQL] [SP]" 第 543 页。

FETCH RELATIVE 0 总是重新读取行。

PURGE 子句　PURGE 子句仅供嵌入式 SQL 使用。它使客户端刷新其所有行的缓冲区，然后向服务器发送读取请求。请注意，读取请求可能返回行块。

FOR UPDATE 子句　FOR UPDATE 子句指示读取的行随后将由 UPDATE WHERE CURRENT OF CURSOR 语句更新。该子句使数据库服务器在行上放置一个写锁。该锁一直保持到当前事务结束时为止。请参阅 "ASA SQL 用户指南 > 使用事务和隔离级别 > 锁定的工作方式 "。

ARRAY 子句　ARRAY 子句仅供嵌入式 SQL 使用。该子句允许所谓的大范围读取，即同时检索多行并会提高性能。

若要在嵌入式 SQL 中使用大范围读取，请按如下所示在代码中包括读取语句：

```
EXEC SQL FETCH . . . ARRAY nnn
```

其中， **ARRAY** *nnn* 是 FETCH 语句的最后一项。读取计数 *nnn* 可以是一个主机变量。 SQLDA 必须包含 **nnn * (columns per row)** 变量。第一行放在 SQLDA 变量 0 到 **(columns per row)**-1，依此类推。

有关如何使用大范围读取的详细示例，请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > 一次读取多个行 " 一节。

用法

FETCH 语句从指定的游标中检索一行。游标在此前必须已打开。

嵌入式 SQL 使用　DECLARE CURSOR 语句在 C 源代码中必须出现在 FETCH 语句的前面，而 OPEN 语句必须在 FETCH 语句之前执行。如果主机变量用于游标名，则实际上是 DECLARE 语句生成代码，因此 DECLARE 语句必须在 FETCH 语句之前执行。

服务器在 SQLCOUNT 中返回读取的记录数，并且总是返回大于零的 SQLCOUNT，除非出现错误或警告。如果在 SQLCOUNT 为零时没有出现错误情况，则表示读取了一个有效行。

如果在读取时返回 SQLSTATE_NOTFOUND 警告，则 SQLCA (SQLCOUNT) 的 *sqlerrd[2]* 字段中包含所尝试的读取超出所允许的游标位置的行数。如果未找到行但位置有效，则值为 0。例如，当定位在游

标的最后一行时执行 FETCH RELATIVE 1。如果所尝试的读取超出了游标的末尾，则为正值；如果所尝试的读取位于游标开头的前面，则为负值。

成功执行了读取语句后， SQLCA (SQLIOCOUNT) 的 *sqlerrd[1]* 字段递增执行读取所需的输入 / 输出操作数。该字段实际上在每个数据库语句执行时都递增。

单行读取 SELECT 语句结果中的一行被放入变量表的变量中。选择表和主机变量表之间是一对一的对应关系。

多行读取 SELECT 语句结果中的一行或多行被放入 *variable-list* 的变量中或 *sqlda-name* 所描述的程序数据区中。在任何一种情况下， *select-list* 和 *hostvar-list* 或 *sqlda-name* 描述符数组之间都是一对一的对应关系。

权限

游标必须是打开的，并且用户必须对游标声明所引用的表具有 SELECT 权限。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"PREPARE 语句 [ESQL]" 第 553 页

"OPEN 语句 [ESQL] [SP]" 第 543 页

"ASA 编程指南 > 嵌入式 SQL 编程 > 在嵌入式 SQL 中使用游标 "

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 在过程和触发器中使用游标 "

PowerScript 参考 中的 FETCH

标准和兼容性

- **SQL/92** 初级特性。在过程中使用的是持久存储模块特性。
- **SQL/99** 核心特性。在过程中使用的是持久存储模块特性。
- **Sybase** 受 Adaptive Server Enterprise 支持。

示例

以下是嵌入式 SQL 的示例。

```
EXEC SQL DECLARE cur_employee CURSOR FOR
SELECT emp_id, emp_lname FROM employee;
EXEC SQL OPEN cur_employee;
EXEC SQL FETCH cur_employee
INTO :emp_number, :emp_name:indicator;
```

以下是过程示例：

```
BEGIN
   DECLARE cur_employee CURSOR FOR
      SELECT emp_lname
      FROM employee;
   DECLARE name CHAR(40);
   OPEN cur_employee;
   LOOP
      FETCH NEXT cur_employee into name;
       ...
   END LOOP
   CLOSE cur_employee;
END
```

FOR 语句

说明

此语句用于对游标中的每一行重复执行一次语句列表。

语法

```
[ statement-label : ]
    FOR for-loop-name AS cursor-name
      CURSOR FOR statement
      [ FOR UPDATE | FOR READ ONLY ]
        DO statement-list
    END FOR [ statement-label ]
```

用法

FOR 语句是控制语句，它允许对游标中的每一行执行一次 SQL 语句列表。 FOR 语句等效于这样一个复合语句：游标有一个 DECLARE，游标结果集中的每一列有一个变量 DECLARE，后面跟一个循环，它将游标中的一行读入局部变量，并对游标中的每一行执行一次 *statement-list*。

每个局部变量的名称和数据类型都是从游标中使用的 *statement* 派生的。在 SELECT 语句中，数据类型是选择列表中表达式的数据类型。名称是选择列表项的别名 （如果存在这样的别名），否则名称是列的名称。任何不是简单列引用的选择列表项都必须有别名。在 CALL 语句中，名称和数据类型从过程定义的 RESULT 子句中获得。

LEAVE 语句可用来指示接下去执行 END FOR 之后第一条语句。如果指定结尾 *statement-label*，它必须与起始 *statement-label* 匹配。

权限

无。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"FETCH 语句 [ESQL] [SP]" 第 475 页

"LEAVE 语句 " 第 527 页

"LOOP 语句 " 第 539 页

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。

- **Sybase**　不受 Adaptive Server Enterprise 支持。

示例　下面的代码段阐释了 FOR 循环的用法。

```
FOR names AS curs CURSOR FOR
SELECT emp_lname
FROM employee
DO
   CALL search_for_name( emp_lname );
END FOR;
```

FORWARD TO 语句

说明 此语句用于将本地语法 SQL 语句发送到远程服务器。

语法 1 **FORWARD TO** *server-name SQL-statement*

语法 2 **FORWARD TO** [*server-name*]

用法 FORWARD TO 语句使用户能够指定需要直通连接的服务器。此语句有两种用法：

- **语法 1** 将单条语句发送到远程服务器。
- **语法 2** 将 Adaptive Server Anywhere 置于将一系列语句发送到远程服务器的直通模式。所有后续语句都直接传递到远程服务器。若要关闭直通模式，在发出 FORWARD TO 时不要指定 *server-name*。

  如果在直通模式下遇到来自远程服务器的错误，仍需发出 FORWARD TO 语句关闭直通模式。

当以用户的名义与 server-name 建立连接时，服务器使用：

- 用 CREATE EXTERNLOGIN 设置的远程登录别名，或者
- 如果未设置远程登录别名，则使用用于与 Adaptive Server Anywhere 通信的名称和口令

如果无法与指定的服务器建立连接，在返回给用户的消息中会指出原因。

在将语句传递给请求的服务器后，所有结果都转换成客户端程序可以识别的形式。

server-name 远程服务器的名称。

SQL-statement 远程服务器的本地 SQL 语法中的命令。该命令或命令组用大括号 ({}) 括起来。

权限 无

副作用

在 FORWARD TO 会话的持续时间内，远程连接设置为 AUTOCOMMIT（非链接）模式。在 FORWARD TO 语句之前未完成的任何工作都自动提交。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　受 Open Client/Open Server 支持。

示例

下面的示例显示远程服务器 *ase_prod* 的直通会话：

```
FORWARD TO aseprod
SELECT * from titles
SELECT * from authors
FORWARD TO
```

FROM 子句

说明

此子句用于指定 SELECT、 UPDATE 或 DELETE 语句中涉及的数据库表或视图。

语法

FROM *table_expression*, …

table_expression:
  table
| *view*
| *derived table*
| *joined table*
| **(** *table_expression*, … **)**

table or *view*:
[*userid***.**] *table-or-view-name* [[**AS**] *correlation-name*] [**WITH (** *table-hint* **)**]

derived table:
(*select-statement* **)** [**AS**] *correlation-name* [**(** *column-name*, … **)**]

joined table:
table_expression *join_operator* *table_expression* [**ON** *join_condition*]

join_operator: [**KEY** | **NATURAL**] [*join_type*] **JOIN**
| **CROSS JOIN**

join_type:
  INNER
| **LEFT** [**OUTER**]
| **RIGHT** [**OUTER**]
| **FULL** [**OUTER**]

table-hint:
  NOLOCK
| **READUNCOMMITTED**
| **READCOMMITTED**
| **REPEATABLEREAD**
| **HOLDLOCK**
| **SERIALIZABLE**
| **FASTFIRSTROW**

用法

SELECT、 UPDATE 和 DELETE 语句需要用一个表列表来指定该语句要使用的表。

视图和派生表
虽然这里的说明针对的是表，但它同样适用于视图和派生表，除非另外说明。

FROM 表列表创建由所有指定表中的所有列组成的结果集。组件表中行的所有组合最初都在结果集中，但 JOIN 条件和 / 或 WHERE 条件通常会减少组合数。

在 CROSS JOIN 中不能使用 ON 短语。

其他用户拥有的表可以通过指定用户 ID 来限定。缺省情况下，当不指定用户 ID 时，找到的是由当前用户所属的组拥有的表 （请参阅 "ASA 数据库管理指南 > 管理用户 ID 和权限 > 引用组拥有的表 "）。

相关名是在查询的 FROM 子句中使用的表或视图的名称，此名称可以是它的原名称，也可以是在 FROM 子句中定义的别名。

如果表表达式中的同一个表使用了两次相同的相关名，该表按仅列出一次处理。例如，在

```
SELECT *
FROM sales_order
KEY JOIN sales_order_items,
sales_order
KEY JOIN employee
```

中， **sales_order** 表的两个实例按一个实例处理，因此等效于：

```
SELECT *
FROM sales_order
KEY JOIN sales_order_items
KEY JOIN employee
```

而

```
SELECT *
FROM Person HUSBAND, Person WIFE
```

按 Person 表的两个实例处理，有两个不同的相关名 HUSBAND 和 WIFE。

在 FROM 子句中可以使用 SELECT 语句代替表名或视图名。这样，不必创建视图即可在组中使用组或与组连接。用此方法创建的表就是派生表。

WITH 表提示 允许您指定 Adaptive Server Anywhere 的仅用于此表和仅用于此语句的行为。您可以使用 WITH *table-hint* 来更改 Adaptive Server Anywhere 的行为而不用更改隔离级别或者设置数据库或连接选项。表提示仅可用于基表和临时表。

注意
WITH table-hint 是一项高级功能，只应在需要时使用，或者仅由有经验的数据库管理员使用。另外，并非在所有情况下都应用此设置。

- **隔离级别提示** 下面的表提示可用于指定表的隔离级别设置。这些提示指定仅用于此表和仅用于此语句的锁定方法。

  这些表提示设置以下隔离级别：

表提示	隔离级别
NOLOCK	0
READUNCOMMITTED	0
READCOMMITTED	1
REPEATABLEREAD	2
HOLDLOCK	3
SERIALIZABLE	3

- **优化提示** FASTFIRSTROW 表提示允许您在不将 OPTIMIZATION_GOAL 选项设置为 **first-row** 的情况下为查询设置优化目标。当您使用 FASTFIRSTROW 时， Adaptive Server Anywhere 选择一种能够缩短读取查询结果第一行的时间的访问计划。

  有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > OPTIMIZATION_GOAL 选项 [database]"。

权限

无。

副作用

无。

另请参阅

"DELETE 语句 " 第 435 页

"SELECT 语句 " 第 587 页

"UPDATE 语句 " 第 642 页

"ASA SQL 用户指南 > 连接：从多个表检索数据 "

标准和兼容性

- **SQL/92** 初级特性。 FROM 子句的复杂性意味着应根据标准检查各子句。
- **SQL/99** 核心特性，但 KEY JOIN （它是供应商扩展）和 FULL OUTER JOIN 与 NATURAL JOIN （它们是核心 SQL 外部的 SQL/基础特性）除外。 FROM 子句的复杂性意味着应根据标准检查各子句。
- **Sybase** 在 12 版之前的 Adaptive Server Enterprise 中不支持 ON 短语。在早期版本中，必须使用 WHERE 子句建立连接。

示例

以下是有效的 FROM 子句：

```
...
FROM employee
...
...
FROM employee NATURAL JOIN department
...
...
FROM customer
KEY JOIN sales_order
KEY JOIN sales_order_items
KEY JOIN product
...
```

下面的查询阐释如何在查询中使用派生表：

```
SELECT lname, fname, number_of_orders
FROM customer JOIN
     ( SELECT cust_id, count(*)
       FROM sales_order
        GROUP BY cust_id )
     AS sales_order_counts ( cust_id,
                             number_of_orders )
ON ( customer.id = sales_order_counts.cust_id )
WHERE number_of_orders > 3
```

GET DATA 语句 [ESQL]

说明

此语句用于获取游标当前行的某列的字符串或二进制数据。GET DATA 通常用于读取 LONG BINARY 或 LONG VARCHAR 字段。请参阅 "SET 语句 " 第 592 页。

语法

GET DATA *cursor-name*
　　COLUMN *column-num*
　　OFFSET *start-offset*
　　[**WITH TEXTPTR**]
　　USING DESCRIPTOR *sqlda-name* | **INTO** *hostvar* [, …]

cursor-name :　*identifier*, or *hostvar*

column-num :　*integer* or *hostvar*

start-offset :　*integer* or *hostvar*

sqlda-name :　*identifier*

参数

COLUMN 子句　*column-num* 的值从 1 开始，标识要读取其数据的列。该列必须是字符串或二进制类型。

OFFSET 子句　*start-offset* 指示在字段值中跳过的字节数。通常情况下，该数字是先前已读取的字节数。在此 GET DATA 语句上读取的字节数由目标主机变量的长度决定。

目标主机变量的指示符值属短整型，因此不能总是包含截断的字节数。然而，如果字段包含 NULL 值，则该指示符值包含负值；如果值被截断，则包含正值 （不一定是截断的字节数）；如果非 NULL 值没有截断，则包含零。

同样，如果 LONGVARCHAR 或 LONGVARCHAR 主机变量用于的偏移量大于零，则 untrunc_len 字段不会准确地指示截断前的大小。

WITH TEXTPTR 子句　如果给出 WITH TEXTPTR 子句，则文本指针检索到另一个主机变量中或 SQLDA 的第二个字段中。该文本指针可用于 Transact-SQL READ TEXT 和 WRITE TEXT 语句。该文本指针是一个 16 位的二进制值，可按如下方式声明：

```
DECL_BINARY( 16 ) textptr_var;
```

WITH TEXTPTR 只能用于长数据类型 （LONG BINARY、 LONG VARCHAR、 TEXT 和 IMAGE）。如果尝试将其用于其它数据类型，则会返回错误 INVALID_TEXTPTR_VALUE。

数据的总长度在 SQLCA 结构的 SQLCOUNT 字段中返回。

用法

从当前游标位置处的行中获取一段列值。

权限

游标必须是打开的，并且必须用 FETCH 定位在某行上。

副作用

无。

另请参阅

"FETCH 语句 [ESQL] [SP]" 第 475 页

"READTEXT 语句 [T-SQL]" 第 564 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Open Client/Open Server 支持。替代方法是 Transact-SQL READTEXT 语句。

示例

下面的示例使用 GET DATA 读取**二进制大对象** （通常称为 **blob**）。

```
EXEC SQL BEGIN DECLARE SECTION;
DECL_BINARY(1000) piece;
short ind;
EXEC SQL END DECLARE SECTION;
int size;
/* Open a cursor on a long varchar field */
EXEC SQL DECLARE big_cursor CURSOR FOR
SELECT long_data FROM some_table
WHERE key_id = 2;
EXEC SQL OPEN big_cursor;
EXEC SQL FETCH big_cursor INTO :piece;
for( offset = 0; ; offset += piece.len ) {
   EXEC SQL GET DATA big_cursor COLUMN 1
   OFFSET :offset INTO :piece:ind;
   /* Done if the NULL value */
   if( ind < 0 ) break;
```

```
   write_out_piece( piece );
   /* Done when the piece was not truncated */
   if( ind == 0 ) break;
}
EXEC SQL CLOSE big_cursor;
```

GET DESCRIPTOR 语句 [ESQL]

说明

此语句用于检索描述符区内某变量的信息或检索它的值。

语法

GET DESCRIPTOR *descriptor-name*
{ *hostvar* = **COUNT** | **VALUE** { *integer* | *hostvar* } *assignment* [, …] }

assignment :
hostvar = **TYPE** | **LENGTH** | **PRECISION** | **SCALE** | **DATA** | **INDICATOR** | **NAME** | **NULLABLE** | **RETURNED_LENGTH**

用法

GET DESCRIPTOR 语句用于检索描述符区内某变量的信息或检索它的值。

{ *integer* | *hostvar* } 值指定描述符区内要检索其信息的变量。在执行 GET … DATA 时会执行类型检查，以确保主机变量和描述符变量的数据类型相同。

如果出现错误，则在 SQLCA 中返回该错误。

权限

无。

副作用

无。

另请参阅

"ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页

"DEALLOCATE DESCRIPTOR 语句 [ESQL]" 第 423 页

"SET DESCRIPTOR 语句 [ESQL]" 第 599 页

"ASA 编程指南 > 嵌入式 SQL 编程 > SQL 描述符区域 (SQLDA)"

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** 受 Open Client/Open Server 支持。

示例

下面的示例返回在 sqlda 中的位置为 col_num 的列的类型。

```
int get_type( SQLDA *sqlda, int col_num )
{
    EXEC SQL BEGIN DECLARE SECTION;
    int ret_type;
    int col = col_num;
    EXEC SQL END DECLARE SECTION;
EXEC SQL GET DESCRIPTOR sqlda VALUE :col :ret_type =
TYPE;
    return( ret_type );
}
```

有关详细示例，请参阅 ""ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页 "。

GET OPTION 语句 [ESQL]

说明

此语句用于获取选项的当前设置。建议改用 connection_property 函数。

语法

GET OPTION [*userid*.]*option-name*
[**INTO** *hostvar*]
[**USING DESCRIPTOR** *sqlda-name*]

userid : *identifier, string,* or *hostvar*

option-name : *identifier, string,* or *hostvar*

hostvar : indicator variable allowed

sqlda-name : *identifier*

用法

提供 GET OPTION 语句是为了与软件的早期版本兼容。建议使用 **connection_property** 系统函数获取选项值。

GET OPTION 语句为用户 *userid* 或连接的用户 （如果未指定 *userid*）获取选项 *option-name* 的选项设置。这是用户的个人设置，如果连接的用户没有设置，则是 **PUBLIC** 设置。如果指定的选项是数据库选项，并且用户有该选项的临时设置，则检索临时设置。

如果 *option-name* 不存在，则 GET OPTION 返回警告 SQLE_NOTFOUND。

权限

必须是 " 无 "。

副作用

无。

另请参阅

"SET OPTION 语句 " 第 601 页

" 系统和分类存储过程 " 第 755 页

"CONNECTION_PROPERTY 函数 [系统]" 第 131 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例　　下面的语句阐释了 GET OPTION 的用法。

```
EXEC SQL GET OPTION 'date_format' INTO :datefmt;
```

GOTO 语句 [T-SQL]

说明

此语句用于分支到标签语句。

语法

label : **GOTO** *label*

用法

Transact-SQL 过程、触发器或批处理中的任何语句都可以带标签。标签名是一个有效标识符后跟一个冒号。在 GOTO 语句中不使用冒号。

权限

无。

副作用

无。

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** Adaptive Server Enterprise 支持 GOTO 语句。

示例

下面的 Transact-SQL 批处理在服务器窗口上输出四次消息 "yes"：

```
declare @count smallint
select @count = 1
restart:
   print 'yes'
   select @count = @count + 1
   while @count <=4
    goto restart
```

GRANT 语句

说明

此语句用于创建新用户 ID、对特定用户授予或拒绝权限以及创建或更改口令。

语法 1

GRANT CONNECT TO *userid*, …
[**AT** *starting-id*]
IDENTIFIED BY *password*, …

语法 2

GRANT {
DBA,
GROUP,
MEMBERSHIP IN GROUP *userid*, …,
[**RESOURCE** | **ALL**]
}
TO *userid*, …

语法 3

GRANT {
ALL [**PRIVILEGES**],
ALTER,
DELETE,
INSERT,
REFERENCES [**(** *column-name*, … **)**],
SELECT [**(** *column-name*, … **)**],
UPDATE [**(** *column-name*, … **)**],
}
ON [*owner*.]*table-name*
TO *userid*, …
[**WITH GRANT OPTION**]
[**FROM** *userid*]

语法 4

GRANT EXECUTE ON [*owner*.]*procedure-name* **TO** *userid*, …

语法 5

GRANT INTEGRATED LOGIN TO *user_profile_name*, … **AS USER** *userid*

参数

CONNECT TO　创建新用户。 GRANT CONNECT 也可由任何用户用来更改他们自己的口令。若要创建用空字符串作为口令的用户，请键入：

```
GRANT CONNECT TO userid IDENTIFIED BY "
```

若要创建没有口令的用户，请键入：

```
GRANT CONNECT TO userid
```

没有口令的用户不能连接到数据库。如果要创建组，但不希望任何人使用组用户 ID 连接到数据库，则这很有用。口令必须是有效的标识符，详见 "" 标识符 " 第 8 页 " 中的说明。

AT starting-id 此子句不用于通用目的。此子句指定用于列表中第一个用户 ID 的内部数值。

实现此子句主要是供卸载实用程序使用。

DBA " 数据库管理员 " 权限赋予用户执行任何操作的权限。它通常是为组织中负责管理数据库的人员保留的。

GROUP 允许用户拥有成员。

有关更多信息，请参阅 "ASA 数据库管理指南 > 管理用户 ID 和权限 > 管理组 "。

MEMBERSHIP IN GROUP 此子句允许用户从组中继承表权限和引用组创建的表，而不必限定表名。

有关更多信息，请参阅 "ASA 数据库管理指南 > 管理用户 ID 和权限 > 管理组 "。

GRANT 语句的语法 3 用于授予个别表或视图的权限。表权限可分别指定，也可以使用 ALL 同时授予所有六个权限。

RESOURCE 允许用户创建表和视图。在语法 2 中， **ALL** 是与 Sybase Adaptive Server Enterprise 兼容的 RESOURCE 的同义词。

ALL 在语法 3 中，这授予下面概括的所有权限。

ALTER 允许用户用 ALTER TABLE 语句变更指定的表。对视图不允许此权限。

DELETE 允许用户从指定的表或视图中删除行。

INSERT 允许用户在指定的表或视图中插入行。

REFERENCES [(column-name, …**)]** 允许用户为指定的表创建索引和引用该表的外键。如果指定了列名，则用户只能引用那些列。列的 REFERENCES 权限不能授予视图，只能授予表。

INDEX 是 REFERENCES 的同义词。

SELECT [(column-name, …)] 允许用户查看此视图或表中的信息。如果指定了列名，则用户只能查看那些列。列的 SELECT 权限不能授予视图，只能授予表。

UPDATE [(column-name, …)] 允许用户更新此视图或表中的行。如果指定了列名，则用户只能更新那些列。列的 UPDATE 权限不能授予视图，只能授予表。

FROM 如果指定了 **FROM** *userid*，该用户 ID 将在系统表中记录为一个授权者用户 ID。此子句是供 Unload 实用程序 (dbunload) 使用的。不要直接使用或修改此选项。

用法

GRANT 语句用于为个别用户 ID 和组授予数据库权限。它也可用于创建和删除用户和组。

如果指定了 WITH GRANT OPTION，则指定的用户 ID 也被授权向其他用户 ID GRANT （授权）同样的权限。

GRANT 语句的语法 4 用于授予执行过程的权限。

GRANT 语句的语法 5 在一个或多个 Windows 用户配置文件和现有数据库用户 ID 之间创建显式集成登录映射，使成功登录到本地机器的用户不必提供用户 ID 或口令，就可以连接到数据库。

有关集成登录的更多信息，请参阅 "ASA 数据库管理指南 > 连接到数据库 > 使用集成登录 "。

权限

语法 1 或 2 必须满足以下条件之一。

- 用 GRANT CONNECT 更改自己的口令。
- 有 DBA 权限。

如果更改其他用户的口令 （利用 DBA 权限），则其他用户一定不能连接到数据库。

语法 3 如果指定了 FROM 子句，则您必须拥有 DBA 权限。否则，必须符合下列条件之一：

- 拥有表

- 已经用 GRANT OPTION 授予您对表的权限
- 有 DBA 权限

语法 4 必须满足以下条件之一：

- 拥有过程
- 有 DBA 权限

语法 5 必须满足以下条件：

- 有 DBA 权限

副作用 自动提交。

另请参阅 "REVOKE 语句 " 第 576 页

标准和兼容性

- **SQL/92** 语法 3 是初级特性。语法 4 是持久存储模块特性。其它语法是供应商扩展。
- **SQL/99** 语法 3 是核心特性。语法 4 是持久存储模块特性。其它语法是供应商扩展。
- **Sybase** 语法 2 和 3 在 Adaptive Server Enterprise 中受支持。Adaptive Server Enterprise 和 Adaptive Server Anywhere 中的安全模型不同，因此其它语法也不同。

示例 为数据库创建两个新用户。

```
GRANT
CONNECT TO Laurel, Hardy
IDENTIFIED BY Stan, Ollie
```

为用户 Laurel 授予雇员表的权限。

```
GRANT
SELECT, UPDATE ( street )
ON employee
TO Laurel
```

可以在一个语句中授予多个权限。各权限用逗号分隔。

允许用户 Hardy 执行 *Calculate_Report* 过程。

```
GRANT
EXECUTE ON Calculate_Report
TO Hardy
```

GRANT CONSOLIDATE 语句 [SQL Remote]

说明

此语句用于标识 SQL Remote 层次中位于当前数据库正上方的数据库，该数据库将从当前数据库接收消息。

语法

GRANT CONSOLIDATE
TO *userid*, …
TYPE *message-system*, …
ADDRESS *address-string*, …
[**SEND** { **EVERY** | **AT** }*'hh:mm:ss'*]

message-system: **FILE** | **FTP** | **MAPI** | **SMTP** | **VIM**

address: *string*

参数

userid 被授予权限的用户的用户 ID。

message-system SQL Remote 支持的消息系统之一。

address 指定消息系统的地址。

用法

在 SQL Remote 安装中，必须为 SQL Remote 层次中位于当前数据库正上方的数据库授予 CONSOLIDATE 权限。远程数据库中发出 GRANT CONSOLIDATE 以标识它的统一数据库。每个数据库只能有一个具有 CONSOLIDATE 权限的用户 ID：远程数据库不能有一个以上的统一数据库。

统一用户由消息系统标识，它标识向统一用户发送消息和从统一用户接收消息的方法。地址名必须是消息系统的有效地址，用单引号括起来。

对于 FILE 消息类型，地址是 SQLREMOTE 环境变量指向的目录的子目录。

统一数据库需要使用 GRANT CONSOLIDATE 语句来接收消息，但此语句本身并不为统一数据库预订任何数据。若要预订数据，必须为统一用户 ID 创建对当前数据库中的某个发布的预订。在统一数据库中运行数据库抽取实用程序可创建一个已发出适当的 GRANT CONSOLIDATE 语句的远程数据库。

可选的 SEND EVERY 和 SEND AT 子句指定发送消息的频率。该字符串包含各消息之间的时间长度 （对于 SEND EVERY）或一天中发送消息的时间 （对于 SEND AT）。对于 SEND AT，每天发送一次消息。

如果授予用户远程权限时未指定 SEND EVERY 或 SEND AT 子句，则 " 消息代理 " 处理消息然后停止。为了不间断运行 " 消息代理 "，必须确保每个具有 REMOTE 权限的用户都指定了 SEND AT 或 SEND EVERY 频率。

预计在许多远程数据库中，" 消息代理 " 都定期运行，并且统一数据库不指定 SEND 子句。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"GRANT PUBLISH 语句 [SQL Remote]" 第 504 页

"GRANT REMOTE 语句 [SQL Remote]" 第 505 页

"REVOKE CONSOLIDATE 语句 [SQL Remote]" 第 578 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_grant_consolidate 过程 "

示例

```
GRANT CONSOLIDATE TO con_db
TYPE mapi
ADDRESS 'Consolidated Database'
```

GRANT PUBLISH 语句 [SQL Remote]

说明

此语句用于标识当前数据库的发布者。

语法

GRANT PUBLISH TO *userid*

用法

SQL Remote 安装中的每个数据库在发出的消息中由一个用户 ID 标识，该用户 ID 称为**发布者**。 GRANT PUBLISH 语句标识与这些发出的消息关联的发布者用户 ID。

只能有一个用户 ID 有 PUBLISH 权限。有 PUBLISH 权限的用户 ID 由特殊常量 CURRENT PUBLISHER 标识。下面的查询标识当前发布者：

```
SELECT CURRENT PUBLISHER
```

如果没有发布者，则该特殊常量为 NULL。

当前发布者特殊常量可用作列的缺省设置。在复制表时将 CURRENT PUBLISHER 列用作主键的一部分通常很有用，因为这有助于防止由于一个以上站点的更新而导致主键冲突。

若要更改发布者，必须先用 REVOKE PUBLISH 语句删除当前发布者，然后用 GRANT PUBLISH 语句创建新的发布者。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"GRANT PUBLISH 语句 [SQL Remote]" 第 504 页

"GRANT CONSOLIDATE 语句 [SQL Remote]" 第 502 页

"REVOKE PUBLISH 语句 [SQL Remote]" 第 579 页

"CREATE SUBSCRIPTION 语句 [SQL Remote]" 第 367 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_publisher 过程 "

示例

```
GRANT PUBLISH TO publisher_ID
```

GRANT REMOTE 语句 [SQL Remote]

说明

此语句用于标识 SQL Remote 层次中位于当前数据库正下方的数据库，该数据库将从当前数据库接收消息。这些用户称为远程用户。

语法

GRANT REMOTE TO *userid*, …
    TYPE *message-system*, …
    ADDRESS *address-string*, …
    [**SEND** { **EVERY** | **AT** } *send-time*]

参数

userid　被授予权限的用户的用户 ID。

message-system　SQL Remote 支持的消息系统之一。它必须是下面的一个值：

- FILE
- FTP
- MAPI
- SMTP
- VIM

address-string　一个字符串，包含指定的消息系统的有效地址。

send-time　一个字符串，包含 *hh:mm:ss* 格式的时间说明。

用法

在 SQL Remote 安装中，每个从当前数据库接收消息的数据库都必须被授予 REMOTE 权限。

唯一的例外是 SQL Remote 层次中位于当前数据库正上方的数据库，该数据库必须被授予 CONSOLIDATE 权限。

远程用户由消息系统标识，它标识向统一用户发送消息和从统一用户接收消息的方法。地址名必须是消息系统的有效地址，用单引号括起来。

对于 FILE 消息类型，地址是 SQLREMOTE 环境变量指向的目录的子目录。

远程数据库需要使用 GRANT REMOTE 语句来接收消息，但此语句本身并不为远程用户预订任何数据。若要预订数据，必须使用数据库抽取实用程序或 CREATE SUBSCRIPTION 语句，为用户 ID 创建对当前数据库中的某个发布的预订。

可选的 SEND EVERY 和 SEND AT 子句指定发送消息的频率。该字符串包含各消息之间的时间长度 （对于 SEND EVERY）或一天中发送消息的时间 （对于 SEND AT）。对于 SEND AT，每天发送一次消息。

如果授予用户远程权限时未指定 SEND EVERY 或 SEND AT 子句，则 " 消息代理 " 处理消息然后停止。为了不间断运行 " 消息代理 "，必须确保每个具有 REMOTE 权限的用户都指定了 SEND AT 或 SEND EVERY 频率。

预计在许多统一数据库中，" 消息代理 " 都不间断运行，因此所有远程数据库都指定了 SEND 子句。典型安装可能涉及每天向膝上型电脑用户发送消息 (SEND AT) 和每隔一两个小时向远程服务器发送一次消息 (SEND EVERY)。为提高效率，所使用的时间应尽量相同。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"GRANT PUBLISH 语句 [SQL Remote]" 第 504 页

"REVOKE REMOTE 语句 [SQL Remote]" 第 581 页

"GRANT CONSOLIDATE 语句 [SQL Remote]" 第 502 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_grant_remote 过程 "

"SQL Remote 用户指南 > SQL Remote 管理 > 授予和撤消 REMOTE 和 CONSOLIDATE 权限 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

- 下面的语句向用户 **SamS** 授予远程权限，使用 MAPI 电子邮件系统，每隔两小时向地址 **Singer, Samuel** 发送一次消息：

```
GRANT REMOTE TO SamS
TYPE mapi
ADDRESS 'Singer, Samuel'
SEND EVERY '02:00'
```

GRANT REMOTE DBA 语句 [SQL Remote]

说明

此语句用于为用户 ID 提供 DBA 权限，但仅在从 " 消息代理 " 中连接时提供。

语法

GRANT REMOTE DBA
TO *userid*, …
IDENTIFIED BY *password*

用法

REMOTE DBA 权限使 " 消息代理 " 能够完全访问数据库以使任何更改包含在消息中，同时避免与分发 DBA 用户 ID 口令相关的安全问题。

REMOTE DBA 有以下属性。

- 如果不是从 " 消息代理 " 连接，则没有独特的权限。被授予 REMOTE DBA 权限的用户 ID 对同 " 消息代理 " 分离的任何连接没有额外的权限。即使 REMOTE DBA 用户的用户 ID 和口令已广泛分发，也不存在安全问题。只要用户 ID 的权限没有数据库上 CONNECT 以外的权限，就没有人能够使用此用户 ID 访问数据库中的数据。
- 完全 DBA 权限 （从 " 消息代理 " 连接时）。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"SQL Remote 用户指南 > 管理 SQL Remote for Adaptive Server Anywhere > 消息代理和复制安全性 "

"REVOKE REMOTE DBA 语句 [SQL Remote]" 第 582 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

HELP 语句 [Interactive SQL]

说明

此语句用于在 Interactive SQL 环境中接收帮助。

语法

HELP [*topic*]

用法

HELP 语句用于访问 SQL Anywhere Studio 文档。

可以选择指定帮助的 *topic*。如果 *topic* 是保留字，则必须用单引号括起来。在某些帮助格式中，无法指定主题；这时会提供联机手册主页的链接。

权限

无。

副作用

无。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不适用

IF 语句

说明

此语句用于控制 SQL 语句的条件执行。

语法

IF *search-condition* **THEN** *statement-list*
[**ELSEIF** { *search-condition* | *operation-type* } **THEN** *statement-list*] …
[**ELSE** *statement-list*]
END IF

用法

IF 语句是控制语句，它允许按条件执行第一个 *search-condition* 计算为 TRUE 的 SQL 语句列表。如果没有 *search-condition* 计算为 TRUE，并且存在 ELSE 子句，则执行 ELSE 子句中的 *statement-list*。

执行在 END IF 后的第一个语句继续。

IF 语句不同于 IF 表达式
不要混淆 IF 语句和 IF 表达式的语法。

有关 IF 表达式的信息，请参阅 ""IF 表达式 " 第 19 页 "。

权限

无。

副作用

无。

另请参阅

"BEGIN 语句 " 第 288 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** Transact-SQL IF 语句的语法略有不同。

示例

下面的过程阐释了 IF 语句的用法：

```
CREATE PROCEDURE TopCustomer (OUT TopCompany CHAR(35),
OUT TopValue INT)
BEGIN
   DECLARE err_notfound EXCEPTION
   FOR SQLSTATE '02000';
   DECLARE curThisCust CURSOR FOR
   SELECT company_name, CAST(
sum(sales_order_items.quantity *
   product.unit_price) AS INTEGER) VALUE
   FROM customer
   LEFT OUTER JOIN sales_order
   LEFT OUTER JOIN sales_order_items
   LEFT OUTER JOIN product
   GROUP BY company_name;
   DECLARE ThisValue INT;
   DECLARE ThisCompany CHAR(35);
   SET TopValue = 0;
   OPEN curThisCust;
   CustomerLoop:
   LOOP
      FETCH NEXT curThisCust
      INTO ThisCompany, ThisValue;
      IF SQLSTATE = err_notfound THEN
         LEAVE CustomerLoop;
      END IF;
      IF ThisValue > TopValue THEN
         SET TopValue = ThisValue;
         SET TopCompany = ThisCompany;
      END IF;
   END LOOP CustomerLoop;
   CLOSE curThisCust;
END
```

IF 语句 [T-SQL]

说明

此语句用于控制 SQL 语句的条件执行，是 Watcom-SQL IF 语句的替代方法。

语法

IF *expression*
statement
[**ELSE**
[**IF** *expression*]
statement]

用法

Transact-SQL IF 条件和 ELSE 条件各控制一个 SQL 语句或复合语句（在关键字 BEGIN 和 END 之间）的执行。

与 Watcom-SQL IF 语句相反， Transact-SQL IF 语句中没有 THEN。Transact-SQL 版本也没有 ELSEIF 或 END IF 关键字。

权限

无。

副作用

无。

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** Adaptive Server Enterprise 支持 Transact-SQL IF 语句。

示例

下面的示例阐释了 Transact-SQL IF 语句的用法：

```
IF (SELECT max(id) FROM sysobjects) < 100
   RETURN
ELSE
       PRINT 'These are the user-created objects'
       SELECT name, type, id
       FROM sysobjects
       WHERE id < 100
   END
```

下面两个语句块阐释了 Transact-SQL 和 Watcom-SQL 的兼容性：

```
/* Transact-SQL IF statement */
IF @v1 = 0
   PRINT '0'
ELSE IF @v1 = 1
   PRINT '1'
ELSE
   PRINT 'other'
/* Watcom-SQL IF statement */
IF v1 = 0 THEN
   PRINT '0'
ELSEIF v1 = 1 THEN
   PRINT '1'
ELSE
   PRINT 'other'
END IF
```

INCLUDE 语句 [ESQL]

说明

此语句用于将文件包含在将由 SQL 预处理器扫描的源程序中。

语法

INCLUDE *filename*

filename : **SQLDA** | **SQLCA** | *string*

用法

INCLUDE 语句与 C 预处理器 **#include** 指令非常相似。 SQL 预处理器读取嵌入式 SQL 源文件并用 C 语言源代码替换所有嵌入式 SQL 语句。如果某文件包含 SQL 预处理器所需的信息，请用嵌入式 SQL INCLUDE 语句包含该文件。

特别识别两个文件名：SQLCA 和 SQLDA。下面的语句在所有嵌入式 SQL 源文件中必须出现在任何嵌入式 SQL 语句之前。

```
EXEC SQL INCLUDE SQLCA;
```

此语句必须出现在 C 程序中允许使用静态变量声明的位置。许多嵌入式 SQL 语句需要变量 （对程序员不可见），它们由 SQL 预处理器在 SQLCA 包含语句的位置声明。如果使用了任何 SQLDA，必须包含 SQLDA 文件。

权限

无。

副作用

无。

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** 受 Open Client/Open Server 支持。

INPUT 语句 [Interactive SQL]

说明

此语句用于将数据从外部文件或从键盘导入到数据库表。

语法

INPUT INTO [*owner*.]*table-name*
[**FROM** *filename* | **PROMPT**]
[**FORMAT** *input-format*]
[**ESCAPE CHARACTER** *character*]
[**BY ORDER** | **BY NAME**]
[**DELIMITED BY** *string*]
[**COLUMN WIDTHS (***integer*, …**)**]
[**NOSTRIP**]
[**(** *column-name*, … **)**]

input-format :
ASCII | **DBASE** | **DBASEII** | **DBASEIII**
| **EXCEL** | **FIXED** | **FOXPRO** | **LOTUS**

参数

FORMAT 子句　每组值必须占一个输入行，格式必须是 FORMAT 子句中指定的格式，如果未指定 FORMAT 子句，则是 SET OPTION INPUT_FORMAT 语句设置的格式。当用户输入时，系统提供一个空屏幕，供用户以输入格式每行输入一行。

某些文件格式包含有关列名和列类型的信息。如果数据库表不存在，则 INPUT 语句会使用这些信息创建此表。这是一种非常简单的向数据库中装入数据的方法。包含的信息足够创建表的格式有：DBASEII、DBASEIII、 FOXPRO 和 LOTUS。

从命令文件输入的内容由包含 END 的行终止。从文件输入的内容在文件尾终止。

允许的输入格式有：

- **ASCII**　输入行被假定为 ASCII 字符，每行一行，值由逗号分隔。字母字符串可以用撇号 （单引号）或引号 （双引号）括起来。包含逗号的字符串必须用单引号或双引号括起来。如果字符串本身包含单引号或双引号，则字符串内使用的引号字符需要两个。也可以使用 DELIMITED BY 子句指定与缺省分隔符 （即逗号）不同的分隔符字符串。

  还识别另外三个特殊序列。两个字符 \n 表示换行符， \\ 表示单个 (\) 字符，而 \xDD 序列表示带十六进制代码 DD 的字符。

- **DBASE** 文件格式为 dBASE II 或 dBASE III。根据文件中的信息， Interactive SQL 会尝试确定格式。如果表不存在，则会创建表。
- **DBASEII** 文件格式为 dBASE II。如果表不存在，则会创建表。
- **DBASEIII** 文件格式为 dBASE III。如果表不存在，则会创建表。
- **EXCEL** 输入文件格式为 Microsoft Excel 2.1。如果表不存在，则会创建表。
- **FIXED** 输入行为固定格式。使用 COLUMN WIDTHS 子句可指定列的宽度。如果没有指定列宽，则文件中的列宽必须与相应数据库列类型的任何值所要求的最大字符数相同。

  FIXED 格式不能与包含嵌入式换行符和文件尾字符序列的二进制列一起使用。
- **FOXPRO** 文件格式为 FoxPro （FoxPro 中的备注字段与 dBASE 中的备注字段不同）。如果表不存在，则会创建表。
- **LOTUS** 文件为 Lotus WKS 格式工作表。 INPUT 假定 Lotus WKS 格式工作表的第一行是列名。如果表不存在，则会创建表。这种情况下，所创建的列的类型和大小可能不正确，因为文件中的信息是关于单元格而不是关于列的。

ESCAPE CHARACTER 子句 十六进制代码和符号的缺省转义字符为反斜线 (\)，因此 \x0A 是换行字符。

使用 ESCAPE CHARACTER 子句可更改转义字符。例如，若要将感叹号用作转义字符，应输入：

```
... ESCAPE CHARACTER '!'
```

只能使用一个单字节字符作为转义字符。

BY 子句 BY 子句允许用户指定输入文件中的列是基于它们在列表中的序号位置 （ORDER，缺省设置）还是按它们的名称 (NAME) 与表列进行匹配。并非所有输入格式在文件中都有列名信息。 NAME 只能用于具有列名信息的格式。那些格式是允许自动创建表的格式：DBASEII、DBASEIII、 FOXPRO 和 LOTUS。

DELIMITED BY 子句 DELIMITED BY 子句允许指定用作 ASCII 输入格式中的分隔符的字符串。

COLUMN WIDTHS 子句 只能为 FIXED 格式指定 COLUMN WIDTHS。它在输入文件中指定列的宽度。如果没有指定 COLUMN WIDTHS，则宽度由数据库的列类型决定。如果在 FIXED 格式中插入 LONG VARCHAR 或 BINARY 数据，不应使用该子句。

NOSTRIP 子句 正常情况下，对于 ASCII 输入格式，在插入值之前，无引号字符串中的尾随空白会被去除。 NOSTRIP 可用于取消去除尾随空白。不管是否使用此选项，带引号字符串中的尾随空白都不会去除。不管 NOSTRIP 选项如何设置，无引号字符串中的前导空白都会去除。

如果 ASCII 文件中有列显示为空值的条目，则被视为 NULL。如果那个位置的列不能为 NULL，则在数字列中插入零，并在字符列中插入空字符串。

用法

INPUT 语句允许向指定的数据库表中进行高效的大量插入。通过输入窗口从用户处读取 （如果指定了 PROMPT）或从文件中读取 （如果指定了 FROM 文件名）输入行。如果两者均未指定，则从包含输入语句的命令文件中读取输入：在 Interactive SQL 中，甚至可以直接从 "SQL 语句 " 窗格中读取。这种情况下，输入以仅包含字符串 END 的行结束。

如果为任何输入格式指定了列列表，则数据将插入到指定表的指定列中。缺省情况下， INPUT 语句假定输入文件中的列值显示顺序与它们在数据库表定义中显示的顺序相同。如果输入文件的列顺序不同，必须在 INPUT 语句末尾列出输入文件的实际列顺序。

例如，如果用下面的语句创建一个表：

```
CREATE TABLE inventory (
quantity INTEGER,
item VARCHAR(60)
)
```

并想从依次包含 *name* 值和 *quantity* 值的输入文件 `stock.txt` 中导入 ASCII 数据，

```
'Shirts', 100
'Shorts', 60
```

则必须在 INPUT 语句的末尾列出输入文件的实际列顺序，才能正确地插入数据：

```
INPUT INTO inventory
FROM stock.txt
FORMAT ascii
(item, quantity);
```

缺省情况下，当 INPUT 语句试图插入导致错误的行时会停止。通过设置 ON_ERROR 和 CONVERSION_ERROR 选项，可以以不同的方式来处理错误 （请参阅 "SET OPTION"）。如果在 INPUT 上截断任何字符串值，则 Interactive SQL 会在 " 消息 " 窗格中输出警告。 NOT NULL 列中缺少的值被设置为零 （对于数值类型）和空字符串 （对于非数值类型）。如果 INPUT 试图插入 NULL 行，则输入文件包含空行。

权限

必须有表或视图的 INSERT 权限。

副作用

无。

另请参阅

"OUTPUT 语句 [Interactive SQL]" 第 546 页

"INSERT 语句 " 第 520 页

"UPDATE 语句 " 第 642 页

"DELETE 语句 " 第 435 页

"SET OPTION 语句 " 第 601 页

"LOAD TABLE 语句 " 第 530 页

"xp_read_file 系统过程 " 第 811 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

以下是一个 ASCII 文本文件中的 INPUT 语句的示例。

```
INPUT INTO employee
FROM new_emp.inp
FORMAT ascii;
```

INSERT 语句

说明

此语句用于将单行 （语法 1）或从数据库的其它位置选择的一组行（语法 2）插入到表中。

语法 1

INSERT [**INTO**] [*owner*.]*table-name* [**(** *column-name*, … **)**]
[**ON EXISTING** { **ERROR** | **SKIP** | **UDPATE** }]**VALUES**
(*expression* | **DEFAULT**, … **)**

语法 2

INSERT [**INTO**] [*owner*.]*table-name*
[**ON EXISTING** { **ERROR** | **SKIP** | **UDPATE** }]
[**WITH AUTO NAME**]
select-statement

参数

WITH AUTO NAME 子句 WITH AUTO NAME 仅适用于语法 2。如果您指定 WITH AUTO NAME，则 SELECT 语句中的项的名称确定这些数据属于哪个列。 SELECT 语句项应当是列引用或带别名的表达式。SELECT 语句中未定义的目标列将被赋予缺省值。当目标表中所包含的列数很大时，可以使用此子句。

ON EXISTING 子句 INSERT 语句的 ON EXISTING 子句对两种语法都适用。它根据主键查找将表中的现有行更新为新值。此子句仅可用于拥有主键的表。如果尝试对没有主键的表使用此子句，则将造成语法错误。

如果指定 ON EXISTING 子句，则服务器为每个输入行分别执行主键查找。如果表中不存在对应的行，它照常插入新行。对于表中已经存在的行，您可以选择悄悄忽略输入行 (SKIP)，更新输入行中的值 (UPDATE)，或为重复键值生成错误消息 (ERROR)。

缺省情况下，如果您不指定 ON EXISTING，则在尝试向表中插入行时，如果相应的行已存在，则会造成重复键值错误。这相当于指定 ON EXISTING ERROR。

用法

INSERT 语句用于将新行添加到数据库表中。

语法 1 插入包含指定的表达式值的单行。关键字 DEFAULT 可用于插入列的缺省值。如果给出可选的列名列表，则值逐个插入到指定的列中。如果没有指定列名列表，则值以创建时的顺序 （与用 SELECT * 检索的顺序相同）插入到表列中。行插入到表中的任意位置。（在关系数据库中，表是不排序的）。

语法 2 向表中执行大量插入，包含完全通用的 SELECT 语句的结果。除非 SELECT 语句包含 ORDER BY 子句，否则插入以任意顺序进行。

如果指定列名称，选择列表中的列将按顺序与列列表中指定的列匹配，或者按这些列的创建顺序匹配。

如果定义视图的查询说明是可更新的而且 FROM 子句中只有一个表，则可以插入到视图中。

固有的不可更新视图由查询表达式或包含以下任何内容的查询说明组成：

- DISTINCT 子句
- GROUP BY 子句
- 集合函数
- 不是基表的 *select-list* 项。

插入到表中的字符串始终以它们输入时的大小写保存，不论数据库是否区分大小写。因此，插入到表中的字符串 **Value** 在数据库中保存时 V 始终为大写，其余的字母为小写。 SELECT 语句返回字符串作为 **Value**。但是，如果数据库不区分大小写，则所有的比较将 **Value** 视为与 **value**、**VALUE** 等相同。而且，如果单列主键已经含有 **Value** 条目，将会拒绝 **value**，因为它会导致主键不唯一。

性能提示
要将许多行插入到表中，应尽可能声明一个游标并通过游标插入行，这比执行许多单独的 INSERT 语句效率要高。 插入数据之前，可以指定每个表页上保留供以后的更新使用的可用空间的百分比。有关更多信息，请参阅 "ALTER TABLE 语句 " 第 272 页。

权限

必须有表的 INSERT 权限。

副作用

无。

另请参阅

"INPUT 语句 [Interactive SQL]" 第 515 页

"UPDATE 语句 " 第 642 页

"DELETE 语句 " 第 435 页

"PUT 语句 [ESQL]" 第 558 页

标准和兼容性

- **SQL/92** 初级特性。 INSERT … ON EXISTING 是一个供应商扩展。
- **SQL/99** 核心特性。 INSERT … ON EXISTING 是一个供应商扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

示例

将 Eastern Sales 部门添加到数据库。

```
INSERT
INTO department ( dept_id, dept_name )
VALUES ( 230, 'Eastern Sales' )
```

创建表 *dept_head*，然后用部门主管名字和部门的名称填充该表。

```
CREATE TABLE dept_head(
      pk int primary key default autoincrement,
      dept_name varchar(128),
      manager_name varchar (128) );
INSERT
INTO dept_head (manager_name, dept_name)
SELECT emp_fname || ' ' || emp_lname AS manager,
       dept_name
FROM employee JOIN department
ON emp_id = dept_head_id
```

创建表 *dept_head*，然后使用 WITH AUTO NAME 语法，以部门主管名字和部门名称填充该表。

```
CREATE TABLE dept_head(
      pk int primary key default autoincrement,
      dept_name varchar(128),
      manager varchar (128) );
INSERT
INTO dept_head WITH AUTO NAME
SELECT emp_fname || ' ' || emp_lname AS manager,
      dept_name
FROM employee JOIN department
ON emp_id = dept_head_id
```

创建表 *mytab*，并使用 WITH AUTO NAME 语法填充该表。

```
CREATE TABLE mytab(
      pk int primary key default autoincrement,
      table_name char(128),
      len int );
INSERT into mytab WITH AUTO NAME
SELECT
      length(t.table_name) AS len,
      t.table_name
FROM SYS.SYSTABLE t
WHERE table_id<=10
```

INSTALL 语句

说明

此语句用于使 Java 类可在数据库中使用。

语法

INSTALL JAVA
[**NEW** | **UPDATE**]
[**JAR** *jar-name*]
FROM { **FILE** *filename* | *expression* }

参数

NEW | UPDATE 关键字 如果指定 NEW 安装模式，则引用的 Java 类必须是新类，而不是当前安装的类的更新。如果数据库中存在同名的类而且使用 NEW 安装模式，则会出现错误。

如果指定 UPDATE，则引用的 Java 类可以包含给定数据库中已安装的 Java 类的替换类。

如果省略 *install-mode* ，则缺省模式为 NEW。

JAR 子句 如果指定了该子句，则 *filename* 必须指定 jar 文件。 Jar 文件通常带扩展名 `.jar` 或 `.zip`。

可以压缩或解压缩已安装的 jar 和 zip 文件。

如果指定了 JAR 选项，安装了 jar 包含的类后，该 jar 保留为 jar 文件。该 jar 是与这些类的每一个关联的 jar。使用 JAR 选项安装在数据中的 jar 称为数据库的保留 jar。

jar-name 是字符串值，最长为 255 个字节。 *jar-name* 用于在后面的 INSTALL UPDATE 和 REMOVE 语句中标识保留 jar。

FROM FILE 子句 指定安装 Java 类的位置。

file-name 支持的格式包括完全限定文件名 （如 `"c:\libs\jarname.jar"` 和 `"/usr/u/libs/jarname.jar"`）和相对文件名 （相对于数据库服务器的当前工作目录）。

filename 必须标识类文件或者 jar 文件。

FROM 表达式子句 表达式必须计算为二进制类型，其值必须包含有效的类文件或 jar 文件。

用法

每个类的类定义是在首次使用该类时由每个连接的 VM 载入的。INSTALL （安装）类时，隐式重新启动连接的 VM。因此，不管 INSTALL 的 *install-mode* 是 NEW 还是 UPDATE，都可以直接访问新类。由于 VM 重新启动，保存在 Java 静态变量中的所有值都丢失，任何类型为 Java 类的 SQL 变量也被删除。

对于其它连接，下次在 VM 第一次访问新类时装入该类。如果该类已由 VM 装入，直到为该连接重新启动 VM （例如，用 STOP JAVA 和 START JAVA）后，该连接才能看到新类。

权限

需要 DBA 权限才能执行 INSTALL 语句。

任何用户可以任何方式引用所有已安装的类。

在 Windows CE 上不受支持。

另请参阅

"REMOVE 语句 " 第 567 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

下面的语句通过提供文件名和类的位置，安装用户创建的 Java 类 Demo。

```
INSTALL JAVA NEW
FROM FILE 'D:\JavaClass\Demo.class'
```

安装后，类由它的名称引用，不再使用它的原文件路径位置。例如，下面的语句使用上一语句中安装的类。

```
create variable d Demo
```

如果 Demo 类是程序包 *sybase.work* 的成员，必须使用该类的完全限定名，例如，

```
CREATE VARIABLE d sybase.work.Demo
```

下面的语句安装 zip 文件中包含的所有类，并将数据库中的类与 JAR 文件名关联。

```
INSTALL JAVA
JAR 'Widgets'
FROM FILE 'C:\Jars\Widget.zip'
```

同样，不保留 zip 文件的位置，必须使用完全限定的类名称 （包名称和类名称）引用类。

LEAVE 语句

说明

此语句用于退出复合语句或循环。

语法

LEAVE *statement-label*

另请参阅

"LOOP 语句 " 第 539 页

"FOR 语句 " 第 480 页

"BEGIN 语句 " 第 288 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 "

用法

LEAVE 语句是控制语句，它允许您退出带标签的复合语句或带标签的循环。执行在复合语句或循环之后的第一个语句重新开始。

复合语句是过程或触发器的主体，它具有与过程或触发器同名的隐式标签。

权限

无。

副作用

无。

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。 BREAK 语句为 Transact-SQL 兼容过程提供类似的功能。

示例

下面的代码段显示如何使用 LEAVE 语句退出循环。

```
SET i = 1;
lbl:
LOOP
   INSERT
   INTO Counters ( number )
   VALUES ( i );
```

```
   IF i >= 10 THEN
      LEAVE lbl;
   END IF;
   SET i = i + 1
END LOOP lbl
```

下面的代码段示例在嵌套循环中使用 LEAVE。

```
outer_loop:
LOOP
   SET i = 1;
   inner_loop:
   LOOP
      ...
      SET i = i + 1;
      IF i >= 10 THEN
         LEAVE outer_loop
      END IF
   END LOOP inner_loop
END LOOP outer_loop
```

LOAD STATISTICS 语句

说明

此语句将统计信息装入系统表 SYSCOLSTAT 中。 dbunload 实用程序使用此语句从旧数据库中卸载列统计信息。不应手工使用此语句。

语法

LOAD STATISTICS [[*owner*.]*table-name*.]*column-name*
format-id, density, max-steps, actual-steps, step-values, frequencies

参数

format_id 内部字段，用于确定 SYSCOLSTAT 系统表中其余行的格式。

density 估计列的单值加权平均选择性，不计算存储在行中的单值选择性大的选择性。

max_steps 直方图中允许的最大梯级数。

actual_steps 此时实际使用的梯级数。

step_values 直方图梯级的界限值。

frequencies 直方图梯级的选择性。

权限

必须拥有 DBA 权限。

副作用

无。

另请参阅

"SYSCOLSTAT 系统表 " 第 674 页

"ASA 数据库管理指南 > 数据库管理实用程序 > 使用 dbunload 实用程序卸载数据库 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

LOAD TABLE 语句

说明

此语句用于将批量数据从外部 ASCII 格式文件导入到数据库表中。*日志文件中不记录插入操作*，这在崩溃时会导致数据丢失风险，而且导致此语句不能用于 SQL Remote 或 MobiLink 远程数据库。

语法

LOAD [**INTO**] **TABLE** [*owner.*]*table-name* [**(** *column-name*, … **)**]
FROM *filename-string*
[*load-option* …]

load-option :
CHECK CONSTRAINTS { **ON** | **OFF** }
| **COMPUTES** { **ON** | **OFF** }
| **DEFAULTS** { **ON** | **OFF** }
| **DELIMITED BY** *string*
| **ESCAPE CHARACTER** *character*
| **ESCAPES** { **ON** | **OFF** }
| **FORMAT** { **ASCII** | **BCP** }
| **HEXADECIMAL** {**ON** | **OFF**}
| **ORDER** {**ON** | **OFF**}
| **PCTFREE** *percent-free-space*
| **QUOTES** { **ON** | **OFF** }
| **STRIP** { **ON** | **OFF** }
| **WITH CHECKPOINT** { **ON** | **OFF** }

参数

Column-name 如果 DEFAULTS 选项是关闭的，则列列表中不存在的任何列都变为 NULL。如果 DEFAULTS 是打开的，而且列有缺省值，则使用该值。如果 DEFAULTS 是关闭的，而且从列列表中省略了不可为空的列，则引擎会尝试将空字符串转换为列的类型。

如果指定了列列表，它列出文件中应存在的列以及列的显示顺序。列名不能重复。列表中没有出现的列名将设置为空值 / 零 / 空或 DEFAULT，具体取决于列的为空性、数据类型和 DEFAULT 设置。使用列名 "filler()" 可指定存在于输入文件中但要被 LOAD TABLE 忽略的列。

FROM 选项 *filename-string* 以字符串的形式传递到服务器。因此，该字符串遵循的格式要求与其它 SQL 字符串一样。具体说来就是：

- 要指示目录路径，反斜线字符 \ 必须用两个反斜线来表示。用于将数据从文件 `c:\temp\input.dat` 装入 employee 表的语句是：

```
LOAD TABLE employee
FROM 'c:\\temp\\input.dat' ...
```

- 路径名相对于数据库服务器而不是客户端应用程序。如果在另一台计算机的数据库服务器上运行此语句，目录名是指服务器计算机上的目录，而不是指客户端计算机上的目录。
- 可以使用 UNC 路径名从服务器以外的计算机上的文件装载数据。例如，在 Windows for Workgroups、 Windows 95 或 Windows NT 网络上，可以使用下面的语句从客户端计算机上的文件装载数据：

```
LOAD TABLE employee
FROM '\\\\client\\temp\\input.dat'
```

CHECK CONSTRAINTS 选项 缺省情况下，此选项为 ON，但是卸载实用程序在此选项设置为 OFF 时写出 LOAD TABLE 语句。

将 CHECK CONSTRAINTS 设置为 OFF 可禁用检查约束。这很有用，例如在重建数据库时。如果表的检查约束调用尚未创建的用户定义函数，重建将失败，除非此选项设置为 OFF。

