IBM Predictive Maintenance and Quality
Version 1.0

Solution Guide

IBM

Note

Before using this information and the product it supports, read the information in “Notices” on page 103.

Product Information

This document applies to IBM Predictive Maintenance and Quality 1.0 and may also apply to subsequent releases.

Licensed Materials - Property of IBM

Contents

Introduction

The IBM® Predictive Maintenance and Quality solution uses data from multiple sources to give you the information to make informed operational, maintenance, or repair decisions.

IBM Predictive Maintenance and Quality provides you with operational intelligence data, which enables you to:

- Understand, monitor, predict, and control product and process variability
- Perform in-depth root cause failure analysis
- Identify incorrect operating practices
- Enhance equipment and process diagnostics capabilities

It also provides you with asset performance management capabilities that enable you to:

- Have forward visibility into equipment and process performance
- Increase asset uptime
- Identify safety issues
- Identify improper maintenance procedures
- Optimize maintenance intervals and procedures.

Audience

This guide is intended to provide users with an understanding of how the IBM Predictive Maintenance and Quality solution works. It is designed to help people who are planning to implement IBM Predictive Maintenance and Quality know what tasks are involved.

Finding information

To find IBM Predictive Maintenance and Quality product documentation on the web, go to the IBM Predictive Maintenance and Quality information center (http://pic.dhe.ibm.com/infocenter/pmq/v1r0m0/index.jsp).

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products. Some of the components included in the IBM Predictive Maintenance and Quality solution have accessibility features.

IBM Predictive Maintenance and Quality 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.

Chapter 1. IBM Predictive Maintenance and Quality

With IBM Predictive Maintenance and Quality, you can monitor, analyze, and report on information that is gathered from devices. In addition, recommendations for actions can be generated by IBM Predictive Maintenance and Quality.

IBM Predictive Maintenance and Quality is an integrated solution that you can use to perform the following tasks:

- Predict the failure of an instrumented asset so that you can prevent costly unexpected downtime.
- Quickly mine maintenance logs to determine the most effective repair procedures and maintenance cycles.
- Identify the root cause of asset failure faster so that you can take corrective actions.

Instrumented assets generate data such as device ID, timestamp, temperature, and status code. This data can be collected and used in models that predict when an asset is likely to fail.

Examples of instrumented assets are manufacturing equipment, mining equipment, drilling equipment, farming equipment, security equipment, cars, trucks, trains, helicopters, engines, cranes, oil platforms, and wind turbines.

For example, an oil refinery is a system that combines thousands of interlocking pieces. It is critical that such a system is able to work safely and efficiently. You can use IBM Predictive Maintenance and Quality to monitor and track the lifecycle of each piece of the refinery, such as each pipe, pump, compressor, valve, furnace, turbine, tank, heat exchange unit, and boiler. Reports give you the information to ensure that you have the right parts available and can schedule repairs during downtimes.

Predictive maintenance

In predictive maintenance, you look for patterns in the usage and environmental information for equipment that correlate with failures that take place. This information is used to create predictive models to score incoming new data. You can predict the likelihood of failure. Scores are generated from this information that give you an indication of the health of the piece of equipment. In addition, key performance indicators (KPIs) are collected, which are used for reporting. KPIs help you to identify assets that do not conform to normal patterns of behavior. You can define rules to generate recommendations when a piece of equipment is identified as having a high probability of failure. Recommendations can be fed into other systems so that people are alerted to them automatically.

Predictive quality in manufacturing

Past operations data, environmental data, and historical defect data can be used to identify the causes of elevated defect rates. This information is used in predictive models, so that when incoming data is fed into the models, you can predict the likely defect rates. The predicted values are then used for analysis and reporting and to drive recommendations such as modification to inspection patterns, or recalibration of machinery. Scoring can be done on a near real-time basis.

IBM Predictive Maintenance and Quality tasks

You must configure your IBM Predictive Maintenance and Quality application before the application can be deployed to users.

The following tasks are necessary to configure IBM Predictive Maintenance and Quality:

- Identify the assets, resource types, their event types, and measurements.
- Load the master data. Master data supplies IBM Predictive Maintenance and Quality with information about the context in which events occur, for example, lists of supported languages, location of a resource or event, definition of a material or production process.
- Load the event data. The event data is data that you want to measure about an event. Data comes from many sources, and it must be transformed into a format that can be used by IBM Predictive Maintenance and Quality.
- Configure the event types, measurement types, and profile variables. Set up the types of measurement that must be done, and the key performance indicators (KPIs) that must be calculated from these measurements. Profiles are a condensed history of resources that help to speed up scoring.
- Configure the predictive models. Run the historical data through the modeler to determine which values are needed. You can then refine the model so that it gives you accurate predictions and generates scores.
- Define rules that determine what actions happen when a score threshold is breached.
- Configure the reports and dashboards that the user sees. Reports and dashboards can be customized, and new ones can be created.

Identification of the assets, resource types, event types, and measurement types

Before you deploy an IBM Predictive Maintenance and Quality application, identify the assets and the information that you want to monitor.

To establish what data is required, and what preparation must be done, ask the following questions.

- Which assets must be monitored, and why?
- What events do you want to monitor for those assets?
- What measurements do you want to capture for the events?

Resource types

The two supported resource types are asset and agent. An asset is a piece of equipment that is used in the production process. An agent is the operator of the equipment. When you define resources, you can use the resource subtype field to identify specific groups of assets or agents.

The following table shows some sample event types in the data model.

Table 1. Sample event types in the data model

Event type code	Event type name
ALARM	Alarm
WARNING	Warning

Table 1. Sample event types in the data model (continued)

Event type code	Event type name
SYSTEM CHECK	System Check
MEASUREMENT	Measurement
RECOMMENDED	Recommended Actions
FAILURE	Failure
REPAIR	Repair

The following table shows some sample measurement types in the data model.

Table 2. Sample measurement types in the data model

Measurement type code	Measurement type name
RECOMMENDED	Recommend Action
RPM	RPM
FAIL	Incident Count
INSP	Inspection Count
LUBE	Lube Count
OPHR	Operating Hours
PRS1	Pressure 1
PRS2	Pressure 2
PRS3	Pressure 3
R_B1	Replace Ball Bearing Count
R_F1	Replace Filter Count
RELH	Relative Humidity
REPT	Repair Time
REPX	Repair Text
TEMP	Ambient Temperature
Z_AC	High Temperature/ Humidity Warning Count
Z_FF	Latent Defect
Z_PF	Probability of Failure
Z_TH	High Temperature/ Humidity Count
OPRI	Operating Hours at Inception
REPC	Repair Count
MTBF	MTBF
MTTR	MTTR
OPRD	Operating Hours Delta

Create a custom application

You can create a custom IBM Predictive Maintenance and Quality application by creating custom IBM WebSphere® Message Broker flows, IBM Cognos® Business Intelligence reports and dashboards, or predictive models.

The following list describe the high-level tasks you can take to create a custom application.

- Customize or create new predictive models by using IBM SPSS® Modeler.
- Create new business rules by using IBM Analytical Decision Management.
- Create new flows that interface with external systems by using IBM WebSphere Message Broker.
- Customize scoring during event processing by using IBM WebSphere Message Broker.
- Customize or create the message flows to orchestrate the activities by using IBM WebSphere Message Broker.
- Customize or create new reports by using IBM Cognos Report Studio.
- Modify the metadata for the reports by using IBM Cognos Framework Manager.

Sample files, model files, and other content is supplied to help you to configure IBM Predictive Maintenance and Quality for the needs of your business. For more information, see Appendix D, "IBM Predictive Maintenance and Quality Artifacts," on page 95.

Integration with IBM Maximo Asset Management

You can integrate IBM Maximo® Asset Management with IBM Predictive Maintenance and Quality.

IBM Predictive Maintenance and Quality can create work orders in IBM Maximo Asset Management based on recommendations from predictive scoring and decision management. IBM Maximo Asset Management can be a source of both master data, and event data.

IBM Maximo is not installed as part of IBM Predictive Maintenance and Quality. If required, it must be purchased separately. However, IBM Predictive Maintenance and Quality contains adapters for IBM Maximo, which allow data integration.

Chapter 2. Orchestration

Orchestration is the process that ties activities in IBM Predictive Maintenance and Quality together.

Orchestration is achieved with message flows in IBM WebSphere Message Broker. The following activities can be tied together:

- Acquiring and storing data
- Aggregating data
- Running predictive models
- Feeding data back to external systems or starting external processes

Important: Training predictive models are not part of the message flows and must be performed manually. For more information, see “Training and scoring predictive models” on page 42.

Message flows are supplied with IBM Predictive Maintenance and Quality and must be customized with IBM WebSphere Message Broker. The message flows are organized into four applications:

- **PMQEventLoad**
- **PMQIntegration**
- **PMQMasterDataLoad**
- **PMQMaximoIntegration**

For more information about customizing message flows, see the IBM WebSphere Message Broker Information Center (publib.boulder.ibm.com/infocenter/wmbhelp/v8r0m0).

By default, IBM WebSphere Message Broker is installed in advanced mode. Advanced mode is the correct mode to use for full functionality.

The following examples describe how orchestration is used in IBM Predictive Maintenance and Quality.

Orchestration example: Load real-time event data

This orchestration example is similar to the message flow used to load batch event data.

1. Incoming equipment measurement data is provided through real-time connectivity.
2. A map must be defined in WebSphere Message Broker to describe the transformation of incoming data into the IBM Predictive Maintenance and Quality event structure.
3. Incoming business keys are converted to internal integer surrogate keys.
4. Event data is written to the datastore.
5. Event data is aggregated. Profile and key performance indicator (KPI) data is written to the datastore.

Orchestration example: Load batch event data

The following steps take place when batch event data is loaded into IBM Predictive Maintenance and Quality.

1. Incoming measurement data is loaded from a file.
2. The file system in polled for new incoming data.
3. A map that is defined in WebSphere Message Broker describes the transformation of incoming data into the IBM Predictive Maintenance and Quality structure.
4. Incoming business keys are converted to internal integer surrogate keys.
5. Event data is written to the datastore.
6. Event data is aggregated. Profile and key performance indicator (KPI) data is written to the datastore.

Orchestration example: Score event data

The following steps take place when event data is scored.

1. New input triggers scoring. For example, to recalculate the health score if a new measurement is reported, then that measurement is processed and the health score is recalculated
2. A map that is defined in WebSphere Message Broker describes the transformation of the data into the model structure.
3. The predictive model is invoked through a web services interface.
4. A map that is defined in WebSphere Message Broker describes the transformation of model outputs to the event structure.
5. Model outputs are written as new events.
6. As with external events, model output events can be aggregated and stored on the profile and as KPIs.

For more information about scoring predictive models, and the triggers to score models, see “Predictive scoring” on page 36.

Orchestration example: Apply business rules to data

The following steps take place when business rules are applied.

1. New input triggers evaluation of business rules.
2. A map that is defined in IBM WebSphere Message Broker describes the transformation of the data into to the model structure.
3. IBM Decision Management Model is invoked through a web services interface.
4. A map that is defined in IBM WebSphere Message Broker describes transformation of model outputs to the event structure.
5. Model outputs are written as new events.
6. As with external events, model output events can be aggregated and stored on the profile and as KPIs.

Orchestration example: Write-back of data

The following steps take place when write-back of data to an external process occurs.

1. Creation of an event triggers the requirement to start an external process.

2. A map that is defined in IBM WebSphere Message Broker describes the transformation of the data to the structure of an external web service.
3. The external web service is called.

Chapter 3. Master data

Master data is the type of resource that you want to manage, for example, definition of a material or production process.

Master data can come from manufacturing engineering systems (MES) such as IBM Maximo, or other existing data sources. IBM InfoSphere® Master Data Management Collaboration Server can be used to complete gaps in the data from these sources or consolidate data from multiple sources. You can also add attributes, create relationships between items, or define data that you do not have another source for. For example, add hierarchy information to indicate which pieces of equipment belong to which site, in which location or classify resources into groups. In a report, the hierarchies and groups can be displayed as additional information or used as drill downs and filters.

Master data is normally loaded by one of the supplied connectors or the Flat File API. The connectors and the Flat File API use IBM WebSphere Message Broker flows to transform the data into the form that is required and update the data in the IBM Predictive Maintenance and Quality database.

Master data process

When a file is placed in the file input directory, IBM WebSphere Message Broker reads and processes it and then removes it from the directory. WebSphere Message Broker stores and retrieves data from the database as required.

The response file indicates whether the operation was successful, and lists any results. If errors occur, a log file is written to the error directory.

The following diagram shows the flow of a file request and its response.

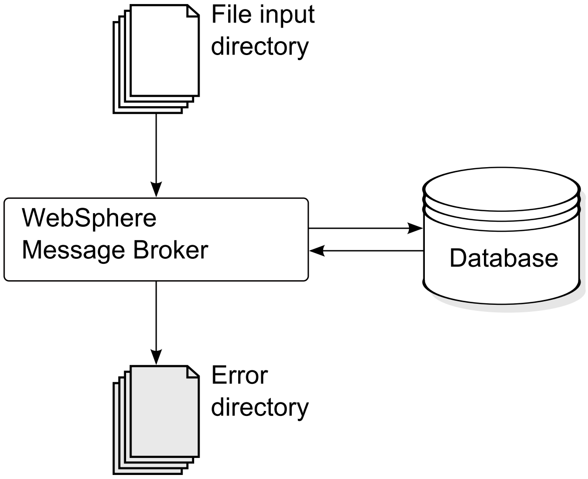


Figure 1. Master data process

Data organization

IBM Predictive Maintenance and Quality processes the following kinds of data:

- Master data supplies IBM Predictive Maintenance and Quality with information about the context in which events occur. Master data includes descriptions of the devices that produce events, the location where events occur, and the material that is used in an event.
- Metadata defines how IBM Predictive Maintenance and Quality processes received events. For more information, see "Metadata in the API" on page 75.
- Event data supplies IBM Predictive Maintenance and Quality with information that you want to measure about an event. For more information, see "How events are processed" on page 25.

The flat file application programming (API) interface

IBM Predictive Maintenance and Quality master data is supplied, accessed, modified, or removed using the flat file API. For more information, see Appendix A, "The flat file API," on page 61.

File format and location

Master data and event data must be in a format that IBM Predictive Maintenance and Quality can recognize. The default file format is flat file, comma separated (.csv) format. Other file formats can be used, but you must create extra IBM WebSphere Message Broker flows.

File location

The file location is determined by the `MQSI_FILENODES_ROOT_DIRECTORY` environment variable. The file location is configured during the installation process.

This location contains the following sub folders:

- `\masterdatain`

  used for loading master data and metadata files
- `\eventdatain`

  used for loading event data files
- `\error`

  used to report errors that occur while loading data
- `\maximointegration`

  used for loading data files from IBM Maximo
- `\control`
- `\restricted`
- `\properties`

File names

Files must follow this naming convention:

record_name_operation`*.csv`

For example, a file that contains a set of resource records to be added to IBM Predictive Maintenance and Quality might be named:

`resource_upsert_01.csv`

File format

The .csv file format is used by default:

- Each line in a file is a record, and contains a sequence of comma-separated values. If a value contains a comma, the value must be contained within double quotation marks ",".
- Each record normally includes a code value (or combination of values) that uniquely identifies the record. These code values are sometimes known as business keys. Because this code value is a unique identifier for a row, it is used in other files as a way to reference that particular row. For example, in a file that contains a list of resources, the row for a resource can contain a location value. The location value is the code that is used to identify a location record.
- Sometimes a code value is required but is not applicable for a particular record. In this case, the special code **`-NA-`** must be used. For example, to avoid defining a location for a particular resource, use the code **`-NA-`** for the location value. The code value cannot be changed.
- In addition to a code value, a record typically has a name value. Both code and name value can hold the same value. However, while the code value must be unique for each row and is not normally shown to users, the name is visible in reports and dashboards. The name can be changed, unlike the code value.

The following example shows the format for a `location.csv` file. The command must be on a single line, not as shown here:

```
location_cd,location_name,region_cd,region_name,country_cd,country_name,
state_province_cd,state_province_name,city_name,latitude,longitude,
language_cd,tenant_cd,is_active
RAVENSWOOD,Ravenswood,NORTH AMERICA,North America,USA,United States,
CA,California,Los Angeles,34.0522,118.2428,,
TARRAGONA,Tarragona,EUROPE,Europe,UK,United Kingdom,ENGLAND,England,
London,51.5171,0.1062,,1
```

The following example shows codes that are used to identify records and used to reference other records. The codes that are used to identify a resource record are different from other records because a resource record is identified by both `serial_no` and `model`, or by `operator_cd`.

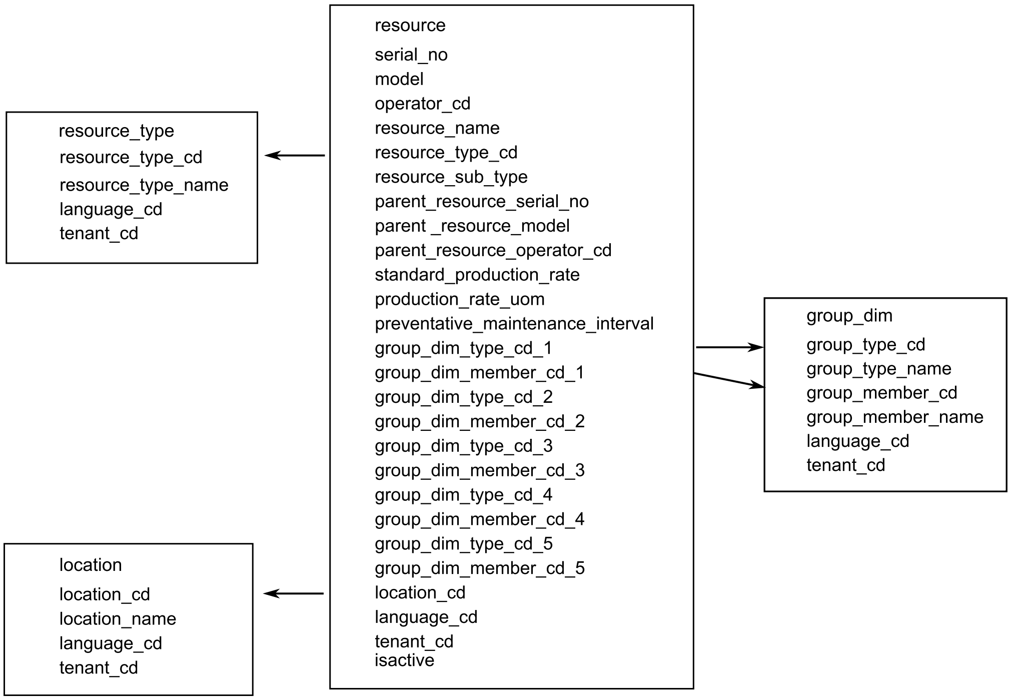


Figure 2. Codes used to identify and to reference records

Modifying a resource or process parent

If you must change a resource or process parent, you must reload the resource or process and all of its children. Modify the parent in a master data .csv file that contains all these rows, and resubmit the file.

Security

Implement security by restricting access to the directories used to supply files for the API.

Master data using InfoSphere MDM Collaboration Server

You can use IBM InfoSphere Master Data Management Collaboration Server to complete gaps in the data from external sources or consolidate data from multiple sources. You can also add attributes, create relationships between items, or define data that you do not have another source for.

For example, add hierarchy information to indicate which pieces of equipment belong to which site, in which location or classify resources into groups. In a report, the hierarchies and groups can be displayed as additional information or used as drill-downs and filters.

IBM InfoSphere Master Data Management Collaboration Server is model driven: you create a specification, and then define the fields. It automatically generates the user interface for the fields, for example, lookup tables, and date pickers. You can embed images in the data such as a picture of an asset.

A model for InfoSphere MDM Collaboration Server is provided with IBM Predictive Maintenance and Quality that simplifies the configuration. To use this model, you must do the following configuration steps.

1. Set the environment variable *PMQ_HOME* to the root of the IBM Predictive Maintenance and Quality installation directory.
2. Create a company for IBM Predictive Maintenance and Quality, see "Creating a company in IBM InfoSphere MDM Collaboration Server" on page 14.
3. Import the metadata (Company deployment), see "Importing metadata into InfoSphere MDM Collaboration Server" on page 17.
4. Configure the InfoSphere MDM Collaboration Server user interface, see "Configuring the IBM InfoSphere MDM Collaboration Server user interface" on page 15.

There are some specific guidelines that you must follow to ensure that you get the results that you expect. See "Guidelines for managing data in IBM InfoSphere MDM Collaboration Server" on page 15.

For additional information about using InfoSphere MDM Collaboration Server, see *Collaborative authoring with InfoSphere MDM Collaboration Server*. This is available from the following IBM information center: IBM Master Data Management Information Center (http://pic.dhe.ibm.com/infocenter/mdm/v10r1/index.jsp?).

IBM Master Data Management Collaboration Server dynamic references

IBM Master Data Management Collaboration Server tasks use several dynamic references.

The following table describes the variables that are used in the InfoSphere MDM Collaboration Server tasks.

Table 3. Dynamic references

Reference	Description
$PMQ_HOME	IBM Predictive Maintenance and Quality installation home directory.
mdm_install_dir	The root directory of the InfoSphere MDM Collaboration Server installation. **$TOP** is an environment variable that is configured with InfoSphere MDM Collaboration Server by default, which points to this location.
mdm_server_ip	The IP address of InfoSphere MDM Collaboration Server, as seen by other IBM Predictive Maintenance and Quality servers, such as IBM WebSphere Message Broker.
pmq_mdm_content_zip	The full path to the content compressed file on the server file system.
mdm_data_export_dir	The directory, mount point, or symbolic link on the InfoSphere MDM Collaboration Server where data exports are configured to be written. The default is <*$PMQ_HOME*>/data/export/mdm.
wmb_server_ip	The IP address of WebSphere Message Broker server, as seen by other IBM Predictive Maintenance and Quality servers.
wmb_fileapi_input_dir	The directory where input data files are to be placed for loading into the IBM Predictive Maintenance and Quality database. The directory can be local or remote. The file location is determined by the **MQSI_FILENODES_ROOT_DIRECTORY** environment variable. The file location is configured during the installation process.
company_code	The company code for InfoSphere MDM Collaboration Server. Make the code short and easy to remember because it must be entered during each login, for example, IBMPMQ.

