IBM Predictive Maintenance and Quality
Version 2.0

Solution Guide

Note
Before using this information and the product it supports, read the information in "Notices" on page 207.

Product Information

This document applies to IBM Predictive Maintenance and Quality Version 2.0 and may also apply to subsequent releases.

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 perform the following tasks:

- 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 help you to achieve these goals:

- 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 information 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 documentation on the web, including all translated documentation, access IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter).

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products. Some of the components included in the IBM Predictive Maintenance and Quality solution have accessibility features. For more information, see Appendix A, “Accessibility features,” on page 125.

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. What's new

There are several new and changed features that affect IBM Predictive Maintenance and Quality for this release.

Quality early warning system

The quality early warning system (QEWS) uses the advanced analytics, visualization, and workflow in IBM Predictive Maintenance and Quality to detect quality problems earlier and more definitively.

QEWS monitors large amounts of quality data automatically, with earlier alerts, definitive alerts, and intelligent prioritization. For more information about the QEWS, see Chapter 6, "Quality early warning system use cases," on page 61.

Maintenance analytics

IBM Predictive Maintenance and Quality maintenance analytics predicts the optimal conditions for a resource by analyzing historical, planned, and breakdown maintenance work orders. The analysis is used to recommend customized changes to the maintenance schedule of the resource.

Predictive Maintenance and Quality maintenance analytics has the following features:

- Advanced modeling that draws maintenance insights from intermittent and censured breakdown and scheduled maintenance events.
- Customized analytics in a format that is compatible with other analytics software. Compatibility with other analytics software enables seamless integration, comparison, and substitution between Predictive Maintenance and Quality and other statistical products.
- Works independently of sensor data. Predictive Maintenance and Quality can produce effective insights before sensor data is at optimal maturity for effective predictive modeling. This capability provides quicker return on investment.
- Intelligent prediction of a machinery health and recommendation that is based on models with predictions customized for each resource.
- Automatic training and refresh of prediction models at preset intervals.
- Option to manually refresh the deployed model during ad hoc or when a sudden change in sensor data occurs.
- Automatic filtering of resources with scarce data for predictive model generation.
- Can be used to combine text analytics, or other custom and compatible analytics format to predict machinery health and maintenance recommendations.

Top failure predictors

This feature helps you to understand the top reasons that a resource fails. You can use the supplied statistical process control (SPC) charts to perform root cause analysis that leads to pattern discovery.

IBM Predictive Maintenance and Quality top failure predictors has the following features:

- Ability to analyze and discover the top percentile or number of parameters that predict the failure or optimal health of a resource.
- Ability to drill through for a selected resource to see a detailed analysis of its patterns and anomaly detection.
- Custom analytics with any number of parameters or profiles for a resource.
- Ability to perform predictor importance analysis on custom profiles, features, and calculations by creating new custom profiles. For example, you cna create profiles for cumulative humidity instead of absolute humidity.

For more information about the TopN Failure Analysis Report and the SPC reports, see Chapter 9, "Reports and dashboards," on page 103.

Reports

IBM Predictive Maintenance and Quality provides new reports for statistical process control and the quality early warning system (QEWS). There is a new key performance indicator (KPI) trend chart. Health score reporting is improved.

The Advanced KPI Trend chart displays separate charts for multiple profiles across multiple resources. The TopN Failure Analysis report shows the profiles that contribute to the failure of a resource.

The Maintenance Overview Report shows the sensor health score, maintenance health score, and the integrated health score for the resources at a location.

Statistical process control charts

The following new reports analyze statistical process control:

- SPC - Histogram
- SPC - X Bar R/S Chart

Quality early warning system charts

The following new reports support QEWS:

- QEWS - Inspection Chart
- QEWSL - Warranty Chart

For more information, see Chapter 9, "Reports and dashboards," on page 103.

Analytics Solutions Foundation

You can use IBM Analytics Solutions Foundation to extend or modify IBM Predictive Maintenance and Quality.

Analytics Solutions Foundation is an alternative to using the flat file application programming interface (API) to extend the Predictive Maintenance and Quality solution. Analytics Solutions Foundation helps you to define orchestrations without writing code to integrate the API.

For more information, see Appendix B, "Analytics Solutions Foundation," on page 127.

Maximo integration

IBM Predictive Maintenance and Quality and IBM Maximo® are seamlessly integrated.

The integration with Maximo includes the following features:

- Support for updating an existing maintenance work order in Maximo with the maintenance recommendation from Predictive Maintenance and Quality.
- Support for processing maintenance Maximo work orders in batch and real time.
- Support for master data loading in real time.

Accessibility

The reports in IBM Predictive Maintenance and Quality are accessible.

For more information, see Appendix A, "Accessibility features," on page 125.

Chapter 2. 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 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.
- Make adjustments to predictive maintenance schedules and tasks to reduce repair costs and minimize 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.
- Identify quality and reliability issues definitively and in a timely way.

Instrumented assets generate data such as device ID, timestamp, temperature, and status code. This data can be collected and used with maintenance records and other data 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.

Predictive Maintenance and Quality can also detect quality and reliability problems faster than traditional techniques.

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, 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
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 Integration Bus 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 Integration Bus.
- Customize scoring during event processing by using IBM Integration Bus.
- Customize or create the message flows to orchestrate the activities by using IBM Integration Bus.
- 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 E, "IBM Predictive Maintenance and Quality Artifacts," on page 187.

Integration with asset management and manufacturing execution systems

Asset management and manufacturing execution systems are an important source of master data and event data. You can feed recommendations and forecasts produced by IBM Predictive Maintenance and Quality into these systems to close the loop and perform action.

Predictive Maintenance and Quality can create work orders in IBM Maximo Asset Management based on recommendations from predictive scoring and decision management. Predictive Maintenance and Quality contains the APIs for integration with these systems and technology for building connectors to the systems. Predictive Maintenance and Quality includes a prebuilt adapter for integration with Maximo.

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 3. Orchestration

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

Message flows

Orchestration is achieved with message flows in IBM Integration Bus.

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

Message flows are supplied with IBM Predictive Maintenance and Quality and must be customized with IBM Integration Bus. The message flows are organized into the following applications:

- **PMQEventLoad**
- **PMQMasterDataLoad**
- **PMQMaximoOutboundIntegration**
- **PMQMaintenance**
- **PMQModelTraining**
- **PMQQEWSInspection**
- **PMQQEWSIntegration**
- **PMQQEWSWarranty**
- **PMQTopNFailure**

For more information about developing message flows, see IBM Integration Bus Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSMKHH_9.0.0/com.ibm.etools.mft.doc/bi12005_.htm).

By default, IBM Integration Bus 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 IBM Integration Bus 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 IBM Integration Bus 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 IBM Integration Bus 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 IBM Integration Bus 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 55.

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 Integration Bus describes the transformation of the data into to the model structure.
3. IBM Analytical Decision Management Model is invoked through a web services interface.
4. A map that is defined in IBM Integration Bus 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 Integration Bus describes the transformation of the data to the structure of an external web service.
3. The external web service is called.

Example of an orchestration XML file

An example file, inspection.xml, demonstrates the purpose and structure of an orchestration file.

Each orchestration flow can be defined in a separate XML file. The file defines the behavior of the orchestration steps. A mapping determines the orchestrations to be performed for an event with an event orchestration key code.

In this example scenario, there are two kinds of events: production and inspection. Therefore, there are two event orchestration key codes, one for each type of event.

The example file "inspection.xml" on page 12 determines the orchestration for an inspection event.

Description

The first part of the file `inspection.xml` lists the event type, the adapter class, and configuration that is required for the particular class of adapter:

- `<event_orchestration_mapping>`

  The type of event is defined as an inspection.
- `<adapter_class>`

  The adapter class that will be executed, in this case `ProfileAdapter`, is called in the step.
- `<adapter_configuration>`

  The profile adapter requires configuration to determine how observations with a specific measurement type will update specfic profile tables.

The remainder of the file specifies how two specific profiles will be updated, depending on whether the measurement type has a value of `INSPECT` or `FAIL`:

- `<observation_profile_update>`

  If the measurement type has a value of INSPECT

  `<profile_update_action>` The `PRODUCT_KPI` table is updated with the shared calculation of `Product_KPI_Inspect_count`. This calculation produces the value for the number of days when an inspection has taken place.
- `<observation_profile_update>`

  If the measurement type has a value of FAIL

  `<profile_update_action>` The `PRODUCT_KPI` table is updated with the shared calculation of `PRODUCT_KPI_FAIL_COUNT`. This calculation produces the value for the number of times an asset has failed.

inspection.xml

The file inspection.xml contains the following code:

```
<event_orchestration_mapping>
   <event_orchestration_key_cd>inspection</event_orchestration_key_cd>
   <orchestration_cd>pmq.inspection</orchestration_cd>
</event_orchestration_mapping>

<orchestration>
  <orchestration_cd>pmq.inspection</orchestration_cd>
    <step>
   <adapter_class>com.ibm.analytics.foundation.adapter.profile.ProfileAdapter</adapter_class>
   <adapter_configuration xsi:type="ns3:profile_adapter_configuration">
        <observation_profile_update>
           <observation_selector table_cd="EVENT_OBSERVATION">
              <observation_field_value>
                <field_name>MEASUREMENT_TYPE_CD</field_name>
           <value>INSPECT</value>
              </observation_field_value>
           </observation_selector>

           <profile_update_action>
              <profile_row_selector>
                 <shared_selector_cd>PRODUCT_KPI</shared_selector_cd>
              </profile_row_selector>
                <shared_calculation_invocation_group_cd>PRODUCT_KPI_INSPECT_COUNT
                </shared_calculation_invocation_group_cd>
           </profile_update_action>
        </observation_profile_update>

        <observation_profile_update>
       <observation_selector table_cd="EVENT_OBSERVATION">
          <observation_field_value>
        <field_name>MEASUREMENT_TYPE_CD</field_name>
     <value>FAIL</value>
        </observation_field_value>
    </observation_selector>
           <profile_update_action>
       <profile_row_selector>
          <shared_selector_cd>PRODUCT_KPI</shared_selector_cd>
       </profile_row_selector>
       <shared_calculation_invocation_group_cd>
PRODUCT_KPI_FAIL_COUNT</shared_calculation_invocation_group_cd>
           </profile_update_action>
       </observation_profile_update>
      </adapter_configuration>
    </step>
</orchestration>
```

Chapter 4. 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 Integration Bus 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 Integration Bus reads and processes it and then removes it from the directory. IBM Integration Bus 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.