COMPUTES 选项 缺省情况下， COMPUTES 设置为 ON。将 COMPUTES 设置为 ON 将重新计算计算列。

将 COMPUTES 设置为 OFF 将禁止重新计算计算列。此选项是有用的，例如，在您重建数据库但表的某个计算列调用尚未创建的用户定义函数时。如果此选项不设置为 OFF，则重建将失败。

卸载实用程序 (dbunload) 在 COMPUTES 选项设置为 OFF 时写出 LOAD TABLE 语句。

DEFAULTS 选项 缺省情况下， DEFAULTS 为 OFF。如果 DEFAULTS 为 OFF，列列表中没有出现的任何列都被赋以 NULL。如果 DEFAULTS 为 OFF 并且从列列表中省略了不可为空的列，数据库服务器会尝试将空字符串转换为列的类型。如果 DEFAULTS 为 ON 而且列有缺省值，则使用该值。

DELIMITED BY 选项 缺省列分隔符是逗号。通过提供字符串可以指定其它列分隔符。相同的格式要求同样适用于其它 SQL 字符串。具体说来，如果您想指定以制表符分隔的值，则使用制表符 (9) 的十六进制 ASCII 代码。 DELIMITED BY 子句如下：

```
...DELIMITED BY '\x09' ...
```

您可以指定最长达 255 字节的分隔符。例如，

```
...DELIMITED BY '###' ...
```

ESCAPE CHARACTER 选项 以十六进制代码和符号存储的字符的缺省转义字符是反斜线 (\)，例如 \x0A 是换行符。

使用 ESCAPE CHARACTER 子句可以更改此行为。例如，若要使用感叹号作为转义字符，请输入

```
... ESCAPE CHARACTER '!'
```

只能使用一个单字节字符作为转义字符。

ESCAPES 选项 ESCAPES 打开时 （缺省设置），数据库服务器将识别反斜线字符后面的字符并解释为特殊字符。换行符可以作为组合 \n 包含在数据中，其它字符可以作为十六进制 ASCII 代码 （如使用 \x09 代替制表符）包含在数据中。两个连续的反斜线字符 (\\ ) 被解释为单个反斜线。 n、 x、 X 或 \ 以外的任何字符前面的反斜线都被解释为两个单独的字符。例如， \q 插入一个反斜线和字母 q。

FORMAT 选项 如果选择 ASCII，则输入行被假定为 ASCII 字符，每个输入行占一行，值由列分隔符分隔。选择 BCP 将允许导入 ASE 生成的、包含 Blob 的 BCP 输出文件。

HEXADECIMAL 选项 缺省情况下， HEXADECIMAL 设置为 ON。当 HEXADECIMAL 为 ON 时，二进制列值读作 **0x***nnnnnn*_，其中每个 *n* 是一个十六进制数字。处理多字节字符集时，使用 HEXADECIMAL ON 很重要。

HEXADECIMAL 选项只能与 FORMAT ASCII 选项一起使用。

ORDER 选项 如果 ORDER 为 ON，并且声明了一个聚簇索引，则 LOAD TABLE 根据聚簇索引排序输入数据并以相同的顺序插入行。如果要上载的数据已经排序，则应当将 ORDER 设置为 OFF。

有关更多信息，请参阅 "ASA SQL 用户指南 > 使用数据库对象 > 使用聚簇索引 "。

QUOTES 选项　当 QUOTES 打开时 （缺省设置），LOAD TABLE 语句要求将字符串用引号字符引起来。引号字符是撇号 （单引号）或引号 （双引号）。字符串中出现的第一个这样的字符被视为该字符串的引号字符。字符串必须以匹配的引号结束。

当 QUOTES 打开时，列值中可以包含列分隔符。另外，假定引号字符不属于该值。因此，格式为

```
'123 High Street, Anytown',(715)398-2354
```

的行按两个值处理，而不是三个，尽管地址中出现了逗号。另外，将地址引起来的引号不插入到数据库中。

若要在值中包含引号字符，当 QUOTES 打开时，必须使用两个引号。下面的行包括第三列中的值 （单个引号字符）：

```
'123 High Street, Anytown','(715)398-2354',''''
```

STRIP 选项　当 STRIP 打开时 （缺省设置），插入之前将尾随空白从值中去除。若要关闭 STRIP 选项，则使用如下子句：

```
...STRIP OFF ...
```

仅去除非引用字符串的尾随空白。被引用字符串保留其尾随空白。除非前导空白是用引号引起来的，否则不管 STRIP 的设置如何，都裁剪前导空白。

WITH CHECKPOINT 选项　缺省设置为 OFF。如果设置为 ON，则当成功完成并记录语句之后会发出检查点。

如果没有指定 WITH CHECKPOINT ON，而且在发出 CHECKPOINT 之前数据库需要自动恢复，必须存在用来装载表的数据文件才能成功完成恢复。如果指定了 WITH CHECKPOINT ON，而且之后需要恢复，则恢复在检查点操作完成后开始，不需要数据文件。

注意
如果将数据库选项 CONVERSION_ERROR 设置为 OFF，可能会将错误的数据装载到表中而且不会报告任何错误。如果没有指定 WITH CHECKPOINT ON，而且需要恢复数据库，则当恢复过程中

CONVERSION_ERROR 为 ON （缺省值）时，恢复可能失败。建议装载表的时候不要将 CONVERSION_ERROR 设置为 OFF，并且不要指定 WITH CHECKPOINT ON。

有关更多信息，请参阅 "CONVERSION_ERROR 选项 "。

如果数据库损坏，而且需要使用备份并应用当前的日志文件，则需要数据文件，与该选项的设置无关。

PCTFREE 选项 指定希望为每个表页保留的可用空间百分比。该设置覆盖表的所有永久设置，但只限于装载过程中。

值 percent-free-space 是 0 到 100 之间的整数。前者指定每页上没有剩余的可用空间：每页完全填满。高值会将每行单独插入到页中。

有关 PCTFREE 的更多信息，请参阅 ""CREATE TABLE 语句 " 第 395 页 "。

用法

LOAD TABLE 语句允许以高效率从 ASCII 文件将大量数据插入到数据库表中。 LOAD TABLE 的效率比 Interactive SQL 语句 INPUT 高。

插入数据之前，可以指定每个表页上保留供以后的更新使用的可用空间的百分比。有关更多信息，请参阅 "ALTER TABLE 语句 "。

LOAD TABLE 在整个表上放置一个排它锁。它不触发与表关联的任何触发器。

LOAD TABLE 在装载数据时捕获列统计信息，以便创建表列的直方图。如果已存在列的直方图，则 LOAD TABLE 不改变现有的直方图，也不创建新的直方图。如果要装载到空表中，先删除统计信息是有益的。

对于超过 90% 的上载行， LOAD TABLE 不为包含 NULL 值的列生成统计信息。

LOAD TABLE 保存基表的统计信息以备以后使用。它不保存全局临时表的统计信息。

如果装载的行数大于数据库选项 MIN_TABLE_SIZE_FOR_HISTOGRAM 中指定的阈值 （缺省值为 1000），则 LOAD TABLE 只添加统计信息。如果表中至少含有那么多行，则按如下方式添加直方图：

数据是否已在表中？	是否存在直方图？	采取的操作
Yes	Yes	使用现有直方图
Yes	No	不建立直方图
No	Yes	使用现有直方图
No	No	建立新的直方图

有关更多信息，请参阅 "ASA SQL 用户指南 > 查询优化与执行 > 优化程序估计值 "。

对临时表可以使用 LOAD TABLE，但是该临时表必须是用 ON COMMIT PRESERVE ROWS 子句创建的，因为 LOAD TABLE 在装载后会执行 COMMIT 操作。

如果 ASCII 文件包含的条目使列看起来为 NULL，则 LOAD TABLE 将该列视为空。如果列在该位置不能为 NULL，它会在数字列中插入零，在字符列中插入空字符串。 LOAD TABLE 跳过输入文件中的空行。

注意
LOAD TABLE 仅用于快速装载大量数据。 LOAD TABLE 不将向事务日志写入单个行。

权限

执行 LOAD TABLE 语句所需的权限在数据库服务器的命令行上用 -gl 选项设置。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -gl 服务器选项 "。

表需要有排它锁。

副作用

日志文件中不记录插入操作。因此，在系统崩溃时可能不能恢复插入的行。此外， LOAD TABLE 语句决不能在 SQL Remote 复制中涉及的数据库中或用作 MobiLink 客户端的数据库中使用，因为这些技术会通过分析日志文件复制更改。

LOAD TABLE 语句不触发触发器，包括参照完整性操作。

操作开始时会执行检查点操作。在操作结束时，会执行另一次可选的检查点操作。

如果加载了大量的信息，则将更新列统计信息。

自动提交。

另请参阅

"UNLOAD TABLE 语句 " 第 640 页

"ASA 数据库管理指南 > 数据库选项 > MIN_TABLE_SIZE_FOR_HISTOGRAM 选项 [database]"

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

以下是 LOAD TABLE 的示例。首先，创建一个表，然后使用输入文件将数据装载到表中。

```
CREATE TABLE T( a char(100), let_me_default int DEFAULT
1, c char(100) )
```

以下是输入文件的内容：文件和正确装载文件的 LOAD TABLE 语句：

```
ignore_me, this_is_for_column_c, this_is_for_column_a
```

以下 LOAD 语句装载输入文件：

```
LOAD TABLE T ( filler(), c, a ) FROM 'input' FORMAT
ASCII DEFAULTS ON
```

表中出现下面的行数据：

```
this_is_for_column_a, 1, this_is_for_column_c
```

LOCK TABLE 语句

说明 此语句用于防止其它并发事务访问或修改表。

语法

LOCK TABLE *table-name*
    [WITH HOLD]
    IN { SHARE | EXCLUSIVE } MODE

参数

table-name 表必须是基表，而不是视图。因为临时表数据是当前连接的本地数据，所以锁定全局临时表或局部临时表都无效。

WITH HOLD 子句 如果指定该子句，锁一直保持到连接结束时。如果没有指定该子句，则提交或回滚当前事务时将释放锁。

SHARE 模式 禁止其它事务修改表，但允许它们读取表。在该模式下，只要没有其它的事务锁定所修改的行，就可以间接地或显式地使用 LOCK TABLE 更改表中的数据。

EXCLUSIVE 模式 禁止其它事务访问表。任何其它事务都不能对表执行查询、任何更新或任何其它操作。

用法

LOCK TABLE 语句允许直接控制表级并发，与当前的隔离级别无关。

虽然事务的隔离级别通常控制当前事务执行请求时设置的锁的种类，但 LOCK TABLE 语句允许更显式地控制表行的锁定。

LOCK TABLE 以 SHARE 模式放置的锁为幻像和防幻像锁， *sa_locks* 过程将它们显示为 PT 和 AT。

权限

若要以 SHARE 模式锁定表，需要 SELECT 权限。

若要以 EXCLUSIVE 模式锁定表，必须是表所有者或者有 DBA 权限。

副作用 需要访问锁定表的其它事务可能被延迟或阻塞。

另请参阅

"SELECT 语句 " 第 587 页

"sa_locks 系统过程 " 第 769 页

标准和兼容性

- **SQL/92** 供应商扩展。

- **SQL/99** 供应商扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。 Adaptive Server Enterprise 不支持 WITH HOLD 子句。 Adaptive Server Enterprise 提供 Adaptive Server Anywhere 不支持的 WAIT 子句。

示例 下面的语句禁止其它事务在当前事务期间修改 *customer* 表：

```
LOCK TABLE customer
IN SHARE MODE
```

LOOP 语句

说明

此语句用于重复执行语句列表。

语法

[*statement-label* :]
[**WHILE** *search-condition*] **LOOP**
statement-list
END LOOP [*statement-label*]

用法

WHILE 和 LOOP 语句都是控制语句，当 *search-condition* 计算为 TRUE 时，这两个语句允许重复执行 SQL 语句列表。 LEAVE 语句可用于在 END LOOP 之后的第一个语句重新开始执行。

如果指定结尾 *statement-label*，它必须与起始 *statement-label* 匹配。

权限

无。

副作用

无。

另请参阅

"FOR 语句 " 第 480 页

"LEAVE 语句 " 第 527 页

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。 WHILE 语句提供 Transact-SQL 存储过程中的循环。

示例

过程中的 While 循环。

```
...
SET i = 1;
WHILE i <= 10 LOOP
   INSERT INTO Counters( number ) VALUES ( i );
   SET i = i + 1;
END LOOP;
...
```

过程中带标签的循环。

```
SET i = 1;
lbl:
LOOP
   INSERT
   INTO Counters( number )
   VALUES ( i );
   IF i >= 10 THEN
      LEAVE lbl;
   END IF;
   SET i = i + 1;
END LOOP lbl
```

MESSAGE 语句

说明 此语句用于显示消息。

语法

MESSAGE *expression*, …
[**TYPE** { **INFO** | **ACTION** | **WARNING** | **STATUS** }]
[**TO** { **CONSOLE** | **CLIENT** | **LOG** }]

参数

TYPE 子句 只有将消息发送到客户端时 TYPE 子句才有效。客户端应用程序必须决定如何处理消息。 Interactive SQL 在以下位置显示消息：

- **INFO** " 消息 " 窗格。 INFO 为缺省类型。
- **ACTION** 含 " 确定 " 按钮的消息框。
- **WARNING** 含 " 确定 " 按钮的消息框。
- **STATUS** " 消息 " 窗格。

TO 子句 该子句指定消息的目标：

- **CONSOLE** 将消息发送到数据库服务器窗口。 CONSOLE 为缺省目标。
- **CLIENT** 将消息发送到客户端应用程序。您的应用程序必须决定如何处理消息，您可以根据 TYPE 信息作出决定。
- **LOG** 将消息发送到 -o 选项指定的服务器日志文件中。

用法 MESSAGE 语句显示消息，该消息可以是任何表达式。子句可以指定显示消息的位置。

有效的表达式可以包含用引号引起来的字符串或其它常量、变量或函数。但是，在 Message 语句的输出中不能使用查询，即使表达式的定义中包含查询。

权限 无。

副作用 无。

另请参阅 "CREATE PROCEDURE 语句 " 第 347 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。 Transact-SQL PRINT 语句提供类似的功能，在 Adaptive Server Anywhere 中也可用。

示例

下面的过程在服务器消息窗口中显示消息：

```
CREATE PROCEDURE message_test ()
BEGIN
MESSAGE 'The current date and time: ', Now();
END
```

语句：

```
CALL message_test()
```

在数据库服务器消息窗口中显示字符串 *The current date and time* 以及当前日期和时间。

OPEN 语句 [ESQL] [SP]

说明

此语句用于打开以前声明的游标以访问数据库中的信息。

语法

OPEN *cursor-name*
[**USING** [**DESCRIPTOR** *sqlda-name* | *hostvar*, …]]
[**WITH HOLD**]
[**ISOLATION LEVEL** *n*]
[**BLOCK** *n*]

cursor-name : identifier or *hostvar*

sqlda-name : identifier

参数

嵌入式 SQL 用法 成功执行 OPEN 语句之后， SQLCA (SQLIOESTIMATE) 的 *sqlerrd[3]* 字段由读取查询的所有行所需的输入 / 输出操作数估计值填充。另外， SQLCA (SQLCOUNT) 的 *sqlerrd[2]* 字段由游标中的实际行数 （大于或等于 0 的值）或它的估计数 （绝对值是估计数的负数）填充。如果数据库服务器不用计算行数就能计算该数，则它是实际的行数。也可以将数据库配置为始终返回实际的行数（请参阅 "ASA 数据库管理指南 > 数据库选项 > ROW_COUNTS 选项 [database]"），但是这样做的开销很大。

如果是由标识符或字符串指定 *cursor-name*，则相应的 DECLARE CURSOR 在 C 程序中必须出现在 OPEN 之前；如果是由主机变量指定 *cursor-name*，则 DECLARE CURSOR 语句必须在 OPEN 语句之前执行。

USING DESCRIPTOR 子句 USING DESCRIPTOR 子句仅用于嵌入式 SQL。它指定绑定到已声明游标的 SELECT 语句中的占位符绑定变量的主机变量。

WITH HOLD 子句 缺省情况下，在当前事务 （COMMIT 或 ROLLBACK）结束时自动关闭所有游标。可选的 WITH HOLD 子句使游标对后面的事务保持打开。游标一直保持打开，直到当前连接结束或执行了显式 CLOSE 语句。终止连接时自动关闭游标。

ISOLATION LEVEL 子句 ISOLATION LEVEL 子句允许以不同于 ISOLATION_LEVEL 选项的当前设置的隔离级别打开该游标。该游标的所有操作都以指定的隔离级别执行，与该选项的设置无关。如果不指定

该子句，则打开游标时，在游标打开的整个时间内，游标的隔离级别是 ISOLATION_LEVEL 选项的值。请参阅 "ASA SQL 用户指南 > 使用事务和隔离级别 > 锁定的工作方式 "。

游标位于第一行的前面 （请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > 在嵌入式 SQL 中使用游标 " 或 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 在过程和触发器中使用游标 "）。

BLOCK 子句 该子句仅用于嵌入式 SQL。客户端应用程序以块的方式（一次读取多行）读取行。缺省情况下，块包含的行数基于行的大小和数据库服务器读取每行所花的时间动态确定。应用程序通过指定 BLOCK 子句，可以指定块中包含的最大行数。例如，如果一次读取和显示 5 行，则使用 **BLOCK 5**。指定 **BLOCK 0** 将导致一次读取一行，还会导致 FETCH RELATIVE 0 总是再次读取该行。

有关更多信息，请参阅 "FETCH 语句 [ESQL] [SP]" 第 475 页。

用法

OPEN 语句打开指定的游标。该游标必须是以前声明的。

当游标位于 CALL 语句上时， OPEN 导致过程一直执行，直到遇到第一个结果集 （不含 INTO 子句的 SELECT 语句）为止。如果该过程结束时未找到任何结果集，则会设置 SQLSTATE_PROCEDURE_COMPLETE 警告。

权限

必须有 SELECT 语句中所有表的 SELECT 权限，或者有 CALL 语句中的过程的 EXECUTE 权限。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"RESUME 语句 " 第 573 页

"PREPARE 语句 [ESQL]" 第 553 页

"FETCH 语句 [ESQL] [SP]" 第 475 页

"RESUME 语句 " 第 573 页

"CLOSE 语句 [ESQL] [SP]" 第 301 页

标准和兼容性

- **SQL/92**　嵌入式 SQL 使用是初级特性。过程使用是持久存储模块特性。
- **SQL/99**　嵌入式 SQL 使用是核心特性。过程使用是持久存储模块特性。
- **Sybase**　Adaptive Server Enterprise 支持简单的 OPEN *cursor-name* 语法。 Adaptive Server Enterprise 存储过程不支持任何其它子句。Open Client/Open Server 支持使用描述符或主机变量语法。

示例

下面的示例显示 OPEN 在嵌入式 SQL 中的用法。

```
EXEC SQL OPEN employee_cursor;
```

和

```
EXEC SQL PREPARE emp_stat FROM
'SELECT empnum, empname FROM employee WHERE name like
?';
EXEC SQL DECLARE employee_cursor CURSOR FOR emp_stat;
EXEC SQL OPEN employee_cursor USING :pattern;
```

下面的示例来自过程或触发器。

```
BEGIN
DECLARE cur_employee CURSOR FOR
   SELECT emp_lname
   FROM employee;
DECLARE name CHAR(40);
OPEN cur_employee;
LOOP
FETCH NEXT cur_employee into name;
    ...
END LOOP
CLOSE cur_employee;
END
```

OUTPUT 语句 [Interactive SQL]

说明

此语句用于将当前的查询结果输出到文件。

语法

OUTPUT TO *filename*
[**APPEND**]
[**VERBOSE**]
[**FORMAT** *output-format*]
[**ESCAPE CHARACTER** *character*]
[**DELIMITED BY** *string*]
[**QUOTE** *string* [**ALL**]]
[**COLUMN WIDTHS (***integer*, *…***)**]
[**HEXADECIMAL** { **ON** | **OFF** | **ASIS** }]

output-format :
ASCII | **DBASEII** | **DBASEIII** | **EXCEL**
| **FIXED** | **FOXPRO** | **HTML** | **LOTUS** | **SQL** | **XML**

参数

APPEND 子句 此可选关键字用于将查询结果附加到现有输出文件的末尾，但不覆盖文件以前的内容。如果不使用 APPEND 子句，缺省情况下 OUTPUT 语句覆盖输出文件的内容。当输出格式为 ASCII、 FIXED 或 SQL 时， APPEND 关键字有效。

VERBOSE 子句 当含有可选的 VERBOSE 关键字时，有关查询的错误消息、用于选择数据的 SQL 语句和数据本身写入输出文件。如果省略 VERBOSE （缺省设置），则只将数据写入文件。当输出格式为 ASCII、FIXED 或 SQL 时， VERBOSE 关键字有效。

FORMAT 子句 允许的输出格式有：

- **ASCII** 输出是 ASCII 格式文件 （文件中每个数据行占一行）。所有的值由逗号分隔，字符串由撇号是 （单引号）引起来的。使用 DELIMITED BY 和 QUOTE 子句可以更改分隔符和引号字符串。如果在 QUOTE 子句中指定 ALL，则所有的值 （不止是字符串）都被引号引起来。

  还使用了三种其它的特殊序列。两个字符 \n 代表换行符； \\ 代表单个 \ ；而序列 \xDD 代表具有十六进制代码 DD 的字符。这是缺省的输出格式。

  如果导出含有字符串返回值的 Java 方法，必须使用 HEXADECIMAL OFF 子句。

- **DBASEII** 输出是 dBASE II 格式文件 （文件的开头是列定义）。注意最多可输出 32 列。还请注意，文件中长度超过 255 个字符的列将被截断。
- **DBASEIII** 输出是 dBASE III 格式文件 （文件的开头是列定义）。注意最多可输出 128 列。还请注意，文件中长度超过 255 个字符的列将被截断。
- **EXCEL** 输出是 Excel 2.1 工作表。工作表的第一行包含列标签（如果没有定义标签，则为列名称）。后面的工作表行包含实际的表数据。
- **FIXED** 输出是每列都有固定宽度的固定格式。使用 COLUMN WIDTHS 子句可以指定每一列的宽度。列标题不以此格式输出。

  如果省略 COLUMN WIDTHS 子句，则根据列的数据类型计算每列的宽度，该宽度足以容纳此数据类型的任何值。例外情况是 LONG VARCHAR 和 LONG BINARY 数据缺省为 32 KB。
- **FOXPRO** 输出是 FoxPro 格式文件，文件的开头是列定义（FoxPro 注释字段与 dBASE 注释字段不同）。注意最多可输出 128 列。还请注意，文件中长度超过 255 个字符的列将被截断。
- **HTML** 输出是超文本标记语言格式。
- **LOTUS** 输出是 Lotus WKS 格式工作表。列名将作为第一行放在工作表中。请注意，在其它软件 （如 Lotus 1-2-3）可以装载的 Lotus WKS 格式工作表的最大大小方面有特定的限制。Interactive SQL 生成的文件没有大小限制。
- **SQL** 输出是重新创建表中的信息所需的 Interactive SQL INPUT 语句。
- **XML** 输出是用 UTF-8 编码并且包含嵌入式 DTD 的 XML 文件。二进制值用 CDATA 块编码，二进制数据显示为两位数十六进制字符串。 INPUT 语句不接受 XML 作为文件格式。

ESCAPE CHARACTER 子句 作为十六进制代码和符号存储的字符的缺省转义字符为反斜线 (\)，例如 \x0A 为换行符。

使用 ESCAPE CHARACTER 子句可以更改此行为。例如，若要使用感叹号作为转义字符，请输入

```
... ESCAPE CHARACTER '!'
```

DELIMITED BY 子句 DELIMITED BY 子句仅用于 ASCII 输出格式。分隔符字符串位于各列之间 （缺省为逗号）。

QUOTE 子句 QUOTE 子句仅用于 ASCII 输出格式。引号字符串位于字符串值的两边。缺省为单引号字符。如果在 QUOTE 子句中指定 ALL，则引号字符串将被置于所有值的两边，而不仅仅是字符串的两边。

COLUMN WIDTHS 子句 COLUMN WIDTHS 子句用于为 FIXED 格式的输出指定列宽。

HEXADECIMAL 子句 HEXADECIMAL 子句指定如何仅为 ASCII 格式卸载二进制数据。设置为 ON 时，二进制数据以 **0xabcd** 格式卸载。设置为 OFF 时，二进制数据在卸载时转义 (**\xab\xcd**)。设置为 ASIS 时，值按原样写入，即不发生任何转义，即使值中包含控制字符。 ASIS 对于含有格式化字符 （如制表符或回车符）的文本很有用。

用法

OUTPUT 语句把当前查询所检索的信息复制到文件。

可以使用可选的 FORMAT 子句指定输出格式。如果不指定 FORMAT 子句，将使用 Interactive SQL OUTPUT_FORMAT 选项设置 （请参阅 "ASA 数据库管理指南 > 数据库选项 > OUTPUT_FORMAT 选项 [ISQL]"）。

当前查询为 SELECT 或 INPUT 语句，此语句生成的信息显示在 " 结果 " 窗格中的 " 结果 " 选项卡上。如果没有任何当前查询， OUTPUT 语句将报告错误。

导出 Java 数据时，可能需要以二进制格式导出对象，或者使用 **toString()** 方法以字符串格式导出数据。可以使用 DESCRIBE_JAVA_FORMAT Interactive SQL 选项控制导出 Java 数据的方式。

例如，请看下面的脚本：

```
CREATE VARIABLE JavaString java.lang.String;
SET JavaString = NEW java.lang.String( 'TestVar' );
SELECT JavaString FROM dummy;
```

如果将 DESCRIBE_JAVA_FORMAT 设置为 **Varchar**：

- 下面的命令以十六进制格式显示输出文件中的 TestVar。

  ```
  OUTPUT TO filename
  ```

- 下面的命令以文本形式显示输出文件中的 **TestVar** （可能发生转义）。

  ```
  OUTPUT TO filename HEXADECIMAL OFF
  ```

如果将 DESCRIBE_JAVA_FORMAT 设置为 **binary**：

- 下面的命令以十六进制格式显示输出文件中的 JavaString。

  ```
  OUTPUT TO filename
  ```

- 下面的命令给出输出文件中实际的 JavaString 对象 （具有转义序列）。

  ```
  OUTPUT TO filename HEXADECIMAL OFF
  ```

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > DESCRIBE_JAVA_FORMAT 选项 [ISQL]"。

权限

无。

副作用

在 Interactive SQL 中， " 结果 " 选项卡仅显示当前查询的结果。所有以前的查询结果都被当前查询结果替换。

另请参阅

"SELECT 语句 " 第 587 页

"INPUT 语句 [Interactive SQL]" 第 515 页

"xp_write_file 系统过程 " 第 813 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

将 employee 表的内容以 ASCII 格式放在文件中：

```
SELECT *
FROM employee;
OUTPUT TO employee.txt
FORMAT ASCII
```

将 employee 表的内容放在现有文件的末尾，并在该文件中包含有关查询的所有信息：

```
SELECT *
FROM employee;
OUTPUT TO employee.txt APPEND VERBOSE
```

将 *JProd* 列中 toString() 方法的内容输出到文件：

```
SELECT JProd>>toString()
FROM jdba.product;
OUTPUT TO d:\temp\temp.txt
FORMAT ASCII HEXADECIMAL OFF
```

假定需要导出含有嵌入式换行符的值。换行符的数值为 10，在 SQL 语句中可以用字符串 "\x0a" 代替该值。如果将 HEXADECIMAL 设为 ON，然后执行下列语句，

```
SELECT 'line1\x0aline2';
OUTPUT TO file.txt HEXADECIMAL ON
```

得到一个文件，其中的一行含有下列文字：

```
line10x0aline2
```

但是，如果将 HEXADEMICAL 设为 OFF 后执行同样的语句，则得到以下文字：

```
line1\x0aline2
```

最后，如果将 HEXADECIMAL 设为 ASIS，得到含有两行的文件：

```
line1
line2
```

使用 ASIS 时得到两行是因为，已导出的嵌入式换行符没有转换成两位数的十六进制形式，也没有使用任何前缀。

PARAMETERS 语句 [Interactive SQL]

说明

此语句用于指定 Interactive SQL 命令文件的参数。

语法

PARAMETERS *parameter1*, *parameter2*, …

用法

PARAMETERS 语句指定命令文件的参数，以便将来在命令文件中引用它们。

引用参数的方法是，将下列语句：

```
{parameter1}
```

放在文件中希望用指定的参数替换的位置。括号和参数名之间不能有空格。

如果用少于所需个数的参数调用命令文件， Interactive SQL 将提示您提供缺少参数的值。

权限

无。

副作用

无。

另请参阅

"READ 语句 [Interactive SQL]" 第 562 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

下面的 Interactive SQL 命令文件带有两个参数。

```
PARAMETERS department_id, file;
SELECT emp_lname
FROM employee
WHERE dept_id = {department_id}
>#{file}.dat;
```

如果将此脚本保存在名为 `test.SQL` 的文件中，可以从 Interactive SQL 中使用下列命令运行它：

```
READ test.SQL [100] [data]
```

PASSTHROUGH 语句 [SQL Remote]

说明

此语句用于启动或停止 SQL Remote 管理所采用的直通模式。第一种和第二种形式启动直通模式，而第三种形式停止直通模式。

语法 1

PASSTHROUGH [**ONLY**] **FOR** *userid*, …

语法 2

PASSTHROUGH [**ONLY**] **FOR SUBSCRIPTION**
TO [**(** *owner* **)**].*publication-name* [**(** *constant* **)**]

语法 3

PASSTHROUGH STOP

用法

在直通模式中，任何 SQL 语句都由数据库服务器执行，并且放在事务日志中，以消息形式发送给预订者。如果使用 ONLY 关键字启动直通模式，则不在服务器上执行这些语句，而仅将它们发送给接收者。直通 SQL 语句的接收者可以是用户 ID 列表 （第一种形式），也可以是预订给定发布的所有预订者。直通模式可用于将统一数据库中的更改应用到远程数据库，或者将远程数据库中的语句发送到统一数据库。

语法 2 将语句发送到已启动预订的远程数据库，它不向已创建预订但未启动预订的远程数据库发送语句。

权限

必须拥有 DBA 权限。

副作用

无。

另请参阅

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_passthrough 过程 "

示例

```
PASSTHROUGH FOR rem_db ;
...
( SQL statements to be executed at the remote database
)
...
PASSTHROUGH STOP ;
```

PREPARE 语句 [ESQL]

说明

此语句用于准备将在以后执行的语句，或者用于定义游标。

语法

PREPARE *statement-name*
　　FROM *statement*
　[**DESCRIBE** *describe-type* **INTO** [[**SQL**] **DESCRIPTOR**] *descriptor*]
　[**WITH EXECUTE**]

statement-name :　*identifier* or *hostvar*

statement :　*string* or *hostvar*

describe-type :
　　[**ALL** | **BIND VARIABLES** | **INPUT** | **OUTPUT** | **SELECT LIST**]
　　[**LONG NAMES** [[[**OWNER.**]**TABLE.**]**COLUMN**]
　　| **WITH VARIABLE RESULT**]

参数

statement-name　语句名称可以是标识符或主机变量。但在使用多个 SQLCA 时不能使用标识符。否则，两个准备的语句可能具有相同的语句号，而这可能导致执行或打开错误的语句。

DESCRIBE 子句　如果使用了 DESCRIBE INTO DESCRIPTOR，准备的语句将在指定的描述符中进行说明。说明类型可能是 DESCRIBE 语句所允许的任何说明类型。

WITH EXECUTE 子句　如果使用了 WITH EXECUTE 子句，则当且仅当此语句不是 CALL 和 SELECT 语句并且不含主机变量时才执行它。成功执行后，此语句立即被删除。如果 PREPARE 和 DESCRIBE （如果有的话）成功，但此语句无法执行，则会设置警告 "SQLCODE 111, SQLSTATE 01W08"，并且不删除此语句。

DESRIBE INTO DESCRIPTOR 和 WITH EXECUTE 子句可以提高性能，因为它们减少了所需的客户 / 服务器通信。

WITH VARIABLE RESULT 子句　WITH VARIABLE RESULT 子句用于描述可能有多个结果集 （具有不同列数或列类型）的过程。

如果使用 WITH VARIABLE RESULT，则数据库服务器在 DESCRIBE 子句之后将 SQLCOUNT 值设置为下列值之一：

- **0** 结果集可能更改：应该在每个 OPEN 语句后重新描述过程调用。
- **1** 结果集是固定的。不需要重新描述。

静态和动态
为了兼容，仍然支持准备 COMMIT、 PREPARE TO COMMIT 和 ROLLBACK 语句。但是，建议用静态嵌入式 SQL 执行所有的事务管理操作，因为某些应用程序环境可能要求使用它。另外，其它嵌入式 SQL 系统不支持动态事务管理操作。

用法

PREPARE 语句准备 *statement* 中指定的 SQL 语句，并使准备的语句与 *statement-name* 关联。在执行语句或者打开游标 （如果是 SELECT 语句的话）时，引用上述的语句名称。 *statement-name* 可能是自动包含的 `sqlca.h` 头文件中定义的 **a_SQL_statement_number** 类型的主机变量。如果用标识符替换 *statement-name*，则每个模块只有一个语句可以用该 *statement-name* 准备。

如果用主机变量替换 *statement-name*，它必须属于 **short int** 类型。在 `sqlca.h` 的 **a_SQL_statement_number** 中有此类型定义。 SQL 预处理器识别此类型，因此可在 DECLARE 节中使用。在执行 PREPARE 语句期间，主机变量由数据库自动填充，不需要程序员进行初始化。

权限

无。

副作用

此前使用同一名称准备的任何语句都会丢失。

仅当使用 WITH EXECUTE 并且成功执行时，使用了此语句后才删除它。在其它情况下，应当确保在使用后删除语句。否则，与此语句关联的内存不回收。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"DESCRIBE 语句 [ESQL]" 第 440 页

"OPEN 语句 [ESQL] [SP]" 第 543 页

"EXECUTE 语句 [ESQL]" 第 465 页

"DROP STATEMENT 语句 [ESQL]" 第 453 页

标准和兼容性

- **SQL/92**　初级特性。
- **SQL/99**　核心特性。
- **Sybase**　受 Open Client/Open Server 支持。

示例

下面的语句准备一个简单查询：

```
EXEC SQL PREPARE employee_statement FROM
'SELECT emp_lname FROM employee';
```

PREPARE TO COMMIT 语句

说明

此语句用于检查 COMMIT 能否成功执行。

语法

PREPARE TO COMMIT

用法

PREPARE TO COMMIT 语句测试能否成功执行 COMMIT。如果不违反数据库完整性就无法执行 COMMIT，则此语句将导致错误。

权限

无。

副作用

无。

另请参阅

"COMMIT 语句 " 第 305 页

"ROLLBACK 语句 " 第 583 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

在下面的语句序列中，由于对 employee 表进行了外键检查，因此导致了错误。

```
EXECUTE IMMEDIATE
   "SET OPTION wait_for_commit = 'on'";
EXECUTE IMMEDIATE "DELETE FROM employee
   WHERE emp_id = 160";
EXECUTE IMMEDIATE "PREPARE TO COMMIT";
```

下面的语句序列在执行删除语句时不会导致错误，即使它导致违反完整性。 PREPARE TO COMMIT 语句返回错误。

```
SET OPTION wait_for_commit= 'ON';
DELETE
FROM department
WHERE dept_id = 100;
PREPARE TO COMMIT;
```

PRINT 语句 [T-SQL]

说明

此语句用于将消息返回到客户端，或者在数据库服务器的消息窗口中显示消息。

语法

PRINT *format-string* [, *arg-list*]

用法

如果从 Open Client 应用程序或 jConnect 应用程序连接，则 PRINT 语句将消息返回到客户端。如果从嵌入式 SQL 或 ODBC 应用程序连接，则消息显示在数据库服务器窗口上。

对于可选参数列表中的参数，格式字符串可以包含占位符。这些占位符的形式为 *%nn!*，其中 *nn* 是介于 1 到 20 之间的整数。

权限

无。

副作用

无。

另请参阅

"MESSAGE 语句 " 第 541 页

标准和兼容性

- **SQL/92**　Transact-SQL 扩展。
- **SQL/99**　Transact-SQL 扩展。
- **Sybase**　受 Adaptive Server Enterprise 支持。

示例

下面的语句显示一条消息：

```
PRINT 'Display this message'
```

下面的语句阐释了如何在 PRINT 语句中使用占位符：

```
DECLARE @var1 INT, @var2 INT
SELECT @var1 = 3, @var2 = 5
PRINT 'Variable 1 = %1!, Variable 2 = %2!', @var1,
@var2
```

PUT 语句 [ESQL]

说明

此语句用于在指定的游标中插入行。

语法

PUT *cursor-name*
[**USING DESCRIPTOR** *sqlda-name* | **FROM** *hostvar-list*]
[**INTO** { **DESCRIPTOR** *into-sqlda-name* | *into-hostvar-list* }]
[**ARRAY** :*nnn*]

cursor-name : *identifier* or *hostvar*

sqlda-name : *identifier*

hostvar-list : may contain indicator variables

用法

在指定的游标中插入行。列的值从第一个 SQLDA 或主机变量列表中获取，与 INSERT 语句中的列 （对于 INSERT 游标）或选择列表中的列（对于 SELECT 游标）是一对一的对应关系。

PUT 语句只能在满足以下条件的 INSERT 或 SELECT 语句的游标上使用：此语句的 FROM 子句引用单个表，或引用由单个基表构成的可更新视图。

如果 SQLDA 中的 **sqldata** 指针为空指针，表明没有为该列指定任何值。如果有 DEFAULT VALUE 与该列关联，则使用缺省值；否则使用 NULL 值。

第二个 SQLDA 或主机变量列表包含 PUT 语句的结果。

可选的 ARRAY 子句可用于执行大范围放入，即一次插入多行，这样可以提高性能。值 **nnn** 是要插入的行数。 SQLDA 必须包含 **nnn * (columns per row)** 变量。第一行放在 SQLDA 变量 0 到 **(columns per row)-1**，依此类推。

插入到游标中
对于滚动 （对值敏感）游标，如果新行与 WHERE 子句匹配并且键集游标未完成填充，则显示插入的行。对于动态游标，如果插入的行与 WHERE 子句匹配，则可能显示该行。不敏感游标不能更新。

有关将 LONG VARCHAR 或 LONG BINARY 值放进数据库的信息，请参阅 ""SET 语句 " 第 592 页 "。

权限

必须有 INSERT 权限。

副作用

如果游标被定义为使用两个或更多个表的连接，则在游标中插入行时可能要求数据库服务器将行插入连接中涉及的多个表中。

另请参阅

"UPDATE 语句 " 第 642 页

"UPDATE （定位）语句 [ESQL] [SP]" 第 647 页

"DELETE 语句 " 第 435 页

"DELETE （定位）语句 [ESQL] [SP]" 第 438 页

"INSERT 语句 " 第 520 页

标准和兼容性

- **SQL/92**　初级特性。
- **SQL/99**　核心特性。
- **Sybase**　受 Open Client/Open Server 支持。

示例

下面的语句阐释了如何在嵌入式 SQL 中使用 PUT：

```
EXEC SQL PUT cur_employee FROM :emp_id, :emp_lname;
```

RAISERROR 语句 [T-SQL]

说明

此语句用于发出错误信号和向客户端发送消息。

语法

RAISERROR *error-number* [*format-string*] [, *arg-list*]

参数

error-number *error-number* 是大于 17000 的五位整数。错误号存储在全局变量 **@@error** 中。

format-string 如果未提供 *format-string*，或者它为空，则使用错误号在系统表中查找错误消息。 Adaptive Server Enterprise 从 SYSMESSAGES 表中获取的消息范围为 17000-19999。在 Adaptive Server Anywhere 中，此表为空视图，因此这个范围内的错误消息应该提供格式字符串。编号为 20000 或更大的错误消息从 SYS.SYSUSERMESSAGES 表中获取。

在 Adaptive Server Anywhere 中， *format-string* 的长度最多可达 255 字节。

Adaptive Server Enterprise 中的 RAISERROR 语句支持的扩展值在 Adaptive Server Anywhere 中不受支持。

对于可选参数列表中的参数，格式字符串可以包含占位符。这些占位符的形式为 **%***nn***!**，其中 *nn* 是介于 1 到 20 之间的整数。

中间 RAISERROR 状态和代码信息在过程终止后会丢失。如果在返回时伴随 RAISERROR 发生了错误，则返回错误信息，而 RAISERROR 信息将丢失。应用程序通过在不同的执行点检查 @@error 全局变量，可以查询中间 RAISERROR 状态。

用法

RAISERROR 语句允许发出用户定义的错误并向客户端发送消息。

权限

无。

副作用

无。

另请参阅

"CREATE TRIGGER 语句 [T-SQL]" 第 415 页

"ASA 数据库管理指南 > 数据库选项 > ON_TSQL_ERROR 选项 [compatibility]"

"ASA 数据库管理指南 > 数据库选项 > CONTINUE_AFTER_RAISERROR 选项 [compatibility]"

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

示例

下面的语句引发错误 23000 （该错误在用户定义的错误范围内）并向客户端发送消息。注意在 *error-number* 和 *format-string* 参数间没有逗号。逗号后的第一项被解释为参数列表中的第一项。

```
RAISERROR 23000 'Invalid entry for this column: %1!',
@val
```

下一个示例使用 RAISERROR 禁止连接。

```
create procedure DBA.login_check()
begin
    // Allow a maximum of 3 concurrent connections
    if( db_property('ConnCount') > 3 ) then
   raiserror 28000
      'User %1! is not allowed to connect -- there are
already %2! users logged on',
      current user,
      cast(db_property('ConnCount') as int)-1;
    else
   call sp_login_environment;
    end if;
end
go
grant execute on DBA.login_check to PUBLIC
go
set option PUBLIC.Login_procedure='DBA.login_check'
go
```

有关禁止连接的其它方法，请参阅 "ASA 数据库管理指南 > 数据库选项 > LOGIN_PROCEDURE 选项 [database]"。

READ 语句 [Interactive SQL]

说明

此语句用于从文件中读取 Interactive SQL 语句。

语法

READ *filename* [*parameters*]

用法

READ 语句从指定的文件中读取一系列 Interactive SQL 语句。此文件可以包含任何有效的 Interactive SQL 语句，包括其它的 READ 语句。READ 语句可以嵌套到任何深度。为查找命令文件， Interactive SQL 首先搜索当前目录，然后在环境变量 **SQLPATH** 中指定的目录中搜索，然后在环境变量 **PATH** 中指定的目录中搜索。如果指定的文件没有文件扩展名，则 Interactive SQL 在每一个目录中搜索带 SQL 扩展名的同名文件。

在命令文件的名称后可以列出参数。这些参数对应于在语句文件开头的 PARAMETERS 语句中指定的参数 （请参阅 ""PARAMETERS 语句 [Interactive SQL]" 第 551 页 "）。 Interactive SQL 在源文件中任何含有

```
{parameter-name}
```

的地方替换对应的参数，其中 *parameter-name* 是相应参数的名称。

传递给命令文件的参数可以是标识符、数字、加引号的标识符或字符串。如果用引号将参数括起来，则在替换时引号也放到文本中。不是标识符、数字或字符串 （包含空格或制表符）的参数都必须用方括号 **([ ])** 括起来。这样可以在命令文件中进行任意文本替换。

如果传递给命令文件的参数不够， Interactive SQL 会提示您提供缺少的参数值。

权限

无。

副作用

无。

另请参阅

"PARAMETERS 语句 [Interactive SQL]" 第 551 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例　　以下是 READ 语句示例。

```
READ status.rpt '160'
READ birthday.SQL [>= '1988-1-1'] [<= '1988-1-30']
```

READTEXT 语句 [T-SQL]

说明

此语句用于从数据库中读取文本值和图像值，从指定的偏移处开始读取指定的字节数。

语法

READTEXT *table-name.column-name*
text-pointer offset size
[**HOLDLOCK**]

用法

READTEXT 用于从数据库中读取图像和文本值。对视图不能执行 READTEXT 操作。

权限

表的 SELECT 权限。

副作用

无。

另请参阅

"WRITETEXT 语句 [T-SQL]" 第 660 页

"GET DATA 语句 [ESQL]" 第 489 页

"TEXTPTR 函数 [文本和图像]" 第 221 页

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

  Adaptive Server Enterprise 支持以下子句 （Adaptive Server Anywhere 不支持它们）：

  ```
  USING { BYTES | CHARS | CHARACTERS }
  ```

  这些选项对所有单字节字符集都是相同的。 Adaptive Server Anywhere 仅使用 **bytes**，这也是 Adaptive Server Enterprise 的缺省设置。

  Adaptive Server Enterprise 还在 READTEXT 语句中提供了隔离级别控制。这在 Adaptive Server Anywhere 中不受支持。

RELEASE SAVEPOINT 语句

说明

此语句用于释放当前事务中的保存点。

语法

RELEASE SAVEPOINT [*savepoint-name*]

用法

释放保存点。 *savepoint-name* 是在当前事务内的 SAVEPOINT 语句上指定的标识符。如果省略 *savepoint-name*，则释放最近的保存点。

释放保存点时并不执行任何类型的 COMMIT 操作。它只是从当前的活动保存点列表中删除保存点。

权限

当前事务中必须有相应的 SAVEPOINT。

副作用

无。

另请参阅

"BEGIN TRANSACTION 语句 " 第 291 页

"COMMIT 语句 " 第 305 页

"ROLLBACK 语句 " 第 583 页

"ROLLBACK TO SAVEPOINT 语句 " 第 584 页

"SAVEPOINT 语句 " 第 586 页

"ASA SQL 用户指南 > 使用事务和隔离级别 > 事务内的保存点 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。通过使用嵌套事务，可以用 Adaptive Server Enterprise 兼容方式实现类似的功能。

REMOTE RESET 语句 [SQL Remote]

说明

在自定义数据库抽取过程中，此语句在单个事务中为远程用户启动所有预订。

语法

REMOTE RESET *userid*

用法

此命令在单个事务中为远程用户启动所有预订。它将 **SYSREMOTEUSER** 表中的 **log_sent** 值和 **confirm_sent** 值设置为事务日志中的当前位置。它还将 **SYSSUBSCRIPTION** 中的创建和启动的值设置为此远程用户的所有预订的事务日志中的当前位置。此语句不执行提交操作。在此调用之后必须执行显式提交。

为写出活动数据库的安全抽取进程，必须在启动预订所使用的同一事务中以隔离级别 3 抽取数据。

此语句是启动预订的替代方法。 START SUBSCRIPTION 的副作用是隐式提交，这样，如果远程用户有多个预订，不可能使用 START SUBSCRIPTION 在一个事务中全部启动它们。

权限

必须拥有 DBA 权限。

副作用

此语句不执行任何自动提交。

另请参阅

"START SUBSCRIPTION 语句 [SQL Remote]" 第 619 页

示例

- 下面的语句重置远程用户 SamS 的预订：

```
REMOTE RESET SamS
```

REMOVE 语句

说明

此语句用于从数据库中删除类或 jar 文件。删除类后，它不能再用作列类型或变量类型。

要删除的类或 jar 文件必须已安装。

语法

REMOVE JAVA *classes_to_remove*

classes_to_remove :
　　class *java_class_name* [, *java_class_name*, …]
　| **jar** *jar_name* [, *jar_name*, …]

参数

CLASS　*java_class_name* 参数是一个或多个要删除的 Java 类的名称。这些类必须是当前数据库中已安装的类。

JAR　*jar_name* 是最大长度为 255 的字符串值。

每一个 *jar_name* 都必须与当前数据库中的某个保留 jar 的 *jar_name* 相等。 *jar_name* 的相等性由 SQL 系统的字符串比较规则确定。

用法

从数据库中删除类或 jar 文件。

权限

必须拥有 DBA 权限。

在 Windows CE 上不受支持。

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不受 Adaptive Server Enterprise 支持。通过使用嵌套事务，可以用 Adaptive Server Enterprise 兼容方式实现类似的功能。

示例

下面的语句从当前数据库中删除名为 Demo 的 Java 类。

```
REMOVE JAVA CLASS Demo
```

REORGANIZE TABLE 语句

说明

当由于需要连续访问数据库而不可能完全重建数据库时，此语句对表做碎片整理。

语法

REORGANIZE TABLE [*owner*.]*table-name*
[{ **PRIMARY KEY**
| **FOREIGN KEY** *foreign_key_name*
| **INDEX** *index_name* }
| **ORDER** {**ON** | **OFF**}
]

参数

PRIMARY KEY 重组表的主键索引。

FOREIGN KEY 重组指定的外键。

INDEX 重组指定的索引。

ORDER 选项 如果 ORDER 设置为 ON （缺省设置），则数据按聚簇索引 （如果有的话）排列。如果没有聚簇索引，则数据按主键值排列。如果将 ORDER 设置为 OFF，则数据按主键排列。

有关聚簇索引的更多信息，请参阅 "ASA SQL 用户指南 > 使用数据库对象 > 使用聚簇索引 "。

用法

表碎片可能使性能下降。此语句用于整理表中的行碎片，或压缩由于 DELETE 操作而变得稀疏的索引。它还可以减少用于存储表及其索引的总页数，并且可以减少索引树中的级别数。但是，它不会使数据库文件的总大小减少。建议使用 sa_table_fragmentation 和 sa_index_density 系统过程选择值得处理的表。

如果未指定索引或键，重组过程通过删除和重新插入行组来整理表中的行碎片。对于每个组获取表的排它锁。处理完组后，立即释放并重新获取锁 （必要时需等待），为其它连接提供访问表的机会。在处理组时，检查点操作被挂起；处理完组后，可能会发生检查点操作。按主键顺序处理行；如果表没有主键，就会产生错误。经过处理的行重新插入到表的末尾，使行在处理结束时按主键聚簇。请注意，无论行最初的破碎程度如何，所需的工作量是相同的。

如果指定了索引或键，则仅处理指定的索引。这种形式的语句只能用于用 Adaptive Server Anywhere 7.0 或更高版本创建的数据库。在操作期间，保持表的排它锁并挂起检查点操作。其他连接对表的任何访问尝试都被阻塞或失败，具体取决于它们的 BLOCKING 选项设置。通过在获取排它锁前预读索引页，可以尽可能缩短锁定期限。

由于两种形式的重组都可能修改许多页，因此检查点日志可能变得很大。对于 7.0 或更早版本的数据库，这可能导致数据库文件增大。对于 8.0 版数据库，这只是导致数据库文件大小暂时增大，因为检查点日志在关闭时会被删除，而数据库文件在这时被截断。另外，对于 8.0 版数据库，可以得到更连续的表页分配。

在事务日志中不记录任何一种形式的语句。

权限

- 必须是表的所有者，或是具有 DBA 权限的用户。

副作用

开始重组之前，会执行一个检查点操作以设法得到尽可能多的空闲页。

示例

下面的示例按主键重组 *employee* 表。

```
REORGANIZE TABLE employee
PRIMARY KEY
```

RESIGNAL 语句

说明

此语句用于重新发出异常情况通知。

语法

RESIGNAL [*exception-name*]

用法

在异常处理程序中， RESIGNAL 允许在异常仍活动时退出复合语句，或停止报告其它指定的异常。异常由其它异常处理程序处理或返回给应用程序。异常处理程序在 RESIGNAL 之前的任何操作都被取消。

权限

无。

副作用

无。

另请参阅

"SIGNAL 语句 " 第 612 页

"BEGIN 语句 " 第 288 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 在过程和触发器中使用异常处理程序 "

"RAISERROR 语句 [T-SQL]" 第 560 页

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。 Transact-SQL 过程中的错误通知是用 RAISERROR 语句完成的。

示例

下面这段代码向应用程序返回除 " 列未找到 " 外的所有异常。

```
...
DECLARE COLUMN_NOT_FOUND EXCEPTION
   FOR SQLSTATE '52003';
...
EXCEPTION
WHEN COLUMN_NOT_FOUND THEN
SET message='Column not found';
WHEN OTHERS THEN
RESIGNAL;
```

RESTORE DATABASE 语句

说明 此语句用于从档案中恢复已备份的数据库。

语法

RESTORE DATABASE *filename*
FROM *archive_root*
[**CATALOG ONLY |**
[[**RENAME** *dbspace_name* **TO** *new_dbspace_name*] ...]]