Table 3. Dynamic references (continued)

Reference	Description
company_name	The display name of the company in InfoSphere MDM Collaboration Server, for example, `IBMPMQ`.

Creating a company in IBM InfoSphere MDM Collaboration Server

You must create a company before you can import IBM Predictive Maintenance and Quality metadata into IBM Infosphere Master Data Management Collaboration Server. A company is similar to the concept of a project.

About this task

For information about the variables used, see "IBM Master Data Management Collaboration Server dynamic references" on page 13.

Procedure

1. Stop the InfoSphere MDM Collaboration Server service.
   a. Change the directory to `cd <mdm_install_dir>/bin/go` where *<mdm_install_dir>* is the root directory of the InfoSphere MDM Collaboration Server installation.
   b. Run the **`stop_local_sh`** command: `./stop_local.sh`
2. Run the company creation script.
   a. Change the directory to `cd <mdm_install_dir>/bin/db`
   b. Run the **`create_cmp.sh`** command: `./create_cmp.sh –code=<company_code> --name=<company_name>`
3. Start the InfoSphere MDM Collaboration Server service.
   a. Change the directory to `cd <mdm_install_dir>/bin/go`
   b. Run the **`start_local.sh`** command:`./start_local.sh`
4. Log in and verify the company. Open your web browser and enter the URL for the InfoSphere MDM Collaboration Server web server, for example: `http://<mdm_host_name>:7507/utils/enterLogin.jsp`

   The following default users are created for the new company:

Table 4. Default roles, users, and passwords created for a new company

Role	User name	Password
Administrator	Admin	trinitron
Basic User	Basic	trinitron

5. Change the default passwords for both the administrator, and for the basic user. You do this in the **Data Model Manager** module > **User Console**.

What to do next

The next step is to import the IBM Predictive Maintenance and Quality metadata into the InfoSphere MDM Collaboration Server.

Configuring the IBM InfoSphere MDM Collaboration Server user interface

Add the IBM Predictive Maintenance and Quality objects in the IBM Master Data Management Collaboration Server navigation area to make it easier to manage data.

Procedure

1. In InfoSphere MDM Collaboration Server, click **Please select a module to add**. A drop-down list is displayed.
2. Select the all of the following modules from the **Catalog** module type.
   - **Asset**
   - **Locations**
   - **Material Types**
   - **Processes**
   - **Products**
   - **Suppliers**
3. Select **Groups by Type** from the **Hierarchy** module type.

What to do next

You can customize group types to suit the needs of the project.

1. In the **Groups by Type** hierarchy, choose a group type, and customize it as required with a new code or name.
2. Save the changes.
3. Update the **Group Hierarchy Lookup** by clicking **Product Manager** > **Lookup Tables**, **Lookup Table Console**.
4. Update the group type record with the new group type code.

Guidelines for managing data in IBM InfoSphere MDM Collaboration Server

You must follow these guidelines to manage data in IBM InfoSphere Master Data Management Collaboration Server to ensure that you get the results that you expect.

Assets

Define assets in the **Unassigned** category.

You can use the default hierarchy to organize items, but the hierarchy is not used by IBM Predictive Maintenance and Quality.

Group assignments:

- Can be assigned up to five groups from the **Groups by Type** hierarchy.
- Each assignment must be from a different group type.
- Must be assigned to Group (Level 2), not Group Type (Level 1.)

Groups

Groups are managed by using the group hierarchy rather than a catalog. Only categories are defined, not items.

The first level must be group type.

The second level must be groups.

Locations

Define the locations as follows:
- The first level must be **Region** (`Location Type=Region`).
- The second level must be **Country** (`Location Type=Country`).
- The third level must be **State** (`Location Type=State / Province`).

The location items must be defined only under `State / Province` (only on a leaf node).

Material types, processes, products, and suppliers

Define items in the **Unassigned** category.

You can use the default hierarchy to organize items, but the hierarchy is not used by IBM Predictive Maintenance and Quality.

Configuring and running data exports

To integrate IBM InfoSphere Master Data Management Collaboration Server into IBM Predictive Maintenance and Quality, data export files must be sent to the data input directory for the flat file API on the IBM WebSphere Message Broker server.

Before you begin

For information about the variables that are used, see “IBM Master Data Management Collaboration Server dynamic references” on page 13.

About this task

The IBM WebSphere Message Broker file location is determined by the `MQSI_FILENODES_ROOT_DIRECTORY` environment variable, and the folder is named `\masterdatain`. The file location is configured during the installation process.

Procedure

1. On the IBM WebSphere Message Broker server, ensure that the Network File System (NFS) is configured to run with the following command.

   ```
   /sbin/chkconfig nfs on
   ```
2. Share the data input directory for the flat file API by adding the following line to `/etc/exports`. Create the directory if it does not exist.

   ```
   <wmb_fileapi_input_dir> <mdm_server_ip>(rw)
   ```
3. Ensure that sufficient permissions are set on the data input directory.

   The following example grants read and write permissions to all users and groups. If you require a more secure configuration, ensure that users, groups, and permissions are consistent with those on the InfoSphere MDM Collaboration Server so that NFS operates correctly.

   ```
   chmod 777 <wmb_fileapi_input_dir>
   ```
4. Restart the NFS service for the settings to take effect.

   ```
   service nfs restart
   ```

5. On the InfoSphere MDM Collaboration Server, ensure that the data export directory exists. If it does not, create the directory.

   ```
   mkdir  <mdm_data_export_dir>
   ```
6. Mount the remote flat file API input directory with NFS.

   ```
   mount -t nfs -o rw wmb_server_ip:wmb_fileapi_input_dir  mdm_data_export_dir
   ```
7. Test NFS sharing.
   a. Create a test file on the InfoSphere MDM Collaboration Server.

      ```
      echo <"NFS Test File"> <mdm_data_export_dir>/nfstest.txt
      ```
   b. Check for the test file on the IBM WebSphere Message Broker server:

      ```
      cat <wmb_fileapi_input_dir>/nfstest.txt
      ```

Results

If the file content is displayed, NFS is working. If you have problems, search for "Red Hat Linux NFS documentation" online for detailed information.

What to do next

To run a data export, in the InfoSphere MDM Collaboration Server Reports Console, select the export and click the **Run** icon. Data export files are written to $PMQ_HOME/<*mdm_data_export_dir*>. The default is $PMQ_HOME/data/export/mdm.

Importing metadata into InfoSphere MDM Collaboration Server

You must import IBM Predictive Maintenance and Quality data into IBM Master Data Management Collaboration Server before you can use MDM to manage data.

About this task

For information about the variables that are used, see "IBM Master Data Management Collaboration Server dynamic references" on page 13.

Procedure

Use the following command to import data into InfoSphere MDM Collaboration Server. The command must be on a single line, not as shown here.

```
<mdmce_install_dir>/bin/importCompanyFromZip.sh
--company_code=<company_code>
--zipfile_path=IBMPMQ.zip
```

Example

See the following example.

```
$TOP/bin/importCompanyFromZip.sh —company_code=IBMPMQ —zipfile_path
=$PMQ_HOME/content/IBMPMQ.zip
```

$TOP is a built-in IBM InfoSphere Master Data Management Collaboration Server environment variable, which points to the root Master Data Management Collaboration Server directory.

IBM Maximo Asset Management

Master data and event data can be supplied from IBM Maximo to IBM Predictive Maintenance and Quality. Recommended actions that are generated by IBM Predictive Maintenance and Quality can also be passed to IBM Maximo Asset Management.

IBM Maximo Asset Management is not installed as part of IBM Predictive Maintenance and Quality. If required, it must be purchased separately. However, IBM Predictive Maintenance and Quality contains adapters for IBM Maximo, which allow data integration.

How master data is mapped in IBM Maximo Asset Management

As an example, the following tables in IBM Predictive Maintenance and Quality can be populated from the default Maximo object model.

group_dim table

The records in the group_dim table provide classifications for resources. You can have up to five classifications for each resource. The classifications might vary.

Table 5. Fields in the group_dim table

Field	Type	Required or optional	Maximo objects/attributes
group_type_cd	string(50)	Required	"MXCLASSIFICATION"
group_type_name	string(200)	Required	"Maximo Classification"
group_member_cd	string(50)	Required	CLASSSTRUCTURE.CLASSSTRUCTUREID
group_member_name	string(200)	Required	CLASSSTRUCTURE.DESCRIPTION

location table

The location table contains the location of a resource or event, such as a room in a factory or a mine site. In Maximo, this information is stored as a LOCATIONS object and in its associated SERVICEADDRESS object.

Table 6. Fields in the location table

Field	Type	Required or Optional	Maximo objects/attributes
location_cd	string(50)	Required	SERVICEADDRESS.ADDRESSCODE
location_name	string(200)	Required	SERVICEADDRESS.DESCRIPTION
region_cd	string(50)	Optional, region_cd and region_name must be supplied together	SERVICEADDRESS.REGIONDISTRICT
region_name	string(200)	Optional	SERVICEADDRESS.REGIONDISTRICT
country_cd	string(50)	Optional, country_cd and country_name must be supplied together	SERVICEADDRESS.COUNTRY

Table 6. Fields in the `location` table (continued)

Field	Type	Required or Optional	Maximo objects/attributes
`country_name`	string(200)	Optional	`SERVICEADDRESS.COUNTRY`
`state_province_cd`	string(50)	Optional, `country_cd` and `country_name` must be supplied together	`SERVICEADDRESS.STATEPROVINCE`
`state_province_name`	string(200)	Optional	`SERVICEADDRESS.STATEPROVINCE`
`city_name`	string(200)	Optional	`SERVICEADDRESS.CITY`
`latitude`	float (in decimal degrees)	Optional	`SERVICEADDRESS.LATITUDE`
`longitude`	float (in decimal degrees)	Optional	`SERVICEADDRESS.LONGITUDE`

`resource` table

A resource defines resources of type asset or agent. An asset is a piece of equipment. An agent is the operator of the equipment. Some asset resources might form a hierarchy (for example, a truck is a parent of a tire). Asset information imported from Maximo includes the asset type, classification, and location.

Table 7. Fields in the `resource` table

Field	Type	Required or Optional	Maximo objects/attributes
`serial_no`	string(50)	Either `serial_no` and `model` or `operator_cd` are required	`ASSET.ASSETNUM`
`model`	string(50)		`ASSET.SITEID`
`operator_cd`	string(50)		
`resource_name`	string(500)	Required	`ASSET.DESCRIPTION`
`resource_type_cd`	string(50)	Required	
`resource_sub_type`	string(50)	Optional	`ASSET.ASSETTYPE`
`parent_resource_serial_no`	string(50)	Optional (`parent_resource_ serial_no` and `parent_resource_model` should be supplied together)	`ASSET.PARENT`
`parent_resounce_model`	string(50)	Optional	`ASSET.SITEID`
`parent_resource_operator_cd`	string(50)	Optional	
`standard_production_rate`	float	Optional	
`production_rate_uom`	string(40)	Optional	
`preventative_maintenance_interval`	float	Optional	
`group_dim_type_cd_1`	string(50)	Group codes are required but a NA value can be specified for a corresponding type and a member	`"MXCLASSIFICATION"`
`group_dim_member_cd_1`	string(50)		`ASSET.CLASSSTRUCTUREID`

Table 7. Fields in the `resource` table (continued)

Field	Type	Required or Optional	Maximo objects/attributes
group_dim_type_cd_2	string(50)		
group_dim_member_cd_2	string(50)		
group_dim_type_cd_3	string(50)		
group_dim_member_cd_3	string(50)		
group_dim_type_cd_4	string(50)		
group_dim_member_cd_4	string(50)		
group_dim_type_cd_5	string(50)		
group_dim_member_cd_5	string(50)		
location_cd	string(50)	Required but a NA code can be specified	ASSET.SADDRESSCODE

Mapping master data in IBM Maximo Asset Management

IBM Predictive Maintenance and Quality includes sample flows that import assets, classifications, and ServiceAddress objects from the default Maximo object model. To enable these flows, master data must be exported out of IBM Maximo as XML files, and is later placed into the `\maximointegration` folder.

About this task

Asset data that is managed in IBM Maximo is mirrored in IBM Predictive Maintenance and Quality. When data is modified in IBM Maximo, it is automatically updated in IBM Predictive Maintenance and Quality. Data that comes from IBM Maximo must be updated and maintained in IBM Maximo. It is not possible for changes that are made in IBM Predictive Maintenance and Quality to be propagated back to IBM Maximo.

A Maximo Publish Channel is used to export assets, classifications, and the **ServiceAddress** attribute. You must invoke the channel manually initially to populate the IBM Predictive Maintenance and Quality database. After, the channel is automatically triggered whenever the contents of one of these objects changes.

For more information, see IBM Maximo Asset Management Information Center (http://pic.dhe.ibm.com/infocenter/tivihelp/v49r1/index.jsp).

Procedure

1. Create an object structure in IBM Maximo based on the base object structures available in IBM Maximo Asset Management.

   IBM Predictive Maintenance and Quality supports data mapping for three object structures: `SPASSET`, `SPSERVICEADDRESS`, and `SPCLASSIFICATION`.

   These object structures are inherited from base object structures in IBM Maximo: `ASSET`, `SERVICEADDRESS`, and `CLASSSTRUCTURE`.

   When the object structure is created, use the **Exclude/Include fields** option from the **Select Action** menu to include or exclude fields.

   For more information, see *Object structures* in the IBM Maximo Asset Management, *Integrating data with external applications, Integration Components* online documentation.
2. Create publish channels that are based on the object structure in IBM Maximo.

   For each publish channel, do the following actions:

- Enable a listener so that changes in data are published and reflected in IBM Predictive Maintenance and Quality.
- Configure the endpoint to be XML.

For more information, see *Publish channels* in the IBM Maximo Asset Management, *Integrating data with external applications, Integration Components, Channels and services* online documentation.

3. Create an external system and configure the corresponding endpoint for the external system as XML.

   Configure the location as the`\maximointegration` folder. The location of the folder is determined by the `MQSI_FILENODES_ROOT_DIRECTORY` environment variable.

   When IBM Maximo and IBM WebSphere Message Broker are installed on different systems, this folder must be shared, or the exported files must be transferred to this folder.
4. Set up publish channels for the external systems. Select each publish channel in turn and click **Data Export** to export data.

   The export screen supports a filter expression to export a subset of the data. The exported XML is stored in the `\maximointegration` folder.
5. Export the Object Structure schema:
   a. Search and select the Object Structure for which the schema file must be generated.
   b. Select **Generate Schema/View XML** action for that object structure. You can select the operation for which schema must be generated. Select the **Publish** operation.

   The generated schema is stored in the same location as the data export XML files. These schema files correspond to the `SPASSETService.xsd`, `SPCLASSIFICATIONService.xsd` and `SPSERVICEADDRESSService.xsd` files in the `PMQMaximoIntegration` IBM WebSphere Message Broker library.

Importing event data from IBM Maximo Asset Manager

IBM Predictive Maintenance and Quality can be customized to import IBM Maximo work orders as events to record activities such as inspections and repairs.

You must do the following tasks:

1. Create a publish channel in IBM Maximo to export the work orders.

   Take care not to import work orders that are created by IBM Predictive Maintenance and Quality.

   Modify the WorkorderCreation flow to set the `EXTREFID` field to `PMQ`. When you import the work order, do not import work orders that have the `EXTREFID` field that is set to `PMQ`.

   For more information, see the IBM Maximo Asset Management Information Center (http://pic.dhe.ibm.com/infocenter/tivihelp/v49r1/index.jsp).
2. Create a flow in IBM WebSphere Message Broker to consume these work orders, map them to the standard event format, and place them on the event processing queue.
3. Create profile variables to determine how these events are processed into key performance indicators (KPIs) and profiles. For more information, see “Profile variables” on page 29
4. Modify the event processing flow to ensure that these events trigger scoring for an appropriate predictive model. For more information, see “Event processing” on page 38.

Creating a work order service in IBM Maximo Asset Management

To create a work order, an enterprise service must be created in IBM Maximo. The enterprise service defines a web service with a WSDL file. The work order creation service is called by a WebSphere Message Broker flow in IBM Predictive Maintenance and Quality.

Before you begin

You must configure a web service in IBM Maximo Asset Management to create work orders in IBM Predictive Maintenance and Quality.

Configure IBM Maximo to expose a web service corresponding to the service defined in the **MaximoWorkOrder.wsdl** file in the **PMQMaximoIntegration** WebSphere Message Broker application.

For more information about creating an enterprise service, see the IBM Maximo Asset Management Information center (http://pic.dhe.ibm.com/infocenter/tivihelp/v49r1/index.jsp).

Procedure

Create a Web Service from the default work order enterprise service (`MXWOInterface`).

1. In IBM Maximo Asset Management, go to the **Web Services Library**, **Select Action**, **Create Web Service**, **Create Web Service from Enterprise Service**.
2. Select **EXTSYS1_MXWOInterface** and click **Create**.
3. Click the generated web service name (`EXTSYS1_MXWOInterface`)
4. Generate a WSDL file that contains the definition of the newly generated service. Click **Generate Schema**, **WSDL**, and **View XML**, and then **OK**.

   The WSDL file is generated into the global directory. To see the location, go to **System Configuration**, **Platform Configuration**, and **System Properties**. Search for `globaldir` and expand the entry.

   The generated WSDL file corresponds to the `MaximoWorkOrder.wsdl` file in the **PMQMaximoIntegration** WebSphere Message Broker application.

Turn on the functionality in IBM Predictive Maintenance and Quality to create work orders in IBM Maximo based on recommendations from the default predictive models.

5. In the IBM WebSphere MQ Explorer, set the **MaximoTRIGGER** user-defined property for the **PMQIntegration** flow to `TRUE`.
   a. In the IBM WebSphere MQ Explorer, go to **Brokers** > **MB8Broker** > **PMQ1**. Right-click the **PMQIntegration** node, and click **Properties**.
   b. Click **User Defined Properties**.
   c. Set the **MaximoTRIGGER** value to `TRUE`.
6. Set the server name in the **Web Service URL** property of the **InvokeWorkOrder** node to the name of the IBM Maximo host. This node is in the sample **WorkorderCreation.msgflow** flow in the **PMQMaximoIntegration** application.
   a. In the IBM WebSphere MQ Explorer, go to **Brokers** > **MB8Broker** > **PMQ1** > **PMQMaximoIntegration** > **Flows**, and click **Workordercreations.msgflow**.
   b. In the graphical display, right-click the **InvokeWorkOrder** node and select **Properties**.

c. In the **Web Services URL** field, enter the URL of the IBM Maximo host.

Chapter 4. Event data

Event data is any data that you want to measure about an event. Data comes from many sources, and it must be transformed into a format that can be used by IBM Predictive Maintenance and Quality.

For example, if the event is recording an inspection result, you might want to record: who was the inspector, when did the event take place, which product lot was it based on, and what was the result of the inspection?

IBM WebSphere Message Broker transforms data into a format that can be used in IBM Predictive Maintenance and Quality.

IBM WebSphere Message Broker has a visual interface that you use to map the data structure of the source data into the expected format.

The loading of event data involves the following steps:

1. In IBM WebSphere Message Broker, define the content and format of the event information that is coming in.
2. Map the data into the format that is expected by IBM Predictive Maintenance and Quality. You can use the graphical mapper, or for more complicated mappings, you can use a programming language such as Java™.
3. A message flow is provided to load data from a file. To use this flow, specify the file and location, and set a predefined time interval for checking the location. The file can be in a comma separated value format, for more information, see "File format and location" on page 10. However, by modifying a message flow, other formats such as XML are supported.

The data is processed:

- The data structure is brought into the correct format, then ported into event tables in the data store.
- The KPI and profile tables are calculated. KPIs are used in predictive models or in reports.
- This information is used to call a scoring service to receive a recommendation that is based on the current state of the event.
- The predictive model to be used is defined.

For information about the file locations and names and file format, see "File format and location" on page 10.

How events are processed

You must connect event sources to IBM Predictive Maintenance and Quality to enable events to be processed.

Events are processed in IBM WebSphere Message Broker and stored in the database. The database has an event store to record events, tables for key performance indicators (KPIs), and profiles that are related to the event source. KPIs provide a history of performance over time. Profiles show the current state of the event, and also include recommended actions from predictive models. Profiles help to speed up scoring.

The following steps occur:

1. IBM WebSphere Message Broker receives the events and maps them into the format that is required by IBM Predictive Maintenance and Quality with a custom flow, if required.
2. Events go into a queue (`PMQ.EVENT.IN`) for further processing, either as single events or multiple events that are processed together for efficiency.
3. Processed events are inserted into the event store. The information in the events immediately updates KPIs for the current KPI period. A historical record of the KPI values for each period is maintained (a period is typically one day). The event data is also used to immediately update profiles, which contain information about the current state of the event source.

This diagram shows the flow of events into IBM WebSphere Message Broker and into the database.

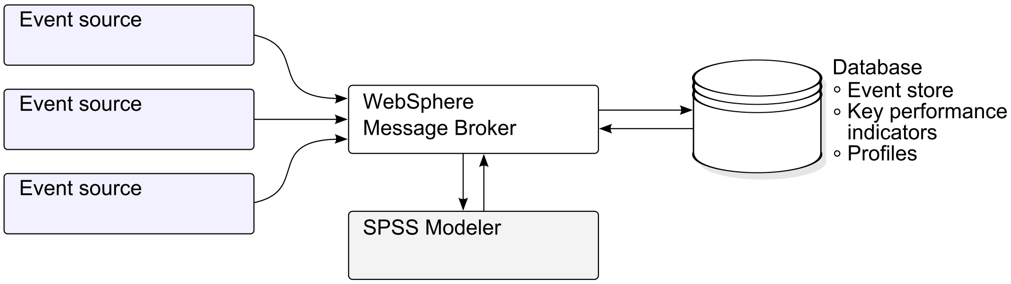


Figure 3. Flow of events into WebSphere Message Broker and into the database

The values in the event, KPI, and profile tables can be used as input to a predictive statistical model to generate recommended actions.

Processing events as they arrive, and immediately updating the aggregated values in the KPI and profile tables means that dashboards and reports are quickly updated with aggregated data.

Events must be loaded in chronological order. If events are not loaded in order then resulting KPIs and profiles may not be correct.

Event definition

Events are stored in the `event`, `event_observation`, and `event_resource` tables. An event can contain one or more event observations.