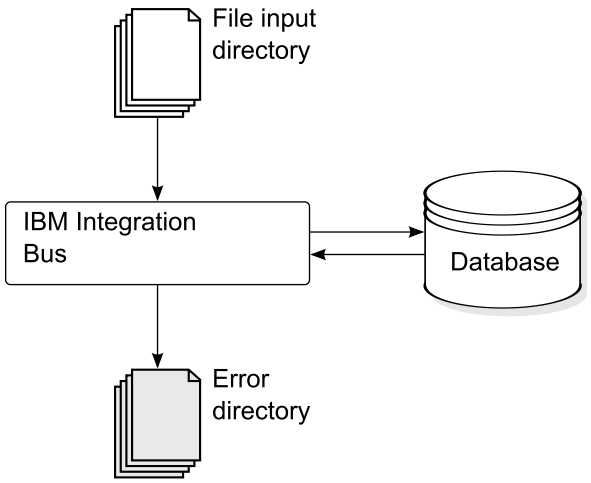


Figure 1. Master data process

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 163.
- 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 45.

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 C, "The flat file API," on page 149.

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 Integration Bus 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 `Resource_CD1` and `Resource_CD2`, or by `operator_cd`.

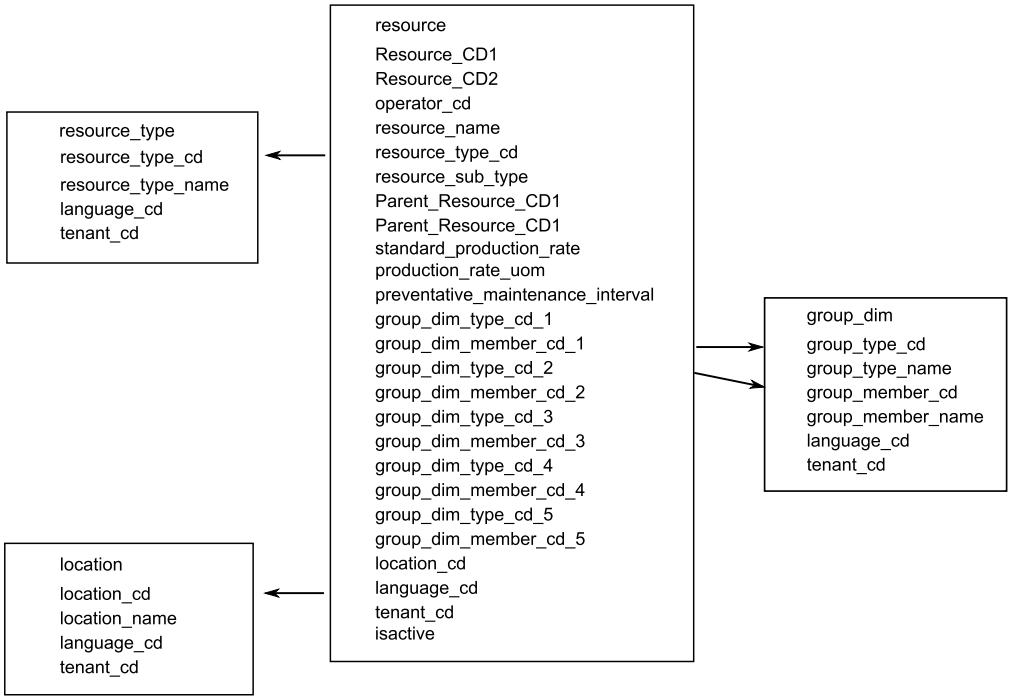


Figure 2. Codes used to identify and to reference records

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 18.
3. Import the metadata (Company deployment), see "Importing metadata into InfoSphere MDM Collaboration Server" on page 21.
4. Configure the InfoSphere MDM Collaboration Server user interface, see "Configuring the IBM InfoSphere MDM Collaboration Server user interface" on page 19.

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 19.

For additional information about using InfoSphere MDM Collaboration Server, see *Collaborative authoring with InfoSphere MDM Collaboration Server*. This is available from IBM Master Data Management Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSWSR9_11.0.0).

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 Integration Bus.
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 IBM Integration Bus 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 17.

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 Integration Bus server.

Before you begin

For information about the variables that are used, see "IBM Master Data Management Collaboration Server dynamic references" on page 17.

About this task

The IBM Integration Bus 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 Integration Bus 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 Integration Bus 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 17.

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.

Solution XML file

The solution XML file defines the master data. The master tables and supporting tables are defined so that database tables may be generated and upserts carried out.

The solution XML file defines the following types of tables:

- Master tables
- Event tables
- Profile or KPI tables

The LANGUAGE table and the columns are defined as shown in the following XML code:

```
<table table_cd="LANGUAGE" is_surrogate_primary_key="true"
     validator_class="com.ibm.pmq.master.validators.LanguageValidate">
   <column column_cd="LANGUAGE_CD" type="string" size="50" is_key="true"/>
   <column column_cd="LANGUAGE_NAME" type="string" size="200"/>
   <column column_cd="DEFAULT_IND" type="int"/>
</table>
```

The TENANT table and the columns are defined as shown in the following XML code:

```
<table table_cd="TENANT" is_surrogate_primary_key="true"
     validator_class="com.ibm.pmq.master.validators.TenantValidate">
   <column column_cd="TENANT_CD" type="string" size="100" is_key="true"/>
   <column column_cd="TENANT_NAME" type="string" size="200"/>
   <column column_cd="DEFAULT_IND" type="int"/>
</table>
```

The definitions of the LANGUAGE, TENANT, CALENDAR, EVENT_TIME, and KEYLOOKUP tables must not be modified and must be included in the solution XML file.

Master tables include language and tenant support. They are defined by using attributes of the table. For example, the following definition of the `Master_Location` table includes the attributes `is_multilanguage`, `is_multitenant`, and `is_row_deactivateable`. The value of "true" indicates that the table is multi-language, multi-tenant, and the table includes a column that indicates whether the row is enabled (active) or disabled (deactivated):

```
<table table_cd="MASTER_LOCATION"
   is_multilanguage="true"is_multitenant="true" is_row_deactivateable="true"
   is_surrogate_primary_key="true"
   validator_class="com.ibm.pmq.master.validators.LocationValidate">
      <column column_cd="LOCATION_CD" is_key="true" size="100"
type="string"/>
      <column column_cd="LOCATION_NAME" is_key="false" size="1024"
type="string"/>
      <column column_cd="REGION_CD" is_key="false" size="50"
type="string" is_nullable="true"/>
      <column column_cd="REGION_NAME" is_key="false" size="200"
type="string" is_nullable="true"/>
      <column column_cd="COUNTRY_CD" is_key="false" size="50"
type="string" is_nullable="true"/>
      <column column_cd="COUNTRY_NAME" is_key="false" size="200"
type="string" is_nullable="true"/>
      <column column_cd="STATE_PROVINCE_CD" is_key="false" size="50"
type="string" is_nullable="true"/>
      <column column_cd="STATE_PROVINCE_NAME" is_key="false" size="200"
type="string" is_nullable="true"/>
      <column column_cd="CITY_NAME" is_key="false" size="200"
```

```
type="string" is_nullable="true"/>
      <column column_cd="LATITUDE" is_key="false" size="10,5"
type="decimal" is_nullable="true"/>
      <column column_cd="LONGITUDE" is_key="false" size="10,5"
type="decimal" is_nullable="true"/>
</table>
```

References

The tables defined in the solution XML file (event, master data, and profile) may also define references to master data tables. For example, `Master_Product_Parameters` references the `Master_Product` table. To reference a specific `Master_Product` row, the flows for `Master_Product_Parameters` take the business keys `Product_Cd` and Product_Type_Cd as input parameters in the CSV file. The following definition for `Master_Product_Parameters` is an example of how to define a reference. `Product_Id` is an identifier of the reference to the `Master_Product` table. The business keys of the `Master_Product` table, `Product_type_cd`, and `Product_cd`, along with `Tenant_cd`, are used to reference a `Master_Product` row:

```
<table table_cd="MASTER_PRODUCT_PARAMETERS"
        is_multilanguage="true" is_multitenant="true">
    <column column_cd="PARAMETER_NAME" type="string" size="50"
        is_key="true"/>
    <column column_cd="PARAMETER_VALUE" type="double"
        is_key="false"/>
    <reference reference_cd="PRODUCT_ID"
        table_reference="MASTER_PRODUCT" is_key="true"/>
</table>
```

The following example shows a more explicit table definition for `Master_Product_Parameters`. This method can be used to make the column names different than the business keys. That is, when `table_column_cd` is different from `reference_colum_cd.`. You must use this mapping to have unique `reference_column_cd` values when there is more than one reference to the same table:

```
<table table_cd="MASTER_PRODUCT_PARAMETERS"
      is_multilanguage="true" is_multitenant="true">
   <column column_cd="PARAMETER_NAME" type="string" size="50"
      is_key="true"/>
   <column column_cd="PARAMETER_VALUE" type="double"
      is_key="false"/>
   <reference reference_cd="PRODUCT_ID"
      table_reference="MASTER_PRODUCT" is_key="true">
      <column_mapping table_column_cd="PRODUCT_CD" reference_column_cd="PRODUCT_CD"/>
      <column_mapping table_column_cd="PRODUCT_TYPE_CD"
          reference_column_cd="PRODUCT_TYPE_CD"/>
   </reference>
</table>
```

Hierarchy table structures

The solution XML file manages the hierarchical structures that are used in IBM Predictive Maintenance and Quality. IBM Predictive Maintenance and Quality maintains hierarchical structures for two Master tables, Resource and Process.

`Master_Resource_hierarchy` is generated based on the solution XML. The following example shows the definition of `Master_Resource` in the solution XML file. The `self_reference` element means that there is a circular reference to the table. The circular reference is required to maintain the hierarchy. The `number_of_levels`

property defines the number of levels of hierarchy. The duplicate_column_cd element refers to the column names that appear across each level of the defined number_of_levels property:

```
<self_reference reference_cd="PARENT_RESOURCE_ID" number_of_levels="10">
   <column_mapping table_column_cd="RESOURCE_CD1"
reference_column_cd="PARENT_RESOURCE_CD1" />
   <column_mapping table_column_cd="RESOURCE_CD2"
reference_column_cd="PARENT_RESOURCE_CD2" />
   <duplicate_column_cd>RESOURCE_CD1</duplicate_column_cd>
   <duplicate_column_cd>RESOURCE_CD2</duplicate_column_cd>
   <duplicate_column_cd>RESOURCE_NAME</duplicate_column_cd>
</self_reference>
```

Master_Process_Hierarchy is generated based on the solution XML. The following example shows the definition of Master_Process in the solution XML file. For Master_Process_Hierarchy, hierarchical information for Process_CD and Process_Name is maintained across five levels:

```
<self_reference
   reference_cd="PARENT_PROCESS_ID" number_of_levels="5">
      <column_mapping table_column_cd="PROCESS_CD"
   reference_column_cd="PARENT_PROCESS_CD"/>
      <duplicate_column_cd>PROCESS_CD</duplicate_column_cd>
      <duplicate_column_cd>PROCESS_NAME</duplicate_column_cd>
</self_reference>
```

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
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 and attributes
Resource_CD1	string(50)	Either serial_no and model or operator_cd are required	ASSET.ASSETNUM
Resource_CD2	string(50)		ASSET.SITEID
resource_name	string(500)	Required	ASSET.DESCRIPTION
resource_type_cd	string(50)	Required	

Table 7. Fields in the resource table (continued)

Field	Type	Required or Optional	Maximo objects and attributes
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
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 Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSWK4A).