参数

CATALOG ONLY 子句 检索有关指定档案的信息，并将它们放到备份历史记录文件 (`backup.syb`) 中，但不从档案中恢复任何数据。

RENAME 子句 指定将各 dbspace 恢复到的新位置。

用法

每个 RESTORE DATABASE 操作都更新名为 `backup.syb` 的历史记录文件；该文件是一个文本文件，与数据库服务器可执行文件保存在同一个目录中。

RENAME 子句提供了更改各 dbspace 的恢复位置的方法。 RENAME 子句中的 dbspace 名称不能为 SYSTEM 或 TRANSLOG。

RESTORE DATABASE 替换要恢复的数据库。如果需要增量备份，请使用 BACKUP 命令的映像格式并仅保存事务日志；但是不支持到磁带的映像备份。

权限

执行此语句所需的权限在服务器命令行上用 -gu 选项来设置。缺省设置为要求具有 DBA 权限。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -gu 服务器选项 "。

副作用 无。

另请参阅

"BACKUP 语句 " 第 285 页

"ASA 数据库管理指南 > 备份和数据恢复 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

- **Sybase** 不受 Adaptive Server Enterprise 支持。
- **Windows CE** 在 Windows CE 平台上不支持。

示例

下面的示例从 Windows NT 磁带驱动器中恢复数据库。所需的反斜线数取决于您在执行 RESTORE DATABASE 时连接的数据库。此数据库影响 ESCAPE_CHARACTER 选项的设置。该选项通常为 ON，但在 utility_db 中为 OFF。如果连接除 utility_db 外的任何其它数据库，需要额外的反斜线。

```
RESTORE DATABASE 'd:\\dbhome\\cust.db'
FROM '\\\\.\\tape0'
```

RESUME 语句

说明

此语句用于重新开始执行返回结果集的游标。

语法

RESUME *cursor-name*

cursor-name : *identifier* or *hostvar*

用法

此语句重新执行返回结果集的过程。该过程一直执行，直到遇见下一个结果集（不带 INTO 子句的 SELECT 语句）为止。如果该过程结束时未找到任何结果集，则会设置 SQLSTATE_PROCEDURE_COMPLETE 警告。当使用 RESUME 重新开始 SELECT 语句的游标时，也会设置上述警告。

权限

游标在此前必须已打开。

副作用

无。

另请参阅

"DECLARE CURSOR 语句 [ESQL] [SP]" 第 426 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 从过程返回结果 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。

示例

以下是嵌入式 SQL 的示例。

```
1. EXEC SQL RESUME cur_employee;
2. EXEC SQL RESUME :cursor_var;
```

以下是 Interactive SQL 的示例。

```
CALL sample_proc();
RESUME ALL;
```

RETURN 语句

说明

此语句用于从函数或过程中无条件退出，并在退出时选择提供返回值。

语法

RETURN [*expression*]

用法

RETURN 语句导致从函数或过程中立即退出。如果提供了 *expression*，则 *expression* 的值作为函数或过程的值返回。

RETURN 语句之后的语句不再执行。

在函数中，表达式的数据类型应当与函数的 RETURNS 数据类型相同。

在过程中，使用 RETURN 是为了与 Transact-SQL 兼容，它用于返回整型错误代码。

权限

无。

副作用

无。

另请参阅

"CREATE FUNCTION 语句 " 第 337 页

"CREATE PROCEDURE 语句 " 第 347 页

"BEGIN 语句 " 第 288 页

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** Transact-SQL 过程使用 RETURN 语句返回整型错误代码。

示例

下面的函数返回三个数字的乘积：

```
CREATE FUNCTION product (
   a numeric,
   b numeric,
   c numeric )
RETURNS numeric
BEGIN
   RETURN ( a * b * c );
END
```

计算三个数字的乘积：

```
SELECT product (2, 3, 4)
```

product(2, 3, 4)
24

下面的过程使用 RETURN 语句避免执行复杂查询 （如果这样做没有意义的话）：

```
CREATE PROCEDURE customer_products
( in customer_id integer DEFAULT NULL)
RESULT ( id integer, quantity_ordered integer )
BEGIN
   IF customer_id NOT IN (SELECT id FROM customer)
   OR customer_id IS NULL THEN
      RETURN
   ELSE
      SELECT product.id,sum(
         sales_order_items.quantity )
      FROM  product,
            sales_order_items,
            sales_order
      WHERE sales_order.cust_id=customer_id
      AND sales_order.id=sales_order_items.id
      AND sales_order_items.prod_id=product.id
      GROUP BY product.id
   END IF
END
```

REVOKE 语句

说明

此语句用于删除指定用户的权限。

语法 1

REVOKE *special-priv*, … **FROM** *userid*, …

special-priv :
CONNECT
| **DBA**
| **INTEGRATED LOGIN**
| **GROUP**
| **MEMBERSHIP IN GROUP** *userid*, …
| **RESOURCE**

语法 2

REVOKE *table-priv*, … **ON** [*owner*.]*table-name* **FROM** *userid*, …

table-priv :
ALL [**PRIVILEGES**]
| **ALTER**
| **DELETE**
| **INSERT**
| **REFERENCES** [**(** *column-name*, …**)**]
| **SELECT** [**(** *column-name*, …**)**]
| **UPDATE** [**(** *column-name*, …**)**]

语法 3

REVOKE EXECUTE ON [*owner*.]*procedure-name* **FROM** *userid*, …

用法

REVOKE 语句删除使用 GRANT 语句授予的权限。第一种形式撤消特殊用户权限。第二种形式撤消表权限。第三种形式撤消执行过程所需的权限。 REVOKE CONNECT 从数据库中删除用户 ID，同时销毁该用户所拥有的任何对象 （表、视图、过程等）以及由该用户授予的任何权限。 REVOKE GROUP 对组的所有成员自动执行 REVOKE MEMBERSHIP 操作。

如果授予用户 GRANT 选项权限，然后又撤消此权限，则还会撤消该用户在拥有 GRANT 选项时授予其他用户的任何权限。

权限

必须是要撤消的权限的授予者，或者是有 DBA 权限的用户。

如果要撤消其他用户的连接权限或表权限，则此时该用户不能与数据库相连。无法撤消 DBO 的连接权限。

副作用 自动提交。

另请参阅 "GRANT 语句 " 第 497 页

标准和兼容性

- **SQL/92** 语法 1 为供应商扩展。语法 2 为初级特性。语法 3 是持久存储模块特性。
- **SQL/99** 语法 1 为供应商扩展。语法 2 是核心特性。语法 3 是持久存储模块特性。
- **Sybase** 语法 2 和语法 3 都受 Adaptive Server Enterprise 支持。Adaptive Server Enterprise 不支持语法 1。 Adaptive Server Anywhere 和 Adaptive Server Enterprise 的用户管理和安全模型不同。

示例 禁止用户 Dave 更新 employee 表。

```
REVOKE UPDATE ON employee FROM dave;
```

撤消用户 Jim 的资源权限。

```
REVOKE RESOURCE FROM Jim;
```

撤消用户配置文件名 Administrator 的集成登录映射。

```
REVOKE INTEGRATED LOGIN FROM Administrator;
```

禁止 Finance 组执行 sp_customer_list 过程。

```
REVOKE EXECUTE ON sp_customer_list
FROM finance;
```

从数据库中删除用户 ID FranW。

```
REVOKE CONNECT FROM FranW
```

REVOKE CONSOLIDATE 语句 [SQL Remote]

说明 此语句用于禁止统一数据库从此数据库接收 SQL Remote 消息。

语法 **REVOKE CONSOLIDATE FROM** *userid*, …

用法 必须在远程数据库中授予代表统一数据库的用户 ID CONSOLIDATE 权限。 REVOKE CONSOLIDATE 语句从接收当前数据库中的消息的用户列表中删除统一数据库的用户 ID。

权限 必须拥有 DBA 权限。

副作用 自动提交。删除用户的所有预订。

另请参阅

"REVOKE PUBLISH 语句 [SQL Remote]" 第 579 页

"REVOKE REMOTE 语句 [SQL Remote]" 第 581 页

"REVOKE REMOTE DBA 语句 [SQL Remote]" 第 582 页

"GRANT CONSOLIDATE 语句 [SQL Remote]" 第 502 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_revoke_consolidate 过程 "

示例

- 下面的语句撤消用户 ID **condb** 的统一状态：

```
REVOKE CONSOLIDATE FROM condb
```

REVOKE PUBLISH 语句 [SQL Remote]

说明

此语句用于终止将指定用户 ID 标识为 CURRENT 发布者的操作。

语法

REVOKE PUBLISH FROM *userid*

用法

在出去的消息中， SQL Remote 安装中的每个数据库都由 **publisher** 用户 ID 标识。使用 CURRENT PUBLISHER 特殊常量可以找到当前的发布者用户 ID。下面的查询标识当前发布者：

```
SELECT CURRENT PUBLISHER
```

REVOKE PUBLISH 语句终止将指定用户 ID 标识为发布者的操作。

当数据库含有活动的 SQL Remote 发布或预订时，不要执行 REVOKE PUBLISH 从数据库中撤消发布。

在数据库中发出 REVOKE PUBLISH 语句对 SQL Remote 安装有以下几个方面的影响：

- 在任何以 CURRENT PUBLISHER 列作为主键一部分的表中都无法插入数据。任何出去的消息不再用发布者用户 ID 标识，因此不会被接收者数据库接受。

如果在 SQL Remote 安装中更改了任何统一或远程数据库中的发布者用户 ID，必须确保授予新的发布者用户 ID 所有从该数据库接收消息的数据库的 REMOTE 权限。这通常需要删除并重新创建所有预订。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"GRANT PUBLISH 语句 [SQL Remote]" 第 504 页

"REVOKE REMOTE 语句 [SQL Remote]" 第 581 页

"REVOKE REMOTE DBA 语句 [SQL Remote]" 第 582 页

"REVOKE CONSOLIDATE 语句 [SQL Remote]" 第 578 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_publisher 过程 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

```
REVOKE PUBLISH FROM publisher_ID
```

REVOKE REMOTE 语句 [SQL Remote]

说明 此语句用于禁止用户接收来自该数据库的 SQL Remote 消息。

语法 **REVOKE REMOTE FROM** *userid*, …

用法 必须有 REMOTE 权限，用户 ID 才能接收 SQL Remote 复制安装中的消息。 REVOKE REMOTE 语句从接收当前数据库中的消息的用户列表中删除用户 ID。

权限 必须拥有 DBA 权限。

副作用 自动提交。删除用户的所有预订。

另请参阅

"REVOKE PUBLISH 语句 [SQL Remote]" 第 579 页

"GRANT REMOTE 语句 [SQL Remote]" 第 505 页

"REVOKE REMOTE DBA 语句 [SQL Remote]" 第 582 页

"REVOKE CONSOLIDATE 语句 [SQL Remote]" 第 578 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_revoke_remote 过程 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

```
REVOKE REMOTE FROM SamS
```

REVOKE REMOTE DBA 语句 [SQL Remote]

说明

此语句用于为用户 ID 提供 DBA 权限，但仅在从 " 消息代理 " 中连接时提供。

语法 1

REVOKE REMOTE DBA
FROM *userid*, …

用法

REMOTE DBA 权限使 " 消息代理 " 能够完全访问数据库以使任何更改包含在消息中，同时避免与分发 DBA 用户 ID 口令相关的安全问题。

- 此语句撤消用户 ID 的 REMOTE DBA 权限。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"REVOKE PUBLISH 语句 [SQL Remote]" 第 579 页

"REVOKE REMOTE 语句 [SQL Remote]" 第 581 页

"GRANT REMOTE DBA 语句 [SQL Remote]" 第 508 页

"REVOKE CONSOLIDATE 语句 [SQL Remote]" 第 578 页

"SQL Remote 用户指南 > 管理 SQL Remote for Adaptive Server Anywhere > 消息代理和复制安全性 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

ROLLBACK 语句

说明

此语句用于结束事务，并撤消自上次执行 COMMIT 或 ROLLBACK 以来所做的任何更改。

语法

ROLLBACK [**WORK**]

用法

事务是指在两个 COMMIT 或 ROLLBACK 语句之间，对一个数据库连接完成的逻辑工作单元。 ROLLBACK 语句可结束当前事务，并撤消自前一个 COMMIT 或 ROLLBACK 以来对该数据库所做的所有更改。

权限

无。

副作用

关闭所有未打开为 WITH HOLD 的游标。

另请参阅

"COMMIT 语句 " 第 305 页

"ROLLBACK TO SAVEPOINT 语句 " 第 584 页

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** 受 Adaptive Server Enterprise 支持。

ROLLBACK TO SAVEPOINT 语句

说明 取消自 SAVEPOINT 以来所做的任何更改。

语法 **ROLLBACK TO SAVEPOINT** [*savepoint-name*]

用法 ROLLBACK TO SAVEPOINT 语句撤消自 SAVEPOINT 以来所做的任何更改。在 SAVEPOINT 之前所做的更改不撤消；它们仍然处于等待状态。

savepoint-name 是在当前事务内的 SAVEPOINT 语句中指定的标识符。如果省略 *savepoint-name*，则使用最近的保存点。指定保存点之后的任何保存点都自动释放。

权限 当前事务中必须有相应的 SAVEPOINT。

副作用 无。

另请参阅

"BEGIN TRANSACTION 语句 " 第 291 页

"COMMIT 语句 " 第 305 页

"RELEASE SAVEPOINT 语句 " 第 565 页

"ROLLBACK 语句 " 第 583 页

"SAVEPOINT 语句 " 第 586 页

"ASA SQL 用户指南 > 使用事务和隔离级别 > 事务内的保存点 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 核心 SQL 外部的 SQL/ 基础特性。
- **Sybase** Adaptive Server Enterprise 不支持保存点。若要用 Adaptive Server Enterprise 兼容方式实现类似的功能，可使用嵌套事务。

  有关嵌套事务的更多信息，请参阅 ""BEGIN TRANSACTION 语句 " 第 291 页 "。

ROLLBACK TRIGGER 语句

说明

此语句用于撤消在触发器中所做的任何更改。

语法

ROLLBACK TRIGGER [**WITH** *raiserror-statement*]

用法

ROLLBACK TRIGGER 语句回退在触发器中完成的工作，包括导致触发器被触发的数据修改。

另外，还可以发出 RAISERROR 语句。如果发出 RAISERROR 语句，则会向应用程序返回错误。如果不发出 RAISERROR 语句，则不返回任何错误。

如果在嵌套触发器中使用 ROLLBACK TRIGGER 语句，并且未使用 RAISERROR 语句，则只撤消最里面的触发器以及导致触发器被触发的语句。

权限

无。

副作用

无

另请参阅

"CREATE TRIGGER 语句 " 第 408 页

"ROLLBACK 语句 " 第 583 页

"ROLLBACK TO SAVEPOINT 语句 " 第 584 页

"RAISERROR 语句 [T-SQL]" 第 560 页

标准和兼容性

- **SQL/92**　Transact-SQL 扩展。
- **SQL/99**　Transact-SQL 扩展。
- **Sybase**　受 Adaptive Server Enterprise 支持。

SAVEPOINT 语句

说明

此语句用于在当前事务中建立保存点。

语法

SAVEPOINT [*savepoint-name*]

用法

在当前事务中建立保存点。 *savepoint-name* 是可以用在 RELEASE SAVEPOINT 或 ROLLBACK TO SAVEPOINT 语句中的标识符。事务结束后，所有保存点都自动释放。请参阅 "ASA SQL 用户指南 > 使用事务和隔离级别 > 事务内的保存点 "。

在执行触发器或原子复合语句时建立的保存点在原子操作结束时自动释放。

权限

无。

副作用

无。

另请参阅

"RELEASE SAVEPOINT 语句 " 第 565 页

"ROLLBACK TO SAVEPOINT 语句 " 第 584 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 核心 SQL 外部的 SQL/ 基础特性。
- **Sybase** 不受 Adaptive Server Enterprise 支持。若要用 Adaptive Server Enterprise 兼容方式实现类似的功能，可使用 SAVE TRANSACTION 语句。

SELECT 语句

说明

此语句用于从数据库中检索信息。

语法

SELECT [**ALL** | **DISTINCT**] [**FIRST** | **TOP** *n*] *select-list*
[**INTO** { *hostvar-list* | *variable-list* }]
[**FROM** *table-expression*]
[**WHERE** *search-condition*]
[**GROUP BY** *group-by-list*]
[**HAVING** *search-condition*]
[**ORDER BY** { *expression* | *integer* } [**ASC** | **DESC**], …]
[**FOR** { **UPDATE** [*cursor-concurrency*] | **READ ONLY** }]

select-list :
{ *column-name* | *expression* } [[**AS**] *alias-name*], …| ***

group-by-list :
{ *column-name* | *alias-name* | *function* | *expression* }, …

cursor-concurrency : **BY** { **VALUES** | **TIMESTAMP** | **LOCK** }

参数

ALL 或 DISTINCT　All（缺省值）返回满足 SELECT 语句的子句的所有行。如果指定 DISTINCT，则会消除重复的输出行。指定 DISTINCT 时，很多语句的执行时间显著延长，因此只有在必要时才应使用 DISTINCT。

FIRST 或 TOP　可以只显式检索查询中的第一行或前 *n* 行。这些关键字主要用于 ORDER BY 查询。不要在派生表查询、视图定义或属于涉及 IN、 ANY、 SOME 或 ALL 的定量判定的子查询中使用 FIRST 或 TOP。

有关使用 FIRST 和 TOP 的更多信息，请参阅 "ASA SQL 用户指南 > 对查询结果进行汇总、分组和排序 > FIRST 和 TOP 的使用限制 "。

select list　选择列表是一个由逗号分隔的表达式列表，用于指定从数据库中检索的内容。星号 (*) 表示选择 FROM 子句中的所有表的所有列。

选择列表中允许使用集合函数 （请参阅 ""SQL 函数 " 第 105 页 "）。选择列表中还可以使用子查询（请参阅 "" 表达式 " 第 17 页 "）。每个子查询必须用括号括起来。

在整个查询中可以使用别名来表示带别名的表达式。

别名必须经过定义才能在选择列表中引用。此行为是在 Adaptive Server Anywhere 7.0.2 版中引入的，因此，在 7.0.2 版以前， SELECT x+1 as y, 1 as x 是一个有效的 SELECT 子句。而在 7.0.2 版中，此公式是无效的，必须重写为 SELECT 1 as x, x+1 as y。

Interactive SQL 还在 SELECT 语句的每个输出列的顶部显示别名。如果表达式后面未指定可选别名，则 Interactive SQL 显示表达式本身。

INTO hostvar-list 该子句仅用于嵌入式 SQL。它指定 SELECT 语句结果的位置。选择列表中的每一项都必须有一个主机变量项。选择列表中的项依次放入主机变量中。每个主机变量还可以有一个指示符主机变量，以便程序可以判定选择列表项是否为 NULL。

INTO variable-list 该子句仅用于过程和触发器。它指定 SELECT 语句结果的位置。选择列表中的每一项都必须有一个变量。选择列表中的项依次放入变量中。

FROM 子句 行是从 *table expression* 指定的表和视图中检索的。不带 FROM 子句的 SELECT 语句可用于显示不是从表中派生的表达式的值。例如，

```
SELECT @@version
```

显示全局变量 @@version 的值。这等效于：

```
SELECT @@version
FROM DUMMY
```

有关更多信息，请参阅 "FROM 子句 " 第 484 页。

WHERE 子句 该子句指定将从 FROM 子句指定的表中选择的行。该子句可用于在多个表之间建立连接，它是 ON 短语 （FROM 子句的组成部分）的替代方法。

有关更多信息，请参阅 "" 搜索条件 " 第 26 页 " 和 ""FROM 子句 " 第 484 页 "。

GROUP BY 子句 可以按列、别名或函数进行分组。查询结果中，指定列、别名或函数的每个不同的值集占一行。所有包含空值的行按一个集处理。结果行通常称为组，因为对于表列表中的每组行，结果中均有一行。然后可将集合函数应用于这些组以获得有意义的结果。

如果使用 GROUP BY， *select-list*、 HAVING 子句和 ORDER BY 子句不能引用 GROUP BY 子句中未指定的任何标识符，除了 *select-list* 和 HAVING 子句可以包含集合函数。

HAVING 子句 该子句根据组值而不是各行值来选择行。只有当此语句有 GROUP BY 子句或选择列表只包括集合函数时，才能使用 HAVING 子句。在 HAVING 子句中引用的任何列名必须存在于 GROUP BY 子句中，或用作 HAVING 子句中集合函数的参数。

ORDER BY 子句 该子句将查询结果排序。 ORDER BY 列表中的每一项均可标记为 ASC 以按升序排序 （缺省值）或 DESC 以按降序排序。如果表达式是整数 *n*，则查询结果按选择列表中的第 *n* 项排序。

确保以特定的顺序返回行的唯一方法是使用 ORDER BY。当缺少 ORDER BY 子句时， Adaptive Server Anywhere 以最有效的顺序返回行。这意味着结果集的外观可能随您上次访问行的时间和其它因素的不同而变化。

在嵌入式 SQL 中， SELECT 语句用于从数据库中检索结果，并通过 INTO 子句将值放入主机变量。 SELECT 语句必须只返回一行。对于多行查询，必须使用游标。

FOR 子句 该子句指定是否允许通过在查询中打开的游标进行更新。

如果在 SELECT 语句中未使用 FOR 子句，则由 API 指定可更新性。对于 ODBC 和 OLE DB，缺省值为只读。对于 JDBC、 Open Client 和嵌入式 SQL，缺省值为更新。

该子句替换嵌入式 SQL 中的 DECLARE CURSOR 语句。但是，它可能被其它编程接口中的并发性设置所替换。在 ODBC 和 OLE DB 中，只读缺省设置替换 FOR 子句，但如果将缺省值更改为只读以外的值，则不替换 FOR 子句。在 JDBC 和 Open Client 中，当前设置总是替换 FOR 子句，无论它是否为缺省设置 （可更新游标）。

语句的可更新性取决于 ANSI_UPDATE_CONSTRAINTS 数据库选项的设置，还会考虑语句的其它特征，包括语句包含的是 DISTINCT、 GROUP BY、 HAVING、 UNION、集合函数、连接还是不可更新的视图。

有关游标敏感性的更多信息，请参阅 "ASA 编程指南 > 在应用程序中使用 SQL > Adaptive Server Anywhere 游标 "。

有关 ODBC 并发性的更多信息，请参阅 "ASA 编程指南 > ODBC 编程 > 选择游标特性 " 中对 SQLSetStmtAttr 的探讨。

有关 ANSI_UPDATE_CONSTRAINTS 数据库选项的更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > ANSI_UPDATE_CONSTRAINTS 选项 [compatibility]"。

用法

SELECT 语句用于从数据库中检索结果。

SELECT 语句在 Interactive SQL 中可用于浏览数据库中的数据，或者将数据从数据库导出到外部文件。

SELECT 语句还可在过程和触发器中或嵌入式 SQL 中使用。带 INTO 子句的 SELECT 语句用于在 SELECT 语句只返回一行时从数据库中检索结果。对于多行查询，必须使用游标。

SELECT 语句还可用于从过程中返回结果集。

权限

必须有指定表和视图的 SELECT 权限。

副作用

无。

另请参阅

" 表达式 " 第 17 页

"FROM 子句 " 第 484 页

" 搜索条件 " 第 26 页

"UNION 操作 " 第 636 页

"ASA SQL 用户指南 > 连接：从多个表检索数据 "

标准和兼容性

- **SQL/92** 初级特性。 SELECT 语句的复杂性意味着应该根据标准检查各子句。
- **SQL/99** 核心特性。 SELECT 语句的复杂性意味着应该根据标准检查各子句。
- **Sybase** 受 Adaptive Server Enterprise 支持，但语法有些差异。

示例

有多少雇员？

```
SELECT count(*)
FROM employee
```

列出所有客户及其订单总值。

```
SELECT company_name,
   CAST( sum(sales_order_items.quantity *
   product.unit_price) AS INTEGER) VALUE
FROM customer
   JOIN sales_order
   JOIN sales_order_items
   JOIN product
GROUP BY company_name
ORDER BY VALUE DESC
```

下面的语句显示嵌入式 SQL 中的 SELECT 语句：

```
SELECT count(*) INTO :size
FROM employee
```

SET 语句

说明

此语句用于向 SQL 变量赋值。

语法

SET *identifier = expression*

用法

SET 语句为变量赋新值。以前必须用 CREATE VARIABLE 语句或 DECLARE 语句创建了变量，或者变量必须是过程的 OUPUT 参数。变量名称可以使用以下 Transact-SQL 约定：在名称前使用 @ 符号。例如，

SET *@localvar = 42*

在 SQL 语句中允许使用列名的任何地方都可以使用变量。如果列名和变量名相同，则使用变量值。

变量是当前连接的本地对象，当您与数据库断开连接或使用 DROP VARIABLE 语句时消失。它们不受 COMMIT 或 ROLLBACK 语句的影响。

从嵌入式 SQL 程序中为 INSERT 或 UPDATE 语句创建文本或二进制大对象时需要使用变量，因为嵌入式 SQL 主机变量仅限于 32,767 个字节。

权限

无。

副作用

无。

另请参阅

"CREATE VARIABLE 语句 " 第 417 页

"DECLARE 语句 " 第 425 页

"DROP VARIABLE 语句 " 第 464 页

" 表达式 " 第 17 页

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。

- **Sybase** 不受支持。在 Adaptive Server Enterprise 中，用不带表的 SELECT 语句给变量赋值，这是一种 Transact-SQL 语法，同样受 Adaptive Server Anywhere 的支持。在 Adaptive Server Enterprise 中， SET 语句用于设置数据库选项。

示例

下面这段代码向数据库中插入一个大文本值。

```
EXEC SQL BEGIN DECLARE SECTION;
DECL_VARCHAR( 500 ) buffer;
/* Note: maximum DECL_VARCHAR size is 32765 */
EXEC SQL END DECLARE SECTION;
EXEC SQL CREATE VARIABLE hold_blob LONG VARCHAR;
EXEC SQL SET hold_blob = '';
for(;;) {
   /* read some data into buffer ... */
   size = fread( buffer, 1, 5000, fp );
   if( size <= 0 ) break;
   /* Does not work if data contains null chars */
   EXEC SQL SET hold_blob = hold_blob || :buffer;
}
EXEC SQL INSERT INTO some_table VALUES( 1, hold_blob );
EXEC SQL DROP VARIABLE hold_blob;
```

下面这段代码向数据库中插入一个大二进制值。

```
EXEC SQL BEGIN DECLARE SECTION;
DECL_BINARY( 5000 ) buffer;
EXEC SQL END DECLARE SECTION;
EXEC SQL CREATE VARIABLE hold_blob LONG BINARY;
EXEC SQL SET hold_blob = '';
for(;;) {
   /* read some data into buffer ... */
   size = fread( &(buffer.array), 1, 5000, fp );
   if( size <= 0 ) break;
   buffer.len = size;
   /* add data to blob using concatenation */
   EXEC SQL SET hold_blob = hold_blob || :buffer;
}
EXEC SQL INSERT INTO some_table VALUES ( 1, hold_blob
);
EXEC SQL DROP VARIABLE hold_blob;
```

SET 语句 [T-SQL]

说明

此语句用于为当前连接设置与 Adaptive Server Enterprise 兼容的数据库选项。

语法

SET *option-name option-value*

用法

可用选项如下：

选项名称	选项值
ANSINULL	**ON** | **OFF**
ANSI_PERMISSIONS	**ON** | **OFF**
CLOSE_ON_ENDTRANS	**ON** | **OFF**
DATEFIRST	**1** | **2** | **3** | **4** | **5** | **6** | **7**
QUOTED_IDENTIFIER	**ON** | **OFF**
ROWCOUNT	*integer*
SELF_RECURSION	**ON** | **OFF**
STRING_RTRUNCATION	**ON** | **OFF**
TEXTSIZE	*integer*
TRANSACTION ISOLATION LEVEL	**0** | **1** | **2** | **3**

在 Adaptive Server Anywhere 中，用 SET OPTION 语句设置数据库选项。不过， Adaptive Server Anywhere 还支持用 Adaptive Server Enterprise 中的 SET 语句设置对兼容性特别有用的选项。

在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中，可以用 Transact-SQL SET 语句设置以下选项：

- **SET ANSINULL { ON | OFF }** 在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中，比较值与空值的缺省行为是不同的。将 ANSINULL 设置为 OFF 可提供与 Transact-SQL 兼容的空值比较。

- **SET ANSI_PERMISSIONS { ON | OFF }** 在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中，关于执行含有列引用的 UPDATE 或 DELETE 所需权限的缺省行为是不同的。将 ANSI_PERMISSIONS 设置为 OFF 可提供与 Transact-SQL 兼容的 UPDATE 和 DELETE 权限。

- **SET CLOSE_ON_ENDTRANS { ON | OFF }** 在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中，在事务结束时关闭游标的缺省行为是不同的。将 CLOSE_ON_ENDTRANS 设置为 OFF 可提供与 Transact-SQL 兼容的行为。

- **SET DATEFIRST { 1 | 2 | 3 | 4 | 5 | 6 | 7 }** 缺省值为 7，即在缺省情况下，每周的第一天是星期日。若要永久设置此选项，请参阅 "ASA 数据库管理指南 > 数据库选项 > FIRST_DAY_OF_WEEK 选项 [database]"。

- **SET QUOTED_IDENTIFIER { ON | OFF }** 控制用双引号括起来的字符串是被解释为标识符 (ON) 还是普通字符串 (OFF)。有关此选项的更多信息，请参阅 "ASA SQL 用户指南 > Transact-SQL 兼容性 > 为实现 Transact-SQL 兼容性设置选项 "。

- **SET ROWCOUNT** *integer* Transact-SQL ROWCOUNT 选项将为任何游标读取的行数限制为指定的整数。这包括通过重新定位游标读取的行。超出此最大值的任何读取操作都返回警告。当应 OPEN 请求返回游标的行数估计值时应考虑此选项设置。

  SET ROWCOUNT 还将受搜索的 UPDATE 或 DELETE 语句影响的行数限制为 *integer*。例如，这可以用来允许 COMMIT 语句定期执行，以限制回退日志和锁定表的大小。应用程序（或过程）需要提供一个循环，为未受第一个操作影响的行重新发出更新 / 删除命令。下面给出了一个简单示例：

```
begin
declare @count integer
set rowcount 20
while(1=1) begin
  update employee set emp_lname='new_name'
  where emp_lname <> 'old_name'
  /* Stop when no rows changed */
  select @count = @@rowcount
  if @count = 0 break
  print string('Updated ',
```

```
            @count,' rows; repeating...')
    commit
  end
  set rowcount 0
  end
```

在 Adaptive Server Anywhere 中，如果 ROWCOUNT 设置比 Interactive SQL 能够显示的行数大，则 Interactive SQL 可能会执行一些额外的读取以重新定位游标。因此，实际显示的行数可能比请求的数目少。另外，如果由于截断警告而重新读取任何行，计数可能不准确。

如果值为零，则重置该选项以获取所有行。

- **SET SELF_RECURSION { ON | OFF }** **self_recursion** 选项在触发器中使用，以允许 (ON) 或禁止 (OFF) 与触发器关联的表的操作触发其它触发器。
- **SET STRING_RTRUNCATION { ON | OFF }** 在 Adaptive Server Anywhere 和 Adaptive Server Enterprise 中，指定 SQL 字符串数据时截断非空格字符的缺省行为是不同的。将 STRING_RTRUNCATION 设置为 ON 可提供与 Transact-SQL 兼容的字符串比较。
- **SET TEXTSIZE** 指定选择语句返回的文本或图像类型数据的最大大小（以字节为单位）。**@@textsize** 全局变量存储当前设置。若要重置为缺省大小 (32K)，请使用下面的命令：

  ```
  set textsize 0
  ```

- **SET TRANSACTION-ISOLATION-LEVEL { 0 | 1 | 2 | 3 }** 设置当前连接的锁定隔离级别，详见 "ASA SQL 用户指南 > 使用事务和隔离级别 > 隔离级别和一致性 "。对于 Adaptive Server Enterprise，只有 1 和 3 是有效选项。对于 Adaptive Server Anywhere， 0、 1、 2 或 3 中的任何一个均是有效选项。

此外，出于兼容性考虑， Adaptive Server Anywhere 允许用 SET 语句设置 PREFETCH 选项，但没有效果。

权限 无。

副作用 无。

另请参阅

"SET OPTION 语句 " 第 601 页

"ASA SQL 用户指南 > Transact-SQL 兼容性 > 为实现 Transact-SQL 兼容性设置选项 "

"ASA 数据库管理指南 > 数据库选项 > 兼容性选项 "

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** Adaptive Server Anywhere 支持 Adaptive Server Enterprise 数据库选项的子集。

SET CONNECTION 语句 [Interactive SQL] [ESQL]

说明

此语句用于更改活动数据库连接。

语法

SET CONNECTION [*connection-name*]

connection-name :
identifier, *string,* or *hostvar*

用法

SET CONNECTION 语句将活动数据库连接更改为 **connection-name**。当前连接状态被保存起来，并在重新成为活动连接时恢复。如果省略 **connection-name**，并且存在未命名的连接，则该连接成为活动连接。

当在嵌入式 SQL 中打开游标时，它们与当前连接关联。如果更改连接，前一个活动连接的游标名将变得不可访问。这些游标在原地保持活动，并在关联的连接再次成为活动连接时恢复可访问性。

权限

无。

副作用

无。

另请参阅

"CONNECT 语句 [ESQL] [Interactive SQL]" 第 308 页

"DISCONNECT 语句 [ESQL] [Interactive SQL]" 第 444 页

标准和兼容性

- **SQL/92** Interactive SQL 使用是供应商扩展。嵌入式 SQL 是完整级别特性。
- **SQL/99** Interactive SQL 是供应商扩展。嵌入式 SQL 是核心特性。
- **Sybase** 受 Open Client/Open Server 支持。

示例

以下是嵌入式 SQL 中的示例。

```
EXEC SQL SET CONNECTION :conn_name;
```

在 Interactive SQL 中，将当前连接设置为名为 conn1 的连接。

```
SET CONNECTION conn1;
```

SET DESCRIPTOR 语句 [ESQL]

说明

此语句用于描述 SQL 描述符区中的变量，并将数据放入描述符区。

语法

SET DESCRIPTOR *descriptor-name*
{ **COUNT** = { *integer* | *hostvar* }
| **VALUE** { *integer* | *hostvar* } *assignment* [, …] }

assignment :
{ **TYPE** | **SCALE** | **PRECISION** | **LENGTH** | **INDICATOR** }
= { *integer* | *hostvar* }
| **DATA** = *hostvar*

用法

SET DESCRIPTOR 语句用于描述描述符区中的变量，并将数据放入描述符区。

SET … COUNT 语句设置描述符区中描述的变量数。此数值不能超过分配描述符区时指定的变量数。

{ *integer* | *hostvar* } 值指定描述符区中要赋值的变量。

在执行 SET … DATA 时会进行类型检查，以确保描述符区中的变量类型与主机变量相同。

如果发生错误， SQLCA 中会返回代码。

权限

无。

副作用

无。

另请参阅

"ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页

"DEALLOCATE DESCRIPTOR 语句 [ESQL]" 第 423 页

"ASA 编程指南 > 嵌入式 SQL 编程 > SQL 描述符区域 (SQLDA)"

标准和兼容性

- **SQL/92** 中间级特性。
- **SQL/99** 核心 SQL 外部的 SQL/ 基础特性。
- **Sybase** 受 Open Client/Open Server 支持。

示例

下面的示例设置 sqlda 中位置为 col_num 的列的类型。

```
void set_type( SQLDA *sqlda, int col_num, int new_type
)
{
    EXEC SQL BEGIN DECLARE SECTION;
    int new_type1 = new_type;
    int col = col_num;
    EXEC SQL END DECLARE SECTION;
EXEC SQL SET DESCRIPTOR sqlda VALUE :col TYPE =
:new_type1;
}
```

有关详细示例，请参阅 ""ALLOCATE DESCRIPTOR 语句 [ESQL]" 第 240 页 "。

SET OPTION 语句

说明

此语句用于更改数据库选项的值。

语法

SET [**EXISTING**] [**TEMPORARY**] **OPTION**
[*userid*.| **PUBLIC**.]*option-name* = [*option-value*]

userid : *identifier* | *string* | *hostvar*

option-name : *identifier* | *string* | *hostvar*

option-value : *hostvar* (indicator allowed)
| *string*
| *identifier*
| *number*

用法

SET OPTION 语句用于更改影响数据库服务器行为的选项。设置选项的值可更改所有用户的行为或仅更改个别用户的行为。更改范围可以是临时的，也可以是永久的。

选项类包括：

- 常规数据库选项
- Transact-SQL 兼容性
- 复制数据库选项

有关所有可用选项的列表及其说明，请参阅 "ASA 数据库管理指南 > 数据库选项 "。

可以在三个范围级别上设置选项：公共、用户和临时。临时选项优先于其它选项，而用户选项优先于公共选项。如果为当前用户设置用户级选项，同样会设置相应的临时选项。

如果使用 EXISTING 关键字，则无法为个别用户 ID 设置选项值，除非该选项已经有 **PUBLIC** 用户 ID 设置。

如果指定用户 ID，则选项值适用于该用户 （对于组用户 ID，则适用于该组的成员）。如果指定 **PUBLIC**，则选项值适用于没有单独设置该选项的所有用户。缺省情况下，选项值适用于发出 **SET OPTION** 语句的当前登录用户 ID。

例如，如果 DBA 是发出 SQL 语句的用户，则下面的语句将选项更改应用于用户 DBA：

```
SET OPTION login_mode = mixed
```

而下面的语句将更改应用于 **PUBLIC** 用户 ID，即所有用户所属的用户组。

```
SET OPTION Public.login_mode = standard
```

只有拥有 DBA 特权的用户才有权设置 **PUBLIC** 用户 ID 的选项。

在嵌入式 SQL 中，只能临时设置数据库选项。

用户可以使用 SET OPTION 语句更改他们自己的用户 ID 值。仅当您有 DBA 权限时，才能为您自己以外的用户 ID 设置选项值。

将 TEMPORARY 关键字添加到 SET OPTION 语句中会改变更改生效的持续时间。缺省情况下，选项值是永久性的：除非用 SET OPTION 语句显式更改它，否则它不会改变。

如果 SET TEMPORARY OPTION 语句没有用用户 ID 限定，则新的选项值仅对当前连接生效。

如果对 PUBLIC 用户 ID 使用 SET TEMPORARY OPTION，则更改在数据库运行时间内一直生效。如果数据库关闭，则 PUBLIC 组的 TEMPORARY 选项恢复为它们的永久值。

为 PUBLIC 用户 ID 设置临时选项可提供安全性优点。例如，如果启用了 LOGIN_MODE 选项，则数据库依赖于运行它的系统的登录安全性。临时启用该选项意味着，对于依赖于 Windows 域的安全性的数据库，如果关闭该数据库并将它复制到本地计算机，它的安全不会受到威胁。在这种情况下，临时启用的 LOGIN_MODE 选项恢复为它的永久值，这可能是 " 标准 "（一种不允许集成登录的模式）。

如果省略 *option-value*，将从数据库中删除指定的选项设置。如果它是个人选项设置，则值恢复为 PUBLIC 设置。如果删除 TEMPORARY 选项，该选项设置将恢复为永久设置。

注意
不支持从游标中读取行时更改选项设置，因为这会导致意外的行为。例如，如果从游标中读取时更改 DATE_FORMAT 设置，会导致结果集中各行的日期格式不一致。不要在读取行时更改选项设置。

权限

设置自己的选项不需要任何权限。

为其他用户或 PUBLIC 设置数据库选项需要 DBA 权限。

副作用

如果不指定 TEMPORARY，将执行自动提交。

另请参阅

"ASA 数据库管理指南 > 数据库选项 > 数据库选项 "

"ASA 数据库管理指南 > 数据库选项 > 兼容性选项 "

"ASA 数据库管理指南 > 数据库选项 > 复制选项 "

"SET OPTION 语句 [Interactive SQL]" 第 605 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Adaptive Server Enterprise 支持。对于某些 Adaptive Server Enterprise 选项，Adaptive Server Anywhere 不支持使用 SET 语句。

示例

将日期格式选项设置为 on：

```
SET OPTION public.date_format = 'Mmm dd yyyy';
```

将日期格式选项设置为 off：

```
SET OPTION public.date_format =;
```

将 wait_for_commit 选项设置为 on：

```
SET OPTION wait_for_commit = 'on';
```

下面是两个嵌入式 SQL 示例。

```
1. EXEC SQL SET OPTION :user.:option_name = :value;
2. EXEC SQL SET TEMPORARY OPTION Date_format = 'mm/dd/
yyyy';
```

SET OPTION 语句 [Interactive SQL]

说明

此语句用于更改 Interactive SQL 选项的值。

语法 1

SET [**TEMPORARY**] **OPTION**
[*userid*. | **PUBLIC**.]*option-name* = [*option-value*]

userid :　*identifier*, *string* or *hostvar*

option-name :　*identifier*, *string* or *hostvar*

option-value :　*hostvar* (indicator allowed), *string*, *identifier*, or *number*

语法 2

SET PERMANENT

语法 3

SET

用法

SET PERMANENT （语法 2）在 SYSOPTIONS 系统表中存储所有当前 Interactive SQL 选项。每次为当前用户 ID 启动 Interactive SQL 时都自动建立这些设置。

语法 3 显示所有的当前选项设置。如果为 Interactive SQL 或数据库服务器设置了临时选项，则显示这些选项设置；否则显示永久选项设置。

另请参阅

"ASA 数据库管理指南 > 数据库选项 > Interactive SQL 选项 "

SET REMOTE OPTION 语句 [SQL Remote]

说明

此语句用于设置 SQL Remote 消息链接的消息控制参数。

语法

SET REMOTE *link-name* **OPTION**
[*userid*.| **PUBLIC**.]*link-option-name = link-option-value*

link-name:
file | **ftp** | **mapi** | **smtp** | **vim**

link-option-name:
common-option | *file-option* | *ftp-option*
| *mapi-option* | *smtp-option* | *vim-option*

common-option:
debug | **output_log_send_on_error**
| **output_log_send_limit** | **output_log_send_now**

file-option:
directory

ftp-option:
active_mode | **host** | **password** | **port** | **root_directory** | **user**

mapi-option:
force_download | **ipm_receive** | **ipm_send** | **profile**

smtp-option:
local_host | **pop3_host** | **pop3_password** | **pop3_userid**
| **smtp_host** | **top_supported**

vim-option:
password | **path** | **receive_all** | **send_vim_mail** | **userid**

link-option-value:
string

参数

userid 如果不指定 *userid*，则假定为当前发布者。

Option 值 选项值取决于消息链接。有关更多信息，请参阅下列位置的内容：

- "SQL Remote 用户指南 > SQL Remote 管理 > FILE 消息系统 "。
- "SQL Remote 用户指南 > SQL Remote 管理 > ftp 消息系统 "。

- "SQL Remote 用户指南 > SQL Remote 管理 > MAPI 消息系统 "。
- "SQL Remote 用户指南 > SQL Remote 管理 > SMTP 消息系统 "。
- "SQL Remote 用户指南 > SQL Remote 管理 > VIM 消息系统 "。

用法

消息代理保存第一次使用消息链接时用户在 " 消息链接 " 对话框中输入的消息链接参数。这种情况下，没有必要显式使用此语句。当为抽取许多数据库准备统一数据库时，此语句最有用。

选项名区分大小写。选项值是否区分大小写取决于选项：布尔值不区分大小写，而口令、目录名及其它字符串是否区分大小写取决于文件系统（对于目录名）或数据库 （对于用户 ID 和口令）是否区分大小写。

权限

必须拥有 DBA 权限。发布者可以设置他们自己的选项。

副作用

自动提交。

另请参阅

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_link_option 过程 "

"SQL Remote 用户指南 > SQL Remote 管理 > 解决远程站点中的问题 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

下面的语句将用户 *myuser* 的 ftp 链接的 FTP 主机设置为 `ftp.mycompany.com`：

```
SET REMOTE FTP OPTION myuser.host = 'ftp.mycompany.com'
```

SET SQLCA 语句 [ESQL]

说明

此语句用于指示 SQL 预处理器使用缺省的全局 *sqlca* 以外的 SQLCA。

语法

SET SQLCA *sqlca*

sqlca : *identifier* or *string*

用法

SET SQLCA 语句指示 SQL 预处理器使用缺省的全局 *sqlca* 以外的 SQLCA。 **sqlca** 必须是对 SQLCA 指针的 C 语言引用的标识符或字符串。

当前 SQLCA 指针隐式传递给各嵌入式 SQL 语句中的数据库接口库。C 源文件中位于此语句后面的所有嵌入式 SQL 语句将使用新的 SQLCA。

仅当写再入式代码时才必须使用此语句（请参阅 "ASA 编程指南 > 嵌入式 SQL 编程 > 多线程代码或再入式代码的 SQLCA 管理 "）。 **sqlca** 应该引用局部变量。任何全局或模块静态变量都可能会被另一线程修改。

权限

无。

副作用

无。

另请参阅

"ASA 编程指南 > 嵌入式 SQL 编程 > 多线程代码或再入式代码的 SQLCA 管理 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不受 Open Client/Open Server 支持。

示例

在 Windows DLL 中可以找到 owning 函数。使用该 DLL 的每个应用程序都有自己的 SQLCA。

```
an_SQL_code FAR PASCAL ExecuteSQL( an_application
*app, char *com )
{
   EXEC SQL BEGIN DECLARE SECTION;
   char *sqlcommand;
   EXEC SQL END DECLARE SECTION;
   EXEC SQL SET SQLCA "&app->.sqlca";
```

```
    sqlcommand = com;
    EXEC SQL WHENEVER SQLERROR CONTINUE;
    EXEC SQL EXECUTE IMMEDIATE :sqlcommand;
return( SQLCODE );
}
```

SETUSER 语句

说明

此语句用于允许数据库管理员模拟其他用户和启用连接池。

语法

{ **SET SESSION AUTHORIZATION** | **SETUSER** }
[[**WITH OPTIONS**] *userid*]

参数

WITH OPTIONS 缺省情况下，只更改权限 （包括组成员资格）。如果指定 WITH OPTIONS，生效的数据库选项更改为 *userid* 的当前数据库选项。

用法

提供 SETUSER 语句是为了使数据库管理变得更容易。它使具有 DBA 权限的用户能够模拟数据库的其他用户。

也可以从应用程序服务器中使用 SETUSER 以利用连接池。连接池可减少需要建立的非重复连接的数量，从而提高了性能。

不带用户 ID 的 SETUSER 撤消前面所有的 SETUSER 语句。

在过程、触发器、事件处理程序和批处理中不能使用 SETUSER 语句。

SETUSER 语句有以下几种用途：

- **创建对象** 可以使用 SETUSER 创建由其他用户拥有的数据库对象。
- **权限检查** 通过充当其他用户 （具有他们的权限和组成员资格）， DBA 可以测试查询、过程、视图等的权限和名称解析。
- **为管理员提供更安全的环境** DBA 有权在数据库中执行任何操作。如果想避免不小心执行不想执行的操作，可以使用 SETUSER 切换到具有较少权限的其他用户 ID。

权限

必须拥有 DBA 权限。

另请参阅

"EXECUTE 语句 [SP]" 第 467 页

"GRANT 语句 " 第 497 页

"REVOKE 语句 " 第 576 页

"SET OPTION 语句 " 第 601 页

标准和兼容性

- **SQL/92**　SET SESSION AUTHORIZATION 与 SQL 92 兼容。SETUSER 是供应商扩展。
- **SQL/99**　SET SESSION AUTHORIZATION 是核心特性。SETUSER 是供应商扩展。
- **Sybase**　Adaptive Server Enterprise 支持 SETUSER 但不支持 WITH OPTIONS 关键字。

示例

下面的语句由名为 DBA 的用户执行，它将用户 ID 更改为 Joe，然后更改为 Jane，最后改回 DBA。

```
SETUSER 'Joe'
// ... operations...
SETUSER WITH OPTIONS 'Jane'
// ... operations...
SETUSER
```

SIGNAL 语句

说明

此语句用于发出异常情况的信号。

语法

SIGNAL *exception-name*

用法

SIGNAL 使您能够引发异常。有关如何处理异常的说明，请参阅 "ASA SQL 用户指南 > 使用过程、触发器和批处理 > 在过程和触发器中使用异常处理程序 "。

exception-name 在当前复合语句的开始处用 DECLARE 语句声明的异常的名称。该异常必须对应于系统定义的 SQLSTATE 或用户定义的 SQLSTATE。用户定义的 SQLSTATE 值必须在 99000 到 99999 之间。

权限

无。

副作用

无。

另请参阅

"DECLARE 语句 " 第 425 页

"RESIGNAL 语句 " 第 570 页

"BEGIN 语句 " 第 288 页

"ASA SQL 用户指南 > 使用过程、触发器和批处理 > 在过程和触发器中使用异常处理程序 "

标准和兼容性

- **SQL/92** 持久存储模块特性。
- **SQL/99** 持久存储模块特性。
- **Sybase** Adaptive Server Enterprise 不支持 SIGNAL。

示例

下面的复合语句声明一个用户定义的异常并发出异常信号。如果从 Interactive SQL 中执行此示例，消息显示在 " 消息 " 窗格中。

```
BEGIN
   DECLARE myexception EXCEPTION
   FOR SQLSTATE '99001';
   SIGNAL myexception;
   EXCEPTION
      WHEN myexception THEN
         MESSAGE 'My exception signaled'
         TO CLIENT;
END
```

START DATABASE 语句

说明

此语句用于启动当前数据库服务器上的数据库。

语法

START DATABASE *database-file*
[**AS** *database-name*]
[**ON** *engine-name*]
[**AUTOSTOP** { **ON** | **OFF** }]
[**KEY** *key*]

参数

START DATABASE 子句 *database-file* 参数是一个字符串。如果 *database-file* 中提供了相对路径，则该路径相对于数据库服务器的起始目录。

AS 子句 如果不指定 *database-name*，则为数据库分配缺省名称。此缺省名称是数据库文件的根目录。例如，`C:\Database Files\asademo.db` 文件中的数据库的缺省名称是 **asademo**。

ON 子句 该子句只受 Interactive SQL 支持。在 Interactive SQL 中，如果不指定 *engine-name*，缺省数据库是当前正在运行的服务器中最先启动的那一个。

AUTOSTOP 子句 AUTOSTOP 子句的缺省设置为 ON。如果 AUTOSTOP 设置为 ON，删除最后一个连接时将卸载数据库。如果 AUTOSTOP 设置为 OFF，则不卸载数据库。

在 Interactive SQL 中，可以使用 YES 或 NO 代替 ON 和 OFF。

KEY 子句 如果数据库是高度加密的，使用该子句输入 KEY 值 （口令）。

用法

在当前数据库服务器上启动指定的数据库。

START DATABASE 语句不将当前应用程序连接到指定的数据库：仍需要显式连接。

Interactive SQL 支持 ON 子句，它允许数据库在非当前数据库服务器上启动。

权限

必需的权限由数据库服务器的 -gd 选项指定。该选项在个人数据库服务器上缺省为 **all**，在网络服务器上缺省为 **DBA**。

副作用 无

另请参阅 "STOP DATABASE 语句 " 第 623 页

"CONNECT 语句 [ESQL] [Interactive SQL]" 第 308 页

"ASA 数据库管理指南 > 数据库服务器 > -gd 服务器选项 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例 在当前服务器上启动数据库文件 `C:\Database Files\sample_2.db`。

```
START DATABASE 'c:\database files\sample_2.db'
```

从 Interactive SQL 中，在名为 **sample** 的服务器上启动 **sam2** 中的数据库文件 `c:\Database Files\sample_2.db`。

```
START DATABASE 'c:\database files\sample_2.db'
AS sam2
ON sample
```

START ENGINE 语句 [Interactive SQL]

说明

此语句用于启动数据库服务器。

语法

START ENGINE AS *engine-name* [**STARTLINE** *command-string*]

用法

START ENGINE 语句启动数据库服务器。如果希望为服务器指定一组选项，请与命令字符串一起使用 STARTLINE 关键字。有效的命令字符串是符合 "ASA 数据库管理指南 > 数据库服务器 > 数据库服务器 " 中的数据库服务器说明的命令行字符串。

权限

无

副作用

无

另请参阅

"STOP ENGINE 语句 " 第 625 页

"ASA 数据库管理指南 > 数据库服务器 > 数据库服务器 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

启动名为 sample 的数据库服务器，但不启动其中的任何数据库。

```
START ENGINE AS sample
```

下面的示例显示 STARTLINE 子句的用法。

```
START ENGINE AS eng1 STARTLINE 'dbeng8 -c 8M'
```

START JAVA 语句

说明　此语句用于启动 Java VM。

语法

START JAVA

用法　START JAVA 语句启动 Java VM。主要的用途是在方便的时候装载 Java VM，以便当用户开始使用 Java 功能时，不会出现装载 Java VM 时的初始暂停。

权限　数据库中必须安装 Java，而且数据库必须支持 Java。

副作用　无

另请参阅　"STOP JAVA 语句 " 第 626 页

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　不适用。

示例　启动 Java VM。

```
START JAVA
```

START LOGGING 语句 [Interactive SQL]

说明

此语句用于开始将执行的 SQL 语句记录到日志文件中。

语法

START LOGGING *filename*

用法

START LOGGING 语句开始将所有后面执行的 SQL 语句复制到指定的日志文件中。如果指定的文件不存在， Interactive SQL 将创建它。记录将一直进行，直到您使用 STOP LOGGING 语句显式停止记录或者结束当前的 Interactive SQL 会话。也可以通过单击 "SQL" → " 开始记录 " 和 "SQL" → " 停止记录 " 来开始和停止记录。

权限

无。

副作用

无。

另请参阅

"STOP LOGGING 语句 [Interactive SQL]" 第 627 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

开始记录到 c: 目录下名为 `filename.SQL` 的文件。

```
START LOGGING 'c:\filename.SQL'
```

START SUBSCRIPTION 语句 [SQL Remote]

说明

此语句用于开始用户对发布的预订。

语法

START SUBSCRIPTION
TO *publication-name* [**(** *subscription-value* **)**]
FOR *subscriber-id*, …

参数

publication-name　用户预订的发布的名称。这可以包括发布的所有者。

subscription-value　与发布的预订表达式进行比较的字符串。如果每个预订者可能有多个发布预订，则此处必须使用该值。

subscriber-id　发布预订者的用户 ID。该用户必须有发布的预订。

用法

当发布更新从统一数据库发送到远程数据库时，被称为 SQL Remote 预订**启动**。

START SUBSCRIPTION 语句是一组管理预订的语句之一。 CREATE SUBSCRIPTION 语句定义预订者接收的数据。 SYNCHRONIZE SUBSCRIPTION 语句确保统一数据库和远程数据库之间保持一致。必须有 START SUBSCRIPTION 语句才能启动发送给预订者的消息。

在预订启动之前，预订两端的数据必须保持一致。建议使用数据库抽取实用程序管理预订的创建、同步及启动。如果使用数据库抽取实用程序，则不需要执行显式 START SUBSCRIPTION 语句。此外，预订同步后，消息代理将启动预订。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"CREATE SUBSCRIPTION 语句 [SQL Remote]" 第 367 页

"REMOTE RESET 语句 [SQL Remote]" 第 566 页

"SYNCHRONIZE SUBSCRIPTION 语句 [SQL Remote]" 第 630 页

"SQL Remote 用户指南 > Adaptive Server Enterprise 的命令参考 > sp_subscription 过程 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

下面的语句启动用户 **SamS** 对 **pub_contact** 发布的预订。

```
START SUBSCRIPTION TO pub_contact
FOR SamS
```

START SYNCHRONIZATION DELETE 语句 [MobiLink]

说明

此语句可用于为 MobiLink 同步重新启动有关删除操作的日志记录。

语法

START SYNCHRONIZATION DELETE

用法

通常， Adaptive Server Anywhere 自动记录对属于同步模板的表或列所做的任何更改，并在下一次同步期间将这些更改上载到统一数据库。使用 STOP SYNCHRONIZATION DELETE 语句可以暂时停止自动记录删除操作。使用 START SYNCHRONIZATION DELETE 语句可以重新开始自动记录。

执行 STOP SYNCHRONIZATION DELETE 语句时，不同步在该连接上执行的任何删除操作。效果一直持续到执行 START SYNCHRONIZATION DELETE 语句为止。效果不嵌套，也就是说，第一个 STOP SYNCHRONIZATION DELETE 之后执行的任何操作没有额外的效果。单个 START SYNCHRONIZATION DELETE 语句重新启动记录，无论在它前面执行了多少个 STOP SYNCHRONIZATION DELETE 语句。

权限

必须拥有 DBA 权限。

副作用

无。

另请参阅

"STOP SYNCHRONIZATION DELETE 语句 [MobiLink]" 第 629 页

"UltraLite 用户指南 > C++ API 参考 > StartSynchronizationDelete 方法 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

下面的 SQL 语句序列阐释了如何使用 START SYNCHRONIZATION DELETE 和 STOP SYNCHRONIZATION DELETE。

```
-- Prevent deletes from being sent
-- to the consolidated database
STOP SYNCHRONIZATION DELETE;
-- Remove all records older than 1 month
-- from the remote database,
-- NOT the consolidated database
DELETE FROM PROPOSAL
WHERE last_modified < months( CURRENT TIMESTAMP, -1 )
-- Re-enable all deletes to be sent
-- to the consolidated database
-- DO NOT FORGET to start this
START SYNCHRONIZATION DELETE;
-- Commit the entire operation,
-- otherwise rollback everything
-- including the stopping of the deletes
commit;
```

STOP DATABASE 语句

说明

此语句用于停止当前数据库服务器上的数据库。

语法

STOP DATABASE *database-name*
[**ON** *engine-name*]
[**UNCONDITIONALLY**]

参数

STOP DATABASE 子句 *database-name* 是当前服务器上运行的数据库（不是当前数据库）的名称。

ON 子句 该子句只在 Interactive SQL 中受支持。如果 Interactive SQL 中不指定 *engine-name*，将在所有运行的引擎中搜索具有指定名称的数据库。

如果不在 Interactive SQL 中使用此语句，则仅停止在当前数据库服务器上启动的数据库。

UNCONDITIONALLY 关键字 即使数据库有连接，仍停止数据库。缺省情况下，如果数据库有连接则不停止。

用法

STOP DATABASE 语句停止当前数据库服务器上的指定数据库。

权限

必需的权限由数据库服务器的 -gk 选项指定。该选项在个人数据库服务器上缺省为 **all**，在网络服务器上缺省为 **DBA**。

对当前连接到的数据库不能使用 STOP DATABASE。

副作用

无

另请参阅

"START DATABASE 语句 " 第 614 页

"DISCONNECT 语句 [ESQL] [Interactive SQL]" 第 444 页

"ASA 数据库管理指南 > 数据库服务器 > -gd 服务器选项 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例　　停止当前服务器上名为 sample 的数据库。

```
STOP DATABASE sample
```

STOP ENGINE 语句

说明 此语句用于停止数据库服务器。

语法 **STOP ENGINE** [*engine-name*] [**UNCONDITIONALLY**]

参数 **STOP ENGINE 子句** *engine-name* 只能用于 Interactive SQL。如果不是在 Interactive SQL 中运行此语句，则停止当前的数据库服务器。

UNCONDITIONALLY 关键字 如果您是数据库服务器的唯一连接，则不需要使用 UNCONDITIONALLY。如果还有其它连接，则仅当使用 UNCONDITIONALLY 关键字时，数据库服务器才停止。

用法 STOP ENGINE 语句停止指定的数据库服务器。如果提供 UNCONDITIONALLY 关键字，则数据库服务器即使有其它连接也会停止。缺省情况下，如果有其它连接，则数据库服务器不会停止。

权限 关闭服务器的权限取决于数据库服务器命令行中的 -gk 设置。个人服务器的缺省设置为 **all**，网络服务器的缺省设置为 **DBA**。

副作用 无

另请参阅 "START ENGINE 语句 [Interactive SQL]" 第 616 页

"ASA 数据库管理指南 > 数据库服务器 > -gk 服务器选项 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例 只要没有其它连接，就停止当前的数据库服务器。

```
STOP ENGINE
```

STOP JAVA 语句

说明 此语句用于停止 Java VM。

语法 **STOP JAVA**

用法 STOP JAVA 语句在 Java VM 没有使用时卸载它。主要用途是有效地利用系统资源。

权限 数据库中必须安装 Java，而且数据库必须支持 Java。

副作用 无

另请参阅 "START JAVA 语句 " 第 617 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例 停止 Java VM。

```
STOP JAVA
```

STOP LOGGING 语句 [Interactive SQL]

说明　此语句用于停止记录当前会话中的 SQL 语句。

语法　**STOP LOGGING**

用法　STOP LOGGING 语句阻止数据库服务器将您执行的每个 SQL 语句写入日志文件。可以使用 START LOGGING 语句开始记录。也可以通过单击 "SQL" → " 开始记录 " 和 "SQL" → " 停止记录 " 来开始和停止记录。

权限　无。

副作用　无。

另请参阅　"START LOGGING 语句 [Interactive SQL]" 第 618 页

示例　下面的示例停止当前记录会话。

```
STOP LOGGING
```

STOP SUBSCRIPTION 语句 [SQL Remote]

说明

此语句用于停止用户对发布的预订。

语法

STOP SUBSCRIPTION
TO *publication-name* [**(** *subscription-value* **)**]
FOR *subscriber-id*, …

参数

publication-name 用户预订的发布的名称。这可以包括发布的所有者。

subscription-value 与发布的预订表达式进行比较的字符串。如果每个预订者可能有多个发布预订，则此处必须使用该值。

subscriber-id 发布预订者的用户 ID。该用户必须有发布的预订。

用法

当发布更新从统一数据库发送到远程数据库时，被称为 SQL Remote 预订**启动**。

STOP SUBSCRIPTION 语句阻止向预订者发送更多的消息。需要 START SUBSCRIPTION 语句才能重新启动发送给预订者的消息。但是，应确保在重新启动之前预订已正确同步：即没有丢失任何消息。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"DROP SUBSCRIPTION 语句 [SQL Remote]" 第 456 页

"SYNCHRONIZE SUBSCRIPTION 语句 [SQL Remote]" 第 630 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

下面的语句启动用户 **SamS** 对 **pub_contact** 发布的预订。

```
STOP SUBSCRIPTION TO pub_contact
FOR SamS
```

STOP SYNCHRONIZATION DELETE 语句 [MobiLink]

说明

此语句可用于为 MobiLink 同步暂时停止有关删除操作的日志记录。

语法

STOP SYNCHRONIZATION DELETE

用法

通常， Adaptive Server Anywhere 自动记录对属于同步模板的表或列所做的任何更改，并在下一次同步期间将这些更改上载到统一数据库。使用此语句可以暂时停止记录对 Adaptive Server Anywhere 远程数据库所做的更改。

执行 STOP SYNCHRONIZATION DELETE 语句时，不同步后面对该连接执行的任何删除操作。效果一直持续到执行 START SYNCHRONIZATION DELETE 语句为止。效果不嵌套，也就是说，第一个 STOP SYNCHRONIZATION DELETE 之后执行的任何操作没有额外的效果。单个 START SYNCHRONIZATION DELETE 语句重新启动记录，无论在它前面执行了多少个 STOP SYNCHRONIZATION DELETE 语句。

此命令对纠正远程数据库很有用，但要谨慎使用，因为它能有效地禁止 MobiLink 同步。

权限

必须拥有 DBA 权限。

副作用

无。

另请参阅

"UltraLite 用户指南 > C++ API 参考 > StartSynchronizationDelete 方法 "

"UltraLite 用户指南 > C++ API 参考 > StopSynchronizationDelete 方法 "

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

有关示例，请参阅 "START SYNCHRONIZATION DELETE 语句 [MobiLink]" 第 621 页。

SYNCHRONIZE SUBSCRIPTION 语句 [SQL Remote]

说明

此语句用于同步用户对发布的预订。

语法

SYNCHRONIZE SUBSCRIPTION
TO *publication-name* [**(** *subscription-value* **)**]
FOR *remote-user*, …

参数

publication-name 用户预订的发布的名称。这可以包括发布的所有者。

subscription-value 与发布的预订表达式进行比较的字符串。如果每个预订者可能有多个发布预订，则此处必须使用该值。

remote-user 发布预订者的用户 ID。该用户必须有发布的预订。

用法

当远程数据库中的数据与统一数据库中的数据一致，使得从统一数据库发送到远程数据库的发布更新不会导致冲突和错误时，就是说 SQL Remote 预订已**同步**。

为了同步预订，需要将统一数据库的发布数据的副本发送到远程数据库。 SYNCHRONIZE SUBSCRIPTION 语句通过消息系统完成该操作。如果可以使用数据库抽取实用程序同步预订，则建议不要通过使用消息系统来同步。

权限

必须拥有 DBA 权限。

副作用

自动提交。

另请参阅

"CREATE SUBSCRIPTION 语句 [SQL Remote]" 第 367 页

"START SUBSCRIPTION 语句 [SQL Remote]" 第 619 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

下面的语句同步用户 **SamS** 对 **pub_contact** 发布的预订。

```
SYNCHRONIZE SUBSCRIPTION
   TO pub_contact
   FOR SamS
```

SYSTEM 语句 [Interactive SQL]

说明

此语句用于从 Interactive SQL 内启动可执行文件。

语法

SYSTEM *'* [*path*] *filename '*

用法

启动指定的可执行文件。

- SYSTEM 语句必须全部位于一行中。
- SYSTEM 语句后面不能有注释。
- 将路径和文件名用单引号引起来。

权限

无。

副作用

无。

另请参阅

"CONNECT 语句 [ESQL] [Interactive SQL]" 第 308 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** 不适用。

示例

下面的语句启动记事本程序，假定记事本可执行程序位于您的 PATH 中。

```
SYSTEM 'notepad.exe'
```

TRIGGER EVENT 语句

说明　此语句用于触发指定的事件。事件可以是为事件触发器定义的事件，也可以是调度事件。

语法　**TRIGGER EVENT** *event-name* [**(** *parm* = *value*, … **)**]

参数　**parm = value**　当触发条件导致执行事件处理程序时，数据库服务器可以使用 *event_parameter* 函数为事件处理程序提供上下文信息。使用 TRIGGER EVENT 语句可以显式提供这些参数，以便模拟事件处理程序的上下文。

用法　操作与具体的触发条件或 CREATE EVENT 语句的调度有关。可以使用 TRIGGER EVENT 语句强制执行事件处理程序，即使未发生调度时间或触发条件。 TRIGGER EVENT 不执行已禁用的事件处理程序。

权限　必须拥有 DBA 权限。

副作用　无。

另请参阅　"ALTER EVENT 语句 " 第 248 页

"CREATE EVENT 语句 " 第 326 页

TRUNCATE TABLE 语句

说明

此语句用于删除表中的所有行，但不删除表定义。

语法

TRUNCATE TABLE [*owner*.]*table-name*

用法

TRUNCATE TABLE 语句删除表中的所有行。 TRUNCATE TABLE 语句与不带 WHERE 子句的 DELETE 语句等效，但 TRUNCATE TABLE 语句不导致触发任何触发器，而且每个行删除操作不记入事务日志。

