IBM Financial Crimes Alerts Insight with Watson
Version 1.0.3

Solution Guide

Note

Before using this information and the product it supports, read the information in “Notices” on page 63.

Product Information

This document applies to Version 1.0.3 and may also apply to subsequent releases.

Contents

Introduction

IBM Financial Crimes Alerts Insight with Watson (FCAI) is designed to reduce the overall cost of monitoring transactions while also complying with anti-money laundering regulations. The solution is designed to save time and money while also protecting financial institutions from fraud.

IBM FCAI helps financial and non-financial institutions to reduce exposure to money laundering and terrorism financing activities. It effectively monitors bank customer transactions daily and by using customer historical information and account profiles, it provides a whole picture to the bank management.

Audience

This guide is intended for administrators and users of the solution. It provides information on installation and configuration of the solution, and information about using the solution.

Finding information and getting help

To find product documentation on the web, access IBM® Knowledge Center (www.ibm.com/support/knowledgecenter).

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products. For more information, see Accessibility features.

The HTML documentation has accessibility features. PDF documents are supplemental and, as such, include no added accessibility features.

Forward-looking statements

This documentation describes the current functionality of the product. References to items that are not currently available may be included. No implication of any future availability should be inferred. Any such references are not a commitment, promise, or legal obligation to deliver any material, code, or functionality. The development, release, and timing of features or functionality remain at the sole discretion of IBM.

Samples disclaimer

Sample files may contain fictional data manually or machine generated, factual data that is compiled from academic or public sources, or data that is used with permission of the copyright holder, for use as sample data to develop sample applications. Product names that are referenced may be the trademarks of their respective owners. Unauthorized duplication is prohibited.

Chapter 1. Product overview

The IBM Financial Crimes Alerts Insight with Watson (FCAI) solution uses advanced analytics to identify behaviors that can help prevent financial crime before it happens while also investigating occurrences to minimize negative financial impact.

Financial crimes are growing in frequency and complexity and technological advances provide malicious insiders and organized criminals more opportunity to commit crimes. At the same time, business must remain compliant. New regulatory guidelines continue to arise with increased scrutiny and complexity, challenging even the most advanced compliance programs.

Suspicious activity reports (SARs) increasingly cross boundaries between fraud, cyber, and anti-money laundering (AML), further increasing the complexity. Regulators are pushing for more detailed reviews of implementations, especially for knowing your customer (KYC) requirements.

Keeping up with compliance risks can be difficult and costly, with high false positive rates.

Use IBM Financial Crimes Alerts Insight with Watson (FCAI) to

- continuously harnesses a powerful array of advanced analytics that are available to proactively fight the long-term war against financial criminals on multiple fronts
- help you more effectively resolve identities, relationships, and ambiguous patterns with entity and predictive analytics, broader cross channel visibility, and stronger risk scoring
- continuously adapt and embed new operating models through retrospective forensic and big data techniques
- help accelerate the investigation and management of cases through SAR filings with improved triage alerts, automation, and deep mining of data sets

Chapter 2. Installing IBM Financial Crimes Alerts Insight

You install IBM Financial Crimes Alerts Insight with Watson (FCAI) after you install the IBM Financial Crimes Insight with Watson core components.

For more information about installing the IBM Financial Crimes Insight with Watson core components, see Installing the Financial Crimes Insight platform.

The IBM Financial Crimes Insight with Watson core components installation provides a Hortonworks Data Platform (HDP) environment and a Kubernetes environment.

The IBM Financial Crimes Insight with Watson core components installation also configures the HDP environment for IBM FCAI.

However, you must install the IBM FCAI components to the Kubernetes environment that was created during the IBM Financial Crimes Insight with Watson core components installation.

Installing IBM FCAI

You must download all of the IBM Financial Crimes Alerts Insight with Watson (FCAI) installation files, and copy them to a common installation directory.

The installation files are available at IBM Fix Central (www.ibm.com/support/fixcentral/quickorder?product=ibm%2FOther+software%2FIBM+Financial+Crimes+Alerts+Insight+with+Watson+Private&fixids=1.0.3.0-Other-IBMFCAI-Linux-PH01445&source=SAR).

Note: Perform the following steps on the Kubernetes master node computer.

Procedure

1. Log on to the Kubernetes master node computer as the root user.
2. Create a directory to save the installation files to.

```
cd ${HOME}
```

```
mkdir -p /fcimedia/fcai
```

3. Download the installation files from IBM Fix Central, and save the files to the `/fcimedia/fcai` directory.
4. Run the following commands:

```
tar -xvf /fcimedia/fcai/SHA1SUM.tar -C /fcimedia/fcai
```

```
tar -O -xf /fcimedia/fcai/CNT2UEN.tar fcai-install-kit.tar.gz | tar xzf -
```

```
cp /fcimedia/fcco/CNT2KEN.tar /fcimedia/fcai/
```

```
cd ${HOME}/fcai-install-kit/helm/
```

```
cp ${HOME}/fci-install-kit/helm/install.hosts.properties .
```

   Enter y to overwrite the `install.hosts.properties` file in the last command.
5. Modify the `install.properties` file for the installation media location, the docker registry server, and the NFS server.

Note: The `install.hosts.properties` file lets you enter specific servers for the docker registry server and NFS server, such as if you are using a dedicated Docker registry server or a dedicated NFS server. If neither of these are specified in the `install.hosts.properties` file, then the FCI Platform `install.sh` script installs the Docker registry server and NFS server on the Kubernetes master node. If you specified a dedicated server for the docker registry or the NFS server, ensure that you enter the value for the appropriate server for your environment in the `install.properties` file.

Run the following commands to modify the `install.properties` file, assuming that the Docker registry server and the NFS server are not specified in the `install.hosts.properties` file (for example, they are running on the Kubernetes master node):

```
sed -i 's/media.dir = \/fcimedia/media.dir = \/fcimedia\/fcai/' ${HOME}/fcai-install-kit/
helm/install.properties
```

```
sed -i '40 i\'"external.docker.registry.url = $(hostname -f):5000" ${HOME}/fcai-install-kit/
helm/install.properties
```

```
sed -i '104 i\'"external.nfsserver = $(hostname -f)" ${HOME}/fcai-install-kit/helm/
install.properties
```

This command changes the following values in the `install.properties` file:

- Changes the media.dir value to the directory where you saved the installation files.
- Adds a line to set the external.docker.registry.url value. If you are using a dedicated Docker registry server, and not the Kubernetes master node, enter the line as:

  ```
  sed -i '40 i\'"external.docker.registry.url = hostname.domain.com:port_number" ${HOME}/
  fcai-install-kit/helm/install.properties
  ```

- Adds a line to set the external.nfsserver value. If you are using a dedicated NFS server, and not the Kubernetes master node, enter the line as:

  ```
  sed -i '104 i\'"external.nfsserver = hostname.domain.com:port_number" ${HOME}/fcai-install-
  kit/helm/install.properties
  ```

6. Open the `${HOME}/fcai-install-kit/helm/fcai-values.yaml` file in a text editor.
   a. Change the <hostname or IP address> value for the kafkaBootstrapServer property to the fully qualified domain name of the hadoop.secondary computer for a production installation.

      For example,

      ```
      kafkaBootstrapServer:mycomputer.mydomain.com:9096
      ```

      Do not enter a space after the : character.

      If you are installing a non-production environment, the kafkaBootstrapServer value should be the fully qualified domain name of the hadoop.master computer.

   b. Change the <hostname or IP address> value for the HDP_MASTER property to the fully qualified domain name of the hadoop.master computer.

   c. Change the <hostname or IP address> value for the HDP_SECONDARY property to the fully qualified domain name of the hadoop.secondary computer.

7. If the Ambari admin password is not admin, then open the `/opt/ibm/fci/install/fcai-hadoop.configure.cluster.sh` file in a text editor. The file is on the Ambari server.

   Change the following value to your admin user password. The default is admin.

   ```
   AMBARI_PASSWORD=admin
   ```

   You can also change this password after installation. For more information, see "Changing the default Ambari password after installation" on page 37.

8. From the ${HOME}/fcai-install-kit/helm directory, run the installation command:

```
./install.sh
```

When the installation finishes, you will see something like the following in the output:

```
2018-08-22 12:05:44-07:00 : [INFO][line 2272] Install - The installation of the Kubernetes
Cluster has completed successfully.  The install duration was [0 hours 10 minutes 27 seconds
elapsed time].
```

9. After the install is complete, wait about 5 - 10 minutes, and the run the following command:

```
kubectl get pods
```

The results should appear as:

```
NAME                                                READY     STATUS      RESTARTS   AGE
fcai-fci-alerts-insight-ar-75566787cb-kblnl         1/1       Running     0          22m
fcai-fci-alerts-insight-db-6db7dd6d7f-sfl4d         2/2       Running     0          22m
fcai-fci-alerts-insight-shiny-bcfdbd6bf-qvqqx       1/1       Running     0          22m
fcco-case-manager-fci-messaging-5645c5b86f-smk8w    8/8       Running     0          4h
fcco-case-manager-fci-solution-7ff4c89dbc-4fk5m     3/3       Running     0          4h
fcco-cedm-datastore-7648f7c9c7-q8w5m                2/2       Running     0          4h
fcco-cedm-integration-5c88889b4b-jqv25              3/3       Running     0          4h
fcco-cedm-ui-64b8884755-jqntm                       1/1       Running     0          4h
fcco-common-scripts-svss5                           0/1       Completed   0          4h
fcco-common-ui-nginx-7764f6576f-8gvqh               1/1       Running     0          4h
fcco-common-ui-nodejs-945bb4bcc-b599s               1/1       Running     0          4h
fcco-kafka-0                                        1/1       Running     0          4h
fcco-kafka-config-b11469f5-gtss2                    0/1       Completed   0          4h
fcco-kafka-zk-0                                     1/1       Running     0          4h
fcco-logging-es-6d55bd7f98-7lkkw                    1/1       Running     0          4h
fcco-logging-fb-mkhhq                               1/1       Running     0          4h
fcco-logging-kb-b89b5596c-trwrl                     1/1       Running     0          4h
fcco-logging-ls-7fd4b98c48-l4m5z                    1/1       Running     0          4h
fcco-security-audit-app-5b7f654b5-xd6x8             3/3       Running     0          4h
fcco-security-audit-datastore-64f6bd9645-k7wjv      2/2       Running     0          4h
fcco-security-auth-nodejs-6cf9677b84-bjkc7          1/1       Running     0          4h
fcco-security-auth-nodejs-6cf9677b84-dvzxk          1/1       Running     0          4h
fcco-security-auth-redis-784fd58d56-qpq2z           1/1       Running     0          4h
```

Post-installation configuration

Procedure

1. On the Kubernetes master node computer, run the following command:

```
kubectl edit cm fcco-common-ui
```

2. Edit the following lines to include `alert-review` at the end of the lines. You must use a comma to separate `case-manager` and `alert-review`.

For example, change:

```
APP_ROLE_ADMIN: case-manager
APP_ROLE_ANALYST: case-manager
APP_ROLE_DATA_SCIENTIST: case-manager
APP_ROLE_INVESTIGATOR: case-manager
APP_ROLE_SUPERVISOR: case-manager
```

So that `alert-review` is added:

```
APP_ROLE_ADMIN: case-manager,alert-review
APP_ROLE_ANALYST: case-manager,alert-review
APP_ROLE_DATA_SCIENTIST: case-manager,alert-review
APP_ROLE_INVESTIGATOR: case-manager,alert-review
APP_ROLE_SUPERVISOR: case-manager,alert-review
```

3. As the root user, run the following commands to modify the `/fci-exports/fci-cui-web/nginx.conf` file:

```
sed '29,31 s/#//' /fci-exports/fci-cui-web/nginx.conf > /fci-exports/fci-cui-web/temp1-nginx.conf
```

```
sed '133,146 s/#//' /fci-exports/fci-cui-web/temp1-nginx.conf > /fci-exports/fci-cui-web/temp2-nginx.conf
```

```
/usr/bin/mv -f /fci-exports/fci-cui-web/temp2-nginx.conf /fci-exports/fci-cui-web/nginx.conf
```

```
/usr/bin/rm /fci-exports/fci-cui-web/temp1-nginx.conf
```

These commands uncomment the following section in the file:

```
#upstream alert-review {
#  server fcai-fci-alerts-insight-ar.default.svc.cluster.local:4200;
#}

#location /alert-review {
      #proxy_pass  https://alert-review;
 #}

 #location /alert-review/api/v1.0/shiny/__sockjs__/ {
  #    rewrite ^/shiny/(.*)$ /$1 break;
  #  proxy_pass  https://alert_review:4200;
  #   proxy_redirect https://localhost:3838/ $scheme://$host/shiny/;
  #   proxy_http_version 1.1;
  #   proxy_set_header Upgrade $http_upgrade;
  #   proxy_set_header Connection $connection_upgrade;
  #   proxy_read_timeout 20d;
  #   proxy_buffering off;
#}
```

So that it becomes:

```
upstream alert-review {
  server fcai-fci-alerts-insight-ar.default.svc.cluster.local:4200;
}

location /alert-review {
      proxy_pass  https://alert-review;
 }

 location /alert-review/api/v1.0/shiny/__sockjs__/ {
      rewrite ^/shiny/(.*)$ /$1 break;
    proxy_pass  https://alert_review:4200;
     proxy_redirect https://localhost:3838/ $scheme://$host/shiny/;
     proxy_http_version 1.1;
     proxy_set_header Upgrade $http_upgrade;
     proxy_set_header Connection $connection_upgrade;
     proxy_read_timeout 20d;
     proxy_buffering off;
}
```

4. Run the following command to delete and re-create the Kubernetes pods so that your changes are applied.

```
kubectl delete pod $(kubectl get pods -l release=fcco,app=common-ui-nodejs -o jsonpath='{.items[*].metadata.name}') $(kubectl get pods -l release=fcco,app=common-ui-nginx -o jsonpath='{.items[*].metadata.name}')
```

The command deletes and re-creates the fcco-common-ui-nodejs and fcco-common-ui-nginx pods.