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 the following publish channels:
   - SPCLASSIFICATIONCHANNEL_R with object structure SPCLASSIFICATION
   - SPPUBLISHCHANNEL_R with object structure SPASSET
   - SPSAPUBLISHCHANNEL with object structure SPSERVICEADDRESS

   For each publish channel, perform the following action:

   - 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.

   The name of the external system must be `SPEXTSYSTEM`.

   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 Integration Bus 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.
   a. Name the publish channels as shown:

      SPPUBLISHCHANNEL
      For Asset.

      SPCLASSIFICATIONCHANNEL
      For Classification.

      SPSAPUBLISHCHANNEL
      For ServiceAddress.

   b. 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 data. For example, if you want to export assets with a specific classification then you must enter a filter expression such as `CLASSSTRUCTUREID='1012'`.

      To find the `CLASSSTRUCTUREID` that an asset belongs to, go to the **Specifications** tab of `ASSET`.

The **Specifications** tab contains classification information. The classification has a CLASSSTRUCTUREID associated with it, which you can see when you export the classification.

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 Integration Bus library.

Enabling master data loading in realtime mode

You can enable master data to load in realtime mode by creating publish channels and configuring their endpoints.

Procedure

1. Create new publish channel for real time master data loading.
   a. Select **Integration** > **Publish Channels** > **New**.
   b. Create the following publish channels:
      - SPCLASSIFICATIONCHANNEL_R, with object structure SPCLASSIFICATION
      - SPPUBLISHCHANNEL_R, with object structure SPASSET
      - SPSAPUBLISHCHANNEL, with object structure SPSERVICEADDRESS
   c. For each publish channel, select **Action** > **Enable Event Listeners**, and then select the **Enable Listener** check box.
2. Configure the Web service endpoints.
   a. Select **GoTo** > **Integration** > **Endpoint**.
   b. Select **New Endpoint** and enter the following information:
      - In the **Endpoint Name** field, type AENDPOINT
      - In the **Handler type** field, type WEBSERVICE
      - In the **EndPointURL** field, type http://*ESB_Node_IP_address*:7800/meaweb/services/asset
      - In the **ServiceName** field, type asset
   c. Select **New Endpoint** and enter the following information:
      - In the **Endpoint Name** field, type CENDPOINT
      - In the **Handler type** field, type WEBSERVICE
      - In the **EndPointURL** field, type http://*ESB_Node_IP_address*:7800/meaweb/services/classification
      - In the **ServiceName** field, type classification
   d. Select **New Endpoint** and enter the following information:
      - In the **Endpoint Name** field, type SAENDPOINT
      - In the **Handler type** field, type WEBSERVICE
      - In the **EndPointURL** field, type http://*ESB_Node_IP_address*:7800/meaweb/services/serviceaddress

- In the **ServiceName** field, type serviceaddress

3. Configure the external system to associate the publish channels and endpoints to the external system for webservice event notification of workorders.
   a. Select **GoTo** > **Integration** > **External Systems** > **filter** for EXTSYS2
   b. Select **Publish channels** > **Add New Row**.
      - Enter SPCLASSIFICATIONCHANNEL : CENDPOINT
      - Select the **Enabled** check box.
   c. Select **Publish channels** > **Add New Row**.
      - Enter SPPUBLISHCHANNEL : AENDPOINT
      - Select the **Enabled** check box.
   d. Select **Publish channels** > **Add New Row**.
      - Enter SPSAPUBLISHCHANNEL : SAENDPOINT
      - Select the **Enabled** check box.

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 IBM Maximo Asset Management Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSWK4A).
2. Create a flow in IBM Integration Bus 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 49
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 56.

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 an IBM Integration Bus 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** IBM Integration Bus application.

For more information about creating an enterprise service, see IBM Maximo Asset Management Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSWK4A).

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`) and **Select Action**, **Deploy to Product Web Service Container**, **Deploy Web Service** and click **OK**.
4. 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. 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`.
5. 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.

Configuring work orders in Maximo

In Maximo, you can configure Maximo for OutBound work orders using either an XML file in batch mode or using a web service in realtime mode.

You can also configure Maintenance work orders to be updated with recommendations in IBM Predictive Maintenance and Quality (PMQ).

Configuring Maximo for OutBound work orders using a web service

You can configure Maximo for OutBound work orders using a web service in realtime mode.

Procedure

1. Define the object structure.
   a. Edit the base object structures available in IBM Maximo Asset Management (MXWO) to add the Service Address object reference to it.

Tip: This ensures that workorder events that are generated from Maximo contain the field reference that is related to the service address.

b. Select **GoTo** > **Integration** > **Object Structure** and search for MXWO.

c. Click the new row and enter the following information
   - In the Object field, type WOSERVICEADDRESS
   - In the Parent Object field, type WORKORDER
   - In the Object Location Path field, type WOSERVICEADDRESS
   - In the Relationship field, type SERVICEADDRESS

The window should appear similar to the following figure.

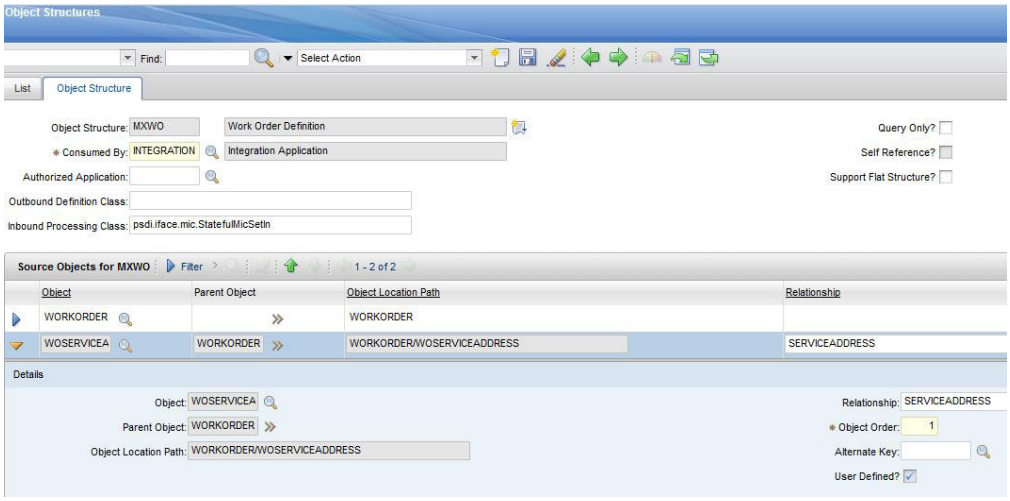


Figure 3. Defining the object structure

2. Export the Object Structure schema for MXWO.
   - Select **Action** > **Generate Schema/View XML**.
     See the following figure.

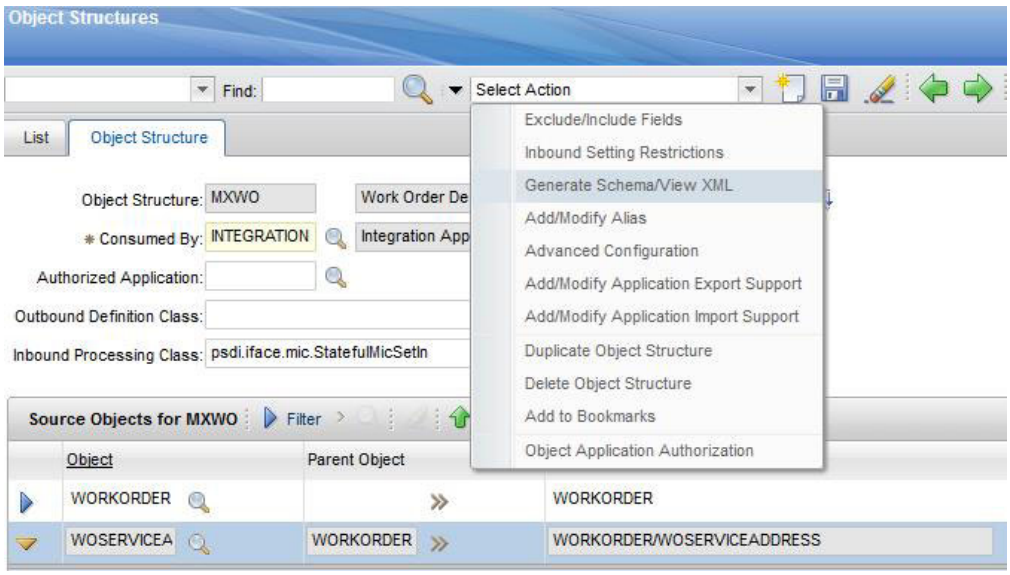


Figure 4. Exporting the Object Structure schema

The generated schema MXWOService.xsd is stored in the same location as the data export XML files. This schema is used for configuring in the mapping node of IIB for work order to event transformation.

3. Enable the Publish channel event listener.
   a. Select **Publish Channel** and then select **MXWOInterface**. The work order publish channel appears.
   b. Select **Action** > **Enable Event Listeners**.
      See the following figure.

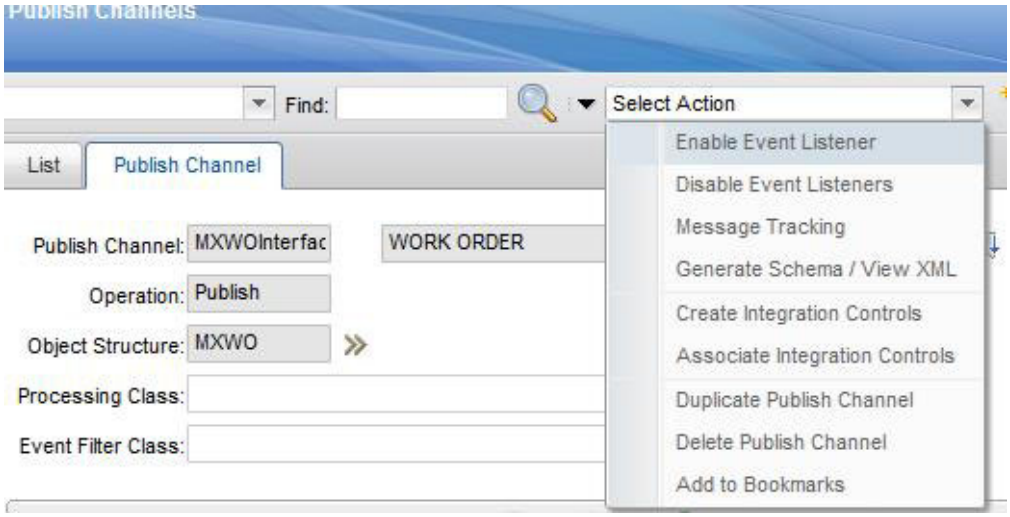


Figure 5. Enabling the Publish channel event listener

      The **Enable Listener** check box is enabled.