执行 TRUNCATE TABLE 语句后，在发出 DROP TABLE 语句前，表结构及所有索引仍然存在。列定义和约束保持不变，触发器和权限保持生效。

与数据定义语句一样， TRUNCATE TABLE 语句作为单个语句记入事务日志。各删除的行不记入事务日志。

如果 TRUNCATE_WITH_AUTO_COMMIT 选项设置为 ON （缺省设置），而且满足以下所有条件，则执行快速表截断：

- 没有到表的外键，也没有来自表的外键。
- TRUNCATE TABLE 语句不在触发器内执行。
- TRUNCATE TABLE 语句不在原子语句内执行。

如果执行快速截断，在操作之前和之后将执行 COMMIT 操作。

权限

必须是表的所有者，具有 DBA 权限或表的 ALTER 权限。

对于基表， TRUNCATE TABLE 语句需要有表的排它访问权限，因为操作是原子操作 （要么删除所有行，要么不删除任何行）。这意味着所有以前打开的游标和引用要截断的表的游标都必须关闭，并且必须发出 COMMIT 或 ROLLBACK 命令释放对表的引用。

对于临时表，每个用户有自己的数据副本，不需要排它访问。

副作用

TRUNCATE TABLE 语句不触发删除触发器。

如果 TRUNCATE_WITH_AUTO_COMMIT 设置为 ON，在截断表之前和之后将执行 COMMIT 操作。

各行删除操作不记入事务日志，因此不复制 TRUNCATE TABLE 操作。不要在 SQL Remote 复制或 MobiLink 远程数据库中使用此语句。

如果表包含的列被定义为 DEFAULT AUTOINCREMENT 或 DEFAULT GLOBAL AUTOINCREMENT，则 TRUNCATE TABLE 将重置该列的下一个可用值。

另请参阅

"DELETE 语句 " 第 435 页

"ASA 数据库管理指南 > 数据库选项 > TRUNCATE_WITH_AUTO_COMMIT 选项 [database]"

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

示例

删除 **department** 表的所有行。

```
TRUNCATE TABLE department
```

UNION 操作

说明

此语句用于合并两个或更多选择语句的结果。

语法

select-without-order-by
UNION [**ALL**] *select-without-order-by*
[**UNION** [**ALL**] *select-without-order-by*] …
[**ORDER BY** *integer* [**ASC** | **DESC**], …]

用法

使用 UNION 可将多个 SELECT 语句的结果合并成一个较大的结果。各构成的 SELECT 语句在选择列表中必须有相同的项目数，并且不能包含 ORDER BY 子句。

UNION ALL 的结果是合并构成的 SELECT 语句的结果。 UNION 的结果与 UNION ALL 的结果相同，但会删除重复的行。删除重复的行需要额外的处理，所以应尽量使用 UNION ALL 而不是 UNION。

如果两个选择列表中的相应项具有不同的数据类型， Adaptive Server Anywhere 将为结果中的相应列选择数据类型，并自动相应地转换各构成的 SELECT 语句中的列。

如果使用 ORDER BY，则排序依据列表中只能使用整数。这些整数指定存储列的位置。

显示的列名称与第一个 SELECT 语句显示的列名称相同。

权限

必须有各构成的 SELECT 语句的 SELECT 权限。

副作用

无。

另请参阅

"SELECT 语句 " 第 587 页

标准和兼容性

- **SQL/92** 初级特性。
- **SQL/99** 核心特性。
- **Sybase** 受 Adaptive Server Enterprise 支持， Adaptive Server Enterprise 还支持 COMPUTE 子句。

示例

列出雇员和客户的所有不重复的姓。

```
SELECT emp_lname
FROM Employee
UNION
SELECT lname
FROM Customer
```

UNLOAD 语句

说明

此语句用于将数据从数据库导出到外部 ASCII 格式文件。

语法

UNLOAD *select-statement* **TO** *filename-string* [*unload-option* …]

unload-option :
    DELIMITED BY *string*
  | **ESCAPE CHARACTER** *character*
  | **ESCAPES** {**ON** | **OFF**}
  | **FORMAT** {**ASCII** | **BCP**}
  | **HEXADECIMAL** {**ON** | **OFF**}
  | **ORDER** {**ON** | **OFF**}
  | **QUOTES** {**ON** | **OFF**}

参数

filename-string 数据要卸载到的文件名。因为是由数据库服务器执行语句，因此文件名指定数据库服务器计算机上的文件。相对文件名指定相对于数据库服务器起始目录的文件。要将数据卸载到客户端计算机，请参阅 ""OUTPUT 语句 [Interactive SQL]" 第 546 页 "。

用法

UNLOAD 语句允许将查询的结果集导出到以逗号分隔的文件。除非查询本身包含 ORDER BY 子句，否则结果集不排序。

当卸载具有二进制数据类型的结果集列时， UNLOAD 以 *\xnnnn* 形式写入十六进制字符串，其中 *n* 是一个十六进制数字。

有关 *unload-option* 参数的说明，请参阅 ""UNLOAD TABLE 语句 " 第 640 页 "。

在卸载和重装拥有代理表的数据库时，必须创建一个外部登录来将本地用户映射到远程用户，即使用户在本地数据库和远程数据库使用相同的口令。如果您没有外部登录，则重载可能失败，因为您无法连接到远程服务器。

有关外部登录的更多信息，请参阅 "ASA SQL 用户指南 > 访问远程数据 > 使用外部登录 "。

权限

执行 UNLOAD 语句所需的权限在数据库服务器命令行中用 -gl 选项设置。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -gl 服务器选项 "。

副作用

无。查询在当前隔离级别上执行。

另请参阅

"UNLOAD TABLE 语句 " 第 640 页

"OUTPUT 语句 [Interactive SQL]" 第 546 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Enterprise 不支持 UNLOAD。

UNLOAD TABLE 语句

说明

此语句用于将数据从数据库表导出到外部 ASCII 格式文件。

语法

UNLOAD [**FROM**] **TABLE** [*owner.*]*table-name* **TO** *filename-string*
[*unload-option* …]

unload-option :
DELIMITED BY *string*
| **ESCAPE CHARACTER** *character*
| **ESCAPES** {**ON** | **OFF**}
| **FORMAT** {**ASCII** | **BCP**}
| **HEXADECIMAL** {**ON** | **OFF**}
| **ORDER** {**ON** | **OFF**}
| **QUOTES** {**ON** | **OFF**}

参数

filename-string 数据要卸载到的文件名。因为是由数据库服务器执行语句，因此文件名指定数据库服务器计算机上的文件。相对文件名指定相对于数据库服务器起始目录的文件。要将数据卸载到客户端计算机，请参阅 ""OUTPUT 语句 [Interactive SQL]" 第 546 页 "。

ESCAPES 选项 ESCAPES 打开 （缺省值）时，在导出时如有必要将使用反斜线字符组合标识特殊字符。

FORMAT 选项 以 ASCII 格式或 BCP 输出格式输出数据。

HEXADECIMAL 选项 缺省情况下， HEXADECIMAL 设置为 ON。二进制列值以 **0x***nnnnnn*_ 形式写入，其中每个 *n* 是一个十六进制数字。处理多字节字符集时，使用 HEXADECIMAL ON 很重要。

HEXADECIMAL 选项只能与 FORMAT ASCII 选项一起使用。

ORDER 选项 如果 ORDER 设置为 ON （缺省设置），则导出的数据按聚簇索引 （如果有的话）排列。如果没有聚簇索引，则导出的数据按主键值排列。 ORDER 设置为 OFF 时，数据导出顺序与不使用 ORDER BY 子句从表中选择数据时的顺序相同。

ORDER 设置为 ON 时，导出速度较慢。不过，由于创建索引步骤变得简单，使用 LOAD TABLE 语句时重装速度较快。

有关聚簇索引的更多信息，请参阅 "ASA SQL 用户指南 > 使用数据库对象 > 使用聚簇索引 "。

QUOTES 选项 QUOTES 打开 （缺省值）时，所有导出的字符串都用单引号括起来。

用法

UNLOAD TABLE 语句允许从数据库表到 ASCII 文件的高效批量导出。UNLOAD TABLE 比 Interactive SQL 的 OUTPUT 语句效率更高，并且可从任何客户端应用程序调用。

UNLOAD TABLE 在整个表上放置一个排它锁。

当卸载具有二进制数据类型的列时， UNLOAD TABLE 以 **\x***nnnn* 格式写入十六进制字符串，其中 *n* 是一个十六进制数字。

有关 FORMAT、 DELIMITED BY 和 ESCAPE CHARACTER 选项的说明，请参阅 ""LOAD TABLE 语句 " 第 530 页 "。

权限

执行 UNLOAD 语句所需的权限在数据库服务器命令行中用 -gl 选项设置。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -gl 服务器选项 "。

副作用

无。

另请参阅

"LOAD TABLE 语句 " 第 530 页

"OUTPUT 语句 [Interactive SQL]" 第 546 页

"UNLOAD 语句 " 第 638 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Enterprise 不支持 UNLOAD TABLE。Adaptive Server Enterprise 批量复制实用程序 (**bcp**) 提供类似的功能。

UPDATE 语句

说明

此语句用于修改数据库表中的现有行。

语法 1

UPDATE [**FIRST** | **TOP** *n*] *table-list* **SET** *set-item*, …
[**FROM** *table-list*]
[**WHERE** *search-condition*]
[**ORDER BY** *expression* [**ASC** | **DESC**], …]

语法 2

UPDATE *table-list*
SET *set-item*, …
[**VERIFY (** *column-name*, … **) VALUES (** *expression*, … **)**]
[**WHERE** *search-condition*]
[**ORDER BY** *expression* [**ASC** | **DESC**], …]

语法 3

UPDATE *table*
PUBLICATION *publication*
{ **SUBSCRIBE BY** *expression*
| **OLD SUBSCRIBE BY** *expression* **NEW SUBSCRIBE BY** *expression*
}
WHERE *search-condition*

set-item :
column-name [.*field-name*…] **=** *expression*
| *column-name*[.*field-name*…].*method-name***(** [*expression*] **)**
| *@variable-name* **=** *expression*

参数

UPDATE 子句 表是基表、临时表或视图。除非定义视图的 SELECT 语句包含 GROUP BY 子句或集合函数，或者涉及 UNION 操作，否则可以更新视图。

FIRST 或 TOP 子句 主要与 ORDER BY 子句一起使用，该子句允许只更新满足 WHERE 子句的行的特定子集。在 FIRST 或 TOP 子句中，不能将变量用作输入。

SET 子句 如果要更新 Java 列，可以使用 *field-name* 更新列中公共字段的值。也可以使用其它方法设置该值。下面的子句使用一种方法更新 *JProd* 列的名称字段：

```
SET JProd.setName( 'Tank Top' )
```

如果要更新非 Java 列， SET 子句的形式如下：

```
SET column-name = expression, ...
```

和 / 或

```
SET @variable-name = expression, ...
```

每个指定的列被设置为等号右边表达式的值。对 *expression* 没有任何限制。如果表达式为 *column-name*，则使用旧值。为变量赋值时，变量必须已经声明，而且变量名必须以 "at" 符 (@) 开头。变量和列赋值可以混合在一起，并且可以使用任意数字。如果 SET 列表中赋值号左边的名称不仅与变量名匹配，还与更新表中的某列匹配，此语句将更新该列。

以下是部分 UPDATE 语句的示例。它不仅更新表，还为变量赋值：

```
UPDATE T SET @var = expression1, col1 = expression2
WHERE...
```

这等效于：

```
SELECT @var = expression1
FROM T
WHERE... ;
UPDATE T SET col1 = expression2
WHERE...
```

FROM 子句 可选 FROM 子句允许根据连接更新表。如果存在 FROM 子句， WHERE 子句将限定 FROM 子句的行。仅更新 UPDATE 子句的表列表中的数据。

如果使用 FROM 子句，在语句的两个部分必须以同样的方式限定表名。如果在一个位置使用了相关名，则在另一个位置也必须使用该相关名，否则将发生错误。

仅当 ANSI_UPDATE_CONSTRAINTS 设置为 OFF 时，才能使用该子句。请参阅 "ASA 数据库管理指南 > 数据库选项 > ANSI_UPDATE_CONSTRAINTS 选项 [compatibility]"。

有关连接的完整说明，请参阅 "ASA SQL 用户指南 > 连接：从多个表检索数据 "。

有关更多信息，请参阅 "FROM 子句 " 第 484 页。

WHERE 子句 如果指定 WHERE 子句，只更新满足搜索条件的行。如果不指定 WHERE 子句，则更新所有行。

ORDER BY 子句 通常情况下，行以什么顺序更新并不重要。不过，与 FIRST 或 TOP 子句一起使用时，顺序非常重要。

除非将 ANSI_UPDATE_CONSTRAINTS 选项设置为 OFF，否则不能更新 ORDER BY 子句中出现的列。请参阅 "ASA 数据库管理指南 > 数据库选项 > ANSI_UPDATE_CONSTRAINTS 选项 [compatibility]"。

区分大小写 插入到表中的字符串始终以它们插入时的大小写形式存储，无论数据库是否区分大小写。用字符串 **Value** 更新的 CHAR 数据类型列在数据库中的形式始终是 V 为大写字母，其它字母均为小写。SELECT 语句返回字符串作为 **Value**。但是，如果数据库不区分大小写，所有的比较均将 **Value** 视为与 **value**、 **VALUE** 等相同。而且，如果单列主键已经含有 **Value** 条目，将会拒绝 **value**，因为它会导致主键不唯一。

使行保持不变的更新 如果新值与旧值没有差别，则不更改数据。但是，每次行上发生 UPDATE （更新）时，无论新值是否与旧值相同，都触发 BEFORE UPDATE 触发器。 AFTER UPDATE 触发器仅在新值不同于旧值时才触发。

用法

UPDATE 语句的语法 1 修改一个或多个表中的行值。语法 2 和语法 3 仅适用于 SQL Remote。

语法 2 仅用于 SQL Remote 中由消息代理执行的单行更新。 VERIFY 子句包含在要更新的行中应存在的值集。如果值不匹配，则在继续 UPDATE 操作之前，将触发任何 RESOLVE UPDATE 触发器。如果只是 VERIFY 子句不匹配， UPDATE 不会失败。

UPDATE 语句的语法 3 用于实现特定的 SQL Remote 特性，并且在 BEFORE 触发器内使用。

每当列表更改时，它都提供 SUBSCRIBE BY 值的完整列表。它被放在 SQL Remote 触发器中，以便数据库服务器能够计算 SUBSCRIBE BY 值的当前列表。这两个列表都放在事务日志中。

消息代理使用这两个列表来确保将行移到任何没有行而且现在需要行的远程数据库。消息代理还从任何含有行而且不再需要行的远程数据库中删除行。 UPDATE 语句不影响含有行而且仍然需要行的远程数据库。

对于在 SUBSCRIBE BY 子句中用子查询创建的发布，必须编写包含 UPDATE 语句的语法 3 的触发器，以确保各行保留在它们正确的预订中。

UPDATE 语句的语法 3 允许显式指定旧 SUBSCRIBE BY 列表和新 SUBSCRIBE BY 列表，这样可使 SQL Remote 触发器的效率更高。如果缺少这些列表，数据库服务器将计算发布定义中的旧 SUBSCRIBE BY 列表。由于新 SUBSCRIBE BY 列表与旧 SUBSCRIBE BY 列表通常只是稍微有点差别，因此可能会执行两次旧列表的计算。通过指定旧列表和新列表两者，可以避免这种额外的工作。

SUBSCRIBE BY 表达式是一个值或一个子查询。

UPDATE 语句的语法 3 在事务日志中创建项，但不更改数据库表。

权限

必须有要修改的列的 UPDATE 权限。

副作用

无。

另请参阅

"DELETE 语句 " 第 435 页

"INSERT 语句 " 第 520 页

"FROM 子句 " 第 484 页

"ASA SQL 用户指南 > 连接：从多个表检索数据 "

标准和兼容性

- **SQL/92** 除 FROM 和 ORDER BY 子句是供应商扩展外，语法 1 是初级特性。语法 2 和语法 3 是仅用于 SQL Remote 的供应商扩展。
- **SQL/99** 除 FROM 和 ORDER BY 子句是供应商扩展外，语法 1 是核心特性。语法 2 和语法 3 是仅用于 SQL Remote 的供应商扩展。

  若要强制实现与 SQL/92 的兼容，请确保将 ANSI_UPDATE_CONSTRAINTS 选项设置为 STRICT。

  有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > ANSI_UPDATE_CONSTRAINTS 选项 [compatibility]"。

- **Sybase** 取决于表达式的兼容性， Adaptive Server Enterprise 和 Adaptive Server Anywhere 之间兼容 UPDATE 语句的语法 （语法 1）。但语法 2 和语法 3 不支持。

示例

将雇员 Philip Chin （雇员 129）从销售部转移到市场部。

```
UPDATE employee
SET dept_id = 400
WHERE emp_id = 129;
```

销售订单通常从 ID 2001 开始。通过从 ID 中减去 2000，将所有现有的销售订单重新编号。

```
UPDATE sales_order AS orders
SET orders.id = orders.id - 2000
ORDER BY items.id ASC
```

仅当使用 ON UPDATE CASCADE 操作定义 *sales_order_items* 表 （引用主健 *sales_order.id*）的外键时，才能进行此更新。 *sales_order_items* 表同样也会更新。

有关外键属性的更多信息，请参阅 ""ALTER TABLE 语句 " 第 272 页 " 和 ""CREATE TABLE 语句 " 第 395 页 "。

UPDATE （定位）语句 [ESQL] [SP]

说明

此语句用于修改游标当前位置处的数据。

语法 1

UPDATE WHERE CURRENT OF *cursor-name*
{ **USING DESCRIPTOR** *sqlda-name* | **FROM** *hostvar-list* }

语法 2

UPDATE *table-list*
SET *set-item*, …
WHERE CURRENT OF *cursor-name*

hostvar-list : indicator variables allowed

set-item :
column-name [.*field-name*…] = *expression*
| *column-name* [.*field-name*…].*method-name***(** [*expression*] **)**

sqlda-name : *identifier*

参数

SET 子句　*set-item* 中引用的列必须位于更新的基表中。它们不能引用别名或其它表或视图中的列。如果要更新的表在游标说明中被赋予了一个相关名，那么在 SET 子句中必须使用此相关名。

SET 子句右边的表达式可以使用常量、变量、查询选择列表中的表达式、或以上各元素的组合 （使用 +、 - …、 COALESCE、 IF 等运算符）。表达式不能包含集合函数、子查询或子选择。

用法

这种形式的 UPDATE 语句更新指定游标的当前行。当前行被定义为从游标中最近成功读取的一行，对游标的最后一项操作不能是一个定位 DELETE 语句。

在语法 1 中， SQLDA 中的列或主机变量列表中的值与指定游标返回的列一一对应。如果 SQLDA 中的 **sqldata** 指针为空指针，则不更新相应的选择列表项。

在语法 2 中，在指定查询的当前行中，请求的列被设置为该行的指定值。这些列不必位于指定的打开游标的选择列表中。这种格式是可以准备的。

USING DESCRIPTOR、 FROM *hostvar-list*、和 *hostvar* 格式仅用于嵌入式 SQL。

权限

必须有所修改的列的 UPDATE 权限。

副作用

无。

另请参阅

"DELETE 语句 " 第 435 页

"DELETE （定位）语句 [ESQL] [SP]" 第 438 页

"UPDATE 语句 " 第 642 页

标准和兼容性

- **SQL/92** 初级特性。如果 ANSI_UPDATE_CONSTRAINTS 选项设置为 OFF，可以更新的游标的范围可能包含供应商扩展。
- **SQL/99** 核心特性。如果 ANSI_UPDATE_CONSTRAINTS 选项设置为 OFF，可以更新的游标的范围可能包含供应商扩展。
- **Sybase** Open Client/Open Server 支持使用嵌入式 SQL， Adaptive Server Anywhere 支持使用过程和触发器。

示例

下面是 UPDATE 语句 WHERE CURRENT OF 游标的示例：

```
UPDATE Employee
SET emp_lname = 'Jones'
WHERE CURRENT OF emp_cursor;
```

UPDATE 语句 [SQL Remote]

说明

此语句用于修改数据库中的数据。

语法 1

UPDATE *table-list*
    SET *column-name* = *expression*, …
    [**VERIFY (** *column-name*, … **) VALUES (** *expression*, … **)**]
    [**WHERE** *search-condition*]
    [**ORDER BY** *expression* [**ASC** | **DESC**], …]

语法 2

UPDATE *table*
    PUBLICATION *publication*
    { **SUBSCRIBE BY** *expression* |
      OLD SUBSCRIBE BY *expression*
      NEW SUBSCRIBE BY *expression* }
    WHERE *search-condition*

expression: value | *subquery*

用法

语法 1 和语法 2 仅适用于 SQL Remote。

不带 OLD 和 NEW SUBSCRIBE BY 表达式的语法 2 必须在 BEFORE 触发器中使用。

带 OLD 和 NEW SUBSCRIBE BY 表达式的语法 2 可在任何地方使用。

UPDATE 语句用于修改一个或多个表中的行。每个指定的列被设置为等号右边表达式的值。对 *expression* 没有任何限制。表达式中甚至可以使用 *column-name*，将使用旧值。

如果不指定 WHERE 子句，则更新所有行。如果指定 WHERE 子句，则仅更新满足搜索条件的行。

通常情况下，以什么顺序更新行并不重要。但是，与 NUMBER(*) 函数一起使用时，排序可用于按指定的顺序在行中添加递增的编号。此外，如果想执行类似向表的主键值中添加 1 这样的操作，必须按主健的递减顺序操作，以便在操作中不会得到重复的主健。

如果定义视图的 SELECT 语句不包含 GROUP BY 子句、集合函数或不涉及 UNION 操作，则可以更新视图。

插入到表中的字符串始终以它们插入时的大小写形式存储，无论数据库是否区分大小写。因此，用字符串 **Value** 更新的字符数据类型列的形式始终是 V 为大写字母，其它字母均为小写。 SELECT 语句返回字符串作为 **Value**。但是，如果数据库不区分大小写，则所有的比较将 **Value** 视为与 **value**、 **VALUE** 等相同。而且，如果单列主键已经含有 **Value** 条目，将会拒绝 **value**，因为它会导致主键不唯一。

可选 FROM 子句允许根据连接更新表。如果存在 FROM 子句，WHERE 子句将限定 FROM 子句的行。仅更新表列表中紧跟在 UPDATE 关键字后面的数据。

如果使用 FROM 子句，在语句的两个部分必须以相同的方式限定要更新的表名。如果在一个位置使用了相关名，则在另一个位置也必须使用该相关名，否则将发生错误。

语法 1 仅用于 SQL Remote 中由消息代理执行的单行更新。 VERIFY 子句包含在要更新的行中应存在的值集。如果值不匹配，则在继续 UPDATE 操作之前，将触发任何 RESOLVE UPDATE 触发器。如果只是 VERIFY 子句不匹配， UPDATE 不会失败。

语法 2 仅用于 SQL Remote。如果不使用 OLD 和 NEW 表达式，它必须在 BEFORE 触发器内使用，以便能够访问相关的值。其目的是每当列表更改时都提供 SUBSCRIBE BY 值的完整列表。它被放在 SQL Remote 触发器中，以便数据库服务器能够计算 SUBSCRIBE BY 值的当前列表。这两个列表都放在事务日志中。

消息代理使用这两个列表来确保将行移到任何没有行而且现在需要行的远程数据库。消息代理还从任何含有行而且不再需要行的远程数据库中删除行。 UPDATE 语句不影响含有行而且仍然需要行的远程数据库。

UPDATE 语句的语法 2 允许显式指定旧 SUBSCRIBE BY 列表和新 SUBSCRIBE BY 列表，这样可使 SQL Remote 触发器的效率更高。如果缺少这些列表，数据库服务器将计算发布定义中的旧 SUBSCRIBE BY 列表。由于新 SUBSCRIBE BY 列表与旧 SUBSCRIBE BY 列表通常只是稍微有点差别，因此可能会执行两次旧列表的计算。通过指定旧列表和新列表两者，可以避免这种额外的工作。

当表达式用相同的预订更新同一触发器内的许多表时， OLD 和 NEW SUBSCRIBE BY 语法尤其有用。这可以动态提高性能。

SUBSCRIBE BY 表达式是一个值或一个子查询。

UPDATE 语句的语法 2 用于实现特定的 SQL Remote 特性，并且在 BEFORE 触发器内使用。

对于在预订表达式中用子查询创建的发布，必须编写包含 UPDATE 语句语法 2 的触发器，以便确保各行保留在它们正确的预订中。

有关此特性的完整说明，请参阅 "SQL Remote 用户指南 > 适用于 Adaptive Server Anywhere 的 SQL Remote 设计 > Contact 示例中的范围重新校准 "。

UPDATE 语句的语法 2 在事务日志中创建项，但不更改数据库表。

权限

必须有要修改的列的 UPDATE 权限。

副作用

无。

另请参阅

"CREATE TRIGGER 语句 [SQL Remote]" 第 412 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。

示例

- 将雇员 Philip Chin （雇员 129）从销售部转移到市场部。

```
UPDATE employee
VERIFY( dept_id ) VALUES( 300 )
SET dept_id = 400
WHERE emp_id = 129
```

VALIDATE INDEX 语句

说明

此语句用于校验索引。

语法

VALIDATE INDEX [[*owner*.]*table-name*.] { *index-name* | *table-name* }

用法

确保表中确实存在索引中引用的各行。对于外键索引，还要确保主表中存在相应的行。这种检查补充了由 VALIDATE TABLE 语句执行的有效性检查。

index-name | table-name 如果提供 *table-name* 而不是 *index-name*，将校验主键索引。

权限

必须是创建索引的表的所有者、具有 DBA 权限或具有 REMOTE DBA 权限 (SQL Remote)。

副作用

无。

另请参阅

"CREATE INDEX 语句 " 第 341 页

"VALIDATE TABLE 语句 " 第 653 页

"ASA 数据库管理指南 > 数据库管理实用程序 > Validation 实用程序 "

VALIDATE TABLE 语句

说明

此语句用于校验数据库中的表。

语法

VALIDATE TABLE [*owner*.]*table-name*
[**WITH** { **DATA** | **EXPRESS** | **FULL** | **INDEX** } **CHECK**]

参数

WITH DATA CHECK　如果您有 LONG BINARY、 LONG VARCHAR、 TEXT 或 IMAGE 项，它们可能跨越多个数据库页。除缺省检查外，此选项还指示数据库服务器检查上述各项使用的所有页。

WITH EXPRESS CHECK　除缺省检查和 WITH DATA 检查外，还检查表中的行数与索引中的总行数是否匹配。该选项不分别为每行执行索引查找。当使用小缓存校验大数据库时，此选项可以显著提高校验性能。

WITH FULL CHECK　除缺省检查外，还执行 DATA CHECK 和 INDEX CHECK。

WITH INDEX CHECK　除缺省检查外，还校验表的每个索引。有关索引校验的信息，请参阅 ""VALIDATE INDEX 语句 " 第 652 页 "。

用法

如果没有其它选项， VALIDATE TABLE 将扫描表的每一行。对于索引中的每项，它检查该项的起始数据库页的有效性，并检查行的项是否存在于正确的索引中。对于表中的每一个索引， VALIDATE TABLE 语句还确保该索引引用的行数不大于表中的行数。

此缺省校验足够满足大多数要求。还提供了用于额外校验的选项，这些选项在少数情况可能有帮助。这些额外的检查可能会显著增加校验所需的时间，具体取决于数据库的内容。

如果表已损坏，则会报告错误。如果确实报告了错误，可以删除表的所有索引和键，然后重新创建它们。还需要重新创建表的外键。解决 VALIDATE TABLE 所报告的错误的另一种方法是卸载然后重装整个数据库。应该使用 DBUNLOAD 的 -u 选项，以便它不会尝试使用可能损坏的索引对数据进行排序。

权限

必须是表的所有者、具有 DBA 权限或具有 REMOTE DBA 权限 (SQL Remote)。

副作用

无。

另请参阅

"ASA 数据库管理指南 > 数据库管理实用程序 > Validation 实用程序 "

"VALIDATE INDEX 语句 " 第 652 页

"sa_validate 系统过程 " 第 795 页

标准和兼容性

- **SQL/92** 供应商扩展。
- **SQL/99** 供应商扩展。
- **Sybase** Adaptive Server Enterprise 不支持 VALIDATE TABLE。**dbcc checktable** 过程提供类似的功能。
- **WITH EXPRESS CHECK 选项** 该选项只受用 Adaptive Server Anywhere 7.0 版或更高版本创建的数据库的支持。

WAITFOR 语句

说明

此语句用于将当前连接的处理推迟指定的时间长度或推迟到特定的时间。

语法

WAITFOR { **DELAY** *time* | **TIME** *time* }

time:　*string*

用法

如果使用 DELAY，处理将暂停特定的时间长度。如果指定 TIME，处理将一直暂停到服务器时间到达指定的时间。

如果当前服务器时间大于指定的时间，则处理将一直暂停到第二天的这个时间。

WAITFOR 提供了替代以下语句的另一种方法，对于数据库中无法使用 Java 的客户可能很有用：

```
call java.lang.Thread.sleep(
<time_to_wait_in_millisecs> )
```

在许多情况下，使用调度事件比使用 WAITFOR TIME 要好，因为调度事件在它们自己的连接中执行。

权限

无

副作用

此语句的实现在等待时使用工作线程。这会使用由 -gn 数据库选项指定的至少一个线程 （缺省为 20 个线程）。

另请参阅

"CREATE EVENT 语句 " 第 326 页

标准和兼容性

- **SQL/92**　供应商扩展。
- **SQL/99**　供应商扩展。
- **Sybase**　Adaptive Server Enterprise 也实现此语句。

示例

下面的示例等待 3 秒钟：

```
WAITFOR DELAY '00:00:03'
```

下面的示例等待 0.5 秒钟 （500 毫秒）：

```
WAITFOR DELAY '00:00:00:500'
```

下面的示例一直等到晚上 8 点：

```
WAITFOR TIME '20:00'
```

WHENEVER 语句 [ESQL]

说明

此语句用于指定嵌入式 SQL 程序中的错误处理。

语法

WHENEVER { **SQLERROR** | **SQLWARNING** | **NOTFOUND** }
    GOTO *label* | **STOP** | **CONTINUE** | { *C-code;* }

label : *identifier*

用法

WHENEVER 语句用于捕获数据库在处理 SQL 语句时遇到的错误、警告和异常情况。可将此语句放置在嵌入式 SQL 程序中的任何位置，它不生成任何代码。预处理器将在每个连续的 SQL 语句后生成代码。错误操作对 WHENEVER 语句源行中的所有嵌入式 SQL 语句一直保持有效，直到发生下一个具有相同错误条件的 WHENEVER 语句或者直到源文件末尾。

基于源文件位置的错误
错误条件的生效依据是 C 语言源文件的位置，而不是语句的执行时间。

缺省操作为 CONTINUE。

请注意，提供此语句是为了在简单程序中方便使用。多数情况下，检查错误的最简单方法是直接检查 SQLCA (SQLCODE) 的 sqlcode 字段。这种情况下不使用 WHENEVER 语句。事实上，所有 WHENEVER 语句均使预处理器在每个语句后生成 *if (SQLCODE)* 测试。

权限

无。

副作用

无。

标准和兼容性

- **SQL/92**　初级特性。
- **SQL/99**　核心特性。
- **Sybase**　受 Open Client/Open Server 支持。

示例

以下是 WHENEVER 语句的示例：

```
EXEC SQL WHENEVER NOTFOUND GOTO done;
EXEC SQL WHENEVER SQLERROR
   {
      PrintError( &sqlca );
      return( FALSE );
   };
```

WHILE 语句 [T-SQL]

说明

此语句用于重复执行语句或复合语句。

语法

WHILE *search-condition-statement*

用法

除非各语句被组织成位于关键字 BEGIN 和 END 之间的复合语句，否则 WHILE 条件只影响单个 SQL 语句的执行。

BREAK 语句和 CONTINUE 语句可用于控制复合语句中的语句执行。BREAK 语句终止循环，执行在标记循环结束的 END 关键字后重新开始。 CONTINUE 语句使 WHILE 循环重新开始，跳过 CONTINUE 后面的任何语句。

权限

无。

副作用

无。

另请参阅

"LOOP 语句 " 第 539 页

标准和兼容性

- **SQL/92**　Transact-SQL 扩展。
- **SQL/99**　Transact-SQL 扩展。
- **Sybase**　受 Adaptive Server Enterprise 支持。

示例

下面的代码阐释了 WHILE 的用法：

```
WHILE ( SELECT AVG(unit_price) FROM product ) < $30
BEGIN
   UPDATE product
   SET unit_price = unit_price + 2
   IF ( SELECT MAX(unit_price) FROM product ) > $50
      BREAK
END
```

如果最贵的产品的价格高于 $50， BREAK 语句将中断 WHILE 循环。否则，循环将一直继续到平均价格高于或等于 $30。

WRITETEXT 语句 [T-SQL]

说明

允许对现有文本或图像列进行交互更新但不进行记录日志。

语法

WRITETEXT *table-name.column-name*
text_pointer [**WITH LOG**] *data*

用法

更新现有文本或图像值。除非提供 WITH LOG 选项，否则事务日志中不记录更新。对视图不能执行 WRITETEXT 操作。

权限

无。

副作用

WRITETEXT 不触发触发器，缺省情况下，事务日志中不记录 WRITETEXT 操作。

另请参阅

"READTEXT 语句 [T-SQL]" 第 564 页

"TEXTPTR 函数 [文本和图像]" 第 221 页

标准和兼容性

- **SQL/92** Transact-SQL 扩展。
- **SQL/99** Transact-SQL 扩展。
- **Sybase** 受 Adaptive Server Enterprise 支持。

示例

下面这段代码阐释了 WRITETEXT 语句的用法。该示例中的 SELECT 语句返回单行。该示例用值 **newdata** 替换指定行中的 *column_name* 列的内容。

```
EXEC SQL create variable textpointer binary(16);
EXEC SQL set textpointer =
   (  SELECT textptr(column_name)
      FROM table_name WHERE id = 5 );
EXEC SQL writetext table_name.column_name
   textpointer 'newdata';
```

第二部分　系统对象

这部分介绍系统表、视图和过程。

第 5 章

系统表

关于本章

每个数据库的结构都是以许多系统表来描述的。

系统表由 **SYS** 用户 ID 所拥有。只能通过数据库系统更改这些系统表的内容。不能使用 UPDATE、 DELETE 和 INSERT 命令来修改这些表的内容。此外，也不能使用 ALTER TABLE 和 DROP 命令更改这些表的结构。

本章包含了有关每个系统表的说明。某些列只有两个可能的值。通常这些值为 "Y" 和 "N"，分别表示 " 是 " 和 " 否 "。这些列是由 "(Y/N)" 来指定的。

DUMMY 系统表

列名称	列类型	列约束	表约束
dummy_col	INTEGER	NOT NULL	

DUMMY 表是以只读表的形式提供的，并且始终只有一行。它有利于抽取数据库信息，如下例中，从数据库中获取当前用户 ID 和当前日期。

```
SELECT USER, today(*) FROM SYS.DUMMY
```

在 FROM 子句中可选用 FROM SYS.DUMMY。如果 FROM 子句中没有指定任何表，将假定该表为 SYS.DUMMY。上例可写成下面的形式：

```
SELECT USER, today(*)
```

dummy_col 不使用该列。它之所以存在，是因为不能创建没有列的表。

SYSARTICLE 系统表

列名称	列类型	列约束	表约束
publication_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSPUBLICATION
table_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSTABLE
where_expr	LONG VARCHAR		
subscribe_by_expr	LONG VARCHAR		
query	CHAR(1)	NOT NULL	

SYSARTICLE 的每一行都描述 SQL Remote 发布中的一个项目。

publication_id　该项目所属的发布。

table_id　每个项目都由单个表中的列和行构成。此列包含该表的表 ID。

where_expr　对于包含 WHERE 子句定义的行的子集的项目，该列包含搜索条件。

subscribe_by_expr　对于包含 SUBSCRIBE BY 表达式定义的行的子集的项目，该列包含表达式。

SYSARTICLECOL 系统表

列名称	列类型	列约束	表约束
publication_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSARTICLE
table_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSARTICLE，SYSCOLUMN
column_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSCOLUMN

每一行都标识项目中的一列。

publication_id 列所属的发布的唯一标识符。

table_id 列所属的表。

column_id 列标识符，来自 SYSCOLUMN 系统表。

SYSATTRIBUTE 系统表

列名称	列类型	列约束
object_type	CHAR(1)	NOT NULL
object_id	UNSIGNED INT	NOT NULL
attribute_id	UNSIGNED INT	NOT NULL
sub_object_id1	UNSIGNED INT	
sub_object_id2	UNSIGNED INT	
attribute_value	LONG VARCHAR	

创建 SYSATTRIBUTE 和 SYSATTRIBUTENAME 是为了将有关数据库对象的新信息添加到系统表，而不用更改模式。对于 8.0.1 版，它们只包含应用于表的新属性 PCTFREE 的信息。 SYSATTRIBUTE 的每一行都描述一个系统对象，例如特定的表。指定属性时会添加行；例如，每个具有 PCTFREE 设置的表都添加到 SYSATTRIBUTE。

object_type　属性描述的对象类型。例如， T 指的是表。

object_id　特定对象的 ID。例如，当 *object_type* 为 T 时， *object_id* 为 SYSTABLE 中的 *table_id*。

attribute_id　所描述的属性。例如， 1 为 PCTFREE。每个属性 ID 的描述性名称都存储在 SYSATTRIBUTENAME 中。

sub_object_id1　属性的附加信息，没有附加信息时为 NULL。对于 PCTFREE 属性，该列为 NULL。

sub_object_id2　属性的附加信息，没有附加信息时为 NULL。对于 PCTFREE 属性，该列为 NULL。

attribute_value　属性的值。例如，对于 PCTFREE 属性，该字段的值为每个表页中剩余可用空间的百分比。

SYSATTRIBUTENAME 系统表

列名称	列类型	列约束
attribute_id	UNSIGNED INT	NOT NULL
attribute_name	CHAR(128)	NOT NULL

该表提供了 SYSATTRIBUTE 中所用属性 ID 的属性名称。

attribute_id 属性的 ID。

attribute_name 属性的名称。

SYSCAPABILITY 系统表

列名称	列类型	列约束	表约束
capid	INTEGER	NOT NULL	主键。外键引用 SYSCAPABILITYNAME
srvid	INTEGER	NOT NULL	主键。外键引用 SYSSERVERS
capvalue	CHAR(128)	NOT NULL	

每一行都标识远程服务器的一种功能。

capid　功能，如 SYSCAPABILITYNAME 中所列。

srvid　在其中应用了该功能的服务器，如 SYSSERVERS 中所列。

capvalue　功能的值。

SYSCAPABILITYNAME 系统表

列名称	列类型	列约束	表约束
capid	INTEGER	NOT NULL	主键
capname	CHAR(128)	NOT NULL	

每一行都标识一种功能。

capid 功能 ID。

capname 功能的名称。

SYSCOLLATION 系统表

列名称	列类型	列约束	表约束
collation_id	SMALLINT	NOT NULL	主键
collation_label	CHAR(10)	NOT NULL	
collation_name	CHAR(128)	NOT NULL	
collation_order	BINARY(1280)	NOT NULL	

该表包含 Adaptive Server Anywhere 可用的归类序列。无法修改该表的内容。

collation_id　标识归类序列的唯一编号。编号为 2 的 **collation_id** 的归类序列是 Adaptive Server Anywhere 早期版本中使用的序列，而且是创建数据库时的缺省序列。

collation_label　标识归类序列的字符串。创建数据库时，会使用 -z 选项指定归类标签，以选择要使用的归类序列。

collation_name　归类序列的名称。

collation_order　一系列字节，它们定义了在进行比较时如何处理 256 个字符代码中的每一个字符代码。在对字符进行比较之前，所有的字符串比较都会根据归类顺序表翻译每个字符。对于其它的 ASCII 代码页，唯一的不同之处在于重音字符的排序方式。一般说来，重音字符被当作非重音字符来排序。

SYSCOLLATIONMAPPINGS 系统表

列名称	列类型	列约束	表约束
collation_label	CHAR(10)	NOT NULL	主键
collation_name	CHAR(128)	NOT NULL	
cs_label	CHAR(128)		
so_case_label	CHAR(128)		
so_caseless_label	CHAR(128)		
jdk_label	CHAR(128)		

collation_label 定义归类序列的字符串。创建数据库时，会使用 -z 选项指定归类标签，以选择要使用的归类序列。

collation_name 用于描述字符集编码的归类名称。

cs_label GPG 字符集映射标签。

so_case_label 区分大小写的 GPG 字符集映射的归类排序顺序。

so_caseless_label 不区分大小写的 GPG 字符集映射的归类排序顺序。

jdk_label JDK 字符集标签。

对于新创建的数据库，该表只有一行，该行包含数据库归类映射。对于使用 Adaptive Server Anywhere 7.x 版或更早版本创建的数据库，该表包含所有内部归类的归类映射。

SYSCOLPERM 系统表

列名称	列类型	列约束	表约束
table_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSCOLUMN
grantee	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM.user_id
grantor	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM.user_id
column_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSCOLUMN
privilege_type	SMALLINT	NOT NULL	主键
is_grantable	CHAR(1)	NOT NULL	

GRANT 语句可以将 UPDATE 权限赋予表中的单个列。 **SYSCOLPERM** 的一行中记录了每个具有 UPDATE 权限的列。

table_id　包含该列的表的表号。

grantee　用户 ID 的用户号，该用户 ID 被赋予该列的 UPDATE 权限。如果 **grantee** 是特殊 **PUBLIC** 用户 ID 的用户号，则 UPDATE 权限赋予所有用户 ID。

grantor　授予权限的用户 ID 的用户号。

column_id　该列编号，与 **table_id** 一同标识被授予 UPDATE 权限的列。

privilege_type　该列中的编号表示列权限的类型 （REFERENCES、SELECT 或 UPDATE）。

is_grantable (Y/N)　表示是否将 " 授予权限选项 " 授予了该列权限。

SYSCOLSTAT 系统表

列名称	列类型	列约束	表约束
table_id	UNSIGNED INT	NOT NULL	主键
column_id	UNSIGNED INT	NOT NULL	主键
format_id	SMALL INT	NOT NULL	
update_time	TIMESTAMP	NOT NULL	
density	FLOAT	NOT NULL	
max_steps	SMALL INT	NOT NULL	
actual_steps	SMALL INT	NOT NULL	
step_values	LONG BINARY		
frequencies	LONG BINARY		

该表存储列统计数据，这些数据以直方图的形式存储并由优化程序使用。使用 **sa_get_histogram** 存储过程可以最有效地检索该表的内容。

table_id 唯一标识该列所属的表或视图的编号。

column_id 唯一标识列的编号。

format_id 用于确定其余行格式的内部字段。

update_time 此行的上次更新时间。

density 估计列的单值加权平均选择性，不计算存储在行中的大单值选择性的选择性。

max_steps　直方图中允许的最大梯级数。

actual_steps　此时实际使用的梯级数。

step_values　直方图梯级的界限值。

frequencies　直方图梯级的选择性。

SYSCOLUMN 系统表

列名称	列类型	列约束	表约束
table_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSTABLE.table_id
column_id	UNSIGNED INT	NOT NULL	主键
pkey	CHAR(1)	NOT NULL	
domain_id	SMALLINT	NOT NULL	外键引用 SYSDOMAIN.domain_id
nulls	CHAR(1)	NOT NULL	
width	SMALLINT	NOT NULL	
scale	SMALLINT	NOT NULL	
unused	INTEGER	NOT NULL	
max_identity	BIGINT	NOT NULL	
column_name	CHAR(128)	NOT NULL	
remarks	LONG VARCHAR		
"default"	LONG VARCHAR		
"check"	LONG VARCHAR		
user_type	SMALLINT		外键引用 SYSUSERTYPE.type_id
format_str	CHAR(128)		
column_type	CHAR(1)	NOT NULL	
remote_name	VARCHAR(128)		

列名称	列类型	列约束	表约束
remote_type	UNSIGNED INT		

每个表或视图中的每一列在 **SYSCOLUMN** 中都有一行描述。

table_id　唯一标识该列所属的表或视图的编号。

column_id　每个表从 1 开始为列编号。列号的顺序决定了列在命令中的显示顺序。如：

```
SELECT * FROM TABLE
```

pkey (Y/N)　指出该列是否是表主键的一部分。

domain_id　该列的数据类型，由 **SYSDOMAIN** 表中所列的数据类型编号指出。

nulls (Y/N)　指出该列中是否允许 NULL 值。

width　字符串列的长度、数值列的精度或任何其它数据类型的存储字节数。

scale　数值数据类型列的小数点后的位数，对于所有其它数据类型该值为零。

unused　未使用。

max_identity　如果该列为 AUTOINCREMENT、 IDENTITY 或 GLOBAL AUTOINCREMENT 列，则为列的最大值。

column_name　列的名称。

remarks　注释字符串。

default　列的缺省值。仅在 INSERT 语句没有指定列值时才使用该值。

check　列上定义的任何 CHECK 条件。

user_type　如果在用户定义的数据类型上定义了列，则该数据类型保存在这里。

format_str 目前未使用。

column_type 列的类型。包含 C（表示计算列）和 R（表示其它列）。

remote_name 远程列的名称。

remote_type 远程列的类型。该值由远程服务器或接口定义。

SYSDOMAIN 系统表

列名称	列类型	列约束	表约束
domain_id	SMALLINT	NOT NULL	主键
domain_name	CHAR(128)	NOT NULL	
type_id	SMALLINT	NOT NULL	
precision	SMALLINT		

每种预定义的数据类型（有时称为 **domains**）都有一个指定的唯一编号。提供 **SYSDOMAIN** 表是为了参考，以显示这些编号与相应数据类型之间的关联。此表永远不会更改。

domain_id　指派给每种数据类型的唯一编号。无法更改这些编号。

domain_name　字符串，该字符串包含通常在 CREATE TABLE 命令中使用的数据类型（如 **char** 或 **integer**）。

type_id　ODBC 数据类型。这与 Transact-SQL 兼容性 dbo.SYSTYPES 表中的 "data_type" 相对应。

precision　使用这种数据类型可以存储的有效位数。对于非数值数据类型，列值为 NULL。

SYSEVENT 系统表

列名称	列类型	列约束	表约束
event_id	INTEGER	NOT NULL	主键
creator	UNSIGNED INT	NOT NULL	
event_name	VARCHAR(128)	NOT NULL	
enabled	CHAR(1)	NOT NULL	
location	CHAR(1)	NOT NULL	
event_type_id	INTEGER		
action	LONG VARCHAR		
external_action	LONG VARCHAR		
condition	LONG VARCHAR		
remarks	LONG VARCHAR		
source	LONG VARCHAR		

SYSEVENT 中的每一行都描述一个使用 CREATE EVENT 创建的事件。

event_id 指派给每个事件的唯一编号。

creator 事件所有者的用户号。通过查看 **SYSUSERPERM** 可以找到用户的名称。

event_name 事件的名称。

enabled (Y/N) 指出是否允许事件触发。

location 将要触发事件的位置：

- C = 统一
- R = 远程
- A = 全部

event_type_id　对于系统事件，**SYSEVENTTYPE** 中所列的事件类型。

action　事件处理程序定义。

external_action　未使用。

condition　用于控制事件处理程序触发的 WHERE 条件。

remarks　注释字符串。

source　如果 preserve_source_format 选项为 ON，该列包含事件处理程序的原始来源。它用于保持原文本的外观。有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > PRESERVE_SOURCE_FORMAT 选项 [database]"。

SYSEVENTTYPE 系统表

列名称	列类型	列约束	表约束
event_type_id	INTEGER	NOT NULL	主键
name	VARCHAR(128)	NOT NULL	
description	LONG VARCHAR		

该表列出了 CREATE EVENT 可以引用的系统事件类型。

event_type_id 指派给每种事件类型的唯一编号。

name 系统事件类型的名称。

description 系统事件类型的说明。

SYSEXTENT 系统表

列名称	列类型	列约束	表约束
file_id	SMALLINT	NOT NULL	主键，外键引用 SYSFILE
extent_id	SMALLINT	NOT NULL	主键
first_page	INTEGER	NOT NULL	
last_page	INTEGER	NOT NULL	
file_name	LONG VARCHAR	NOT NULL	

该表未被使用。

SYSEXTERNLOGINS 系统表

列名称	列类型	列约束	表约束
user_id	UNSIGNED INT	NOT NULL	主键。SYSUSERPERM 的外键
srvid	INTEGER	NOT NULL	主键。SYSSERVERS 的外键
remote_login	VARCHAR(128)		
remote_password	VARBINARY(128)		

每行都描述一个远程数据访问的外部登录。

user_id 本地数据库上的用户 ID。

srvid SYSSERVERS 中所列的远程服务器。

remote_login 该用户的登录名 （对于远程服务器）。

remote_password 该用户的口令 （对于远程服务器）。

SYSFILE 系统表

列名称	列类型	列约束	表约束
file_id	SMALLINT	NOT NULL	主键
file_name	LONG VARCHAR	NOT NULL	唯一索引
dbspace_name	CHAR(128)	NOT NULL	
store_type	CHAR(8)	NOT NULL	

每个数据库都由一个或多个操作系统文件组成。每个文件都记录在 **SYSFILE** 中。

file_id　数据库中的每个文件都有一个指定的唯一编号。该文件标识符为 **SYSFILE** 的主键。所有的系统表都存储在 **file_id** 为 0 的文件中。

file_name　创建数据库时会存储数据库名称。该名称仅供参考。

dbspace_name　每个文件都有一个唯一的 dbspace 名称。该名称用在 CREATE TABLE 命令中。

store_type　该字段供内部使用。

SYSFKCOL 系统表

列名称	列类型	列约束	表约束
foreign_table_id	UNSIGNED INT	NOT NULL	主键。外键引用 SYSCOLUMN.table_id。外键引用 SYSFOREIGNKEY
foreign_key_id	SMALLINT	NOT NULL	主键，外键引用 SYSFOREIGNKEY。foregin_key_id
foreign_column_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSCOLUMN column_id
primary_column_id	UNSIGNED INT	NOT NULL	

SYSFKCOL 的每一行都描述了关系中外表的**外列**与主表的**主列**之间的关联。

foreign_table_id 外表的表号。

foreign_key_id 外表中外键的键号。**foreign_table_id** 和 **foreign_key_id** 共同唯一标识 **SYSFOREIGNKEY** 中的一行。从此行可以获得主表的表号 （使用 **SYSFOREIGNKEY** 表）。

foreign_column_id 此列号和 **foreign_table_id** 标识 **SYSCOLUMN** 中的外列说明。

primary_column_id 此列号和从 **SYSFOREIGNKEY** 获得的 **primary_table_id** 标识 **SYSCOLUMN** 中的主列说明。

SYSFOREIGNKEY 系统表

外键是两个表（外表和主表）之间的关系。每个外键都由

列名称	列类型	列约束	表约束
foreign_table_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSTABLE.table_id。唯一索引
foreign_key_id	SMALLINT	NOT NULL	主键
primary_table_id	UNSIGNED INT	NOT NULL	外键引用 SYSTABLE.table_id
root	INTEGER	NOT NULL	
check_on_commit	CHAR(1)	NOT NULL	
nulls	CHAR(1)	NOT NULL	
role	CHAR(128)	NOT NULL	唯一索引
remarks	LONG VARCHAR		
primary_index_id	UNISGNED INT	NOT NULL	
fk_not_enforced	CHAR(1)	NOT NULL	
hash_limit	SMALLINT	NOT NULL	

SYSFOREIGNKEY 中的一行和 **SYSFKCOL** 中的一行或多行来定义。**SYSFOREIGNKEY** 包含外键的一般信息，而 **SYSFKCOL** 则标识外键中的列并将外键中的每列与主表的主键中的列关联。

foreign_table_id　外表的表号。

foreign_key_id　每个外键都有一个外键号，该号在以下方面是唯一的：

- 外表的所有其它外键的键号
- 主表的所有外键的键号
- 外表的所有索引的索引号

primary_table_id 主表的表号。

root 外键以 B 树形式存储在数据库中。 **root** 标识数据库文件中 B 树的根的位置。

check_on_commit (Y/N) 指出 INSERT 和 UPDATE 命令是否应等到下一个 COMMIT 命令再检查外键是否有效。对于外键中的每一行来说，如果外键列中的值包含 NULL 值或者与主表的某一行中的主键值匹配，则外键有效。

nulls (Y/N) 指出是否允许外键中的列包含 NULL 值。注意该设置与外键中所含列的 **nulls** 设置无关。

role 外表和主表之间的关系名称。除非另外指定， **role** 名称将与主表的名称相同。外表不能有两个角色名相同的外键。

remarks 注释字符串。

primary_index_id 主键的 index_id，如果主键是复合索引的一部分，则为 **root**。

fk_not_enforced (Y/N) 如果表中有一个是远程表，则为 N。

hash_limit 包含有关物理索引表示形式的信息。

SYSGROUP 系统表

列名称	列类型	列约束	表约束
group_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM.user_id
group_member	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM.user_id

每组的每一个成员在 **SYSGROUP** 中都有一行。该表描述了组与成员之间的多对多关系。一个组可以有许多成员，而一个用户也可以是许多组的成员。

group_id　组的用户号。

group_member　成员的用户号。

SYSINDEX 系统表

列名称	列类型	列约束	表约束
table_id	UNSIGNED INT	NOT NULL	主键，唯一索引。外键引用 SYSTABLE
index_id	UNSIGNED INT	NOT NULL	主键
root	INTEGER	NOT NULL	
file_id	SMALLINT	NOT NULL	
"unique"	CHAR(1)	NOT NULL	
creator	UNSIGNED INT	NOT NULL	外键引用 SYSUSERPERM.user_id
index_name	CHAR(128)	NOT NULL	唯一索引
hash_limit	SMALLINT	NOT NULL	
index_owner	CHAR(4)	NOT NULL	
index_type	CHAR(4)	NOT NULL	
remarks	LONG, VARCHAR		

数据库中的每条索引在 **SYSINDEX** 中都有一行描述。索引中的每一列在 **SYSIXCOL** 中都有一行描述。

table_id 唯一标识了包含该索引的表。

index_id 一个特定表的每条索引都有一个指定的唯一索引号。

root 索引以 B 树的形式存储在数据库中。 **root** 标识数据库文件中 B 树的根的位置。

file_id 索引全部包含在具有该 **file_id** 的文件中 （请参阅 **SYSFILE**）。

unique 指出索引是唯一索引 ("Y")、非唯一索引 ("N") 还是唯一约束 ("U")。唯一索引可以避免索引表中的两行在索引列中具有相同的值。

creator　索引创建者的用户号。该用户始终与 table_id 所标识的表的创建者相同。

index_name　索引的名称。用户 ID 在它所拥有的表中不能有两条同名的索引。

index_owner　所有者。该字段始终为 SA。

index_type　类型。该字段始终为 SA。

remarks　注释字符串。

SYSINFO 系统表

列名称	列类型	列约束	表约束
page_size	INTEGER	NOT NULL	
encryption	CHAR(1)	NOT NULL	
blank_padding	CHAR(1)	NOT NULL	
case_sensitivity	CHAR(1)	NOT NULL	
default_collation	CHAR(10)		
database_version	SMALLINT	NOT NULL	
classes_version	CHAR(10)		

该表指出了创建数据库时定义的数据库特性。它始终只有一行。

page_size 指定的页面大小 （以字节为单位）。缺省值是 2048。

encryption (Y/N) 指出是否 DBINIT 使用了 -e 开关。

blank_padding (Y/N) 指出创建数据库时是否在数据库中使用了空白填充以进行字符串比较 （`dbinit` 使用了 -b 开关）。

case_sensitivity (Y/N) 指出创建数据库时是否区分大小写。大小写敏感性会影响值的比较，但不会影响表名和列名的比较。例如，如果数据库不区分大小写，可以以任意大小写的形式指定表名 （如 **SYSCATALOG**），但是在区分大小写的数据库中， 'abc' = 'ABC' 是错误的。

default_collation 与 **SYSCOLLATE** 中的 **collation_label** 相对应的字符串，它还与使用 DBINIT 指定的归类序列相对应。缺省值对应于多语言归类序列 （代码页 850），这是 Watcom SQL 3.2 之前版本的缺省值。归类序列用于所有的字符串比较，包括字符串搜索以及列名和表名的比较。

database_version　指出数据库格式的小整数值。当有更新的版本可供使用时，新的功能可能会要求更改数据库文件的格式。 Adaptive Server Anywhere 软件的版本号，它确定该数据库是否是用该软件的更新版本创建的，因此无法被正在使用的软件识别。

classes_version　一个小字符串，用于描述您的计算机上当前安装的 **SYS.JAVA.CLASSES** 库的当前版本。

SYSIXCOL 系统表

列名称	列类型	列约束	表约束
table_id	UNSIGNED INT	NOT NULL	主键。外键引用 SYSCOLUMN. 外键引用 SYSINDEX。
index_id	UNSIGNED INT	NOT NULL	主键。外键引用 SYSINDEX
sequence	SMALLINT	NOT NULL	主键
column_id	UNSIGNED INT	NOT NULL	外键引用 SYSCOLUMN
"order"	CHAR(1)	NOT NULL	

每个索引的每一列在 **SYSIXCOL** 中都有一行。

table_id 标识在其中应用了该索引的表。

index_id 标识使用该列的索引。 **table_id** 与 **index_id** 共同标识 **SYSINDEX** 中描述的一个索引。

sequence 索引中的每一列都有一个指定的从 0 开始的唯一编号。这些编号的顺序决定了列在索引中的相对重要性。最重要的列的 **sequence** 号为 0。

column_id 列号标识带索引的列。 **table_id** 与 **column_id** 共同标识 **SYSCOLUMN** 中的一列。

order (A/D) 指出该列在索引中是升序还是降序。

SYSJAR 系统表

列名称	列类型	列约束	表约束
jar_id	INTEGER	NOT NULL	主键
creator	UNSIGNED INT	NOT NULL	
jar_name	LONG VARCHAR	NOT NULL	唯一索引
jar_file	LONG VARCHAR		
create_time	TIMESTAMP	NOT NULL	
update_time	TIMESTAMP	NOT NULL	
remarks	LONG VARCHAR		

jar_id　包含 jar 文件 ID 的字段。该字段还引用 **SYSJAR** 系统表。

creator　jar 文件创建者的名称。

jar_name　jar 文件的名称。

jar_file　jar 文件的文件名。

create_time　创建 jar 文件的时间。

update_time　上次更新 jar 文件的时间。

remarks　注释字段。

SYSJARCOMPONENT 系统表

列名称	列类型	列约束	表约束
component_id	INTEGER	NOT NULL	主键
jar_id	INTEGER		外键引用 SYSJAR
component_name	LONG VARCHAR		
component_type	CHAR(1)		
create_time	TIMESTAMP	NOT NULL	
contents	LONG BINARY		
remarks	LONG VARCHAR		

component_id 包含组件 ID 的主键。

jar_id 包含 jar 的 ID 号的字段。该字段还引用 **SYSJAR** 系统表。

component_name 组件的名称。

component_type 组件的类型。

create_time 包含组件创建时间的字段。

contents jar 文件的字节代码。

remarks 注释字段。

SYSJAVACLASS 系统表

列名称	列类型	列约束	表约束
class_id	INTEGER	NOT NULL	主键
replaced_by	INTEGER		外键引用 SYSJAVACLASSES. class_id
creator	UNSIGNED INT	NOT NULL	外键引用 SYSUSERPERM.user_id
jar_id	INTEGER		
type_id	SMALLINT		外键引用 SYSUSERTYPE
class_name	LONG VARCHAR	NOT NULL	
public	CHAR(1)	NOT NULL	
component_id	INTEGER		外键引用 SYSJARCOMPONENT
create_time	TIMESTAMP	NOT NULL	
update_time	TIMESTAMP	NOT NULL	
class_descriptor	LONG BINARY		
remarks	LONG VARCHAR		