The command output should be similar to the following:

```
pod "fcco-common-ui-nodejs-6dbc7c775-4gnxb" deleted
pod "fcco-common-ui-nginx-7f4846b9ff-6rpwq" deleted
```

5. Wait a few minutes, and the run the following command:

```
kubectl get pods
```

The results should appear as:

```
NAME                                                READY     STATUS      RESTARTS   AGE
fcai-fci-alerts-insight-ar-75566787cb-kblnl         1/1       Running     0          42m
fcai-fci-alerts-insight-db-6db7dd6d7f-sfl4d         2/2       Running     0          42m
fcai-fci-alerts-insight-shiny-bcfdbd6bf-qvqqx       1/1       Running     0          42m
fcco-case-manager-fci-messaging-5645c5b86f-smk8w    8/8       Running     0          4h
fcco-case-manager-fci-solution-7ff4c89dbc-4fk5m     3/3       Running     0          4h
fcco-cedm-datastore-7648f7c9c7-q8w5m                2/2       Running     0          4h
fcco-cedm-integration-5c88889b4b-jqv25              3/3       Running     0          4h
fcco-cedm-ui-64b8884755-jqntm                       1/1       Running     0          4h
fcco-common-scripts-svss5                           0/1       Completed   0          4h
fcco-common-ui-nginx-7764f6576f-bnnwl               1/1       Running     0          5m
fcco-common-ui-nodejs-945bb4bcc-g2qtd               1/1       Running     0          5m
fcco-kafka-0                                        1/1       Running     0          4h
fcco-kafka-config-b11469f5-gtss2                    0/1       Completed   0          4h
fcco-kafka-zk-0                                     1/1       Running     0          4h
fcco-logging-es-6d55bd7f98-7lkkw                    1/1       Running     0          4h
fcco-logging-fb-mkhhq                               1/1       Running     0          4h
fcco-logging-kb-b89b5596c-trwrl                     1/1       Running     0          4h
fcco-logging-ls-7fd4b98c48-l4m5z                    1/1       Running     0          4h
fcco-security-audit-app-5b7f654b5-xd6x8             3/3       Running     0          4h
fcco-security-audit-datastore-64f6bd9645-k7wjv      2/2       Running     0          4h
fcco-security-auth-nodejs-6cf9677b84-bjkc7          1/1       Running     0          4h
fcco-security-auth-nodejs-6cf9677b84-dvzxk          1/1       Running     0          4h
fcco-security-auth-redis-784fd58d56-qpq2z           1/1       Running     0          4h
```

6. After the pods have been up for at least 2 minutes, you can access Alerts Insight from `https://fully_qualified_hostname_master:3080`

Verifying the installation

You can verify the IBM FCAI installation by using the Kubernetes `get pods` command.

Procedure

1. Log in to the Kubernetes master node computer as either the root user or the fciadmin user.
2. Run the following command:

```
kubectl get pods
```

The output is similar to the following:

```
NAME                                                READY     STATUS      RESTARTS   AGE
fcai-fci-alerts-insight-ar-cbccf956f-5trc8          1/1       Running     0          1d
fcai-fci-alerts-insight-db-75fc98f775-k7bnx         2/2       Running     0          1d
fcai-fci-alerts-insight-shiny-74459b54db-4hfmz      1/1       Running     0          1d
fcco-case-manager-fci-messaging-5787cf5c56-2vx4g    8/8       Running     0          1d
fcco-case-manager-fci-solution-56ff9c9ffd-s5n5n     3/3       Running     0          1d
fcco-cedm-datastore-7f995b9498-5pxbm                2/2       Running     0          1d
fcco-cedm-integration-564b569fcb-h4685              3/3       Running     0          1d
fcco-cedm-ui-7598b4b6d9-56sw5                       1/1       Running     0          1d
fcco-common-scripts-w2b8h                           0/1       Completed   0          1d
fcco-common-ui-nginx-7f4846b9ff-6rpwq               1/1       Running     0          1d
fcco-common-ui-nodejs-6dbc7c775-4gnxb               1/1       Running     0          1d
fcco-logging-es-5bc5ccfc7b-ltg29                    1/1       Running     0          1d
fcco-logging-fb-7c8p6                               1/1       Running     0          1d
fcco-logging-kb-5d69c59c-dxznr                      1/1       Running     0          1d
fcco-logging-ls-64cf675d7d-gt6ck                    1/1       Running     0          1d
fcco-rms-datastore-cdb9788f5-8gw85                  2/2       Running     0          1d
fcco-rms-designstudio-6978cdb966-8db4t              3/3       Running     0          1d
fcco-rms-odm-54657f49f-td4wk                        3/3       Running     0          1d
fcco-rms-odm-datastore-7f95d44c94-fbljh             2/2       Running     0          1d
fcco-security-audit-app-665bb4f5d9-tm22t            3/3       Running     0          1d
fcco-security-audit-datastore-5cbd799688-wvplv      2/2       Running     0          1d
fcco-security-auth-nodejs-7c885757d-s8ljq           1/1       Running     0          1d
```

```
fcco-security-auth-nodejs-7c885757d-v5dkc          1/1       Running     0          1d
fcco-security-auth-redis-79bddbcb7d-bnttc          1/1       Running     0          1d
```

3. In a web browser, log in to the application:

```
https://<hostname-of-Kubernetes-Master>:3080/
```

Chapter 3. Configuring IBM Financial Crimes Alerts Insight

You must perform some configuration tasks for IBM Financial Crimes Alerts Insight with Watson (FCAI).

Configuring the database

Before you run the analytics for IBM Financial Crimes Alerts Insight with Watson (FCAI), you must configure the database to allow alerts and their associated data to be stored.

Procedure

1. Log on to the Kubernetes master node computer as root, and connect to the database.

   You can run the following SQL command to verify the data types that are already defined:

   ```
   kubectl exec -ti $(kubectl get pods | grep insight-db | awk '{print $1}') -- su - db2inst1 -
   c 'db2 connect to amldb; db2 "select * from \"ALERT_REVIEW\".ALERT_DISPOSITIONS" | tr -s
   [:space:] | tr -s - | column -t'
   ```

2. Configure the tables in the amldb database as follows:

   "ALERT_REVIEW"."ALERT_DISPOSITIONS"
   Add the possible alert dispositions that might be set for an alert, such as Waived or Fraudulent. These values will be read from the column in the incoming alert defined in the Alert:DispositionColumn in the AMLAnalyticConfiguration.json (`disposition` by default). The possible values should be written to the TYPEDEF column.

   "ALERT_REVIEW"."TRANSACTION_TYPE"
   Add the valid transaction types. These values will be read from the column in the transaction table defined in the Transaction:TransactionTypeColumn in the AMLAnalyticConfiguration.json (`tx_type` by default). The possible values should be written to the TYPEDEF column.

   "ALERT_REVIEW"."ALERT_STATUS"
   Add the possible alert states that might be set for an alert, such as Open or Closed. These values will be read from the column in the incoming alert defined in the Alert:StatusColumn in the AMLAnalyticConfiguration.json (`status` by default). The possible values should be written to the TYPEDEF column.

   "ALERT_REVIEW"."ACCOUNT_TYPE"
   Add the possible account types, such as Checking or Saving. These values will be read from the column in the account table that is defined in the Account:TypeColumn in the AMLAnalyticConfiguration.json (`type` by default). The possible values should be written to the TYPEDEF column.

   "ALERT_REVIEW"."ACCOUNT_STATUS"
   Add the possible account type states, such as Checking or Saving. These values will be read from the column in the account table that is defined in the Account:StatusColumn in the AMLAnalyticConfiguration.json (`acct_stat` by default). The possible values should be written to the TYPEDEF column

Example

The following SQL statements have been created based on the values in the sample CSV files. To successfully process the sample data, you must run this SQL in the Db2 database. To process non-sample

data, update the SQL statements with the appropriate values, as identified in step 2, before you run the SQL in the Db2 database. The `cat` command creates a sample data file.

```
cat << EOF > /fci-exports/amldb/r103.sql <<EOF
INSERT INTO ALERT_REVIEW.ALERT_DISPOSITIONS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Unknown',
'The alert disposition is unknown.', 'unknown');
INSERT INTO ALERT_REVIEW.ALERT_DISPOSITIONS (NAME, DESCRIPTION, TYPEDEF) VALUES ('SAR', 'A SAR
was filed for the alert.', 'sar');
INSERT INTO ALERT_REVIEW.ALERT_DISPOSITIONS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Non-SAR', 'No
SAR was filed for the alert.', 'non-sar');

INSERT INTO ALERT_REVIEW.ALERT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Unknown', 'The
alert status is unknown', 'unknown');
INSERT INTO ALERT_REVIEW.ALERT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Open', 'The alert
is open', 'open');
INSERT INTO ALERT_REVIEW.ALERT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Closed', 'The alert
is closed', 'closed');

INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Unknown', 'Unknown
type', 'unknown');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Checking',
'Checking account', 'chk');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Money Market',
'Money market account', 'mm');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Savings', 'Savings
account', 'sav');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES
('Retirement','Retirement Account','retire');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES
('Investment','Investment Account','inv');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Loan','Loan
Account','loan');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Health
Savings','Health Savings Account','hlth sav');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Insurance
Policy','Insurance Policy','ins');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Credit
Card','Credit Card','cc');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Stored Value
Card','Stored Value Card','svcard');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Term Deposit','Term
Deposit','term');
INSERT INTO ALERT_REVIEW.ACCOUNT_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES
('Others','Others','o');

INSERT INTO ALERT_REVIEW.ACCOUNT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Unknown',
'Unknown', 'unknown');
INSERT INTO ALERT_REVIEW.ACCOUNT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Active', 'The
account is active', 'a');
INSERT INTO ALERT_REVIEW.ACCOUNT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES ('Closed', 'The
account is closed', 'c');
INSERT INTO ALERT_REVIEW.ACCOUNT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES
('Inactive','Inactive','i');
INSERT INTO ALERT_REVIEW.ACCOUNT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES
('Dormant','Extended inactivity','d');
INSERT INTO ALERT_REVIEW.ACCOUNT_STATUS (NAME, DESCRIPTION, TYPEDEF) VALUES
('Purge','Purge','x');

INSERT INTO ALERT_REVIEW.REASON_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Overall', 'Overall
score', 'overall');
INSERT INTO ALERT_REVIEW.REASON_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Anomaly', 'Anomaly
score', 'anomaly');
INSERT INTO ALERT_REVIEW.REASON_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Geo', 'Location
score', 'location');
INSERT INTO ALERT_REVIEW.REASON_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Customer', 'Customer
score', 'customer');
INSERT INTO ALERT_REVIEW.REASON_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('Rounding', 'Rounding
rule score', 'rounding');
INSERT INTO ALERT_REVIEW.REASON_TYPE (NAME, DESCRIPTION, TYPEDEF) VALUES ('FPTP', 'False
positive / True positive score', 'fptp');

INSERT INTO ALERT_REVIEW.TRANSACTION_TYPE (NAME, DESCRIPTION, TYPEDEF,IS_CREDIT,IS_DEBIT)
VALUES ('Unknown', 'Unknown', 'unknown','Y','Y');
INSERT INTO ALERT_REVIEW.TRANSACTION_TYPE (NAME, DESCRIPTION, TYPEDEF,IS_CREDIT,IS_DEBIT)
VALUES ('ACH', 'Automated Clearing House', 'ach','Y','Y');
INSERT INTO ALERT_REVIEW.TRANSACTION_TYPE (NAME, DESCRIPTION, TYPEDEF,IS_CREDIT,IS_DEBIT)
VALUES ('ATM', 'Automated Teller Machine', 'atm','Y','Y');
INSERT INTO ALERT_REVIEW.TRANSACTION_TYPE (NAME, DESCRIPTION, TYPEDEF,IS_CREDIT,IS_DEBIT)
VALUES ('Wire', 'Wire Transfer', 'wire','Y','Y');
EOF
```

Run the following command to load the data:

```
kubectl exec -ti $(kubectl get pods | grep insight-db | awk '{print $1}') -- /usr/bin/su -
db2inst1 -c 'db2 connect to amldb; db2 -tvf /aml/amldb/r103.sql'
```

Configuring the analytics

Before you run any of the analytics programs, you must configure the analytics.

The analytics configuration is done in the `AMLAnalyticConfiguration.json` file, which is located in the `/home/spark/code` directory on the gateway server computer.

You can edit the `AMLAnalyticConfiguration.json` file to allow the analytics to be run on different data sources.

Configuration options

The configuration options available in the `AMLAnalyticConfiguration.json` file are:

- `Sparklyr`—defines the Spark connection information
- `CountryType`—the country type that is used in the input files. This value should be one of the following values: `Name`, `ISO2`, `ISO3`, or `ISONumeric`
- `DataDirectory`—the directory where the analytics artifacts are stored. In production, this is a hadoop directory. In a test environment, you can use a local directory.
- `ModelDirectory`—the directory where the analytic models are stored. This directory should not be a hadoop directory. And it must be accessible from each node.
- `WriteResultsToDB`—use this option to run the analytics without writing the results to the database
- `Training`—contains references to the transactions and scored alerts that you want to use to train the model
- `Files`—contains references to the alerts, countries (country risk data), customers, and the transactions that are used in the analytics
- `Rules`—defines the variables and thresholds for the analytic rules
- `AlertProcessing`—defines the parameters that are to be used when the alerts are processed
- `Analytics`—defines the analytics that are to be run

File object parameters

You must define some parameters for each file object that you reference in the `AMLAnalyticConfiguration.json` file. For example, the `Training` and `Training` sections includes different file elements. Each of these file elements can have the following entries:

- `Type`—the record type that the source file contains
- `StoredAs`—contains the following elements:
  - `Name`—the CSV file name or the hive table name
  - `Type`—either `csv` or `hive`
  - `Path`—the directory that contains the CSV file. This value is not used for a hive table name.
  - `Delimiter`—the column separator in the CSV file. The default is a semi-colon (;).
  - Debug—for aggregation output, this option writes the head of the aggregated data to the log
- `Column Definitions`—the column names that are to be used when the source is read. If the source does not contain an equivalent column, set the value to " "

Analytics storage

By default, the intermediate analytics output (aggregates, flags, etc.) is saved to the HDFS file system in compressed format. If you need to manually inspect this data, you must change the default so that the data is stored in uncompressed format.

To output in uncompressed format, add the following line to the `AMLAnalyticConfiguration.json` file in the `Sparklyr` > `config` section:

```
"spark.hadoop.mapred.output.compress": false
```

After you save the changes, you must regenerate the output.

Configuring feature engineering

Feature engineering within the product can be grouped into three types: flags, rules, and aggregators.

There are distinct differences between the usage of the three types:

- Flags use the dataset as the input, and generate a row of feature flags for each row of the input dataset as the output
- Rules use the dataset augmented with the feature flags as the input, and groups the input dataset before creating the features, resulting in one row for each group that is found in the dataset
- Aggregators use the dataset as the input and then groups the input dataset before creating the statistical features, resulting in one row for each group found in the dataset

The features are configured by attaching the feature configurations to the appropriate `File` object in the analytic configuration. The `File` object serves as the input dataset, while the feature configuration defines the features that you want to generate for that `File`. The most common input File for feature engineering is the `Transactions` file, although features can be generated for other files.

Flags

The flag features can be generated for a `File` by adding a `Flags` object to the `File` configuration. The `Flags` object consists of two fields:

- `StoredAs`—the definition of the output file to hold the generated flag features.
- `FlagFeatures`—an array of flag feature definitions to be generated for the `File`.