The calculated key performance indicators (KPIs) are stored in the `process_kpi` and `resource_kpi` tables. Event_observations update the values in the `process_kpi` and `resource_kpi` tables.

The calculated profile values are stored in the `process_profile`, `resource_profile`, and `material_profile` tables. The values in the row are updated as the events arrive. The tables contain values for the current period (day), prior period (prior day), and lifetime to date.

KPIs are calculated at the day level.

Flat file event input

Events can be in a flat file format (.csv) or .xml format that must conform to the format required by IBM Predictive Maintenance and Quality. Events can be in other forms, such as web services; however the WebSphere Message Broker flows must be modified and extended.

Each event contains information recorded by one or more measurements or observations. An event can be associated with one or more materials. Each event can also have an associated operator and or device.

However, each row of the input file can only define a single event, a single material, a single operator, and a single device. Therefore an event containing more than one of these must have more than one row.

The values supplied for `material_cd` associate these materials with the event.

An event requiring more than one row must set the optional `multi_row_no` to 1 in the first row of the event. Additional rows must be directly beneath this row and increase the value set in `multi_row_no` by 1 for each additional row.

The value supplied for `operator_cd` associates the event with an operator. The values in `model` and `serial_no` associate the event with a device. Only one operator and one device can be associated with an event.

Each row of a multi-row event usually has a different observation. The columns marked as observation in the following table have different values in each row of a multi-row event.

Ensure that events are pre-mapped into this format to enable them to be loaded via the application programming interface (API).

In the following table, the first ten fields, `incoming_event_cd` to `tenant_cd` are common to all of the rows of a multirow event. Only the values in the first row are used. Many of these field are codes which reference values in the master data tables. See Appendix A, "The flat file API," on page 61.

Table 8. Fields in the Events table

Field	Type	Optional or required	Event or observation	Description
`incoming_event_cd`	string(50)	optional	event	A unique code that identifies the event.
`event_type_cd`	string(50)	required	event	The event type, such as measurement, alarm, inspection.
`source_system_cd`	string(50)	optional	event	The system generating the event.
`process_cd`	string(50)	optional	event	The production process related to the event.
`production_batch_cd`	string(50)	optional	event	The production batch related to the event.
`location_cd`	string(50)	optional	event	The location of the event.

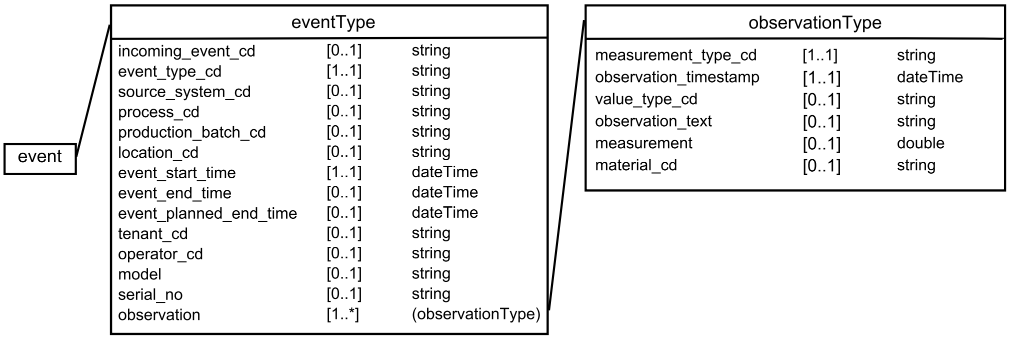
Table 8. Fields in the Events table (continued)

Field	Type	Optional or required	Event or observation	Description
event_start_time	datetime	required	event	Time the event started in Coordinated Universal Time (UTC) format, for example 2002-05-30T09:30:10-06:00.
event_end_time	datetime	optional	event	Time the event ended in UTC format.
event_planned_end_time	datetime	optional	event	Time the event was planned to end in UTC format.
tenant_cd	string(50)	optional	event	The organization associated with the event.
operator_cd	string(50)	optional	event	The operator associated with the event.
model	string(50)	optional	event	The model number of the device associated with the event.
serial_no	string(50)	optional	event	The serial number of the device associated with the event.
measurement_type_cd	string(50)	required	observation	The measurement type determines how the event observation will be processed.
observation_timestamp	datetime	required	observation	The time associated with the observation in UTC format.
value_type_cd	string(50)	optional	observation	The type of observation (actual, planned, or forecast).
observation_text	string(400)	optional (see note)	observation	A description associated with the event.
measurement	float	optional (see note)	observation	A measurement associated with the event.
material_cd	string(50)	optional	observation	The material used for an event.
multirow_no	integer	optional		For multiple row events (more than one observation) use 1 to n for each row of the event.

Note: Either measurement or observation_text is required.

Schema definition of the event format

Events are processed in the event format that is shown in the following diagram. If you are extending IBM Predictive Maintenance and Quality to process external events from other sources, you must map those events to this internal event format.



eventType		
incoming_event_cd	[0..1]	string
event_type_cd	[1..1]	string
source_system_cd	[0..1]	string
process_cd	[0..1]	string
production_batch_cd	[0..1]	string
location_cd	[0..1]	string
event_start_time	[1..1]	dateTime
event_end_time	[0..1]	dateTime
event_planned_end_time	[0..1]	dateTime
tenant_cd	[0..1]	string
operator_cd	[0..1]	string
model	[0..1]	string
serial_no	[0..1]	string
observation	[1..*]	(observationType)

observationType		
measurement_type_cd	[1..1]	string
observation_timestamp	[1..1]	dateTime
value_type_cd	[0..1]	string
observation_text	[0..1]	string
measurement	[0..1]	double
material_cd	[0..1]	string

Figure 4. The event format used by IBM Predictive Maintenance and Quality

Error reporting

Errors can occur while processing events; during mapping to the required format or during the updating of the event, KPI, and profile tables.

You can add extra properties to the message to provide event source information for reporting while mapping to the IBM Predictive Maintenance and Quality format.

Profile and KPI tables

In addition to the event store and the master data, the IBM Predictive Maintenance and Quality database includes profile and KPI tables. The content of these tables is determined by a metadata driven aggregation mechanism that determines what calculations are performed when an event is processed.

The measurement_type and the resource_type or material_type values associated with the event and a particular event_observation, form the key that is used to look up the metadata.

Profile variables

The profile_variable table drives event processing in IBM Predictive Maintenance and Quality.

When an event_observation value arrives, its associated measurement_type value, and its associated resource_type value are used to find all of the profile_variable rows that are relevant for that observation. Each of these rows indicates a calculation, which must be performed for the event. The calculation updates rows in the kpi and profile tables as indicated by the profile_variable. The profile_variable table can also contain parameters for use with the calculation. IBM Predictive Maintenance and Quality implements a standard set of calculations, but you can add a custom calculation and refer to it in a profile_variable row. The standard set of calculations include the following calculations:

- Measurement of type
- Event of type count
- Measurement of type count
- Measurement in range count
- Measurement text contains count
- Interval calculation
- Last date of event type

- Last date of measurement type
- Last date of measurement in range
- Lifespan analysis
- Lifespan analysis failure
- Measurement above limit
- Measurement below limit
- Measurement delta

These calculations are described in “Profile calculations” on page 33.

To be able to process some events, you must load mandatory profile variables and measurement types. For more information, see “Mandatory profile variables and measurement types” on page 78.

For example, a temperature event with the measurement_type value "Ambient temperature" from a device can be aggregated by defining a profile_variable for the measurement_type "Ambient temperature" with the profile_calculation "Measurement of type". This causes a row to be added to the resource_kpi table at each period for this device and profile_variable. This row aggregates the temperature values across each period (day). In addition, the defined profile_variable causes a row to be added to the resource_profile table for this device that is updated as each temperature event is processed.

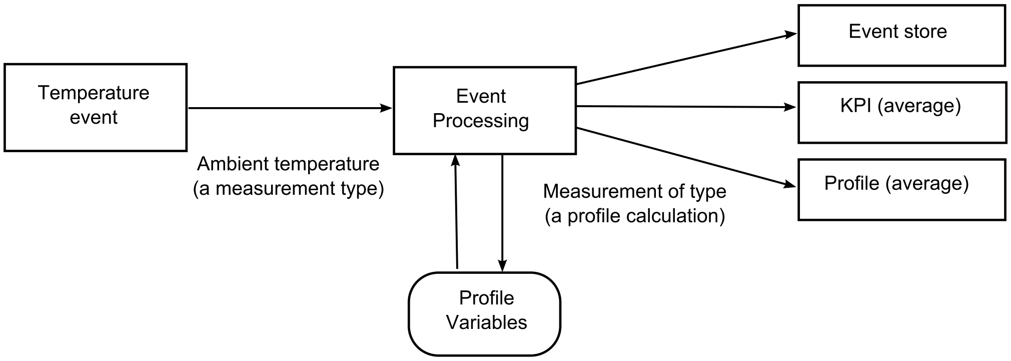


Figure 5. Temperature event workflow

Making a profile variable inactive

To make a profile variable inactive, for example, if you want to prevent a calculation from being performed, set the kpi_indicator and profile_indicator values to 0.

KPI tables

The IBM Predictive Maintenance and Quality key performance indicator (KPI) tables: resource_kpi and process_kpi hold aggregated values for each day.

In the resource_kpi table, the key for each row is determined by
- The profile_variable that triggered the calculation of the KPI
- The date
- The resource that is associated with the event
- The event code that is associated with the event observation

- The location that is associated with the event
- The process that is associated with the event
- The production batch that is associated with the event
- The tenant_id.

The fields in resource_kpi are described in the following table.

Table 9. Fields in the resource_kpi table

Field	Type	Description
kpi_date	date	The date for which the KPI is calculated. The time grain for KPI calculation is a single day.
profile_variable_id	integer	The profile variable that is the source of this KPI.
resource_id	integer	The resource that is associated with the event.
event_code_id	integer	The event code that is associated with the event observation. Event codes are codes for alarms, failures, issues, and so on. When an event arrives with a measurement_type value that has an event_code_indicator value of 1, the text from the event_observation_text field is assumed to contain an event_code value.
location_id	integer	The location that is associated with the event.
process_id	integer	The process that is associated with the event.
production_batch_id	integer	The production batch that is associated with the event.
actual_value	float	The actual value for this KPI. It is important to understand that for Business Intelligence reporting purposes, this value is typically divided by the measure count. Even if the value is meant to be an average, this value must be a sum of the values from the event and the measure_count must be the number of events. actual_value field supports the average calculation for dimensional reporting.
plan_value	float	The planned value for the KPI for this date.
forecast_value	float	The forecast value for the KPI for this date
measure_count	integer	The measure count for this date. Typically this value is used to divide the actual_value for reporting.
current_indicator	integer	Indicates that this row is the current row for a KPI. Typically the date of the current row is the current day.
tenant_id	integer	The tenant_id of the profile_variable that is the source of this KPI.

The fields in the process_kpi table are described in the following table.

Table 10. Fields in the process_kpi table

Field	Type	Description
process_id	integer	The process that is associated with the resource.
kpi_date	date	The date for which the KPI is calculated. The time grain for KPI calculation is a single day.
profile_variable_id	integer	The profile variable that is the source of this KPI.
material_id	integer	The material that is associated with the resource.

Table 10. Fields in the `process_kpi` table (continued)

Field	Type	Description
`event_code_id`	integer	The event code that is associated with the event observation. Event codes are codes for alarms, failures, issues, and so on. When an event arrives with a `measurement_type` value that has an `event_code_indicator` value of 1, the text from the `event_observation_text` field is assumed to contain an `event_code` value.
`location_id`	integer	The location that is associated with the resource.
`production_batch_id`	integer	The production batch that is associated with the event.
`actual_value`	float	The actual value for this KPI. It is important to understand that for Business Intelligence reporting purposes, this value is typically divided by the measure count. Even if the value is meant to be an average, this value must be a sum of the values from the resource and the `measure_count` must be the number of resources. `actual_value` field supports the average calculation for dimensional reporting.
`plan_value`	float	The planned value for the KPI for this date.
`forecast_value`	float	The forecast value for the KPI for this date.
`measure_count`	integer	The measure count for this date. Typically this value is used to divide the `actual_value` for reporting.
`current_indicator`	integer	Indicates that this row is the current row for a KPI. Typically the date of the current row is the current day.
`tenant_id`	integer	The `tenant_id` of the `profile_variable` that is the source of this KPI.

Profiles

Profiles provide preaggregated values to enable near real-time display in reports and dashboards.

The fields in the `resource_profile` table are described in the following table.

Table 11. Fields in the `resource_profiles` table

Field	Type	Description
`resource_id`	integer	The resource associated with this profile.
`profile_variable_id`	integer	The `profile_variable` that is the source of this profile.
`value_type_id`	integer	Value type of this profile. One of actual, plan, and forecast.
`event_code_id`	integer	The event code associated with the event observation. These are codes for alarms, failures, issues, and so on. When an event arrives with a `measurement_type` having an `event_code_indicator` of 1, the text from `event_observation_text` is assumed to contain an `event_code`.
`location_id`	integer	The location associated with the event.

Table 11. Fields in the `resource_profiles` table (continued)

Field	Type	Description
`profile_date`	datetime	This date is based on the timestamp of the most recent event used to update the profile.
`last_profile_date`	datetime	
`period_average`	float	The average value for the period.
`period_min`	float	The minimum value for the period.
`period_max`	float	The maximum value for the period.
`period_total`	float	The total value for the period.
`period_std_dev`	float	The standard deviation for the period.
`period_msr_count`	integer	The number of events contributing to this profile for the current period.
`prior_average`	float	The average value for the prior period.
`prior_min`	float	The minimum value for the prior period.
`prior_max`	float	The maximum value for the prior period.
`prior_total`	float	The total value for the prior period.
`prior_std_dev`	float	The standard deviation for the prior period.
`prior_msr_count`	integer	The number of events contributing to this profile for the prior period.
`ltd_average`	float	The average value lifetime to date.
`ltd_min`	float	The minimum value lifetime to date.
`ltd_max`	float	The maximum value lifetime to date.
`ltd_total`	float	The total value lifetime to date.
`ltd_std_dev`	float	The standard deviation lifetime to date.
`ltd_msr_count`	integer	The number of events contributing to this profile for the lifetime to date.
`last_value`	float	The most recent value in `event_observation.measurement` that updated this profile.
`tenant_id`	integer	The `tenant_id` of the `profile_variable` that is the source of this KPI.

Profile calculations

Profile calculations update both the key performance indicator (KPI) and profile table, indicated by a `profile_variable` (the `kpi_indicator` and `profile_indicator`) values are updated.

The following section describes the default profile calculations.

Measurement of type

This calculation is based on the value of a particular `measurement_type`.

- KPI: the `actual_value` column contains the sum of all the `event_observation.measurement` values. The `measure_count` column contains a count of all the `event_observation` events.
- Profile: the average, minimum, maximum, total, and standard deviations are calculated for the present, prior (previous day), and lifetime to date periods. The

average value in the profile is the true average and, unlike KPI, is not divided by the corresponding msr_count value. These values can be calculated on a running basis for efficiency. The msr_count values record the count of all the event_observation events in the period. The last_value column contains the most recent event_observation.measurement values.

Event of type count

A count of the number of times for a measurement_type value that an event with a particular event_type value occurs. The event type is specified in the profile_variable.comparison string. This value corresponds to an event_type.event_type_cd value.

- KPI: the actual_value and measure_count columns contain a count of the occurrences of the specified event_observation events.
- Profile: the msr_count values record the count of the event_observation events in the period.

Measurement of type count

A count of the number of times an event with a particular measurement_type occurs.

- KPI: the actual_value and measure_count columns contain a count of the occurrences of the specified event_observation.
- Profile: the msr_count values record the count of the event_observation events in the period.

Measurement in range count

A count of the number of times the event_observation.measurement value falls within the range of the high and low values of the profile variable, high_value_number, low_value_number.

- KPI: the actual_value and measure_count columns contain a count of the occurrences of the specified event_observation events.
- Profile: The msr_count values record the count of the event_observation events in the period.

Measurement text contains count

A count of the number of times an event observation text contains a string. The string is the value of the profile_variable.comparison_string.

- KPI: the actual_value and measure_count columns contain a count of the occurrences of the specified event_observation events.
- Profile: the msr_count values record the count of the event_observation events in the period.

Interval calculation

The amount of time between the occurrence of two event observations. This calculation requires that a profile is calculated. The value of resource_profile.profile_date, or process_profile.profile_date, contains the timestamp of the previous event observation and is used to calculate the interval value in minutes.

- KPI: the actual_value column contains a sum of all the interval values. The measure_count column contains a count of all the event_observation events.

- Profile: the average, minimum, maximum, total, and standard deviations of the interval value is calculated for the present, prior (previous day), and lifetime to date periods. The average value in the profile is the average and is not divided by the corresponding `msr_count` value. These values are calculated on a running basis for efficiency. The `msr_count` values record the count of all the `event_observation` events in the period. The `last_value` column contains the most recent interval.

Last date of event type

The most recent date on which an event with a particular `event_type` occurs. The event type is specified in `profile_variable.comparison` string. This value corresponds to an `event_type.event_type_cd` value.

- KPI: the `actual_value` column contains a value of 1 if the `event_observation` event occurred. The `measure_count` column contains a value of 1.
- Profile: the `msr_count` value is 1 if the `event_observation` event occurs in the period. `profile_date` has the timestamp of the most recent `event_observation` event.

Last date of measurement type

The most recent date on which an event with a particular measurement type occurred.

- KPI: the `actual_value` column contains a value of 1 if the `event_observation` event occurred. The `measure_count` column contains a value of 1.
- Profile: the `msr_count` value is 1 if the `event_observation` event occurs in the period. `profile_date` has the timestamp of the most recent `event_observation` event.

Last date of measurement in range

The most recent date on which an event that has an `event_observation.measurement` value that falls within the range of the high and low values of the profile variable occurred (`high_value_number`, `low_value_number`).

- KPI: the `actual_value` column contains a value of 1 if the `event_observation` has occurred. The `measure_count` column contains a value of 1.
- Profile: the `msr_count` value is 1 if the `event_observation` event occurs in the period. `profile_date` has the timestamp of the most recent `event_observation`.

Lifespan analysis

This is a special calculation to support lifespan analysis predictive scoring.

- KPI: the `actual_value` column contains the most recent measurement value. The `measure_count` column contains a count of all the `event_observation` events.
- Profile: the average, minimum, maximum, total, and standard deviations are calculated for the present, and lifetime to date periods. The `msr_count` values record the count of all the `event_observations` in the period. The `last_value` column contains the most recent `event_observation.measurement`.

Lifespan analysis failure

This is a special calculation to support lifespan analysis predictive scoring.

- KPI: a KPI is not calculated.

- Profile: the msr_count value is 1 if the event_observation event occurs in the period. profile_date has the timestamp of the most recent event_observation event.

Measurement above limit

This is a count of the number of times the event_observation.measurement value falls above the value of the profile variable (high_value_number).

- KPI: the actual_value and measure_count columns contain a count of the occurrences of the specified event_observation.
- Profile: the msr_count values record the count of the event_observation events in the period.

Measurement below limit

This is a count of the number of times the event_observation.measurement value falls below the value of the profile variable (low_value_number).

- KPI: the actual_value and measure_count columns contain a count of the occurrences of the specified event_observation events.
- Profile: the msr_count values record the count of the event_observation events in the period.

Measurement delta

This is the change from one measurement value to the next.

- KPI: the actual_value column contains a sum of all the changes in measurement values. The measure_count column contains a count of all the event_observation events.
- Profile: The msr_count value should be 1 if the event_observation event occurs in the period. The profile_date value has the timestamp of the most recent event_observation event.

Custom calculations

You can modify the event processing flow to support extra calculations.

A context is provided for the implementation of custom calculations. This context provides access to the event being processed, the event store, and the key performance indicator (KPI) and profile tables. The context also provides the parameters present in the profile_variable referencing the calculation.

Predictive scoring

To provide a health score for predictive models, code is required in the event processing flow.

A scoring service requires a defined set of inputs and returns a result. The score returns either a numerical value, a recommendation, or both. The sources of data for the input to the scoring service are the Event, KPI (key performance indicator), and Profile tables. Code transforms the data that is needed to supply the exact set of input parameters that are required by the scoring service. The scoring service is called by a web service call from IBM WebSphere Message Broker.

When the results are returned from the scoring service, they are written back as new events. Measurement types and profile variables can be defined for these events.

For example, a health score and recommendation can be recorded as an `event_observation.measurement` and `event_observation.observation_text`. In addition to being stored in the event tables, this score and recommendation can be aggregated for IBM Cognos Business Intelligence Reporting by defining two `profile_variables`.

To aggregate the health score, define a `profile_variable` with the Measurement of Type calculation.

To aggregate the occurrences of a specific recommendation, define a `profile_variable` with the Text Contains Count calculation and set the `comparison_string` attribute of `profile_variable` to the name of the recommendation. Additionally, define a `profile_variable` with Measurement of Type Count for the health score.

The processing of an event that contains the result of a predictive scoring service can invoke a second scoring service.

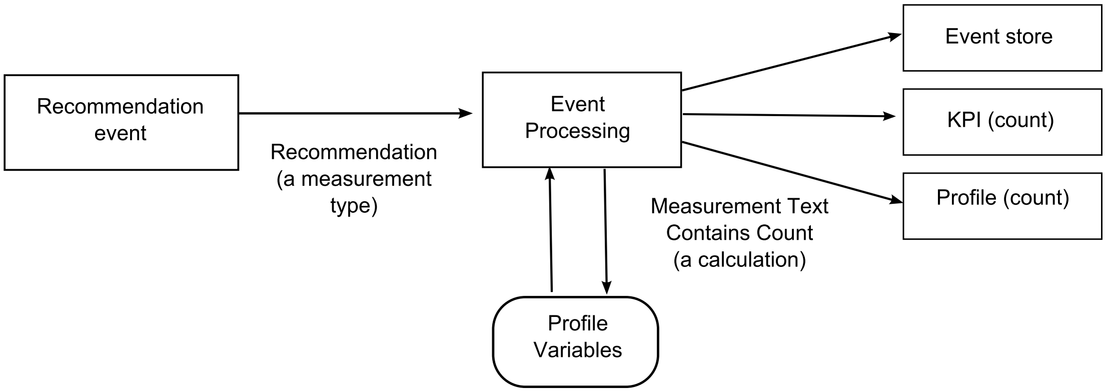


Figure 6. The flow of a scoring service

How scoring is triggered

Scoring for predictive models is triggered based on following conditions:

- Invoke scoring for resource type ASSET
- Invoke scoring for events with value type ACTUAL
- Invoke scoring for all events except when source system is PREDMAIT

Based on the design of the predictive model, conditions can be modified or added to trigger scoring, using IBM SPSS Modeler.