SYSJAVACLASS 系统表包含与 Java 类相关的所有信息。

class_id　该字段包含 java 类的 ID。也是表的主键。

replaced_by　引用主键字段 class_id 的字段。

creator　该字段包含类创建者的 user_id。该字段引用 **SYSUSERPERM** 系统表中的 user_id 字段以获取用户名。

jar_id 该字段包含了产生类的 jar 文件的 ID。

type_id 该字段包含用户类型的 ID。该字段引用 SYSUSERTYPE 系统表以获取用户类型的 ID。

class_name 该字段包含 Java 类的名称。

public 该字段决定类是否为公共类或专用类。

component_id 该字段引用 SYSJARCOMPONENT 系统表，并包含组件的 ID。

create_time 包含组件的创建时间。

update_time 包含组件的上次更新时间。

class_descriptor jar 文件的字节代码。

remarks 包含注释字符串。

SYSLOGIN 系统表

列名称	列类型	列约束	表约束
integrated_login_id	CHAR(128)	NOT NULL	主键
login_uid	UNSIGNED INT	NOT NULL	外键引用 SYSUSERPERM. user_id
Remarks	LONG VARCHAR		

该表包含可用于使用集成注册连接到数据库的所有 " 用户配置文件 " 名。作为安全措施，只有具有 DBA 权限的用户才能查看该表的内容。

integrated_login_id　一个字符串值，其中包含 " 用户配置文件 " 名称，该名称用于映射到数据库中的用户 ID。如果用户使用该 " 用户配置文件 " 名成功登录，而且数据库可以接受集成注册，用户不用提供用户 ID 或口令就可以连接到数据库。

login_uid　系统表 **SYSUSERPERM** 的外键。

remarks　注释字符串

SYSOPTBLOCK 系统表

该表保留供系统使用。

SYSOPTION 系统表

列名称	列类型	列约束	表约束
user_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM
"option"	CHAR(128)	NOT NULL	主键
"setting"	LONG VARCHAR	NOT NULL	

选项设置由 SET 命令存储在 **SYSOPTION** 表中。对于每个选项，每个用户都可以有自己的设置。此外， **PUBLIC** 用户 ID 的设置定义了缺省设置，以便用于没有自己的设置的用户 ID。

user_id　对其应用该选项设置的用户号。

option　选项的名称。

setting　指定选项的当前设置。

SYSOPTJOINSTRATEGY 系统表

该表保留供系统使用。

SYSOPTORDER 系统表

该表保留供系统使用。

SYSOPTQUANTIFIER 系统表

该表保留供系统使用。

SYSOPTREQUEST 系统表

该表保留供系统使用。

SYSOPTREWRITE 系统表

该表保留供系统使用。

SYSOPTSTAT 系统表

该表存储有关成本模型的信息。它保留供系统使用。

SYSPROCEDURE 系统表

列名称	列类型	列约束	表约束
proc_id	UNSIGNED INT	NOT NULL	主键
creator	UNSIGNED INT	NOT NULL	唯一索引。外键引用 SYSUSERPERM.user_id
proc_name	CHAR(128)	NOT NULL	唯一索引
proc_defn	LONG VARCHAR		
remarks	LONG VARCHAR		
replicate	CHAR(1)	NOT NULL	
srvid	INTEGER		外键引用 SYSSERVERS
source	LONG VARCHAR		

数据库中的每个过程在 **SYSPROCEDURE** 中都有一行描述。

proc_id 每个过程都有一个指定的唯一编号 (**procedure number**)，这是 **SYSPROCEDURE** 的主键。

creator 该用户号标识过程的所有者。通过查看 **SYSUSERPERM** 可以找到用户的名称。

proc_name 过程的名称。一个创建者不能有两个同名过程。

proc_defn 用于创建过程的命令。

remarks 注释字符串。

replicate (Y/N) 指出过程是否为 Replication Server 系统中的主数据源。

srvid 如果是远程数据库服务器上的过程，则指明远程服务器。

source　如果 preserve_source_format 选项为 ON，则该列包含过程的原始来源。它用于保持原文本的外观。有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > PRESERVE_SOURCE_FORMAT 选项 [database]"。

SYSPROCPARM 系统表

列名称	列类型	列约束	表约束
proc_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSPROCEDURE
parm_id	SMALLINT	NOT NULL	主键
parm_type	SMALLINT	NOT NULL	
parm_mode_in	CHAR(1)	NOT NULL	
parm_mode_out	CHAR(1)	NOT NULL	
domain_id	SMALLINT	NOT NULL	外键引用 SYSDOMAIN
width	SMALLINT	NOT NULL	
scale	SMALLINT	NOT NULL	
parm_name	CHAR(128)	NOT NULL	
remarks	LONG VARCHAR		
"default"	LONG VARCHAR		
user_type	INTEGER		

数据库中的过程的每个参数在 **SYSPROCPARM** 中都有一行描述。

proc_id 唯一标识该参数所属的过程。

parm_id 每个过程都从 1 开始对参数进行编号。参数编号的顺序对应于定义参数时采取的顺序。

parm_type 参数类型将为下列类型之一：

- 普通参数 （变量）
- 结果变量 - 与返回结果集的过程一起使用

- SQLSTATE 错误值
- SQLCODE 错误值

parm_mode_in (Y/N)　指出该参数是否为过程提供值（**IN** 或 **INOUT** 参数）。

parm_mode_out (Y/N)　指出该参数是否从过程返回值（**OUT** 或 **INOUT** 参数）。

domain_id　用 **SYSDOMAIN** 表中列出的数据类型号标识参数的数据类型。

width　包含字符串参数的长度、数值参数的精度或任何其它数据类型的存储字节数。

scale　数值数据类型参数的小数点后的位数，对于所有其它数据类型该值为零。

parm_name　过程参数的名称。

remarks　注释字符串。

default　未使用。

user_type　参数的用户类型。

SYSPROCPERM 系统表

列名称	列类型	列约束	表约束
proc_id	UNSIGNED INT	NOT NULL	主键。外键引用 SYSPROCEDURE
grantee	UNSIGNED INT	NOT NULL	主键。外键引用 SYSUSERPERM.user_id

只有得到授权的用户才能调用过程。 **SYSPROCPERM** 表的每一行对应于一个有权调用一个过程的用户。

proc_id 过程号唯一标识被授予权限的过程。

grantee 接收权限的用户 ID 的用户号。

SYSPUBLICATION 系统表

列名称	列类型	列约束	表约束
publication_id	UNSIGNED INT	NOT NULL	主键
creator	UNSIGNED INT	NOT NULL	唯一索引。外键引用 SYSUSERPERM.user_id
publication_name	CHAR(128)	NOT NULL	唯一索引
remarks	LONG VARCHAR		
type	CHAR(1)	NOT NULL	

每一行描述一个 SQL Remote 发布。

publication_id　发布的唯一标识号。

creator　发布的所有者。

publication_name　发布的名称 （必须是有效标识符）。

remarks　说明性注释。

type　此列不推荐使用。

SYSREMOTEOPTION 系统表

作用

每一行都描述一个 SQL Remote 消息链接参数的值。

列

Column	数据类型	列约束	表约束
option_id	UNSIGNED INT	NOT NULL	主键
user_id	UNSIGNED INT	NOT NULL	主键
"setting"	VARCHAR(255)	NOT NULL	

option_id 消息链接参数的标识号。

user_id 为其设置参数的用户 ID。

setting 消息链接参数的值。

SYSREMOTEOPTIONTYPE 系统表

作用

每一行都描述一个 SQL Remote 消息链接参数。

列

Column	数据类型	列约束	表约束
option_id	UNSIGNED INT	NOT NULL	主键
type_id	UNSIGNED INT	NOT NULL	
"option"	VARCHAR(128)	NOT NULL	

option_id　消息链接参数的标识号。

type_id　使用该参数的消息类型的标识号。

option　消息链接参数的名称。

SYSREMOTETYPE 系统表

列名称	列类型	列约束	表约束
type_id	SMALLINT	NOT NULL	主键
type_name	CHAR(128)	NOT NULL	唯一索引
publisher_address	LONG VARCHAR	NOT NULL	
remarks	LONG VARCHAR		

SYSREMOTETYPE 系统表包含有关 SQL Remote 的信息。

type_id 标识使用 SQL Remote 所支持的哪种消息系统向该用户发送消息。

type_name SQL Remote 所支持的消息系统的名称。

publisher_address 远程数据库发布者的地址。

remarks 说明性注释。

SYSREMOTEUSER 系统表

列名称	列类型	列约束	表约束
user_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM
consolidate	CHAR(1)	NOT NULL	
type_id	SMALLINT	NOT NULL	外键引用 SYSREMOTETYPE
address	LONG VARCHAR	NOT NULL	
frequency	CHAR(1)	NOT NULL	唯一索引
send_time	TIME		唯一索引
log_send	NUMERIC(20,0)	NOT NULL	
time_sent	TIMESTAMP		
log_sent	NUMERIC(20,0)	NOT NULL	
confirm_sent	NUMERIC(20,0)	NOT NULL	
send_count	INTEGER	NOT NULL	
resend_count	INTEGER	NOT NULL	
time_received	TIMESTAMP		
log_received	NUMERIC(20,0)	NOT NULL	
confirm_received	NUMERIC(20,0)		
receive_count	INTEGER	NOT NULL	
rereceive_count	INTEGER	NOT NULL	

每一行都描述一个具有 REMOTE 权限的用户 ID （预订者），以及发送给该用户和该用户发送来的 SQL Remote 消息的状态。

user_id　具有 REMOTE 权限的用户的用户号。

consolidate (Y/N) 指出是否已授予用户 CONSOLIDATE 权限 (Y) 或 REMOTE 权限 (N)。

type_id 标识使用 SQL Remote 所支持的哪种消息系统向该用户发送消息。

address 将接收 SQL Remote 消息的地址。该地址必须适合于 **address_type**。

frequency 发送 SQL Remote 消息的频率。

send_time 下一次将消息发送给该用户的时间。

log_send 只将消息发送到其 **log_send** 大于 **log_sent** 的预订者。

time_sent 将最新的消息发送到该预订者的时间。

log_sent 最近的发送操作的日志偏移。

confirm_sent 该预订者的最近确认操作的日志偏移。

send_count 已发送的 SQL Remote 消息数。

resend_count 用于确保在预订者数据库上只应用一次消息的计数器。

time_received 收到来自该预订者的最新消息的时间。

log_received 预订者数据库中当前数据库上最近收到的操作的日志偏移。

confirm_received 预订者数据库中已发送确认消息的最近操作的日志偏移。

receive_count 已收到的消息数。

rereceive_count 用于确保在当前数据库上只应用一次消息的计数器。

SYSSCHEDULE 系统表

列名称	列类型	列约束	表约束
event_id	INTEGER	NOT NULL	主键
sched_name	VARCHAR(128)	NOT NULL	主键
recurring	TINYINT	NOT NULL	
start_time	TIME	NOT NULL	
stop_time	TIME		
start_date	DATE		
days_of_week	TINYINT		
days_of_month	UNSIGNED INT		
interval_units	CHAR(10)		
interval_amt	INTEGER		

SYSSCHEDULE 中的每一行都描述事件的触发时间 （由 CREATE EVENT 的 SCHEDULE 子句指定）。

event_ id　指派给每个事件的唯一编号。

sched_name　与调度关联的名称。

recurring (0/1)　指出调度是否重复。

start_time　调度开始时间。

stop_time　调度停止时间 （如果使用了 BETWEEN）。

start_date　调度事件执行的第一个日期。

days_of_week　用于指出在星期几调度事件的位掩码：

- x01 = 星期日
- x02 星期一

- x04 = 星期二
- x08 = 星期三
- x10 = 星期四
- x20 = 星期五
- x40 = 星期六

days_of_month 用于指出在每月几号调度事件的位掩码：

- x01 = 1 号
- x02 = 2 号
- x40000000 = 31 号
- x80000000 = 每月的最后 1 天

interval_units EVERY 指定的间隔单位：

- HH = 小时
- NN = 分钟
- SS = 秒

interval_amt EVERY 指定的周期。

SYSSERVERS 系统表

列名称	列类型	列约束	表约束
srvid	INTEGER	NOT NULL	主键
srvname	VARCHAR(128)	NOT NULL	
srvclass	LONG VARCHAR	NOT NULL	
srvinfo	LONG VARCHAR		
srvreadonly	CHAR(1)	NOT NULL	

每一行都描述一个远程服务器。

srvid　远程服务器的标识符。

srvname　远程服务器的名称。

srvclass　服务器类（在 CREATE SERVER 语句中指定）。

srvinfo　服务器信息。

srvreadonly　如果服务器为只读，则为 Y；否则为 N。

SYSSQLSERVERTYPE 系统表

列名称	列类型	列约束	表约束
ss_user_type	SMALLINT	NOT NULL	主键
ss_domain_id	SMALLINT	NOT NULL	
ss_type_name	VARCHAR (30)	NOT NULL	
primary_sa_domain_id	SMALLINT	NOT NULL	
primary_sa_user_type	SMALLINT		

该表包含有关与 Adaptive Server Enterprise 兼容性相关的信息。

ss_user_type 描述 Adaptive Server Enterprise 用户类型的 UNSIGNED INT 字段。

ss_domain_id 描述 Adaptive Server Enterprise 域 ID 的 UNSIGNED INT 字段。

ss_type_name 包含 Adaptive Server Enterprise 类型名称。

primary_sa_domain_id 包含 Adaptive Server Anywhere 主域 ID 的 UNSIGNED INT 字段。

primary_sa_user_type 包含 Adaptive Server Anywhere 主用户类型的 UNSIGNED INT 字段。

SYSSUBSCRIPTION 系统表

列名称	列类型	列约束	表约束
publication_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSPUBLICATION
user_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSREMOTEUSER
subscribe_by	CHAR(128)	NOT NULL	主键
created	NUMERIC(20,0)	NOT NULL	
started	NUMERIC(20,0)		

每一行都描述一个从一个用户 ID （它必须具有 REMOTE 权限）到一个发布的预订。

publication_id　用户 ID 预订的发布的标识符。

user_id　预订发布的用户号。

subscribe_by　预订的 SUBSCRIBE BY 表达式 （如果有的话）的值。

created　事务日志中创建预订的位置所在的偏移。

started　事务日志中启动预订的位置所在的偏移。

SYSSYNC 系统表

列名称	列类型	列约束	表约束
sync_id	UNSIGNED INT	NOT NULL	主键
type	CHAR(1)	NOT NULL	
publication_id	UNSIGNED INT		
progress	NUMERIC(20,0)		
site_name	CHAR(128)		
option	LONG VARCHAR		
server_connect	LONG VARCHAR		
server_conn_type	LONG VARCHAR		
last_download_time	TIMESTAMP		

该表包含与 MobiLink 同步相关的信息。

sync_id 唯一标识行的 SMALLINT 字段。

type 描述同步对象类型的 CHAR(1) 字段："D" 表示定义， "T" 表示模板， "S" 表示站点。

publication_id SYSPUBLICATIONS 表中的 *publication_id*。

progress 上次成功上载的日志偏移。

site_name 包含 MobiLink 用户 ID 的 CHAR(128) 字段。

option 包含所有同步选项的 LONG VARCHAR。

server_connect　包含 MobiLink 同步服务器的地址或 URL 的 LONG VARCHAR 字段。

server_conn_type　标识同步时使用的通信协议 （如 TCP/IP）的 LONG VARCHAR 字段。

last_download_time　指出上次从 MobiLink 同步服务器接收下载流的时间的 TIMESTAMP 字段。

SYSTABLE 系统表

列名称	列类型	列约束	表约束
table_id	UNSIGNED INT	NOT NULL	主键
file_id	SMALLINT	NOT NULL	外键引用 SYSFILE
count	UNSIGNED BIGINT	NOT NULL	
first_page	INTEGER	NOT NULL	
last_page	INTEGER	NOT NULL	
primary_root	INTEGER	NOT NULL	
creator	UNSIGNED INT	NOT NULL	唯一索引。外键引用 SYSUSERPERM.user_id
first_ext_page	INTEGER	NOT NULL	
last_ext_page	INTEGER	NOT NULL	
table_page_count	INTEGER	NOT NULL	
ext_page_count	INTEGER	NOT NULL	
table_name	CHAR(128)	NOT NULL	唯一索引
table_type	CHAR(10)	NOT NULL	
view_def	LONG VARCHAR		
remarks	LONG VARCHAR		
replicate	CHAR(1)	NOT NULL	
existing_obj	CHAR(1)		
remote_location	LONG VARCHAR		

列名称	列类型	列约束	表约束
remote_objtype	CHAR(1)		
srvid	INTEGER		外键引用 SYSSERVERS
server_type	CHAR(4)	NOT NULL	
primary_hash_limit	SMALL INT	NOT NULL	
page_map_start	INTEGER	NOT NULL	
source	LONG VARCHAR		

SYSTABLE 的每一行都描述数据库中的一个表或视图。

table_id　每个表或视图都有一个指定的唯一编号 （表号），这是 **SYSTABLE** 的主键。

file_id　指出包含表的数据库文件。 **file_id** 是 **SYSFILE** 的一个外键。

count　表中的行数在每次成功的 CHECKPOINT 过程中都会更新。 Adaptive Server Anywhere 在优化数据库访问时使用这个数字。 **count** 对于视图始终为 0。

first_page　每个数据库都分为若干个大小固定的页。该值标识包含该表信息的第一页，并在内部用于查找该表的起点。 **first_page** 对于视图始终为 0。

last_page　包含该表信息的最后一页。 **last_page** 对于视图始终为 0。对于全局临时表， 0 表示表是使用 ON COMMIT PRESERVE ROWS 创建的， 1 表示表是使用 ON COMMIT DELETE ROWS 创建的。

primary_root　主键是以 B 树的形式存储在数据库中的。 **primary_root** 为表的主键定位 B 树的根。它对于视图和没有主键的表为 0。

creator　表或视图的所有者的用户号。通过查看 **SYSUSERPERM** 可以找到用户的名称。

first_ext_page　用于存储行扩展和 Blob 的第一页。

last_ext_page 用于存储行扩展和 Blob 的最后一页。这些页是作为双向链接表来维护的。

table_page_count 该表使用的主页总数。

ext_page_count 该表使用的扩展 (blob) 页总数。

table_name 表或视图的名称。一个创建者不能有两个同名的表或视图。

table_type 该列为 **BASE**（表示基表）、**VIEW**（表示视图）和 **GBL TEMP**（表示全局临时表）。没有为局部临时表创建任何条目。

view_def 对于视图，该列包含用于创建视图的 CREATE VIEW 命令。对于表，该列包含表的所有 CHECK 约束。

remarks 注释字符串。

replicate (Y/N) 指出表是否为 Replication Server 系统中的主数据源。

existing_obj (Y/N) 指出表先前是否存在。

remote_location 指出远程对象的存储位置。

remote_objtype 指出远程对象的类型：如果是表则为 "T"；如果是视图则为 "V"；如果是 rpc 则为 "R"；如果是 JavaBean 则为 "B"。

srvid 服务器的唯一 ID。

server_type 表的数据位置。它要么为 SA 要么为 OMNI。

primary_hash_limit 该表的主键索引的散列大小。

page_map_start 为该表维护的页映射的起点。页映射用于在顺序扫描过程中促进块 I/O。

source 如果 preserve_source_format 选项为 ON，则该列包含过程的原始来源。它用于保持原文本的外观。有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > PRESERVE_SOURCE_FORMAT 选项 [database]"。

SYSTABLEPERM 系统表

列名称	列类型	列约束	表约束
stable_id	UNSIGNED INT	NOT NULL	主键，外键引用 SYSTABLE table_id
grantee	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM .user_id
grantor	UNSIGNED INT	NOT NULL	主键，外键引用 SYSUSERPERM .user_id
ttable_id	UNSIGNED INT	NOT NULL	外键引用 SYSTABLE table_id
selectauth	CHAR(1)	NOT NULL	
insertauth	CHAR(1)	NOT NULL	
deleteauth	CHAR(1)	NOT NULL	
updateauth	CHAR(1)	NOT NULL	
updatecols	CHAR(1)	NOT NULL	
alterauth	CHAR(1)	NOT NULL	
referenceauth	CHAR(1)	NOT NULL	

GRANT 命令所赋予的权限存储在 **SYSTABLEPERM** 中。该表中的每一行都对应于一个表、一个授予权限的用户 ID (**grantor**) 和一个被授予权限的用户 ID (**grantee**)。

有几种可以授予的权限。每种权限都可以有下列三个值之一。

- **N**　否，授予者没有授予被授予者该权限。
- **Y**　是，授予者已授予被授予者该权限。

- **G** 被授予者已被授予该权限并能将同样的权限授予其他用户（授予权限选项）。

权限
另一个授予者可能已授予被授予者同一个表的权限。如果是这样，该信息将记录在 **SYSTABLEPERM** 的另一行中。

stable_id 在其中应用了权限的表或视图的表号。

grantor 授予权限的用户 ID 的用户号。

grantee 接收权限的用户 ID 的用户号。

ttable_id 在 Adaptive Server Anywhere 的当前版本中，该表号始终与 **stable_id** 相同。

selectauth (Y/N/G) 指出是否已授予 SELECT 权限。

insertauth (Y/N/G) 指出是否已授予 INSERT 权限。

deleteauth (Y/N/G) 指出是否已授予 DELETE 权限。

updateauth (Y/N/G) 指出是否已为表中的所有列授予 UPDATE 权限。（只有 UPDATE 权限可以在单个的列上授予。其它所有权限则适用于表中的所有列。）

updatecols (Y/N) 指出是否仅为表中的某些列授予了 UPDATE 权限。如果 **updatecols** 的值为 Y，则 **SYSCOLPERM** 中将有一行或多行为该表中的列授予 UPDATE 权限。

alterauth (Y/N/G) 指出是否已授予 ALTER 权限。

referenceauth (Y/N/G) 指出是否已授予 REFERENCE 权限。

SYSTRIGGER 系统表

列名称	列类型	列约束	表约束
trigger_id	UNSIGNED INT	NOT NULL	主键
table_id	UNSIGNED INT	NOT NULL	外键引用 SYSTABLE.table_id
event	CHAR(1)	NOT NULL	唯一
trigger_time	CHAR(1)	NOT NULL	唯一
trigger_order	SMALLINT		唯一
foreign_table_id	UNSIGNED INT		唯一。外键引用 SYSFOREIGNKEY
foreign_key_id	SMALLINT		唯一。外键引用 SYSFOREIGNKEY
referential_action	CHAR(1)		
trigger_name	CHAR(128)		唯一
trigger_defn	LONG VARCHAR	NOT NULL	
remarks	LONG VARCHAR		
source	LONG VARCHAR		

数据库中的每个触发器在 **SYSTRIGGER** 中都有一行描述。该表还包含数据库自动为外键定义创建的触发器，它们有参照触发动作 （如 ON DELETE CASCADE）。

trigger_id　每个触发器都有一个指定的唯一编号 (**trigger number**)，这是 **SYSTRIGGER** 的主键。

table_id　表号将唯一标识该触发器所属的表。

event 导致触发器触发的事件。该单字符值对应于创建触发器时指定的触发器事件。

- **A** INSERT, DELETE
- **B** INSERT, UPDATE
- **C** UPDATE
- **D** DELETE
- **E** DELETE, UPDATE
- **I** INSERT
- **U** UPDATE
- **M** INSERT, DELETE, UPDATE

trigger_time 触发器的触发时间。该单字符值对应于创建触发器时指定的触发器时间。

- **A** AFTER
- **B** BEFORE

trigger_order 触发器的触发顺序。这决定了当有相同类型 （INSERT、UPDATE 或 DELETE）的触发器同时 （BEFORE 或者 AFTER）触发时触发器的触发顺序。

foreign_table_id 包含外键定义的表的表号，它有一个参照触发动作（如 ON DELETE CASCADE）。

foreign_key_id **foreign_table_id** 所引用表的外键的外键号。

referential_action 外键所定义的动作。该单字符值对应于创建外键时指定的动作。

- **C** CASCADE
- **D** SET DEFAULT
- **N** SET NULL

- **R**　RESTRICT

trigger_name　触发器的名称。一个表中不能有两个同名的触发器。

trigger_defn　用于创建触发器的命令。

remarks　注释字符串。

source　如果 preserve_source_format 选项为 ON，则该列包含过程的原始来源。它用于保持原文本的外观。有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库选项 > PRESERVE_SOURCE_FORMAT 选项 [database]"。

SYSTYPEMAP 系统表

列名称	列类型	列约束	表约束
ss_user_type	SMALLINT	NOT NULL	
sa_domain_id	SMALLINT	NOT NULL	外键引用 SYSDOMAIN
sa_user_type	SMALLINT		
nullable	CHAR(1)		

SYSTYPEMAP 系统表包含 SYSSQLSERVERTYPE 系统表的兼容性映射值。

ss_user_type 包含 Adaptive Server Enterprise 用户类型。

sa_domain_id 包含 Adaptive Server Anywhere 6.0 domain_id。

sa_user_type 包含 Adaptive Server Anywhere 6.0 用户类型。

nullable 该字段描述类型是否可以为空值或者不能为空值。

SYSUSERMESSAGES 系统表

列名称	列类型	列约束	表约束
error	INTEGER	NOT NULL	
uid	UNSIGNED INT	NOT NULL	
description	VARCHAR(255)	NOT NULL	
langid	SMALLINT	NOT NULL	

每一行都包含有关错误情况的用户定义消息。

error　错误情况的唯一标识号。

uid　定义消息的用户号。

description　对应于错误情况的消息。

langid　保留。

SYSUSERPERM 系统表

列名称	列类型	列约束	表约束
user_id	UNSIGNED INT	NOT NULL	主键
user_name	CHAR(128)	NOT NULL	唯一索引
password	BINARY(36)		
resourceauth	CHAR(1)	NOT NULL	
dbaauth	CHAR(1)	NOT NULL	
scheduleauth	CHAR(1)	NOT NULL	
publishauth	CHAR(1)	NOT NULL	
remotedbaauth	CHAR(1)	NOT NULL	
user_group	CHAR(1)	NOT NULL	
remarks	LONG VARCHAR		

必需 DBA 权限
SYSUSERPERM 包含口令，所以必须有 DBA 权限才能从其中 SELECT。

SYSUSERPERM 的每一行都描述一个用户 ID。

user_id 每个新的用户 ID 都有一个指定的唯一编号 (**user number**)，它是 **SYSUSERPERM** 的主键。

user_name 包含用户 ID 唯一名称的字符串。

password 用户 ID 的口令。对于特殊的用户 ID **SYS** 和 **PUBLIC**，该口令包含 NULL 值。它可防止任何人连接到这些用户 ID。

resourceauth (Y/N) 指出用户是否具有 RESOURCE 权限。必须有资源权限才能创建表。

dbaauth (Y/N)　指出用户是否具有 DBA（数据库管理员）权限。 DBA 权限非常强大，出于安全考虑，能够使用该权限用户 ID 应尽可能少。

scheduleauth (Y/N)　指出用户是否具有 SCHEDULE 权限。目前未使用。

publishauth (Y/N)　指出用户是否具有 SQL Remote 发布者权限。

remotedbaauth (Y/N)　指出用户是否具有 SQL Remote 远程 DBA 权限。

user_group (Y/N)　指出用户是否是一个组。

remarks　注释字符串。

初始化数据库时，将创建下列用户 ID：

- **SYS**　所有系统表的创建者。
- **PUBLIC**　用于记录 PUBLIC 权限的特殊用户 ID。
- **DBA**　数据库管理员用户 ID 是已初始化的系统中唯一可用的用户 ID。初始口令为 SQL。

无法连接到 **SYS** 或 **PUBLIC** 用户 ID。

SYSUSERTYPE 系统表

列名称	列类型	列约束	表约束
type_id	SMALLINT	NOT NULL	主键
creator	UNSIGNED INT	NOT NULL	外键引用 SYSUSERPERM.user_id
domain_id	SMALLINT	NOT NULL	外键引用 SYSDOMAIN
nulls	CHAR(1)	NOT NULL	
width	SMALLINT	NOT NULL	
scale	SMALLINT	NOT NULL	
type_name	CHAR(128)	NOT NULL	
"default"	LONG VARCHAR		
"check"	LONG VARCHAR		
format_str	CHAR(128)		
super_type_id	SMALLINT		外键引用 SYSUSERTYPE.type_id

每一行都包含对一种用户定义数据类型的描述。

type_id 用户定义数据类型的唯一标识号。

creator 数据类型所有者的用户号。

domain_id 该用户定义数据类型所依据的数据类型，由 **SYSDOMAIN** 表中列出的数据类型号指出。

nulls (Y/N) 指出用户定义数据类型是否允许空值。

width 字符串列的长度、数值列的精度或任何其它数据类型的存储字节数。

scale　数值数据类型列的小数点后的位数，对于所有其它数据类型该值为零。

type_name　数据类型的名称 （必须是有效标识符）。

default　数据类型的缺省值。

check　数据类型的 CHECK 条件。

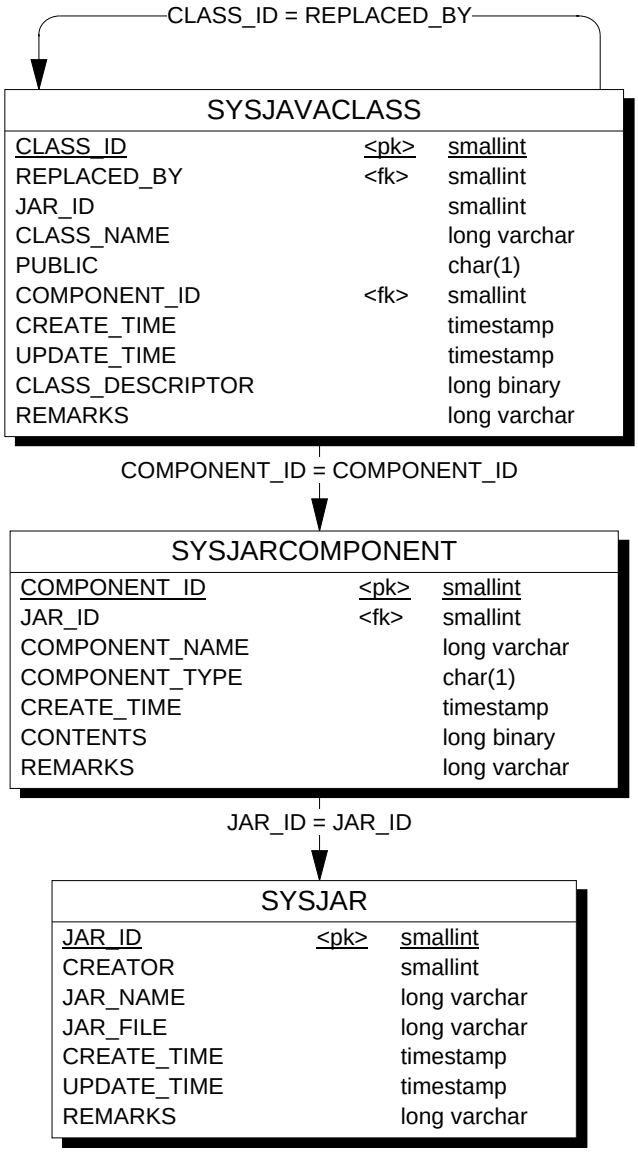
format_str　目前未使用。

其它系统表

以下是 Java 在数据库和 SQL Remote 中使用的系统表的信息。

Java 系统表

以下列出了 Java 在数据库中使用的系统表。箭头指出了表之间的外键关系：箭头从外表指向主表。



SQL Remote 系统表

有关 SQL Remote 系统表的信息，请参阅 "SQL Remote 用户指南 > Adaptive Server Anywhere 系统对象 > SQL Remote 系统表 "。

第 6 章

系统视图

关于本章 本章列出了系统表的预定义视图。

简介

" 系统表 " 第 663 页中所描述的系统表使用数字来标识表、用户 ID 等内容。虽然这提高了内部使用的效率，但也给人们解释这些表带来了不便。我们提供了若干个预定义的系统视图，这些视图以一种可读性更强的形式显示系统表中的信息。

系统视图定义

Sybase Central 中提供了有关系统视图的详细信息 （包括视图定义）：

- 要查看系统视图，请右击一个已连接的数据库，选择 " 按所有者过滤对象 "，然后选择 "SYS"。
- 打开该数据库的 " 视图 " 文件夹。
- 右击视图并选择 " 编辑 "，就可以看到视图定义。要显示数据，请在左窗格中打开 " 视图 " 文件夹并选择视图；在右窗格中选择 " 数据 " 选项卡，视图数据随即出现在右窗格中。

SYSARTICLECOLS 系统视图

显示表 SYSARTICLECOL 的可读性版本。

SYSARTICLES 系统视图

显示表 SYSARTICLE 的可读性版本。

SYSCAPABILITIES 系统视图

显示表 SYSCAPABILITY 和 SYSCAPABILITYNAME 的可读性版本。

SYSCATALOG 系统视图

以一种可读性形式列出 SYSTABLE 中的所有表和视图。

SYSCOLAUTH 系统视图

以一种可读性更强的形式显示 SYSCOLPERM 中的列更新权限信息。

SYSCOLSTATS 系统视图

以一种可读性更强的形式显示 SYSCOLSTAT 中的信息。

SYSCOLUMNS 系统视图

显示表 SYSCOLUMN 的可读性版本。

SYSFOREIGNKEYS 系统视图

以一种可读性更强的形式显示 SYSFOREIGNKEY 和 SYSFKCOL 中的外键信息。

SYSGROUPS 系统视图

以一种可读性更强的形式显示 SYSGROUP 中的组信息。

SYSINDEXES 系统视图

以一种可读性更强的形式显示 SYSINDEX 和 SYSIXCOL 中的索引信息。

SYSOPTIONS 系统视图

以一种可读性更强的形式显示表 SYSOPTION 中包含的选项设置。

SYSOPTORDERS 系统视图

此视图保留供系统使用。

SYSOPTPLAN 系统视图

此视图保留供系统使用。

SYSOPTSTRATEGIES 系统视图

此视图保留供系统使用。

SYSPROCAUTH 系统视图

以一种可读性更强的形式显示 SYSUSERPERM 中的过程权限。

SYSPROCPARMS 系统视图

以一种可读性更强的形式显示 SYSPROCPARM 中的过程参数。

SYSPUBLICATIONS 系统视图

为所有创建者显示 SYSUSERPERM 表中的用户名，并以一种可读性更强的形式显示 SYSPUBLICATION 表中的发布名称和备注。

SYSREMOTEOPTIONS 系统视图

以一种可读性更强的形式显示 SYSREMOTEOPTION 和 SYSREMOTEOPTIONTYPE 中的数据。

SYSREMOTETYPES 系统视图

以一种可读性更强的形式显示 SYSREMOTETYPES 中的过程远程类型。

SYSREMOTEUSERS 系统视图

以一种可读性更强的形式显示 SYSREMOTEUSER 中的信息。

SYSSUBSCRIPTIONS 系统视图

以一种可读性更强的形式显示 SYSPUBLICATION 表中的预订信息，如发布名称、创建时间和启动时间。

SYSSYNCDEFINITIONS 系统视图

MobiLink 同步的同步定义视图。此视图已被否决。

SYSSSYNCPUBLICATIONDEFAULTS 系统视图

与 MobiLink 同步中涉及的发布关联的缺省同步设置视图。

SYSSSYNCS 系统视图

SYSSYNCHPUBLICATIONDEFAULTS, **SYSSYNCHUSERS** 和 **SYSSYNCHSUBSCRIPTIONS** 的联合。

SYSSYNCSITES 系统视图

MobiLink 同步的同步站点视图。此视图已被否决。

SYSSYNCSUBSCRIPTIONS 系统视图

与 MobiLink 同步预订关联的同步设置视图。

SYSSYNCTEMPLATES 系统视图

与 MobiLink 同步模板关联的同步设置视图。此视图已被否决。

SYSSYNCUSERS 系统视图

与 MobiLink 同步用户关联的同步设置视图。

SYSTABAUTH 系统视图

以一种可读性更强的形式显示 SYSTABLEPERM 中的表权限信息。

SYSTRIGGERS 系统视图

以一种可读性形式列出 SYSTRIGGER 中的所有触发器。

SYSUSERAUTH 系统视图

显示表 SYSUSERPERM 中的所有信息 （用户号除外）。此视图显示口令，因此这种系统视图没有 PUBLIC 选择权限。（所有其它的系统视图都有 PUBLIC 选择权限。）

SYSUSERLIST 系统视图

显示 SYSUSERAUTH 中的所有信息（口令除外）。

SYSUSEROPTIONS 系统视图

显示对每个用户都有效的永久选项设置。如果用户没有某选项的设置，此视图将显示该选项的公共设置。

SYSUSERPERMS 系统视图

除省略口令外，包含的信息与 SYSUSERPERM 表完全相同。所有用户都可以读取此视图，但只有 DBA 有权访问基表 (SYSUSERPERM)。

SYSVIEWS 系统视图

列出视图以及它们的定义。

Transact-SQL 兼容性视图

Adaptive Server Enterprise 和 Adaptive Server Anywhere 有不同的系统目录，体现了这两个产品的不同用途。

在 Adaptive Server Enterprise 中，单一的 master 数据库包含一组系统表，其中的信息适用于服务器上的所有数据库。 master 数据库可能包括许多数据库的系统信息，而且每个数据库都有另外的系统表与之关联。

在 Adaptive Server Anywhere 中，每个数据库都是独立存在的，而且有自己的系统表。数据库集合没有包含系统信息的 master 数据库。每台服务器都可以同时运行多个数据库，根据需要动态装载和卸载每个数据库。

Adaptive Server Enterprise 和 Adaptive Server Anywhere 的系统目录是有区别的。 Adaptive Server Enterprise 系统表和视图由特殊用户 *dbo* 所拥有，而且一部分存在于 master 数据库中，一部分存在于 **sybsecurity** 数据库中，还有一部分存在于各个单独的数据库中；而 Adaptive Server Anywhere 系统表和视图由特殊用户 SYS 所拥有，并且分别存在于每个数据库中。

为帮助准备兼容的应用程序， Adaptive Server Anywhere 提供了一套由特殊用户 *dbo* 所拥有的视图，这些视图与 Adaptive Server Enterprise 系统表和视图相对应。当结构上的不同使具体某个 Adaptive Server Enterprise 表或视图的内容在 Adaptive Server Anywhere 环境中没有意义时，视图是空的，仅包含列名称和数据类型。

下面几个表列出了 Adaptive Server Enterprise 系统表和它们在 Adaptive Server Anywhere 系统目录中的实现。在每个 DBMS 中， *dbo* 是所有表的所有者。

存在于每个 Adaptive Server Enterprise 数据库中的表

表名	说明	数据？
sysalternates	每个映射到一个数据库用户的用户都占一行	否
syscolumns	表或视图中的每一列都占一行，过程中的每个参数都占一行	是
syscomments	每个视图、规则、缺省值、触发器和过程都占一行或多行，以给出 SQL 定义语句	是

表名	说明	数据？
sysconstraints	与表或列关联的每个参照约束或检查约束都占一行	否
sysdepends	过程、视图或触发器引用的每个过程、视图或表都占一行	否
sysindexes	每个聚簇索引或非聚簇索引都占一行，每个没有索引的表都占一行，每个包含文本或图像数据的表另外占一行。	是
syskeys	每个主键、外键或公用键都占一行；由用户设置 （而不是由 Adaptive Server Enterprise 维护）	否
syslogs	事务日志	否
sysobjects	每个表、视图、过程、规则、触发器、缺省值、日志或 （仅在 tempdb 中）临时对象都占一行	仅包含兼容数据
sysprocedures	每个视图、规则、缺省值、触发器或过程都占一行，以给出内部定义	否
sysprotects	用户权限信息	否
sysreferences	在表或列上声明的每个参照完整性约束都占一行	否
sysroles	将服务器范围内的角色映射到本地数据库组	否
syssegments	每一段 （指定的磁盘块集合）占一行	否
systhresholds	为数据库定义的每个阈值都占一行	否
systypes	每个系统提供或用户定义的数据类型都占一行	是
sysusermessages	每个用户定义的消息都占一行	是 （这是一个 Adaptive Server Anywhere 系统表）
sysusers	数据库允许的每个用户都占一行	是

存在于 Adaptive Server Enterprise master 数据库中的表

表名	说明	数据？
syscharsets	每个字符集或排序顺序都占一行	否
sysconfigures	每个用户可设置的配置参数都占一行	否
syscurconfigs	有关服务器当前正在使用的配置参数的信息	否
sysdatabases	服务器上的每个数据库都占一行	否
sysdevices	每个磁带转储设备、磁盘转储设备、数据库磁盘或数据库磁盘分区都占一行	否
sysengines	每个当前联机的服务器都占一行	否
syslanguages	服务器已知的每种语言 （美国英语除外）都占一行	否
syslocks	有关活动锁的信息	否
sysloginroles	每个拥有系统定义角色的服务器登录都占一行	否
syslogins	每个有效用户帐户都占一行	是
sysmessages	每个系统错误或警告都占一行	否
sysprocesses	有关服务器进程的信息	否
sysremotelogins	每个远程用户都占一行	否
sysservers	每个远程服务器都占一行	否
syssrvroles	每个服务器范围内的角色都占一行	否
sysusages	每个分配给数据库的磁盘块都占一行	否

存在于 Adaptive Server Enterprise sybsecurity 数据库中的表

表名	说明	数据？
sysaudits	每个审计记录都占一行	否
sysauditoptions	每个全局审计选项都占一行	否

第 7 章

系统过程和函数

关于本章

本章将介绍 Adaptive Server Anywhere 数据库中的分类存储过程，它们由系统提供，用来检索系统信息。本章还将介绍系统提供的扩展过程，包括用于在 MAPI 电子邮件系统上发送电子邮件的过程。

系统过程概述

Adaptive Server Anywhere 中包含以下几种系统过程：

- 分类存储过程，用于以表格形式显示系统信息。
- 扩展存储过程，用于提供 MAPI 电子邮件支持和其它函数。
- Transact-SQL 系统和分类过程。

  有关这些系统过程的列表，请参阅 "Adaptive Server Enterprise 系统和分类过程 " 第 814 页。
- 作为存储过程实现的系统函数。

  有关信息，请参阅 " 系统函数 " 第 115 页。

本章将介绍分类存储过程以及用于提供 MAPI 电子邮件支持和其它外部函数的扩展存储过程。

系统过程和函数定义

Sybase Central 中有关于系统过程和函数的详细信息：

- 要查看系统过程和函数，请右击某个已连接的数据库，选择 " 按所有者过滤对象 "，然后选择 "DBO"。
- 打开该数据库的 " 过程和函数 " 文件夹。
- 右击一个过程并选择 " 编辑 "，就会看到过程的定义。

系统和分类存储过程

系统和分类存储过程由用户 ID *dbo* 所拥有。其中一些过程供系统内部使用。本节只介绍不仅仅供系统内部使用的过程。在 Windows CE 上不能调用外部函数。

sa_audit_string 系统过程

作用　将字符串添加到事务日志中。

语法　**sa_audit_string (** *string* **)**

权限　需要 DBA 权限

副作用　无

另请参阅　"ASA 数据库管理指南 > 数据库选项 > AUDITING 选项 [database]"

"ASA 数据库管理指南 > 保护数据的安全 > 审计数据库活动 "

说明　如果审计功能是打开的，此系统过程会在审计日志中添加一条注释。字符串的最大长度可以是 200 字节。

示例　以下调用在审计日志中添加一条注释：

```
CALL sa_audit_string( 'Auditing test' )
```

sa_check_commit 系统过程

作用　在提交前检查是否存在未完成的参照完整性违规。

语法　**sa_check_commit(** *out_table_name*, *out_key_name* **)**

权限　无

副作用　无

另请参阅

"ASA 数据库管理指南 > 数据库选项 > WAIT_FOR_COMMIT 选项 [database]"

"CREATE TABLE 语句 " 第 395 页

说明

如果数据库选项 WAIT_FOR_COMMIT 为 ON，或者如果在 CREATE TABLE 语句中使用 CHECK ON COMMIT 定义了外键，您可以以违反参照完整性的方式更新数据库，只要在提交更改前解决了这些违规即可。

您可以使用 *sa_check_commit* 系统过程，在试图提交更改前检查是否存在未完成的参照完整性违规情况。

返回的参数表示某行当前违反参照完整性的表的名称和对应的外键索引名。

示例

以下命令集可以通过 Interactive SQL 执行。它从示例数据库的 *department* 表中删除行，以此方式违反参照完整性。 *sa_check_commit* 调用将检查哪些表和键有未完成的违规，然后回退以取消此更改：

```
SET TEMPORARY OPTION WAIT_FOR_COMMIT='ON'
go
DELETE FROM department
go
CREATE VARIABLE tname VARCHAR( 128 );
CREATE VARIABLE keyname VARCHAR( 128 );
go
CALL sa_check_commit( tname, keyname )
go
SELECT tname, keyname
go
ROLLBACK
go
```

sa_conn_activity 系统过程

作用

为服务器上的每个数据库连接返回一个最近准备的 SQL 语句。

语法

sa_conn_activity

权限

无

副作用　无

说明　*sa_conn_activity* 过程为每个连接都返回一个包含最近准备的 SQL 语句的结果集。当数据库服务器忙而您想获取为每个连接准备的 SQL 语句的信息时，此过程很有用。此功能可以替代请求级记录。

有关派生了这些值的属性的信息，请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "。

sa_conn_compression_info 系统过程

作用　提供压缩比摘要信息。

语法　**sa_conn_compression_info (** [*connection-id*] **)**

权限　无

副作用　无

说明　*sa_conn_compression_info* 过程为每个提供的连接都返回一个包含以下连接属性的结果集。如果没有提供连接 ID，则此系统过程返回服务器上的所有当前数据库连接的信息。参数包括：

Type　一个字符串，用于确定它后面的压缩统计是表示一个连接还是所有服务器连接。

ConnNumber　一个表示连接 ID 的整数。如果 Type 是 Server，则返回 NULL。

Compression　一个字符串，表示连接是否启用了压缩。如果 Type 是 Server，则返回 Null。如果 Type 是 Connection，则返回 ON/OFF。

TotalBytes　一个整数，表示发送和收到的实际字节总数。

TotalBytesUncomp　一个整数，表示当禁用压缩时本来会发送和收到的字节数。

CompRate　一个 numeric (5,2)，表示总压缩比。例如，值 0 表示没有发生压缩。值 75 表示数据压缩比是 75%，即一直压缩到原大小的四分之一。

CompRateSent 一个 numeric (5,2)，表示发送到客户端的数据的压缩比。

CompRateReceived 一个 numeric (5,2)，表示从客户端收到的数据的压缩比。

TotalPackets 一个整数，表示发送和收到的实际数据包总数。

TotalPacketsUncomp 一个整数，表示当禁用压缩时本来会发送和收到的数据包总数。

ComPktRate 一个 numeric (5,2)，表示数据包的总压缩比。

CompPktRateSent 一个 numeric (5,2)，表示发送到客户端的数据包的压缩比。

CompPktRateReceived 一个 numeric (5,2)，表示从客户端收到的数据包的压缩比。

有关派生了这些值的属性的信息，请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "。

sa_conn_info 系统过程

作用 报告连接属性信息。

语法 **sa_conn_info (** [*connection-id*] **)**

权限 无

副作用 无

说明 为提供的连接返回包含以下连接属性的结果集。如果没有提供连接 ID，则返回服务器上的所有当前数据库连接的信息。

- Number
- Name
- Userid

- DBNumber
- LastReqTime
- ProcessTime
- Port
- ReqType
- CommLink
- NodeAddr
- LastIdle
- CurrTaskSwitch
- BlockedOn
- UncmtOps
- LockName

当发生阻塞时，使用此过程所返回的 BlockedOn 值可以检查哪些用户被阻塞以及是谁阻塞的。 *sa_locks* 过程可以用来显示阻塞连接控制的锁，并且如果 A 控制多个表的锁，您可以在 *sa_locks* 和 *sa_conn_info* 之间匹配 LockName 值。

有关这些属性的信息，请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "。

sa_conn_properties 系统过程

作用　报告连接属性信息。

语法　**sa_conn_properties (** [*connection-id*] **)**

权限　无

副作用　无

另请参阅

"sa_conn_properties_by_conn 系统过程 " 第 760 页

"sa_conn_properties_by_name 系统过程 " 第 761 页

" 系统函数 " 第 115 页

"ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "

说明

以 Number 形式返回连接 ID，以及返回每个可用连接属性的 PropNum、PropName、 PropDescription 和 Value。

如果没有提供 *connection-id*，则返回所有当前的服务器连接的属性。

sa_conn_properties_by_conn 系统过程

作用

报告连接属性信息。

语法

sa_conn_properties_by_conn ([*property-name*] **)**

权限

无

副作用

无

另请参阅

"sa_conn_properties 系统过程 " 第 759 页

"ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "

说明

这是 *sa_conn_properties* 系统过程的一种变化形式，它返回同样的结果列。它仅返回匹配 *property-name* 字符串的连接属性的结果。在 *property-name* 中可以使用通配符，因为比较时使用 LIKE 运算符。结果集按连接号和属性名排序。

有关可用连接属性的列表，请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "。

示例

- 以下语句返回所有连接的 AnsiNull 选项设置：

```
CALL sa_conn_properties_by_conn( 'ansinull' )
```

- 以下选项返回所有连接的 ANSI 相关选项设置：

```
CALL sa_conn_properties_by_conn( 'ansi%' )
```

sa_conn_properties_by_name 系统过程

作用　报告连接属性信息。

语法　**sa_conn_properties_by_name (** [*connection-id*] **)**

权限　无

副作用　无

另请参阅　"sa_conn_properties 系统过程 " 第 759 页

"ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 "

说明　这是 *sa_conn_properties* 系统过程的一种变化形式，它返回同样的结果列。返回的信息按属性名和连接号排序。

有关可用连接属性的列表，请参阅 " 系统函数 " 第 115 页。

sa_db_info 系统过程

作用　报告数据库属性信息。

语法　**sa_db_info (** [*database-id*] **)**

权限　无

副作用　无

另请参阅　"sa_db_properties 系统过程 " 第 762 页

"ASA 数据库管理指南 > 数据库性能和连接属性 > 数据库级属性 "

说明　返回一行，其中包含了指定数据库的 Number、 Alias、 File、 ConnCount、 PageSize 和 LogName。

示例

- 以下语句返回一行，其中描述了当前数据库：

```
CALL sa_db_info
```

以下是示例值：

属性	值
Number	0
Alias	asademo
File	C:\program files\Sybase\SQL Anywhere 8\asademo.db
ConnCount	1
PageSize	1024
LogName	C:\program files\Sybase\SQL Anywhere 8\asademo.log

sa_db_properties 系统过程

作用 报告数据库属性信息。

语法 **sa_db_properties (** [*database-id*] **)**

权限 无

副作用 无

另请参阅 "sa_db_info 系统过程 " 第 761 页

"ASA 数据库管理指南 > 数据库性能和连接属性 > 数据库级属性 "

说明 返回数据库 ID 号以及每个可用数据库属性的 Number、 PropNum、 PropName、 PropDescription 和 Value。

sa_disk_free_space 系统过程

作用　报告 dbspace、事务日志、事务日志镜像和 / 或临时文件的可用空间信息。

语法　**sa_disk_free_space (** [*string*] **)**

参数　*string* 可以是 *dbspace-name*、 **log**、 **mirror** 或 **temp** 中的一个。

如果有一个名为 log、 mirror 或 temp 的 dbspace，您可以在关键字前面加上一个下划线。例如，当存在一个名为 log 的 dbspace 时，使用 **_log** 可以获得日志文件的信息。

如果没有指定 *string* 或者它为空，则结果集中每个 dbspace 都占一行，另外每个事务日志、事务日志镜像和临时文件 （如果它们存在）也都占一行。如果指定了 *string*，则正好返回一行或零行 （如果不存在这样的 dbspace，或者指定了 "log" 或 "mirror" 但是没有日志文件或镜像文件，则返回零）。

权限　需要 DBA 权限

副作用　无

说明　结果集中有两列：一列是 dbspace 或文件名，一列是卷上的可用字节数。

sa_eng_properties 系统过程

作用　报告数据库服务器属性信息。

语法　**sa_eng_properties**

权限　无

副作用　无

另请参阅　"ASA 数据库管理指南 > 数据库性能和连接属性 > 服务器级属性 "

说明 返回每个可用服务器属性的 PropNum、 PropName、 PropDescription 和 Value。

有关可用引擎属性的列表，请参阅 " 系统函数 " 第 115 页。

示例

- 以下语句返回可用服务器属性的集合

```
CALL sa_eng_properties()
```

PropNum	PropName	…
1	IdleWrite	…
2	IdleChkPt	…
…	…	…

sa_flush_cache 系统过程

作用 清空数据库服务器高速缓存中的所有页面。

语法 **sa_flush_cache ()**

权限 需要 DBA 权限

副作用 无

说明 数据库管理员可以使用此过程清空数据库服务器高速缓存中的内容。这用于在性能测量中确保得到可重复的结果。

sa_flush_statistics 系统过程

作用 保存在数据库服务器高速缓存中的所有成本模型统计。

语法 **sa_flush_statistics ()**

权限 需要 DBA 权限

副作用 无

说明　使用此过程，数据库管理员可以确保数据库服务器高速缓存中已经创建和 / 或收集但尚未保存到磁盘上的成本模型统计立即被刷新。通常操作情况下不需要执行此过程，因为服务器自动定期将统计写入磁盘。

sa_get_dtt 系统过程

作用　报告磁盘传送时间 (DTT) 模型 （成本模型的一部分）的当前值。

语法　**sa_get_dtt (** *file-id* **)**

权限　无。

副作用　无。

说明　从系统表 **SYSFILE** 中可以获得 *file-id*。

此过程检索系统表 **SYSOPTSTAT** 中的数据。其目的是用于内部诊断。

sa_get_histogram 系统过程

作用　检索列的直方图。

语法　**sa_get_histogram (** *column*, *table* [, *owner*] **)**

权限　需要 SELECT 权限。

副作用　无。

另请参阅　"ASA 数据库管理指南 > 数据库管理实用程序 > Histogram 实用程序 "

"ESTIMATE 函数 [杂类]" 第 151 页

"ESTIMATE_SOURCE 函数 [杂类]" 第 152 页

说明　此过程检索系统表 **SYSCOLSTAT** 中的数据。其目的是用于内部诊断。建议使用 Histogram 实用程序查看直方图。

要确定关于字符串列的判定的选择性，应该使用 ESTIMATE 或 ESTIMATE_SOURCE 函数。对于字符串列， sa_get_histogram 和 Histogram 实用程序从 **SYSCOLSTAT** 中都检索不到任何内容。

sa_get_request_profile 系统过程

作用 分析请求级日志以确定类似语句的执行时间。

语法 **sa_get_request_profile (** [*request-log-filename* [, *connection-id*]] **)**

权限 需要 DBA 权限。

副作用 自动提交。

另请参阅 "sa_get_request_times 系统过程 " 第 766 页

"sa_statement_text 系统过程 " 第 793 页

"sa_server_option 系统过程 " 第 790 页

说明 此过程调用 **sa_get_request_times** 以处理日志文件，然后将结果汇总到全局临时表 *satmp_request_profile* 中。该表包含日志中的语句，以及每个语句的执行次数和它们的总执行时间、平均和最大执行时间。该表可以按多种方式排序以确定性能优化目标。

如果不指定日志文件，则缺省为当前日志文件，该文件在命令提示符下由 -zo 指定，或者已由以下语法指定：

sa_server_option('request_level_log_file', *filename* **)**

如果指定了连接 ID，则用它来过滤日志中的信息，仅检索该连接的请求。

sa_get_request_times 系统过程

作用 分析请求级日志以确定语句执行时间。

语法 **sa_get_request_times (** [*request-log-filename* [, *connection-id*]] **)**

权限　需要 DBA 权限。

副作用　自动提交。

另请参阅　"sa_get_request_profile 系统过程 " 第 766 页

"sa_statement_text 系统过程 " 第 793 页

"sa_server_option 系统过程 " 第 790 页

说明　此过程读取指定的请求级日志，并用日志中的语句和它们的执行时间填充全局临时表 *satmp_request_time*。

对于像插入和更新这样的语句，执行时间简单明了。而对于查询，时间是从准备语句开始一直计算到将其删除，包括描述语句、打开游标、读取行和关闭游标等一系列操作。对于大多数查询，这个值精确反映了查询所用的时间。而在有些情况中，当执行其它操作时游标仍然是打开的，这时时间显示为一个很大的值，但这并不是查询实际所用的时间。

如果不指定日志文件，则缺省为当前日志文件，该文件在命令提示符下由 -zo 指定，或者已由以下语法指定：

sa_server_option('request_level_log_file', *filename* **)**

如果指定了连接 ID，则用它来过滤日志中的信息，仅检索该连接的请求。

sa_get_server_messages 系统过程

作用　用于以结果集的形式返回服务器的消息窗口。

语法　**sa_get_server_messages (** *integer* **)**

权限　无。

副作用　无。

说明　此过程接受一个指定要显示的起始行号的整数参数，并为那一行和它后面的所有行返回一行。如果起始行是负数，则结果集从第一个可用行开始。结果集中包含行号、消息文本和消息时间。

sa_index_density 系统过程

作用 报告数据库索引的分段量信息。

语法 **sa_index_density (** [*table_name* [, *owner_name*]] **)**

权限 需要 DBA 权限

副作用 无

另请参阅 "ASA SQL 用户指南 > 监控和提高性能 > 索引碎片 "

说明 数据库管理员可以使用此过程获取数据库索引的分段程度信息。

此过程返回包含表名、索引名、叶页数和索引密度的结果集。密度是介于 0 和 1 之间的分数。

如果不指定参数，则显示所有表的信息。否则，此过程仅检查指定的表。

Interactive SQL 的 " 结果 " 窗格中显示以下内容的表结果集：

TableName, IndexName, LeafPages, Density

密度是介于 0 和 1 之间的分数。对于有较多叶页的索引，最好使用高密度值。

sa_index_levels 系统过程

作用 通过报告索引的级数，帮助优化性能。

语法 **sa_index_levels (** [*table_name* [, *owner_name*] **)**

权限 需要 DBA 权限

副作用 无

另请参阅 "CREATE INDEX 语句 " 第 341 页

"ASA SQL 用户指南 > 监控和提高性能 > 使用索引 "

说明　索引树的级数决定了使用索引访问一行所需的 I/O 操作数。级别少的索引比级别多的索引效率高。

此过程返回包含表名、索引名和索引级数的结果集。

如果不提供参数，则返回数据库中的所有索引的级数。如果仅提供 *table_name*，则提供那个表的所有索引的级数。如果 *table_name* 为 NULL 并且提供了 *owner_name*，则仅返回那个用户所拥有的表的索引的级数。

sa_java_loaded_classes 系统过程

作用　列出当前由数据库虚拟机装载的类。

语法　**sa_java_loaded_classes ()**

权限　无

副作用　无

另请参阅　"ASA 编程指南 > 在数据库中使用 Java > 将 Java 类安装到数据库中 "。

说明　返回的结果集包含当前由数据库 Java 虚拟机装载的所有 Java 类名。

当第一次调用虚拟机时，它装载若干类。如果在调用 **sa_java_loaded_classes** 之前没有先在数据库功能中使用 Java，则它返回这个类集。

此过程在诊断遗失的类时很有用。它还可以用来确定具体某个 jar 中的哪些类被给定的应用程序使用。

sa_locks 系统过程

作用　显示数据库中的所有锁。

语法　**sa_locks (** [*connection*,] [[*owner*.]*table_name*,] [*max_locks*] **)**

权限　需要 DBA 权限

副作用 无

另请参阅 "ASA SQL 用户指南 > 使用事务和隔离级别 > 锁定的工作方式 "