Each `FlagFeatures` object consists of the following fields (at a minimum) with more fields required depending on the operation:

- `Operation`—the function to execute to generate the flag.
- `Label`—the name of the generated flag in the resulting dataset.

Flag if two values are equal (Operation: flagIfEqual)

Sets the flag to true if the selected fields are equal, false otherwise. There are two possible configurations for this operation:

```
"Columns": ["column1","column2"]
```

Compares the values in column1 and column2. Exactly two column references must be supplied in the `Columns` array for this option.

```
"Columns": ["column1"], "FixedValue": "value"
```

Compares the value in column1 to a fixed value `value`. Only one column reference may be supplied in the `Columns` array for this option.

Flag if value is found in a list (Operation: flagIfFound)

Sets the flag to true if the value in the selected field is found in a list, false otherwise. There are two possible configurations for this operation:

```
"Columns": ["column1"],
"FixedValue": ["value1","value2"]
```

Checks to see if the value in `column1` is present in the fixed value list.

Only one column reference may be supplied in the `Columns` array for this option. The `FixedValue` array must contain one or more elements. The elements can be character (quoted strings) or numeric.

```
"Columns": ["column1"],
"Source": {
    "SourceType": "File",
    "Type": "FileType",
    "AddTimes": false,
    "Columns": "column2"
}
```

Checks to see if the value in `column1` is present in a column read from another file.

Only one column reference may be supplied in the `Columns` array for this option. The `Source` > `Type` entry must reference another `File` defined in the configuration and the `Source` > `Columns` must contain a single column reference from the referenced file.

Flag if value matches a pattern (Operation: flagRegex)

Sets the flag to true if the value matches a regex pattern, false otherwise.

```
"Columns": "column1",
"Regex": "pattern"
```

Checks to see if the value in `column1` matches the pattern defined in `pattern`. Only one column reference may be supplied as the `Columns` value. The `Regex` value must be a valid regex pattern. For more information, see the R documentation (http://stat.ethz.ch/R-manual/R-devel/library/base/html/regex.html).

Flag if a numeric value is in a range (Operation: flagNumCompare)

Sets the flag to true if the value is in a range, false otherwise.

```
"Columns": "column1",
"Inclusive": true/false,
"Min": lowvalue,
"Max": highvalue
```

Checks to see if the value in `column1` is greater than the number in `lowvalue` and less than the number in `highvalue`. If `Inclusive` is set to `true`, the checks become greater than or equal and less than or equal.

Only one column reference may be supplied as the `Columns` value and the column must be numeric. The `lowvalue` can be set to `"-Inf"` (negative infinity) to remove the lower bound or the `highvalue` can be set to `"Inf"` (infinity) to remove the upper bound.

Flag if a date value is in a range (Operation: flagDateCompare)

Sets the flag to true if the value is in a range, false otherwise.

```
"Columns": "column1",
"BaseDate": "YYYY-MM-DD",
"Range": days
```

Checks to see if the date in `column1` is in the range bounded by days `days` before `YYYY-MM-DD` and `YYYY-MM-DD`. If `BaseDate` is set to `""`, the `BaseDate` is set to the current date (based on system time).

Only one column reference may be supplied as the `Columns` value and the column must be a date. The `BaseDate` must be in the specified format `YYYY-MM-DD`. The `Range` value must be an integer value.

Count unique values (Operation: flagCntUniq)

Sets the flag to the count of unique values for the fields

```
"Columns": ["column1","column2"],
"Exclude": ["value1","value2"]
```

Counts the number of unique values in the selected columns (`column1` - `columnX`) excluding any values defined in `Exclude`.

Multiple column references may be supplied in the `Columns` array. The `Exclude` field must be provided, but it can be an empty array (`[]`) if no values should be excluded.

Flag if this record is a "top" record (Operation: flagTopRecords)

Sets the flag to true if, after grouping and sorting, the value is in the top X records, false otherwise.

```
"GroupBy": ["column1"],
"OrderBy": ["column2"],
"Descending": true/false,
"NumRecords": X
```

The input records are grouped by `column1`, sorted by `column2`, then if the record is in the first X records the flag is set to true.

Only one column reference may be supplied in the `GroupBy` array. Only one column reference may be supplied in the `OrderBy` array. `Descending` must be either `true` or `false`. `NumRecords` must be an integer greater than 0.

Flag rounded amounts (Operation: flagIfDivByTenPower)

Sets the flag to true if the value is rounded to a power of ten (such as 1000, 10000, etc.) above a certain threshold, false otherwise.

```
"Columns": ["column1"],
"MinAmount": threshold,
"Digits": X
```

Checks to see if the value in `column1` is greater than the number in `threshold` and is rounded to X number of digits.

Only one column reference may be supplied in the `Columns` array and the column must be numeric. `MinAmount` must be numeric. `Digits` must be an integer greater than 0.

Copy column as flag value (Operation: flagCopy)

Copies the column from the incoming dataset to the flag column.

```
"Column": "column1"
```

Creates a column (named by `Label`) with the contents of `column1`.

Only a single column reference may be supplied as the `Column` value.

Rules
The rules features can be generated for a `File` by adding a `Rules` object to the `File` configuration.

The `Rules` object consists of four fields:

- `Operation`—the function to execute to generate the rule feature
- `Label`—the name of the generated rule feature in the resulting dataset

Sum based on conditions (Operation: featureSum)

After aggregating by the `AggregateBy` columns, sums the column specified in the `Column` property for rows that meet the conditions set in the corresponding `Flags`/`FlagValues` arrays:

```
"Column": "column1"
"Flags": ["flag1",...],
"FlagValues":["TRUE/FALSE",..]
```

Only one column reference may be supplied in the `Column` property and the referenced column must be numeric.

Count based on conditions (Operation: featureCnt)

After aggregating by the `AggregateBy` columns, counts the number of rows that meet the conditions set in the corresponding `Flags`/`FlagValues` arrays:

```
"Flags": ["flag1",...],
"FlagValues":["TRUE/FALSE",..]
```

Aggregators
The aggregates can be generated for a `File` by adding an `Aggregators` object to the `File` configuration.

The `Aggregators` object consists of four required fields with two optional fields:

- `AggregateName`—the reference name of the aggregator
- `AggregateBy`—one or more columns used to group the data, values must be keys referencing columns in the parent file
- `StoredAs`—the definition of the output file to hold the generated aggregates
- `AggregatedFeatures`—an array of aggregate feature definitions to be generated for the `File`
- `GII` (optional)—if present, this object will trigger the generation of the General Interestingness Index (GII) for the aggregate features
- `JoinFeatures` (optional)—if present, an array of join feature definitions to be generated for the `File`

The GII element consists of the following fields:

- `Weights`—the weights that are applied when calculating the GII
  - FPS1—the weight for the ratio of missing to valid records
  - FPS3—the weight for the number of outliers

  - FPS4—the weight for Skewness
  - FPS5—the weight for Standard Deviation
  - FPS6—the weight for Kurtosis

  Weights are automatically standardized to sum to 1.
- `StoredAs`—the definition of the output file to hold the generated GII values

AggregateFeatures

Each `AggregateFeature` object consists of the following fields (at a minimum) with additional fields required depending on the operation:

- `Operation`—the function to execute to generate the aggregate feature
- `BaseLabel`—the root for the resulting column names

The following aggregate features are available. Each of the operations are performed after the data is grouped by the `AggregateBy` columns. Valid `TimeRange` values are `Day`, `Week`, `Month`, `Quarter`, or `Year`.

Statistics by date range (Operation: summarizeByStats)

For the specified column and timerange(s), calculate the following statistics: min, max, mean. Each combination of a timerange and a statistic will result in a feature in the output.

```
"Column": "column1"
"TimeRanges": ["Day","Week",...]
```

Only one column reference may be supplied in the `Column` property and the referenced column must be numeric. The label for each feature will consist of the `BaseLabel` appended with the statistic and the timerange.

Statistics by count over date range (Operation: summarizeByCount)

For the specified timerange(s), count the number of occurrences and calculate the following statistics: min, max, mean, variance and percentile. Each combination of a timerange and a statistic will result in a feature in the output.

```
"TimeRanges": ["Day","Week",...]
```

The label for each feature will consist of the `BaseLabel` appended with the statistic and the timerange.

Statistics by grouped count over date range (Operation: summarizeByCountGrouped)

For the specified timerange(s) and secondary grouping, count the number of groups and calculate the following statistics: min, max, and mean. Each combination of a timerange, grouping column and a statistic will result in a feature in the output.

```
"GroupingColumns": ["column1", ...]
"TimeRanges": ["Day","Week",...]
```

`GroupingColumns` are one or more secondary columns to be grouped and counted, must be the identifier of one of the columns in the parent file. The label for each feature will consist of the `BaseLabel` appended with the statistic, grouping column and the timerange.

Statistics by unique count over date range (Operation: summarizeByUniqueCount)

For the specified timerange(s) and column, count the number of unique values and calculate the following statistics: min, max, and mean. Each combination of a timerange and a statistic will result in a feature in the output.

```
"CountColumn": "column1"
"TimeRanges": ["Day","Week",...]
"CountHighRisk": true/false,
"Source": {
    "SourceType": "File",
    "Type": "HighRiskCountries",
    "Columns": "IDColumn"
},
"filters": [
    {
        "filterColumn": "column2",
        "filterValue": ["matchValue",...],
        "filterMatch": "include"/"exclude"
    }
]
```

`CountColumn` is a column with distinct values and must be the identifier of one of the columns in the parent file.

If `countHighRisk` is `true`, then the `Source` object is required. The values in the source are used as a filter on the `CountColumn`. Only values contained within the source object are counted.

If `countHighRisk` is `false`, then the `Source` object can be omitted.

Other columns can be used to filter the data prior to counting. Filtering is optional. If a filter is defined, the `filterColumn` must be the identifier of one of the columns in the parent file. `filterValue` is an array of values to be matched in the filter and `filterMatch` must be set to `include` to leave only the rows with values in `filterValue`, while `exclude` will remove all rows with values in `filterValue`. The label for each feature consists of the `BaseLabel` appended with the timerange and the statistic.

Statistics by frequency (Operation: summarizeByFrequency)

Derive the time between occurrences and calculate the following statistics: min, max, and mean. Each statistic will result in a feature in the output.

```
"filters": [
    {
        "filterColumn": "column2",
        "filterValue": ["matchValue",...],
        "filterMatch": "include"/"exclude"
    }
]
```

Filtering is optional. If a filter is defined, the `filterColumn` must be the identifier of one of the columns in the parent file. `filterValue` is an array of values to be matched in the filter and `filterMatch` must be set to `include` to leave only the rows with values in `filterValue`, while `exclude` removes all rows with values in `filterValue`. The label for each feature consists of the `BaseLabel` appended with the statistic.

Statistics of variance (Operation: summarizeByVariance)

For the specified timerange(s) and column, calculate the variance and then the following statistics: min, max, and mean. Each combination of a timerange and a statistic will result in a feature in the output.

```
"Column": "column1",
"TimeRanges": ["Week","Month"]
```

The value in `Column` must be an identifier of one of the columns in the parent file and the referenced column must be numeric. The label for each feature will consist of the `BaseLabel` appended with the statistic.

Summarize by percentage (Operation: summarizeByPercentage)

For the specified column and category, calculate the percentage of records that match the value defined in the category field. Optionally filter the records prior to calculating the percentage.

```
"Column": "column1",
"Category": "matchValue",
"filters": [
    {
        "filterColumn": "column2",
        "filterValue": ["matchValue2",...],
        "filterMatch": "include"/"exclude"
    }
]
```

`Column` must be the identifier of one of the columns in the parent file. `Category` contains the value to be matched in `Column`. Additional columns can be used to filter the data prior to counting, although filtering is optional.

If a filter is defined, the `filterColumn` must be the identifier of one of the columns in the parent file. `filterValue` is an array of values to be matched in the filter and `filterMatch` must be set to `include` to leave only the rows with values in `filterValue`, while `exclude` will remove all rows with values in `filterValue`. The label for the feature will be the `BaseLabel`.

Summarize by risk count (Operation: summarizeByRiskCount)

Count the total number of records involved with a risky country (as defined in the `HighRiskCountries` file type).

```
"Columns": ["countryColumn1","countryColumn2",...]
```

Each value in `Columns` must be an identifier of one of the columns in the parent file. The label for the feature will be the `BaseLabel`.

JoinFeatures

Each `JoinFeature` object currently supports one operation: `copyExternalColumn`.

Each operation is performed after the data is grouped by the `AggregateBy` columns and the specified column will be added to the aggregated data.

Copy a column (Operation: copyExternalColumn)

Copies a column from another datasource into the aggregated data.

```
"Label": "columnLabel"
"Source": [
    {
        "Type": "fileType",
        "Column": "column1",
        "JoinColumns": ["columnA",...]
    }
]
```

`Source` specifies where the column data is obtained. Multiple sources can be used. If multiple sources are specified, they will be combined (rowwise) before joining to the aggregate data.

Type references a file record type (for example, Individuals) defined elsewhere in the configuration.

Column is the reference to the column to be copied from the datasource

JoinColumns are the keys for the columns in the Source used to join the data to the aggregate data. The column definitions must correspond to the columns defined as the AggregateBy columns.

Configuring the analytics execution

The analytics that are executed, and the aggregation, are configured in the Analytics array. The Analytics array can contain one or more analytic configuration objects.

The following fields are common across all analytic types. More configuration fields are required for most analytic types.

- Type—the type of analytic to be executed, the default analytics are AnomalyClustering, Univariate, SupervisedML, and CountryRisk.
- Source— the R file sourced to run the analytics. The R file should contain the BandingFunction, AssessmentFunction, and ArtifactsFunction, if they are defined.
- Results—defines how the analytic results should be handled:
  - Weight—the weighting of this analytic within the overall analytic score. The weights of all analytics are automatically standardized to sum to 1.
  - ReasonType—the reason type for this analytic. This is the primary identifier for the analytic instance and it should be unique across all analytics that are defined within the configuration. This value must also be defined as a reason type NAME in the ALERT_REVIEW.REASON_TYPE table.
  - ScoreColumn—the column within the results that contains the score. The analytics write to this column and the alert processing reads the data in this column for alert scoring.
  - InsightColumn—the column within the results that contains the insights. The analytics write the insights to this column (in JSON format. The alert processing passes the data in this column to display in the FCAI user interface.
- BandingFunction—the function that is used to calculate the scores over the full set of input data. The function provides the banding ranges for this analytic instance.
- AssessmentFunction—the function to be called to score a set of data. This function returns the data with the ScoreColumn and InsightColumn populated.
- ArtifactsFunction—the function to be called to return any artifacts that should be stored with the model audit artifacts after processing.
- InputSource—One or more data sources that are used as input for the analytics
  - SourceType—can be Aggregate, File, or Function:
    - Aggregate contains the following fields:
      - Type—the file type for the aggregate, which is defined within a Files entry in the configuration
      - AggregateName—the aggregate within the file type
      - Thrd_GII—the GII threshold, This is an optional value that should be in the range 0 - 1. If it is specified, the input data is filtered so that only features with a GII greater than the threshold are included in the analytic processing.
    - File contains the following fields:
      - Type—the file type, which is defined in a Files entry in the configuration
      - AddTimes—the calculate and append time ranges to the data. This value is optional.
      - Columns—a subset of columns from the data to be used in the analytics. This value is optional.
    - Function contains the following fields:
      - Source—the R file that is sourced. The R file contains the function.
      - FunctionName—the function to run to retrieve the source data.