Events and actual, planned, and forecast values

Typically, events contain actual values. Special events may contain planned values, and forecast values.

At least one event containing planned or forecast values, should be supplied for each KPI reporting period (day). This allows planned and forecast values to appear on IBM Cognos Business Intelligence reports along with actual values.

Event processing queue

Two queues are used to gather events for processing. One queue is for events read from .csv files, or transformation flows that you have developed. The other queue is for events that are generated from scoring results. You can use additional queues for processing but only one queue can contain events that update the same Key Performance Indicators (KPI) or profile rows. Typically, a queue supports events from an exclusive set of resources or processes.

A queue is used to hold events for processing in a single thread. The queue only contains events already mapped to the IBM Predictive Maintenance and Quality format.

Event processing

Event processing consists of the following steps.

1. Lookup of primary keys for supplied business keys.
2. Insertion of events.
3. Lookup of profile variables.
4. Updating and inserting the KPI and Profile rows.
5. Scoring using an IBM SPSS predictive model.
6. Making a recommendation using IBM Analytical Decision Management.
7. Work order creation.

Recording and acting on predictive scores and recommendations

Profile variables are used to determine what key performance indicators (KPI) and profile calculations must be performed for an event. However, profile variables do not determine whether scoring or decision management is performed for an event. Scoring or decision management is determined by ESQL code and this code must be modified to provide customized scoring and decision making.

Scores that are returned by a predictive model and recommendations that are returned by decision management are processed and recorded in the same way as events that are received from a device. This means that scores and recommendation results are written to a content store, KPIs and profiles are calculated for these values, and the values are displayed in reports.

This reuse of the event processing mechanism is implemented by creating an event that uses the standard event format. An appropriate event type and measurement type are used for the event. The event is then placed on an internal event processing queue (`PMQ.EVENT.INTEGRATION.IN`). Events on this internal event processing queue are processed by the same flow as external events. Profile variables are defined to control the processing of these internal events to calculate KPI and Profile values.

If IBM Predictive Quality and Maintenance is configured to work with IBM Maximo Asset Management, a recommendation can result in the creation of a work order in IBM Maximo. Customizing this behavior also requires modifying ESQL code.

For more information, see Chapter 6, “Recommendations,” on page 45

Threads

Events are processed by only one flow that runs in a single thread. If more than one flow is implemented to process events, these different flows must not update the same KPI or Profile rows. A single thread is required to ensure that only a single thread is calculating and updating a row in the KPI and Profile tables.

Batch processing

Event processing can occur faster by processing more than one event at the same time through batch processing. For example, if you want to process and load event data for one year, then you can do so by processing the events through multiple .csv files.

Use this approach only if the separate files contain events from separate devices.

- Create copies of `MultiRowEventLoad` flow and deploy on the broker. Each copy of message flow processes one .csv file at a time.
- Ensure that you do not set the `AdditionalInstances` property of the `MultiRowEventLoad` flow to greater than 0 for processing the batch concurrently.
- Ensure that the events from the same resource are combined into a single file in chronological order.

Parallel processing

Event processing can also occur faster by processing more than one event at the same time. However, it is important that only one thread at a time updates a row in the KPI or profile tables. Since the rows in these tables are related to resources and measurement types, achieve thread isolation by ensuring that events from an individual resource or of a particular measurement type are processed by a single thread. You can implement parallel processing by using multiple queues to manage the separation of events.

Event processing assumes that only one thread updates an individual row in the `resource_kpi`, `resource_profile`, `process_kpi`, `process_profile`, and `material_profile` tables. This is true for events from external devices and internal events that record recommendations. This means that parallelism can be achieved only by segmenting the events into groups that do not share resources, processes, or materials. To achieve parallelism, you must deploy multiple copies of event and integration flows, and ensure that each copy of the message flow uses a unique set of queues.

Remove events

Normally events are not deleted from the analytic database. During testing and development, events can be removed by deleting the appropriate rows from the `event`, `event_observation`, and `event_resource` tables.

As events are processed, extra internal events are added when predictive scoring and decision management are performed. You can also remove these events.

Sample remove event code

The following SQL code is an example and must be modified.

```
DELETE FROM SYSREC.EVENT_RESOURCE ER WHERE...
DELETE FROM SYSREC.EVENT_OBSERVATION EO WHERE...
DELETE FROM SYSREC.EVENT E WHERE...
```

Event processing also adds row to the KPI and profile tables, and you can remove those rows by modifying the following SQL.

```
DELETE FROM SYSREC.RESOURCE_KPI RK WHERE...
DELETE FROM SYSREC.RESOURCE_PROFILE RP WHERE...
DELETE FROM SYSREC.PROCESS_KPI PK WHERE...
DELETE FROM SYSREC.PROCESS_PROFILE PP WHERE...
DELETE FROM SYSREC.MATERIAL_PROFILE MP WHERE...
```

Chapter 5. Predictive models

Use predictive models to generate the information you need to make informed operational, maintenance, repair, or component replacement decisions.

You can use a single predicted output to drive other predicted outputs. For example, the likelihood of failure can be used to predict the lifetime of an asset.

The following models form the basis of the predictive models in IBM Predictive Maintenance and Quality.

- Health score, also known as failure analysis
- Lifespan analysis

Asset anomaly detection is also available with IBM Predictive Maintenance and Quality, however, this is calculated with IBM WebSphere Message Broker, and IBM Cognos Business Intelligence.

Sample predictive models are provided. For more information, see "IBM SPSS Artifacts" on page 96.

Health score

The health score is the average score for all assets or processes at a site. The health score determines how likely an asset or process is likely to fail.

The health score model continuously monitors machine or asset health, and predicts potential machine faults in real time. It uses historical defect data, operational information, and environmental data to determine the current health of an asset. The health score model can also be used to predict the future health of an asset.

The value of the health score of an asset is in between 0 and 1, the higher the value of the health scores, the better the health of the asset. If the input data model/structure is modified, the health score model must be retrained on the new data.

The health score model is based on the IBM SPSS Modeler regression model. Depending upon the input data, you might need to consider a different approach for the health score modeling.

Lifespan analysis

Lifespan analysis estimates a product's life when it functions in a real-world scenario. The lifespan can be measured in hours, miles, stress cycles, or any other metric with which the life or exposure of a product can be measured. Data on the functioning condition of this product is collected from laboratory experiments.

Using lifespan analysis, you can calculate the expected future lifespan given that the parts are up and running at a specific time point. Lifespan analysis models time-to-failure event data.

Lifespan analysis is an offline back-end process and can be done on demand, or at regular intervals.

Lifespan analysis must be done on selected data samples on which lab experiments have run.

Outputs are the likely age (lifespan) of the product under the current product scenario. Reliability measures, such as failure rates can be included.

Example: Lifespan of manufactured items

Manufactured items with mechanical or electric components are often subjected to lifespan tests to obtain information about their durability. Lifespan tests involve putting items in operation, often in a laboratory setting, and observing them until they fail. Lifespans are also known as failure times because when an item ceases operating satisfactorily, it is said to have failed.

Example: Lifespan of LEDs

Solid state devices such as LEDs have low wear and tear if operated at low currents and at low temperatures. Many of the LEDs made in the 1970s and 1980s are still in service today. Typical lifespans that are quoted are 25,000 - 100,000 hours, but heat and current settings can extend or shorten this time significantly. Lifespan analysis can help to identify the expected life time of LEDs.

Training and scoring predictive models

Predictive models must be trained to determine which data is useful and which data is not needed. When a model is giving you accurate predictions, it can be used to score real-time data.

You train a model by using historical data. Historical measurement data is pre-aggregated, run through predictive models in IBM SPSS Modeler, trained to determine which data is useful and which data is not needed. When the model gives you accurate predictions, you are ready to create a runtime predictive model. This process generates the scores that are used in scorecards and helps you to predict, for example, the likely failure of a process or asset.

Models must be retrained periodically to adjust for changing behavior patterns.

For information about using IBM SPSS Modeler, see *IBM SPSS Modeler 15 User's Guide*.

Sample training stream for model development

A regression model is used to determine the probability of an asset to fail. This model results in a `HEALTH_SCORE` value.

The sample data file includes a field named FAIL. For training, a new **Boolean** field is derived as `IS_FAIL`, which is set in the type node as the target field or value, that you want to predict. The other fields contain information such as temperature, relative humidity, RPM values, and can be used as predictors.

The following diagram shows an example training stream.

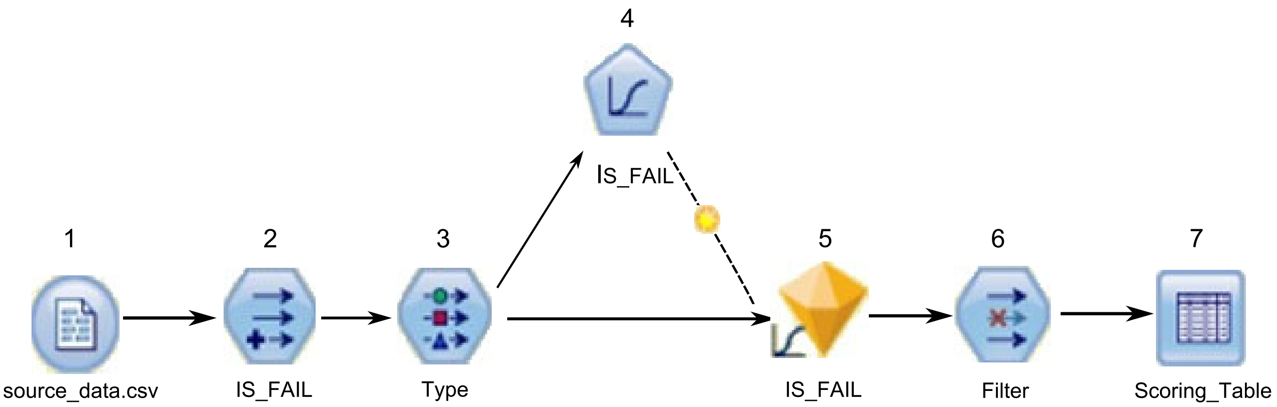


Figure 7. Example training stream

The following list is a key to the training stream diagram:

1. Read the input data. Using different source nodes, read the input data that is required for model development.
2. Define the dependent or target variable for which you are creating the predictive model by using the derive node.
3. Define the roles of all the fields for the modeling by using a type node.
4. Choose the appropriate modeling node.
5. When the modeling node is run, a reusable golden model nugget is generated.
6. Filter the final output to keep just the fields that you require by using the filter node.
7. Store or export the output into the required format.

Sample scoring stream

A scoring data file contains the same fields as a training data file, except that the `IS_FAIL` flag is not set as the target field. After training models using a dataset where the `IS_FAIL` value is known, you can score records where the `IS_FAIL` value is not known by using the nugget that is generated during the training process.

The following diagram shows an example scoring stream.

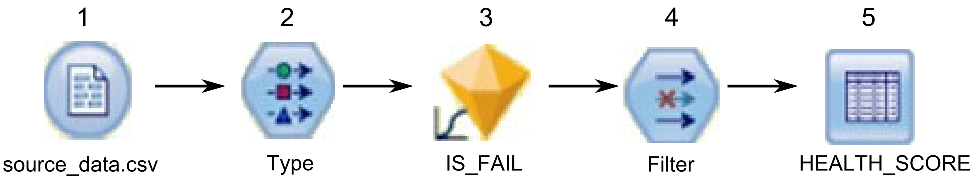


Figure 8. Example scoring stream

The following is a key to the scoring stream diagram:

1. Read the scoring data. Using different source nodes, read the scoring data.
2. Define the roles of all the fields by using a type node.
3. Connect the reusable golden model nugget that is generated in the model development phase with the type node so that it scores the data.
4. Filter the final output to keep just the fields that you require by using the filter node.
5. Store or export the output into the required format.

Deploying the scoring stream

Deploy a scoring stream using the IBM SPSS Collaboration and Deployment Services repository.

Procedure

1. Select a model to use for scoring from the IBM SPSS Collaboration and Deployment Services Repository.
2. Define a scoring configuration for the model.
3. Supply the configured model with data, and generate scores.

Results

You can view the scores in The IBM SPSS Collaboration and Deployment Services Deployment Portal, or in custom client applications.

Chapter 6. Recommendations

When an asset or a process is scored and identified as having a high probability of failure, recommendations can be generated.

Define recommended actions by using rules in IBM Analytical Decision Management. Use IBM Analytical Decision Management to understand the drivers that are used to define the rules, and to determine what happens based on the scores received. For example, if a score breaches a threshold, what is the resulting action? You can automate alerts for recommended actions by integrating with other systems or by defining a routing rule to send emails. Depending on the manufacturing execution systems (MES) you use, the recommendation can be acted on automatically. You can also predict the success rate of the corrective action that is based on previous actions.

For information about using IBM Analytical Decision Management, see the IBM Analytical Decision Management 7 Information Center (http://pic.dhe.ibm.com/infocenter/spssdm/v7r0m0/index.jsp).

When IBM Predictive Maintenance and Quality generates recommendations, for example, to inspect an asset, you can configure the system so that the recommendation results in a work order that is created by IBM Maximo. The work order is populated with the information needed to complete the task, for example, a device identifier and a location.

Two IBM Analytical Decision Management templates are provided with IBM Predictive Maintenance and Quality:

- Prioritize application template
- Combine application template

Prioritize application template

Use the prioritize application template when you have a good understanding of the predictive analytics scores, and the interaction between the predictive scores. You can use this template to prioritize your business objective that is based on, for example, profit maximization, or downtime minimization.

The template is stored in the following location: `/opt/IBM/SPSS/Deployment/5.0/Server/components/decision-management/Templates/PredictiveMaintenanceQuality.xml`

This template contains the following information that can be customized:

- Input source data: contains the health score and expected device life time data from the IBM SPSS Modeler stream output. Additionally, it contains the calculations such as Mean, Minimum, Maximum values for a particular resource for a specific time stamp.
- Defined rules: resource recommendations are given based on the rules defined. The recommended actions are classified as **Urgent inspection**, **Need monitoring**, or **Within limits**.
- Prioritization: you can define the optimization objective for the business, for example, “profit maximization” or “downtime/loss minimization”.

Combine application template

Use the combine application template to use existing rules along side new predictive scores. This is useful if there are many rules that you do not want to replace with new predictive scores straight away. You can define a precedence structure for these rules to enable them to co-exist.

The template is stored in the following location: `/opt/IBM/SPSS/Deployment/5.0/Server/components/decision-management/Templates/PredictiveMaintenance.xml`

This template contains the following information that can be customized:

- Input source data: contains the health score and expected device life time data from the IBM SPSS Modeler stream output. Additionally, it contains calculations such as Mean, Minimum, Maximum values for a particular resource for a particular time stamp.
- Defined rules: logic-based business rules with appropriate risk points.
- Combine: specify the precedence order when the actions from the business rules and the model do not match.

Preventing scoring for incoming events

You can prevent scoring from being performed by IBM SPSS for incoming events. If you require IBM Maximo work order creation, you must not prevent scoring. By default, scoring is enabled (**SPSSTRIGGER** is set to `TRUE`).

Procedure

1. In the IBM WebSphere MQ Explorer, expand the **Brokers** node, the **MB8Broker** node, the **PMQ1** node, the **PMQEventLoad** node, right-click the **StdEventLoad** item, and click **Properties**.
2. Click **User Defined Properties**.
3. Set the **SPSSTRIGGER** property to `FALSE`. To re-enable scoring, set the **SPSSTRIGGER** property to `TRUE`.

Disabling work order creation

If IBM Maximo is not integrated with your IBM Predictive Maintenance and Quality installation, or if you want to disable work order creation, do the following steps:

Procedure

1. In the IBM WebSphere MQ Explorer, go to **Brokers** > **MB8Broker** > **PMQ1**. Right-click the **PMQIntegration** node, and click **Properties**.
2. Click **User Defined Properties**.
3. Set the **MaximoTRIGGER** value to `FALSE`. To re-enable work order creation, set the **MaximoTRIGGER** property to `TRUE`. By default, the **MaximoTRIGGER** property is set to `FALSE`.

Chapter 7. Reports and dashboards

You can customize and extend the reports and dashboards that are supplied with IBM Predictive Maintenance and Quality. You can also add more reports and dashboards to the menu.

IBM Cognos Report Studio is used to create scorecards, and reports, and is a powerful tool with a wide range of functions. For more information about how to use Report Studio, see *IBM Cognos Report Studio User Guide*. You can obtain this user guide from the IBM Cognos Business Intelligence information center, (http://pic.dhe.ibm.com/infocenter/cbi/v10r2m0/index.jsp).

The metadata for these reports can be modified by using IBM Cognos Framework Manager. For more information, see Appendix C, “IBM Cognos Framework Manager model description,” on page 85.

The following table describes the reports available from the Site overview dashboard.

Table 12. Reports available from the Site overview dashboard

Reports	Description
Overview	Provides a high-level summary of the health of all of your assets at all sites, it shows the key performance indicators (KPIs) with the greatest impact. You can change the detail that is displayed by selecting items from the list boxes. For example, you can change the date and the equipment type.
Top 10 Contributors	Identifies the equipment, locations, and operators responsible for the most failures.
KPI Trending	You can select multiple key performance indicators (KPIs) to be plotted side-by-side in a line chart. You can identify correlations between the KPIs and see whether there is any lagging behavior. For example, if there is a spike in one KPI, how long does it take to impact the other KPIs?
Actual vs Plan	You can monitor how closely the metrics track against the plan. Variances are highlighted.
Equipment Listing	The health score for a site is derived from the lower-level scores from each piece of equipment in the site. This report shows you all the pieces of equipment on the site and the health scores and relevant KPIs for that equipment.
Equipment Outliers	Lists the equipment (or assets) that are performing outside of allowable limits. The measures that are shown differ depending on the equipment, but examples are operating temperature, lateral strain, hydraulic pressure, average value, last value, and control limits.
Recommended actions	A summary of all recommended actions for each piece of equipment, for the health score measurement.

The following table describes the reports that are available from the Equipment reports dashboard.

Table 13. Reports available from the Equipment reports dashboard

Reports	Description
Equipment Profile	A detailed report that shows everything that is known about a piece of equipment: how it is performing today and how it performed in the past.
Equipment Control Chart	Shows the upper and lower control limits and the average limits for selected measures.
Equipment Run Chart	Shows the measures for a particular piece of equipment.
Equipment Outliers	Shows detailed measures for a piece of equipment that shows anomalies.
Event Type History	Lists the events for a device.

The following table describes the reports that are available from the Product quality dashboard.

Table 14. Reports available from the Product quality dashboard

Reports	Description
Defect Analysis	Shows product defects and inspection rates.
Inspection Rate Analysis	Examines the relationship between inspections and defects over time to find the optimal rate of inspection.
Material Usage By Process	Provides an overview of material usage in the production processes.

The following table describes reports that are also available.

Table 15. Other reports that are available

Reports	Description
Audit Report	Shows the counts of rows in the major master data tables.
Resource List	Lists the resources by resource type.
Profile Variables	Lists all measures and key performance indicators that are being tracked in daily profiles and historical snapshots.
Process List	Lists all production processes.
Material List	Lists materials that are used in the production process.
Production Batch List	Lists production batches.
Material Usage By Production Batch	This report provides an overview of material usage by production batch. By correlating production batches with defects to material usage by production batch, the impact of defective materials can begin to be traced.

Site overview dashboard

The Site overview dashboard provides a high-level summary of the health of all of your assets at all sites, showing the key performance indicators (KPIs) with the greatest impact.

The Site overview dashboard is made up of a number of reports: the Site summary report, Health score trend bar chart, Health score contributors pie chart, and the Incident/recommendation analysis bar chart

Site summary report

The following table describes the measures for the **Site summary** report.

Table 16. Site summary measures

Measures	**Description**
Health Score	The evaluation of the health of a resource that is based on predictive models.
Lifespan Analysis (in Days)	The predicted remaining lifespan of a resource.
Resource Count	The number of resources.
Incident Count	Counts the number of failures that are recorded by resources.
Alarm Count	Counts the number of alarms that are issued by resources.
Recommendation Count	A recommendation is issued by the predictive model when a resource approaches a failure. This measure counts the number of recommendations issued.
MTTR (Mean time to repair)	Average time, for example, in hours between the occurrence of an incident and its resolution, which is calculated by using the following calculation: `Repair Time / Repair Count`
MTBF (Mean time between failures)	The average time between equipment failures over a period. For example, the average time a device functions before it fails. It is the reliability rating that indicates the expected failure rate of equipment. It is calculated by using the following calculation:`Operating Hours Delta / Incident Count`

The measure source is the `resource_kpi` table. The measures are displayed by the Location hierarchy.

The following prompt filters are applied to this report:
- Date
- Location
- Resource sub type

Health score trend bar chart

The Health score trend bar chart uses the Health score measure. The measure source is the `resource_kpi` table. The measure is displayed by the Calendar hierarchy.

The following prompt filters are applied to this report:
- Date
- Location
- Resource sub type

Health score contributors pie chart

The Health score contributors pie chart uses the Health score measure. The measure source is the `resource_kpi` table. The measure is displayed by the Resource hierarchy.

The following prompt filters are applied to this report:

- Date
- Location
- Resource sub type

Incident/recommendation analysis bar chart

You can use this report to analyze incidents and recommendations. You can drill through to the Incident/recommendation event list.

The following prompt filters are applied to this report:

- Date
- Location
- Resource sub type

The following table describes the measures for the **Incident/recommendation analysis** bar chart. The measure source is the `resource_kpi` table. The measures are displayed by the Calendar hierarchy.

Table 17. Incident analysis bar chart

Measures	Description
Recommendation Count	A recommendation is issued by the predictive model when a resource approaches a failure. This measure counts the number of recommendations that issued.
Incident Count	Counts the number of failures that are recorded by resources.

Top 10 Contributors dashboard

The Top 10 Contributors dashboard identifies the top 10 equipment, locations, and operators responsible for the most failures.

The following table indicates which dimension is used to display the Actual Value measure in each report.