说明 *sa_locks* 过程返回的结果集包含数据库中所有锁的信息。

输入参数如下：

connection 一个整数，表示连接 ID。此过程仅返回指定连接的锁信息。缺省值为零，此情况下返回所有连接的信息。

table_name 一个 char(128) 参数，表示一个表名。此过程仅返回指定表的信息。缺省值为 NULL，此情况下返回所有表的信息。

如果不提供 *owner*，则假定表为过程的调用者所有。

max_locks 一个整数参数，表示为其返回信息的最大锁数。缺省值为 1000。值 -1 表示返回所有锁信息。

为每个锁返回的信息包括以下内容：

connection ID 拥有锁的连接 ID。

user name 通过连接 ID 连接的用户。

table name 控制锁的表。

lock type 锁类型是一个表示锁类型的字符串。对于非 NULL 的 lock_name，这些字符是：

- **S** 共享
- **E** 排它
- **P** 幻像
- **A** 防幻像

所有列出的锁必须是指定的 S 或 E 中的一个，还可能有 P、 A 或者两者。如果一个锁是幻像或防幻像锁，则锁类型中会添加一个限定符。限定符如下：

- **T** 锁针对有序扫描

- ***** 锁针对所有扫描。
- **nnn** 一个索引号。锁针对具体某个索引。

当 lock_name 为 NULL 时， lock_type 可以是以下锁的组合：

- **S** 共享模式锁
- **E** 排它模式锁
- **AT** 共享行锁
- **PT** 行意图模式锁

lock_name 标识锁的 **LockName** 值。此值可与 *sa_conn_info* 输出进行匹配，以确定导致阻塞情况的锁。

Lock_name 可以是行 ID 或为 NULL。

NULL lock_name

如果 lock_name 为 NULL，则此行包含两类锁的信息：模式锁和行锁。

模式锁表示禁止其它事务修改表模式。在共享 (S) 或排它 (E) 模式下可获得此模式锁。

行锁适用于表中的行。在共享或意图模式下可获得此锁。共享模式由锁类型 AT 表示，意图模式由锁类型 PT 表示。如果在共享模式下获得，则其它事务不能修改行，除非它们在意图模式下获得此锁。然而，如果表上没有由其它事务所做的未提交修改，则只能在共享模式下获得此锁。

例如，如果一个连接修改了表但尚未提交或回退，则 sa_locks 会返回表的 NULL lock_name 和至少是 SPT 的 lock_type。 S 表示表模式的共享锁， PT 表示表行的意图锁。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 连接级属性 " 和 "sa_conn_info 系统过程 " 第 758 页。

sa_make_object 系统过程

作用

在 SQL 脚本中， **sa_make_object** 过程可用来确保在执行 ALTER 语句（提供实际的定义）前已存在对象的框架实例。

语法

sa_make_object (
objtype,
objname,
[, *owner* [, *tabname*,] **)**

object-type:
'procedure' | 'function' | 'view' | 'trigger'

权限

创建或修改数据库对象需要有资源权限。

副作用

自动提交。

另请参阅

"ALTER FUNCTION 语句 " 第 250 页

"ALTER PROCEDURE 语句 " 第 252 页

"ALTER TRIGGER 语句 " 第 279 页

"ALTER VIEW 语句 " 第 281 页

说明

在那些需要重复运行来创建或修改数据库模式的脚本或命令文件中，此过程尤其有用。这类脚本中的一个共同问题是：当这些脚本在第一次运行时，必须执行一个 CREATE 语句，以后运行时必须执行 ALTER 语句。使用此过程，就不必通过查询系统表来检查对象是否存在。

若要使用此过程，请在此过程后使用一个包含整个对象定义的 ALTER 语句。

参数

object-type 要创建的对象的类型。此参数必须是 **'procedure'**、**'function'**、 **'view'** 或 **'trigger'**。

objname 要创建的对象的名称。

owner 要创建的对象的所有者。缺省值为 CURRENT USER。

仅在 objtype 为 **'trigger'** 时需要它，在这种情况下它指定将要在其上创建触发器的表的名称。

示例

以下语句可确保创建框架过程定义，可以定义过程，并可授予对其的权限。您可以对数据库重复运行包含这些指令的命令文件而不会出现任何错误。

```
CALL sa_make_object( 'procedure','myproc' );
ALTER PROCEDURE myproc( in p1 int, in p2 char(30) )
BEGIN
    // ...
END;
GRANT EXECUTE ON myproc TO public;
```

sa_migrate 系统过程

作用　将一组远程表迁移到 Adaptive Server Anywhere 数据库中。

语法

sa_migrate (
local_table_owner,
server_name,
[*table_name*,]
[*owner_name*,]
[*database_name*,]
[*migrate_data*,]
[*drop_proxy_tables*]
[*migrate_fkeys*] **)**

权限　无

副作用　无

另请参阅

"sa_migrate_create_remote_table_list 系统过程 " 第 779 页

"sa_migrate_create_tables 系统过程 " 第 781 页

"sa_migrate_data 系统过程 " 第 782 页

"sa_migrate_create_remote_fks_list 系统过程 " 第 777 页

"sa_migrate_create_fks 系统过程 " 第 776 页

"sa_migrate_drop_proxy_tables 系统过程 " 第 784 页

"ASA SQL 用户指南 > 导入和导出数据 > 将数据库迁移到 Adaptive Server Anywhere"

说明

可以使用此过程将表从远程 Oracle、 DB2、 SQL Server、 Adaptive Server Enterprise、 Adaptive Server Anywhere 或 Access 数据库迁移到 Adaptive Server Anywhere 中。使用此过程，仅需要一步，即可从指定的服务器迁移一组远程表，包括它们的外键映射。 *sa_migrate* 存储过程调用以下存储过程：

- *sa_migrate_create_remote_table_list*
- *sa_migrate_create_tables*
- *sa_migrate_data*
- *sa_migrate_create_remote_fks_list*
- *sa_migrate_create_fks*
- *sa_migrate_drop_proxy_tables*

如果需要更大的灵活性，可能需要使用这些存储过程而不是 *sa_migrate*。例如，如果迁移外键关系由不同用户拥有的表，则使用 *sa_migrate* 时无法保持外键关系。

在可以迁移任何表之前，必须先使用 CREATE SERVER 语句创建远程服务器以连接到远程数据库。可能还需要使用 CREATE EXTERNLOGIN 语句创建到远程数据库的外部登录。

有关更多信息，请参阅 "CREATE SERVER 语句 " 第 364 页和 "CREATE EXTERNLOGIN 语句 " 第 335 页。

通过仅指定 *local_table_owner* 和 *server_name* 参数，可以将远程数据库中的所有表都迁移到 Adaptive Server Anywhere 数据库中。然而，如果仅指定这两个参数，所有迁移的表都将属于目标 Adaptive Server Anywhere 数据库中的一个所有者。如果表在远程数据库上有不同的所有者，并且您希望它们在 Adaptive Server Anywhere 数据库上有不同的所有者，则必须分别为每个所有者迁移表，每次调用 *sa_migrate* 过程时都指定 *local_table_owner* 和 *owner_name* 参数。要使用此过程，必须有为本地 Adaptive Server Anywhere 用户创建表的必要权限。

注意

不要将 *table_name* 和 *owner_name* 参数都指定为 NULL。

如果为 *table_name* 和 *owner_name* 两个参数都提供 NULL 值，则会迁移数据库中的所有表，包括系统表。此外，远程数据库中那些名称相同但所有者不同的表在目标数据库中均属同一所有者。建议您一次迁移与同一所有者相关联的表。

参数

local_table_owner　目标 Adaptive Server Anywhere 数据库中拥有迁移表的用户。使用 GRANT CONNECT 语句创建此用户。此参数需要值。

有关更多信息，请参阅 "GRANT 语句 " 第 497 页。

server_name　用于连接到远程数据库的远程服务器的名称。使用 CREATE SERVER 语句创建此服务器。此参数需要值。

有关更多信息，请参阅 "CREATE SERVER 语句 " 第 364 页。

table_name　如果要迁移单个表，请使用 *table_name* 参数指定那个表的名称。否则，应该为此参数指定 NULL （缺省值）。不要将 *table_name* 和 *owner_name* 参数都指定为 NULL。

owner_name　如果仅迁移属于一个所有者的表，请使用 *owner_name* 参数指定该所有者的名称。否则，应该为此参数指定 NULL （缺省值）。不要将 *table_name* 和 *owner_name* 参数都指定为 NULL。

database_name　远程数据库的名称。如果想仅迁移远程服务器上的一个数据库中的表，必须指定数据库名。否则，为此参数键入 NULL （缺省值）。

migrate_data　指定是否迁移远程表中的数据。此参数可为 0 （不迁移数据）或 1 （迁移数据）。缺省情况下迁移数据。

drop_proxy_tables　指定迁移完成后是否删除为迁移进程创建的代理表。此参数可为 0 （不删除代理表）或 1 （删除代理表）。缺省情况下删除代理表。

migrate_fkeys　指定迁移完成后是否迁移外键映射。此参数可为 0 （不迁移外键映射）或 1 （迁移外键映射）。缺省情况下迁移外键映射。

示例

以下语句迁移远程数据库中所有属于用户 *p_chin* 的表 （包括外键映射），迁移远程表中的数据，并且在迁移完成后删除代理表。此例中，所有迁移的表都属于目标 Adaptive Server Anywhere 数据库中的 *local_user*。

```
CALL sa_migrate( 'local_user', 'server_a', NULL,
'p_chin', NULL, 1, 1, 1 )
```

以下语句仅迁移属于远程数据库中的用户 *remote_a* 的表。在目标 Adaptive Server Anywhere 数据库中，这些表属于用户 *local_a*。

```
CALL sa_migrate( 'local_a', 'server_a', NULL,
'remote_a', NULL, 1, 0, 1 )
```

sa_migrate_create_fks 系统过程

作用

为表 *dbo.migrate_remote_fks_list* 中列出的每个表创建外键。

语法

sa_migrate_create_fks (*local_table_owner* **)**

权限

无

副作用

无

另请参阅

"sa_migrate 系统过程 " 第 773 页

"sa_migrate_create_remote_table_list 系统过程 " 第 779 页

"sa_migrate_create_tables 系统过程 " 第 781 页

"sa_migrate_data 系统过程 " 第 782 页

"sa_migrate_create_remote_fks_list 系统过程 " 第 777 页

"sa_migrate_drop_proxy_tables 系统过程 " 第 784 页

"ASA SQL 用户指南 > 导入和导出数据 > 将数据库迁移到 Adaptive Server Anywhere"

说明

sa_migrate_create_fks 存储过程与其它迁移存储过程一起使用。这些过程必须按以下顺序执行：

1. *sa_migrate_create_remote_table_list*
2. *sa_migrate_create_tables*
3. *sa_migrate_data*

4. *sa_migrate_create_remote_fks_list*

5. *sa_migrate_create_fks*

6. *sa_migrate_drop_proxy_tables*

此过程为 *dbo.migrate_remote_fks_list* 表中列出的每个表创建外键。*local_table_owner* 参数所指定的用户拥有目标数据库中的外键。

如果目标 Adaptive Server Anywhere 数据库中的表不是全都属于同一个所有者，则必须为拥有您想迁移其外键的表的每个用户执行此过程。

另一种选择是，可以使用 *sa_migrate* 系统过程在一步中迁移所有的表。

参数

local_table_owner　目标 Adaptive Server Anywhere 数据库中拥有迁移外键的用户。如果想迁移属于不同用户的表，则必须为拥有您想迁移的表的每个用户执行此过程。 *local_table_owner* 是用 GRANT CONNECT 语句创建的。此参数需要值。

有关更多信息，请参阅 GRANT 语句。

示例

以下语句基于 *dbo.migrate_remote_fks_list* 表创建外键。外键属于本地 Adaptive Sever Anywhere 数据库中的用户 *local_a*。

```
CALL sa_migrate_create_fks( 'local_a' )
```

sa_migrate_create_remote_fks_list 系统过程

作用

填充 *dbo.migrate_remote_fks_list* 表。

语法

sa_migrate_create_remote_fks_list (*server_name* **)**

权限

无

副作用

无

另请参阅

"sa_migrate 系统过程 " 第 773 页

"sa_migrate_create_remote_table_list 系统过程 " 第 779 页

"sa_migrate_create_tables 系统过程 " 第 781 页

"sa_migrate_data 系统过程 " 第 782 页

"sa_migrate_create_fks 系统过程 " 第 776 页

"sa_migrate_drop_proxy_tables 系统过程 " 第 784 页

"ASA SQL 用户指南 > 导入和导出数据 > 将数据库迁移到 Adaptive Server Anywhere"

说明

sa_migrate_create_remote_fks_list 存储过程与其它迁移存储过程一起使用。这些过程必须按以下顺序执行：

1. *sa_migrate_create_remote_table_list*
2. *sa_migrate_create_tables*
3. *sa_migrate_data*
4. *sa_migrate_create_remote_fks_list*
5. *sa_migrate_create_fks*
6. *sa_migrate_drop_proxy_tables*

此过程用可从远程数据库中迁移的外键列表填充 *dbo.migrate_remote_fks_list* 表。可以从此表中删除不想迁移的外键的行。

另一种选择是，可以使用 *sa_migrate* 系统过程在一步中迁移所有的表。

参数

server_name 用于连接到远程数据库的远程服务器的名称。远程服务器是用 CREATE SERVER 语句创建的。此参数需要值。

有关更多信息，请参阅 "CREATE SERVER 语句 " 第 364 页。

示例

以下语句创建远程数据库中的外键的列表。

```
CALL sa_migrate_create_remote_fks_list ( 'local_a' )
```

sa_migrate_create_remote_table_list 系统过程

作用

填充 *dbo.migrate_remote_table_list* 表。

语法

sa_migrate_create_remote_table_list (
server_name,
[*table_name*,]
[*owner_name*,]
[*database_name*] **)**

权限

无

副作用

无

另请参阅

"sa_migrate 系统过程 " 第 773 页

"sa_migrate_create_tables 系统过程 " 第 781 页

"sa_migrate_data 系统过程 " 第 782 页

"sa_migrate_create_remote_fks_list 系统过程 " 第 777 页

"sa_migrate_create_fks 系统过程 " 第 776 页

"sa_migrate_drop_proxy_tables 系统过程 " 第 784 页

"ASA SQL 用户指南 > 导入和导出数据 > 将数据库迁移到 Adaptive Server Anywhere"

说明

sa_migrate_create_remote_table_list 存储过程与其它迁移存储过程一起使用。这些过程必须按以下顺序执行：

1. *sa_migrate_create_remote_table_list*
2. *sa_migrate_create_tables*
3. *sa_migrate_data*
4. *sa_migrate_create_remote_fks_list*
5. *sa_migrate_create_fks*
6. *sa_migrate_drop_proxy_tables*

此过程用可从远程数据库中迁移的列表填充 *dbo.migrate_remote_table_list* 表。可以从此表中删除不想迁移的远程表的行。

如果不希望所有的迁移表都属于目标 Adaptive Server Anywhere 数据库中的同一个所有者，则必须为每个拥有您想迁移的表的用户执行此过程。

另一种选择是，可以使用 *sa_migrate* 系统过程在一步中迁移所有的表。

注意
不要将 *table_name* 和 *owner_name* 参数都指定为 NULL。

如果为 *table_name* 和 *owner_name* 两个参数都提供 NULL 值，则会迁移数据库中的所有表，包括系统表。此外，远程数据库中那些名称相同但所有者不同的表在目标数据库中均属同一所有者。建议您一次迁移与同一所有者相关联的表。

参数

server_name 用于连接到远程数据库的远程服务器的名称。远程服务器是用 CREATE SERVER 语句创建的。此参数需要值。

有关更多信息，请参阅 "CREATE SERVER 语句 " 第 364 页。

table_name 要迁移的表的名称，如果迁移所有的表，则为 NULL。缺省为 NULL。不要将 *table_name* 和 *owner_name* 参数都指定为 NULL。

owner_name 拥有远程数据库中您想迁移的表的用户，如果迁移所有用户的表，则为 NULL。缺省为 NULL。不要将 *table_name* 和 *owner_name* 参数都指定为 NULL。

database_name 您想从中迁移表的远程数据库的名称或为 NULL。此参数缺省为 NULL。从 Adaptive Server Enterprise 和 Microsoft SQL Server 数据库中迁移表时，必须指定数据库名。

示例

以下语句创建属于远程数据库中所有用户的表列表。

```
CALL sa_migrate_create_remote_table_list( 'local_a',
NULL, NULL, NULL )
```

以下语句创建属于远程数据库中的用户 *remote_a* 的表列表。

```
CALL sa_migrate_create_remote_table_list( 'local_a',
NULL, 'remote_a', NULL )
```

以下语句创建名为 *mydb* 的数据库中的表列表。

```
CALL sa_migrate_create_remote_table_list( 'local_a',
NULL, NULL, 'mydb' )
```

sa_migrate_create_tables 系统过程

作用　为 *dbo.migrate_remote_table_list* 表中列出的每个表创建代理表和基表。

语法　**sa_migrate_create_tables (** *local_table_owner* **)**

权限　无

副作用　无

另请参阅

"sa_migrate 系统过程 " 第 773 页

"sa_migrate_create_remote_table_list 系统过程 " 第 779 页

"sa_migrate_data 系统过程 " 第 782 页

"sa_migrate_create_remote_fks_list 系统过程 " 第 777 页

"sa_migrate_create_fks 系统过程 " 第 776 页

"sa_migrate_drop_proxy_tables 系统过程 " 第 784 页

"ASA SQL 用户指南 > 导入和导出数据 > 将数据库迁移到 Adaptive Server Anywhere"

说明　*sa_migrate_create_tables* 存储过程与其它迁移存储过程一起使用。这些过程必须按以下顺序执行：

1. *sa_migrate_create_remote_table_list*
2. *sa_migrate_create_tables*
3. *sa_migrate_data*

4. *sa_migrate_create_remote_fks_list*

5. *sa_migrate_create_fks*

6. *sa_migrate_drop_proxy_tables*

此过程为表 *dbo.migrate_remote_table_list*（使用 *sa_migrate_create_remote_table_list* 过程创建）中列出的每个表创建基表和代理表。这些代理表和基表由 *local_table_owner* 参数指定的用户所有。此过程还为远程表在远程数据库中的新表创建同样的主键索引和其它索引。

如果确实希望所有的迁移表与目标 Adaptive Server Anywhere 数据库的表属于同一个所有者，则必须为每个将拥有迁移表的用户执行 *sa_migrate_create_remote_table_list* 过程和 *sa_migrate_create_tables* 过程。

另一种选择是，可以使用 *sa_migrate* 系统过程在一步中迁移所有的表。

参数 **local_table_owner** 目标 Adaptive Server Anywhere 数据库中拥有迁移表的用户。此用户是用 GRANT CONNECT 语句创建的。此参数需要值。

有关更多信息，请参阅 "GRANT 语句 " 第 497 页。

示例 以下语句创建目标 Adaptive Server Anywhere 数据库中的基表和代理表。这些表属于用户 *local_a*。

```
CALL sa_migrate_create_tables( 'local_a' )
```

sa_migrate_data 系统过程

作用 将远程数据库表中的数据迁移到目标 Adaptive Server Anywhere 数据库中。

语法 **sa_migrate_data (** *local_table_owner* **)**

权限 无

副作用 无

另请参阅 "sa_migrate 系统过程 " 第 773 页

"sa_migrate_create_remote_table_list 系统过程 " 第 779 页

"sa_migrate_create_tables 系统过程 " 第 781 页

"sa_migrate_create_remote_fks_list 系统过程 " 第 777 页

"sa_migrate_create_fks 系统过程 " 第 776 页

"sa_migrate_drop_proxy_tables 系统过程 " 第 784 页

"ASA SQL 用户指南 > 导入和导出数据 > 将数据库迁移到 Adaptive Server Anywhere"

说明

sa_migrate_data 存储过程与其它迁移存储过程一起使用。这些过程必须按以下顺序执行：

1. *sa_migrate_create_remote_table_list*
2. *sa_migrate_create_tables*
3. *sa_migrate_data*
4. *sa_migrate_create_remote_fks_list*
5. *sa_migrate_create_fks*
6. *sa_migrate_drop_proxy_tables*

此过程为属于 *local_table_owner* 参数所指定的用户的表，将远程数据库中的数据迁移到 Adaptive Server Anywhere 数据库中。

当目标 Adaptive Server Anywhere 数据库中的表不是全都属于同一个所有者时，您必须为每个拥有您想迁移的数据的表的用户执行此过程。

另一种选择是，可以使用 *sa_migrate* 系统过程在一步中迁移所有的表。

参数

local_table_owner　目标 Adaptive Server Anywhere 数据库中拥有迁移表的用户。此用户是用 GRANT CONNECT 语句创建的。此参数需要值。

有关更多信息，请参阅 "GRANT 语句 " 第 497 页。

示例 以下语句为属于用户 *local_a* 的表将数据迁移到目标 Adaptive Server Anywhere 数据库中。

```
CALL sa_migrate_data( 'local_a' )
```

sa_migrate_drop_proxy_tables 系统过程

作用 删除为进行迁移而创建的代理表。

语法 **sa_migrate_drop_proxy_tables (** *local_table_owner* **)**

权限 无

副作用 无

另请参阅 "sa_migrate 系统过程 " 第 773 页

"sa_migrate_create_remote_table_list 系统过程 " 第 779 页

"sa_migrate_create_tables 系统过程 " 第 781 页

"sa_migrate_data 系统过程 " 第 782 页

"sa_migrate_create_remote_fks_list 系统过程 " 第 777 页

"sa_migrate_create_fks 系统过程 " 第 776 页

"ASA SQL 用户指南 > 导入和导出数据 > 将数据库迁移到 Adaptive Server Anywhere"

说明 *sa_migrate_drop_proxy_tables* 存储过程与其它迁移存储过程一起使用。这些过程必须按以下顺序执行：

1. *sa_migrate_create_remote_table_list*
2. *sa_migrate_create_tables*
3. *sa_migrate_data*
4. *sa_migrate_create_remote_fks_list*

5. *sa_migrate_create_fks*

6. *sa_migrate_drop_proxy_tables*

此过程删除为进行迁移而创建的代理表。拥有这些代理表的用户是由 *local_table_owner* 参数指定的。

如果迁移的表不是全都属于目标 Adaptive Server Anywhere 数据库中的同一个用户，则必须为每个用户调用此过程以删除所有代理表。

另一种选择是，可以使用 *sa_migrate* 系统过程在一步中迁移所有的表。

参数　**local_table_owner**　目标 Adaptive Server Anywhere 数据库中拥有代理表的用户。此用户是用 GRANT CONNECT 语句创建的。此参数需要值。

有关更多信息，请参阅 "GRANT 语句 " 第 497 页。

示例　以下语句删除目标 Adaptive Server Anywhere 数据库中属于用户 *local_a* 的代理表。

```
CALL sa_migrate_drop_proxy_tables( 'local_a' )
```

sa_procedure_profile 系统过程

作用　报告已经在数据库中执行的过程中每一行的执行时间信息。

语法　**sa_procedure_profile (** [*p_object_name* [, *p_owner_name*] [, *p_table_name*]] **)**

权限　需要 DBA 权限

副作用　无

另请参阅　"sa_server_option 系统过程 " 第 790 页

"sa_procedure_profile_summary 系统过程 " 第 787 页

说明　在可以分析数据库之前，必须启用分析。

有关启用过程分析的更多信息，请参阅 "sa_server_option 系统过程 " 第 790 页。

结果集中包含过程中每一行的执行时间信息，以及那些行使用的时间与总过程执行时间之间的百分比。 DBA 可以使用这些分析信息来调整那些可能会降低性能的较慢的过程。此过程返回与 Sybase Central 的 " 配置文件 " 选项卡中相同的存储过程、函数、事件和触发器信息。

结果集如下：

object_type 对象的类型。可以是：

- **P** 存储过程
- **F** 函数
- **E** 事件
- **T** 触发器

object_name 存储过程、函数、事件或触发器的名称。

owner_name 对象的所有者。

table_name 与触发器关联的表 （对于其它对象类型，该值为 NULL）。

line_num 过程中的行号。

executions 该行的执行次数。

millisecs 该行的执行时间 （以毫秒计）。

percentage 特定行所需要的总执行时间的百分比。

分析会话开始前，通过调用相关的过程，可以消除第一次装载过程和访问数据库表所需要的启动时间。

参数

此过程接受三个可选参数：

p_object_name 选择具体的对象。

p_owner_name 选择属于一个所有者的所有对象。

p_table_name 选择与指定表关联的所有触发器。

如果指定这些参数中的多个，必须按以下顺序列出它们 (*p_object_name, p_owner_name, p_table_name*)。这些参数为字符串，必须用引号括起来。如果不包括任何参数，则服务器返回数据库中的所有过程的数据。

示例

以下语句返回有关 *tr_manager* 触发器的分析信息：

```
CALL sa_procedure_profile (p_object_name =
'tr_manager')
```

sa_procedure_profile_summary 系统过程

作用

报告数据库中已执行的所有过程的执行时间的汇总信息。

语法

sa_procedure_profile_summary ([*p_table_name* [, *p_owner_name*] [, *p_object_name*] [, *p_object_type*] [, *p_ordering*]] **)**

权限

需要 DBA 权限

副作用

无

另请参阅

"sa_server_option 系统过程 " 第 790 页

"sa_procedure_profile_summary 系统过程 " 第 787 页

说明

在可以分析数据库之前，必须启用分析。

有关启用过程分析的更多信息，请参阅 "sa_server_option 系统过程 " 第 790 页。

此过程显示有关使用频率以及存储过程、函数、事件和触发器效率的信息。可以使用此信息优化较慢的过程以提高数据库性能。此过程返回与 Sybase Central 的 " 配置文件 " 选项卡中相同的存储过程、函数、事件和触发器信息。

此过程返回以下结果：

object_type　对象的类型。可以是：

- **P**　存储过程
- **F**　函数

- **E** 事件
- **T** 触发器

object_name 存储过程、函数、事件或触发器的名称。

owner_name 对象的所有者。

table_name 与触发器关联的表 （对于其它对象类型，该值为 NULL）。

executions 每个过程的执行次数。

millisecs 过程的执行时间，以毫秒计。

分析会话开始前，通过调用相关的过程，可以消除第一次装载过程和访问数据库表所需要的启动时间。

参数

此过程接受五个可选参数：

p_object_name 选择具体的对象。

p_owner_name 选择属于一个所有者的所有对象。

p_table_name 选择指定表中的所有触发器。

p_object_type 选择要分析的对象类型。可以是以下值之一：

- **P** 存储过程
- **F** 函数
- **E** 事件
- **T** 触发器

p_ordering 确定结果集中的列顺序。如果没有给定值，则结果按从最长执行事件到最短执行时间的顺序列出。值和结果顺序为：

- **P** *object_type, owner_name, object_name, table_name desc*
- **N** *object_name, owner_name, table_name, object_type desc*
- **O** *owner_name, object_type, object_name, table_name desc*

- **T** *table_name, owner_name, object_name, object_type desc*
- **E** *executions desc, object_name, owner_name, table_name, object_type desc*

如果指定这些参数中的多个，必须按以下顺序列出它们 (*p_object_name, p_owner_name, p_table_name, p_object_type, p_ordering*)。如果指定了其中某一个参数，则过程仅返回与参数匹配的行；否则，服务器返回数据库中的所有过程的数据。注意，参数值为字符串，必须用引号括起来。

示例　以下语句返回有关 Product 表中由 DBA 拥有的所有触发器的分析信息：

```
CALL sa_procedure_profile_summary (p_owner_name =
'dba', p_table_name = 'Product', p_object_type = 'T')
```

sa_reset_identity 系统过程

作用　允许为表设置下一个可用的标识值。使用它可以改变下一行的自动增量值。

语法　**sa_reset_identity (** [*table_name*], [*owner*], [*new_identity_value*] **)**

权限　需要 DBA 权限。

副作用　导致在值更新后执行检查点操作。

说明　为插入到表中的行生成的下一个值将是 *new_identity_value* + 1。

new_identity_value 上不发生检查，以确保它不与表中现有的行冲突。无效的值可能会导致后面的插入失败。

参数　此过程接受三个参数：

table_name　标识要为其设置标识值的表。

owner　标识要为其设置标识值的表的所有者。

new_identity_value　标识一个编号，从该编号开始进行值计数。

示例　以下语句将标识值重新设置成 101：

```
CALL sa_reset_identity ('employee', 'dba', 100)
```

sa_server_option 系统过程

作用 当服务器正在运行时替换服务器选项。

语法 **sa_server_option (** *option_name*, *option_value* **)**

权限 需要 DBA 权限

副作用 无

说明 使用此过程，数据库管理员可以替换某些数据库服务器选项，而不需要重新启动数据库服务器。

可以重新设置的选项如下：

选项名称	值	缺省值
Disable_connections	ON 或 OFF	OFF
Liveness_timeout	整数，以秒计	120
Procedure_profiling	ON、 OFF、 RESET、 CLEAR	OFF
Quitting_time	有效的日期和时间	
Remember_last_statement	ON 或 OFF	OFF
Request_level_log_file	`Filename`	
Request_level_logging	ALL、 SQL、 NONE	NONE

disable_connections 当设置为 ON 时，不允许与数据库服务器上的任何数据库建立任何其它连接。

liveness_timeout 通过客户 / 服务器 TCP/IP 或 SPX 网络定期发送活动包以确认连接是否完整。如果网络服务器运行了一个 **liveness_timeout** 周期但没有检测到活动包，则切断连接。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -tl 服务器选项 "。

procedure_profiling　控制存储过程、函数、事件和触发器的过程分析。Sybase Central 的 " 数据库 " 属性表中同样提供了分析命令。

- **ON**　对您当前连接到的数据库启用过程分析。
- **OFF**　禁用过程分析，但使分析数据可供查看。
- **RESET**　将分析计数器恢复到零，并且不改变 ON 或 OFF 设置。
- **CLEAR**　将分析计数器恢复到零，并且禁用过程分析。

启用了分析后，可以使用 *sa_procedure_profile_summary* 和 *sa_procedure_profile store* 过程从数据库中检索分析信息。

有关过程分析的更多信息，请参阅 "ASA SQL 用户指南 > 监控和提高性能 > 建立数据库过程配置文件 "。

quitting_time　指示数据库服务器在指定的时间关闭。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -tq time 服务器选项 "。

remember_last_statement　指示数据库服务器为服务器上的每个数据库连接捕获最近准备的 SQL 语句。对于存储过程调用，仅显示最外面的过程调用，不显示过程中的语句。

按如下方式使用 **LastStatement** 属性函数，可以获得 *remember_last_statement* 设置的当前值：

```
select connection_property( 'LastStatement' )
```

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库性能和连接属性 > 服务器级属性 " 和 "ASA 数据库管理指南 > 数据库服务器 > -zl 服务器选项 "。

request_level_log_file　用来记录日志信息的文件的名称。 NULL 名会停止记入文件。

文件名中的任何反斜线字符必须写两遍，因为这是 SQL 字符串。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -zo 服务器选项 "。

request_level_logging 可以是 ALL、 SQL 或 NONE。 ON 和 ALL 是等效的。 OFF 和 NONE 是等效的。此调用对发送到数据库服务器的各 SQL 语句启用日志，以便用于疑难解答，此选项同数据库服务器选项 -zr 和 -zo 一起使用。 **request_level_debugging** 和 **request_level_logging** 设置是等效的。

如果将 **request_level_logging** 设置为 OFF，则会关闭请求级日志文件。

如果选择 SQL，则仅记录以下类型的请求：

- START DATABASE
- STOP ENGINE
- STOP DATABASE
- 语句准备
- 语句执行
- EXECUTE IMMEDIATE 语句
- 选项设置
- COMMIT 语句
- ROLLBACK 语句
- PREPARE TO COMMIT 操作
- 连接
- 断开连接
- 事务开头
- DROP STATEMENT 语句
- 游标解释
- 游标关闭
- 游标重新开始

- 错误

使用 **RequestLogging** 属性函数可以找到 *request_level_logging* 设置的当前值。

有关更多信息，请参阅 "ASA 数据库管理指南 > 数据库服务器 > -zr 服务器选项 " 和 "ASA 数据库管理指南 > 数据库性能和连接属性 > 服务器级属性 "。

示例　以下语句不允许与数据库服务器建立新连接。

```
CALL sa_server_option( 'disable_connections', 'ON')
```

sa_statement_text 系统过程

作用　格式化 SELECT 语句，使各项显示在不同的行中。当从请求级日志（所有的换行符都已被删除）中查看长语句时，此过程很有用。

语法　**sa_statement_text (** *select-statement* **)**

权限　无

副作用　无

另请参阅　"sa_get_request_times 系统过程 " 第 766 页

"sa_get_request_profile 系统过程 " 第 766 页

说明　输入的 SELECT 语句必须是字符串 （用单引号括起来）。

sa_table_fragmentation 系统过程

作用　报告数据库表的分段信息。

语法　**sa_table_fragmentation (** [*table_name* [, *owner_name*]] **)**

权限　需要 DBA 权限

副作用　无

另请参阅

"ASA SQL 用户指南 > 监控和提高性能 > 表碎片 "

"ASA SQL 用户指南 > 监控和提高性能 > 整理表碎片 "

"ASA SQL 用户指南 > 导入和导出数据 > 重建数据库 "

"REORGANIZE TABLE 语句 " 第 568 页

说明

数据库管理员可以使用此过程获得数据库表的分段信息。如果不提供参数，则返回数据库中的所有表的密度。

此过程返回包含以下列的结果集：

- **TableName** 表名
- **rows** 表中的行数
- **row_segments** 表中的行分段数
- **segs_per_row** 每行的分段数

当数据库表的分段过多时，可以运行 REORGANIZE TABLE 或重建数据库以回收磁盘空间并提高性能。

sa_table_page_usage 系统过程

作用

报告有关数据库表的页面使用情况的信息。

语法

sa_table_page_usage

权限

需要 DBA 权限

副作用

无

另请参阅

"ASA 数据库管理指南 > 数据库管理实用程序 > Information 实用程序 "

说明

- 结果包括由 Information 实用程序提供的同样信息。

有关 Information 实用程序的信息，请参阅 "ASA 数据库管理指南 > 数据库管理实用程序 > Information 实用程序 "。

sa_validate 系统过程

作用　校验数据库中的所有表。

语法　**sa_validate** [*tbl_name*,] [*owner_name*,] [*check_type*]

权限　需要 DBA 权限

副作用　无

说明　此过程相当于为数据库中的每个表调用 VALIDATE TABLE 语句。

有关信息，请参阅 "VALIDATE TABLE 语句 " 第 653 页。

tbl_name　仅校验指定的表。当为 NULL （缺省值）时，校验所有表。

owner_name　仅校验指定用户所拥有的表。当为 NULL （缺省值）时，校验所有用户的表。

check_type　当为 NULL （缺省值）时，使用 VALIDATE TABLE 语句检查每个表，没有额外的检查。 *check_type* 值可以是下列情况之一：

- **data**　使用 WITH DATA CHECK 对表进行校验。
- **index**　使用 WITH INDEX CHECK 对表进行校验。
- **full**　使用 WITH FULL CHECK 对表进行校验。
- **express**　使用 WITH EXPRESS CHECK 对表进行校验。

tbl_name、 *owner_name* 和 *check_type* 参数的值都是字符串，必须用引号括起来。

此过程返回一列，其名称为 *Messages*。如果所有表都有效，则此列包含未检测到错误。

示例　以下语句使用索引检查对 DBA 拥有的所有表进行校验：

```
CALL sa_validate (owner_name = 'DBA', check_type =
'index')
```

sp_login_environment 系统过程

作用 设置用户登录时的连接选项。

语法 **sp_login_environment**

权限 无

副作用 无

另请参阅 "ASA 数据库管理指南 > 数据库选项 > LOGIN_PROCEDURE 选项 [database]"

说明 *sp_login_environment* 是由 LOGIN_PROCEDURE 数据库选项调用的缺省过程。

建议不要编辑此过程。相反，如果要改变登录环境，请将 LOGIN_PROCEDURE 选项设置为指向另一个过程。

以下是 *sp_login_environment* 过程的文本：

```
CREATE PROCEDURE dbo.sp_login_environment()
BEGIN
  IF connection_property('CommProtocol')='TDS' THEN
    CALL dbo.sp_tsql_environment()
  END IF
END
```

sp_remote_columns 系统过程

作用 生成远程表上的列的列表以及它们的数据类型的描述。

要使用此系统过程，必须用 CREATE SERVER 语句对服务器进行定义。

语法 **sp_remote_columns** *servername*, *tablename* [, *owner*] [, *database*]

权限 无

副作用 无

另请参阅 "ASA SQL 用户指南 > 访问远程数据 "

"ASA SQL 用户指南 > 用于进行远程数据访问的服务器类 "

"CREATE SERVER 语句 " 第 364 页

说明　如果您输入 CREATE EXISTING 语句并且指定列列表，获取远程表上可用的列的列表可能会有帮助。*sp_remote_columns* 生成远程表上的列的列表以及它们的数据类型的描述。如果指定了数据库，则必须指定所有者或提供 **null** 值。

标准和兼容性

- **Sybase**　受 Open Client/Open Server 支持。

示例

- 以下示例返回 *sysobjects* 表中的列，该表位于名为 *asetest* 的 Adaptive Server Enterprise 服务器上的 *production* 数据库中。未指定所有者。

```
sp_remote_columns asetest, sysobjects,
null, production
```

sp_remote_exported_keys 系统过程

作用　提供表的信息，这些表包含指定主键表的外键。

要使用此系统过程，必须用 CREATE SERVER 语句对服务器进行定义。

语法　**sp_remote_exported_keys** *@server_name, @sp_name* [, *@sp_owner*] [, *@sp_qualifier*]

权限　无

副作用　无

另请参阅　"CREATE SERVER 语句 " 第 364 页

"ASA 入门手册 > 数据库和应用程序 > 表是由外键关联的 "

说明　此过程提供远程表的信息，该远程表包含特定主键表的外键。*sp_remote_exported_keys* 存储过程的结果集包含数据库、所有者、表、列、主键和外键的名称以及外键列的外键序列。由于基础 ODBC 和 JDBC 调用的缘故，结果集可能不同，但总是返回外键的表信息和列信息。

要使用 *sp_remote_exported_keys* 存储过程，必须用 Adaptive Server Anywhere 7.0.2 或更高版本创建或更新数据库。

参数

此过程接受四个参数：

@server_name 标识主键表所在的服务器。此参数需要值。

@sp_name 标识包含主键的表。此参数需要值。

@sp_owner 标识主键表的所有者。此参数是可选的。

@sp_qualifier 标识包含主键表的数据库。此参数是可选的。

示例

- 获取有关包含引用 *sysobjects* 表的外键的远程表的信息， *sysobjects* 表位于名为 *asetest* 的服务器上的 *production* 数据库中：

```
call sp_remote_exported_keys (@server_name='asetest',
@sp_name='sysobjects', @sp_qualifier='production')
```

sp_remote_imported_keys 系统过程

作用

提供有关远程表的信息，这些表包含对应于指定外键的主键。

要使用此系统过程，必须用 CREATE SERVER 语句对服务器进行定义。

语法

sp_remote_imported_keys *@server_name, @sp_name* [, *@sp_owner*] [, *@sp_qualifier*]

权限

无

副作用

无

另请参阅

"CREATE SERVER 语句 " 第 364 页

"ASA 入门手册 > 数据库和应用程序 > 表是由外键关联的 "

说明

外键参照对应主键表中的一行。使用此过程可以获取远程表的列表，这些表包含对应于特定外键表的主键。 *sp_remote_imported_keys* 结果集包含数据库、所有者、表、列、主键和外键的名称以及外键列的外键序列。由于基础 ODBC 和 JDBC 调用的缘故，结果集可能不同，但总是返回主键的表信息和列信息。

要使用 *sp_remote_imported_keys* 存储过程，必须用 Adaptive Server Anywhere 7.0.2 或更高版本创建或更新数据库。

参数

此过程接受四个参数：

@server_name　标识外键表所在的服务器。此参数需要值。

@sp_name　标识包含外键的表。此参数需要值。

@sp_owner　标识外键表的所有者。此参数是可选的。

@sp_qualifier　标识包含外键表的数据库。此参数是可选的。

示例

- 获取表信息，这些表包含对应于 *sysobjects* 表 （在 *asetest* 服务器上的 production 数据库）的外键的主键：

```
call sp_remote_imported_keys (@server_name='asetest',
@sp_name='sysobjects', @sp_qualifier='production')
```

sp_remote_tables 系统过程

作用

返回服务器上表的列表。

要使用此系统过程，必须用 CREATE SERVER 语句对服务器进行定义。

语法

sp_remote_tables *server_name* [, *table_name*] [, *table_owner*] [, *table_qualifier*] [, *with_table_type*]

权限

无

副作用

无

另请参阅

"ASA SQL 用户指南 > 访问远程数据 "

"ASA SQL 用户指南 > 用于进行远程数据访问的服务器类 "

"CREATE SERVER 语句 " 第 364 页

说明

当您配置数据库服务器以获取特定服务器上可用的远程表的列表时，此过程可能会有帮助。此过程返回服务器上表的列表。

此过程接受五个参数：

server_name 选择远程表所在的服务器。

table_name 选择远程表。

table_owner 选择远程表的所有者。

table_qualifier 选择数据库。

with_table_type 选择远程表的类型。此参数是位类型，接受两个值：0（缺省值）和 1。如果希望结果集中包含列出表类型的列，必须输入值 1。

with_table_type 参数仅适用于用 Adaptive Server Anywhere 7.0.2 或更高版本创建的数据库。如果在旧版本数据库中使用此参数，则返回以下错误消息：

number of parameters to function 'sp_remote_tables'

如果给定了表、所有者或数据库名，则表列表中仅包含与这些参数匹配的表。

标准和兼容性

- **Sybase** 受 Open Client/Open Server 支持。

示例

- 获取名为 *excel* 的 ODBC 数据源中可用的所有 Microsoft Excel 工作表的列表：

```
sp_remote_tables  excel
```

- 获取 *production* 数据库（在名为 *asetest* 的 Adaptive Server Enterprise 服务器中，由 *fred* 拥有）中所有表的列表：

```
sp_remote_tables  asetest, null, fred, production
```

sp_servercaps 系统过程

作用

显示远程服务器的容量信息。

要使用此系统过程，必须用 CREATE SERVER 语句对服务器进行定义。

语法

sp_servercaps *servername*

权限

无

副作用　无

另请参阅　"ASA SQL 用户指南 > 访问远程数据 "

"ASA SQL 用户指南 > 用于进行远程数据访问的服务器类 "

"CREATE SERVER 语句 " 第 364 页

说明　此过程显示有关远程服务器的容量的信息。 Adaptive Server Anywhere 使用此容量信息来确定可以向远程服务器转发多少 SQL 语句。直到 Adaptive Server Anywhere 第一次连接到远程服务器后，才填充包含服务器容量的系统表。此信息来自 SYSCAPABILITY 和 SYSCAPABILITYNAME。指定的服务器名必须与 CREATE SERVER 语句中使用的服务器名相同。

标准和兼容性

- **Sybase**　受 Open Client/Open Server 支持。

示例

- 要显示有关远程服务器 testasa 的信息，请发出以下存储过程：

```
sp_servercaps  testasa
```

sp_tsql_environment 系统过程

作用　设置用户从 jConnect 或 Open Client 应用程序连接时的连接选项。

语法　**sp_tsql_environment**

权限　无

副作用　无

另请参阅　"sp_login_environment 系统过程 " 第 796 页

"ASA 数据库管理指南 > 数据库选项 > LOGIN_PROCEDURE 选项 [database]".

说明　启动时， sp_login_environment 是由 LOGIN_PROCEDURE 数据库选项调用的缺省过程。如果连接使用的是 TDS 通信协议（即，如果是 Open Client 或 jConnect 连接），则 *sp_login_environment* 接着调用 *sp_tsql_environment*。

此过程设置数据库选项，使它们与缺省 Sybase Adaptive Server Enterprise 行为兼容。

如果希望改变缺省行为，建议创建新过程并改变 LOGIN_PROCEDURE 选项，使其指向这些新过程。

示例

- 以下是 sp_tsql_environment 过程的文本：

```
CREATE PROCEDURE dbo.sp_tsql_environment()
BEGIN
  IF db_property('IQStore')='OFF' THEN
    -- ASA datastore
    SET TEMPORARY OPTION AUTOMATIC_TIMESTAMP='ON'
  END IF;
  SET TEMPORARY OPTION ANSINULL='OFF';
  SET TEMPORARY OPTION TSQL_VARIABLES='ON';
  SET TEMPORARY OPTION ANSI_BLANKS='ON';
  SET TEMPORARY OPTION TSQL_HEX_CONSTANT='ON';
  SET TEMPORARY OPTION CHAINED='OFF';
  SET TEMPORARY OPTION QUOTED_IDENTIFIER='OFF';
  SET TEMPORARY OPTION
ALLOW_NULLS_BY_DEFAULT='OFF';
  SET TEMPORARY OPTION FLOAT_AS_DOUBLE='ON';
  SET TEMPORARY OPTION ON_TSQL_ERROR='CONTINUE';
  SET TEMPORARY OPTION ISOLATION_LEVEL='1';
  SET TEMPORARY OPTION DATE_FORMAT='YYYY-MM-DD';
  SET TEMPORARY OPTION TIMESTAMP_FORMAT='YYYY-MM-
DD HH:NN:SS.SSS';
  SET TEMPORARY OPTION TIME_FORMAT='HH:NN:SS.SSS';
  SET TEMPORARY OPTION DATE_ORDER='MDY';
  SET TEMPORARY OPTION ESCAPE_CHARACTER='OFF'
END
```

系统扩展存储过程

Adaptive Server Anywhere 数据库中包含了一组系统扩展过程。这些过程为 *dbo* 用户 ID 所有。

以下各节描述了每个存储过程。

用于 MAPI 和 SMTP 的扩展存储过程

Adaptive Server Anywhere 中包含使用 Microsoft Messaging API 标准 (MAPI) 或 Internet 标准 " 简单邮件传输协议 (SMTP)" 发送电子邮件的系统过程。这些系统过程是作为扩展存储过程来实现的：每个过程都调用外部 DLL 中的一个函数。

要使用 MAPI 或 SMTP 存储过程，必须可以从数据库服务器上访问 MAPI 或 SMTP 电子邮件系统。

这些存储过程是：

- **xp_startmail**　用指定的邮件帐户登录到 MAPI 消息系统，以启动邮件会话
- **xp_startsmtp**　用指定的邮件帐户登录到 SMTP 消息系统，以启动邮件会话
- **xp_sendmail**　将邮件消息发送给指定的用户
- **xp_stopmail**　关闭 MAPI 邮件会话
- **xp_stopsmtp**　关闭 SMTP 邮件会话

以下过程通知一组用户：备份已经完成。

```
CREATE PROCEDURE notify_backup()
BEGIN
   CALL xp_startmail( mail_user='ServerAccount',
                  mail_password='ServerPassword'
                     );
   CALL xp_sendmail( recipient='IS Group',
```

```
                        subject='Backup',
                        "message"='Backup completed'
                        );
    CALL xp_stopmail( )
END
```

以下各节讨论了邮件系统过程。

xp_startmail 系统过程

作用

在 MAPI 下启动电子邮件会话。

语法

[[*variable* =] **CALL**] **xp_startmail (**

[**mail_user** = *mail-login-name*]

[, **mail_password** = *mail-password*]

)

权限

在 NetWare 上不受支持。

说明

xp_startmail 是一个启动电子邮件会话的系统存储过程。

mail-login-name 和 *mail-password* 值是包含在邮件会话中使用的 MAPI 登录名和口令的字符串。

如果您使用的是 Microsoft Exchange，则 *mail_login_name* 参数是 Exchange 配置文件名，在过程调用中不应该包含口令。

返回代码

xp_startmail 系统过程发出以下返回代码之一：

返回代码	含义
0	成功
2	失败

xp_startsmtp 系统过程

作用

在 SMTP 下启动电子邮件会话。

语法

[[*variable* =] **CALL**] **xp_startsmtp (**

smtp_sender = *email_address*

, smtp_server = *smtp_server*

[, smtp_port = *port_number*]

[, timeout = *timeout*]

)

权限

在 NetWare 上不受支持。

说明

xp_startsmtp 是一个系统存储过程，它通过连接到 SMTP 服务器启动指定电子邮件地址的邮件会话。

email_address 是发件人的电子邮件地址

smtp_server 指定要使用的 SMTP 服务器，可以是服务器名称或 IP 地址。

= *port_number* 指定在 SMTP 服务器上要连接到的端口号。缺省值为 25。

timeout 指定取消当前的 xp_sendmail 调用前等待服务器响应的时间（以秒计）。缺省值为 60 秒。

xp_startsmtp 启动与服务器的连接。此连接可能会超时。因此，建议在执行 *xp_sendmail* 前才启动 SMTP。

SMTP 上的 *xp_sendmail* 不支持附件。

返回代码

有关返回代码的列表，请参阅 "SMTP 返回代码 " 第 807 页。

xp_sendmail 系统过程

作用

发送电子邮件消息。

语法

[[*variable* =] **CALL**] **xp_sendmail (**

recipient = *mail-address*

[, **subject** = *subject*]

[, **cc_recipient** = *mail-address*]

[, **bcc_recipient** = *mail-address*]

[, **"message"** = *message-body*]

[, **include_file** = *file-name*]

)

权限

必须执行了 *xp_startmail* 才能在 MAPI 下启动电子邮件会话，或执行了 *xp_startsmtp* 才能在 SMTP 下启动电子邮件会话。

在 NetWare 上不受支持。

说明

xp_sendmail 是一个系统存储过程，在使用 *xp_startmail* 启动了会话后，它将电子邮件消息发送给指定的收件人。此过程接受任何长度的消息。

此参数值为字符串。每个参数的长度仅受系统的可用内存量的限制。*message* 参数名需要用双引号括起来，因为 MESSAGE 是个关键字。

SMTP 上的 *xp_sendmail* 不支持附件。

返回代码

xp_sendmail 系统过程发出以下返回代码之一：

MAPI 返回代码

返回代码	含义
0	成功。
5	失败 （常规）。
11	收件人不明确。

返回代码	含义
12	没有找到附件。
13	磁盘已满。
14	内存不足。
15	会话无效。
16	文本太大。
17	文件太多。
18	收件人太多。
19	未知的收件人。

SMTP 返回代码

返回代码	含义
0	成功。
100	套接字错误。
101	套接字超时。
102	无法解析 SMTP 服务器主机名。
103	无法连接到 SMTP 服务器。
104	服务器错误，不理解响应。（例如，消息的格式不正确，或者服务器不是 SMTP）。
421	<*domain*> 服务不可用，正在关闭传输通道。
450	请求的邮件操作未执行：邮箱不可用。
451	请求的操作未执行：处理时发生本地错误。
452	请求的操作未执行：系统存储不足。
500	语法错误，命令无法识别。（这可能包括命令太长这样的错误。）
501	参数语法错误。

返回代码	含义
502	命令未实现。
503	命令的顺序错误。
504	命令参数未实现。
550	请求的操作未执行：邮箱不可用。（例如，未找到邮箱，没有访问权或者不允许转发。）
551	非本地用户，请尝试 <*forward-path*>
552	请求的邮件操作已取消：超过分配的存储。
553	请求的操作未执行：邮箱名被禁用。（例如，邮箱语法不正确。）
554	事务处理失败。

示例

以下调用向用户 ID **Sales Group** 发送一条将 `prices.doc` 文件作为邮件附件发送的消息：

```
CALL xp_sendmail(recipient='Sales Group',
      subject='New Pricing',
      include_file = 'C:\\DOCS\\PRICES.DOC'
      )
```

xp_stopmail 系统过程

作用

关闭 MAPI 电子邮件会话。

语法

[*variable* =] [**CALL**] **xp_stopmail ()**

权限

在 NetWare 上不受支持。

说明

xp_stopmail 是结束电子邮件会话的系统存储过程。

返回代码

xp_stopmail 系统过程发出以下返回代码之一：

返回代码	含义
0	成功
3	失败

xp_stopsmtp 系统过程

作用　关闭 SMTP 电子邮件会话。

语法　[*variable* =] [**CALL**] **xp_stopsmtp ()**

权限　在 NetWare 上不受支持。

说明　*xp_stopsmtp* 是结束电子邮件会话的存储过程。

返回代码　有关返回代码的列表，请参阅 "SMTP 返回代码 " 第 807 页。

其它系统扩展存储过程

包含的其它系统扩展存储过程有：

- **xp_cmdshell**　执行系统命令。
- **xp_msver**　返回包含版本信息的字符串。
- **xp_sprintf**　用格式字符串和一组输入字符串建立字符串。
- **xp_scanf**　从输入字符串和格式字符串中抽取子串。

以下各节提供了有关这些过程的更详细信息。

xp_cmdshell 系统过程

作用　从过程中执行操作系统命令。

语法　[*variable* = **CALL**] **xp_cmdshell (** *string* [**, 'no_output'**] **)**

权限

无

说明

xp_cmdshell 执行系统命令，然后将控制返回给调用环境。

升级到 8.0.2 或更高版本的数据库的缺省行为是显示输出。对于旧版本数据库，可以使用 **'no_output'** 之外的字符串作为显式参数来强制显示输出。

第二个参数仅影响 Windows 操作系统 （Windows CE 除外）。对于 UNIX，无论如何设置第二个参数，都不显示任何输出。但对于 NetWare，无论如何设置第二个参数，执行的任何命令都会在服务器控制台上显示。

示例

以下语句列出文件 `c:\temp.txt` 中当前目录下的文件

```
xp_cmdshell('dir > c:\\temp.txt')
```

下面的语句执行相同的操作，但在执行时不显示命令窗口。

```
xp_cmdshell('dir > c:\\temp.txt', 'no_output' )
```

xp_msver 系统过程

作用

检索有关数据库服务器的版本和名称信息。

语法

xp_msver (*string* **)**

此字符串必须是以下字符串之一，并且用字符串分隔符括起来。

参数	说明
ProductName	产品名 (Sybase Adaptive Server Anywhere)
ProductVersion	版本号，其后是内部版本号。格式如下： `8.0.0 (1200)`
CompanyName	返回以下字符串： `Sybase Inc.`
FileDescription	返回产品名，其后是操作系统的名称。
LegalCopyright	返回软件的版权字符串

参数	说明
LegalTrademarks	返回软件的商标信息

权限　无

另请参阅　" 系统函数 " 第 115 页

说明　*xp_msver* 返回产品、公司、版本和其它信息。

示例

- 以下语句请求版本和操作系统说明：

```
SELECT xp_msver( 'ProductVersion') Version,
xp_msver('FileDescription') Description
```

以下是输出示例：

版本	说明
8.0.0 (1912)	Sybase Adaptive Server Anywhere Windows NT

Sybase Adaptive Server Anywhere Windows NT

xp_read_file 系统过程

作用　以 LONG BINARY 变量的形式返回文件的内容。

语法　[*variable* = **CALL**] **xp_read_file (** *filename* **)**

权限　需要 DBA 权限

另请参阅　"xp_write_file 系统过程 " 第 813 页

说明　此函数读取指定文件的内容，然后以 LONG BINARY 值的形式返回结果。

文件名是相对于数据库服务器的起始目录的文件名。

此函数可用于将文件中存储的整个文档或图像插入到表中。如果无法读取文件，此函数返回 NULL。

示例

以下语句将一个图像插入到表 *t1* 的名为 *picture* 的列中 （假设其它所有列都接受 NULL）：

```
INSERT INTO t1 ( picture)
SELECT xp_read_file( 'portrait.gif' )
```

xp_sprintf 系统过程

作用

用格式字符串和一组输入字符串建立字符串

语法

[*variable* = **CALL**] **xp_sprintf (** *out-string*,
format-string
[*input-string*] **)**

权限

无

说明

xp_sprintf 用格式字符串和一组输入字符串建立字符串。格式字符串最多可以包含 50 个字符串占位符 (*%s*)。这些占位符由 *input-string* 参数填充。

所有参数都必须是少于 254 个字符的字符串。

示例

以下语句将字符串 *Hello World!* 放到变量 **mystring** 中。

```
CREATE VARIABLE mystring CHAR(254) ;
xp_sprintf( mystring, 'Hello %s', 'World!' )
```

xp_scanf 系统过程

作用

从输入字符串和格式字符串中抽取子串。

语法

[*variable* = **CALL**] **xp_scanf (** *in-string*, *format-string* [*output-string*] **)**

权限

无

说明

xp_scanf 从输入字符串和格式字符串中抽取子串。格式字符串最多可以包含 50 个字符串占位符 (*%s*)。这些占位符的值放在 *output-string* 变量中。

所有参数都必须是少于 254 个字符的字符串。

示例

- 以下语句将字符串 World! 放在 **mystring** 变量中。

```
CREATE VARIABLE mystring CHAR(254) ;
xp_scanf( 'Hello World!', 'Hello %s', mystring )
```

xp_write_file 系统过程

作用

从 SQL 语句将数据写入文件。

语法

[*variable* = **CALL**] **xp_write_file (** *filename*, *file_contents* **)**

权限

需要 DBA 权限

另请参阅

"xp_read_file 系统过程 " 第 811 页

说明

此函数将 *file_contents* 写入 *filename* 文件。如果成功则返回 0，如果失败则返回非零值。

filename 是相对于数据库服务器的当前工作目录的文件名。如果文件已存在，其内容将被覆盖。

此函数可用于将长的二进制数据卸载到文件中。

示例

请看一个含有以下列的表 *t1*：

- **filename**　相对于服务器的文件名。
- **picture**　一个包含图像的 LONG BINARY 列。

以下语句将图片卸载到指定的文件中：

```
SELECT xp_write_file( filename, picture)
FROM t1
```

Adaptive Server Enterprise 系统和分类过程

Adaptive Server Enterprise 提供了执行许多管理功能和获取系统消息的系统和分类过程。 Adaptive Server Anywhere 已实现了对这些过程中的某一些过程的支持。

系统过程是用于从系统中获取报告和更新系统表的内部存储过程。分类存储过程以表格形式从系统表中检索信息。

Adaptive Server Enterprise 系统过程

以下列表描述了 Adaptive Server Anywhere 中提供的 Adaptive Server Enterprise 系统过程。

虽然这些过程执行的功能与它们在 Adaptive Server Enterprise 和 Adaptive Server IQ 12 版之前中执行的功能相同，但是它们并不完全相同。如果您的先前存在的脚本使用这些过程，您可能需要检查一下这些过程。要查看存储过程的文本，可以在 Sybase Central 中打开它，或者在 Interactive SQL 中运行以下命令。

```
sp_helptext procedure_name
```

为了看到所有的文本，可能需要重新设置 Interactive SQL 输出的宽度，方法是选择 " 命令 "-" 选项 "，然后输入一个新的 " 限制显示列 " 值。

系统过程	说明
sp_addgroup *group-name*	将组添加到数据库中
sp_addlogin *userid, password[, defdb [, deflanguage [, fullname]]]*	将新用户帐户添加到数据库中
sp_addmessage *message-num, message_text [, language]*	将用户定义的消息添加到 SYSUSERMESSAGES 中，供存储过程 PRINT 和 RAISERROR 调用使用
sp_addtype *typename, data-type [, "identity" | nulltype]*	创建用户定义的数据类型

系统过程	说明
sp_adduser *login_name [, name_in_db [, grpname]]*	将新用户添加到数据库中
sp_changegroup *new-group-name, userid*	更改用户所在组或将用户添加到组中
sp_dboption *[dbname, optname, {true | false}]*	显示或更改数据库选项
sp_dropgroup *group-name*	从数据库中删除组
sp_droplogin *loginid*	从数据库服务器中删除用户
sp_dropmessage *message-number [, language]*	删除用户定义的消息
sp_droptype *typename*	删除用户定义的数据类型
sp_dropuser *userid*	从数据库中删除用户
sp_getmessage *message-num, @msg-var output [, language]*	从 SYSUSERMESSAGES 中检索存储消息字符串，供 PRINT 和 RAISERROR 语句使用。
sp_helptext *object-name*	显示系统过程、触发器或视图的文本
sp_password *caller_passwd, new_passwd [, userid]*	添加或更改用户 ID 的口令

Adaptive Server Enterprise 分类过程

Adaptive Server Anywhere 实现了 Adaptive Server Enterprise 分类过程的子集。下表描述了已实现的分类过程。

分类过程	说明
sp_column_privileges	不支持
sp_columns *table-name [, table-owner] [, table-qualifier] [, column-name]*	返回指定列的数据类型