Parameters for the default analytics are described in the following sections.

AnomalyClustering

- Thrd_Crlt—the threshold of the correlation cutoff
- Thrd_GII—the GII threshold. This value should be 0 - 1. Only features with a GII that is greater than the threshold are included in the analytic processing.
- minClust—the minimum number of clusters. The default is 10.
- maxClust—the maximum number of clusters. The default is 20.
- optClust—the user-defined optimal number of clusters. The default is 0.

CountryRisk

There are no extra parameters for CountryRisk.

SupervisedML

- FP_Training_only—to do false positive training only, rather than training and validation
- TP_Training_only—to do true positive training only, rather than training and validation
- Debug—to output partial intermediate values to the log
- InputSource—the input source for supervised machine learning can be complex, with multiple inputs joined into a single input source. As a result, the InputSource for supervisedML can be an array if you are using multiple input sources, or a simple object similar to the other analytics if you are using a single source. If you are using multiple sources, the first array entry is the primary source and it should contain all of the ID columns that are required to join the additional sources. The input sources require more parameters to allow the joins to be done:
  - InputSource[1]
    - CountryColumns—an array of country columns within this source so that country risk can be included in the supervisedML
    - IDColumns—an array of ID columns
  - InputSource[2]
    - JoinColumnsX—an array of the column IDs from InputSource[1] to use in the join
    - JoinColumnsY—an array of the column IDs from this InputSource to use in the join
- TagSource—an extra input source that contains tagged results. The source is usually a SourceType of File with the corresponding parameters (Type, AddTimes, Columns). It requires the following parameters:
  - TagColumn—the column containing the tagged results (such as true positive)
  - TagTrueValues—an array of values that represent true positives
  - JoinColumnsX—an array of the column IDs from InputSource[1] to be used in the join
  - JoinColumnsY—an array of the column IDs from the tag source to use in the join

Univariate

Note: The Univariate analysis can be executed only on an input source that has a SourceType of Aggregate.

- `ScoringCurve`—The curve type to be used for scoring the data. The value can be `Growth` or `Decline`.
- `GroupingSource`—An extra input source that contains a column to allow grouping of the InputSource data. In addition to the normal InputSource parameters, GroupingSource requires the following parameters:
  - `IDColumn`—The column to in the join with the InputSource. This value should correspond to the `AggregateBy` column specified for the aggregate.
  - `PeerGroupColumn`—The column that contains the grouping information.

Run the analytics

To run the analytics you must load the data, train the model, and then process the transactions and alerts.

Ingesting the party and non-party data

There are two types of data that are used in IBM Financial Crimes Alerts Insight with Watson (FCAI): party data and non-party data.

Party data is loaded into IBM FCAI from the Common Entity Data Model (CEDM) by using the bulk loading functions of the IBM Financial Crimes Insight with Watson core components.

- Party data

  Party data includes the following:

  - Individuals
  - Organization
  - Address
  - Identifications
  - Relationships
  - Resolved Entities
  - GDPR-related delete messages

  For more information about bulk loading, see Importing data into the FCI system (https://www.ibm.com/support/knowledgecenter/SSCKRH_1.0.3/platform/c_getting_data_into_icfm_data.html).
- Non-party data

  Non-party data is loaded from CSV files. The CSV files are placed in the `/amldata` directory, and the files are then processed by IBM FCAI.

  Non-party data includes the following:

  - Transactions
  - Account
  - Account mapping
  - Alerts
  - Alert mapping

Sample non-party data is provided with IBM FCAI. Use the following steps to load the non-party data.

Procedure

1. Log on to the gateway node computer.
2. Change to the flume user.
   For example, enter `su - flume`

3. As the flume user, run the following command to load the sample data from CSV files:

```
hdfs dfs -copyFromLocal -f /opt/ibm/aml/sample/*.csv /aml/data/
```

4. As the flume user, copy the non-party data CSV files to the `/amldata` directory.

   Note: This step should be repeated each time that new data is available for ingesting into IBM FCAI.

   Sample non-party data CSV files are provided. If you are ingesting the sample data you can use the following commands to copy the non-party data sample CSV files to the `/amldata` directory.

   If you are ingesting non-sample data, ensure that you replace the sample directory and filename with the full path and name of the corresponding data file.

```
cp /opt/ibm/aml/sample/AML_sample_accounts.csv /amldata/accounts-$(date +%s).csv
```

```
cp /opt/ibm/aml/sample/AML_sample_accountMapping.csv /amldata/accountmappings-$(date +%s).csv
```

```
cp /opt/ibm/aml/sample/AML_sample_alertTxnMapping.csv /amldata/alerttxmappings-$(date +
%s).csv
```

```
cp /opt/ibm/aml/sample/AML_sample_transactions.csv /amldata/transactions-$(date +%s).csv
```

```
cp /opt/ibm/aml/sample/AML_sample_alert.csv /amldata/alerts-$(date +%s).csv
```

Training the model

There are two model training tasks. One for the supervised machine learning models and one for the customer clustering model.

You generate the supervised machine learning models, False Positive and True Positive, by training on an existing set of scored alerts and transactions.

Procedure

1. Log on to the gateway node computer.
2. Change to the flume user.
   For example, enter `su - flume`
3. Run the following R client command to create the supervised machine learning models:

```
/opt/ibm/aml/bin/merge_compact_and_rscript.sh --user=ambari_username --pass=ambari_password
--port=8081 --r_args=AML_supervisedML_FPTP_batch.R 2>&1 | tee supervised.log
```

   You can run `/opt/ibm/aml/bin/merge_compact_and_rscript.sh -h` or `/opt/ibm/aml/bin/merge_compact_and_rscript.sh --help` to see more details about running the script.

   The transaction file and the scored alerts file are read from the `AMLAnalyticConfiguration.json` file.
4. Run the following R client command to train the Customer clustering model:

```
/opt/ibm/aml/bin/merge_compact_and_rscript.sh --user=ambari_username --pass=ambari_password
--port=8081 --r_args=FCAI_AnomalyClustering_batch.R 2>&1 | tee clustering.log
```

   The transaction file is read from the `AMLAnalyticConfiguration.json` file in the `Training, Files` entry.

Scheduling transaction processing

After you install IBM Financial Crimes Alerts Insight with Watson (FCAI), you must schedule the daily transaction processing job.

You can schedule the daily transaction processing job in the flume user's crontab by using the `make_flume_crontab.sh` script, which is available in the `/opt/ibm/aml/bin/` directory on the gateway node computer.

Note: In a test or demonstration environment, you can run the transaction processing script manually rather than scheduling the job to be run.

To run the script manually, do the following:

1. Change to the flume user. For example, `su - flume`
2. Go to the `/home/spark/code/` directory.
3. Run the following command:

```
/opt/ibm/aml/bin/merge_compact_and_rscript.sh --user=ambari_username --pass=ambari_password
--port=ambari_port --r_args=AML_process_transactions.R 2>&1 | tee processtran_debug.log
```

Procedure

1. Log in to the gateway node computer as the flume user.
2. Go to the `/opt/ibm/aml/bin/` directory.
3. Run the following command:

```
./make_flume_crontab.sh -h ambari_host -p ambari_port -u ambari_username -s ambari_password -
H processing_hour -M processing_minute
```

For example:

```
./make_flume_crontab.sh -h ambari_server.domain.com -p 8081 -u admin -s admin -H 12 -M 00
```

Ingesting alerts

Alert data is ingested by copying the files to the `/amldata` directory and then using the `ingestAlerts.sh` script.

Alerts are ingested in the same manner as other non-party data. For more information , see “Ingesting the party and non-party data” on page 21.

However, the alerts are processed differently than the other non-party data types. In addition to copying the data into the Hive table, the alert file is copied to the `hdfs:///aml/data/YY-MM-DD` directory, where `YY-MM-DD` is the current date. This file is then split and copied into smaller files in the same directory. Each split file has up to 1000 lines and is named with an extension such as `.aaa`, `.aab`, etc.

The size of the split file can be adjusted in the Flume configuration as one of the parameters to the call to `preprocessAlerts.sh`.

After the alerts file has been ingested through Flume, the temporary tables must be merged into the permanent tables by using the `hive_merge.sh`, and the alerts processed by using the `ingestAlerts.sh` script.

`ingestAlerts.sh` does the following steps:

- compacts the permanent tables
- runs the feature engineering for the current data
- looks in the directories for yesterday's and today's date and processes any split files (extensions `.aaa`, etc)
  - if the split file is processed successfully, it is deleted
  - if the split file fails during processing, it is renamed with an extension `.failed`

After you determine the cause of the failure, the file can be retried by removing the `.failed` extension and you can run `ingestAlerts.sh` again.

Procedure

1. Log on to the gateway node computer.
2. Change to the flume user.
   For example, enter `su - flume`
3. Copy the alert data CSV files to the `/amldata` directory.

   For example,

   ```
   cp /opt/ibm/aml/sample/AML_sample_alert.csv /amldata/alerts-$(date +%s).csv
   ```

4. Run the `hive_merge.sh` script.

   ```
   /opt/ibm/aml/bin/hive_merge.sh --host=ambari.host.com --user=ambari_username --
   pass=ambari_password --port=8081
   ```

   You can run `/opt/ibm/aml/bin/hive_merge.sh -h` or `/opt/ibm/aml/bin/hive_merge.sh --help` to see more details about running the script.
5. Run the `ingestAlerts.sh` script:

   ```
   /opt/ibm/aml/bin/ingestAlerts.sh --user=ambari_username --pass=ambari_password --port=8081
   ```

   You can run `/opt/ibm/aml/bin/ingestAlerts.sh -h` or `/opt/ibm/aml/bin/ingestAlerts.sh --help` to see more details about running the script.

   If you schedule `ingestAlerts.sh` to run by using cron, you must also run `hive_merge.sh` before you run `ingestAlerts.sh`.

Viewing processed transaction artifacts

You can capture the artifacts (such as the configuration, models) that you used when you processed transactions by running `AML_process_transactions.R`. The artifacts are saved to a single model audit artifacts file to allow you to review the conditions that were in place when you processed the transactions.

Saved contents

The model audit artifacts file is stored in the `DataDirectory` that you defined in the `AMLAnalyticConfiguration.json` configuration file. The model audit artifacts file is named `ModelAuditArtifacts`*`yyyy-MM-dd_hh.mm.ss`*`.rda`, where *`yyyy-MM-dd_hh.mm.ss`* is the timestamp of when the file was generated.

The model audit artifacts file contains R objects, which are determined by the `AMLAnalyticConfiguration.json` configuration file. At a minimum, the file contains any GII files that are defined in the analytic configuration file. The analytics that you defined in the analytic configuration file can also produce artifacts that are stored in the file. These objects are retrieved from each analytic for which an ArtifactsFunction is defined.

Viewing the artifacts

You can view the contents of a model audit artifacts file by opening the file in an R session.

1. From the analytics directory (`/home/spark/code` on the gateway computer), start R.

2. Enter the following commands in the R prompt:

```
source("AML_utils.R")
```

```
readAnalyticConfig()
```

```
loadModelArtifacts()
```

The latest model audit artifacts file is loaded into memory and the objects that are contained in the file are displayed.

To load a model audit artifacts file other than the latest, you must include the file name of the model audit artifacts file in the loadModelArtifacts() method. For example,

```
loadModelArtifacts("ModelAuditArtifacts2018-07-25_09.25.29.rda"
```

Example

Each object that is displayed in the console output includes a header that shows the R object name and a description in parentheses. For example:

```
Reading model from file ../sample_data/ModelAuditArtifacts2018-07-25_09.25.29.rda

Loaded model artifacts from ModelAuditArtifacts2018-07-25_09.25.29.rda


*** configuration (AMLAnalyticConfig.json) ***

{
  "Sparklyr": {
    "master": ["local"],
    "version": ["2.2.0"],
    "sparkHome": [""],
    "config": {
      "sparklyr.shell.driver-memory": ["8G"]
    }
  },
  "CountryType": ["ISO3"],
  "DataDirectory": ["../sample_data"],
  "ModelDirectory": ["../sample_data"],
  "IGAMessageDirectory": [""],
  "WriteResultsToDB": [false],
  "Training": {
    "Files": [
      {
        "Type": ["Transactions"],
        "StoredAs": {
          "Name": ["AML_sample_transactions.csv"],
          "Type": ["csv"],
          "Path": ["../sample_data"]
        }
      },
      {
        "Type": ["Alerts"],
        "StoredAs": {
          "Name": ["AML_sample_alert_scored.csv"],
          "Type": ["csv"],
          "Path": ["../sample_data"]
        },
        "FP_TP_TagColumn": ["fp_tp_tag"]
      }
    ]
  },
  "Files": [
    {
      "Type": ["Accounts"],
      "StoredAs": {
        "Name": ["AML_sample_accounts.csv"],
        "Type": ["csv"],
        "Path": ["../sample_data"]
      },
      "IDColum ...
```

```
*** res_Anomaly (NBClust Model, ReasonType: Anomaly) ***

[1] 12


*** fit_TP.rf_FPTP (True Positive Model, ReasonType: FPTP) ***

Random Forest

3080 samples
  42 predictor
   2 classes: 'NotTP', 'TP'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 2463, 2464, 2465, 2464, 2464
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.9892862  0.9754705
  22    0.9857137  0.9671451
  43    0.9857148  0.9672142

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 2.


*** fit_FP.rf_FPTP (False Positive Model, ReasonType: FPTP) ***

Random Forest

3080 samples
  42 predictor
   2 classes: 'FP', 'NotFP'

No pre-processing
Resampling: Cross-Validated (5 fold)
Summary of sample sizes: 2463, 2464, 2465, 2464, 2464
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
   2    0.9892862  0.9754705
  22    0.9857137  0.9671451
  43    0.9857148  0.9672142

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 2.


*** CountryRisk_Geo (Country Risk Scores) ***

# A tibble: 280 x 9
   GeoID Country        ISO2  ISO3  ISONumeric RiskScore RiskRank RiskRating TypeDef
   <int> <chr>          <chr> <chr>      <int>     <dbl>    <int> <chr>      <chr>
 1     1 Afghanistan    AF    AFG            4     21.1         4 High       afghanistan
 2     2 Akrotiri       <NA>  <NA>          NA      1.05      177 Low        akrotiri
 3     3 Albania        AL    ALB            8      7.80       63 High       albania
 4   269 Alderney       <NA>  <NA>          NA      1.05      177 Low        alderney
 5     4 Algeria        DZ    DZA           12      4.35      107 Medium     algeria
 6     5 American Samoa AS    ASM           16      1.00      221 Low        american_samoa
 7     6 Andorra        AD    AND           20      1.10      175 Low        andorra
 8     7 Angola         AO    AGO           24      6.64       71 High       angola
 9     8 Anguilla       AI    AIA          660      3.69      123 Medium     anguilla
10     9 Antarctica     AQ    ATA           10      0.        272 Low        antarctica
# ... with 270 more rows


*** Transactions_ByBeneficiary_GII (General Interestingness Index) ***

# A tibble: 112 x 2
   Feature               GII
   <chr>               <dbl>
 1 TotalAmt_Year_Max   0.817
 2 TotalAmt_Year_Min   0.814
 3 TotalAmt_Year_Avg   0.814
 4 TotalAmt_Year_Med   0.814
 5 TotalAmt_Quarter_Max 0.813
 6 TotalAmt_Month_Max  0.812
 7 TotalAmt_Day_Max    0.812
 8 TotalAmt_Week_Max   0.812
```

```
 9 TotalAmt_Quarter_Min 0.811
10 TotalAmt_Quarter_Med 0.811
# ... with 102 more rows


*** Transactions_ByBeneficiaryTransactionType_GII (General Interestingness Index) ***

# A tibble: 80 x 2
   Feature               GII
   <chr>               <dbl>
 1 TotalAmt_Year_Max    0.815
 2 TotalAmt_Year_Min    0.813
 3 TotalAmt_Year_Avg    0.813
 4 TotalAmt_Year_Med    0.813
 5 TotalAmt_Quarter_Max 0.812
 6 TotalAmt_Month_Max   0.811
 7 TotalAmt_Week_Max    0.811
 8 TotalAmt_Day_Max     0.811
 9 TxAmt_Day_Max        0.811
10 TxAmt_Week_Max       0.811
# ... with 70 more rows


*** JoinedAlerts_ByAlertID_GII (General Interestingness Index) ***

# A tibble: 2 x 2
  Feature                GII
  <chr>                <dbl>
1 TxRiskCntryCnt       0.157
2 TxPercentForeign_TRUE 0.150
```