Table 18. Dimensions that display the Actual Value measure in Top 10 Contributors reports

Report	Dimension
Top 10 Contributors by Resource	Resource
Top 10 Contributors by Location	Location
Top 10 Contributors by Organization	Group Dimension

The Actual Value measure is the aggregated value of an actual event measurement. Depending on the aggregation type of the Profile Variable, it can be calculated by using the following calculation: `sum (Actual Value) / sum (Measure Count) or sum (Actual Value)`

The measure source is the `resource_kpi` table. The measures are displayed by the Location hierarchy.

The following prompt filters are applied to this report:
- Date
- Location
- Resource sub type
- Profile variable

KPI trending report

Users can select multiple key performance indicators (KPIs) to be plotted side-by-side in a line chart. You can identify correlations between the KPIs and see whether there is any lagging behavior. For example, if there is a spike in one KPI, how long does it take to impact the other KPIs?

The KPI trending report uses the Actual Value measure, which is the aggregated value of an actual event measurement. Depending on the aggregation type of the Profile Variable, it can be calculated by using the following calculation: `sum(Actual Value) / sum (Measure Count) or sum (Actual Value)`. The measure is displayed by the Calendar hierarchy, and the measure source is the `resource_kpi` table.

The following prompt filters are applied to this report:
- Date
- Location
- Resource sub type
- Profile variable

Actual vs plan report

This report monitors how closely the metrics are tracking against the plan. Variances are highlighted when a metric is off-track.

The following table describes the measures, and the measure source for the **Actual vs plan** report.

Table 19. Measures and the measure source in the **Actual vs plan** *report*

Measures	Measure description	Measure source
Plan Last Value	The last recorded planned value for the resource. "Planned" is determined by the Value Type	`resource_profile` table
Actual Last Value	The last recorded actual value for the resource. "Actual" is determined by the Value Type	Resource Profile
Variance	Plan value - Actual Value	Report calculation

The following prompt filters are applied to this report:
- Date
- Location
- Resource sub type
- Profile variable

Equipment listing report

The health score for a site is derived from the lower-level scores from each piece of equipment in the site. Use this report to view all the pieces of equipment on the site and the health scores and relevant key performance indicators (KPIs) for that equipment.

The following table describes the measures for the Equipment listing report. The measure source is the `resource_kpi` table. The measures are displayed by the Resource hierarchy.

Table 20. Measures in the Equipment listing report

Measures	**Description**
Health Score	The evaluation of the health of a resource based on predictive models.
Work Order Count	This counts the number of work orders issued. A Work Order is a separate event type from resource measurements.
Incident Count	This counts the number of failures recorded by resources.
Recommendation Count	A recommendation is issued by the predictive model when a resource approaches a failure. This measure counts the number of recommendations that have been issued.
MTBF (Mean time between failures)	The average time between equipment failures over a given period, for example, the average time a device will function before failing. It is the reliability rating indicating the expected failure rate of equipment. It is calculated using the following calculation: `Operating Hours Delta / Incident Count`.
MTTR (Mean time to repair)	Average time (for example, in hours) between the occurrence of an incident and its resolution, calculated using the following calculation: `Repair Time / Repair Count`

The following prompt filters are applied to this report:

- Date
- Location
- Resource sub type

Outliers report

This reports lists the equipment or assets that are performing outside of allowable limits.

The following table provides the measure details for the Outliers report.

Table 21. Measures in the Outliers report

Measures	**Measure description**	**Measure source**
Life to Date Average	The daily average measurement for the resource.	Resource Profile

Table 21. Measures in the Outliers report (continued)

Measures	Measure description	Measure source
Upper Control Limit	This is calculated using the following calculation: `[Life-to-Date Average] + [Sigma Level] * [Life-to-Date Standard Deviation]`	Report calculation
Lower Control Limit	This is calculated using the following calculation: `[Life-to-Date Average] - [Sigma Level] * [Life to Date Standard Deviation]`	Report calculation
Last Value	The most recent recorded measurement for this resource.	Resource Profile

The following prompt filters are applied to this report:

- Date
- Location
- Resource sub type
- Sigma level

Recommended actions report

This report summarizes all recommended actions for each piece of equipment.

The Recommended actions report uses the Health Score measure, which is the evaluation of the health of a resource that is based on predictive models. The measure is displayed by the Event Observation Resource hierarchy, and the measure source is the `event` table.

The following prompt filters are applied to this report:

- Date
- Location
- Resource sub type

Equipment reports dashboard

The Equipment reports dashboard gives you access to the Equipment profile report, the Equipment control chart, the Equipment run chart, the Equipment outliers chart, and Event type history chart.

Equipment profile report

The Equipment profile report is a detailed report that shows everything that is known about a piece of equipment: how it is performing today and how it performed in the past.

The following table provides the measure description for the **Equipment profile** report. The measure source is the `resource_profile` table. The measures are displayed by the Profile Variable hierarchy.

Table 22. Measures in the Equipment profile report

Measures	Measure description
Period Minimum	The lowest actual reading that is recorded for the resource measurement this period.
Period Maximum	The highest actual reading that is recorded for the resource measurement this period.
Period Average	The daily average measurement for the resource.
Last Value	The most recent recorded measurement for this resource.
Period Total	The total actual reading that is recorded for the resource measurement this period.

The following prompt filters are applied to this report:

- Resource sub type
- Resource name
- Resource serial number
- Location
- Event code

Equipment control chart

The Equipment control chart shows the upper and lower control limits and the average limits for selected measures.

The following table provides the measure details for the **Equipment control chart** report.

Table 23. Measures in the Equipment control chart

Measures	Measure description	Measure source
Life to Date Average	This is an average measurement that is calculated over the life of the resource.	`resource_profile` table
Upper Control Limit	This is calculated by using the following calculation: `[Life-to-Date Average] + [Sigma Level] * [Life-to-Date Standard Deviation]`	Report calculation
Lower Control Limit	This is calculated by using the following calculation: `[Life-to-Date Average] - [Sigma Level] * [Life-to-Date Standard Deviation]`	Report calculation
Measurement	The actual value recorded on an event.	`event` table

The following prompt filters are applied to this report:

- Resource sub type
- Resource name
- Resource serial number
- Location
- Event code

- Calendar date
- Start time
- End time
- Measurement type
- Profile variable
- Sigma level

Equipment run chart

Shows the measures for a particular piece of equipment.

The Equipment run chart uses the Measurement measure, which is the actual value that is recorded on an event. The measure source is the event table, and the measure is displayed by the event time hierarchy.

The following prompt filters are applied to this report:

- Resource sub type
- Resource name
- Resource serial number
- Location
- Event code
- Calendar date
- Start time
- End time
- Measurement type

Equipment outliers

Shows detailed measures for a piece of equipment that shows anomalies.

The following table describes the measures for the Equipment outliers report. The measures are displayed by the Profile Variable hierarchy

Table 24. Measures in the Equipment outliers report

Measures	Measure description	Measure source
Life to Date Average	This is an average measurement calculated over the life of the resource.	`resource_profile`
Upper Control Limit	This is calculated by using the following calculation: `[Life-to-Date Average] + [Sigma Level] * [Life-to-Date Standard Deviation]`	Report calculation
Lower Control Limit	This is calculated by using the following calculation: `[Life-to-Date Average] - [Sigma Level] * [Life-to-Date Standard Deviation]`	Report calculation
Last Value	The most recent recorded measurement for this resource.	`resource_profile`

The following prompt filters are applied to this report:

- Resource sub type
- Resource name
- Resource serial number
- Location
- Event code

Event type history report

This report lists the events for a device.

The Event type history report uses the Measurement measure, which is the actual value that is recorded on an event. The measure source is the `event` table, and the measure is displayed by the Event Time, Measurement Type and Event Observation.

The following prompt filters are applied to this report:

- Resource sub type
- Resource name
- Resource serial number
- Location
- Event code
- Calendar date
- Event type

Product quality dashboard

The Product quality dashboard highlights areas of the production process that are affected by defects, and enables you to see if any relationships exist between the rate of inspections, and the rate of defects.

Defect analysis dashboard

The Defect analysis dashboard provides an overview of product defects and inspection rates. The dashboard is made up of a number of reports that analyze defects by event code, location, and production batch.

Defect summary

This report analyzes product defects and inspection rates.

The following table describes the measures for the **Defect summary** report. The measure source is the `process_kpi` table. The measures are displayed by the Product hierarchy.

Table 25. Measures in the Defect summary report

Measures	**Measure Description**
Defect Count	The number of defects reported.
Quantity Produced	The quantity produced.
Defect Rate	Calculated by using the following calculation: `Defect Count / Qty Produced`
Quantity Planned	The quantity that is expected to be produced.
Defect Target	The acceptable number of defects.

Table 25. Measures in the Defect summary report (continued)

Measures	Measure Description
Test Failure Rate	The rate of Test Failures over Number of Tests.
Target Defect Rate	The rate of Defect Target over Quantity Planned.
Inspection Time	The amount of time that is spent inspecting the product.
Assembly Time	The amount of time that is spent producing the product.
Inspection Time Rate	Calculated by using the following calculation: `Inspection Time / Assembly Time`.
Inspection Count	The number of inspections performed.
Inspection Rate	Calculated by using the following calculation: `Inspection Count / Quantity Produced`
Average Assembly Time	Calculated by using the following calculation: `Assembly Time / Quantity Produced`

The following prompt filters are applied to this report:
- Process hierarchy
- Calendar from/to date

Defects by event code

This pie chart shows product defects by event code, also known as failure code.

The Defects by event code report uses the Actual Value measure, which is the aggregated value of an actual event measurement. Depending on the aggregation type of the Profile Variable, it can be calculated by using the following calculation: `sum (Actual Value) / sum (Measure Count) or sum (Actual Value)`

The measure is displayed by the Event Code hierarchy, and the measure source is the `process_kpi` table.

The following prompt filters are applied to this report:
- Process hierarchy
- Calendar from/to date

Defects by location

This pie chart shows product defects by location.

The Defects by location report uses the Actual Value measure, which is the aggregated value of an actual event measurement. Depending on the aggregation type of the Profile Variable, it can be calculated by using the following calculation: `sum (Actual Value) / sum (Measure Count) or sum (Actual Value)`

The measure is displayed by the Location hierarchy, and the measure source is the `process_kpi` table.

The following prompt filters are applied to this report:
- Process hierarchy
- Calendar from/to date

Defects by production batch

This pie chart shows product defects by production batch.

The Defects by production batch report uses the Actual Value measure, which is the aggregated value of an actual event measurement. Depending on the aggregation type of the Profile Variable, it can be calculated by using the following calculation:

```
sum (Actual Value) / sum (Measure Count) or sum (Actual Value)
```

The measure is displayed by the Production batch hierarchy, and the measure source is the `process_kpi` table.

The following prompt filters are applied to this report:

- Process hierarchy
- Calendar from/to date

Inspection rate analysis

This report examines the relationship between inspections and defects over time in order to find the optimal rate of inspection.

It is made up of the Defect Summary report, the Defect plan vs actual bar chart, and the Defect rate vs inspection rate line chart.

Defect plan vs actual report

The following table provides the measure details for the Defect plan vs actual report. The measure is displayed by the Product hierarchy, and the measure source is the `process_kpi` table.

Table 26. Measures in the Defect plan vs actual report

Measures	Measure description
Defect Rate	Calculated by using the following calculation: `Defect Count / Qty Produced`
Target Defect Rate	The rate of the`Defect Target over Quantity Planned` measure.

The following prompt filters are applied to this report:

- Process hierarchy
- Calendar from/to date

Defect rate vs inspection rate line chart

The following table provides the measure details for the Defect rate vs inspection rate line chart. The measure is displayed by the Calendar hierarchy, and the measure source is the `process_kpi` table.

Table 27. Measures in the Defect rate vs inspection rate line chart

Measures	Measure description
Defect Rate	Calculated by using the following calculation: `Defect Count / Quantity Produced`

Table 27. Measures in the Defect rate vs inspection rate line chart (continued)

Measures	Measure description
Inspection Rate	Calculated by using the following calculation: `Inspection Count / Quantity Produced`

The following prompt filters are applied to this report:

- Process hierarchy
- Calendar from/to date

Material usage by process crosstab

This report provides an overview of material usage in the production processes.

This report includes the Defect Summary report.

This report uses the Period Measure Count, which is the number of measurements that are taken in one period. By default, a period is one day. The measure is displayed by the Material by Type, Supplier, and Batches by Product hierarchies, and the measure source is the `material_profile` table.

The Process hierarchy prompt filter id applied to this report:

Audit report

Shows the counts of rows in the major master data tables.

The following table provides a description of measures in the Audit report. The measure source is the report calculation.

Table 28. Measures in the Audit report

Measures	Measure Description	Hierarchies
Count of Resources by Type	A count of rows in the dimension	Resource by Type
Count of Materials by Type	A count of rows in the dimension	Material by Type
Count of Profile Variables	A count of rows in the dimension	Profile Variable
Count of Processes	A count of rows in the dimension	Process
Count of Production Batches by Product	A count of rows in the dimension	Batches by Product

Drill through lists

The following drill-through lists are also available:

- Resource list, displayed by the Resource by Type hierarchy
- Material list, displayed by the Material by Type hierarchy
- Process list, displayed by the Process hierarchy
- Production batch list, displayed by the Batches by Product hierarchy
- Profile variable list, as displayed by the Profile Variable hierarchy

Material usage by production batch

This report provides an overview of material usage by production batch. By correlating production batches with defects to material usage by production batch, you can begin to trace the impact of defective materials.

The material usage by production batch report uses the Period Measure Count, which is the number of measurements that are taken in one period. By default, a period is a day. The measure is displayed by the following hierarchies:

- Batches by Product
- Supplier
- Material by Type

The measure source is the `material_profile` table.

The following prompt filters are applied to this report:

- Process hierarchy
- Event code

Appendix A. The flat file API

Use the flat file application programming interface (API) to supply and modify IBM Predictive Maintenance and Quality master data.

The IBM Predictive Maintenance and Quality API supports the **upsert** operation.

The **upsert** operation attempts to update an existing row. If the matching row cannot be found, a new row is created that uses the values in the input record.

All the values for the row must be included even if only a single value of the row is being changed.

An isactive indicator is used to mark records as no longer in use (isactive = 0).

The isactive indicator is not used to take any decisions while loading the master or event data. For example, while loading a resource, if the associated location has the following indicator: isactive=0, that resource is loaded and associated with that location. Similarly if the event is reported by the resource with isactive=0, the event is processed and stored in the datastore.

Master data in the API

Use master data to supply IBM Predictive Maintenance and Quality with information about the context in which events occur.

The following records are supported by the master data section of the application programming interface (API). They are listed in alphabetical order but functionally they fall into one of four logical groups:

- Resource-related records include the location, resource, and resource_type records
- Process-related records include the batch_batch, process, product, and production_batch records
- Material-related records include the material and material_type records
- Other records can be related to both devices and processes. These records include the group_dim, source_system, and supplier records

No Delete operation is supported for master data. The upsert API can be used to mark a master data row as no longer active. In this case, the item in the row is no longer used in reports.

Load order

Some tables include references to rows in other tables. A row must be loaded before it can be referenced from another table.

The language and tenant tables must be loaded before any other data is loaded. The language_cd and tenant_cd rows are referenced in many tables. The values that are supplied for the language_cd and tenant_cd rows must reference rows already present in the language and tenant tables.

In addition, the rows of some tables refer to other rows in the same table, for example, parent rows. The referenced rows must be added before the rows that reference them.

Master files must be loaded sequentially.

The following table lists tables that contain references to other tables.

Table 29. Tables that must exist before other tables can be loaded

Table	Prerequisite tables
batch_batch	production_batch
material	material_type, supplier
process	process (parent process)
production_batch	product
resource	group_dim, location, resource (parent resource)
profile_variable	measurement_type, material_type

batch_batch

Creates a many-to-many relationship between production batches.

Use the **batch_batch** for batch traceability so that batches that share materials can be enumerated when a defect is found at any point. Every batch must relate to every batch in its lineage for full traceability.

For example, batch 1 splits into 2 and 3, and batch 3 splits into 4 and 5. **batch_batch** holds these pairs:

```
1,1  1,2  1,3  1,4  1,5   2,1  2,3  3,1  3,2  3,4  3,5  4,1  4,3  4,5  5,1  5,3  5,4
```

The fields in the **batch_batch** table are listed in the following table.

*Table 30. Fields in the **batch_batch** table*

Field	Type	Comments
production_batch_cd	string(50)	Required
related_production_batch_cd	string(50)	Required

batch_batch code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT PB1.PRODUCTION_BATCH_CD, PB2.PRODUCTION_BATCH_CD FROM
SYSREC.MASTER_BATCH_BATCH M JOIN SYSREC.MASTER_PRODUCTION_BATCH PB1 ON
M.PRODUCTION_BATCH_ID = PB1.PRODUCTION_BATCH_ID JOIN
SYSREC.MASTER_PRODUCTION_BATCH PB2 ON M.RELATED_PRODUCTION_BATCH_ID =
PB2.PRODUCTION_BATCH_ID;
```

event_code

Contains codes for alarms, failures, issues, and so on.

When an event arrives with a measurement type which has an event code indicator of 1, the text from the **event_observation_text** value is assumed to contain an event code. The measurement type of the event defines the **event_code_set** value.

The fields in the **event_code** table are listed in the following table.

Table 31. Fields in the **event_code** *table*

Field	Type	Comments
event_code_set	string(50)	Required
event_code_set_name	string(200)	Required
event_code	string(50)	Required
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

event_code code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.EVENT_CODE_SET, M.EVENT_CODE_SET_NAME, M.EVENT_CODE, L.LANGUAGE_CD,
T.TENANT_CD FROM SYSREC.MASTER_EVENT_CODE M JOIN SYSREC.MASTER_LANGUAGE L ON
M.LANGUAGE_ID = L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID =
T.TENANT_ID;
```

group_dim

Provides classifications for resources.

Up to five classifications are possible for each resource. The classifications vary depending on how IBM Predictive Maintenance and Quality is used. For example, a classification may be manufacturer or organization.

The fields for the **group_dim** table are listed in the following table.

Table 32. Fields in the **group_dim** *table*

Field	Type	Comments
group_type_cd	string(50)	Required
group_type_name	string(200)	Required
group_member_cd	string(50)	Required
group_member_name	string(200)	Required

Table 32. Fields in the **group_dim** *table (continued)*

Field	Type	Comments
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

group_dim code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.GROUP_TYPE_CODE, M.GROUP_TYPE_TEXT, M.GROUP_MEMBER_CODE,
M.GROUP_MEMBER_TEXT, L.LANGUAGE_CD, T.TENANT_CD FROM SYSREC.MASTER_GROUP_DIM M
JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID
JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

language

Contains the list of supported languages.

The fields in the **language** table are listed in the following table.

Table 33. Fields in the **language** *table*

Field	Type	Comments
language_cd	string(50)	Required. For example, EN
language_name	string(200)	Required. For example, English.
isdefault	0 or 1	Optional. A value of 1 indicates that this language is the default language for the system. No value, or a value of 0, indicates the language is not the default.

language code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line.

```
SELECT LANGUAGE_CD, LANGUAGE_NAME, DEFAULT_IND FROM SYSREC.MASTER_LANGUAGE;
```

New languages and tenants

After you have added new languages, or new tenants, you must populate the NA rows in the database for all new valid combinations of language and tenant. See the following example.

```
db2 "call SCHEMA.POP_NA( 'LANGUAGE_CD' , 'LANGUAGE_NAME' , 'TENANT_CD' , 'TENANT_NAME' )"
```

Where schema is a valid DB2 schema, such as db2inst1.

location

The location of a resource or event.

The location can be as specific, such as a room in a factory or general, such as a mine site.

The fields in the **location** table are listed in the following table.

Table 34. Fields in the **location** *table*

Field	Type	Comments
location_cd	string(50)	Required
location_name	string(200)	Required
region_cd	string(50)	Optional. The region_cd and region_name parameters must be supplied together.
region_name	string(200)	Optional
country_cd	string(50)	Optional. The country_cd and country_name parameters must be supplied together.
country_name	string(200)	Optional
state_province_cd	string(50)	Optional. The state_province_cd and state_province_name parameters must be supplied together.
state_province_name	string(200)	Optional
city_name	string(200)	Optional
latitude	decimal (in signed decimal degrees. N is + and S is -)	Optional
longitude	decimal (in signed decimal degrees. E is + and W is -)	Optional
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.
isactive	0 or 1	Optional. A value of 0 indicates that the record is inactive. No value, or a value of 1, indicates that the record is active.

location code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.LOCATION_CD, M.LOCATION_NAME, M.REGION_CD, M.REGION_NAME, M.COUNTRY_CD,
 M.COUNTRY_NAME, M.STATE_PROVINCE_CD, M.STATE_PROVINCE_NAME, M.CITY_NAME,
M.LATITUDE, M.LONGITUDE, L.LANGUAGE_CD, T.TENANT_CD, M.ISACTIVE FROM
SYSREC.MASTER_LOCATION M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

material

Defines the material that is used for an event.

The fields in the **material** table are defined as a specific instance of a material type, including a link to the supplier. It can be material that is used in a repair or material that is used in a production process.

The fields in the **material** table are listed in the following table.

Table 35. Fields in the **material** *table*

Field	Type	Comments
material_cd	string(50)	Required
material_name	string(200)	Required
material_type_cd	string(50)	Required
supplier_cd	string(50)	Required
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.
isactive	0 or 1	Optional. A value of 0 indicates that the record is inactive. No value, or a value of 1, indicates that the record is active.

material code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.MATERIAL_CD, M.MATERIAL_NAME, MT.MATERIAL_TYPE_CD, S.SUPPLIER_CD,
L.LANGUAGE_CD, T.TENANT_CD, M.ISACTIVE FROM SYSREC.MASTER_MATERIAL M
JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID JOIN
SYSREC.MASTER_MATERIAL_TYPE MT ON M.MATERIAL_TYPE_ID = MT.MATERIAL_TYPE_ID AND
M.LANGUAGE_ID = MT.LANGUAGE_ID JOIN SYSREC.MASTER_SUPPLIER S ON M.SUPPLIER_ID =
S.SUPPLIER_ID  AND M.LANGUAGE_ID = S.LANGUAGE_ID;
```

material_type

A categorization of material by type.

Material type is material that is used in a repair, such as engine filters or parts, or it can be material that is used in a production process.

The fields in the **material type** table are listed in the following table.

Table 36. Fields in the **material type** *table*

Field	Type	Comments
material_type_cd	string(50)	Required
material_type_name	string(200)	Required
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

material_type code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.MATERIAL_TYPE_CD, M.MATERIAL_TYPE_NAME, L.LANGUAGE_CD, T.TENANT_CD FROM
SYSREC.MASTER_MATERIAL_TYPE M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

process

Represents a production process.

A process can be part of a hierarchy of processes.

The fields in the **process** table are listed in the following table.

Table 37. Fields in the **process** *table*

Field	Type	Comments
process_cd	string(50)	Required
process_name	string(200)	Required
parent_process_cd	string(50)	Optional
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

process code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.PROCESS_CD, M.PROCESS_NAME, P.PROCESS_CD AS PARENT_PROCESS_CD,
L.LANGUAGE_CD, T.TENANT_CD FROM SYSREC.MASTER_PROCESS M JOIN
SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID JOIN SYSREC.MASTER_PROCESS
P ON M.PARENT_PROCESS_ID = P.PARENT_PROCESS_ID  AND M.LANGUAGE_ID = P.LANGUAGE_ID;
```

product

Defines the product being produced by the events.