4. Add a new processing rule for the publish channel MXWOInterface.
   a. Select **New Row**.
   b. Specify the following values:
      - In the **Rule** column, type PMQ.
      - In the **Description** column, type PMQ Maintenance related Rule.
      - In the **Action** column, specify SKIP.
      - In the **Enabled** column, select the checkbox.

      See the following figure.

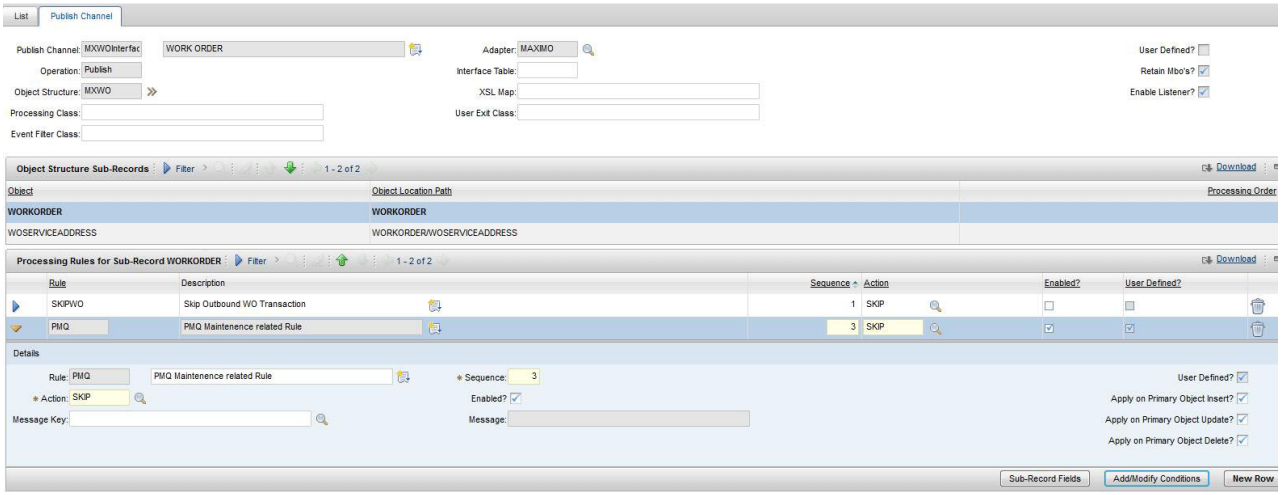


Figure 6. Adding a new processing rule

c. Select **Add/Modify Conditions**.
d. Select **New Row**.
e. Specify the following values:
   - In the **Field** field, specify DESCRIPTION.
   - In the **Evaluation Type** field, specify NOTEQUALS.
   - In the **Evaluation When** field, specify ALWAYS.
   - In the **Value** field, specify MAINTENANCE.

   A condition is added to skip the MAINTENANCE work order. See the following figure.

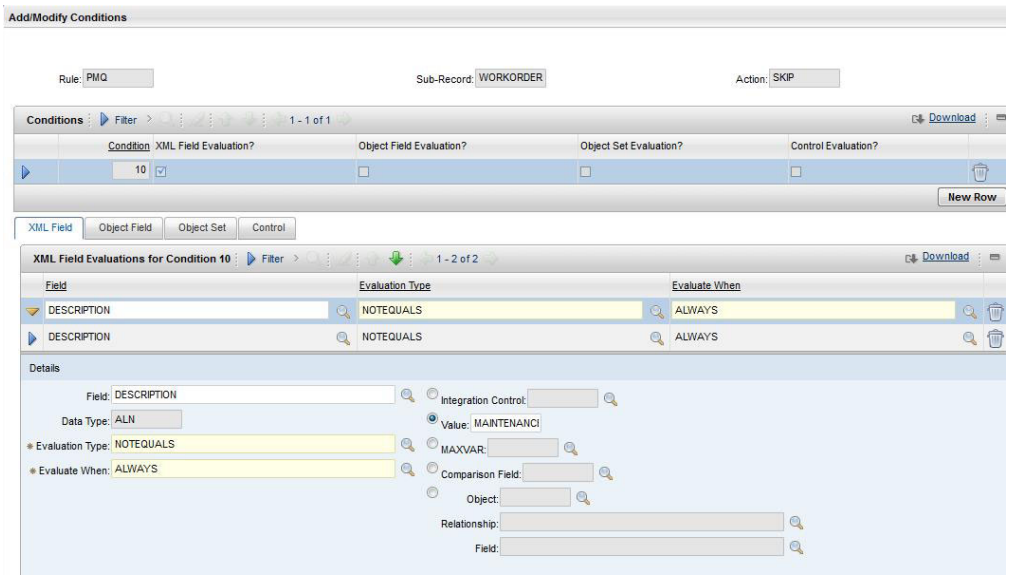


Figure 7. Adding a condition to skip the MAINTENANCE work order

f. Select **New Row**.
g. Specify the following values:
   - In the **Field** field, specify DESCRIPTION.
   - In the **Evaluation Type** field, specify NOTEQUALS.

- In the **Evaluation When** field, specify ALWAYS.
- In the **Value** field, specify BREAKDOWN.

A condition is added to skip the BREAKDOWN work order. See the following figure.

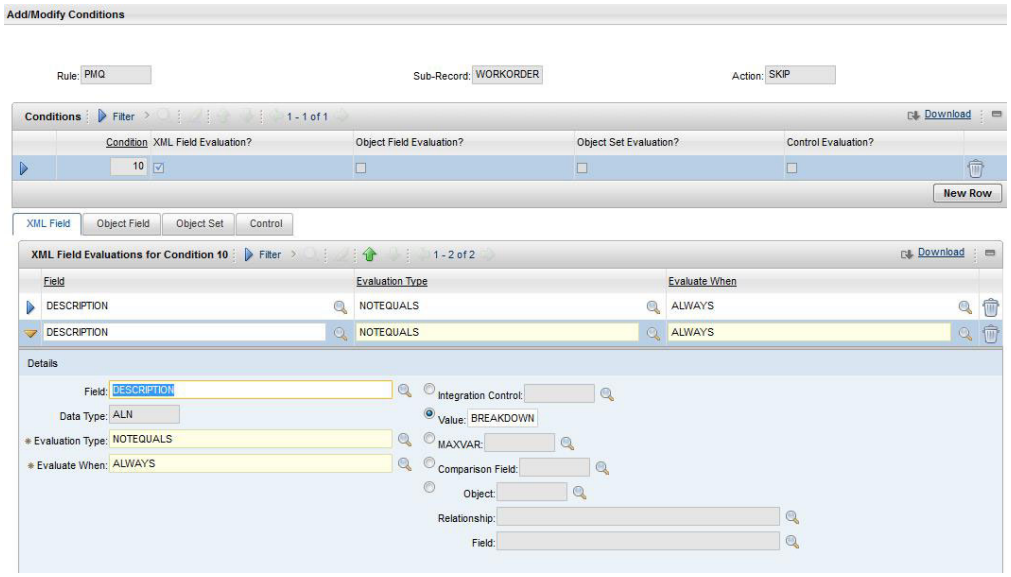


Figure 8. Adding a condition to skip the BREAKDOWN work order

5. Activate the JMS cron task.
   a. Select **GoTo** > **System Configuration** > **Platform Configuration** > **Cron Task Setup**.
   b. Filter on **JMSQSEQCONSUMER**.
   c. Select the **SEQQOUT** cron task instance name.
   d. Click **Active** and save the record.

   The JMS cron task is activated. See the following figure.

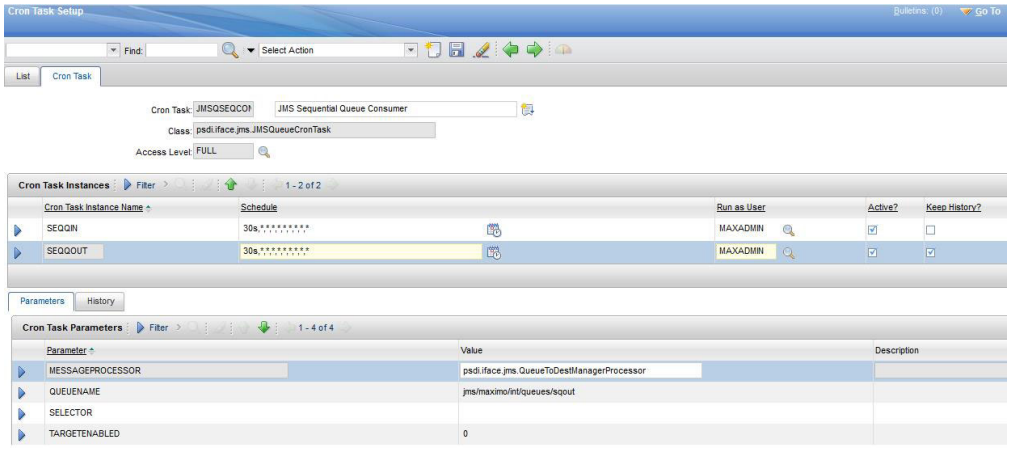


Figure 9. Activating the JMS cron task

6. Configure the Web service endpoint.
   a. Select **GoTo** > **Integration** > **Endpoint**.
   b. Select **New Endpoint** and enter the following information:

- In the **Endpoint Name** field, type MXWOENDPOINT
- In the **Handler type** field, type WEBSERVICE
- In the **EndPointURL** field, type http://*ESB_Node_IP_address*:7800/meaweb/services/MXWOInterface
- In the **ServiceName** field, type OutboundWOService

See the following figure.

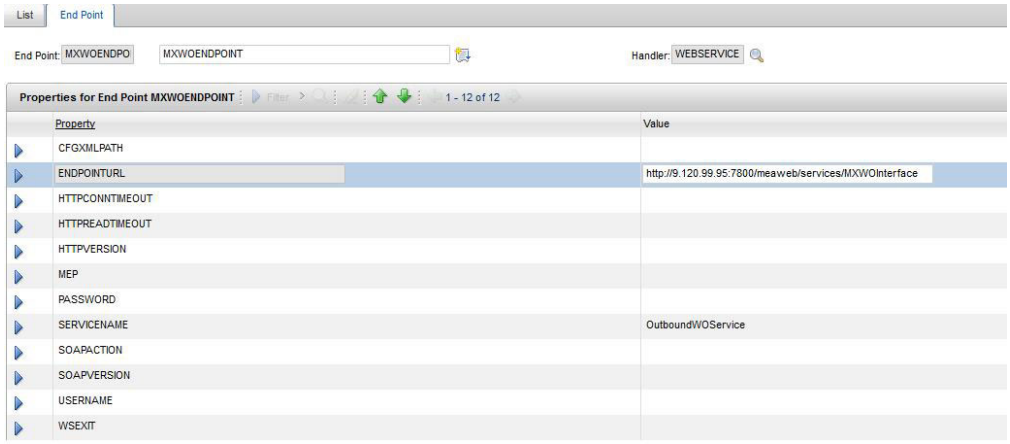


Figure 10. Configuring the web service endpoint

7. Configure the external system to associate publish channels and endpoints to the external system for web service event notification of work orders.
   a. Select **GoTo** > **Integration** > **External Systems** > **New External System**
   b. Enter the following information:
      - In the **System** field, type EXTSYS2
      - In the **Description** field, type PMQ External System
      - In the **EndPoint** field, type MXXMLFILE
      - In the **Outbound Sequential Queue** field, type jms/maximo/int/queues/sqout
      - In the **Inbound Sequential Queue** field, type jms/maximo/int/queues/sqin
      - In the **Inbound Continuous Queue** field, type jms/maximo/int/queues/cqin
      - Select the **Enabled** check box.
   c. Select **Publish channels** > **Add New Row**.
      - Add a New Row to add MXWOInterface to the publish channel with Endpoint as MXWOENDPOINT.
      - Select the **Enabled** check box.