Enable alert messaging

To send alert messages from IBM Financial Crimes Alerts Insight with Watson (FCAI) to the IBM Financial Crimes Insight with Watson platform components, you must edit the `AML_platform_connection.json` file.

`AML_platform_connection.json` is in the `/home/spark/code` directory on the gateway computer.

IBM FCAI

The `AML_platform_connection.json` file's default configuration for IBM FCAI messaging is as follows:

```
{
    "sendMessage": false,
    "platformHost": "https://<hostname_kubernetes_master>:9443/ibm/fci/platform/
external_assessment/analysis_result?waitTime=15",
    "platformUser": "<platform_user>",
    "platformPassword": "<platform_password>",
    "sslValidationEnabled": true,
    "tokenURL": "https://<hostname_kubernetes_master>:3000/security-auth/api/v1.0/login/ldap?",
    "remoteSystemRef": "FCAI_ALERT",
    "context": "ai_fraudcontext",
    "resultProperties": {
        "createPriority": "PriorityColumn",
        "cf.caseprop.status": "StatusColumn"
    }
}
```

<hostname_kubernetes_master> is the fully qualified domain name of the platform server that receives the messages.

Port 9443 is the default fci-analytics pod port number.

sendMessage
Set to true to enable sending messages to the platform. Set to false and no messages are sent.

platformHost
Replace <hostname> with the fully qualified domain name of the platform server that receives the messages.

platformUser
Enter the user name that is required to access the platform UI.

platformPassword
Enter the password for the user that is required to access the platform UI.

sslValidationEnabled
Set to true to enable SSL validation. Set to false to skip validation.

tokenURL
Replace <hostname> with the FQDN of the platform server that receives the messages.

remoteSystemRef
This value must be set to the Reference ID parameter when you register the analysis flow. The default is FCAI_ALERT.

context
The context that is used to display the alert messages in the case manager. The context name must be selected when you register the analysis flow.

resultProperties
Additional properties that are displayed in the alert in the case manager, such as priority and status. Properties can be removed and added, depending on user preferences.

After the file is edited and sendMessage set to true, any time alert ingestion is run, alert messages will automatically be generated and sent to the platform.

Alert messaging content

For each alert, all of the transactions and the parties involved with each transaction are sent to the case manager.

Transactions

The following items are included in a transaction block:

- The Transaction ID
- The value of the currency involved
- The transaction type (wire, ATM)
- The timestamp of the transaction
- The remote system name where the transaction came from, such as the internal bank data system

Parties

There are two party types that can be sent through alert messaging: Individual and Organization.

Individuals

The following items are included in a party block for an individual:

- The party's ID
- The party type
- The remote system name, such as the internal bank data system

Organizations

The following items are included in a party block for an organization:

- The party's ID
- The party type
- The remote system name, such as the internal bank data system

In addition to the transaction and party blocks, the beginning of each alert consists of several important components:

- The registered name of the analysis flow. This is defaulted to FCAI_ALERT but this needs to be registered on the platform side before the platform recognizes incoming alerts with this name

  You can change the analysis flow name in the `AML_platform_connection.json` file.
- The alert ID
- The score of the alert
- The timestamp of when the alert was generated

All information is sent to the platform in XML format.

Configuring alert messaging

More setup of the IBM Financial Crimes Insight with Watson components is required before alert messaging can be successfully sent to the Case Manager.

Loading the data into the Case Manager database

Data must be loaded into the Case Manager database before any alert messaging can occur. Data is loaded by using the bulk loading features of the IBM Financial Crimes Insight with Watson core components.

If data is not loaded into the database, then you will see errors such as:

```
COVCF3026E: The Party object is not valid. It is missing the data for either an Individual or
an Organization.
```

This error can mean that there is nothing in Case Manager to reference the data that you are sending, or you are sending data that does not match with what is loaded into Case Manager.

If you do not have your own data, two sample data files are provided for testing the installation, `AML_sample_individuals-bulkload.csv` and `AML_sample_organizations-bulkload.csv`. These files define the various party attributes of all of the individual and organization parties that are involved in the sample alerts.

Selecting or creating a context

When you log in to the Case Manager and click an alert that was sent from Alerts Insight, the bottom half of the page display cards for each object that is in the alert, such as the parties and transactions.

You can configure Case Manager to display the Alerts Insight application interface, rather than the cards. To do this you must create a new context and run some SQL commands to modify the Case Manager database.

The SQL script that you run looks like the following:

```
insert into "CFFACT"."FRAUD_ASSESSMENT_CONTEXT" ("NAME", "DESCRIPTION", "STEREOTYPE",
"INITIAL_STATE") values ('AML', 'Alert  AML Fraud Context', 'ai_fraudcontext',
'alert_triage_review');
```

The "STEREOTYPE" parameter in the SQL command must also be set as the "context" parameter in the `AML_platform_connection.json` file.

For example, in `AML_platform_connection.json`, it appears as:

```
"context": "ai_fraudcontext"
```

The alert message will not send successfully if the two values do not match.

The current default value of "context" in the AML_platform_connection.json file is 'ai_fraudcontext'. You can change the value or use the default value for the "STEREOTYPE" parameter.

The "Name" and "Description" fields are displayed in the Case Manager. These values do not have default values. For example, you might set the "Name" value to "AML" and the "Description" value to "Alert AML Fraud Context".

To allow for an analysis flow to be properly approved with a custom context, you must map some assessment values to the context. You map the values by using the following SQL commands:

```
insert into "CFFACT"."CONTEXT_ASSESSMENT_VALUES" ("CONTEXT_ID", "MIN_SCORE",
"FRAUD_ASSESSMENT_VALUE_ID") values(
(SELECT CONTEXT_ID   FROM CFFACT.FRAUD_ASSESSMENT_CONTEXT  WHERE STEREOTYPE =
'ai_fraudcontext'), 000 ,
(SELECT FRAUD_ASSESSMENT_VALUE_ID  FROM CFFACT.FRAUD_ASSESSMENT_VALUE   WHERE STEREOTYPE =
'low'));
```

```
insert into "CFFACT"."CONTEXT_ASSESSMENT_VALUES" ("CONTEXT_ID", "MIN_SCORE",
"FRAUD_ASSESSMENT_VALUE_ID") values(
(SELECT CONTEXT_ID   FROM CFFACT.FRAUD_ASSESSMENT_CONTEXT  WHERE STEREOTYPE =
'ai_fraudcontext'), 400 ,
(SELECT FRAUD_ASSESSMENT_VALUE_ID  FROM CFFACT.FRAUD_ASSESSMENT_VALUE   WHERE STEREOTYPE =
'medium'));
```

```
insert into "CFFACT"."CONTEXT_ASSESSMENT_VALUES" ("CONTEXT_ID", "MIN_SCORE",
"FRAUD_ASSESSMENT_VALUE_ID") values(
(SELECT CONTEXT_ID   FROM CFFACT.FRAUD_ASSESSMENT_CONTEXT  WHERE STEREOTYPE =
'ai_fraudcontext'), 800 ,
(SELECT FRAUD_ASSESSMENT_VALUE_ID  FROM CFFACT.FRAUD_ASSESSMENT_VALUE   WHERE STEREOTYPE =
'high'));
```

If you do not map these values, you can get the following error in Case Manager when you try to save your analysis flow:

```
"COVCF0094E: No Fraud Assessment Values were mapped to a Fraud Assessment Context. The analysis
flow cannot be approved"
```

Ensure that the value that you use for the CONTEXT_ID stereotype value in the script matches the stereotype value from the CFFACT.FRAUD_ASSESSMENT_CONTEXT table when you created the context. For example, 'ai_fraudcontext'. If they do not match, then the values will not map properly.

To enable the fragment to appear in the Case Manager interface, you must run the SQL script:

```
update CFCONFIG.SYSTEM_PROPS set VALUE='["ai_fraudcontext"]' where
STEREOTYPE='CF.UI.AI.FragmentList';
update CFCONFIG.SYSTEM_PROPS set VALUE='true' where STEREOTYPE='CF.UI.AI.FragmentEnable';
```

The CFCONFIG.SYSTEM_PROPS value is set to 'ai_fraudcontext', which matches the example. The value must match what was set as the CFFACT.FRAUD_ASSESSMENT_CONTEXT stereotype, otherwise there will be a mismatch when alerts are sent.

Creating an analysis flow

Analysis flows can be quite customizable depending on how you want the alert to appear in the Case Manager. However, some values are required for alert messaging to work.

Go to the CM UI and go to the "Analysis Flow" menu. Click the purple "Register" button and a menu will pop up with fields that you need to fill in.

1. Open Case Manager as an admin user.
2. In the **Analysis Flow** view, click **Register**.
3. Enter the following values:
   - In **Type**, select **Monitored Analytic**.

- In **Name**, enter a name for the flow. For example, enter `FCAI Alerts Ingestion flow`.
- In **Description**, enter a description. For example, enter `AML Monitored investigation`.
- In **Deployment Version**, enter a version for the flow. For example, enter `1.0.0` for an initial version.
- In **Context**, select `Default Context` to view the information as cards. If you want to view the Alerts Insight interface, enter the name of the context that you created. For example, select `AML`.
- For **Reference ID**, enter the "remoteSystemRef" value from the `AML_platform_connection.json`. For example, "remoteSystemRef": "FCAI_ALERT". The default value is "FCAI_ALERT". You can use any value you like, but the reference ID must match the parameter in the `AML_platform_connection.json` file.

4. Click **OK**.
5. In the left pane, select the flow that you registered, and click **Assessment Actions**.
6. Click **Add**.
   - Click the **Assessment Value** column, and enter `Very High`.
   - Click the **Action Type** column, and enter an action, such as `Create or Update Investigation`.
   - Click the **Action Stereotype** column, and enter `createName`. This sends the alerts to the **Undefined** basket in Case Manager.
   - Click the **Action Value** column, and enter an action value, such as `AML Very High`.
7. Repeat the above step and enter the following values:

Table 1: Case Manager actions

Assessment Value	Action Type	Action Stereotype	Action Value
High	Create or Update Investigation	createName	AML High
Medium	Create or Update Investigation	createName	AML Medium
Low	Create or Update Investigation	createName	AML Low

   createName is an example of an **Action Stereotype** value.
8. Click **Manage Values**.
9. Ensure that **isDetection** is set to **on** for each action, and click **Save**.
10. Click **Initialization Criteria**, select **All Account Lookup**, and click **OK**.
11. Click **Set Lifecycle State**, and select **Approved**.
12. Click **Save**.

Referencing external system names

Part of the message that is sent to the Case Manager includes the system name of wherever the object (such as party or transaction) came from. This system name must be added to the Case Manager database in order for the message to be properly received. You add the system name by using the following SQL script:

```
INSERT INTO CFFACT.SYSTEM_REF(SYSTEM_ID,SYSTEM_TYPE_ID,DESCRIPTION,STEREOTYPE) VALUES
(1,3,'Systemref used by AML sample', <system name here>);
```

Replace the STEREOTYPE value with the system name. If you do not know the system name, you can submit an alert message without running this SQL script and the error message will display the system name.

Additional stereotypes

If you send an alert and a message indicates that something is missing, you can find that table in **Code Tables** in Case Manager and add the value or write an SQL script to add the value if the table does not appear **Code Tables**.

Integrate entity resolution with the FCI platform

To configure FCAI integration with FCI Entity Resolution, you must edit the `/home/spark/code/AML_platform_connection.json` file on the gateway computer.

The Financial Crimes Insight Entity Resolution feature searches through party data to identify possible matches with variances in spelling and other properties. For example, James Smith and James Smythe might be the same party with different spellings in different records. For more information about configuring the entity resolution method, see Entity Resolution.

You can configure FCI entity resolution to run automatically as a result of ingesting organization and individual party data into IBM Financial Crimes Alerts Insight with Watson (FCAI). For more information about ingesting data into IBM FCAI, see Ingesting the data.