The fields in the **product** table are listed in the following table.

Table 38. Fields in the **product** *table*

Field	Type	Comments
product_cd	string(50)	Required
product_name	string(200)	Required
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.
isactive	0 or 1	Optional. A value of 0 indicates that the record is inactive. No value, or a value of 1, indicates that the record is active.

product code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.PRODUCT_CD, M.PRODUCT_NAME, L.LANGUAGE_CD, T.TENANT_CD, M.ISACTIVE FROM
SYSREC.MASTER_PRODUCT M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

production_batch

Contains information about product groupings during the production event.

A batch can split and merge throughout the production process, and so one batch can be related to many other batches.

The fields in the **production_batch** table are listed in the following table.

Table 39. Fields in the **production_batch** *table*

Field	Type	Comments
production_batch_cd	string(50)	Required
production_batch_name	string(200)	Required
product_cd	string(50)	Required
language_cd	string(50)	Optional. This value must reference a row in the **language** table.

Table 39. Fields in the **production_batch** *table (continued)*

Field	Type	Comments
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

production_batch code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.PRODUCTION_BATCH_CD, M.PRODUCTION_BATCH_NAME, P.PRODUCT_CD,
L.LANGUAGE_CD, T.TENANT_CD FROM SYSREC.MASTER_PRODUCTION_BATCH M JOIN
SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID JOIN SYSREC.MASTER_PRODUCT
P ON M.PRODUCT_ID = P.PRODUCT_ID AND M.LANGUAGE_ID = P.LANGUAGE_ID;
```

profile_calculation

These records define a set of profile calculation names.

Profile calculations aggregate event values into KPIs and Profiles.

The fields in the **profile_calculation** table are listed in the following table.

Table 40. Fields in the **profile_calculation** *table*

Field	Type	Comments
profile_calculation_name	string(200)	Required
language_cd	string(50)	Optional
tenant_cd	string(50)	Optional

profile_calculation code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.PROFILE_CALCULATION_NAME, T.TENANT_CD FROM
SYSREC.MASTER_PROFILE_CALCULATION M JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID
= T.TENANT_ID;
```

resource

Defines resources of type asset or agent. Asset or agent are the only resource types allowed.

An asset is a piece of equipment. An agent is the operator of the equipment. Some asset resources can form a hierarchy. For example, a truck is a parent of a tire.

Parent resources must be loaded before child resources. Resources cannot be their own parent.

More specific types of resources can be named in the resource_sub_type column.

The fields in the **resource** table are listed in the following table.

Table 41. Fields in the **resource** *table*

Field	Type	Comments
serial_no	string(50)	Optional, but either serial_no and model are required, or operator_cd is required.
model	string(50)	Optional
operator_cd	string(50)	Optional
resource_name	string(500)	Required
resource_type_cd	string(50)	Required
resource_sub_type	string(50)	Optional
parent_resource_serial_no	string(50)	Optional. The parent_resource_serial_no and parent_resource_model parameters must be supplied together.
parent_resource_model	string(50)	Optional
parent_resource_operator_cd	string(50)	Optional
standard_production_rate	decimal	Optional
production_rate_uom	string(40)	Optional
preventative_maintenance_interval	decimal	Optional
group_dim_type_cd_1	string(50)	Optional. The type and a member must be supplied together.
group_dim_member_cd_1	string(50)	Optional
group_dim_type_cd_2	string(50)	Optional
group_dim_member_cd_2	string(50)	Optional
group_dim_type_cd_3	string(50)	Optional
group_dim_member_cd_3	string(50)	Optional
group_dim_type_cd_4	string(50)	Optional
group_dim_member_cd_4	string(50)	Optional
group_dim_type_cd_5	string(50)	Optional
group_dim_member_cd_5	string(50)	Optional
location_cd	string(50)	Optional
language_cd	string(50)	Optional. This value must reference a row in the **language** table
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table

Table 41. Fields in the **resource** table (continued)

Field	Type	Comments
isactive	0 or 1	Optional. A value of 0 indicates that the record is inactive. No value, or a value of 1, indicates that the record is active.

resource code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.SERIAL_NO, M.MODEL, M.OPERATOR_CD, M.RESOURCE_NAME, RT.RESOURCE_TYPE_CD,
 M.RESOURCE_SUB_TYPE, P.SERIAL_NO AS PARENT_RESOURCE_SERIAL_NO,
P.MODEL AS PARENT_RESOURCE_MODEL, P.OPERATOR_CD AS PARENT_RESOURCE_OPERATOR_CD,
M.STANDARD_PRODUCTION_RATE, M.PRODUCTION_RATE_UOM,
M.PREVENTIVE_MAINTENANCE_INTERVAL, G1.GROUP_TYPE_CODE AS GROUP_TYPE_CD_1,
G1.GROUP_MEMBER_CODE AS GROUP_MEMBER_CD_1, G2.GROUP_TYPE_CODE AS GROUP_TYPE_CD_2,
 G2.GROUP_MEMBER_CODE AS GROUP_MEMBER_CD_2, G3.GROUP_TYPE_CODE AS GROUP_TYPE_CD_3,
G3.GROUP_MEMBER_CODE AS GROUP_MEMBER_CD_3, G4.GROUP_TYPE_CODE AS GROUP_TYPE_CD_4,
G4.GROUP_MEMBER_CODE AS GROUP_MEMBER_CD_4, G5.GROUP_TYPE_CODE AS GROUP_TYPE_CD_5,
G5.GROUP_MEMBER_CODE AS GROUP_MEMBER_CD_5, LC.LOCATION_CD, L.LANGUAGE_CD,
T.TENANT_CD, M.ISACTIVE FROM SYSREC.MASTER_RESOURCE M JOIN SYSREC.MASTER_LANGUAGE
L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID =
T.TENANT_ID LEFT OUTER JOIN SYSREC.MASTER_RESOURCE P ON M.PARENT_RESOURCE_ID =
P.RESOURCE_ID  AND M.LANGUAGE_ID = P.LANGUAGE_ID JOIN SYSREC.MASTER_GROUP_DIM G1 ON
M.GROUP_DIM_ID_1 = G1.GROUP_DIM_ID   AND M.LANGUAGE_ID = G1.LANGUAGE_ID JOIN
SYSREC.MASTER_GROUP_DIM G2 ON M.GROUP_DIM_ID_2 = G2.GROUP_DIM_ID  AND M.LANGUAGE_ID
 = G2.LANGUAGE_ID JOIN SYSREC.MASTER_GROUP_DIM G3 ON M.GROUP_DIM_ID_3 =
G3.GROUP_DIM_ID  AND M.LANGUAGE_ID = G3.LANGUAGE_ID JOIN SYSREC.MASTER_GROUP_DIM G4
ON M.GROUP_DIM_ID_4 = G4.GROUP_DIM_ID   AND M.LANGUAGE_ID = G4.LANGUAGE_ID JOIN
SYSREC.MASTER_GROUP_DIM G5 ON M.GROUP_DIM_ID_5 = G5.GROUP_DIM_ID  AND M.LANGUAGE_ID
 = G5.LANGUAGE_ID JOIN SYSREC.MASTER_LOCATION LC ON M.LOCATION_ID = LC.LOCATION_ID
AND M.LANGUAGE_ID = LC.LANGUAGE_ID JOIN SYSREC.MASTER_RESOURCE_TYPE RT ON
M.RESOURCE_TYPE_ID = RT.RESOURCE_TYPE_ID   AND M.LANGUAGE_ID = RT.LANGUAGE_ID;
```

resource_type

These records categorize resources.

The two supported resource types are asset and agent. An asset is a piece of equipment that is used in the production process. An agent is the operator of the equipment.

The fields in the **resource_type** table are listed in the following table.

Table 42. Fields in the **resource_type** table

Field	Type	Comments
resource_type_cd	string(50)	Required
resource_type_name	string(200)	Required
language_cd	string(50)	Optional. This value must reference a row in the **language** table.

Table 42. Fields in the **resource_type** *table (continued)*

Field	Type	Comments
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

resource_type code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.RESOURCE_TYPE_CD, M.RESOURCE_TYPE_NAME, L.LANGUAGE_CD, T.TENANT_CD FROM
SYSREC.MASTER_RESOURCE_TYPE M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

source_system

Contains information about the system generating an event.

The fields in the **source_system** table are listed in the following table.

Table 43. Fields in the **source_system** *table*

Field	Type	Comments
source_system_cd	string(50)	Required.
source_system_name	string(200)	Required.
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.
isactive	0 or 1	Optional. A value of 0 indicates that the record is inactive. No value, or a value of 1, indicates that the record is active.

source_system code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.SOURCE_SYSTEM_CD, M.SOURCE_SYSTEM_NAME, L.LANGUAGE_CD, T.TENANT_CD,
M.ISACTIVE FROM SYSREC.MASTER_SOURCE_SYSTEM M JOIN SYSREC.MASTER_LANGUAGE L ON
M.LANGUAGE_ID = L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID =
T.TENANT_ID;
```

supplier

Contains material supplier information.

The fields in the **supplier** table are listed in the following table.

Table 44. Fields in the **supplier** *table*

Field	Type	Comments
supplier_cd	string(50)	Required.
supplier_name	string(200)	Required.
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.
isactive	0 or 1	Optional. A value of 0 indicates that the record is inactive. No value, or a value of 1, indicates that the record is active.

supplier code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.SUPPLIER_CD, M.SUPPLIER_NAME, L.LANGUAGE_CD, T.TENANT_CD, M.ISACTIVE
FROM SYSREC.MASTER_SUPPLIER M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

tenant

Contain the list of supported tenants.

The fields in the **tenant** table are listed in the following table.

Table 45. Fields in the **tenant** *table*

Field	Type	Comments
tenant_cd	string(50)	Required.
tenant_name	string(200)	Required.
isdefault	0 or 1	Optional. A value of 0 indicates that the record is inactive. No value, or a value of 1, indicates that the record is active.

tenant code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line.

```
SELECT TENANT_CD, TENANT_NAME, DEFAULT_IND FROM SYSREC.MASTER_TENANT;
```

For information on adding new languages and tenants, see the following information: “New languages and tenants” on page 64.

Changing the tenant code and name

You can rename the tenant code and name. For example, in the sample data, the tenant code and name by default is PMQ.

Procedure

1. Type the following command to connect to the **IBMPMQ** database by connecting to the DB2 node:

   ```
   db2 "connect to IBMPMQ user user_name using password"
   ```

2. Type the following command:

   ```
   db2 "update sysrec.master_tenant set tenant_code='CODE',
   tenant_name='NAME' where tenant_code='PMQ'"
   ```

   Where *CODE* is the tenant code, and *NAME* is the tenant name.

   For example, the following code renames the tenant code to XY, and the tenant name to XY Ltd.

   ```
   db2 "update sysrec.master_tenant set tenant_code='XY',
   tenant_name=XY Ltd' where tenant_code='PMQ'"
   ```

3. Type the following command to commit the transaction:

   ```
     db2 "commit"
   ```

4. Type the following command to disconnect from the database:

   ```
   db2 "connect reset"
   ```

value_type

Defines the set of possible numerical observations, including actual, planned or forecast.

The fields for the **value_type** table are listed in the following table.

Table 46. Fields for the **value_type** *table*

Field	Type	Comments
value_type_cd	string(50)	Required
value_type_name	string(200)	Required
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

value_type code snippet

You can use the following SQL Code snippet to retrieve master data in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load master data, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.VALUE_TYPE_CD, M.VALUE_TYPE_NAME, L.LANGUAGE_CD, T.TENANT_CD FROM
SYSREC.MASTER_VALUE_TYPE M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

Metadata in the API

The following are a list of the records supported by the metadata section of the application programming interface (API). They are listed in alphabetical order.

event_type

These records define a categorization of events.

Some examples of event types are measurement, alarm, and inspection.

The fields in the **event_type** table are listed in the following table.

Table 47. Fields in the **event_type** *table*

Field	Type	Comments
event_type_cd	string(50)	Required.
event_type_name	string(200)	Required.
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

event_type code snippet

You can use the following SQL Code snippet to retrieve metadata in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load metadata, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
 SELECT M.EVENT_TYPE_CD, M.EVENT_TYPE_NAME, L.LANGUAGE_CD, T.TENANT_CD FROM
SYSREC.MASTER_EVENT_TYPE M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID
```

measurement_type

Contains all the measures and event code sets that can be observed for the **resource**, **process**, and **material** records.

Some examples of measurement types are engine oil pressure, ambient temperature, fuel consumption, conveyor belt speed, capping pressure, and so on.

In the case of measurement types where the **event_code_indicator** value is 1, there is a special class to capture failure codes, issue codes, and alarm codes as **event_code** records. The **measurement_type_code** and **measurement_type_name** records become the **event_code_set** and **event_code_set_name** records respectively. This is a trigger to the event integration process to begin recording event codes from the **observation_text** record.

The fields for the **measurement_type** table are listed in the following table.

Table 48. Fields for the **measurement_type**

Field	Type	Comments
measurement_type_cd	string(50)	Required
measurement_type_name	string(200)	Required
unit_of_measure	string(100)	Optional
carry_forward_indicator	0 or 1	Optional
aggregation_type	string(100)	Optional
event_code_indicator	0 or 1	Optional
language_cd	string(50)	Optional. This value must reference a row in the **language** table.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

measurement_type code snippet

You can use the following SQL Code snippet to retrieve metadata in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load metadata, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.MEASUREMENT_TYPE_CD, M.MEASUREMENT_TYPE_NAME, M.UNIT_OF_MEASURE,
M.CARRY_FORWARD_INDICATOR, M.AGGREGATION_TYPE, M.EVENT_CODE_INDICATOR,
L.LANGUAGE_CD, T.TENANT_CD FROM SYSREC.MASTER_MEASUREMENT_TYPE M JOIN
SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID;
```

profile_variable

These records relate the measurement_type, resource_type, and material_type values to profile calculations.

The fields in the **profile_variable** table are listed in the following table.

Table 49. Fields in the **profile_variable** *table*

Field	Type	Comments
profile_variable_cd	string(50)	Required
profile_variable_name	string(200)	Required
profile_calculation_name	string(200)	Required
measurement_type_cd	string(50)	Required
resource_type_cd	string(50)	Optional
material_type_cd	string(50)	Optional
profile_units	string(100)	Optional
comparison_string	string(200)	Optional
low_value_date	datetime	Optional
high_value_date	datetime	Optional

Table 49. Fields in the **profile_variable** *table (continued)*

Field	Type	Comments
low_value_number	decimal	Optional
high_value_number	decimal	Optional
kpi_indicator	0 or 1	Optional. To disable a profile variable, set its kpi_indicator and profile_indicator to 0
profile_indicator	0 or 1	Optional. To disable a profile variable, set its kpi_indicator and profile_indicator to 0
data_type	string(100)	Optional
aggregation_type	string(100)	Optional
carry_forward_indicator	0 or 1	Optional
process_indicator	0 or 1	Optional
variance_multiplier	-1 or 1	Required. A value of 1 indicates that a higher measurement value is preferred. A value of -1 that a lower value is preferred.
tenant_cd	string(50)	Optional. This value must reference a row in the **tenant** table.

Due to references from the KPI and Profile tables, the upsert API for a profile_variable only allows the values of the following fields to be updated

- profile_units
- comparison_string
- low_value_date
- high_value_date
- low_value_number
- kpi_indicator
- profile_indicator
- data_type
- aggregation_type
- process_indicator
- profile_variable_name

profile_variable code snippet

You can use the following SQL Code snippet to retrieve metadata in the format that is required by the **upsert** API.

For example, if you lost the original files that are used to load metadata, you can use the snippet to retrieve the data, make changes, and submit the changes by using the **upsert** API.

The command must be on a single line, not as shown here.

```
SELECT M.PROFILE_VARIABLE_CD, M.PROFILE_VARIABLE_NAME, PC.PROFILE_CALCULATION_NAME,
MSRT.MEASUREMENT_TYPE_CD, RT.RESOURCE_TYPE_CD, MT.MATERIAL_TYPE_CD, M.PROFILE_UNITS,
M.COMPARISON_STRING, M.LOW_VALUE_DATE, M.HIGH_VALUE_DATE, M.LOW_VALUE_NUMBER,
M.HIGH_VALUE_NUMBER, M.KPI_INDICATOR, M.PROFILE_INDICATOR, M.DATA_TYPE,
M.AGGREGATION_TYPE, M.CARRY_FORWARD_INDICATOR, M.PROCESS_INDICATOR,
M.VARIANCE_MULTIPLIER, L.LANGUAGE_CD, T.TENANT_CD FROM
SYSREC.MASTER_PROFILE_VARIABLE M JOIN SYSREC.MASTER_LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.MASTER_TENANT T ON M.TENANT_ID = T.TENANT_ID JOIN
```

```
SYSREC.MASTER_PROFILE_CALCULATION PC ON M.PROFILE_CALCULATION_ID =
PC.PROFILE_CALCULATION_ID  JOIN SYSREC.MASTER_MEASUREMENT_TYPE MSRT ON
M.MEASUREMENT_TYPE_ID = MSRT.MEASUREMENT_TYPE_ID  AND M.LANGUAGE_ID =
MSRT.LANGUAGE_ID JOIN SYSREC.MASTER_RESOURCE_TYPE RT ON M.RESOURCE_TYPE_ID =
RT.RESOURCE_TYPE_ID AND M.LANGUAGE_ID = RT.LANGUAGE_ID JOIN
SYSREC.MASTER_MATERIAL_TYPE MT ON M.MATERIAL_TYPE_ID = MT.MATERIAL_TYPE_ID  AND
M.LANGUAGE_ID = MT.LANGUAGE_ID;
```

Mandatory profile variables and measurement types

To be able to process some events, you must load mandatory profile variables and measurement types.

Mandatory profile variables

The following profile variables must be loaded:

HS Required for health score related calculations.

LSC Required for current life span related calculations.

RC Required for calculations that are related to the recommendations count.

You can see examples in the `profile_variable_upsert_sample_pmq.csv` file. This is installed on the Enterprise Service Bus (ESB) node computer in the `/var/PMQ/MQSIFileInput/PMQSampleData/Sample_PMQ/MasterData-Set2` folder.

Define profile variables that are based on the design of the IBM Cognos Business Intelligence reports and predictive models.

For example, for the sample models shipped with IBM Predictive Maintenance and Quality, profile variables and corresponding measurement types must be defined for the following `profile_variable_cd`:

- AC
- ATIME
- CELLLDX
- CELLLDXX
- CLTX
- CLTXX
- FAIL
- HS
- INSP
- ITIME
- LSC
- OPHD
- QTY
- RC
- REPC
- REPT
- SETX
- SETXX
- SLTX
- SLTXX

Mandatory measurement types

The following measurement types must be loaded:

HS Required for health score related calculations.

LSC Required for current life span related calculations.

You can see examples of these measurement types in the `measurement_type_upsert_sample_pmq.csv` file. This is installed on the Enterprise Service Bus (ESB) node computer in the `/var/PMQ/MQSIFileInput/PMQSampleData/Sample_PMQ/MasterData-Set1`.

Sample health score, Lifespan, and IBM Analytical Decision Management services are configured for these measurement types:

- `FAIL`
- `INSP`
- `LUBE`
- `OPHR`
- `PRS1`
- `PRS2`
- `PRS3`
- `RELH`
- `REPT`
- `REPX`
- `RPM`
- `R_B1`
- `R_F1`
- `TEMP`

For health score, define profile variables with the profile calculations for the listed measurement types:

- Measurement of Type
- Measurement Above Limit (except for `FAIL`)
- Measurement Below Limit (except for `FAIL`)

For lifespan, define profile variables with the profile calculation Lifespan Analysis failure for `FAIL`, and Lifespan Analysis (except for `FAIL`) for the listed measurement types.

Remove master data

Normally master data is not deleted from the analytic database. During testing and development, master data that is not referenced can be removed.