      See the following figure.

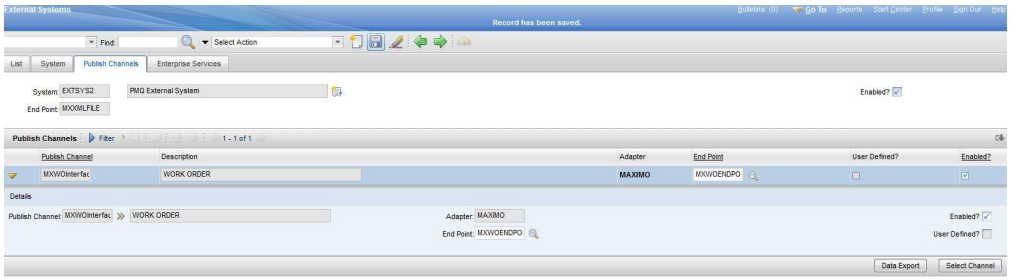


Figure 11. Adding MXWOInterface to the publish channel

Configuring Maximo for OutBound work orders using an XML file

You can configure Maximo for OutBound work orders using an XML file in batch mode.

Procedure

1. Create a new publish channel SPWO.
   a. Select **GoTo** > **Integration** > **Publish Channels**.
   b. Enter the following information:
      - In the **Publish Channel** field, type SPWO.
      - In the **Description** field, type PMQ WorkOrder Publish Channel.
      - In the **Object Structure** field, type MXWO.

      See the following figure.

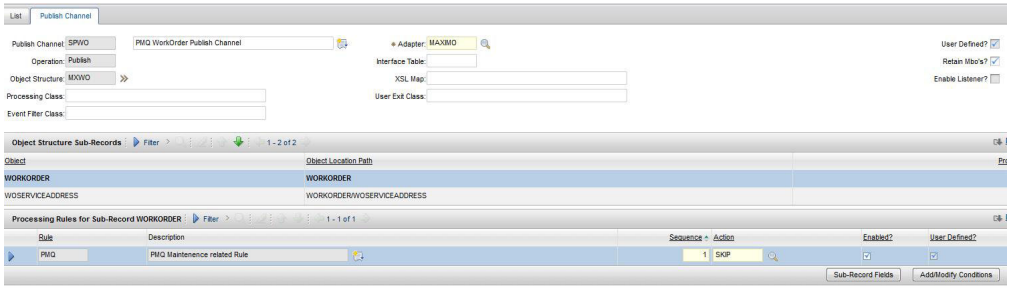


Figure 12. Create a new publish channel SPWO

2. Add a new processing rule for the publish channel SPWO.
   a. Select **New Row**.
   b. Specify the following values:
      - In the **Rule** column, type PMQ.
      - In the **Description** column, type PMQ Maintenance related Rule.
      - In the **Action** column, specify SKIP.
      - In the **Enabled** column, select the checkbox.

      See the following figure.

Figure 13. Adding a new processing rule for the publish channel SPWO

c. Select **Add/Modify Conditions**.

d. Select **New Row** under XML field evaluation.

e. Specify the following values:

- In the **Field** field, specify DESCRIPTION.
- In the **Evaluation Type** field, specify NOTEQUALS.
- In the **Evaluation When** field, specify ALWAYS.
- In the **Value** field, specify MAINTENANCE.

A condition is added to skip the MAINTENANCE work order. See the following figure.

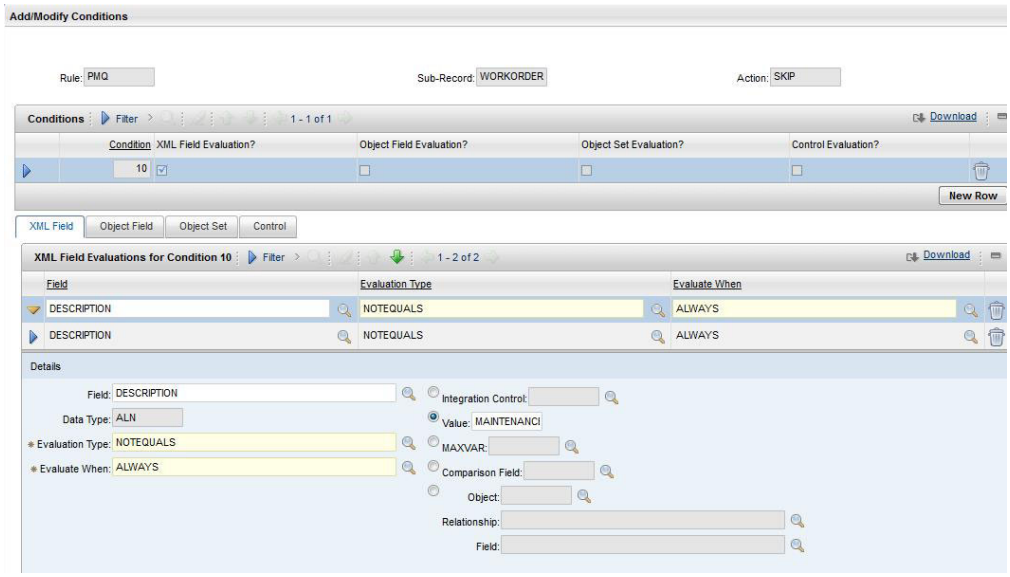


Figure 14. Adding a condition to skip the MAINTENANCE work order

f. Select **New Row** under XML field evaluation.

g. Specify the following values:

- In the **Field** field, specify DESCRIPTION.
- In the **Evaluation Type** field, specify NOTEQUALS.
- In the **Evaluation When** field, specify ALWAYS.
- In the **Value** field, specify BREAKDOWN.

A condition is added to skip the BREAKDOWN work order. See the following figure.

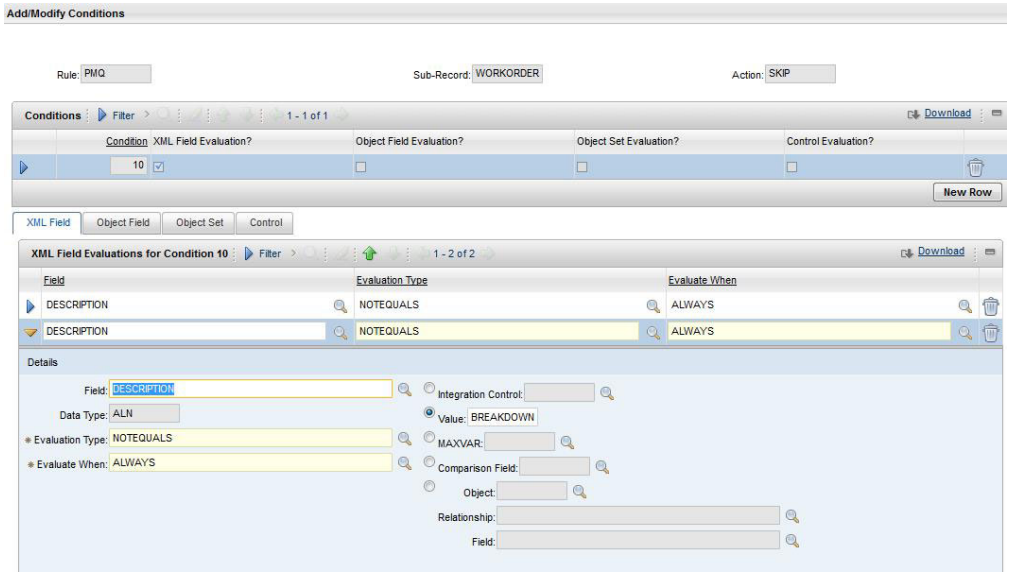


Figure 15. Adding a condition to skip the BREAKDOWN work order

3. Configure the external system to associate the publish channel and endpoint to the external system for the XML export of work orders.
   a. Select **GoTo** > **Integration** > **External Systems**.
   b. Filter on SPEXTSYSTEM.
   c. Select **Publish channels filter**.
   d. Enter the following information:
      - In the **Publish Channel Name** field, type SPWO
      - In the **EndPoint** field, type MXXMLFILE
      - Enable the MXWOInterface for the external system SPEXTSYSTEM by selecting the **Enabled** check box.
      - Activate the external system (SPEXTSYSTEM) by selecting by selecting the **Enabled** check box.

      See the following figure.

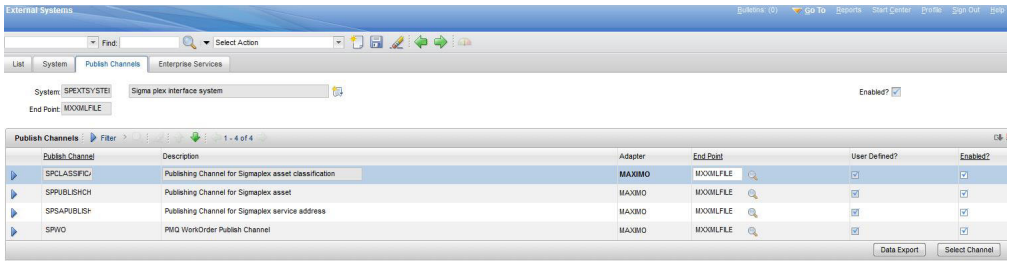


Figure 16. Enabling the external system SPEXTSYSTEM

Configuring Maximo to update recommendations in a work order

You can configure Maximo so that Maintenance work orders are updated in PMQ with PMQ recommendations.

The work order status is changed to `CHANGED` and Memo is updated to `Refer LONGDESCRIPTION for PMQ recommendation`. PMQ recommendation will get updated in the `LONGDESCRIPTION` field of PMQ.

The Maximo configuration described in this section creates the custom status `CHANGED`. The custom status `CHANGED` can be used to filter out all the work orders which were updated by PMQ with the recommendations.

Procedure

1. In Maximo, select **GoTo** > **System Configuration** > **Platform Configuration** > **Domains**.
2. Find the SYNONYM domain WOSTATUS to which you want to add a synonym value.

   See the following figure.