The `AML_platform_connection.json` file includes the following:

```
{
    "sendMessage": false,
    "platformHost": "https://<hostname>:9444/ibm/fci/platform/external_assessment/
analysis_result",
    "platformUser": "",
    "platformPassword": "",
    "sslValidationEnabled": true,
    "tokenURL": "https://<hostname>:9444/ibm/fci/platform/token_service/token",
    "entityBulkLoad" : {
        "enabled" : false,
        "bulkLoadURL" : "https://<hostname>:9444/ibm/api/batch/jobinstances/",
        "localDir" : "/home/flume/tempBulkLoad",
        "sftpInfo" : {
            "sshKeyFile" : "/home/flume/.ssh/id_rsa",
            "remoteUser" : "wlpadmin",
            "remoteHost" : "<hostname>",
            "remoteFileSystem" : "/fci-shared/bulkload"
        },
        "jobParameters" : {
            "fciSolutionDir" : "/fci-shared/bulkload"
        }
    },
    "entityResolution": {
        "enabled": false,
        "driverClass" : "com.ibm.db2.jcc.DB2Driver",
        "driverJar" : "db2jcc4.jar",
        "dbUrl": "jdbc:db2://<dbhostplatform>:<dbhostport>/
CFDB:currentSchema=CFFACT;sslConnection=true;sslCertLocation=/<dir>/db2.crt",
        "dbUser": "<dbuserplatform>",
        "dbPassword": "<dbpasswordplatform>"
    }
}
```

Where:

- The `<hostname>:9444` value is the address of the fci-analytics service. Depending on the installed topology, this value is a Kubernetes cluster address or a Docker host address.
- The `<dbhostplatform>` value is the host component of the fci-primaryds service. Depending on the installed topology, this value is a Kubernetes cluster address or a Docker host address.
- The `<dbhostport>` value is the port component of the fci-primaryds service.

- The <dbuserplatform> value is the user name to connect to the fci-primaryds service.
- The <dbpasswordplatform> value is the password to connect to the fci-primaryds service. This value must be base64 encoded.
- The /<dir>/db2.crt value is the local path and file name of the certificate for the fci-primaryds service.

Edit the following values:

- Set platformUser to the user name that is required to access the platform UI.
- Set platformPassword to the password for the user that is required to access the platform UI. This value must be base64 encoded.
- Set sslValidationEnabled to `true` to enable SSL validation. Set the value to `false` to skip the validation.
- Set entityBulkLoad.enabled to `true`.
- Set entityBulkLoad.localDir to a local directory on the gateway computer where the platform `/fci-shared/bulkload` directory is mounted. For example, `/home/flume/tempBulkLoad`.
- Set entityBulkLoad.sftpInfo.remoteHost to the server where the platform `/fci-shared/bulkload` directory resides.
- Set entityBulkLoad.entityResolution.enabled to `true`.

Note: After this integration is enabled, the FCI bulk load and entity resolution batch jobs are submitted as part of the Flume flows that ingest the organization and individual data. After you edit the JSON file, you must restart the Flume service.

Configuring insights

You can configure insights that are displayed for alerts in the system in IBM Financial Crimes Alerts Insight with Watson (FCAI).

IBM FCAI provides a way to configure insights that are displayed for alerts in the system, based on reasons generated by the analytics.

This configuration should be handled by an Administrator.

Configuring an insight type

Configuration can be handled by updating DB2 directly, either using a database client such as SQuirreL, or by using SQL commands in the terminal or in a script.

Insight types are located in the `“AML-FSDM”.”INSIGHT-TYPE”` table. Insight types have the following columns:

ID
This is automatically generated.

NAME
The name for your insight. This will be displayed on the user interface.

DESCRIPTION_HIGH_SCORE
A description displayed on the user interface for your insight when the insight has a high score.

DESCRIPTION_LOW_SCORE
A description displayed on the user interface for your insight when the insight has a low score.

VISUALIZATION_TYPE_ID
Foreign key to the `VISUALIZATION_TYPE_ID` in the `“AML-FSDM”.”VISUALIZATION_TYPE”` table. All insights are shown on the user interface with a table, by default, but will show also show the specified visualization if provided.

Configuring features in an insight type description

You can configure features in an insight type description in IBM Financial Crimes Alerts Insight with Watson (FCAI).

When you configure the description for an insight, you can use static text or you can include features that provide more information to the user about the particular insight. By including one of the following features in the insight's description (either DESCRIPTION_HIGH_SCORE or DESCRIPTION_LOW_SCORE), IBM FCAI automatically populates values that are associated with the insight for a particular alert on the user interface.

For example, a description such as "{numTransactions} alerted transactions involved {countryRiskLevel} countries ({countryNames})" might become "4 alerted transactions involved High Risk countries (Afghantistan and Lebanon)" when it is displayed on the user interface, depending on the data that is identified by the analytics as being important.

Features that are recognized by IBM FCAI are as follows:

numTransactions
Total number of transactions that are shown for the insight.

countryRiskLevel
A list of country risks (for example, "High Risk and Low Risk") for all countries for the insight transactions. This risk comes from the Atlas country data.

countryNames
A list of all country names for the insight transactions.

startTransactionDate
The date of the earliest transaction in the insight.

endTransactionDate
The date of the most recent transaction in the insight.

totalTransactionAmount
The sum of all transaction amounts in the insight.

minTransactionAmount
The smallest transaction amount in the insight.

maxTransactionAmount
The largest transaction amount in the insight.

If you surround the features in an insight description with curly braces, for example, { }, it creates a variable. IBM FCAI replaces the variable with the pertinent data in the user interface.

Important: If an unsupported feature is used, or if a feature is misspelled, the entire insight description is not replaced and the value appears as the unformatted description on the user interface.

Associating an insight type with a reason type

For an insight to be displayed in IBM Financial Crimes Alerts Insight with Watson (FCAI), it must first be associated with a reason type.

Reasons generated by the analytics have associated reason types, so an insight will be displayed on the user interface when an alert has a reason that matches a defined insight.

The association between an Insight Type and a Reason Type is done in the "AML-FSDM"."INSIGHT_TYPE_REASON_TYPE" table. This table contains the following columns:

- ID – This is automatically generated.
- INSIGHT_TYPE_ID – Foreign key to the ID in the "AML-FSDM"."INSIGHT_TYPE" table.
- REASON_TYPE_ID – Foreign key to the REASON_TYPE_ID in the "AML-FSDM"."REASON_TYPE" table.

Creating and modifying Flume agents

Flume agents are used by IBM Financial Crimes Alerts Insight with Watson (FCAI) to move data in the system. You can create new or modify existing agents.

For information about using Apache Flume agents and creating your own, see the Flume 1.5.0 User Guide (https://flume.apache.org/releases/1.5.0.html).

Flume agents consist of three elements: a source, a channel, and a sink. The channel connects the source to the sink. You must configure each element in the Flume agent. Different source, channel, and sink types have different configurations, as described in the Flume documentation.

IBM FCAI uses some custom sources and sinks:

- `com.ibm.aml.flume.HiveCommandSink`—used to execute a Hive command
- `com.ibm.aml.flume.NewFileSource`—used to provide the file name as a parameter
- `com.ibm.aml.flume.SendToExecutableSink`—used to execute a bash command
- `com.ibm.aml.flume.SpoolDirectorySource`—used to set the spooldDir source

Flume agents are defined by a configuration file. The configuration file values and examples are provided by the Flume documentation.

The following is an example of a Flume agent configuration:

```
EXAMPLE_AGENT.channels =  example-truncate-channel
EXAMPLE_AGENT.sources =  example-truncate-source
EXAMPLE_AGENT.sinks =  example-truncate-sink

EXAMPLE_AGENT.sources.example-truncate-source.type = com.ibm.aml.flume.NewFileSource
EXAMPLE_AGENT.sources.example-truncate-source.channels = example-truncate-channel
EXAMPLE_AGENT.sources.example-truncate-source.spoolDir = /amldata
EXAMPLE_AGENT.sources.example-truncate-source.includePattern = ^example-.*\.csv$
EXAMPLE_AGENT.sources.example-truncate-source.fileSuffix = .FLOWED_TO_TRUNCATE
EXAMPLE_AGENT.sources.example-truncate-source.recursiveDirectorySearch = true
EXAMPLE_AGENT.sources.example-truncate-source.fileHeader = true
EXAMPLE_AGENT.sources.example-truncate-source.fileHeaderKey = file

EXAMPLE_AGENT.channels.example-truncate-channel.type = memory

EXAMPLE_AGENT.sinks.example-truncate-sink.type = com.ibm.aml.flume.HiveCommandSink
EXAMPLE_AGENT.sinks.example-truncate-sink.channel = example-truncate-channel
EXAMPLE_AGENT.sinks.example-truncate-sink.hiveJdbcUrl = jdbc:hive2://localhost:2181/
default;serviceDiscoveryMode=zooKeeper;zooKeeperNamespace=hiveserver2
EXAMPLE_AGENT.sinks.example-truncate-sink.hiveCommand = truncate table example_table
EXAMPLE_AGENT.sinks.example-truncate-sink.sourceFileHeaderKey = file
EXAMPLE_AGENT.sinks.example-truncate-sink.sourceFileSuffix = .FLOWED_TO_TRUNCATE
EXAMPLE_AGENT.sinks.example-truncate-sink.completedCommandSuffix = .TABLE_TRUNCATED
```

You can view IBM FCAI Flume configuration template files to see how other custom components are used. The Flume template files are located in the `/opt/ibm/fcai/install/config/flume-templates` directory on each node computer.

The file can be edited by the root user.

Agent naming convention

Flume agents are Java processes. To stop them, you can kill the process.

Stopping the agents can also be done automatically through the Ambari console. Ambari used PID files to track each agent's process ID. However, the Ambari API uses grep commands to search for agents. If the agents are named to similarly, then Ambari might not correctly stop an agent.

For example, if two agents are named AML_Account and AML_AccountMapping, the Ambari API will not correctly resolve their names.

When the Flume agents are started, the process ID (PID) for AML_AccountMapping will be put into both that agents PID file and the AML_Account PID file. When the agents are stopped, only the AML_AccountMapping agent is stopped and the AML_Account will still be running.

The names do not have to be that different. You can check whether the names are different enough by using the `ps -ef | grep your_agent_name` command. If multiple agent processes appear, you must change the agent names. For example, in the above example, changing AML_Account to AML_Accounts is enough to ensure that they are unique enough.

The PID files are found in the `var/run/flume` directory.

Agent component naming convention

You must define the sources, channels, and sinks, for each agent. For example, for `AML_Account` the sources, channels, and sinks might be:

```
AML_Account.sources = account-example-source
AML_Account.channels = account-example-channel
AML_Account.sinks = account-example-sink
```

No source, channel, or sink component name can be the same among any agent. For example, the following names would not be valid, given the above agent:

```
AML_AccountsMapping.sources = accountmapping-example-source
AML_AccountsMapping.channels = account-example-channel
AML_AccountsMapping.sinks = accountmapping-example-sink
```

In this case `AML_Account.channels` and `AML_AccountsMapping.channels` use the same channel, `account-example-channel` .

Customizing text for redacted data

If data is deleted for GDPR compliance requirements, the deleted data is replaced with <REDACTED> in the dashboard. You can replace the default string by modifying the `/opt/ibm/aml/bin/hive_merge.sh` script.

Procedure

1. Log in to the gateway node as the root user.
2. Enter the following command to open the `hive_merge.sh`:

   ```
   vi /opt/ibm/aml/bin/hive_merge.sh ambari.host.com 8081 ambari-username ambari-password
   ```

3. In the `Clean up DB2 database` section, locate the following line:

   ```
   java -cp "$fcaijar:$jacksoncorejar:$jacksonmapjar:$db2jar"
   com.ibm.aml.gdpr.AlertReviewRedactor "$db2jsonpath"  $id
   ```

4. At the end of that line, enter the text that you would like to use instead of <REDACTED>. Enter the replacement text in quotation marks.

   For example, to use **REDACTED**, change the line to:

   ```
   java -cp "$fcaijar:$jacksoncorejar:$jacksonmapjar:$db2jar"
   com.ibm.aml.gdpr.AlertReviewRedactor "$db2jsonpath" $id "**REDACTED**"
   ```

5. Save and close the file.

Changing the default Ambari password after installation

Ambari is installed with a default password. You can change the password by using the Ambari console.

If you changed the password before you ran the installation, you do not need to follow these steps.

Procedure

1. Log in to the Ambari console.

   The URL is `https://ambari.server:8081`

   The default credentials are `admin` / `admin`

   a) Click **Admin**, and select **Manage Ambari**.

   b) In the **User + Group Management** box, click **Users**.

   c) In the list, click the admin user.

   d) Click **Change Password**.

   e) Enter the current password in**Your Password**. For example, enter the default `admin`.

   f) Enter a new password in **New User Password**.

   g) Click **OK**.
2. For IBM FCAI, you must rebuild the crontab that you use to process the transactions.

   For more information, see “Scheduling transaction processing” on page 23.

Updating the Db2 user's password for the amldb database

If the amldb database user's password is changed, you must update the `AML_db2_connection.json` file with the new password.

Procedure

1. Log on to the gateway computer as the flume user.
2. Go to the `/home/spark/code` directory.
3. Get the base64 value of the new password by using the following command:

   ```
   echo -n 'newpassword' | base64
   ```
4. Edit the `AML_db2_connection.json` file and replace the dbPassword value with the base64 value for the new password.
5. Save and close the file.

   The next time that you run an analytics process, the updated password is used.

Updating the platform password for the Case Manager user

If the password for the Case Manager user that is listed in the `AML_platform_connection.json` file has changed, you must update the file with the new password.

The default user is fciadmin.

Procedure

1. Log on to the gateway computer as the flume user.
2. Go to the `/home/spark/code` directory.

3. Get the base64 value of the new password by using the following command:

```
echo -n 'newpassword' | base64
```

4. Edit the `AML_platform_connection.json` file and replace the platformPassword value with the base64 value for the new password.
5. Save and close the file.

   The next time that you run an alert ingestion, the updated password is used to connect to the Case Manager during the alert messaging step.

Chapter 4. Using IBM Financial Crimes Alerts Insight with Watson

This section will introduce you to key concepts of the IBM Financial Crimes Alerts Insight with Watson (FCAI) user interface.

Log in to the IBM FCAI dashboard

You access the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard from a supported web browser.

Procedure

1. From your browser, go to the IBM Financial Crimes Insight with Watson URL:

   `https://hostname.mycompany.com:3080/`
2. Enter your user name and password, and click **Login**.
3. Optional: Click **Alerts Insight**.

   If your user role allows you to access multiple components, a page is displayed showing the components, such as **Alerts Insight**. If your role allows you to access only a single installed component, then you will automatically be taken to that component.

   To log out of Alerts Insight, click the user icon, and then click **Logout**. Logging out will automatically clear your user cookie from the browser.

Use the dashboard

IBM Financial Crimes Alerts Insight with Watson (FCAI) provides a dashboard where you can quickly see all of the alerts in the system and interact with them.

From the dashboard you can:

- Filter the displayed alerts by various criteria
- Search for a specific alert ID or customer name
- Specify an insight status for an alert
- Export the filtered alert data as a CSV file

The dashboard is not intended to replace your case management system.

Viewing alerts as a list

By default, when the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard is first displayed, all alerts in the system are displayed in a List view.