Sample code to remove master data

The following SQL code is an example and must be modified.

```
-- batch batch
DELETE FROM SYSREC.MASTER_BATCH_BATCH M WHERE
M.PRODUCTION_BATCH_ID = (SELECT PB1.PRODUCTION_BATCH_ID FROM
 SYSREC.MASTER_PRODUCTION_BATCH PB1
JOIN SYSREC.MASTER_LANGUAGE L ON PB1.LANGUAGE_ID = L.LANGUAGE_ID
JOIN SYSREC.MASTER_TENANT T ON PB1.TENANT_ID = T.TENANT_ID WHERE
PB1.PRODUCTION_BATCH_CD = '1007' AND L.LANGUAGE_CD = 'EN' AND T.TENANT_CD = 'PMQ')
AND
M.RELATED_PRODUCTION_BATCH_ID = (SELECT PB2.PRODUCTION_BATCH_ID FROM
 SYSREC.MASTER_PRODUCTION_BATCH PB2
JOIN SYSREC.MASTER_LANGUAGE L ON PB2.LANGUAGE_ID = L.LANGUAGE_ID
JOIN SYSREC.MASTER_TENANT T ON PB2.TENANT_ID = T.TENANT_ID WHERE
PB2.PRODUCTION_BATCH_CD = '1010' AND L.LANGUAGE_CD = 'EN' AND T.TENANT_CD = 'PMQ');

-- event code
DELETE FROM SYSREC.MASTER_EVENT_CODE M WHERE
M.EVENT_CODE_SET = 'FAIL' AND
M.EVENT_CODE = 'X101' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- event type
DELETE FROM SYSREC.MASTER_EVENT_TYPE M WHERE
M.EVENT_TYPE_CD = 'ALARM' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- group dim
DELETE FROM SYSREC.MASTER_GROUP_DIM M WHERE
M.GROUP_TYPE_CODE = 'ORG' AND
M.GROUP_MEMBER_CODE = 'C1' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- location
DELETE FROM SYSREC.MASTER_LOCATION M WHERE
M.LOCATION_CD = 'Room1' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- material
DELETE FROM SYSREC.MASTER_MATERIAL M WHERE
M.MATERIAL_CD = '20390' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- material type
DELETE FROM SYSREC.MASTER_MATERIAL_TYPE M WHERE
M.MATERIAL_TYPE_CD = 'PROD' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- measurement type
DELETE FROM SYSREC.MASTER_MEASUREMENT_TYPE M WHERE
M.MEASUREMENT_TYPE_CD = 'SET' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- process hierarchy
```

```
DELETE FROM SYSREC.PROCESS_HIERARCHY M WHERE
M.PROCESS_ID = (SELECT P.PROCESS_ID FROM SYSREC.MASTER_PROCESS P WHERE
 P.PROCESS_CD = 'SET') AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- process
DELETE FROM SYSREC.MASTER_PROCESS M WHERE
M.PROCESS_CD = 'SET' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- product
DELETE FROM SYSREC.MASTER_PRODUCT M WHERE
M.PRODUCT_CD = '2190890' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- production_batch
DELETE FROM SYSREC.MASTER_PRODUCTION_BATCH M WHERE
M.PRODUCTION_BATCH_CD = '1000' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- profile variable
DELETE FROM SYSREC.MASTER_PROFILE_VARIABLE M WHERE
M.PROFILE_VARIABLE_CD = 'SET' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- resource hierarchy
DELETE FROM SYSREC.RESOURCE_HIERARCHY M WHERE
M.RESOURCE_ID = (SELECT R.RESOURCE_ID FROM SYSREC.MASTER_RESOURCE R WHERE
 R.SERIAL_NO = '13580' AND R.MODEL = 'M100' ) AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- resource
DELETE FROM SYSREC.MASTER_RESOURCE M WHERE
M.SERIAL_NO = '13580' AND
M.MODEL = 'M100' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- source system
DELETE FROM SYSREC.MASTER_SOURCE_SYSTEM M WHERE
M.SOURCE_SYSTEM_CD = 'PREDMAIT' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');

-- supplier
```

```
DELETE FROM SYSREC.MASTER_SUPPLIER M WHERE
M.SUPPLIER_CD = 'WS' AND
M.LANGUAGE_ID = (SELECT L.LANGUAGE_ID FROM SYSREC.MASTER_LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.MASTER_TENANT T WHERE
 T.TENANT_CD = 'PMQ');
```

Note:

The contents of the SYSREC.MASTER_LANGUAGE, SYSREC.MASTER_PROFILE_CALCULATION, SYSREC.MASTER_TENANT, SYSREC.MASTER_VALUE_TYPE, and SYSREC.MASTER_RESOURCE_TYPE tables are normally not deleted when master data is removed.

Appendix B. Event schema

The event schema for IBM Predictive Maintenance and Quality is shown in the following code sample.

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetName-
space="http:\\www.ibm.com\predmain">