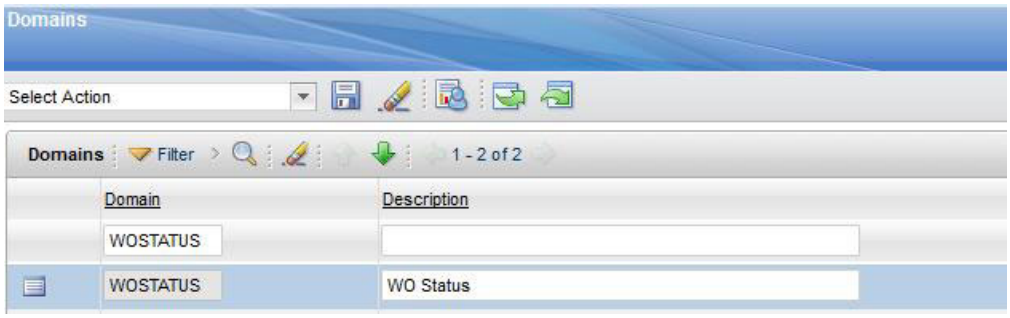


Figure 17. Finding the SYNONYM domain WOSTATUS

3. Click the **Edit details** icon.
4. Select **New Row** and specify the following values:
   - In the **Internal Value** field, specify WAPPR.
   - In the **Value** field, specify Change.
   - In the **Description** field, specify Recommendation Updated.

   See the following figure.

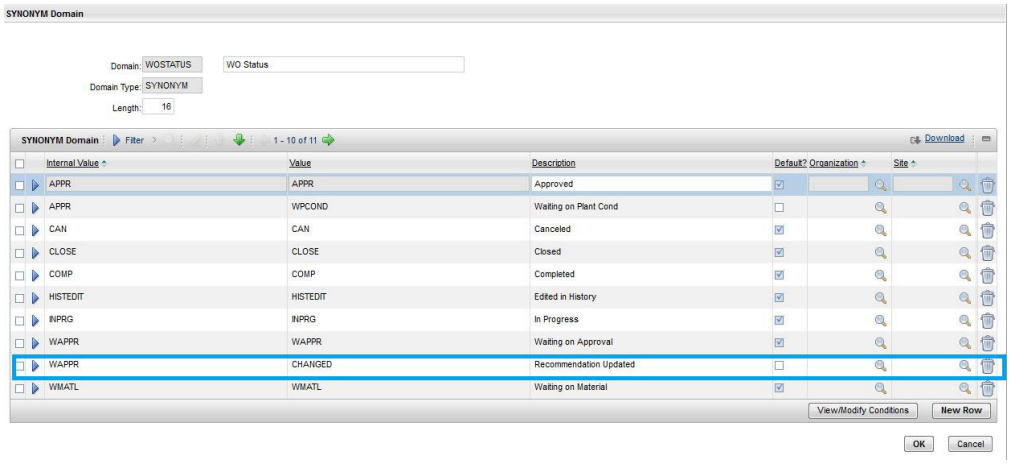


Figure 18. Specifying the values of the new row.

Viewing work orders updated with PMQ recommendations

You can view work orders that have been updated with recommendations from IBM Predictive Maintenance and Quality.

Procedure

1. Select **Goto** > **Work Orders** > **Work Order Tracking**.
2. Select **Filter** and, in the **STATUS** field, specify CHANGED.
3. Open the work order and select the **Long description** button in the **Work Order** row.

   See the following figure.

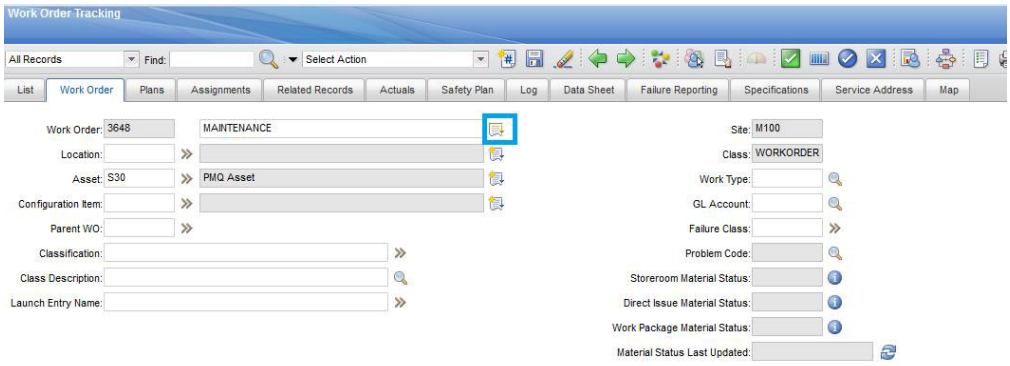


Figure 19. Opening the Long Description window

The PMQ recommendation appears, as shown in the following figure.

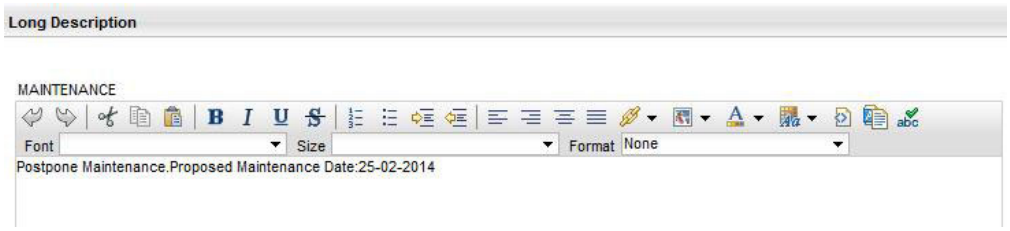


Figure 20. Viewing the PMQ recommendation

Creating a workorder in Maximo

You can create a MAINTENANCE workorder or a BREAKDOWN workorder in Maximo.

Procedure

1. Select **Goto** > **WorkOrders** > **Work Order Tracking** > **New Work Order**.
2. Specify the following values:
   - In the **Description** field, specify either BREAKDOWN or MAINTENANCE.
   - In the **Site** field, specify the Model No of the resource.
   - In the **Asset** field, specify the Serial No of the resource.
   - In the **Service Address** field, specify the location.
3. If you are creating a MAINTENANCE workorder, specify the following values:
   - In the **Scheduled Start** field, specify the scheduled maintenance start timestamp.
   - In the **Actual Start** field, specify the actual maintenance start timestamp, if applicable.
4. If you are creating a BREAKDOWN workorder, specify the following values:
   - In the **Reported Date** field, specify the Breakdown timestamp.

Results

For an example of a BREAKDOWN workorder, see the following figure.

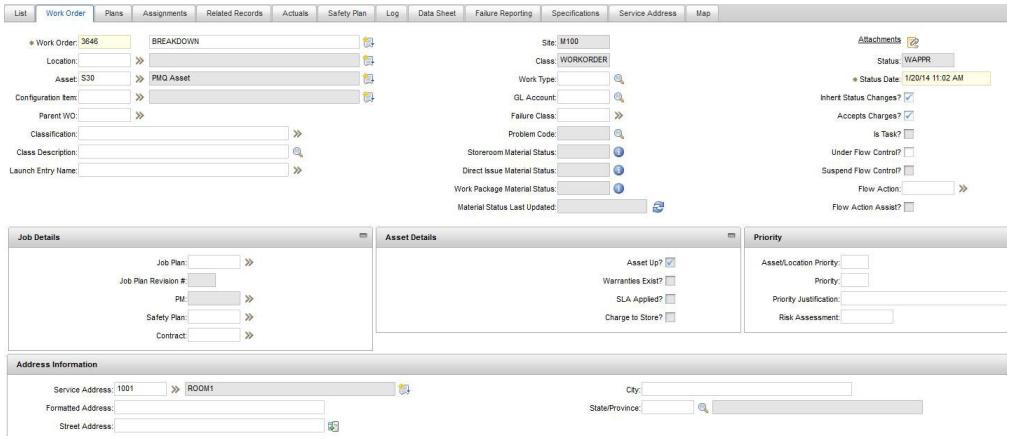


Figure 21. Creating a BREAKDOWN work order

Mapping work orders for maintenance

You can map IBM Predictive Maintenance and Quality (PMQ) events to work orders for maintenance.

There are two types of work orders you can use for maintenance:

- Maintenance work orders
- Breakdown work orders

Mapping PMQ events to a Maintenance work order

Two PMQ events are generated from a maintenance work order: one event for scheduled maintenance (SM) and one event for actual maintenance (AM).

The event mapping is shown in the following table

Table 8. Mapping PMQ events to a Maintenance work order

PMQ event	Work Order	Remarks
incoming_event_cd	WONUM	
event_type_cd		Hardcoded to "MAINTENANCE"
source_system_cd		Hardcoded to "MAXIMO"
process_cd		
production_batch_cd		
location_cd	WOSERVICEADDRESS.SADDRESSCODE	
event_start_time	Scheduled Start	Timestamp field
event_end_time		
event_planned_end_time		
tenant_cd		Hardcoded to "PMQ"
operator_cd		
Model	SITEID	

Table 8. Mapping PMQ events to a Maintenance work order (continued)

PMQ event	Work Order	Remarks
serial_no	ASSETNUM	
measurement_type_cd		Hardcoded to "SM" for scheduled maintenance event and "AM" for actual maintenance
observation_timestamp	Scheduled Start for scheduled maintenance Actual Start for actual maintenance	Timestamp field
value_type_cd		Hardcoded to "ACTUAL"
observation_text	DESCRIPTION_LONGDESCRIPTION	
Measurement		
material_cd		
multirow_no		Hardcoded to 1

Mapping PMQ events to a Breakdown work order

The event mapping is shown in the following table

Table 9. Mapping PMQ events to a Breakdown work order

PMQ event	Work Order	Remarks
incoming_event_cd	WONUM	
event_type_cd		Hardcoded to "MAINTENANCE"
source_system_cd		Hardcoded to "MAXIMO"
process_cd		
production_batch_cd		
location_cd	WOSERVICEADDRESS.SADDRESSCODE	
event_start_time	Reported date	Timestamp field
event_end_time		
event_planned_end_time		
tenant_cd		Hardcoded to "PMQ"
operator_cd		
Model	SITEID	
serial_no	ASSETNUM	
measurement_type_cd		Hardcoded to "BREAKDOWN"
observation_timestamp	Reported date	Timestamp field
value_type_cd		Hardcoded to "ACTUAL"
observation_text	DESCRIPTION_LONGDESCRIPTION	
measurement		
material_cd		
multirow_no		Hardcoded to 1

Migrating historical work orders from Maximo to PMQ

You can migrate historical work orders from Maximo to PMQ using the following process:

1. Perform a manual export of the work orders in Maximo.
2. In PMQ, import the work orders on the ESB node.
3. Work orders with a description of MAINTENANCE or BREAKDOWN are mapped with PMQ events and loaded into PMQ DataStore through a file processing flow.

Note: Loading historical work orders is a onetime activity.