About this task

The **List** view includes the **Watson Insight Score**, the **Alert ID**, **Customer Name**, **Date Created**, **Alert Type**, and the **Original Alert Score**. The alert insight status specified by users of the system is also displayed.

- If you are viewing the dashboard in Group view and want to switch back to the **List** view, click **View as** > **List**.
- If you want to see the group a particular alert is in, click **Group** next to the alert and the **Group** view is displayed with the appropriate sub-group opened.

The number of alerts included in the dashboard is shown.

Viewing alerts grouped by customer name

You can view alerts grouped by customer name in IBM Financial Crimes Alerts Insight with Watson (FCAI).

About this task

If you want to see alerts in the IBM FCAI dashboard with all alerts associated with a customer grouped together, switch to the **Group** view.

Procedure

1. Click **View as** > **Group**. Alerts for each customer are displayed together.
2. Click a customer name. All alerts for that customer are displayed in sub-groups by the month and year the alerts were created.
3. Click a month and year sub-group. All alerts for that customer opened in that month and year are displayed. The same columns available in the **List** view are displayed for each alert.
4. Collapse groups by clicking the group or sub-group heading.

   The number of alerts included in the dashboard is shown.

Sorting alerts

When alerts are displayed in the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard **List** or **Group** views, the data can be sorted by any column heading.

Procedure

- In the **List** view, the displayed alerts can be sorted by clicking the column headers.
- In the **Group** view, sorting by column headers can only be done after a customer group and month and year sub-group are expanded.

Filtering alerts

The alerts that are displayed by the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard can be filtered by different criteria.

Procedure

- Select the **Date Range** values to display only alerts that were created before, after, or within a certain date range.
- Select the alert types that you want to view in **Alert Type**.
- Adjust the **Watson Insight Score** slider to specify the score range. The scores are inclusive within the selected range.
- Click an **Alert Insight Status** button to select the status types to display.

  When filters are selected, the applied filters are displayed above the list or group table. To clear a single filter, click the **X** on the button with the filter to be cleared. To clear all filters and search criteria, click **Clear All**.

Searching for alerts

You can search for a specific alert ID or customer name in the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard.

Procedure

- The value specified in the Search box will be matched against the **Alert ID** and **Customer Name** columns. As you type in the Search box, the dashboard will be updated with alerts with matching values. Searches are not case-sensitive.
- To clear only the search, delete the text you entered into the Search box. To clear both the search and any applied filters, click **Clear All**.

Specifying a status for an alert

You can specify a status for an individual alert, or group of alerts, by using the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard.

By default, IBM FCAI uses the following status values:

- **Not Reviewed**
- **In Review**
- **Reviewed**
- **The alert status is unknown**
- **The alert is open**
- **The alert is closed**

Important: When multiple users are signed on to the dashboard, and one user changes an alert's status, other users will not see the status change on their system until they log off and log back on.

Procedure

To change the status of an individual alert in the **List** or **Group** views, click the **Alert Insight Status** for the alert and select the desired status.

Exporting alerts

You can export the alerts displayed in the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard.

About this task

The alerts displayed in the IBM FCAI dashboard List or Group view can be exported as a CSV file. Only the data displayed in the columns on the dashboard is exported. The alerts displayed on the dashboard are included in the exported CSV file. By using the filter and search options, you can limit what is included in the exported CSV file.

To export alerts from the dashboard:

Procedure

1. Search and filter the alerts as desired from either the **List** or **Group** views.
2. Click **Export**. A CSV file is written to the file system.

Opening an alerts insight view

You can access the details for any alert from the IBM Financial Crimes Alerts Insight with Watson (FCAI) dashboard.

Procedure

- Click a cell in the alerts table to display an alert's details. The alert insights details for the alert are displayed in a new tab.

  Clicking the **Alert Insight Status** column shows a drop-down list that allows you to change the status of the alert.

  Clicking **Group** in the **View** column displays the group view.

View alerts

The IBM Financial Crimes Alerts Insight with Watson (FCAI) user interface provides an at-a-glance view into suspicious activity in the form of alerts.

The default view shows the **Insight Summary** tab that shows all of the insights that the analytics detected. The left pane provides a link for each insight. Click the link to show more details for the insight.

Click the party or organization name to expand the menu bar and display more details about the party or organization. Some of the displayed information depends on the type of party. For example, birth dates and social security numbers might be shown for individuals, whereas company websites and tax IDs might be shown for organizations. Account numbers and contact information is shown for both.

The following information is available for an alert:

Watson™ Insight Score
: This score represents the secondary score that is generated by the analytics. The score provides an indication of the risk of the alert and is not meant to be a confidence score.

Entity Summary
: This collapsible panel is a summary of the entity that triggered the alert. This entity can be an individual, a group, or a corporation. The summary includes information about any related parties, any known name variances, including aliases for the current entity, and any identifying properties known about the current entity (for instance, date of birth, gender, telephone number, address, country of residence, social insurance number, tax ID).

Key Insights
: This section provides a summary of all insights that are found for the alert.

Insights
: The user interface consists of insights that are generated by the analytics. Together, the insights make up the components of the **Watson Insight Score**. Each insight is displayed as a separate section that contains a brief explanation about the insight, along with a display in the form of a map, or bar or line graph, and a table of transaction data.

Notes
: There is a **Notes** button at the bottom of each tab of an alert. Analysts can use the notes to collect information such as transactions that are related to an alert that lead to a certain conclusion. Notes can be added to the narrative generation to augment the information about the alert and the decision to proceed or not.

  When you first click **Notes**, you are shown a list of notes that you can edit or delete. Click **New Note** to create a new note instance. Click **Go back to all notes** to return to display the saved notes.

  For each note, analysts can add the following:

  - Enter a name for the note in **Note Name**. You can include more than one note for each alert.
  - Select **Include Selected Transactions** to add the selected transactions to the note. The transactions are shown in a table within the note.

- Click **Browse Image** to select an image file (such as a jpg) to add to the note. Hovering over the image thumbnail allows you to delete the image.
- Enter text in **Description** box.

Using the Insight Summary view

The IBM Financial Crimes Alerts Insight with Watson (FCAI) **Insight Summary** tab shows all of the insights that the analytics detected.

The left pane shows the overall risk and the analytic score (as a percentage range). The pane also shows the analytics category along with the analytics results for the category.

The left pane is not a static list. The displayed list is different for each alert and it is generated by back-end analytics services. There are three analytic categories: transactions, customer, and counterparty. For each category, there are many different insights.

The right pane, shows the insights with a description. Click an insight to show the analytics view. Each analytics view provides an **Add to Narrative** button.

The insights display important information about the behavior for the alert. Each insight provides a description of the behavior and might also include data in a table, map, or graph.

Using the Profile view

The **Profile** tab shows the party information (either individual or organization) that is associated with the alert.

The left pane, shows the customer information:

- Accounts that are associated with alert's party. All transaction records that are associated with the alert should be in or out of these accounts.
- Potential aliases that are associated with the alert's party that appear in the party database from CEDM.

Procedure

1. Open an alert from the dashboard or directly from your case management system.
2. Click **Profile**.

Using the Transactions view

The IBM Financial Crimes Alerts Insight with Watson (FCAI) **Transactions** view displays all of the transactions that are associated with the alerted party. The party can be the originator or the recipient of the transaction.

Procedure

1. Open an alert from the dashboard or directly from your case management system.
2. Click **Transactions**.

   The **Transactions** view opens and displays a list of all of the transactions that are associated with the alerted party.

   Select a transaction, and click the information icon ( ) to display the transaction details.

   The displayed transactions can be filtered by using the filters on the page. For example, you can filter on a date range, amount range, or transaction type.

   Click **Clear All** to clear any filters that you have applied.
3. Click **Export** to save the transaction information as a CSV file.

Generating a narrative

IBM Financial Crimes Alerts Insight with Watson (FCAI) provides an analyst with the ability to summarize their findings by generating a narrative about a particular entity.

You can generate a narrative for the alert, and select which insights to include in the narrative. You can also add standard information that is provided by your administrator to the beginning of the narrative.

Procedure

1. Click **Narrative** to generate a narrative.
2. If narrative templates are configured, you can select one of the templates for the narrative. The template information is included in the narrative before the selected insights and notes.
3. Click **Edit** and enter text for the narrative. Variables that are substituted when the narrative is downloaded are shown in square brackets. Invalid variables are displayed in red.

   The template editor allows you to add Rich Text Formatting (RTF) to your template. This includes formatting options such as bold, italic, underscore, or text color changes.

   Click the **Preview** tab to see the narrative with the variables that are substituted with the appropriate values for the entity. Any variables where data is unavailable are displayed as N/A.
4. Under **Narrative Additions**, select the **Insights** and **Notes** that you want to include.

   Select a note by clicking the **+** icon for the note. The contents of the note are added to the narrative. After you add a note, the **+** sign changes to an **X**. Click the **X** to remove the note from the narrative.

   Notes usually have a title and a creation date. Notes might also have a transaction table of selected transactions that are relevant to the narrative or images, such as screen captures of information that is relevant to the parties in the transaction. The last section of the note is a text area that contains content that was provided by the investigator when they created the note.
5. Click **Download Narrative** to save the narrative as an HTML file. The file is a standalone HTML file that you can email or save and use later without having to be log in to IBM FCAI. The file includes the rich text content as well as images, HTML transaction tables (that come from notes), and the insight data.
6. Alternately, you can select the insights to be included from the **Key Insights** page itself. Click **Add insight to narrative**. Any insights that are selected on the **Key Insights** page are shown when the **Generate Narrative** dialog is displayed.

Using the Six Month Variance view

The IBM Financial Crimes Alerts Insight with Watson (FCAI) Six Month Variance view displays information about the transactions that were made by the alerted party in the six months before the alert.

Procedure

1. Open an alert from the dashboard or directly from your case management system.
2. Click **Six Month Variance**.

   The **Six Month Variance** view opens and displays a graph that shows the transactions that were made by the party for the 6 months before the alert date. The graph shows both inbound and outbound transactions for each month.

   You can filter the view to show specific types of transactions, such as deposits or account transfers.

   A table that shows a summary of the inbound and outbound transactions of each type is also displayed. The summary includes the totals for both of the three-month periods within the six months, the differences between those periods, and the percentage difference between those periods.

Using custom narrative templates

You can use customer-defined templates for generating alert insight narratives in IBM Financial Crimes Alerts Insight with Watson (FCAI).

Only administrator users can edit, upload, or manage narrative templates.

Procedure

1. From the dashboard or an alert insight page, click the user icon, and then click **Manage Templates**.
2. Click **Upload New Template**. Click **Select file (.txt)** and select a file from your system.

   Important: Template files must be text files and can be no larger than 64 KB. The file name cannot be longer than 200 characters. Also, the text files cannot contain non-keyboard characters. An error is displayed when you upload a text file that contains invalid characters.
3. Optional: Enter a name in **Enter template name**. This name will be displayed in the list of templates in which users can select when generating narratives. If you do not provide a name, the file name of the uploaded file will be used.

   Note: Template names must be unique.
4. Click **Upload** to upload the template to the system.
5. If you want the uploaded template included in the list of template available to users, select **Available to User**. **Available to User** is selected by default.
6. To delete a template from the system, click **Delete Template** next to the template to be deleted.

   For more information about narrative generation, see “Generating a narrative” on page 44.

Editing narrative templates

Templates that were uploaded to IBM Financial Crimes Alerts Insight with Watson (FCAI) can be edited directly without having to upload them again.

You must have administrative permissions to edit narrative templates.

Procedure

1. From the dashboard or an alert insight page, select **your user name** > **Manage Templates**.
2. Click the name of the template you want to edit.
3. The name of the template, contents, and whether the template is available to be uses by users can be changed.

   The contents can include text and dynamic variables. Dynamic variables are replaced by specific entity and insight information when a narrative is generated by a user. You can add dynamic variables by selecting them from the list of available variables, or by typing the variable names in the body of the template. Valid variables are indicated in blue. Any invalid variables are indicated in red.

   For more information about using variables, see “Narrative template variables” on page 46.

   The template editor allows you to add Rich Text Formatting (RTF) to your template. This includes formatting options such as bold, italic, underscore, or text color changes.

   Note: Template names cannot be longer than 200 characters and the template itself can be no larger than 64 KB.
4. Click **Save Changes** to save your changes or **Cancel** to cancel them.

Narrative template variables

You can include specific information that is associated with an entity in generated narratives by using variables. The variables are substituted with information that is specific to the entity and insights when the narrative is generated.

The following tables show the variables that you can use.

Table 2: Alerts variables

Variable	Description
[AlertType]	The type of alert
[AlertDate]	The date that the alert was triggered
[TodaysDate]	The current date
[Score]	The alert score
[Title]	The alert title
[Priority]	The priority of the alert

Table 3: Transactions variables

Variable	Description
[TransactionStartDate]	Date of the first transaction
[TransactionEndDate]	Date of the last transaction
[TransactionTotal]	The total amount for all transactions
[TransactionCount]	The total count of all transactions
[TotalTransactionsSent]	The total count of all sent transactions
[TotalTransactionsReceived]	The total count of all received transactions
[TotalAlertedTransactions]	The total count of all triggering transactions
[TotalAmountSent]	The total amount sent in the transactions
[TotalAmountReceived]	The total amount received in the transactions
[TotalAmountAlerted]	The total amount of all triggering transactions
[TransactionTypeTotal]	The total count of different transaction types
[CounterpartyTotal]	The total number of unique counterparties
[AllCounterparties]	The list of all of the counterparty names
[CounterpartyTransCount]	The list of the total transaction count per counterparty
[CounterpartyCountryTransCount]	The list of the total transaction count per counterparty country
[TransTypeCount]	The list of the total transaction count per transaction type
[TransTypeSentAmount]	The list of the total sent transaction amount per transaction type

Table 3: Transactions variables (continued)	
Variable	**Description**
[TransTypeRecAmount]	The list of the total received transaction amount per transaction type

Table 4: Customer variables	
Variable	**Description**
[CustomerName]	The name of the customer
[AccountNumber]	The customer's account number
[DateOfBirth]	The customer's birth date
[Occupation]	The customer's occupation
[Industry]	The customer's industry or NAICS code
[IncorporationDate]	The customer's established date
[PhoneNumber]	The customer's phone number
[FirstAccInfo]	A summary of the first account's number, type, and opened date
[AddtlAccInfo]	A summary of additional account's numbers, types, and opened dates
[HomeStreet]	The home address
[HomeCity]	The home address city
[HomeState]	The home address state
[HomeZipCode]	The home address zip code
[HomeCountry]	The home address country
[WorkStreet]	The work address
[WorkCity]	The work address city
[WorkState]	The work address state
[WorkZipCode]	The work address zip code
[WorkCountry]	The work address country

Table 5: Transaction review variables	
Variable	**Description**
[TransReviewLength]	The total length of transaction review period
[TransReviewStart]	The transaction review period start date
[TransReviewEnd]	The transaction review period end date
[TransReviewTransStart]	The start date of transactions within the transaction review period

Table 5: Transaction review variables (continued)	
Variable	**Description**
[TransReviewTransEnd]	The end date of transactions within the transaction review period
[TransReviewSentTotal]	The total amount sent in transactions during the transaction review period
[TransReviewRecTotal]	The total amount received in transactions during the transaction review period

For example, if you use the following narrative template:

```
[CustomerName] born [DateOfBirth] made [TransactionCount] transactions in the following
countries: [CounterpartyCountryTransCount].
```

The resulting narrative could appear as:

```
DR Jones born 1-8-1947 made 4 transactions in the following countries: 2 transactions
involved England, 2 transactions involved Switzerland.
```

Variables must be enclosed in square brackets. If you are creating the template directly in the template editor, or are editing an existing template, the editor validates the variables as you type. If an invalid variable is entered, the editor indicates the incorrect variable in red.