分类过程	说明
sp_databases	不支持
sp_datatype_info	不支持
sp_fkeys *pktable_name [, pktable-owner][, pktable-qualifier] [, fktable-name] [, fktable_owner] [, fktable-qualifier]*	返回指定表的外键信息
sp_pkeys *table-name [, table_owner] [, table_qualifier]*	返回指定表的主键信息
sp_server_info	不支持
sp_special_columns *table_name [, table-owner] [, table-qualifier] [, col-type]*	返回唯一标识指定表中的行的优化列集
sp_sproc_columns *proc-name [, proc_owner] [, proc-qualifier] [, column-name]*	返回存储过程的输入和返回参数信息
sp_stored_procedures [*sp-name] [, sp-owner] [, sp-qualifier]*	返回有关一个或多个存储过程的信息
sp_statistics [*table_name] [, table_owner] [, table_qualities] [, index_name] [, is_unique]*	返回有关表及其索引的信息
sp_tables *table-name [, table-owner] [, table-qualifier] [, table-type]*	返回可以显示在指定表的 FROM 子句中的对象列表

索引

字母

% 运算符
  模函数 183
% 注释指示符 53
&
  位运算符 15
/* 注释指示符 53
// 注释指示符 53
>> 运算符
  Java 方法和字段 22
@@char_convert 全局变量
  支持的全局变量的列表 48
@@client_csid 全局变量
  支持的全局变量的列表 48
@@client_csname 全局变量
  支持的全局变量的列表 48
@@connections 全局变量
  支持的全局变量的列表 48
@@cpu_busy 全局变量
  支持的全局变量的列表 48
@@dbts 全局变量
  全局变量列表 46
@@error 全局变量
  全局变量列表 46
  支持的全局变量的列表 48
@@fetch_status 全局变量
  全局变量列表 46
@@identity 全局变量
  触发器 52
  全局变量列表 46
  说明 51
  支持的全局变量的列表 48
@@idle 全局变量
  支持的全局变量的列表 48
@@io_busy 全局变量
  支持的全局变量的列表 48
@@isolation 全局变量
  全局变量列表 46
  支持的全局变量的列表 48
@@langid 全局变量
  支持的全局变量的列表 48
@@language 全局变量
  支持的全局变量的列表 48
@@max_connections 全局变量
  支持的全局变量的列表 48
@@maxcharlen 全局变量
  支持的全局变量的列表 48
@@ncharsize 全局变量
  支持的全局变量的列表 48
@@nestlevel 全局变量
  支持的全局变量的列表 48
@@pack_received 全局变量
  支持的全局变量的列表 48
@@pack_sent 全局变量
  支持的全局变量的列表 48
@@packet_errors 全局变量
  支持的全局变量的列表 48
@@procid 全局变量
  全局变量列表 46
  支持的全局变量的列表 48
@@rowcount 全局变量
  全局变量列表 46
  支持的全局变量的列表 48
@@servername 全局变量
  全局变量列表 46
  支持的全局变量的列表 48

@@spid 全局变量
支持的全局变量的列表 48
@@sqlstatus 全局变量
全局变量列表 46
支持的全局变量的列表 48
@@textsize 全局变量
支持的全局变量的列表 48
@@thresh_hysteresis 全局变量
支持的全局变量的列表 48
@@timeticks 全局变量
支持的全局变量的列表 48
@@total_errors 全局变量
支持的全局变量的列表 48
@@total_read 全局变量
支持的全局变量的列表 48
@@total_write 全局变量
支持的全局变量的列表 48
@@tranchained 全局变量
支持的全局变量的列表 48
@@trancount 全局变量
支持的全局变量的列表 48
@@transtate 全局变量
支持的全局变量的列表 48
@@version 全局变量
全局变量列表 46
支持的全局变量的列表 48
[ESQL]
语句指示符 238
[Interactive SQL]
语句指示符 238
[SP]
语句指示符 238
[T-SQL]
语句指示符 238
^
位运算符 15
|
位运算符 15
~
位运算符 15
2 月 29 日
关于 102
ABS 函数
SQL 语法 119
ACOS 函数
SQL 语法 119
ActiveSync
MobiLink 客户端使用 388
Adaptive Server Enterprise
CREATE DATABASE 语句 316
转换存储过程 229
ADDRESS 子句
CREATE SYNCHRONIZATION USER 380
AES 加密算法
CREATE DATABASE 语句 316
ALL
SELECT 语句中的关键字 587
条件和 SQL 语法 27
ALLOCATE DESCRIPTOR 语句
SQL 语法 240
ALTER DATABASE 语句
SQL 语法 242
ALTER DBSPACE 语句
SQL 语法 246
ALTER EVENT 语句
SQL 语法 248
ALTER FUNCTION 语句
SQL 语法 250
ALTER PROCEDURE 语句
SQL 语法 252
ALTER PUBLICATION 语句
SQL 语法 254
ALTER REMOTE MESSAGE TYPE 语句
SQL 语法 256
ALTER SERVER 语句
SQL 语法 258
ALTER SYNCHRONIZATION DEFINITION 语句

SQL 语法 260
ALTER SYNCHRONIZATION SITE 语句
SQL 语法 263
ALTER SYNCHRONIZATION SUBSCRIPTION 语句
SQL 语法 265
ALTER SYNCHRONIZATION TEMPLATE 语句
SQL 语法 267
ALTER SYNCHRONIZATION USER 语句
SQL 语法 270
ALTER TABLE 语句
SQL 语法 272
ALTER TRIGGER 语句
SQL 语法 279
ALTER VIEW 语句
SQL 语法 281
ALTER WRITEFILE 语句
SQL 语法 283
AND
逻辑运算符 13
三值逻辑 33
ANSI
等价于使用 REWRITE 函数 202
ANSI_PERMISSIONS
Adaptive Server Enterprise 兼容性 594
ANSINULL 选项
Adaptive Server Enterprise 兼容性 594
ANY
条件和 SQL 语法 27
ARGN 函数
SQL 语法 120
asademo.db 文件
关于 xviii
ASCII
函数和 SQL 语法 121
ASIN 函数
SQL 语法 121
AT 子句
创建现有表 333
ATAN 函数
SQL 语法 122
ATAN2 函数
SQL 语法 122
ATN2 函数
SQL 语法 122
AUTO_COMMIT 选项
Interactive SQL 选项 605
AUTOINCREMENT
@@identity 51
GET_IDENTITY 函数 159
关于 397
AVG 函数
SQL 语法 123
BACKUP 语句
SQL 语法 285
BEGIN DECLARE 语句
SQL 语法 424
BEGIN TRANSACTION 语句
SQL 语法 291
BEGIN 关键字
兼容性 289
BEGIN 语句
SQL 语法 288
BETWEEN 条件
SQL 语法 28
BIGINT 数据类型
关于 65
BIT 数据类型 74
BLOB
GET DATA SQL 语句 489
SET 语句示例 593
插入 592, 811
导出 813
导入 ASE 生成的 BCP 文件 532
二进制数据类型 83
事务日志注意事项 246
BREAK 语句

Transact-SQL 语法 659
buffer_size 流参数
使用 HTTP 的 MobiLink 客户端 383
使用 HTTPS 的 MobiLink 客户端 386
BYE 语句
SQL 语法 471
BYTE_LENGTH 函数
SQL 语法 124
BYTE_SUBSTR 函数
SQL 语法 125
CALL 语句
SQL 语法 294
在 Transact-SQL 中 469
CASCADE 操作
CREATE TABLE 语句 403
CASE 表达式
NULLIF 函数 189
SQL 语法 20
CASE 语句
SQL 语法 296
CAST 函数
SQL 语法 125
数据类型转换 94
CEILING 函数
SQL 语法 126
certificate_company 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 HTTPS 的 MobiLink 客户端 388
使用 TCP/IP 的 MobiLink 客户端 382
certificate_name 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 HTTPS 的 MobiLink 客户端 388
使用 TCP/IP 的 MobiLink 客户端 382
certificate_unit 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 HTTPS 的 MobiLink 客户端 388
使用 TCP/IP 的 MobiLink 客户端 382
CHAR 函数
SQL 语法 127
CHAR 数据类型
关于 61
CHAR_LENGTH 函数
SQL 语法 128
CHARACTER VARYING 数据类型
关于 62
CHARACTER 数据类型
关于 61
CHARINDEX 函数
SQL 语法 127
CHECK CONSTRAINTS 选项
LOAD TABLE 语句 531
CHECK ON COMMIT 子句
参照完整性 403
CHECK 条件
关于 401
CHECK 子句
搜索条件 26
CHECKPOINT 语句
SQL 语法 299
CLEAR 语句
SQL 语法 300
client_port 号
使用 HTTPS 的 MobiLink 客户端的缺省值 386
client_port 流参数
使用 HTTP 的 MobiLink 客户端 383
使用 TCP/IP 的 MobiLink 客户端 381
CLOSE ON ENDTRANS 选项
Adaptive Server Enterprise 兼容性 594
CLOSE 语句
SQL 语法 301
COALESCE 函数
SQL 语法 129
col_length Adaptive Server Enterprise 函数 115
col_name Adaptive Server Enterprise 函数 115
COMMENT 语句
SQL 语法 303
COMMIT 语句

SQL 语法 305
参照完整性 755
COMPARE 函数
SQL 语法 129
CONFIGURE 语句
SQL 语法 307
ConflictRetries 同步选项
MobiLink 同步客户端 389
CONNECT 权限
授予权限 497
CONNECT 语句
SQL 语法 308
CONNECTION_PROPERTY 函数
SQL 语法 131
CONSOLIDATE 权限
撤消 578
授予 502
CONTINUE 语句
Transact-SQL 语法 659
CONVERT 函数
SQL 语法 132
数据类型转换 94
COS 函数
SQL 语法 134
COT 函数
SQL 语法 135
COUNT 函数
SQL 语法 135
CREATE COMPRESSED DATABASE 语句
SQL 语法 312
CREATE DATABASE 语句
SQL 语法 314
CREATE DATATYPE 语句
SQL 语法 322
CREATE DBSPACE 语句
SQL 语法 319
CREATE DECRYPTED FILE 语句
SQL 语法 321
CREATE DOMAIN 语句
SQL 语法 322
使用 86
CREATE ENCRYPTED FILE 语句
SQL 语法 324
CREATE EVENT 语句
SQL 语法 326
CREATE EXISTING TABLE 语句
SQL 语法 332
代理表 797, 800
CREATE EXPANDED DATABASE 语句
SQL 语法 312
CREATE EXTERNLOGIN 语句
SQL 语法 335
CREATE FUNCTION 语句
SQL 语法 337
CREATE INDEX 语句
SQL 语法 341
表使用 342
CREATE MESSAGE 语句
Transact-SQL 语法 345
CREATE PROCEDURE 语句
SQL 语法 347
Transact-SQL 语法 355
CREATE PUBLICATION 语句
SQL 语法 357
CREATE REMOTE MESSAGE TYPE 语句
SQL 语法 360
CREATE SCHEMA 语句
SQL 语法 362
CREATE SERVER 语句
SQL 语法 364
CREATE STATISTICS 语句
SQL 语法 366
CREATE SUBSCRIPTION 语句
SQL 语法 367
CREATE SYNCHRONIZATION DEFINITION 语句
SQL 语法 369
CREATE SYNCHRONIZATION SITE 语句

SQL 语法 372
CREATE SYNCHRONIZATION SUBSCRIPTION 语句
SQL 语法 375
CREATE SYNCHRONIZATION TEMPLATE 语句
SQL 语法 377
CREATE SYNCHRONIZATION USER 语句
SQL 语法 380
CREATE TABLE 语句
Java 类 92
SQL 语法 395
Transact-SQL 404
远程表 396
CREATE TRIGGER 语句
SQL 语法 408, 412
Transact-SQL 语法 415
CREATE VARIABLE 语句
SQL 语法 417
CREATE VIEW 语句
SQL 语法 418
CREATE WRITEFILE 语句
SQL 语法 420
CROSS JOIN
SQL 语法 484
CSCONVERT 函数
SQL 语法 136
CURRENT DATABASE
特殊值 37
CURRENT DATE
特殊值 37
CURRENT PUBLISHER 504
特殊值 37
CURRENT TIME
特殊值 38
CURRENT TIMESTAMP
特殊值 38
CURRENT USER
特殊值 38
CURRENT UTC TIMESTAMP
特殊值 39
curunreservedpgs Adaptive Server Enterprise 函数 115
data_pgs Adaptive Server Enterprise 函数 115
datalength Adaptive Server Enterprise 函数 115
DATALENGTH 函数
SQL 语法 138
DATE 函数
SQL 语法 139
DATE 数据类型
关于 79
DATE_ORDER 选项
ODBC 78
使用 78
DATEADD 函数
SQL 语法 139
DATEDIFF 函数
SQL 语法 140
DATEFIRST 选项
Adaptive Server Enterprise 兼容性 594
DATEFORMAT 函数
SQL 语法 142
DATENAME 函数
SQL 语法 143
DATEPART 函数
SQL 语法 143
DATETIME 函数
SQL 语法 144
DATETIME 数据类型
关于 80
DAY 函数
SQL 语法 144
DAYNAME 函数
SQL 语法 145
DAYS 函数
SQL 语法 145
db_id Adaptive Server Enterprise 函数 115
DB_ID 函数

SQL 语法 147
db_name Adaptive Server Enterprise 函数 115
DB_NAME 函数
SQL 语法 147
DB_PROPERTY 函数
SQL 语法 148
DBA 权限
授予权限 497
在系统表中 736
DBFreePercent 事件条件
关于 153
DBFreeSpace 事件条件
关于 153
DBSize 事件条件
关于 153
DBSPACE
SYSFILE 系统表 685
dbspace
变更 246
创建 319
确定可用空间 763
删除 445
DEALLOCATE DESCRIPTOR 语句
SQL 语法 423
DEALLOCATE 语句
SQL 语法 422
DECIMAL 数据类型
关于 65
DECLARE CURSOR 语句
SQL 语法 426
Transact-SQL 语法 431
DECLARE LOCAL TEMPORARY TABLE 语句
SQL 语法 433
DECLARE 语句
SQL 语法 288, 425
Transact-SQL 兼容性 289
复合语句 289
DEFAULT TIMESTAMP 列 399
DEFAULTS 选项
LOAD TABLE 语句 531
DEGREES 函数
SQL 语法 148
DELETE 语句
SQL 语法 435
DELETE （定位）语句
SQL 语法 438
DELETING 条件
触发器 33
DELIMITED BY 选项
LOAD TABLE 语句 531
DESCRIBE 语句
SQL 语法 440
长列名 441
DIFFERENCE 函数
SQL 语法 149
DisablePolling 同步选项
MobiLink 同步客户端 389
DISCONNECT 语句
SQL 语法 444
DISTINCT 关键字 587
DOUBLE 数据类型
关于 66
DOW 函数
SQL 语法 150
DownloadBufferSize 同步选项
MobiLink 同步客户端 389
DROP CONNECTION 语句
SQL 语法 448
DROP DATABASE 语句
SQL 语法 447
DROP DATATYPE 语句
SQL 语法 445
DROP DBSPACE 语句
SQL 语法 445
DROP DOMAIN 语句
SQL 语法 445
DROP EVENT 语句

SQL 语法 445
DROP EXTERNLOGIN 语句
SQL 语法 449
DROP FUNCTION 语句
SQL 语法 445
DROP INDEX 语句
SQL 语法 445
DROP MESSAGE 语句
SQL 语法 445
DROP OPTIMIZER STATISTICS 语句
SQL 语法 454
DROP PROCEDURE 语句
SQL 语法 445
DROP PUBLICATION 语句
SQL 语法 450
DROP REMOTE MESSAGE TYPE 语句
SQL 语法 451
DROP SERVER 语句
SQL 语法 452
DROP STATEMENT 语句
SQL 语法 453
DROP STATISTICS 语句
SQL 语法 454
DROP SUBSCRIPTION 语句
SQL 语法 456
DROP SYNCHRONIZATION DEFINITION 语句
SQL 语法 457
DROP SYNCHRONIZATION SITE 语句
SQL 语法 459
DROP SYNCHRONIZATION SUBSCRIPTION 语句
SQL 语法 460
DROP SYNCHRONIZATION TEMPLATE 语句
SQL 语法 461
DROP SYNCHRONIZATION USER 语句
SQL 语法 463
DROP TABLE 语句
SQL 语法 445
DROP TRIGGER 语句
SQL 语法 445
DROP VARIABLE 语句
SQL 语法 464
DROP VIEW 语句
SQL 语法 445
DROP 语句
SQL 语法 445
DUMMY 系统表
关于 664
DYNAMIC SCROLL 游标
声明 426
ELSE
CASE 表达式 20
IF 表达式 19
END
CASE 表达式 20
END DECLARE 语句
SQL 语法 424
END LOOP 语句
SQL 语法 539
END 关键字
兼容性 289
END 语句
SQL 语法 288
ENDIF
IF 表达式 19
ErrorLogSendLimit 同步选项
MobiLink 同步客户端 389
ERRORMSG 函数
SQL 语法 150
ErrorNumber 事件条件
关于 153
ESCAPE CHARACTER 选项
LOAD TABLE 语句 532
ESCAPES 选项
LOAD TABLE 语句 532
ESQL

语句指示符 238
ESTIMATE 函数
SQL 语法 151
ESTIMATE_SOURCE 函数
SQL 语法 152
EVENT_CONDITION 函数
SQL 语法 153
EVENT_CONDITION_NAME 函数
SQL 语法 155
EVENT_PARAMETER 函数
SQL 语法 155
EXCEPTION 语句
SQL 语法 288
EXECUTE IMMEDIATE 语句
SQL 语法 467
EXECUTE 权限
授予 497
EXECUTE 语句
SQL 语法 465
Transact-SQL 语法 469
EXISTS 条件
SQL 语法 32
EXIT 语句
SQL 语法 471
EXP 函数
SQL 语法 156
EXPERIENCE_ESTIMATE 函数
SQL 语法 157
EXPLAIN 语句
SQL 语法 473
EXPLANATION 函数
SQL 语法 157
FALSE 条件
IS FALSE 条件 32
三值逻辑 33
FETCH 语句
SQL 语法 475
FILE 消息
关于 360, 451
FILE 消息类型
关于 256
FireTriggers 同步选项
MobiLink 同步客户端 389
FIRST 子句
SELECT 语句 587
FLOAT 数据类型
关于 67
FLOOR 函数
SQL 语法 159
FOR 语句
SQL 语法 480
FOR 子句
SELECT 语句 589
FORMAT 选项
LOAD TABLE 语句 532
FORWARD TO 语句
SQL 语法 482
FROM 子句
SELECT 语句 588
SQL 语法 484
FTP 消息
关于 360
FTP 消息类型
关于 256
GET DATA 语句
SQL 语法 489
GET DESCRIPTOR 语句
SQL 语法 492
GET OPTION 语句
SQL 语法 494
GET_IDENTITY 函数
SQL 语法 159
GETDATE 函数
SQL 语法 161
GLOBAL_DATABASE_ID 选项
CREATE TABLE 语句 398
GOTO 语句
Transact-SQL 语法 496

GRANT CONNECT 语句
SQL 语法 497
GRANT CONSOLIDATE 语句
SQL 语法 502
GRANT DBA 语句
SQL 语法 497
GRANT GROUP 语句
SQL 语法 497
GRANT PUBLISH 语句
SQL 语法 504
GRANT REMOTE DBA 语句
SQL 语法 508
GRANT REMOTE 语句
SQL 语法 505
GRANT RESOURCE 语句
SQL 语法 497
GRANT 语句
SQL 语法 497
查看权限 673
GRAPHICAL_PLAN 函数
SQL 语法 161
GRAPHICAL_ULPLAN 函数
SQL 语法 163
GREATER 函数
SQL 语法 164
GROUP BY 子句
SELECT 语句 588
GROUP 权限
授予权限 497
GUID
NEWID 函数 SQL 语法 185
STRTOUUID 函数 SQL 语法 217
UNIQUEIDENTIFIER 数据类型 84
UUIDTOSTR 函数 SQL 语法 227
HAVING 子句
SELECT 语句 589
搜索条件 26
HELP 语句
SQL 语法 509
HEXTOINT 函数
SQL 语法 164
host 流参数
使用 HTTP 的 MobiLink 客户端 383
使用 HTTPS 的 MobiLink 客户端 386
使用 TCP/IP 的 MobiLink 客户端 381
host_id Adaptive Server Enterprise 函数 115
host_name Adaptive Server Enterprise 函数 115
hostvar
语法 236
HOUR 函数
SQL 语法 165
HOURS 函数
SQL 语法 165
HTTP
MobiLink 客户端使用 383
HTTPS
MobiLink 客户端使用 386
I/O
重新校准 I/O 成本模型 244
IDENTITY 函数
SQL 语法 166
IDENTITY 列
@@identity 51
IdleTime 事件条件
关于 153
IF UPDATE 子句
在 Transact-SQL 的触发器中 415
在触发器中 408, 412
IF 表达式
SQL 语法 19
搜索条件 26
IF 语句
SQL 语法 510
Transact-SQL 语法 512
IFNULL 函数
SQL 语法 167
IgnoreHookErrors 同步选项
MobiLink 同步客户端 389

IgnoreScheduling 同步选项
MobiLink 同步客户端 389
IMAGE 数据类型
关于 84
IN 条件
SQL 语法 31
INCLUDE 语句
SQL 语法 514
Increment 同步选项
关于 389
index_col Adaptive Server Enterprise 函数 115
INDEX_ESTIMATE 函数
SQL 语法 168
INNER JOIN
SQL 语法 484
INPUT 语句
SQL 语法 515
INSERT 语句
SQL 语法 520
INSERTING 条件
触发器 33
INSERTSTR 函数
SQL 语法 168
INSTALL 语句
SQL 语法 524
INT 数据类型
关于 68
INTEGER 数据类型
关于 68
INTEGRATED LOGIN 权限
授予 497
Interactive SQL
BYE 语句语法 471
CLEAR 语句语法 300
CONFIGURE 语句语法 307
CONNECT 语句语法 308
DISCONNECT 语句语法 444
EXIT 语句语法 471
HELP 语句语法 509
INPUT 语句语法 515
OUTPUT 语句语法 546
PARAMETERS 语句语法 551
QUIT 语句语法 471
READ 语句语法 562
SET CONNECTION 语句语法 598
SET OPTION 语句语法 605
START ENGINE 语句语法 616
START LOGGING 语句语法 618
STOP LOGGING 语句语法 627
SYSTEM 语句语法 632
按字母顺序排序的语句列表 235
返回代码 471
过程分析 790
连接到数据库 309
Interval 事件条件
关于 153
INTO 子句
SELECT 语句 588
INTTOHEX 函数
SQL 语法 169
IS
逻辑运算符 13
三值逻辑 33
IS FALSE 条件
SQL 语法 32
IS NOT NULL 条件
SQL 语法 32
IS NULL 条件
SQL 语法 32
IS TRUE 条件
SQL 语法 32
IS UNKNOWN 条件
SQL 语法 32
ISDATE 函数
SQL 语法 170
ISNULL 函数
SQL 语法 170

ISNUMERIC 函数
SQL 语法 171
jar 文件
安装 524
删除 567
Java
SQL 表达式 22
不受支持的类 90
方法签名 351
受支持的类 88
系统表 741
用户定义的函数 109
转换 Java 和 SQL 97
Java VM
启动 617
停止 626
Java 表达式
SQL 语法 22
Java 到 SQL 的数据类型转换 97
Java 开发工具包
准备类 91
Java 类
CREATE DATABASE 语句 317
安装 91
列 92
内置 88
数据类型 88
疑难解答 769
用户定义的 88, 91
装载在数据库中 769
Java 类创建向导
使用 91
Java 数据类型
大小写敏感性 92
转换到 SQL 97, 98
Java 字段
引用 22
java.applet 包
不受支持的类 90
java.awt 包
不受支持的类 90
java.awt.datatransfer 包
不受支持的类 90
java.awt.event 包
不受支持的类 90
java.awt.image 包
不受支持的类 90
java.beans 包
受支持的类 88
java.io 包
受支持的类 88
java.io.File
部分受支持的类 90
java.io.FileDescriptor
部分受支持的类 90
java.io.FileInputStream
部分受支持的类 90
java.io.FileOutputStream
部分受支持的类 90
java.io.RandomAccessFile
部分受支持的类 90
java.lang 包
受支持的类 89
java.lang.ClassLoader
部分受支持的类 90
java.lang.Compiler
部分受支持的类 90
java.lang.reflect 包
受支持的类 89
java.lang.Runtime
部分受支持的类 90
java.lang.Thread
部分受支持的类 89
java.math 包
受支持的类 89
java.net 包
受支持的类 89

java.net.PlainDatagramSocketImpl
受支持的类 89
java.rmi 包
受支持的类 89
java.rmi.dgc 包
受支持的类 89
java.rmi.registry 包
受支持的类 89
java.rmi.server 包
受支持的类 89
java.security 包
受支持的类 89
java.security.acl 包
受支持的类 89
java.security.interfaces 包
受支持的类 89
java.SQL 包
受支持的类 89
java.text 包
受支持的类 89
java.util 包
受支持的类 89
java.util.zip 包
受支持的类 89
java.util.zip.Deflater
部分受支持的类 90
java.util.zip.Inflater
部分受支持的类 90
Java/SQL 数据类型转换 97
javax
数据库中的 Java 88
jConnect
CREATE DATABASE 语句 317
JDBC
升级数据库 242
数据类型转换 97
JDK
准备类 91
keep_alive 流参数
使用 HTTP 的 MobiLink 客户端 384
使用 HTTPS 的 MobiLink 客户端 387
使用 TCP/IP 的 MobiLink 客户端 381
KEY JOIN
SQL 语法 484
LAST USER
特殊值 39
LCASE 函数
SQL 语法 172
lct_admin Adaptive Server Enterprise 函数 115
LEAVE 语句
SQL 语法 527
LEFT OUTER JOIN
SQL 语法 484
LEFT 函数
SQL 语法 172
LENGTH 函数
SQL 语法 173
LESSER 函数
SQL 语法 174
LIKE 条件
SQL 语法 28
和大小写敏感性 29
和归类 29
最大模式长度 29
LIST 函数
SQL 语法 174
LOAD STATISTICS 语句
SQL 语法 529
LOAD TABLE 语句
SQL 语法 530
LOCATE 函数
SQL 语法 175
LOCK TABLE 语句
SQL 语法 537
LockTables 同步选项
MobiLink 同步客户端 389
LOG 函数
SQL 语法 176

LOG10 函数
SQL 语法 177
LogFreePercent 事件条件
关于 153
LogFreeSpace 事件条件
关于 153
LogSize 事件条件
关于 153
LONG BINARY 数据类型
关于 83
LONG VARCHAR 数据类型
关于 62
LONG_ULPLAN 函数
SQL 语法 177
LOOP 语句
SQL 语法 539
LOWER 函数
SQL 语法 178
LTRIM 函数
SQL 语法 179
MAPI
返回代码 806
扩展存储过程 803
MAPI 消息
关于 360, 451
MAPI 消息类型
关于 256
MAX 函数
SQL 语法 179
MDSR 加密算法
CREATE DATABASE 语句 316
MEMBERSHIP 子句
授予权限 497
Memory 同步选项
MobiLink 同步客户端 389
MESSAGE 语句
SQL 语法 541
MIN 函数
SQL 语法 180
MINUTE 函数
SQL 语法 181
MINUTES 函数
SQL 语法 181
MobiLink
创建发布 357
MobiLinkPwd 同步选项
MobiLink 同步客户端 389
MOD 函数
SQL 语法 183
MONEY 数据类型
关于 72
MONTH 函数
SQL 语法 183
MONTHNAME 函数
SQL 语法 184
MONTHS 函数
SQL 语法 184
NATURAL JOIN
SQL 语法 484
network_connect_timeout 流参数
使用 HTTP 的 MobiLink 客户端 384
使用 HTTPS 的 MobiLink 客户端 387
使用 TCP/IP 的 MobiLink 客户端 382
network_leave_open 流参数
使用 HTTP 的 MobiLink 客户端 384
使用 HTTPS 的 MobiLink 客户端 387
使用 TCP/IP 的 MobiLink 客户端 382
network_name 流参数
使用 HTTP 的 MobiLink 客户端 384
使用 HTTPS 的 MobiLink 客户端 387
使用 TCP/IP 的 MobiLink 客户端 382
NEWID 函数
SQL 语法 185
NewMobiLinkPwd 同步选项
MobiLink 同步客户端 389
NEXT_CONNECTION 函数
SQL 语法 187
NEXT_DATABASE 函数

SQL 语法 188
NO SCROLL 游标
声明 426
NOT
逻辑运算符 13
三值逻辑 33
NOW 函数
SQL 语法 189
NULL
Adaptive Server Enterprise 兼容性 56
ISNULL 函数 170
NULL 值 55
三值逻辑 33, 55
NULLIF 函数 21
关于 189
NUMBER 函数
SQL 语法 190
更新 644, 649
object_id Adaptive Server Enterprise 函数 115
object_name Adaptive Server Enterprise 函数 115
ODBC
声明静态游标 426
OfflineDirectory 同步选项
MobiLink 同步客户端 389
ON EXCEPTION RESUME 子句
关于 349
ON 短语
条件 26
ON_TSQL_ERROR 选项
与 ON EXCEPTION RESUME 子句 349
OPEN 语句
SQL 语法 543
OPTION 子句
CREATE SYNCHRONIZATION USER 389
OR
逻辑运算符 13
三值逻辑 33
ORDER BY 子句 589
OUTPUT 语句
Java 数据 548
OUTPUT 语句 SQL 语 546
PARAMETERS 语句
SQL 语法 551
PASSTHROUGH 语句
SQL 语法 552
PATINDEX 函数
SQL 语法 191
PCTFREE 设置
系统表 667
persistent 流参数
使用 HTTP 的 MobiLink 客户端 384
使用 HTTPS 的 MobiLink 客户端 387
PI 函数
SQL 语法 192
PLAN 函数
SQL 语法 193
PollingPeriod 同步选项
MobiLink 同步客户端 389
port 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 HTTPS 的 MobiLink 客户端 388
使用 TCP/IP 的 MobiLink 客户端 381
POWER 函数
SQL 语法 194
PREPARE TO COMMIT 语句
SQL 语法 556
PREPARE 语句
SQL 语法 553
PRINT 语句
Transact-SQL 语法 557
proc_role Adaptive Server Enterprise 函数 115
PROPERTY 函数
SQL 语法 195
PROPERTY_DESCRIPTION 函数
SQL 语法 195
PROPERTY_NAME 函数
SQL 语法 196
PROPERTY_NUMBER 函数

SQL 语法 197
proxy_host 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 HTTPS 的 MobiLink 客户端 388
proxy_port 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 HTTPS 的 MobiLink 客户端 388
PUBLIC 组
在系统表中 737
PURGE 子句
FETCH 语句 477
PUT 语句
SQL 语法 558
QUARTER 函数
SQL 语法 197
QUIT 语句
SQL 语法 471
QUOTED_IDENTIFIER 选项
Adaptive Server Enterprise 兼容性 594
T-SQL 表达式兼容性 24
QUOTES 选项
LOAD TABLE 语句 532
RADIANS 函数
SQL 语法 198
RAISERROR 语句
Transact-SQL 语法 560
RAND 函数
SQL 语法 198
READ 语句
SQL 语法 562
READTEXT 语句
Transact-SQL 语法 564
REAL 数据类型
关于 69
REFERENCES 权限
授予 497
RELEASE SAVEPOINT 语句
SQL 语法 565
REMAINDER 函数
SQL 语法 199
remember_last_statement
关于 790
REMOTE RESET 语句
SQL 语法 566
remoteoption 表
关于 714
remoteoptiontype 表
关于 715
REMOVE 语句
SQL 语法 567
REORGANIZE TABLE 语句
SQL 语法 568
REPEAT 函数
SQL 语法 199
REPLACE 函数
SQL 语法 200
REPLICATE 函数
SQL 语法 201
request_level_debugging
关于 790
request_level_logging
关于 790
reserved_pgs Adaptive Server Enterprise 函数 115
RESIGNAL 语句
SQL 语法 570
RESOURCE 权限
授予权限 497
RESTORE DATABASE 语句
SQL 语法 571
RESTRICT 操作
CREATE TABLE 语句 403
RESUME 语句
SQL 语法 573
RETURN 语句
SQL 语法 574
REVOKE CONSOLIDATE 语句
SQL 语法 578

REVOKE PUBLISH 语句
SQL 语法 579
REVOKE REMOTE DBA 语句
SQL 语法 582
REVOKE REMOTE 语句
SQL 语法 581
REVOKE 语句
SQL 语法 576
REWRITE 函数
SQL 语法 202
RIGHT OUTER JOIN
SQL 语法 484
RIGHT 函数
SQL 语法 204
Rijndael 加密算法
CREATE DATABASE 语句 316
ROLLBACK TO SAVEPOINT 语句
SQL 语法 584
ROLLBACK TRIGGER 语句
SQL 语法 585
ROLLBACK 语句
SQL 语法 583
ROUND 函数
SQL 语法 204
rowcnt Adaptive Server Enterprise 函数 115
ROWCOUNT 选项
Adaptive Server Enterprise 兼容性 594
RTRIM 函数
SQL 语法 205
sa_audit_string 系统过程
语法 755
sa_check_commit 系统过程
语法 755
sa_conn_activity 系统过程
语法 756
sa_conn_compression_info 系统过程
语法 757
sa_conn_info 系统过程
语法 758
sa_conn_properties 系统过程
语法 759, 760
sa_conn_properties_by_name 系统过程
语法 761
sa_db_info 系统过程
语法 761
sa_db_properties 系统过程
语法 762
sa_disk_free_space 系统过程
语法 763
sa_eng_properties 系统过程
语法 763
sa_flush_cache 系统过程
语法 764
sa_flush_statistics 系统过程
语法 764
sa_get_dtt 系统过程
语法 765
sa_get_histogram 系统过程
语法 765
sa_get_request_profile 系统过程
语法 766
sa_get_request_times 系统过程
语法 766
sa_get_server_messages 系统过程
语法 767
sa_index_density 系统过程
语法 768
sa_index_levels 系统过程
语法 768
sa_java_loaded_classes 系统过程
语法 769
sa_locks 系统过程
语法 769
sa_make_object 系统过程
语法 771
sa_migrate 系统过程
语法 773
sa_migrate_create_fks 系统过程

语法 776
sa_migrate_create_remote_fks_list 系统过程
语法 777
sa_migrate_create_remote_table_list 系统过程
语法 779
sa_migrate_create_tables 系统过程
语法 781
sa_migrate_data 系统过程
语法 782
sa_migrate_drop_proxy_tables 系统过程
语法 784
sa_procedure_profile 系统过程
语法 785
sa_procedure_profile_summary 系统过程
语法 787
sa_reset_identity 系统过程
语法 789
sa_server_option 系统过程
语法 790
sa_statement_text 系统过程
语法 793
sa_table_fragmentation 系统过程
语法 793
sa_table_page_usage 系统过程
语法 794
sa_validate 系统过程
语法 795
SAVEPOINT 语句
SQL 语法 586
Schedule 同步选项
MobiLink 同步客户端 389
ScriptSiteName 同步选项
MobiLink 同步客户端 389
ScriptVersion 同步选项
MobiLink 同步客户端 389
SCROLL 游标
声明 426
SECOND 函数
SQL 语法 205
SECONDS 函数
SQL 语法 206
security 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 TCP/IP 的 MobiLink 客户端 382
SELECT 权限
授予 497
SELECT 语句
SQL 语法 587
SELF_RECURSION 选项
Adaptive Server Enterprise 兼容性 594
SEND AT 子句 502, 505
发布 504
SEND EVERY 子句 502, 505
SendColumnNames 同步选项
MobiLink 同步客户端 389
SendTriggers 同步选项
MobiLink 同步客户端 389
SET CONNECTION 语句
SQL 语法 598
SET DEFAULT 操作
CREATE TABLE 语句 403
SET DESCRIPTOR 语句
SQL 语法 599
SET NULL 操作
CREATE TABLE 语句 403
SET OPTION 语句
Interactive SQL 语法 605
SQL 语法 601
Transact-SQL 语法 594
SET PERMANENT 语句
Interactive SQL 语法 605
SET REMOTE OPTION 语句
SQL 语法 606
SET SQLCA 语句
SQL 语法 608
SET TEMPORARY OPTION 语句
Interactive SQL 语法 605
SQL 语法 601

SET 语句
SQL 语法 592
Transact-SQL 语法 594
SETUSER 语句
SQL 语法 610
SHORT_ULPLAN 函数
SQL 语法 207
show_role Adaptive Server Enterprise 函数 115
SIGN 函数
SQL 语法 208
SIGNAL 语句
SQL 语法 612
SIMILAR 函数
SQL 语法 209
SIN 函数
SQL 语法 209
SMALLDATETIME 数据类型
关于 81
SMALLINT 数据类型
关于 70
SMTP
返回代码 806
扩展存储过程 803
SOME 条件
SQL 语法 27
SORTKEY 函数
SQL 语法 210
SOUNDEX 函数
SQL 语法 213
SP
语句指示符 238
sp_addgroup 系统过程 814
sp_addlogin 系统过程 814
sp_addmessage 系统过程 345, 814
sp_addtype 系统过程 814
sp_adduser 系统过程 814
sp_changegroup 系统过程 814
sp_column_privileges
分类过程 815
sp_column_privileges 分类过程 815
sp_columns 分类过程 815
sp_dboption 系统过程 814
sp_dropgroup 系统过程 814
sp_droplogin 系统过程 814
sp_dropmessage 系统过程 814
sp_droptype 系统过程 814
sp_dropuser 系统过程 814
sp_fkeys 分类过程 815
sp_getmessage 系统过程 814
sp_helptext 系统过程 814
sp_login_environment 系统过程
语法 796
sp_password 系统过程 814
sp_pkeys 分类过程 815
sp_remote_columns 系统过程
语法 796
sp_remote_exported_keys 系统过程
语法 797
sp_remote_imported_keys 系统过程
语法 798
sp_remote_tables 系统过程
语法 799
sp_servercaps 系统过程
语法 800
sp_special_columns 分类过程 815
sp_sproc_columns 分类过程 815
sp_stored_procedures 分类过程 815
sp_tables 分类过程 815
sp_tsql_environment 系统过程
语法 801
SPACE 函数
SQL 语法 214
SQL
按字母顺序排序的语句列表 235
SQL Anywhere Studio
文档 xiii
SQL Remote
创建发布 357
创建预订 367
设置远程选项 606

系统表 665, 666
项目 665, 666
SQL Remote 系统表
remoteoption 714
SQL 变量
创建 417
删除 464
设置值 592
声明 425
SQL 到 Java 的数据类型转换 98
SQL 函数
ABS 函数语法 119
ACOS 函数语法 119
ARGN 函数语法 120
ASCII 函数语法 121
ASIN 函数语法 121
ATAN 函数语法 122
ATAN2 函数语法 122
ATN2 函数语法 122
AVG 函数语法 123
BYTE_LENGTH 函数语法 124
BYTE_SUBSTR 函数语法 125
CAST 函数语法 125
CEILING 函数语法 126
CHAR 函数语法 127
CHAR_LENGTH 函数语法 128
CHARINDEX 函数语法 127
COALESCE 函数语法 129
COMPARE 函数语法 129
CONNECTION_PROPERTY 函数语法 131
CONVERT 函数语法 132
COS 函数语法 134
COT 函数语法 135
COUNT 函数语法 135
CSCONVERT 函数语法 136
DATALENGTH 函数语法 138
DATE 函数语法 139
DATEADD 函数语法 139
DATEDIFF 语法 140
DATEFORMAT 函数语法 142
DATENAME 函数语法 143
DATEPART 函数语法 143
DATETIME 函数语法 144
DAY 函数语法 144
DAYNAME 函数语法 145
DAYS 函数语法 145
DB_ID 函数语法 147
DB_NAME 函数语法 147
DB_PROPERTY 函数语法 148
DEGREES 函数语法 148
DIFFERENCE 函数语法 149
DOW 函数语法 150
ERRORMSG 函数语法 150
ESTIMATE 函数语法 151
ESTIMATE_SOURCE 函数语法 152
EVENT_CONDITION 函数语法 153
EVENT_CONDITION_NAME 函数语法 155
EVENT_PARAMETER 函数语法 155
EXP 函数语法 156
EXPERIENCE_ESTIMATE 函数语法 157
EXPLANATION 函数语法 157
FLOOR 函数语法 159
GET_IDENTITY 函数语法 159
GETDATE 函数语法 161
GRAPHICAL_PLAN 函数语法 161
GRAPHICAL_ULPLAN 函数语法 163
GREATER 函数语法 164
HEXTOINT 函数语法 164
HOUR 函数语法 165
HOURS 函数语法 165
IDENTITY 函数语法 166
IFNULL 函数语法 167
INDEX_ESTIMATE 函数语法 168

INSERTSTR 函数语法 168
INTTOHEX 函数语法 169
ISDATE 函数语法 170
ISNULL 函数语法 170
ISNUMERIC 函数语法 171
LCASE 函数语法 172
LEFT 函数语法 172
LENGTH 函数语法 173
LESSER 函数语法 174
LIST 函数语法 174
LOCATE 函数语法 175
LOG 函数语法 176
LOG10 函数语法 177
LONG_ULPLAN 函数语法 177
LOWER 函数语法 178
LTRIM 函数语法 179
MAX 函数语法 179
MIN 函数语法 180
MINUTE 函数语法 181
MINUTES 函数语法 181
MOD 函数语法 183
MONTH 函数语法 183
MONTHNAME 函数语法 184
MONTHS 函数语法 184
NEWID 函数语法 185
NEXT_CONNECTION 函数语法 187
NEXT_DATABASE 函数语法 188
NOW 函数语法 189
NULLIF 函数语法 189
NUMBER 函数语法 190
PATINDEX 函数语法 191
PI 函数语法 192
PLAN 函数语法 193
POWER 函数语法 194
PROPERTY 函数语法 195
PROPERTY_DESCRIPTION 函数语法 195
PROPERTY_NAME 函数语法 196
PROPERTY_NUMBER 函数语法 197
QUARTER 函数语法 197
RADIANS 函数语法 198
RAND 函数语法 198
REMAINDER 函数语法 199
REPEAT 函数语法 199
REPLACE 函数语法 200
REPLICATE 函数语法 201
REWRITE 函数语法 202
RIGHT 函数语法 204
ROUND 函数语法 204
RTRIM 函数语法 205
SECOND 函数语法 205
SECONDS 函数语法 206
SHORT_ULPLAN 函数语法 207
SIGN 函数语法 208
SIMILAR 函数语法 209
SIN 函数语法 209
SORTKEY 函数语法 210
SOUNDEX 函数语法 213
SPACE 函数语法 214
SQLDIALECT 函数语法 215
SQRT 函数语法 215
STR 函数语法 216
STRING 函数语法 217
STRTOUUID 函数语法 217
STUFF 函数语法 218
SUBSTR 函数语法 219
SUBSTRING 函数语法 219
SUM 函数语法 220
TAN 函数语法 221
TEXTPTR 函数语法 221
TODAY 函数语法 222
TRACEBACK 函数语法 222
TRANSACTSQL 函数语法 223

TRIM 函数语法 223
TRUNCATE 函数语法 224
TRUNCNUM 函数语法 225
UCASE 函数语法 226
UPPER 函数语法 226
UUIDTOSTR 函数语法 227
VAREXISTS 函数语法 228
WATCOMSQL 函数语法 229
WEEKS 函数语法 229
YEARS 函数语法 231
YMD 函数语法 232
函数的类型 106
集合 106
日期和时间 107
数据类型转换 106
数字 111
图像 118
文本 118
系统 115
用户定义的 109
杂类 110
字符串 113
SQL 描述符区
DESCRIBE 语句 440
INCLUDE 语句 514
使用游标插入行 558
SQL 语法
ALL 条件 27
ANY 条件 27
BETWEEN 条件 28
CASE 表达式 20
CURRENT DATABASE 特殊值 37
CURRENT DATE 特殊值 37
CURRENT PUBLISHER 特殊值 37
CURRENT TIME 特殊值 38
CURRENT TIMESTAMP 特殊值 38
CURRENT USER 特殊值 38
CURRENT UTC TIMESTAMP 特殊值 39
EXISTS 条件 32
IF 表达式 19
IN 条件 31
IS NULL 条件 32
IS TRUE 或 FALSE 条件 32
Java 表达式 22
LAST USER 特殊值 39
LIKE 条件 28
NULL 值 55
SOME 条件 27
SQLCODE 特殊值 40
SQLSTATE 特殊值 40
TIMESTAMP 特殊值 40
Transact-SQL 表达式兼容性 23
USER 特殊值 41
UTC TIMESTAMP 特殊值 41
按字母顺序排序的语句列表 235
保留字 4
比较运算符 12
变量 43
标识符 8
表达式 17
常量 19
关键字 4
函数 106
局部变量 43
连接级变量 45
列名称 19
逻辑运算符 13
全局变量 45
三值逻辑 33
搜索条件 26
搜索条件中的子查询 27
算术运算符 14
特殊值 37

位运算符 15
谓词 26
文档约定 236
运算符 12
运算符优先级 16
注释 53
子查询 19
字符串 10
字符串运算符 15
SQL 语法中的常见元素 236
SQL 语句
ALLOCATE DESCRIPTOR 语法 240
ALTER DATABASE 语法 242
ALTER DBSPACE 语法 246
ALTER EVENT 语法 248
ALTER FUNCTION 语法 250
ALTER PROCEDURE 语法 252
ALTER PUBLICATION 语法 254
ALTER REMOTE MESSAGE TYPE 语法 256
ALTER SERVER 语法 258
ALTER SYNCHRONIZATION DEFINITION 语法 260
ALTER SYNCHRONIZATION SITE 语法 263
ALTER SYNCHRONIZATION SUBSCRIPTION 语法 265
ALTER SYNCHRONIZATION TEMPLATE 语法 267
ALTER SYNCHRONIZATION USER 语法 270
ALTER TABLE 语法 272
ALTER TRIGGER 语法 279
ALTER VIEW 语法 281
ALTER WRITEFILE 语法 283
BACKUP 语法 285
BEGIN DECLARE 语法 424
BEGIN TRANSACTION 语法 291
BEGIN 和 END 语法 288
BREAK Transact-SQL 语法 659
BYE 语法 471
CALL 语法 294
CASE 语法 296
CHECKPOINT 语法 299
CLEAR 语法 300
CLOSE 语法 301
COMMENT 语法 303
COMMIT 语法 305
CONFIGURE 语法 307
CONNECT 语法 308
CONTINUE Transact-SQL 语法 659
CREATE COMPRESSED DATABASE 语法 312
CREATE DATABASE 语法 314
CREATE DBSPACE 语法 319
CREATE DECRYPTED FILE 语法 321
CREATE DOMAIN 语法 322
CREATE ENCRYPTED FILE 语法 324
CREATE EVENT 语法 326
CREATE EXISTING TABLE 语法 332
CREATE EXPANDED DATABASE 语法 312
CREATE EXTERNLOGIN 语法 335
CREATE FUNCTION 语法 337
CREATE INDEX 语法 341
CREATE MESSAGE Transact-SQL 语法 345
CREATE PROCEDURE Transact-SQL 语法 355
CREATE PROCEDURE 语法 347
CREATE PUBLICATION 语法 357
CREATE REMOTE MESSAGE TYPE 语法 360
CREATE SCHEMA 语法 362
CREATE SERVER 语法 364
CREATE STATISTICS 语法 366
CREATE SUBSCRIPTION 语法 367
CREATE SYNCHRONIZATION DEFINITION 语法 369
CREATE SYNCHRONIZATION SITE 语法 372
CREATE SYNCHRONIZATION SUBSCRIPTION 语法 375

CREATE SYNCHRONIZATION TEMPLATE 语法 377
CREATE SYNCHRONIZATION USER 语法 380
CREATE TABLE 语法 395
CREATE TRIGGER Transact-SQL 语法 415
CREATE TRIGGER 语法 408, 412
CREATE VARIABLE 语法 417
CREATE VIEW 语法 418
CREATE WRITEFILE 语法 420
DEALLOCATE DESCRIPTOR 语法 423
DEALLOCATE 语法 422
DECLARE CURSOR Transact-SQL 语法 431
DECLARE CURSOR 语法 426
DECLARE LOCAL TEMPORARY TABLE 语法 433
DECLARE 语法 425
DELETE 语法 435
DELETE （定位）语法 438
DESCRIBE 语法 440
DISCONNECT 语法 444
DROP CONNECTION 语法 448
DROP DATABASE 语法 447
DROP DATATYPE 语法 445
DROP DBSPACE 语法 445
DROP DOMAIN 语法 445
DROP EVENT 语法 445
DROP EXTERNLOGIN 语法 449
DROP FUNCTION 语法 445
DROP INDEX 语法 445
DROP MESSAGE 语法 445
DROP PROCEDURE 语法 445
DROP PUBLICATION 语法 450
DROP REMOTE MESSAGE TYPE 语法 451
DROP SERVER 语法 452
DROP STATEMENT 语法 453
DROP STATISTICS 语法 454
DROP SUBSCRIPTION 语法 456
DROP SYNCHRONIZATION DEFINITION 语法 457
DROP SYNCHRONIZATION SITE 语法 459
DROP SYNCHRONIZATION SUBSCRIPTION 语法 460
DROP SYNCHRONIZATION TEMPLATE 语法 461
DROP SYNCHRONIZATION USER 语法 463
DROP TABLE 语法 445
DROP TRIGGER 语法 445
DROP VARIABLE 语法 464
DROP VIEW 语法 445
DROP 语法 445
END DECLARE 语法 424
EXECUTE IMMEDIATE 语法 467
EXECUTE Transact-SQL 语法 469
EXECUTE 语法 465
EXIT 语法 471
EXPLAIN 语法 473
FETCH 语法 475
FOR 语法 480
FORWARD TO 语法 482
FROM 子句语法 484
GET DATA 语法 489
GET DESCRIPTOR 语法 492
GET OPTION 语法 494
GOTO Transact-SQL 语法 496
GRANT CONSOLIDATE 语法 502
GRANT PUBLISH 语法 504
GRANT REMOTE DBA 语法 508
GRANT REMOTE 语法 505
GRANT 语法 497
HELP 语法 509
IF Transact-SQL 语法 512
IF 语法 510
INCLUDE 语法 514
INPUT 语法 515

INSERT 语法 520
INSTALL 语法 524
LEAVE 语法 527
LOAD STATISTICS 语法 529
LOAD TABLE 语法 530
LOCK TABLE 语法 537
LOOP 语法 539
MESSAGE 语法 541
OPEN 语法 543
OUTPUT 语法 546
PARAMETERS 语法 551
PASSTHROUGH 语法 552
PREPARE TO COMMIT 语法 556
PREPARE 语法 553
PRINT Transact-SQL 语法 557
PUT 语法 558
QUIT 语法 471
RAISERROR Transact-SQL 语法 560
READ 语法 562
READTEXT Transact-SQL 语法 564
RELEASE SAVEPOINT 语法 565
REMOTE RESET 语法 566
REMOVE 语法 567
REORGANIZE TABLE 语法 568
RESIGNAL 语法 570
RESTORE DATABASE 语法 571
RESUME 语法 573
RETURN 语法 574
REVOKE CONSOLIDATE 语法 578
REVOKE PUBLISH 语法 579
REVOKE REMOTE DBA 语法 582
REVOKE REMOTE 语法 581
REVOKE 语法 576
ROLLBACK TO SAVEPOINT 语法 584
ROLLBACK TRIGGER 语法 585
ROLLBACK 语法 583
SAVEPOINT 语法 586
SELECT 语法 587
SET CONNECTION 语法 598
SET DESCRIPTOR 语法 599
SET OPTION Transact-SQL 语法 594
SET OPTION 语法 601, 605
SET REMOTE OPTION 语法 606
SET SQLCA 语法 608
SET Transact-SQL 语法 594
SET 语法 592
SETUSER 语法 610
SIGNAL 语法 612
START DATABASE 语法 614
START ENGINE 语法 616
START JAVA 语法 617
START LOGGING 语法 618
START SUBSCRIPTION 语法 619
START SYNCHRONIZATION DELETE 语法 621
STOP DATABASE 语法 623
STOP ENGINE 语法 625
STOP JAVA 语法 626
STOP LOGGING 语法 627
STOP SUBSCRIPTION 语法 628
STOP SYNCHRONIZATION DELETE 语法 629
SYNCHRONIZE SUBSCRIPTION 语法 630
SYSTEM 语法 632
TRIGGER EVENT 语法 633
TRUNCATE TABLE 语法 634
UNION 语法 636
UNLOAD TABLE 语法 640
UNLOAD 语法 638
UPDATE 语法 642, 649
UPDATE （定位）语法 647
VALIDATE INDEX 语法 652
VALIDATE TABLE 语法 653
WAITFOR 语法 655

WHENEVER 语法 657
WHILE Transact-SQL 语法 659
WHILE 语法 539
WRITETEXT Transact-SQL 语法 660
按字母顺序排序的语句列表 235
发送到远程服务器 482
文档约定 236
SQL 语言元素
关于 3
SQL 远程系统表
remoteoptiontype 715
SQLCA
INCLUDE 语句 514
设置 608
SQLCODE
特殊值 40
SQLDA
DESCRIBE SQL 语句 440
EXECUTE SQL 语句 465
INCLUDE 语句 514
UPDATE （定位）语句 647
分配内存 240
获取信息 492
设置 599
使用游标插入行 558
释放 423
SQLDIALECT 函数
SQL 语法 215
SQLSTATE
特殊值 40
SQRT 函数
SQL 语法 215
START DATABASE 语句
SQL 语法 614
START ENGINE 语句
Interactive SQL 语法 616
START JAVA 语句
SQL 语法 617
START LOGGING 语句
Interactive SQL 语法 618
START SUBSCRIPTION 语句
SQL 语法 619
START SYNCHRONIZATION DELETE 语句
SQL 语法 621
STOP DATABASE 语句
SQL 语法 623
STOP ENGINE 语句
SQL 语法 625
STOP JAVA 语句
SQL 语法 626
STOP LOGGING 语句
Interactive SQL 语法 627
STOP SUBSCRIPTION 语句
SQL 语法 628
STOP SYNCHRONIZATION DELETE 语句
SQL 语法 629
STR 函数
SQL 语法 216
StreamCompression 同步选项
MobiLink 同步客户端 389
STRING 函数
SQL 语法 217
STRING_RTRUNCATION 选项
Adaptive Server Enterprise 兼容性 594
STRIP 选项
LOAD TABLE 语句 533
STRTOUUID 函数
SQL 语法 217
STUFF 函数
SQL 语法 218
su
设置用户 610
SUBSCRIBE BY 子句 357
SUBSTR 函数
SQL 语法 219
SUBSTRING 函数
SQL 语法 219

SUM 函数
SQL 语法 220
sun 类
数据库中的 Java 88
sun.* 包
不受支持的类 90
suser_id Adaptive Server Enterprise 函数 115
suser_name Adaptive Server Enterprise 函数 115
Sybase Central
安装 Java 类 91
SYNCHRONIZE SUBSCRIPTION 语句
SQL 语法 630
SYS
系统表 663
SYS 组
在系统表中 737
SYSARTICLE 系统表
关于 665
SYSARTICLECOL 系统表
关于 666
SYSATTRIBUTE 系统表
关于 667
SYSATTRIBUTENAME 系统表
关于 668
SYSCAPABILITY 系统表
关于 669
SYSCAPABILITYNAME 系统表
关于 670
SYSCOLLATION 系统表
关于 671
SYSCOLLATIONMAPPINGS 系统表
关于 672
SYSCOLPERM 系统表
关于 673
SYSCOLSTAT 系统表 674
装载统计信息 529
SYSCOLUMN 系统表
关于 676
SYSDOMAIN 系统表
关于 679
SYSEVENT 系统表
关于 680
SYSEVENTTYPE 系统表
关于 682
SYSEXTENT 系统表
关于 683
SYSEXTERNLOGINS 系统表
关于 684
SYSFILE 系统表
关于 685
SYSFKCOL 系统表
关于 686
SYSFOREIGNKEY 系统表
关于 687
SYSGROUP 系统表
关于 689
SYSINDEX 系统表
关于 690
SYSINFO 系统表
关于 692
SYSIXCOL 系统表
关于 694
SYSJAR 系统表
关于 695
SYSJARCOMPONENT 系统表
关于 696
SYSJAVACLASS 系统表
关于 697
SYSLOGIN 系统表
关于 699
SYSOPTBLOCK 系统表 700
SYSOPTION 系统表
关于 701
SYSOPTJOINSTRATEGY 系统表 702
SYSOPTORDER 系统表 703
SYSOPTQUANTIFIER 系统表 704
SYSOPTREQUEST 系统表 705
SYSOPTREWRITE 系统表 706
SYSOPTSTAT 系统表 707

SYSPROCEDURE 系统表
关于 708
SYSPROCPARM 系统表
关于 710
SYSPROCPERM 系统表
关于 712
SYSPUBLICATION 系统表
关于 713
SYSREMOTEOPTION 系统表
关于 714
SYSREMOTEOPTIONTYPE 系统表
关于 715
SYSREMOTETYPE 系统表
关于 716
SYSREMOTEUSER 系统表
关于 717
SYSSCHEDULE 系统表
关于 719
SYSSERVERS 系统表
关于 721
添加服务器 364
sysservers 系统表
组件集成服务的远程服务器 365, 393
SYSSQLSERVERTYPE 系统表
关于 722
SYSSUBSCRIPTION 系统表
关于 723
SYSSYNC 系统表
关于 724
SYSTABLE 系统表
关于 726
SYSTABLEPERM 系统表
关于 729
SYSTEM 语句
Interactive SQL 语法 632
SYSTRIGGER 系统表
关于 731
SYSTYPEMAP 系统表
关于 734
SYSUSERMESSAGES 系统表
关于 735
SYSUSERPERM 系统表
关于 736
SYSUSERTYPE 系统表
关于 738
TableOrder 同步选项
MobiLink 同步客户端 389
TAN 函数
SQL 语法 221
TCP/IP
MobiLink 客户端使用 381
TempFreePercent 事件条件
关于 153
TempFreeSpace 事件条件
关于 153
TempSize 事件条件
关于 153
TEXT 数据类型
关于 63
TEXTPTR 函数
SQL 语法 221
TEXTSIZE 选项
Adaptive Server Enterprise 兼容性 594
THEN
IF 表达式 19
TIME 数据类型
关于 81
time 数据类型 75
TIMESTAMP
特殊值 40
TIMESTAMP 数据类型
关于 82
timestamp 数据类型 75
TINYINT 数据类型
关于 70
TODAY 函数
SQL 语法 222
TOP 子句

SELECT 语句 587
TRACEBACK 函数
SQL 语法 222
TRANSACTION ISOLATION LEVEL 选项
Adaptive Server Enterprise 兼容性 594
Transact-SQL
ANSI 等效性 202
bit 数据类型兼容性 74
BREAK 语句语法 659
CONTINUE 语句语法 659
CREATE MESSAGE SQL 语句语法 345
CREATE PROCEDURE 语句语法 355
CREATE SCHEMA 语句语法 362
CREATE TABLE 语句语法 404
CREATE TRIGGER 语句语法 415
DECLARE CURSOR 语句语法 431
DECLARE 节 289
EXECUTE 语句语法 469
GOTO 语句语法 496
IF 语句语法 512
PRINT 语句语法 557
QUOTED_IDENTIFIER 选项 24
RAISERROR 语句语法 560
READTEXT 语句语法 564
SET OPTION 语句语法 594
SET 语句语法 594
SQL 表达式兼容性 23
WHILE 语句语法 659
WRITETEXT 语句语法 660
按字母顺序排序的语句列表 235
比较运算符 12
常量 23
分类过程 815
货币数据类型 72
局部变量 43, 44
全局变量 45
日期时间兼容性 75
时间兼容性 75
外连接运算符 16
位运算符 15
系统过程 814
系统函数 115
系统目录 750
用户定义的数据类型 87
域 87
转换存储过程 229
TRANSACTSQL 函数
SQL 语法 223
TRIGGER EVENT 语句
SQL 语法 633
TRIM 函数
SQL 语法 223
TRUE 条件
IS TRUE 条件 32
三值逻辑 33
TRUNCATE TABLE 语句
SQL 语法 634
TRUNCATE 函数
SQL 语法 224
TRUNCNUM 函数
SQL 语法 225
trusted_certificates 流参数
使用 HTTP 的 MobiLink 客户端 385
使用 HTTPS 的 MobiLink 客户端 388
使用 TCP/IP 的 MobiLink 客户端 382
tsequal Adaptive Server Enterprise 函数 115
TSQL
语句指示符 238
TYPE 子句
CREATE SYNCHRONIZATION USER 380
UCASE 函数
SQL 语法 226
UNION 语句
SQL 语法 636
UNIQUEIDENTIFIER