Migrating real-time work orders from Maximo to PMQ

You can migrate real-time work orders from Maximo to PMQ using the following process:

1. In Maximo, a new work order is created with the description MAINTENANCE or BREAKDOWN.
2. A web service is invoked from Maximo to IBM Integration Bus (IIB).
3. When the work order is updated with the maintenance date, the web service sends the work order details to PMQ in the form of a SOAP XML message.
4. The SOAP message is mapped to PMQ events and loaded into PMQ DataStore.

Chapter 5. 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 Integration Bus transforms data into a format that can be used in IBM Predictive Maintenance and Quality.

IBM Integration Bus 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 Integration Bus, 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 14. 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 14.

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 Integration Bus 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 Integration Bus 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 Integration Bus and into the database.

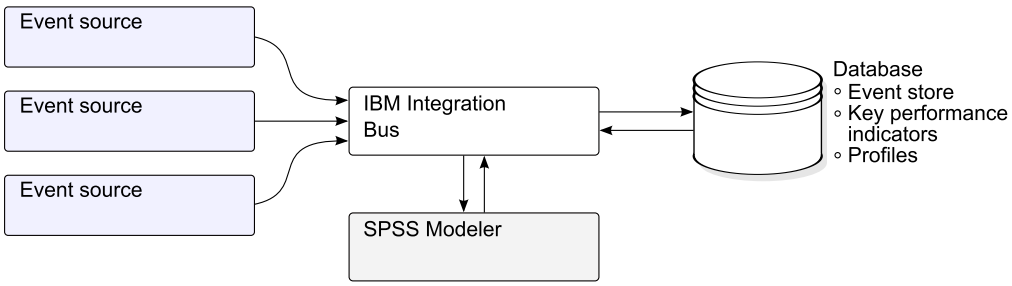


Figure 22. Flow of events into Integration Bus and into the database

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` and `event_observation` tables. An event can contain one or more event observations. Resource information is stored in the event table using Resource_cd1 and Resource_cd2.

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 IBM Integration Bus 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 observation 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.

If `Resource_cd1` has a value and `Resource_cd2` is blank or null, then this event must be associated with an Agent or Operator. If `Resource_cd1` and `Resource_cd2` both have non-blank values and have rows in the `Master_Resource` table with `Resource_type` being ASSET, then they are termed as events from a device or a resource.

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 multi-row 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 C, "The flat file API," on page 149.

Table 10. 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.

Table 10. Fields in the Events table (continued)

Field	Type	Optional or required	Event or observation	Description
location_cd	string(50)	optional	event	The location of the event.
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.
Resource_cd1	string(50)	optional	event	The operator associated with the event.
Resource_cd2	string(50)	optional	event	The model number of the device associated with the event.
Resource_cd1	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.

The event schema is stored in the project PMQEventDataLibrary.

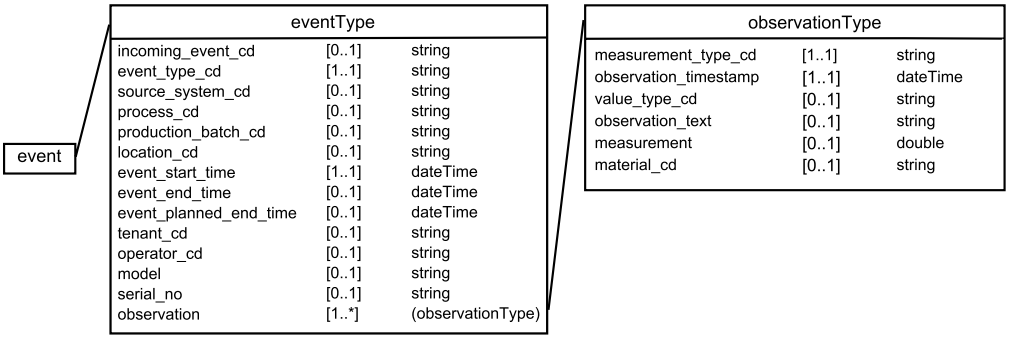


Figure 23. The event format used by IBM Predictive Maintenance and Quality

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, according to the orchestration defined for the event. 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`. 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 count

- Measurement text contains count
- Interval calculation
- Measurement above limit
- Measurement below limit
- Measurement delta

These calculations are described in "Profile calculations" on page 53.

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 166.

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" and adding a profile update for the measurement_type to the orchestration. 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.

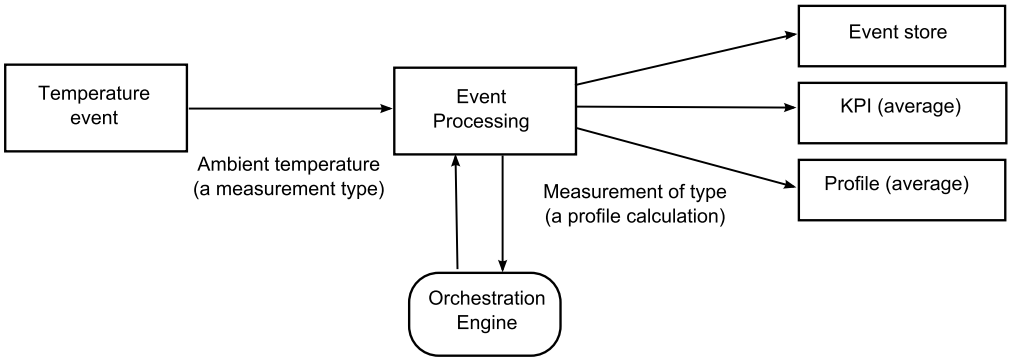


Figure 24. Temperature event workflow

Making a profile variable inactive

To make a profile variable inactive, for example, if you want to prevent a calculation from being performed, remove the profile update from the orchestration.

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 11. 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 12. 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 12. 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 pre-aggregated 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 13. 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 13. 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 the key performance indicator (KPI) and profile table (the kpi_indicator and profile_indicator values are updated). A profile variable specifies the profile calculations to perform for an observation with a given measurement type.

A profile variable maps a measurement type to a profile calculation. There may be zero or more profile variables for a given measurement type.

The following section describes the default profile calculations.

Note: Not all of the profile calculations are covered. Only the profile calculations used by BI and Analytics are covered as a part of Foundation porting.

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.

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 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.

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.

The custom calculation must be defined in the solution definition file. The custom calculation must be implemented as a Java Class implementing the com.ibm.analytics.foundation.calculation.api.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 Integration Bus.

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 and the corresponding profile updates in the profile adapter configuration of an orchestration.

To aggregate the health score, define a profile_variable and profile_adapter configuration for the Measurement of type calculation.

To aggregate the occurrences of a specific recommendation, one needs to define a profile_variable and Profile_adapter configuration for a Text contain calculation and set the comparision_string attribute of profile_variable and profile_adapter to the name of the recommendation.

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

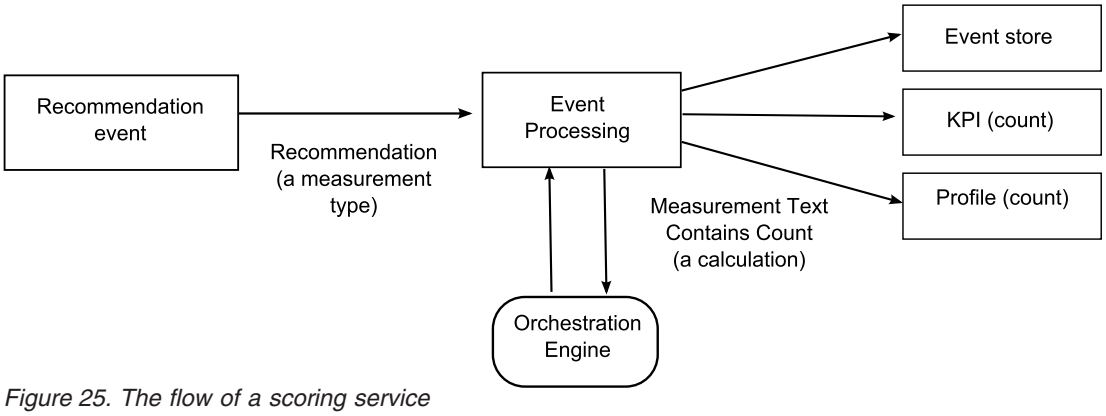


Figure 25. The flow of a scoring service

How scoring is triggered

Scoring for predictive models is triggered based on the service adapter configuration defined in the orchestration xml file. To construct any custom scoring, the orchestration xml file must be defined accordingly.

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. Updating and inserting the KPI and Profile rows.
4. Scoring using an IBM SPSS predictive model.
5. Making a recommendation using IBM Analytical Decision Management.
6. 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 the service adapter definition in the Orchestration XML. This Orchestration XML must be modified to provide customization in 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 further processed, based on the service adapter definition defined in the orchestration xml file. Events on this internal event processing queue are processed by the same flow as external events. Profile variables and profile updates in the profile adapter configuration 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 8, “Recommendations,” on page 101

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...
```

Configuring solution.xml for the event flow

The event definition, like the master data definition, is part of the solution XML file.

Within the solution.xml for event handling, there is one xml structure for a table where both event and event_observation is covered. The event_resource used in PMQ 1.0 is removed by having the resource_information defined within the event xml. Within the event definition, there is a separate tag called observation with the table_cd element.

```
<event_definition>
    <table table_cd="EVENT">
        <column column_cd="EVENT_START_TIME" type="timestamp" />
        <column column_cd="EVENT_END_TIME" type="timestamp" is_nullable="true" />
        <column column_cd="EVENT_PLANNED_END_TIME" type="timestamp" is_nullable="true" />
```

```
      <column column_cd="INCOMING_EVENT_CD" type="string" size="200" is_nullable="true" />
      <reference reference_cd="ASSET_ID" table_reference="MASTER_RESOURCE">
          <column_mapping reference_column_cd="SERIAL_NO" table_column_cd="RESOURCE_CD1"/>
          <column_mapping reference_column_cd="MODEL" table_column_cd="RESOURCE_CD2"/>
      </reference>
      <reference reference_cd="AGENT_ID" table_reference="MASTER_RESOURCE">
          <column_mapping reference_column_cd="OPERATOR_CD" table_column_cd="RESOURCE_CD1"/>
          <column_mapping reference_column_cd="OPERATOR_NA" table_column_cd="RESOURCE_CD2"/>
      </reference>
      <reference reference_cd="EVENT_TYPE_ID" table_reference="MASTER_EVENT_TYPE" />
      <reference reference_cd="SOURCE_SYSTEM_ID" table_reference="MASTER_SOURCE_SYSTEM" />
      <reference reference_cd="PROCESS_ID" table_reference="MASTER_PROCESS" />
      <reference reference_cd="PRODUCTION_BATCH_ID" table_reference="MASTER_PRODUCTION_BATCH" />
      <reference reference_cd="LOCATION_ID" table_reference="MASTER_LOCATION"/>
      <observation table_cd="EVENT_OBSERVATION">
          <column column_cd="OBSERVATION_TIMESTAMP" is_key="true" type="timestamp" />
          <column column_cd="OBSERVATION_TEXT" type="string" size="800" is_nullable="true" />
 <column column_cd="MEASUREMENT" type="double" is_nullable="true"/>
          <reference reference_cd="MEASUREMENT_TYPE_ID" is_key="true" table_reference="MASTER_MEASUREMENT_TYPE" />
          <reference reference_cd="VALUE_TYPE_ID" is_key="true" table_reference="MASTER_VALUE_TYPE" />
          <reference reference_cd="EVENT_CODE_ID" is_key="true" table_reference="MASTER_EVENT_CODE"/>
          <reference reference_cd="MATERIAL_ID" table_reference="MASTER_MATERIAL"/>
          <event_interval_column column_cd="OBSERVATION_DATE" type="date" />
          <event_interval_column column_cd="OBSERVATION_TIME" type="time" />
      </observation>
  </table>
</event_definition>
```

For handling the resource related information there are two references defined in the event xml.