Variables can also be included in templates that are created outside of IBM FCAI and then uploaded into the system. Any invalid variables are indicated in red when the template is edited.

Chapter 5. Troubleshooting

HDP cluster configuration script fails

You are running the HDP cluster configuration script for IBM FCAI, but the script fails because the httpd service is not running.

This can occur if the httpd service is not enabled to automatically restart after a system reboot, and you have rebooted the computer.

You can resolve this problem, by starting the httpd service.

Run the following commands to start the service:

```
systemctl enable httpd
```

```
systemctl start httpd
```

NFS gateway not restarting after system restart

After you restart your computer, the NFS gateway does not start after you start the Ambari server environment. This can be caused by some NFS daemons not restarting successfully.

In a Terminal window, type the following command:

```
systemctl status rpcbind
```

A typical error message for this situation is:

```
dependency job for rpcbind.service failed. See 'journalctl -xe' for details
```

To resolve this issue, you can edit the `rpcbind.socket` file and restart the rpcbind service:

1. Make a backup of `/usr/lib/systemd/system/rpcbind.socket`
2. Open `/usr/lib/systemd/system/rpcbind.socket` in a text editor.
3. Edit the file to look like the following:

   ```
   [Unit]
   Description=RPCbind Server Activation Socket

   [Socket]
   ListenStream=/var/run/rpcbind.sock
   #ListenStream=[::]:111
   ListenStream=0.0.0.0:111
   BindIPv6Only=default

   [Install]
   WantedBy=sockets.target
   ```

4. Restart the rpcbind service:

   ```
   systemctl start rpcbind
   ```

   If the rpcbind service does not start after you edit the `rpcbind.socket` file, reboot the server.
5. In the Ambari console, click **Actions** > **Start All**.

Ranger KMS service not started in Ambari console

In the Ambari console, the Ranger KMS service is not started.

This can happen if the URL that is used for the service uses `localhost` rather than the fully qualified domain name or IP address.

To resolve this issue, under **Advanced ranger-kms-audit**, in the Ambari console, ensure that the **xasecure.audit.destination.hdfs.dir** value uses the FQDN or IP address of the server where the Ranger KMS service is running. The server host name is shown when you select the Ranger KMS Server in the Ambari console.

The value should appear as `hdfs://`*`hostname.domain.com`*`:8020/ranger/audit`

If you change the value, restart the Ranger KMS service.

Error in force(code) : Failed while connecting to sparklyr to port (8880) for sessionid (7122)

While running the analytics, you receive the following error messages:

```
Reading AMLAnalyticConfiguration.json, training = FALSE
Using SPARK_HOME from AMLAnalyticConfiguration: /usr/hdp/current/spark2-client
Error in force(code) :
Failed while connecting to sparklyr to port (8880) for sessionid (7122): ignoring SIGPIPE signal
Path: /usr/hdp/2.6.4.0-91/spark2/bin/spark-submit
Parameters: --driver-memory, 8G, --class, sparklyr.Shell, '/usr/lib64/R/library/sparklyr/java/
sparklyr-2.2-2.11.jar', 8880, 7122
Log: /tmp/RtmpC6ovNO/file8817a09ae1d_spark.log

---- Output Log ----
18/06/20 18:48:16 INFO sparklyr: Session (7122) is starting under 127.0.0.1 port 8880
18/06/20 18:48:16 INFO sparklyr: Session (7122) found port 8880 is available
18/06/20 18:48:16 INFO sparklyr: Gateway (7122) is waiting for sparklyr client to connect to
port 8880

---- Error Log ----
Calls: sparkConnect ... tryCatchOne -> -> abort_shell -> -> force
Execution halted
```

If this error occurs, rerun the analytics process.

HdfsApiException error in Ambari console

You're running a query by using Hive View or Hive View 2.0 in the Ambari console, and you see an `HdfsApiException` error.

This error can occur if HDFS isn't configured properly in Ambari. For example, if the configuration was changed after IBM FCAI was installed, such as when IBM InfoSphere Big Match for Hadoop is installed after IBM FCAI.

Do the following to resolve this problem:

1. Log in to the Ambari console.
2. In the left pane, click **HDFS**.
3. Click the **Configs** tab, and then click the **Advanced** tab.
4. Expand **Custom core-site**.
5. Enter * for the following properties:
   - **hadoop.proxyuser.root.groups**
   - **hadoop.proxyuser.root.hosts**
6. Click **Save**.

7. In the left pane, click **Actions** > **Restart all required**.
8. Rerun the query.

FOLIO_PRIORITY_TYPE type was not found error

After you register an analytic flow, you see a FOLIO_PRIORITY_TYPE error messages in the log files.

The error appears as:

```
COVCF7035E: The creation or update failed because the MEDIUM parameter for the
FOLIO_PRIORITY_TYPE type was not found. Update the JSON or XML to supply a valid stereotype for
the FOLIO_PRIORITY_TYPE type.
```

To resolve this error, go to the CFFOLIO.FOLIO_PRIORITY_TYPE table in the Case Manager **Code Tables** view. In the **Stereotype** column, change the lowercase **high**, **medium**, and **low** values to all capitals: **HIGH**, **MEDIUM** and **LOW**.

Chapter 6. Reference

The reference topics provide information about IBM Financial Crimes Alerts Insight with Watson (FCAI) system properties and ports.

Port reference

IBM Financial Crimes Alerts Insight with Watson (FCAI) uses several ports to communicate between components and for the initial installation.

Table 6: IBM FCAI Ports

Container port	Description
3080	Application user interface
5601	Kibana. This port should not be accessible from the internet.
56000	IBM Db2

For ports that are used by the Ambari server, see the Apache Ambari Administration documentation (https://docs.hortonworks.com/HDPDocuments/Ambari-2.6.1.5/bk_ambari-administration/content/default_network_port_numbers_-_ambari.html).

Formatting data files

The IBM Financial Crimes Alerts Insight with Watson (FCAI) data must be provided as CSV files, including the header, in a specific format.

For information about the IBM Financial Crimes Insight platform bulk loader data formatting, see Mapping CSV columns to CEDM table fields (https://www.ibm.com/support/knowledgecenter/SSCKRH_1.0.3/platform/r_bulk_loader_csv_103.html).

Note:

All timestamps should be in one of the following two formats:

- yyyy-MM-dd'T'HH:mm:ssZ (example: "2017-04-07T16:15:01Z")
- yyyy-MM-dd'T'HH:mm:ss.SSSSSSZ (example: "2017-04-07T16:15:01.000000Z")

All country fields must use the same format as specified in the `AMLAnalyticConfiguration.json` CountryType field. They must either Name, ISOAlpha2, ISOAlpha3, or ISONumeric.

Account data

acct_id: string

tax_id: string

alt_id: string

dsply_nm: string

type: string

initial_deposit: float

balance: float
net_wrth_prd_end: timestamp
acct_peer_grp_intrl_id: string
acct_rptng_crncy: string
branch_id: string
relationship_mgr_id: string
open_dt: timestamp
close_dt: timestamp
close_reason: string
acct_stat: string
acct_stat_dt: timestamp
acct_last_updated_ts: timestamp
acct_segment: string
dormant_status_flag: string
frozen_status_flag: string
acct_efctv_risk: int
acct_bus_risk: int
cstm_risk1: int
past_confirmed_fraud: int
past_suspected_fraud: int
prior_sar_count: int
customized_1: string

AccountCustomerMapping data

cust_acct_mapping_id: string
acct_id: string
cust_id: string
cust_acct_role: string
src_sys: string
data_dump_dt: timestamp

Alerts data

alert_id: string
src_sys: string
src_id: string
cust_id: string
acct_id: string
subject_display_name: string

start_ts: timestamp
end_ts: timestamp
priority: string
susptype1: string
susptype2: string
susptype3: string
alert_score: float
peer_group: string
disposition: string
desc_1: string
desc_2: string
desc_3: string
creater_id: string
create_ts: timestamp
review_state: string
review_ts: timestamp
review_oper: string
status: string
closed_ts: timestamp
disposition_oper: string

AlertTxnMappings data

alert_id: string
type: string
tran_id: string
party: string

Transactions data

tran_id: string
reference_source: string
reference_id: string
cxl_trxn_id: string
entry_timestamp: timestamp
tran_timestamp: timestamp
post_timestamp: timestamp
curr: string
base_amt: float
tx_type: string

paymt_method: string

trxn_type: string

dbt_cdt: string

is_back_office: boolean

is_bank_to_bank: boolean

instruction: string

tran_chanl: string

chanl_risk_level: string

chanl_risk_score: int

is_frgn_trxn: boolean

orig_cust_id: string

orig_acct: string

orig_inst_id: string

orig_display_nm: string

orig_ctry: string (in CountryType format)

origbk_cust_id: string

origbk_acct: string

origbk_inst_id: string

origbk_display_nm: string

origbk_ctry: string (in CountryType format)

benebk_cust_id: string

benebk_acct: string

benebk_inst_id: string

benebk_display_nm: string

benebk_ctry: string (in CountryType format)

bene_cust_id: string

bene_acct: string

bene_inst_id: string

bene_display_nm: string

bene_ctry: string (in CountryType format)

intermediary1_cust_id: string

intermediary1_id: string

intermediary1_acct: string

intermediary1_display_nm: string

intermediary1_ctry: string (in CountryType format)

intermediary2_cust_id: string

intermediary2_id: string

intermediary2_acct: string

intermediary2_display_nm: string

intermediary2_ctry: string (in CountryType format)

intermediary3_cust_id: string

intermediary3_id: string

intermediary3_acct: string

intermediary3_display_nm: string

intermediary3_ctry: string (in CountryType format)

intermediary4_cust_id: string

intermediary4_id: string

intermediary4_acct: string

intermediary4_display_nm: string

intermediary4_ctry: string (in CountryType format)

Chapter 7. Viewing log file information

You can view log file information for the IBM Financial Crimes Alerts Insight with Watson (FCAI) Hortonworks Data Platform (HDP) components by using Kibana.

For more information, see Viewing log file information in the FCI platform documentation.

Appendix A. Software License Metric (SLM) usage

Software License Metric (SLM) usage information provides the capability for a product to report its consumption of license metrics (resources that are related to the use of the software asset).

For more information, see IBM License Metric Tool.

IBM Financial Crimes Alerts Insight with Watson (FCAI) uses a Kibana report for the product usage. Messages are logged when the product ingests an alert. The report shows the unique IDs ingested per month for a year.

Each time an alert is ingested, an alert ID is recorded in a log file that is on the gateway server (where Flume is running). Through Logstash and Elasticsearch, the alert ID is saved to a field (called alert_id) in the logstash-flume index. In Kibana, the index is queried on the "Unique Count" of the alert_id field for each month.

For example, 10 alerts were submitted in January, but the alerts were two repetitions of 5 different alert IDs (for example, Alert 1, Alert 1, Alert 2, Alert 2, ..., Alert 5, Alert 5). The "Unique Count" of the alert_id field for January would register as 5. If the same 10 alerts were also ingested in February, then February would also have a Unique Count of 5. However, if alerts 1 to 10 were ingested in March, then March would have a unique count of 10.

The time window that IBM FCAI uses for the "Unique Count per Month" measure is from now (whatever point in time the user opens the report) through 1 year in the past. The report shows the Unique Count IDs for the previous 12 months.

You can access the report from a URL:

```
https://kubernetes.master:5601/app/kibana#/visualize/create?
type=histogram&indexPattern=beb774b0-6934-11e8-8395-3fd5f545aae8&_g=(refreshInterval:
(display:Off,pause:!f,value:0),time:(from:now-1y,mode:quick,to:now))&_a=(filters:!(),linked:!
f,query:(language:lucene,query:%27%27),uiState:(),vis:(aggs:!((enabled:!t,id:%271%27,params:
(field:alert_id.keyword),schema:metric,type:cardinality),(enabled:!t,id:%272%27,params:
(customInterval:%272h%27,extended_bounds:(),field:%27@timestamp
%27,interval:M,min_doc_count:1),schema:segment,type:date_histogram)),params:(addLegend:!
t,addTimeMarker:!f,addTooltip:!t,categoryAxes:!((id:CategoryAxis-1,labels:(show:!
t,truncate:100),position:bottom,scale:(type:linear),show:!t,style:(),title:
(),type:category)),grid:(categoryLines:!f,style:
(color:%23eee)),legendPosition:right,seriesParams:!((data:(id:%271%27,label:%27Unique%20count
%20of%20alert_id.keyword%27),drawLinesBetweenPoints:!t,mode:stacked,show:true,showCircles:!
t,type:histogram,valueAxis:ValueAxis-1)),times:!(),type:histogram,valueAxes:!
((id:ValueAxis-1,labels:(filter:!f,rotate:0,show:!
t,truncate:100),name:LeftAxis-1,position:left,scale:(mode:normal,type:linear),show:!t,style:
(),title:(text:%27Unique%20count%20of%20alert_id.keyword%27),type:value))),title:%27New
%20Visualization%27,type:histogram))
```

Ensure that you change *kubernetes.master* to the fully qualified domain name of the Kubernetes master node computer.

You can adjust the report to show different 12-month periods. In the Kibana report, click the clock icon in the upper-right corner, and choose the options that you want to display.

For more information about message processing, see Chapter 7, “Viewing log file information,” on page 59.

Notices

This information was developed for products and services offered worldwide.

This material may be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service. This document may describe products, services, or features that are not included in the Program or license entitlement that you have purchased.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Software Group
Attention: Licensing

3755 Riverside Dr.
Ottawa, ON
K1V 1B7
Canada

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

This Software Offering does not use cookies or other technologies to collect personally identifiable information.

Trademarks

IBM, the IBM logo and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at " Copyright and trademark information " at www.ibm.com/legal/copytrade.shtml.