二进制数据类型 84
UNKNOWN 条件
IS UNKNOWN 条件 32
UNLOAD TABLE 语句
SQL 语法 640
UNLOAD 语句
SQL 语法 638
UPDATE
IF UPDATE 子句 408, 412
Transact-SQL 中的 IF UPDATE 子句 415
UPDATE 权限
授予 497
UPDATE 语句
SQL 语法 642, 649
UPDATE （定位）语句
SQL 语法 647
UPDATING 条件
触发器 33
UPPER 函数
SQL 语法 226
url_suffix 流参数
使用 HTTP 的 MobiLink 客户端 386
使用 HTTPS 的 MobiLink 客户端 388
used_pgs Adaptive Server Enterprise 函数 115
USER
特殊值 41
user_id Adaptive Server Enterprise 函数 115
user_name Adaptive Server Enterprise 函数 115
ust 文件
创建 210
UTC TIMESTAMP
特殊值 41
UUID
NEWID 函数 SQL 语法 185
STRTOUUID 函数 SQL 语法 217
UNIQUEIDENTIFIER 数据类型 84
UUIDTOSTR 函数 SQL 语法 227
UUIDTOSTR 函数
SQL 语法 227
valid_name Adaptive Server Enterprise 函数 115
valid_user Adaptive Server Enterprise 函数 115
VALIDATE INDEX 语句
SQL 语法 652
VALIDATE TABLE 语句
SQL 语法 653
VARBINARY 数据类型
关于 85
VARCHAR 数据类型 62
VAREXISTS 函数
SQL 语法 228
Verbose 同步选项
MobiLink 同步客户端 389
version 流参数
使用 HTTP 的 MobiLink 客户端 386
使用 HTTPS 的 MobiLink 客户端 388
VIM 消息
关于 360, 451
VIM 消息类型
关于 256
VM
启动 Java 617
停止 Java 626
WAITFOR 语句
SQL 语法 655
Watcom-SQL
DECLARE 语句 425
WATCOMSQL 函数
SQL 语法 229
WEEKS 函数
SQL 语法 229
WHEN
CASE 表达式 20
WHENEVER 语句
嵌入式 SQL 语法 657
WHERE 子句
SELECT 语句 588
搜索条件 26
WHILE 语句

SQL 语法 539
Transact-SQL 语法 659
WITH CHECKPOINT 选项
LOAD TABLE 语句 533
WITH HOLD 子句
OPEN SQL 语句 543
WRITETEXT 语句
Transact-SQL 语法 660
xp_cmdshell 系统过程
语法 809
xp_msver 系统过程
语法 810
xp_read_file 系统过程
语法 811
xp_scanf 系统过程
语法 812
xp_sendmail 系统过程
语法 806
xp_sprintf 系统过程
语法 812
xp_startmail 系统过程
语法 804
xp_startsmtp 系统过程
语法 805
xp_stopmail 系统过程
语法 808
xp_stopsmtp 系统过程
语法 809
xp_write_file 系统过程
语法 813
Y2K 100
YEARS 函数
SQL 语法 231
YMD 函数
SQL 语法 232

A

安全性
复制 508, 582
安装
Java 类 524

B

版本号
检索 810
版权
检索 810
绑定变量
EXECUTE SQL 语句 465
OPEN 语句 543
描述游标 440
包
不受支持的 90
受支持的 88
保存点
创建 586
回退到保存点 584
释放 565
保留字
SQL 语法 4
用作标识符 24
备份
创建 285
创建事件 326
到磁带 285
恢复数据库 571
比较
搜索条件 26
比较日期和时间 77
比较运算符
SQL 语法 12
Transact-SQL 兼容性 12
数据转换 95
与 Adaptive Server Enterprise 的兼容性 12
变更

dbspace 246
表 272
触发器 279
发布 254
过程 252
函数 250
事件 248
视图 281
授予权限 497
数据库 242
写文件 283
远程服务器属性 258
远程消息类型 256
变量
SQL 语法 43
创建 SQL 417
从描述符区内获取 492
局部变量 43
全局变量 45
删除 SQL 464
设置值 592
声明 SQL 425
标签
语句 237, 496
标识符
ASA 中的最大长度 8
SQL 语法 8
关于 8
表
变更 272
插入行到 520
创建 395
创建代理 332
创建局部临时 433
从表导出数据到文件 546
从文件将数据导入到 515
复制 272
更新 649
截断 634
批量装载 530
删除 445
锁定 537
校验 653
卸载 640
重命名 276
重组 568
表达式
CASE 表达式 20
IF 表达式 19
Java 表达式 22
SQL 语法 17
SQL 运算符优先级 16
Transact-SQL 兼容性 23
常量 19
列名称 19
与 Adaptive Server Enterprise 的兼容性 18
子查询 19
表达式的兼容性 23
表达式中的常量 19
表达式中的列名称 19
表的批量装载 530
表号 727
表列表
FROM 子句 484
表提示
FROM 子句 486
表约束 400
别名
列 587
在 DELETE 语句中 435
并发
锁定表 537
捕获
嵌入式 SQL 中的错误 657
不明确的字符串到日期的转换 102

不受支持的类
java.applet 90
java.awt 90
java.awt.datatransfer 90
java.awt.event 90
java.awt.image 90
sun.* 90
部分受支持的类
java.io.File 90
java.io.FileDescriptor 90
java.io.FileInputStream 90
java.io.FileOutputStream 90
java.io.RandomAccessFile 90
java.lang.ClassLoader 90
java.lang.Compiler 90
java.lang.Runtime (exec/load/loadlibrary) 90
java.lang.Thread 89
java.util.zip.Deflater 90
java.util.zip.Inflater 90

C

参数
Interactive SQL 命令文件 551
参照完整性
FROM 子句 484
操作 403
操作
参照完整性 403
操作系统
执行命令 632
插入
多行 465
宽插入 465
批量插入行 530
使用游标的行 558
行到表中 520
插入 BLOB 811
查看
Interactive SQL 过程分析数据 785, 787
产品名
检索 810
常量
SQL 字符串 10
SQL 语法 19
长列名 441
检索 441
超类型 95
撤消
发布权限 579
权限 576
通过回退事务撤消更改 583
统一权限 578
远程 DBA 权限 582
远程权限 581
成本模型
校准服务器 242
重新校准 242
程序包
安装 Java 类 524
删除 Java 类 567
池
启用连接池 610
初始化
数据库 314
触发
事件 633
触发器
@@identity 全局变量 52
TRUNCATE TABLE 语句 634
Watcom-SQL 412
变更 279
创建 408, 412
回退 585
删除 445

行级 409
语句级 409
在 Transact-SQL 中创建 415
触发条件
区分触发器操作 33
处理
Transact-SQL 中的错误 560
嵌入式 SQL 中的错误 657
创建
SQL 变量 417, 425
Transact-SQL 中的触发器 415
Transact-SQL 中的游标 431
保存点 586
表 395
触发器 408, 412
存储过程 347
代理表 332, 396, 800
发布 357
服务器 364
函数 337
局部临时表 433
模式 362
视图 418
数据库 314
数据库备份 285
数据库文件 319
数据类型 86
索引 341
消息 345
写文件 420
压缩数据库 312
游标 426
域 86, 322
预订 367
远程消息类型 360
在 Transact SQL 中创建存储过程 355
磁带
创建数据库备份 285
磁盘传送时间模型
恢复缺省值 242
校准 242
磁盘空间
创建空间不足事件 326
创建事件 326
磁盘空间不足
创建事件 326
从文件中读取 SQL 语句 562
存储过程
sa_audit_string 755
sa_check_commit 755
sa_conn_activity 756
sa_conn_compression_info 757
sa_conn_info 758
sa_conn_properties 759, 760
sa_conn_properties_by_name 761
sa_db_info 761
sa_db_properties 762
sa_disk_free_space 763
sa_eng_properties 763
sa_flush_cache 764
sa_flush_statistics 764
sa_get_dtt 765
sa_get_histogram 765
sa_get_request_profile 766
sa_get_request_times 766
sa_get_server_messages 767
sa_index_density 768
sa_index_levels 768
sa_java_loaded_classes 769
sa_locks 769
sa_make_object 771
sa_migrate 773
sa_migrate_create_fks 776
sa_migrate_create_remote_fks_list 777

sa_migrate_create_remote_table_list 779
sa_migrate_create_tables 781
sa_migrate_data 782
sa_migrate_drop_proxy_tables 784
sa_procedure_profile 785
sa_procedure_profile_summary 787
sa_reset_identity 789
sa_server_option 790
sa_statement_text 793
sa_table_fragmentation 793
sa_table_page_usage 794
sa_validate 795
sp_login_environment 796
sp_remote_columns 796
sp_remote_exported_keys 797
sp_remote_imported_keys 798
sp_remote_tables 799
sp_servercaps 800
sp_tsql_environment 801
xp_cmdshell 809
xp_msver 810
xp_read_file 811
xp_scanf 812
xp_sendmail 806
xp_sprintf 812
xp_startmail 804
xp_startsmtp 805
xp_stopmail 808
xp_stopsmtp 809
xp_write_file 813
查看分析数据 785, 787
创建 347
扩展列表 803
外部函数调用 337, 350
系统过程 753
在 Transact SQL 中创建 355
在 Transact-SQL 中执行 469
在动态 SQL 中执行 467
转换 T-SQL 229
错误
创建事件 326
发信号 612
用户定义的消息 735
在 Transact-SQL 中引发 560
在嵌入式 SQL 中捕获 657
错误消息
ERRORMSG 函数 150

D

打开
游标 543
打印
消息窗口中的消息 557
大小写敏感性
Java 数据类型 92
比较运算符 12
和 SQL LIKE 条件 29
在目录中 692
大型数据库
索引存储 341
代理表
创建 332, 396
代理过程
创建 347
代码页
CREATE DATABASE 语句 315
SYSCOLLATION 系统表 671
数据存储 60
档案
创建数据库备份 285
从中恢复数据库 571
档案备份
支持的操作系统 285

导出
BLOB 813
Java 数据 548
卸载表 640
卸载结果集 638
导出数据
从表到文件 546
导入数据
从文件到表 515
登录
删除远程服务器 449
为远程服务器指派 335
地址
发布者 256
点运算符
和 >> 22
电子邮件
扩展存储过程 803
系统过程 806
调度
MobiLink 同步客户端 391
WAITFOR 655
创建事件 326
事件 248
调度事件
WAITFOR 语句 655
触发 633
调用
过程 294
迭代
通过游标 480
定位 DELETE 语句
SQL 语法 438
定义
变更表 272
动态 SQL
执行过程 467
读取
文本和图像值来自数据库 564
游标中的行 475
读取文件
存储过程 811, 813
断开连接
创建事件 326
多行插入 465
多行读取
FETCH 语句 476
OPEN 语句 544
多字节字符集
卸载数据 532, 640

E

二进制大对象
GET DATA SQL 语句 489
SET 语句示例 593
插入 811
从列获取 489
导出 813
导入 ASE 生成的 BCP 文件 532
二进制数据类型 83
事务日志注意事项 246
二进制数据
从列获取 489
二进制数据类型
BINARY 83
IMAGE 84
LONG BINARY 83
UNIQUEIDENTIFIER 84
VARBINARY 85
关于 83

F

发布

变更 254
创建 357
更新 644, 650
删除 450
设计 650
发布权限
撤消 579
授予 504
发布者
GRANT PUBLISH 语句 504
地址 256, 360, 451
远程 505
发送
SQL 语句到远程服务器 482
发送和检索日期值 101
发信号
Transact-SQL 中的错误 560
错误 612
异常 570
反馈
提供 xix
文档 xix
反斜线
SQL 字符串中 10
返回
过程值 574
返回代码
Interactive SQL 471
方法签名
Java 351
方括号
SQL 标识符 8
数据库对象 8
放入
行在游标中 558
非链接的事务模式 291
分隔 SQL 字符串 8
分类过程
列表 755
分类过程 (ASE)
sp_column_privileges 815
sp_columns 815
sp_fkeys 815
sp_pkeys 815
sp_special_columns 815
sp_sproc_columns 815
sp_stored_procedures 815
sp_tables 815
Transact-SQL 列表 814
Transact-SQL, 列表 815
分类系统过程
关于 753
分配
磁盘空间 246
描述符区的内存 240
分组
语句 288
服务器
变更远程属性 258
创建 364
创建空闲事件 326
启动数据库 616
删除远程 452
停止数据库 625
赋值
为 SQL 变量赋值 592
复合语句
关于 288
兼容性 289
复制
过程 252

G

高度加密

CREATE DATABASE 语句 316
高速缓存
刷新 764
隔离级别
游标 543
更改
口令 497
更新
表和列 649
发布和预订 644
根据连接 644
连接 650
列但不记录 660
行 642
更新列权限 673
功能
远程服务器 669, 670
估计
显式选择性估计 34
关闭
Interactive SQL 471
连接 448
游标 301
关键字
SQL 语法 4
关系
在系统表中 687
规则
SQL 语言语法 4
归类
SORTKEY 函数 210
归类序列 3
CREATE DATABASE 语句 315
SYSCOLLATION 系统表 671
和 SQL LIKE 条件 29
过程
CREATE PROCEDURE SQL 语句 355
Transact-SQL, 列表 814
变更 252
创建 347
调用 294
返回值 574
复制 252
可变结果集 348, 441, 553
扩展列表 803
目录 , 列表 755
删除 445
退出 574
外部函数调用 337, 350
系统 753
在 Transact-SQL 中创建 355
在 Transact-SQL 中引发错误 560
在 Transact-SQL 中执行存储 469
在动态 SQL 中执行 467
重新开始执行 573
过程分析
查看 Interactive SQL 785
过程摘要 785, 787
在 Interactive SQL 中 790
在 Interactive SQL 中查看 787
在 Interactive SQL 中禁用 791
在 Interactive SQL 中启用 791

H

函数
ABS 函数 SQL 语法 119
ACOS Han2Shu4 SQL Yu3Fa3 119
ARGN 函数 SQL 语法 120
ASCII 函数 SQL 语法 121
ASIN 函数 SQL 语法 121
ATAN 函数 SQL 语法 122
ATAN2 函数 SQL 语法 122
ATN2 函数 SQL 语法 122

AVG 函数 SQL 语法 123
BYTE_LENGTH 函数 SQL 语法 124
BYTE_SUBSTR 函数 SQL 语法 125
CAST 函数 SQL 语法 125
CEILING 函数 SQL 语法 126
CHAR 函数 SQL 语法 127
CHAR_LENGTH 函数 SQL 语法 128
CHARINDEX 函数 SQL 语法 127
COALESCE 函数 SQL 语法 129
col_length Adaptive Server Enterprise 函数 115
col_name Adaptive Server Enterprise 函数 115
COMPARE 函数 SQL 语法 129
CONNECTION_PROPERTY 函数 SQL 语法 131
CONVERT 函数 SQL 语法 132
COS 函数 SQL 语法 134
COT 函数 SQL 语法 135
COUNT 函数 SQL 语法 135
CSCONVERT 函数 SQL 语法 136
curunreservedpgs Adaptive Server Enterprise 函数 115
data_pgs Adaptive Server Enterprise 函数 115
datalength Adaptive Server Enterprise 函数 115
DATALENGTH 函数 SQL 语法 138
DATE 函数 SQL 语法 139
DATEADD 函数 SQL 语法 139
DATEDIFF 函数 SQL 语法 140
DATEFORMAT 函数 SQL 语法 142
DATENAME 函数 SQL 语法 143
DATEPART 函数 SQL 语法 143
DATETIME 函数 SQL 语法 144
DAY 函数 SQL 语法 144
DAYNAME 函数 SQL 语法 145
DAYS 函数 SQL 语法 145
db_id Adaptive Server Enterprise 函数 115
DB_ID 函数 SQL 语法 147
db_name Adaptive Server Enterprise 函数 115
DB_NAME 函数 SQL 语法 147
DB_PROPERTY 函数 SQL 语法 148
DEGREES 函数 SQL 语法 148
DIFFERENCE 函数 SQL 语法 149
DOW 函数 SQL 语法 150
ERRORMSG 函数 SQL 语法 150
ESTIMATE 函数 SQL 语法 151
ESTIMATE_SOURCE 函数 SQL 语法 152
EVENT_CONDITION 函数 SQL 语法 153
EVENT_CONDITION_NAME 函数 SQL 语法 155
EVENT_PARAMETER 函数 SQL 语法 155
EXP 函数 SQL 语法 156
EXPERIENCE_ESTIMATE 函数 SQL 语法 157
EXPLANATION 函数 SQL 语法 157
FLOOR 函数 SQL 语法 159
GET_IDENTITY 函数 SQL 语法 159
GETDATE 函数 SQL 语法 161
GRAPHICAL_PLAN 函数 SQL 语法 161
GRAPHICAL_ULPLAN 函数 SQL 语法 163
GREATER 函数 SQL 语法 164
HEXTOINT 函数 SQL 语法 164
host_id Adaptive Server Enterprise 函数 115
host_name Adaptive Server Enterprise 函数 115
HOUR 函数 SQL 语法 165
HOURS 函数 SQL 语法 165
IDENTITY 函数 SQL 语法 166
IFNULL 函数 SQL 语法 167
index_col Adaptive Server Enterprise 函数 115
INDEX_ESTIMATE 函数 SQL 语法 168
INSERTSTR 函数 SQL 语法 168
INTTOHEX 函数 SQL 语法 169
ISDATE 函数 SQL 语法 170
ISNULL 函数 SQL 语法 170
ISNUMERIC 函数 SQL 语法 171
Java 109
LCASE 函数 SQL 语法 172
lct_admin Adaptive Server Enterprise 函数 115

LEFT 函数 SQL 语法 172
LENGTH 函数 SQL 语法 173
LESSER 函数 SQL 语法 174
LIST 函数 SQL 语法 174
LOCATE 函数 SQL 语法 175
LOG 函数 SQL 语法 176
LOG10 函数 SQL 语法 177
LONG_ULPLAN 函数 SQL 语法 177
LOWER 函数 SQL 语法 178
LTRIM 函数 SQL 语法 179
MAX 函数 SQL 语法 179
MIN 函数 SQL 语法 180
MINUTE 函数 SQL 语法 181
MINUTES 函数 SQL 语法 181
MOD 函数 SQL 语法 183
MONTH 函数 SQL 语法 183
MONTHNAME 函数 SQL 语法 184
MONTHS 函数 SQL 语法 184
NEWID 函数 SQL 语法 185
NEXT_CONNECTION 函数 SQL 语法 187
NEXT_DATABASE 函数 SQL 语法 188
NOW 函数 SQL 语法 189
NULLIF 函数 SQL 语法 189
NUMBER 函数 SQL 语法 190
object_id Adaptive Server Enterprise 函数 115
object_name Adaptive Server Enterprise 函数 115
PATINDEX 函数 SQL 语法 191
PI 函数 SQL 语法 192
PLAN 函数 SQL 语法 193
POWER 函数 SQL 语法 194
proc_role Adaptive Server Enterprise 函数 115
PROPERTY 函数 SQL 语法 195
PROPERTY_DESCRIPTION 函数 SQL 语法 195
PROPERTY_NAME 函数 SQL 语法 196
PROPERTY_NUMBER 函数 SQL 语法 197
QUARTER 函数 SQL 语法 197
RADIANS 函数 SQL 语法 198
RAND 函数 SQL 语法 198
REMAINDER 函数 SQL 语法 199
REPEAT 函数 SQL 语法 199
REPLACE 函数 SQL 语法 200
REPLICATE 函数 SQL 语法 201
reserved_pgs Adaptive Server Enterprise 函数 115
REWRITE 函数 SQL 语法 202
RIGHT 函数 SQL 语法 204
ROUND 函数 SQL 语法 204
rowcnt Adaptive Server Enterprise 函数 115
RTRIM 函数 SQL 语法 205
SECOND 函数 SQL 语法 205
SECONDS 函数 SQL 语法 206
SHORT_ULPLAN 函数 SQL 语法 207
show_role Adaptive Server Enterprise 函数 115
SIGN 函数 SQL 语法 208
SIMILAR 函数 SQL 语法 209
SIN 函数 SQL 语法 209
SORTKEY 函数 SQL 语法 210
SOUNDEX 函数 SQL 语法 213
SPACE 函数 SQL 语法 214
SQLDIALECT 函数 SQL 语法 215
SQRT 函数 SQL 语法 215
STR 函数 SQL 语法 216
STRING 函数 SQL 语法 217
STRTOUUID 函数 SQL 语法 217
STUFF 函数 SQL 语法 218
SUBSTR 函数 SQL 语法 219
SUBSTRING 函数 SQL 语法 219
SUM 函数 SQL 语法 220
suser_id Adaptive Server Enterprise 函数 115
suser_name Adaptive Server Enterprise 函数 115
TAN 函数 SQL 语法 221
TEXTPTR 函数 SQL 语法 221
TODAY 函数 SQL 语法 222

TRACEBACK 函数 SQL 语法 222
Transact-SQL 115
TRANSACTSQL 函数 SQL 语法 223
TRIM 函数 SQL 语法 223
TRUNCATE 函数 SQL 语法 224
TRUNCNUM 函数 SQL 语法 225
tsequal Adaptive Server Enterprise 函数 115
UCASE 函数 SQL 语法 226
UPPER 函数 SQL 语法 226
used_pgs Adaptive Server Enterprise 函数 115
user_id Adaptive Server Enterprise 函数 115
user_name Adaptive Server Enterprise 函数 115
UUIDTOSTR 函数 SQL 语法 227
valid_name Adaptive Server Enterprise 函数 115
valid_user Adaptive Server Enterprise 函数 115
VAREXISTS 函数 SQL 语法 228
WATCOMSQL 函数 SQL 语法 229
WEEKS 函数 SQL 语法 229
YEARS 函数 SQL 语法 231
YMD 函数 SQL 语法 232
变更 250
创建 337
从用户定义的函数返回值 574
从用户定义的函数退出 574
函数的类型 106
集合 106
日期和时间 107
删除 445
数据类型转换 SQL 106
数字 111
图像 SQL 118
文本 SQL 118
系统 115
用户定义的 109
杂类 110
字符串 113
函数 , 数字
  ATAN 122
合并
  多个选择语句的结果 636
换行字符
  SQL 字符串中 10
恢复
  LOAD TABLE 534
  数据库从档案 571
回退
  触发器 585
  事务 583
  事务到保存点 584
活动超时
  数据库服务器 790
获取
  来自列的二进制数据 489
  来自描述符区的信息 492
  选项值 494

J

基表
  创建 404
集合函数
  字母顺序列表 106
技术支持
  新闻组 xix
计划
  SQL 语法 157, 161, 193
  和游标 157, 161, 193
  获取文本说明 473
记录
  更新列但不 660
  在 Interactive SQL 中开始 618
  在 Interactive SQL 中停止 627
加密
  数据库文件 316

文件 324
加密算法
CREATE DATABASE 语句 316
监控性能
确定执行时间 766
兼容性
NULL 56
Transact-SQL 比较运算符 12
Transact-SQL 表达式 23
Transact-SQL 局部变量 43
Transact-SQL 全局变量 45
T-SQL 表达式和 QUOTED IDENTIFIER 选项 24
日期时间 75
检查点操作
数据库 299
检查点日志
CHECKPOINT SQL 语句 299
建立
保存点 586
角色名
关于 402
外键 403
截断
表 634
结果集
可变 348, 441, 553
卸载 638
形状 441
重新开始过程执行 573
结束
回退事务 583
解密
文件 321
近似数据类型
关于 64
静态游标
声明 426
局部变量
SQL 语法 43
局部临时表
创建 433
拒绝
授予权限 497, 576

K

开始
用户定义的事务 291
预订 619
在 Interactive SQL 中记录 618
可变结果集
来自过程 348, 441, 553
可更新视图 521
空白填充
CREATE DATABASE 语句 316
空闲服务器
创建事件 326
控制语句
CALL SQL 语法 294
CASE SQL 语句 296
GOTO Transact-SQL 语句 496
IF SQL 语句 510
LEAVE SQL 语句 527
LOOP SQL 语句 539
Transact-SQL BREAK 语句 659
Transact-SQL CONTINUE 语句 659
Transact-SQL IF 语句 512
Transact-SQL WHILE 语句 659
WHILE SQL 语句 539
口令
更改 497
在系统表中 736
块读取
FETCH 语句 476
OPEN 语句 544
宽插入 465

L

类
  Java 方法 109
  不受支持的 90
  部分受支持的 90
  删除 Java 567
  受支持的 88
  在数据库中安装 Java 524
类型
  关于数据类型 59
类型转换
  关于 94
联合
  多个选择语句 636
连接
  ANSI 等价 202
  FROM 子句语法 484
  创建失败事件 326
  创建事件 326
  到数据库 308
  根据连接更新 644, 650
  更新 649
  基于连接删除行 435
  启用池 610
  删除 448
  设置 598
  设置最大号 561
  使用 RAISERROR 禁止 561
  在 Interactive SQL 中删除 444
连接级变量
  SQL 语法 45
连接运算符
  与 Adaptive Server Enterprise 的兼容性 16
连接字符串
  字符串运算符 15
链接的事务模式 291
两阶段提交
  准备 556
列
  SYSCOLUMNS 系统视图 745
  变更 272
  别名 587
  更新 649
  更新但不记录 660
  获取二进制数据 489
  权限 673
  用户定义的数据类型 87
  域 87
  约束 87, 400
  在系统表中 676
  重命名 276
列名称
  SQL 语法 19
列统计
  SYSCOLSTAT 系统表 674
  SYSCOLSTATS 系统视图 745
  使用 CREATE STATISTICS 更新 366
  选择性估计 34
列统计信息
  使用 LOAD TABLE 更新 534
临时表
  Transact-SQL 404
  不允许本地视图 418
  创建 395, 404
  声明局部 433
临时文件
  确定可用空间 763
临时选项
  设置 601
  在 in Interactive SQL 中设置 605
逻辑运算符
  SQL 语法 13
  三值逻辑 33
  与 Adaptive Server Enterprise 的兼容性 14

M

描述
游标 440
游标行为 349
描述符
DESCRIBE 语句 440
FETCH SQL 语句 475
准备语句 553
描述符区
EXECUTE SQL 语句 465
UPDATE （定位）语句 647
分配内存 240
获取信息 492
设置 599
释放 423
名称
列名称 19
命令
执行操作系统 632
命令文件
Interactive SQL 参数 551
读取 SQL 语句 562
模式
创建 362
系统表 663
模式匹配
LIKE 条件 28
PATINDEX 函数 191
和大小写敏感性 29
和归类 29
通配符 191
最大模式长度 29
目录
系统表 663

N

内存
为描述符区分配 240

P

排序
SORTKEY 函数 210
在系统表中 671
派生表
FROM 子句 484
批量操作
卸载 638
卸载表 640
撇号
SQL 字符串中 10
频率
发送消息 502, 505

Q

启动
Java VM 617
创建事件 326
数据库 614
数据库服务器 616
在数据库抽取中预订 566
直通模式 552
千年虫
问题 100
千年虫问题 100
签名
Java 方法 351
嵌入式 SQL
ALLOCATE DESCRIPTOR 语法 240
BEGIN DECLARE 语句语法 424
CLOSE 语句语法 301

CONNECT 语句语法 308
DEALLOCATE DESCRIPTOR 语句语法 423
DEALLOCATE 语句语法 422
DECLARE CURSOR 语句语法 426
DELETE （定位）语句语法 438
DESCRIBE 语句语法 440
DISCONNECT 语句语法 444
DROP STATEMENT 语句语法 453
END DECLARE 语句语法 424
EXECUTE IMMEDIATE 语句语法 467
EXECUTE 语句语法 465
EXPLAIN 语句语法 473
FETCH 语句语法 475
GET DATA 语句语法 489
GET DESCRIPTOR 语句语法 492
GET OPTION 语句语法 494
INCLUDE 语句语法 514
OPEN 语句语法 543
PREPARE 语句语法 553
PUT 语句语法 558
SET CONNECTION 语句语法 598
SET DESCRIPTOR 语句语法 599
SET SQLCA 语句语法 608
WHENEVER 语句语法 657
嵌套
用户定义的事务 291
清除
Interactive SQL 窗格 300
权限
CONSOLIDATE 502
SYSCOLAUTH 系统视图 745
撤消 576
撤消发布 579
撤消统一 578
撤消远程 581
撤消远程 DBA 582
授予 497
授予发布 504
授予统一 502
授予远程 505
授予远程 DBA 508
系统视图 748
在系统表中 673, 729
全局变量
@@char_convert 48
@@client_csid 48
@@client_csname 48
@@connections 48
@@cpu_busy 48
@@error 48
@@identity 48, 51
@@idle 48
@@io_busy 48
@@isolation 48
@@langid 48
@@language 48
@@max_connections 48
@@maxcharlen 48
@@ncharsize 48
@@nestlevel 48
@@pack_received 48
@@pack_sent 48
@@packet_errors 48
@@procid 48
@@rowcount 48
@@servername 48
@@spid 48
@@sqlstatus 48
@@textsize 48
@@thresh_hysteresis 48
@@timeticks 48
@@total_errors 48
@@total_read 48

@@total_write 48
@@tranchained 48
@@trancount 48
@@transtate 48
@@version 48
SQL 语法 45
触发器和 @@identity 52
全局变量 s
@@dbts 46
@@error 46
@@fetch_status 46
@@identity 46
@@isolation 46
@@procid 46
@@rowcount 46
@@servername 46
@@sqlstatus 46
@@version 46
全局临时表
创建 395
全局唯一标识符
NEWID 函数 SQL 语法 185
全局自动增量
创建事件 326
缺省值
CURRENT DATABASE 37
CURRENT DATE 37
CURRENT PUBLISHER 37
CURRENT TIME 38
CURRENT TIMESTAMP 38
CURRENT USER 38
CURRENT UTC TIMESTAMP 39
LAST USER 39
SQLCODE 40
SQLSTATE 40
TIMESTAMP 40
USER 41
UTC TIMESTAMP 41
自动增量 397

R

日期 75
2000 年 100
比较 77
不明确的字符串转换 102, 104
插入 79
查询 76
从字符串转换 75
检索 79
将字符串解释为日期 78
解释 79
明确说明 78
向数据库发送 75
转换函数 107
转换问题 104
日期部分
关于 108
日期到字符串的转换 104
日期的存储方式 100
日期函数
字母顺序列表 107
日期时间
转换函数 107
日志文件
分配空间 246
分析请求级日志 766
确定可用空间 763
儒略日 145
闰年
关于 102

S

三值逻辑
NULL 值 55
SQL 语法 33
删除
DBSPACE 445
Interactive SQL 中的连接 444
Java 类 567
SQL 变量 464
表 445
表中的所有行 634
触发器 445
发布 450
过程 445
函数 445
连接 448
权限 576
事件 445
视图 445
授予权限 497, 576
数据库文件 447
索引 445
行 435
用户 576
优化程序统计信息 454
游标中的行 438
域 445
预订 456
远程服务器 452
远程服务器的登录 449
远程消息类型 451
准备的语句 453
商标信息
检索 810
舍入误差
关于 64
设置
Interactive SQL 中的选项 307, 605
SQL 变量的值 592
SQLCA 608
Transact-SQL 中的选项 594
连接 598
描述符区 599
选项 601
用户 610
远程选项 606
审计
添加注释 755
声明
Transact-SQL 中的游标 431
变量 SQL 425
嵌入式 SQL 中的主机变量 424
游标 426
升级
数据库 242
十六进制常量
关于 64
十六进制转义序列
SQL 字符串中 10
时间
比较 77
查询 76
向数据库发送 75
转换函数 107
时间戳列 399
时间函数
字母顺序列表 107
使用 SQL 语句参考 236
示例数据库
关于 asademo.db xviii
事件
变更 248
触发 633
创建和调度 326
调度 248

删除 445
事件条件
列表 153
事务
创建保存点 586
回退 583
回退到保存点 584
开始用户定义的 291
嵌套用户定义的 291
提交 305
事务管理
BEGIN TRANSACTION SQL 语句 291
Transact-SQL 305
在 Transact-SQL 中 291
事务模式
非链接的 291
链接的 291
事务日志
TRUNCATE TABLE 语句 634
备份 285
分配空间 246
确定可用空间 763
事务日志镜像
确定可用空间 763
释放
保存点 565
描述符区 423
游标 422
视图
变更 281
创建 418
可更新 521
删除 445
索引 342
系统视图 749
授予
发布权限 504
权限 497
统一权限 502
远程 DBA 权限 508
远程权限 505
受支持的类
Java 88
java.beans 88
java.io 88
java.lang 89
java.lang.reflect 89
java.math 89
java.net 89
java.net.PlainDatagramSocketImpl 89
java.rmi 89
java.rmi.dgc 89
java.rmi.registry 89
java.rmi.server 89
java.security 89
java.security.acl 89
java.security.interfaces 89
java.SQL 89
java.text 89
java.util 89
java.util.zip 89
属性
变更远程服务器 258
服务器 195
数据
从表导出到文件 546
从文件导入到表 515
选择行 587
数据访问计划
获取文本说明 473
数据库
Java 启用 242
备份 285
创建 314
创建文件 319

创建压缩 312
从档案中恢复 571
检查点操作 299
将批量数据装入 530
结构 663
连接到 308
模式 663
启动 614
迁移 773
删除文件 447
升级 242
停止 623
文件 685
系统表 663
系统过程 753
校验 795
卸载表 640
卸载数据 638
数据库抽取
启动预订 566
数据库对象
标识 8
注释 303
数据库服务器
启动 616
停止 625
选项 790
数据库模式
关于 663
数据库文件
存储索引 341
删除 447
数据库选项
DATE_ORDER 和明确的日期 78
QUOTED_IDENTIFIER 与 T-SQL 兼容性 24
Transact-SQL 兼容性 801
Transact-SQL 中的设置 594
初始设置 796, 801
数据库中的 Java
CREATE DATABASE 语句 317
OUTPUT 语句 548
安装类 524
导出 548
启用数据库 242
删除类 567
升级数据库 242
数据类型
bit 74
Java 类 88
SQL 转换函数 106
UNIQUEIDENTIFIER 84
比较运算符转换 95
创建 322
二进制 83
关于 59
货币 72
兼容性 75
日期 75
删除用户定义的 445
舍入误差 64
时间 75
数值 64
用户定义的 738
用户定义的域 86
在系统表中 679, 738
转换 Java 和 SQL 97
字符 60
数据类型转换
Java-to-SQL 97
SQL-to-Java 98
比较运算符 95
关于 94
数据类型转换函数 106
数学表达式

算术运算符 14
数值数据类型
关于 68
数字函数
字母顺序列表 111
双引号
SQL 标识符 8
数据库对象 8
搜索条件
ANY 或 ALL 条件 27
EXISTS 条件 32
IS NULL 条件 32
IS TRUE 或 FALSE 条件 32
IS UNKNOWN 条件 32
LIKE 条件 28
SQL BETWEEN 条件 28
SQL IN 条件 31
SQL 语法 26
三值逻辑 33
显式选择性估计 34
子查询 27
算术
运算符和 SQL 语法 14
算术运算符
与 Adaptive Server Enterprise 的兼容性 14
碎片
表 568
碎片整理
REORGANIZE TABLE 568
索引
表使用 342
创建 341
命名 342
删除 445
视图 342
所有者 342
外键 342
唯一 341
唯一名称 342
系统视图 745
校验 652
压缩 568
在系统表中 690, 694
主键 342
自动创建 342
锁
显示 769
锁定
表 537
阻塞 758

T

特殊表
关于 663
特殊值
CURRENT DATABASE 37
CURRENT DATE 37
CURRENT PUBLISHER 37
CURRENT TIME 38
CURRENT TIMESTAMP 38
CURRENT USER 38
CURRENT UTC TIMESTAMP 39
LAST USER 39
NULL 55
SQL 语法 37
SQLCODE 40
SQLSTATE 40
TIMESTAMP 40
USER 41
UTC TIMESTAMP 41
特殊字符
SQL 字符串中 10
提交
事务 305

准备两阶段 556
替换对象
sa_make_object 771
添加
Java 类 524
服务器 364
索引 341
消息 345
条件
ALL 27
ANY 27
BETWEEN 28
EXISTS 32
IN 31
IS NULL 32
LIKE 28
SQL 搜索条件 26
三值逻辑 33
搜索 26
真值 32
子查询 27
停止
Java VM 626
数据库服务器 625
在 Interactive SQL 中记录 627
直通模式 552
停止数据库 623
停止预订 628
通配符
LIKE 条件 28
模式匹配 191
通信协议
MobiLink 380
通用唯一标识符
NEWID 函数 SQL 语法 185
同步预订 630
统计
CREATE STATISTICS 语句 366
SYSCOLSTAT 系统表 674
刷新 764
统计信息
删除优化程序 454
使用 LOAD TABLE 更新 534
装载 529
统一数据库
撤消权限 578
发布 579
图标
手册中使用的 xvi
图像
从数据库中读取 564
图像 SQL 函数 118
退出
Interactive SQL 471
过程 574
退出时间
数据库服务器 790

W

外表
在系统表中 687
外部登录
删除远程服务器 449
为远程服务器指派 335
外键
角色名 402
完整性约束 402
未命名 402
系统视图 745
远程表 797, 798
在系统表中 686, 687
完整性
约束 400
唯一
约束 400

唯一索引 341
位运算符
  SQL 语法 15
谓词
  ANY 或 ALL 条件 27
  EXISTS 条件 32
  IS NULL 条件 32
  IS TRUE 或 FALSE 条件 32
  IS UNKNOWN 条件 32
  LIKE 条件 28
  SQL BETWEEN 条件 28
  SQL IN 条件 31
  SQL 语法 26
  SQL 子查询 27
  比较运算符 12
  三值逻辑 33
  显式选择性估计 34
文本
  从数据库中读取 564
文本函数 118
文档
  SQL Anywhere Studio xiii
  SQL 语法约定 236
  约定 xv
文件
  xp_read_file 系统过程 811
  xp_write_file 系统过程 813
  创建数据库 319
  创建写 420
  读取 SQL 语句 562
  加密 324
  将数据从文件导出到 546
  将数据导入到表 515
  解密 321
  为数据库分配空间 246
文件大小
  创建事件 326
文字字符串
  SQL 语法 10

X

系统表
  DUMMY 664
  Java 741
  SYSARTICLE 665
  SYSARTICLECOL 666
  SYSATTRIBUTE 667
  SYSATTRIBUTENAME 668
  SYSCAPABILITY 669
  SYSCAPABILITYNAME 670
  SYSCOLLATE 671
  SYSCOLLATIONMAPPINGS 672
  SYSCOLPERM 673
  SYSCOLSTAT 674
  SYSCOLUMN 676
  SYSDOMAIN 679
  SYSEVENT 680
  SYSEVENTTYPE 682
  SYSEXTENT 683
  SYSEXTERNLOGINS 684
  SYSFILE 685
  SYSFKCOL 686
  SYSFOREIGNKEY 687
  SYSGROUP 689
  SYSINDEX 690
  SYSINFO 692
  SYSIXCOL 694
  SYSJAR 695
  SYSJARCOMPONENT 696
  SYSJAVACLASS 697
  SYSLOGIN 699
  SYSOPTBLOCK 700
  SYSOPTION 701

SYSOPTJOINSTRATEGY 702
SYSOPTORDER 703
SYSOPTQUANTIFIER 704
SYSOPTREQUEST 705
SYSOPTREWRITEK 706
SYSOPTSTAT 707
SYSPROCEDURE 708
SYSPROCPARM 710
SYSPROCPERM 712
SYSPUBLICATION 713
SYSREMOTETYPE 716
SYSREMOTEUSER 717
SYSSCHEDULE 719
SYSSERVERS 721
SYSSQLSERVERTYPE 722
SYSSUBSCRIPTION 723
SYSSYNC 724
SYSTABLE 726
SYSTABLEPERM 729
SYSTRIGGER 731
SYSTYPEMAP 734
SYSUSERMESSAGES 735
SYSUSERPERM 736
SYSUSERTYPE 738
Transact-SQL 750
关于 663
系统视图 743
系统表中的数据类型 684
系统调用
xp_cmdshell 系统过程 809
从存储过程中 809
系统过程
sa_audit_string 755
sa_check_commit 755
sa_conn_activity 756
sa_conn_compression_info 757
sa_conn_info 758
sa_conn_properties 759, 760
sa_conn_properties_by_name 761
sa_db_info 761
sa_db_properties 762
sa_disk_free_space 763
sa_eng_properties 763
sa_flush_cache 764
sa_flush_statistics 764
sa_get_dtt 765
sa_get_histogram 765
sa_get_request_profile 766
sa_get_request_times 766
sa_get_server_messages 767
sa_index_density 768
sa_index_levels 768
sa_java_loaded_classes 769
sa_locks 769
sa_make_object 771
sa_migrate 773
sa_migrate_create_fks 776
sa_migrate_create_remote_fks_list 777
sa_migrate_create_remote_table_list 779
sa_migrate_create_tables 781
sa_migrate_data 782
sa_migrate_drop_proxy_tables 784
sa_procedure_profile 785
sa_procedure_profile_summary 787
sa_reset_identity 789
sa_server_option 790
sa_statement_text 793
sa_table_fragmentation 793
sa_table_page_usage 794
sa_validate 795
sp_addgroup 814
sp_addlogin 814
sp_addmessage 814

sp_addtype 814
sp_adduser 814
sp_changegroup 814
sp_dboption 814
sp_dropgroup 814
sp_droplogin 814
sp_dropmessage 814
sp_droptype 814
sp_dropuser 814
sp_getmessage 814
sp_helptext 814
sp_login_environment 796
sp_password 814
sp_remote_columns 796
sp_remote_exported_keys 797
sp_remote_imported_keys 798
sp_remote_tables 799
sp_servercaps 800
sp_tsql_environment 801
Sybase Central 754
Transact-SQL 814
Transact-SQL 列表 814
xp_cmdshell 809
xp_msver 810
xp_read_file 811
xp_scanf 812
xp_sendmail 806
xp_sprintf 812
xp_sstopsmtp 809
xp_startmail 804
xp_startsmtp 805
xp_stopmail 808
xp_write_file 813
查看定义 754
创建消息 345
概述 754
关于 753
扩展列表 803
目录列表 755
系统过程和函数
关于 753
系统函数
兼容性 115
字母顺序列表 115
系统和分类存储过程 755
系统目录 745
Transact-SQL 750
关于 663
系统视图
Sybase Central 744
SYSARTICLECOLS 744
SYSARTICLES 744
SYSCAPABILITIES 744
SYSCATALOG 745
SYSCOLAUTH 745
SYSCOLSTATS 745
SYSCOLUMNS 745
SYSFOREIGNKEYS 745
SYSGROUPS 745
SYSINDEXES 745
SYSOPTIONS 746
SYSOPTORDERS 746
SYSOPTPLAN 746
SYSOPTSTRATEGIES 746
SYSPROCAUTH 746
SYSPROCPARMS 746
SYSPUBLICATIONS 746
SYSREMOTEOPTIONS 747
SYSREMOTETYPES 747
SYSREMOTEUSERS 747
SYSSCYNCUSERS 748
SYSSUBSCRIPTIONS 747
SYSSYNCDEFINITIONS 747

SYSSYNCPUBLICATIONDEFAULTS 747
SYSSYNCS 747
SYSSYNCSITES 748
SYSSYNCSUBSCRIPTIONS 748
SYSSYNCTEMPLATES 748
SYSTABAUTH 748
SYSTRIGGERS 748
SYSUSERAUTH 748
SYSUSERLIST 749
SYSUSEROPTIONS 749
SYSUSERPERMS 749
SYSVIEWS 749
定义 744
关于 743
显式选择性估计
关于 34
显示
消息 541
消息窗口中的消息 557
限制返回的行数 587
相关名
在 DELETE 语句中 435
项目
系统表 665, 666
向导
Java 类创建 91
消息
变更远程类型 256
创建 345
创建远程类型 360
删除远程类型 451
显示 541
消息窗口
打印消息 557
消息控制参数
设置 606
校验
表 653
数据库 795
索引 652
校准服务器 242
协议
使用 ActiveSync 的 MobiLink 客户端 388
使用 HTTP 的 MobiLink 客户端 383
使用 HTTPS 的 MobiLink 客户端 386
使用 TCP/IP 的 MobiLink 客户端 381
写文件
变更 283
创建 420
卸载
表 640
结果集 638
卸载数据
多字节字符集 532, 640
新闻组
技术支持 xix
行
插入到表中 520
从表中删除所有 634
从游标中读取 475
从游标中删除 438
更新 642
批量插入 530
使用游标插入 558
限制返回的数目 587
卸载 638
选择 587
行级触发器 409
行数 727
行数限制 587
性能
更新 650
压缩统计 757
预分配空间 246
重新校准 I/O 成本模型 244
重新校准服务器 242

选项
MobiLink 同步客户端 389
QUOTED_IDENTIFIER 与 T-SQL 兼容性 24
Transact-SQL 兼容性 801
Transact-SQL 中的设置 594
初始设置 796, 801
关于 601
获取值 494
设置 601
设置远程 606
替换 790
系统视图 746, 749
在 Interactive SQL 中设置 307, 605
在系统表中 701
选择
构成联合 636
进行卸载 638
行 587
选择列表
描述游标 440
选择性估计
估计源 152
用户定义的 34
循环
通过游标 480

Y

压缩
统计 757
压缩数据库
创建 312
页
查看 PCTFREE 设置 667
页面大小
创建数据库 315
页面使用
表 794
疑难解答
request_level_logging 790
非标准磁盘驱动器 244
日志操作 790
锁 769
异常
发信号 612
重发信号 570
引发
Transact-SQL 中的错误 560
引号
SQL 标识符 8
数据库对象 8
与 Adaptive Server Enterprise 的兼容性 23
引擎
启动数据库 616
停止数据库 625
用户
删除 576
设置 610
用户 ID
撤消 576
创建 497
更改权限和口令 497
系统视图 748
在系统表中 727, 736
用户定义的函数
Java 109
创建 337
返回值 574
退出 574
字母顺序列表 109
用户定义的数据类型
Transact-SQL 87
创建 322
删除 445
用户定义的域
关于 86

用户估计
关于 34
用户号 736
用户提供的选择性估计
关于 34
优化
定义现有表和 332
优化程序
CREATE STATISTICS 语句 366
显式选择性估计 34
优化程序计划
获取文本说明 473
优化程序统计信息
删除 454
优先级
SQL 运算符优先级 16
游标
EXPLAIN 语句语法 473
插入行使用 558
打开 543
读取行 475
关闭 301
描述 440
描述行为 349
删除行 438
声明 426
释放 422
循环通过 480
在 SELECT 语句中设置可更新性 589
在 Transact-SQL 中声明 431
重新描述 349
准备语句 553
语法
ALL 条件 27
ANY 条件 27
CASE 表达式 20
CURRENT DATABASE 特殊值 37
CURRENT DATE 特殊值 37
CURRENT PUBLISHER 特殊值 37
CURRENT TIMESTAMP 特殊值 38
CURRENT USER 特殊值 38
CURRENT UTC TIMESTAMP 特殊值 39
EXISTS 条件 32
IF 表达式 19
IN 条件 31
IS NULL 条件 32
IS TRUE 或 FALSE 条件 32
Java 表达式 22
LAST USER 特殊值 39
LIKE 条件 28
NULL 值 55
SOME 条件 27
SQL BETWEEN 条件 28
SQL CURRENT TIME 特殊值 38
SQL 保留字 4
SQL 变量 43
SQL 标识符 8
SQL 表达式 17
SQL 关键字 4
SQL 函数 106
SQL 语句 236
SQL 运算符 12
SQL 运算符优先级 16
SQL 子查询 19
SQLCODE 特殊值 40
SQLSTATE 特殊值 40
TIMESTAMP 特殊值 40
Transact-SQL 表达式兼容性 23
USER 特殊值 41
UTC TIMESTAMP 特殊值 41
比较运算符 12
常量 19
局部变量 43
连接级变量 45

列名称 19
逻辑运算符 13
全局变量 45
三值逻辑 33
搜索条件 26
搜索条件中的 SQL 子查询 27
算术运算符 14
特殊值 37
位运算符 15
谓词 26
文档约定 236
注释 53
字符串 10
字符串运算符 15
语法约定
SQL 语句 237
语句
ALLOCATE DESCRIPTOR 语法 240
ALTER DATABASE 语法 242
ALTER DBSPACE 语法 246
ALTER EVENT 语法 248
ALTER FUNCTION 语法 250
ALTER PROCEDURE 语法 252
ALTER PUBLICATION 语法 254
ALTER REMOTE MESSAGE TYPE 语法 256
ALTER SERVER 语法 258
ALTER SYNCHRONIZATION DEFINITION 语法 260
ALTER SYNCHRONIZATION SUBSCRIPTION 语法 265
ALTER SYNCHRONIZATION TEMPLATE 语法 267
ALTER SYNCHRONIZATION USER 语法 270
ALTER TABLE 语法 272
ALTER TRIGGER 语法 279
ALTER VIEW 语法 281
ALTER WRITEFILE 语法 283
BACKUP 语法 285
BEGIN DECLARE 语句 SQL 语法 424
BEGIN TRANSACTION 语法 291
BEGIN 语法 288
BREAK Transact-SQL 语法 659
BYE 语法 471
CALL 语法 294
CASE 语法 296
CHECKPOINT 语法 299
CLEAR 语法 300
CLOSE 语法 301
COMMENT 语法 303
COMMIT 语法 305
CONFIGURE 语法 307
CONNECT 语法 308
CONTINUE Transact-SQL 语法 659
CREATE COMPRESSED DATABASE 语法 312
CREATE DATABASE 语法 314
CREATE DBSPACE 语法 319
CREATE DECRYPTED FILE 语法 321
CREATE DOMAIN 语法 322
CREATE ENCRYPTED FILE 语法 324
CREATE EVENT 语法 326
CREATE EXISTING TABLE 语法 332
CREATE EXPANDED DATABASE 语句 SQL 语法 312
CREATE EXTERNLOGIN 语法 335
CREATE FUNCTION 语法 337
CREATE INDEX 语法 341
CREATE MESSAGE Transact-SQL 语法 345
CREATE PROCEDURE Transact-SQL 语法 355
CREATE PROCEDURE 语法 347
CREATE PUBLICATION 语法 357
CREATE REMOTE MESSAGE 语法 360
CREATE SCHEMA 语法 362
CREATE SERVER 语法 364

CREATE STATISTICS 语法 366
CREATE SUBSCRIPTION 语法 367
CREATE SYNCHRONIZATION DEFINITION 语法 369
CREATE SYNCHRONIZATION SITE 语法 372
CREATE SYNCHRONIZATION SUBSCRIPTION 语法 375
CREATE SYNCHRONIZATION TEMPLATE 语法 377
CREATE SYNCHRONIZATION USER 语法 380
CREATE TABLE 语法 395
CREATE TRIGGER Transact-SQL 语法 415
CREATE TRIGGER 语法 408, 412
CREATE VARIABLE 语法 417
CREATE VIEW 语法 418
CREATE WRITEFILE 语法 420
DEALLOCATE DESCRIPTOR 语法 423
DEALLOCATE 语法 422
DECLARE CURSOR Transact-SQL 语法 431
DECLARE CURSOR 语法 426
DECLARE LOCAL TEMPORARY TABLE 语法 433
DECLARE 语法 425
DELETE 语法 435
DELETE （定位）语法 438
DESCRIBE 语法 440
DISCONNECT 语法 444
DROP CONNECTION 语法 448
DROP DATABASE 语法 447
DROP DATATYPE 语法 445
DROP DBSPACE 语法 445
DROP DOMAIN 语法 445
DROP EVENT 语法 445
DROP EXTERNLOGIN 语法 449
DROP FUNCTION 语法 445
DROP INDEX 语法 445
DROP MESSAGE 语法 445
DROP OPTIMIZER STATISTICS 语法 454
DROP PROCEDURE 语法 445
DROP PUBLICATION 语法 450
DROP REMOTE MESSAGE TYPE 语法 451
DROP SERVER 语法 452
DROP STATEMENT 语法 453
DROP SUBSCRIPTION 语法 456
DROP SYNCHRONIZATION DEFINITION 语法 457
DROP SYNCHRONIZATION SITE 语法 459
DROP SYNCHRONIZATION SUBSCRIPTION 语法 460
DROP SYNCHRONIZATION TEMPLATE 语法 461
DROP SYNCHRONIZATION USER 语法 463
DROP TABLE 语法 445
DROP TRIGGER 语法 445
DROP VARIABLE 语法 464
DROP VIEW 语法 445
DROP 语法 445
END DECLARE 语法 424
EXECUTE IMMEDIATE 语法 467
EXECUTE Transact-SQL 语法 469
EXECUTE 语法 465
EXIT 语法 471
EXPLAIN 语法 473
FETCH 语法 475
FOR 语法 480
FORWARD TO 语法 482
FROM 子句 484
GET DATA 语法 489
GET DESCRIPTOR 语法 492
GET OPTION 语法 494
GOTO Transact-SQL 语法 496
GRANT CONSOLIDATE 语法 502
GRANT PUBLISH 语法 504
GRANT REMOTE DBA 语法 508

GRANT REMOTE 语法 505
GRANT 语法 497
HELP 语法 509
IF Transact-SQL 语法 512
IF 语法 510
INCLUDE 语法 514
INPUT 语法 515
INSERT 语法 520
INSTALL 语法 524
LEAVE 语法 527
LOAD STATISTICS 语法 529
LOAD TABLE 语法 530
LOCK TABLE 语法 537
LOOP 语句 539
MESSAGE 语法 541
OPEN 语法 543
OUTPUT 语法 546
PARAMETERS 语法 551
PASSTHROUGH 语法 552
PREPARE TO COMMIT 语法 556
PREPARE 语法 553
PRINT Transact-SQL 语法 557
PUT 语法 558
QUIT 语法 471
RAISERROR Transact-SQL 语法 560
READ 语法 562
READTEXT Transact-SQL 语法 564
RELEASE SAVEPOINT 语法 565
REMOTE RESET 语法 566
REMOVE 语法 567
REORGANIZE TABLE 语法 568
RESIGNAL 语法 570
RESTORE DATABASE 语法 571
RESUME 语法 573
RETURN 语法 574
REVOKE CONSOLIDATE 语法 578
REVOKE PUBLISH 语法 579
REVOKE REMOTE DBA 语法 582
REVOKE REMOTE 语法 581
REVOKE 语法 576
ROLLBACK TO SAVEPOINT 语法 584
ROLLBACK TRIGGER 语法 585
ROLLBACK 语法 583
SAVEPOINT 语法 586
SELECT 语法 587
SET CONNECTION 语法 598
SET DESCRIPTOR 语法 599
SET OPTION 语法 601, 605
SET REMOTE OPTION 语法 606
SET SQLCA 语法 608
SET Transact-SQL 语法 594
SET 语法 592
SETUSER 语法 610
SIGNAL 语法 612
START DATABASE 语法 614
START ENGINE Interactive SQL 语法 616
START JAVA 语法 617
START LOGGING 语法 618
START SUBSCRIPTION 语法 619
START SYNCHRONIZATION DELETE 语法 621
STOP DATABASE 语法 623
STOP ENGINE 语法 625
STOP JAVA 语法 626
STOP LOGGING Interactive SQL 语法 627
STOP SUBSCRIPTION 语法 628
STOP SYNCHRONIZATION DELETE 语法 629
SYNCHRONIZE SUBSCRIPTION 语法 630
SYSTEM Interactive 语法 632
Transact-SQL 语法 660
TRIGGER EVENT 语法 633
TRUNCATE TABLE 语法 634
UNION 语法 636

UNLOAD TABLE 语法 640
UNLOAD 语法 638
UPDATE 语法 642, 649
UPDATE （定位）语法 647
VALIDATE INDEX 语法 652
VALIDATE TABLE 语法 653
WAITFOR 语法 655
WHENEVER 嵌入式 SQL 语法 657
WHILE Transact-SQL 语法 659
分组 288
删除准备的 453
执行准备的 465
准备 553
语句标签
GOTO Transact-SQL 语句 496
关于 237
语句级触发器 409
语句适用性指示符 238
语句语法
BEGIN 和 END SQL 语句 288
FROM 子句 484
按字母顺序排序的语句列表 235
文档约定 236
语言元素
SQL 语法 4
域
Transact-SQL 87
创建 322
关于 86
删除 445
预订
创建 367
更新 644, 650
开始 619
删除 456
停止 628
同步 630
在数据库抽取中启动 566
元素
SQL 语言语法 4
远程 DBA 权限
撤消 582
授予 508
远程表
创建 396
列 796
列出 799
外键 797, 798
主键 797, 798
远程服务器
变更属性 258
创建表 395
功能 669, 670
将 SQL 语句发送到 482
容量 800
删除 452
删除登录 449
指派登录 335
远程过程
创建 347, 350
在 Transact SQL 中创建 355
远程权限
撤消 581
授予 505
远程数据访问
FORWARD TO 语句 482
远程消息类型
变更 256
创建 360
删除 451
远程选项
设置 606
远程用户
REVOKE REMOTE 语句 581
约定

SQL 语言语法 4
文档 xv
约束
ALTER TABLE 语句 274
列 400
运算符
比较运算符 12
关于 12
逻辑运算符 13
算术运算符 14
位运算符 15
运算符优先级 16
字符串运算符 15
运算符优先级
SQL 语法 16
运算顺序
SQL 运算符优先级 16

Z

支持
新闻组 xix
直方图
SYSCOLSTAT 系统表 674
使用 CREATE STATISTICS 更新 366
使用 LOAD TABLE 更新 534
选择性估计 34
直通模式
启动 552
停止 552
执行
Transact-SQL 中的存储过程 469
操作系统命令 632
来自文件的 SQL 语句 562
重新开始过程执行 573
准备的语句 465
值
从过程返回 574
指派
远程服务器的登录 335
指示符
注释 53
指示符变量
关于 236
只读
锁定表 537
中括号
SQL 标识符 8
数据库对象 8
重发信号
异常 570
重命名
表 276
列 276
重新开始
过程执行 573
重新描述游标 349
重新校准成本模型 242
重组
表 568
主表
在系统表中 687
主机变量
语法 236
在嵌入式 SQL 中声明 424
主键
UUID 和 GUID 185
列顺序 402
生成唯一值 185
使用 UUID 生成唯一值 185
完整性约束 402
远程表 797, 798
在系统表中 677, 727
主控台
显示消息 541
注释

SQL 语法 53
数据库对象 303
注释指示符 53
转换
字符串到日期 75
转换不明确的日期 102
转换函数
数据类型 106
字母顺序列表 106
转换字符串
关于 113
转义字符
INPUT SQL 语句 515
OUTPUT SQL 语句 546
装载
批量插入 530
准备
两阶段提交 556
语句 553
准备的语句
删除 453
执行 465
资源权限
在系统表中 736
子查询
SQL 语法 19
在 SQL 搜索条件中 27
子串
关于 219
替换 200
自动拨号
使用 HTTP 的 MobiLink 客户端 384
使用 HTTPS 的 MobiLink 客户端 387
使用 TCP/IP 的 MobiLink 客户端 382
自动增量
重新设置值 789
字
保留 4
字符串
SQL 函数 113
到日期的不明确转换 102, 104
更改分隔字符串的解释 24
关于 10
替换 200
与 Adaptive Server Enterprise 的兼容性 10
转换到日期 75
字符串函数
字母顺序列表 113
字符串运算符
SQL 语法 15
与 Adaptive Server Enterprise 的兼容性 15
字符函数
字母顺序列表 113
字符集
SORTKEY 函数 210
存储 60
字符数据
Adaptive Server Enterprise 兼容性 60
关于 60
字符串 10
字母字符
定义的 8
阻塞
识别 758
疑难解答 769