    <xsd:element name="event">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="incoming_event_cd"
        type="xsd:string" maxOccurs="1" minOccurs="0">
       </xsd:element>
       <xsd:element name="event_type_cd" type="xsd:string" maxOccurs="1"
minOccurs="1"></xsd:element>
       <xsd:element name="source_system_cd"
        type="xsd:string" maxOccurs="1" minOccurs="0">
       </xsd:element>
       <xsd:element name="process_cd" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
       <xsd:element name="production_batch_cd"
        type="xsd:string" maxOccurs="1" minOccurs="0">
       </xsd:element>
       <xsd:element name="location_cd" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
       <xsd:element name="event_start_time"
        type="xsd:dateTime" maxOccurs="1" minOccurs="1">
       </xsd:element>
       <xsd:element name="event_end_time"
        type="xsd:dateTime" maxOccurs="1" minOccurs="0">
       </xsd:element>
       <xsd:element name="event_planned_end_time"
        type="xsd:dateTime" maxOccurs="1" minOccurs="0">
       </xsd:element>
       <xsd:element name="tenant_cd" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
       <xsd:element name="operator_cd" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
       <xsd:element name="model" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
       <xsd:element name="serial_no" type="xsd:string" maxOccurs="1"
minOccurs="0"></xsd:element>
       <xsd:element name="observation" maxOccurs="unbounded" minOccurs="1">
        <xsd:complexType>
         <xsd:sequence>
          <xsd:element name="measurement_type_cd"
           type="xsd:string" maxOccurs="1" minOccurs="1">
          </xsd:element>
          <xsd:element name="observation_timestamp"
           type="xsd:dateTime" maxOccurs="1" minOccurs="0">
          </xsd:element>
          <xsd:element name="value_type_cd"
           type="xsd:string" maxOccurs="1" minOccurs="0">
          </xsd:element>
          <xsd:element name="observation_text"
           type="xsd:string" maxOccurs="1" minOccurs="1">
          </xsd:element>
          <xsd:element name="measurement"
           type="xsd:double" maxOccurs="1" minOccurs="0">
          </xsd:element>
          <xsd:element name="material_cd"
       type="xsd:string" maxOccurs="1" minOccurs="0">
      </xsd:element>
     </xsd:sequence>
    </xsd:complexType>
   </xsd:element>
  </xsd:sequence>
 </xsd:complexType>
</xsd:element></xsd:schema>
```

Appendix C. IBM Cognos Framework Manager model description

IBM Predictive Maintenance and Quality uses IBM Cognos Framework Manager to model the metadata for reports.

IBM Cognos Framework Manager is a metadata modeling tool that drives query generation for IBM Cognos software. A model is a collection of metadata that includes physical information and business information for one or more data sources. IBM Cognos software enables performance management on normalized and denormalized relational data sources and a variety of OLAP data sources.

For information on modifying or creating Framework Manager models, see the *IBM Cognos Framework Manager User Guide* and *IBM Cognos Framework Manager - Guidelines for Modeling Metadata*. This is available from the IBM Cognos Business Intelligence information center (http://pic.dhe.ibm.com/infocenter/cbi/v10r2m0/index.jsp).

The Framework Manager model consists of three layers:

- Database layer
- Logical layer
- Dimensional layer

Each of these layers is in a separate namespace. The dimensional layer is published to a package for use in reporting.

IBM Cognos Framework Manager model database layer

The physical, or database, layer contains a database query subject for every table in the physical data model. In addition, it includes role playing versions of some entities by using shortcut aliases.

The shortcut aliases are provided for two situations:

- To eliminate ambiguity for an entity that may be involved in multiple relationships, including the following items:
  - `location` and `location` (resource)
  - `material_type` and `material_type` (`profile_variable`)
  - `resource_type` and `resource_type` (`profile_variable`)
  - `production_batch` and `production_batch` (related)
- To enable you to query multiple copies of the same table in different roles, including the group_dim_1 to 5 values

If a database entity includes the `language_id` or `tenant_id` attributes, the database query subject includes a parameterized filter for each that selects only one tenant or language.

The database layer contains all of the entity relationships. The central entities are largely modeled in star or snowflake schemas, shown in the following diagrams. These parameters must be set after master data has been loaded or reloaded and before publishing the package. If these parameters are not set correctly, no data will be returned in the reports. To change the values, simply open the parameter map, double click on the value for each parameter and type over it.

Typically the central fact has cardinality 1,N and related objects are 1,1, in order to eliminate the need for relationships outside of the database layer. All joins are modeled as inner joins on the understanding that the data integration layer populates a default value for all references in the absence of a valid value.

The cardinality from the event_observation to event_resource tables is 1,N to 1,1 whereas the physical model is 1,N to 1,N. The difference in cardinality simplifies the Framework Manager model and is possible because reporting includes filters on the resource type.

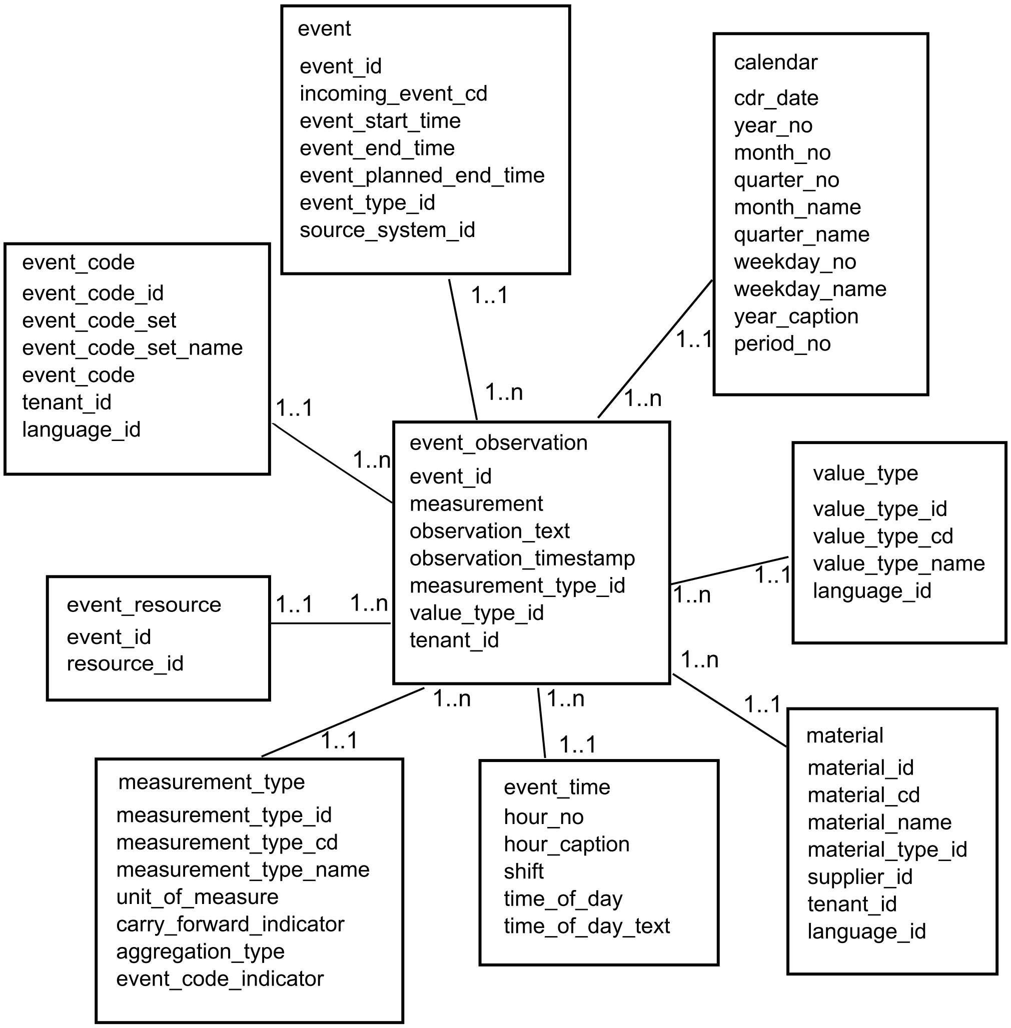


Figure 9. The event_observation star schema

The following diagram shows the star schema for the resource_profile table.

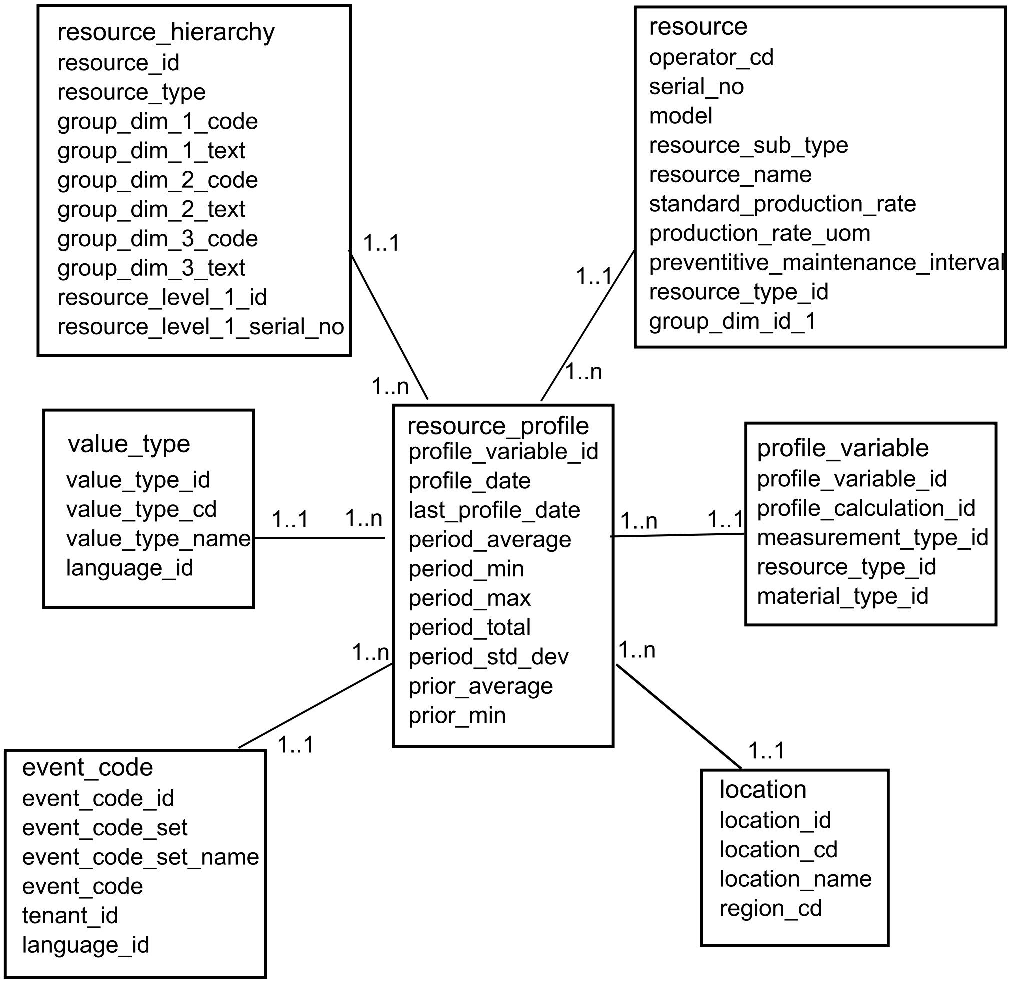


Figure 10. The resource_profile star schema

The following diagram shows the star schema for the `resource_kpi` table.

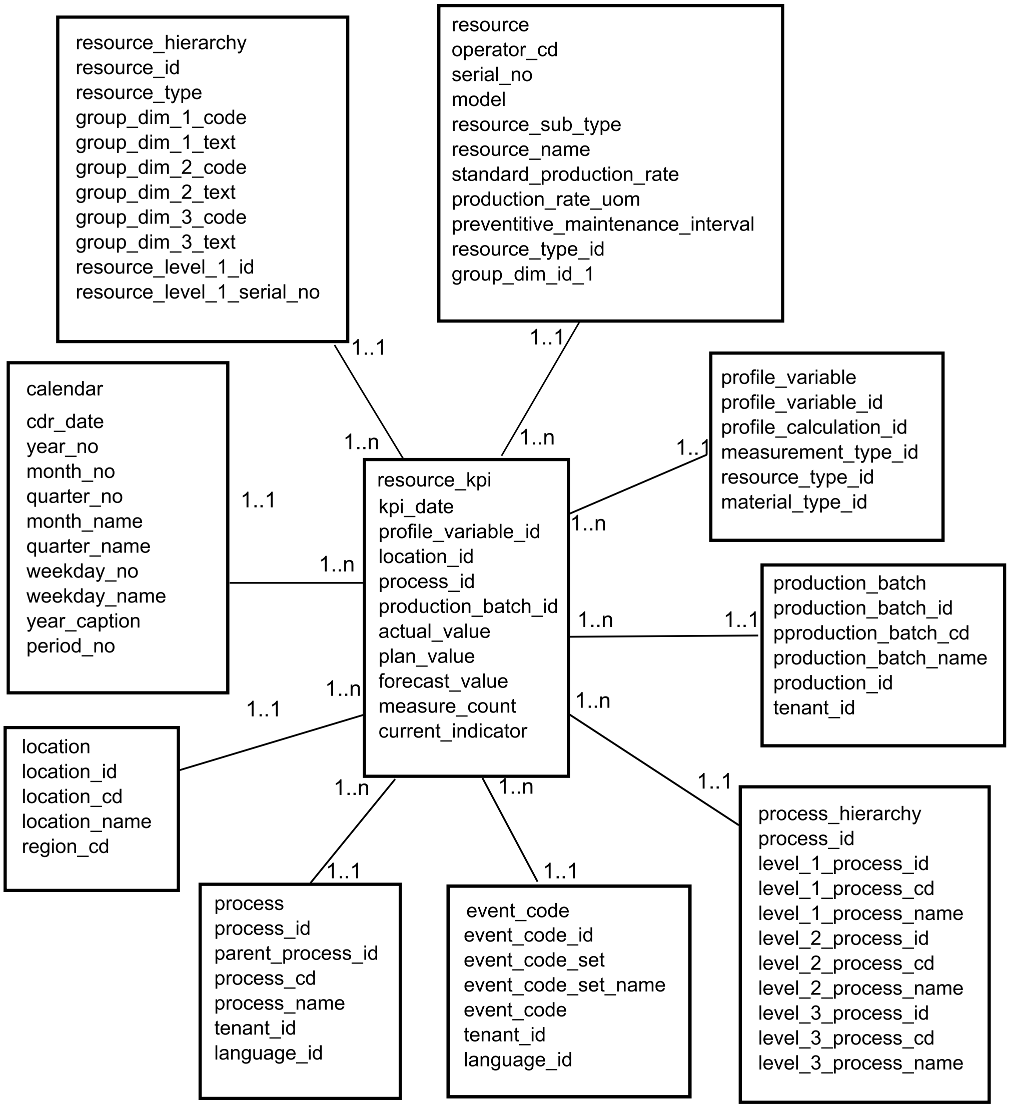


Figure 11. The resource_kpi star schema

The following diagram shows the star schema for the `material_profile` table.

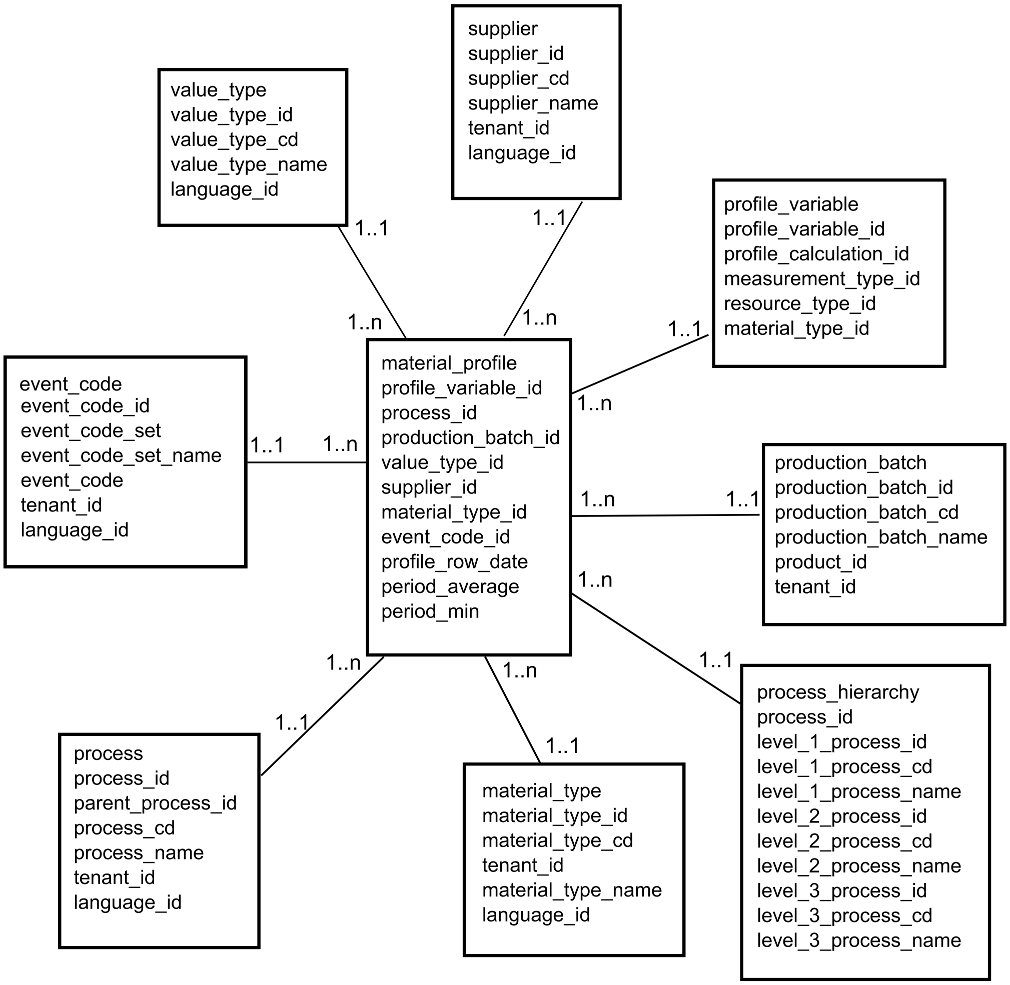


Figure 12. The material_profile star schema

The following diagram shows the star schema for the `process_profile` table.

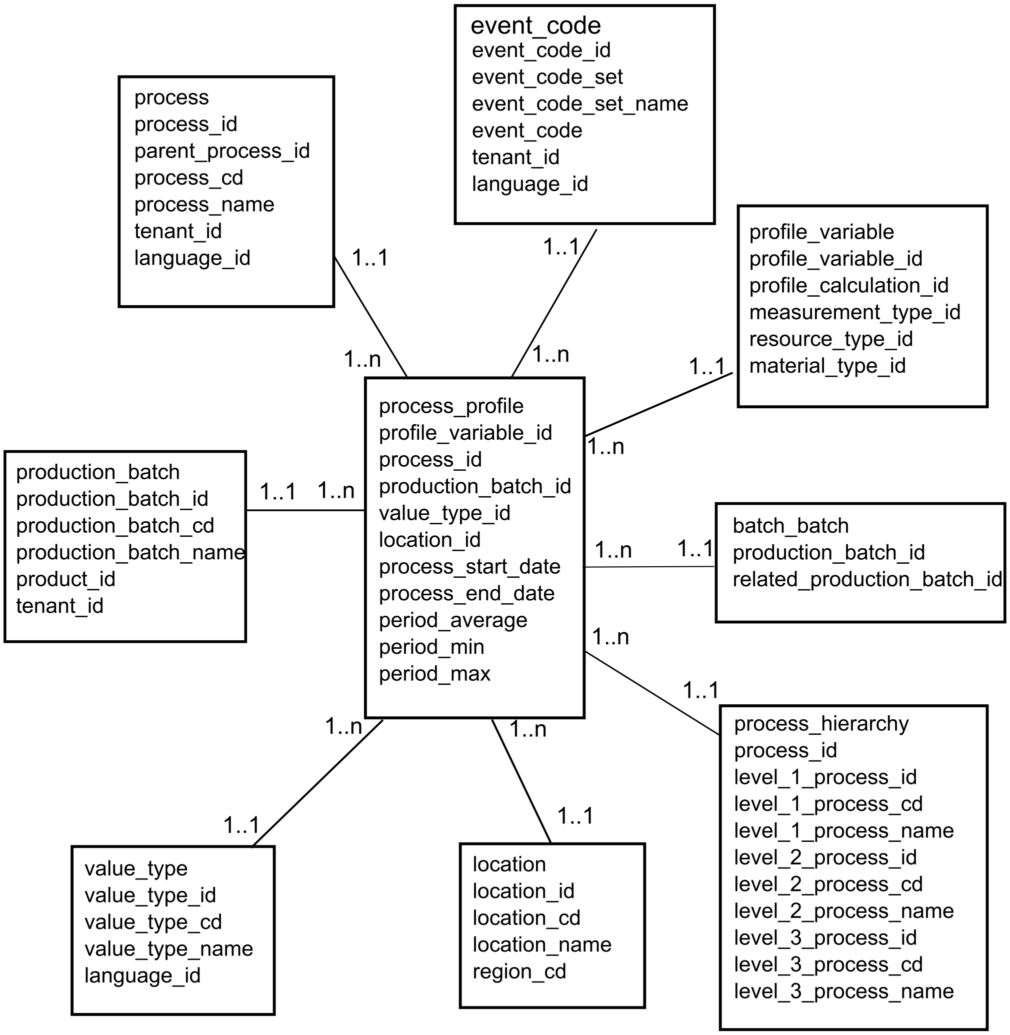


Figure 13. The process_profile star schema

The following diagram shows the star schema for the `process_kpi` table.

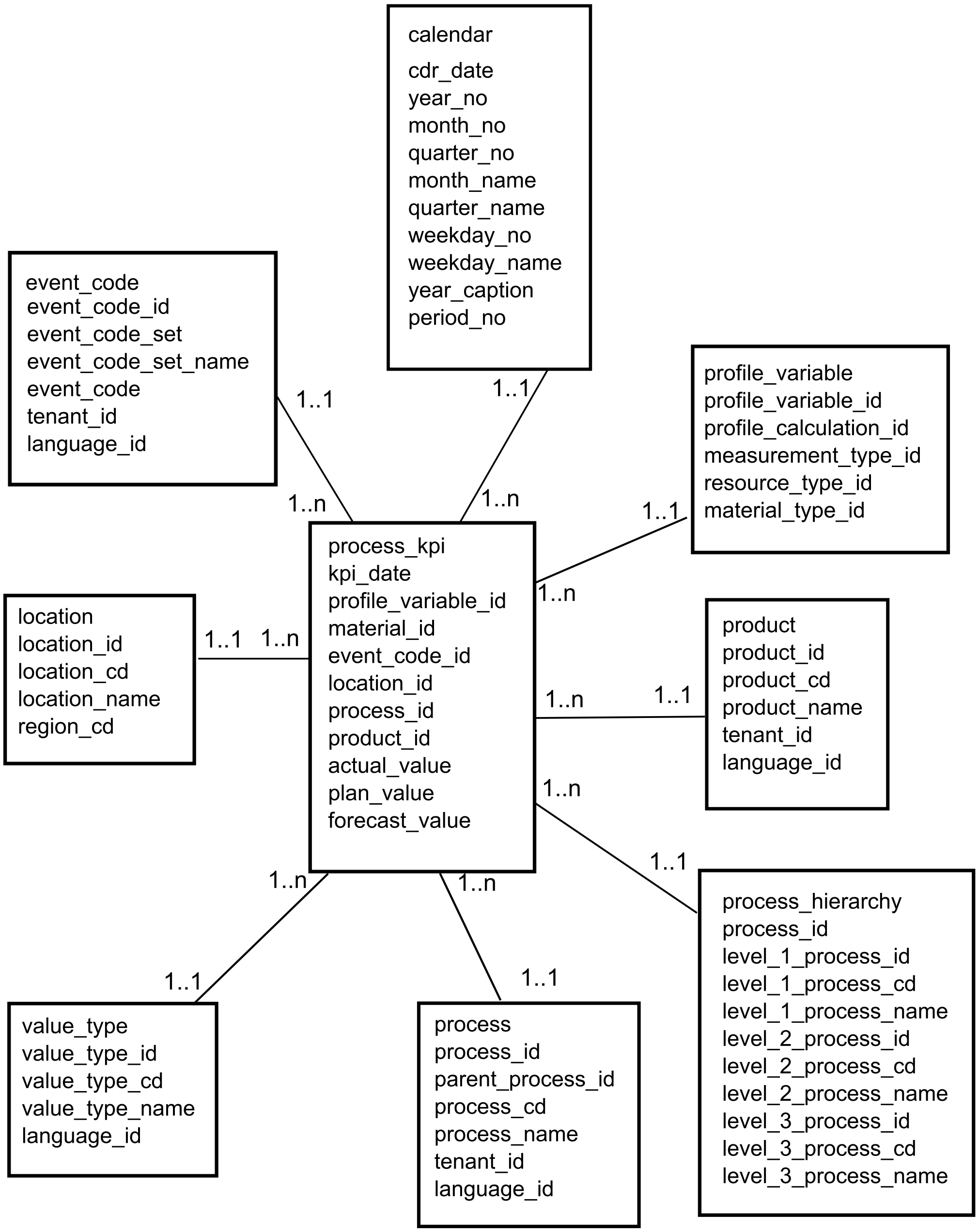


Figure 14. The process_kpi star schema

IBM Cognos Framework Manager model logical layer

The logical layer contains query subjects that draw data from the database query subjects and present it in a more consumable format.

Attributes are renamed to eliminate underscores and to use sentence case. In some cases, physical entities are combined into one query subject. In particular, the following snowflake dimensions are combined together to accommodate master data reporting without cross-products:

- Query subject Profile Variable contains attributes of profile_variable, measurement_type, profile_calculation, resource_type (profile_variable), and material_type (profile_variable).
- Query subject Material contains attributes of material, supplier and material_type.
- Query subject Production Batch contains attributes of production_batch and product
- Query subject Related Batch contains attributes of production_batch, batch_batch and production_batch (related).
- Query subject Resource contains attributes of resource, resource_type, location (resource), and group_dim_1 to 5.
- Query subject Event Observation contains attributes of event, event_observation, and event_resource.

The query subjects are organized into a folder for dimensions and a separate namespace for each logical fact. The fact query subjects contain additional calculated attributes that are included in dimensional layer measure dimensions.

IBM Cognos Framework Manager model dimensional layer

The dimensional layer contains the hierarchies and measure dimensions for publication to a package. Each dimension in the logical layer has a dimension in the dimensional layer with one or more hierarchies defined. The hierarchies usually include the caption field twice, once as a caption for the level, once as an attribute that can be used in report filters. All hierarchies are sorted.

Each measure dimension is in a separate namespace for the fact. Also included in the namespace are shortcuts to all the dimensions that have scope for that fact. Any dimension shortcut that is inside the namespace of the fact can also be consumed from outside of the namespace by IBM Cognos Business Intelligence reports.

Key Perfomance Indicator (KPI) tables include one measure with flexible aggregation. Based on the aggregation type in Profile Variable, the measure will either total the Actual Value or will calculate an average based on Sum of Actual Value / Sum of Measure Count. This requires that the data integration layer populates Measure Count with the actual number of observations for measures with aggregation type of Average and that it adds together measures that do not naturally seem to be additive, for example, temperature or pressure. The profile tables include a similar measure for flexible aggregation, with the addition of a check for Value Type = Actual.

IBM Cognos Framework Manager model security

No security is defined for the IBM Cognos Framework Manager model other than the provision for filtering by the tenant_id parameter on the physical layer. These query subject filters can be converted to security filters that are based on user IDs, allowing multi-tenant access to one database.

The Framework Manager model gives the ability to filter by the tenant_id parameter on the physical layer. As a preliminary measure to define security for the Framework Manager model, convert database query subject filters to security filters that are based on user IDs, allowing multi-tenant access to one database.

Query mode

The IBM Predictive Maintenance and Quality reports uses IBM Cognos Compatible Query Mode, which is the supported mode for all of the reports.

Using Compatible Query Mode to see real-time data

To see real-time data, you must ensure that caching is disabled in Dynamic Query mode and switch IBM Predictive Maintenance and Quality to use Compatible Query Mode.

Procedure

1. To turn off query reuse, open the `CQEConfig.xml` file that is in `{IBM Cognos Install Directory}/configuration` and edit the QueryEngine section by typing the following information.

   ```
   <section name="QueryEngine">
           <!-- Description: queryReuse feature -->
           <!-- value="0" means disable the feature -->
           <!-- default is value="5" which means cache up to 5result sets per session -->
           <entry name=queryReuse" value="0"/>
           ...
   </section>
   ```

2. Restart the IBM Cognos Business Intelligence server.
3. In IBM Cognos Administration, ensure that the data source defined for the IBM Predictive Maintenance and Quality database has native and JDBC connection definitions.
4. In IBM Framework Manager, select the project and change the **Query Mode** property to Compatible.
5. Publish the **IBMPMQ** package in Compatible mode by not selecting the check box to publish in Dynamic Query Mode when prompted.

Appendix D. IBM Predictive Maintenance and Quality Artifacts

Sample files, model files, and other content is supplied to help you to configure IBM Predictive Maintenance and Quality for the needs of your business.

Data model

The data model file name is IBMPMQ.sql. It is deployed to your IBM DB2® location. This script creates the database tables and populates the data in the IBMPMQ database.

IBM InfoSphere Master Data Management Collaboration Server file

The IBM InfoSphere MDM Collaboration Server data model file name is IBMPMQ.zip. This is a company archive file that contains the data model.

IBM WebSphere Message Broker artifacts

IBM WebSphere Message Broker archive (BAR) files, script files, and project interchange files are provided.

IBM WebSphere Message Broker archive files

The following WebSphere Message Broker archive (BAR) files are configured by using command-line parameters:

- PMQEventDataLoad.bar
- PMQMaximoIntegration.bar
- PMQIntegration.bar
- PMQMasterDataLoad.bar

IBM WebSphere Message Broker script files

The following script files are used to deploy the BAR files by using command-line parameters:

- credentials.properties
- PMQIntegration.properties
- PMQMaximoIntegration.properties
- Queues.txt
- sql.properties

IBM WebSphere Message Broker project interchange files

The following files are project interchange files:

- PMQEventDataLoad_PI.zip
- PMQMaximoIntegration_PI.zip
- PMQIntegration_PI.zip
- PMQMasterDataLoad_PI.zip

Sample master and event files

Sample master and event files are provided.

Master files:

- batch_batch_upsert_Sample_PMQ.csv
- event_code_upsert_Sample_PMQ.csv
- event_type_upsert_Sample_PMQ.csv
- group_dim_upsert_Sample_PMQ.csv
- language_upsert_Sample_PMQ.csv
- location_upsert_Sample_PMQ.csv
- material_type_upsert_Sample_PMQ.csv
- material_upsert_Sample_PMQ.csv
- measurement_type_upsert_Sample_PMQ.csv
- process_upsert_Sample_PMQ.csv
- product_upsert_Sample_PMQ.csv
- production_batch_upsert_Sample_PMQ.csv
- profile_calculation_upsert_Sample_PMQ.csv
- profile_variable_upsert_Sample_PMQ.csv
- resource_type_upsert_Sample_PMQ.csv
- resource_upsert_Sample_PMQ.csv
- source_system_upsert_Sample_PMQ.csv
- supplier_upsert_Sample_PMQ.csv
- tenant_upsert_Sample_PMQ.csv
- value_type_upsert_Sample_PMQ.csv

Event file:

- event_observation_Sample_PMQ.csv

IBM SPSS Artifacts

Predictive model streams, sample source files, and IBM Analytical Decision Management templates are provided.

Predictive model streams

The predictive model streams are supplied in the PredictiveMaintenance.pes file. This file is deployed to IBM SPSS Collaboration and Deployment Services.

Health_Score_Model.str
The health score model provides a continuous real-time monitoring of machine or asset health, and prediction of potential machine faults.

Life_Time_Analysis_Model2.str
Using lifetime analysis, you can calculate the expected future lifetime of the parts.

Life_Time_Analysis_Model1.str
Using lifetime analysis, you can calculate the expected future lifetime of the parts.

Combine_Resource_Recommendation.str
A sample stream to identify the probability of failure of the asset. This stream is invoked in the Combine template in the IBM Analytical Decision Management Stream directly.

Sample source files

Sample source files are provided that are used in the IBM SPSS Modeling and Decision Management streams. These files are installed in the IBM SPSS Modeler Server Installation location: /usr/IBM/SPSS/ModelerServer/15.0/data. These files are not required during the real-time scoring and deployment of IBM Predictive Maintenance and Quality.

- DM_Input.csv
- model_data.csv
- scoring_data.csv
- source_data.csv

IBM Analytical Decision Management applications and templates

Two IBM Analytical Decision Management applications and templates are packaged with the product. You must choose the relevant application for the needs of your business.

The PredictiveMaintenance and PredictiveMaintenanceQuality applications are installed to /opt/IBM/SPSS/Deployment/5.0/Server/Decision_Management/Applications/.

The PredictiveMaintenance.xml and PredictiveMaintenanceQuality.xml templates are installed to this location: /opt/IBM/SPSS/Deployment/5.0/Server/Decision-Management/Templates/.

IBM Cognos Business Intelligence Artifacts

An IBM Framework Manager model, and a compressed file that contains reports is provided.

The IBM Framework Manager Model is named IBMPMQ Model.

The IBM Cognos BI reports are provided in a compressed file named IBMPMQ.zip. They are deployed to /opt/ibm/cognos/c10_64/deployment.

The following reports are included:

- Site overview dashboard
- Equipment reports dashboard
- Product quality dashboard
- Audit report
- Resource list
- Profile variables list - drill through
- Process list - drill through
- Material list - drill through
- Production batch list - drill through
- Material usage by production batch

- Incident event list - drill through
- Incident/recommendation event list - drill through
- Recommendation event list - drill through
- Reusable object library

Installation files provided with IBM Predictive Maintenance and Quality

Installation files are provided so that you can install and configure IBM Predictive Maintenance and Quality.

The following installation files are provided.

Prerequisite script
The prerequisite script checks for required tools, space, and root user on the target systems.

Server installer
IBM Predictive Maintenance and Quality server installer that installs and configures all of the components.

Client deployment accelerator
IBM Predictive Maintenance and Quality client installer installs the components' client installers.

Content deployment accelerator
IBM Predictive Maintenance and Quality content deployment accelerator deploys IBM Predictive Maintenance and Quality content on the target system.

Launchpad
The launch pad provides a platform from where the server installer, content deployment accelerator, and the client installer are triggered.

Appendix E. Troubleshooting a problem

Troubleshooting is a systematic approach to solving a problem. The goal of troubleshooting is to determine why something does not work as expected and how to resolve the problem.

Review the following table to help you or customer support resolve a problem.

Table 50. Actions and descriptions

Actions	Description
A product fix might be available to resolve your problem.	Apply all known fix packs, or service levels, or program temporary fixes (PTF).
Look up error messages by selecting the product from the IBM Support Portal, and then typing the error message code into the **Search support** box (http://www.ibm.com/support/entry/portal/).	Error messages give important information to help you identify the component that is causing the problem.
Reproduce the problem to ensure that it is not just a simple error.	If samples are available with the product, you might try to reproduce the problem by using the sample data.
Ensure that the installation successfully finished.	The installation location must contain the appropriate file structure and the file permissions. For example, if the product requires write access to log files, ensure that the directory has the correct permission.
Review all relevant documentation, including release notes, technotes, and proven practices documentation.	Search the IBM knowledge bases to determine whether your problem is known, has a workaround, or if it is already resolved and documented.
Review recent changes in your computing environment.	Sometimes installing new software might cause compatibility issues.

If the items on the checklist did not guide you to a resolution, you might need to collect diagnostic data. This data is necessary for an IBM technical-support representative to effectively troubleshoot and assist you in resolving the problem. You can also collect diagnostic data and analyze it yourself.

Troubleshooting resources

Troubleshooting resources are sources of information that can help you resolve a problem that you are having with an IBM product.

Support Portal

The IBM Support Portal is a unified, centralized view of all technical support tools and information for all IBM systems, software, and services.

The IBM Support Portal lets you access all the IBM support resources from one place. You can tailor the pages to focus on the information and resources that you need for problem prevention and faster problem resolution. Familiarize yourself with the IBM Support Portal by viewing the demo videos (*https://www.ibm.com/blogs/SPNA/entry/the_ibm_support_portal_videos*).

Find the content that you need by selecting your products from the IBM Support Portal (*http://www.ibm.com/support/entry/portal/*).

Gathering information

Before contacting IBM Support, you will need to collect diagnostic data (system information, symptoms, log files, traces, and so on) that is required to resolve a problem. Gathering this information will help to familiarize you with the troubleshooting process and save you time

Service requests

Service requests are also known as Problem Management Reports (PMRs). Several methods exist to submit diagnostic information to IBM Software Technical Support.

To open a PMR or to exchange information with technical support, view the IBM Software Support Exchanging information with Technical Support page (http://www.ibm.com/software/support/exchangeinfo.html).

Fix Central

Fix Central provides fixes and updates for your system's software, hardware, and operating system.

Use the pull-down menu to navigate to your product fixes on Fix Central *(http://www-947.ibm.com/systems/support/fixes/en/fixcentral/help/getstarted.html)*. You may also want to view Fix Central help.

Knowledge bases

You can find solutions to problems by searching IBM knowledge bases.

You can use the IBM masthead search by typing your search string into the Search field at the top of any ibm.com page.

IBM Redbooks

IBM Redbooks® are developed and published by IBM's International Technical Support Organization, the ITSO.

IBM Redbooks (*http://www.redbooks.ibm.com/*) provide in-depth guidance about such topics as installation and configuration and solution implementation.

IBM developerWorks

IBM developerWorks provides verified technical information in specific technology environments.

As a troubleshooting resource, developerWorks provides easy access to the top ten most popular practices for Business analytics, in addition to videos and other information: developerWorks for Business analytics (*http://www.ibm.com/developerworks/analytics/practices.html*).

Software support and RSS feeds

IBM Software Support RSS feeds are a quick, easy, and lightweight format for monitoring new content added to websites.

After you download an RSS reader or browser plug-in, you can subscribe to IBM product feeds at IBM Software Support RSS feeds (*https://www.ibm.com/software/support/rss/*).

Log files

Log files can help you troubleshoot problems by recording the activities that take place when you work with a product.

IBM WebSphere Message Broker log files

Errors that occur within IBM WebSphere Message Broker message flows are written to error logs in the following folder: `/error`. The location of this folder is determined by the **`MQSI_FILENODES_ROOT_DIRECTORY`** environment variable during the installation process.

Errors for message flows are as follows:

Master data flows
Rejected records are written to *`input_filename`*`_error.csv`

Errors are logged in *`input_filename`*`_error.txt`

Event flow - `MultiRowEventLoad`
Rejected records are written to *`input_filename`*`_error.csv`

Errors are logged in *`input_filename`*`_error.txt`

Event flow - `StdEventLoad`
Failed event messages are written to the error queue `PMQ.EVENT.ERROR`

Errors are logged in `EventError.txt`

`PMQIntegration` flow
Failed event request and web service fault messages are written to the error queue: `PMQ.INTEGRATION.ERROR`

Errors are logged in `IntegrationError.txt`

Maximo flow - `Maximomasterdataasset`, `Maximomasterdataclassification`, `Maximomasterdatalocation`
Rejected records are written to *`input_filename`*`_error.xml`

Errors are logged in *`input_filename`*`_error.txt`

Maximo flow - `WorkorderCreation`
Failed Maximo requests and web service fault message are written to the error queue: `PMQ.MAXIMO.ERROR`

Log files generated during the installation process

Errors that occur during the prerequisite checks that happen during the installation process are written to the following location on the node where installation is taking place:

```
/var/IBMPMQ/PreReq.log
```

The following errors can be reported:

Error, Can't proceed as the user is a Non Root User
The installer must be run as a Root user.

Error, *<package_name>* **not installed**
Install the package using the following command:

```
# rpm -i software-2.3.4.rpm
```

Error *<MEM>* **is less than the required 8GB memory**
Ensure that there is 8 GB of memory available.

Error *<TMP>* **KB is available for TMP, need 100GB**

Error *<File System Size in KB>* **KB is available for /opt, need 100GB**
The /opt filesystem must have a minimum of 100 GB space for installation.

Error / filesystem requires more than 150 GB of freespace
Ensure that the file system has at least 150 GB available.

Error *<Version information>* **is not supported for IBMPMQ**
Uninstall the current DB2 version, and ensure that the system is clean.

Error, Port *<portno>* **is not open**
Ensure that the port is open to the firewall, if used.

Error, Connection to *<SERVER>* **on port** *<PORT>* **failed**
Ensure that the port is open to the firewall, if used.

Notices

This information was developed for products and services offered worldwide.

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.

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.

The following terms are trademarks or registered trademarks of other companies:

- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Index