```
       <reference reference_cd="ASSET_ID" table_reference="MASTER_RESOURCE">
          <column_mapping reference_column_cd="SERIAL_NO" table_column_cd="RESOURCE_CD1"/>
          <column_mapping reference_column_cd="MODEL" table_column_cd="RESOURCE_CD2"/>
      </reference>
      <reference reference_cd="AGENT_ID" table_reference="MASTER_RESOURCE">
          <column_mapping reference_column_cd="OPERATOR_CD" table_column_cd="RESOURCE_CD1"/>
          <column_mapping reference_column_cd="OPERATOR_NA" table_column_cd="RESOURCE_CD2"/>
      </reference>
```

If the referenced resource is a ASSET or AGENT.

The xml structure with in the event for handling the observation part is defined by a separate xml element called observation.

```
     <observation table_cd="EVENT_OBSERVATION">
          <column column_cd="OBSERVATION_TIMESTAMP" is_key="true" type="timestamp" />
          <column column_cd="OBSERVATION_TEXT" type="string" size="800" is_nullable="true" />
<column column_cd="MEASUREMENT" type="double" is_nullable="true"/>
          <reference reference_cd="MEASUREMENT_TYPE_ID" is_key="true" table_reference="MASTER_MEASUREMENT_TYPE" />
          <reference reference_cd="VALUE_TYPE_ID" is_key="true" table_reference="MASTER_VALUE_TYPE" />
          <reference reference_cd="EVENT_CODE_ID" is_key="true" table_reference="MASTER_EVENT_CODE"/>
          <reference reference_cd="MATERIAL_ID" table_reference="MASTER_MATERIAL"/>
          <event_interval_column column_cd="OBSERVATION_DATE" type="date" />
          <event_interval_column column_cd="OBSERVATION_TIME" type="time" />
      </observation>
```

Chapter 6. Quality early warning system use cases

The quality early warning system (QEWS) in IBM Predictive Maintenance and Quality detects emerging quality problems sooner and with fewer false alarms than is typically achieved by traditional statistical process control. To achieve earlier detection, QEWS is sensitive to subtle changes in data values, such as shifts that are small in magnitude or trends that grow slowly over time. For a given level of statistical confidence, QEWS typically needs fewer data points than traditional statistical process control.

Early detection of quality problems is essential where delayed detection can have significant negative consequences, such as in the following scenarios:

- Building a large inventory of defective products results in high scrap costs.
- Shipping a large quantity of defective products to distribution channels or customers causes high warranty expenses.
- Having widespread quality or reliability problems in the field results in damage to brand value.
- Compromised production of supply-constrained materials or components prevents on-time shipment.
- Compromised production of products with long manufacturing times results in shipment delays.

Products are the subjects of QEWS analyses. A product is typically a part or a part assembly, but it can also be a process or a material. Products might be used in larger finished assemblies, which QEWS calls *resources*.

QEWS offers two use cases. *Quality inspection* detects unfavorable changes in the quality of components. *Warranty* detects warranty issues sooner than they might otherwise be detected, which results in fewer warranty claims and lower costs.

Quality inspection

In a manufacturing environment, defects can occur in a manufacturing process because of variations in factors like process, raw materials, design, and technology. The resulting low quality of products creates a larger inventory of defective lots, which leads to increased inspection effort.

A small delay in detecting a quality problem can result in large costs, lost opportunity, and lost brand value.

The quality early warning system (QEWS) in IBM Predictive Maintenance and Quality evaluates evidence to determine whether the rate of failures is at an acceptable level. QEWS highlights combinations for which the evidence exceeds a specified threshold. QEWS can detect emerging trends earlier than traditional statistical process control, such as trend analysis. QEWS maintains a specified low rate of false alarms. Post-warning analysis of charts and tables identifies the point of origin, the nature and severity of the problem, and the current state of the process.

The QEWS quality inspection use case analyzes data from the inspection, testing, or measurement of a product or process operation over time. The data can be obtained from the following sources:

- suppliers (for example, the final manufacturing test yield of a procured assembly)
- manufacturing operations (for example, the acceptance rate for a dimensional check of a machined component)
- customers (for example, survey satisfaction ratings)

You can adjust the frequency at which data is captured and input to QEWS, and the frequency at which QEWS analyses are run, according to the requirements of each situation. For example, monitoring the quality levels of assemblies that are procured from a supplier might best be done on a weekly basis; monitoring the quality levels of units that are moving through a manufacturing operation might best be done on daily basis.

Business and technical challenges

You need the best techniques to examine quality data from tens of thousands of products and to provide proactive quality management.

You need to be able to detect process variability that is not visible through traditional means such as trend analysis. QEWS can evaluate trace data and predict with a tunable confidence whether the variability in the data is natural "noise" or a subtle indication of an impending problem. This ability is a significant improvement over traditional statistical process control.

Business challenges

Better analytical methods are available but are difficult to implement. This is due to complex computational challenges and constraints in software implementation.

Technical challenges

Manufacturing process variations can occur slowly. Gradual changes in product quality are not detected or they are detected too late, which leads to a large inventory of suspicious or defective lots. This results in increased inspection effort, more low quality products, and more waste.

Defining the quality inspection solution

To define the quality inspection solution, you must load master data, load event data, define messages flows, and define the output location of the inspection analysis.

Procedure

1. Load master data. For more information about loading master data, see Chapter 4, "Master data," on page 13.
2. Load event data. You can load event data in batch mode or in real time. For more information about loading event data, see Chapter 5, "Event data," on page 45.
3. Define message flows. For more information about message flows, see "Message flows" on page 9.

4. Define the output location of inspection analysis and reports. For more information about the quality inspection reports, see "QEWS - Inspection Chart" on page 120.

Quality inspection solution details

There are requirements that you must consider when loading the master data and event data tables.

Master data tables are loaded by the master flows. The following tables are required to implement an inspection use case:

Master_Event_Type

You must define the following event types in the Master_Event_Type table:

PRODUCTION
Defines the products that are produced by the process.

INSPECTION
Defines the sample set of products that are being inspected.

The following text is an example of a CSV file that is used to load the Master_Value_type table:

```
value_type_cd,value_type_name,language_cd,tenant_cd
ACTUAL,Actual,EN,PMQ
PLAN,Plan,EN,PMQ
FORECAST,Forecast,EN,PMQ
```

Master_Value_Type

There are three possible values for `value_type_cd` in the Master_Value_Type table: ACTUAL, PLAN, FORECAST. Usually, the data that is associated with PRODUCTION or INSPECTION events is ACTUAL.

The following text is an example of a CSV file that is used to load the Master_Value_Type table:

```
value_type_cd,value_type_name,language_cd,tenant_cd
ACTUAL,Actual,EN,PMQ
PLAN,Plan,EN,PMQ
FORECAST,Forecast,EN,PMQ
```

Master_Location

The Master_Location table contains information that is specific to the location where the event occurs, or the resource that produces the event.

The following text is an example of a CSV file that is used to load the Master_Location table:

```
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
Tokyo,Tokyo,AP,Asia Pacfic,JP,Japan,TY,Tokyo,TokyoCity, 35.41,139.45,
EN,PMQ,1
```

Master_Measurement_Type

The Master_Measurement_Type table defines how the observation is read or used. For inspection, the measurement_type is INSPECT and FAIL. The INSPECT measurement defines how many units of product were inspected or tested for quality. The FAIL measurement defines whether the outcome of the inspection is successful or not, which is identified by a flag with the FAIL.

The following text is an example of a CSV file that is used to load the Master_Measurement_Type table:

```
measurement_type_cd,measurement_type_name,unit_of_measure,
carry_forward_indicator,aggregation_type,event_code_indicator,language_cd,
tenant_cd
INSPECT,INSPECTION,,0,AVERAGE,0,EN,PMQ
FAIl,FAIL QTY INDICATOR,,0,AVERAGE,0,EN,PMQ
```

Master_Product

The Master_Product table contains the core data for the inspection use case. This table stores information that is related to a product and the product_type.

The following text is an example of a CSV file that is used to load the Master_Product table:

```
product_cd,product_name,product_type_cd,product_type_name,language_cd,tenant_cd,is_active
WT2444,Wind Turbine,Type Turbine,Type Turbine,EN,PMQ,1
Prd_No_1,Product Name 1,Type1,Type1,EN,PMQ,1
Prd_No_2,Product Name 2,Type2,Type2,EN,PMQ,1
Prd_No_3,Product Name 3,Type3,Type3,EN,PMQ,1
Prd_No_4,Product Name 4,Type4,Type4,EN,PMQ,1
Prd_No_5,Product Name 5,Type5,Type5,EN,PMQ,1
Prd_No_6,Product Name 6,Type6,Type6,EN,PMQ,1
Prd_No_7,Product Name 7,Type7,Type7,EN,PMQ,1
Prd_No_8,Product Name 8,Type8,Type8,EN,PMQ,1
Prd_No_9,Product Name 9,Type9,Type9,EN,PMQ,1
Prd_No_10,Product Name 10,Type10,Type10,EN,PMQ,1
```

Master_Product_Parameters

The Master_Product_Parameters table is specific for the inspection and warranty use cases. This table stores information about the parameters that are used in the computation of the fault detection. The parameters that are used are acceptable limits, unacceptable limits, and confidence probability.

The following text is an example of a CSV file that is used to load the Master_Product_Parameters table:

```
product_cd,product_type_cd,parameter_name,parameter_value,language_cd,tenant_cd
WT2444,Type Turbine,LAM0,0.2,EN,PMQ
WT2444,Type Turbine,LAM1,0.9,EN,PMQ
WT2444,Type Turbine,CW0,1.2,EN,PMQ
WT2444,Type Turbine,CW1,1.9,EN,PMQ
WT2444,Type Turbine,PROB0,0.29,EN,PMQ
WT2444,Type Turbine,PROBW0,0.98,EN,PMQ
WT2444,Type Turbine,INSPECT_NO_DAYS,244,EN,PMQ
```

Master_Production_Batch

The Master_Production_Batch table contains information about each production batch that is used to produce a product. The information includes the product that is produced, the date it is produced, and the batch information.

The following text is an example of a CSV file that is used to load the Master_Product table:

```
production_batch_cd,
production_batch_cd,production_batch_name,product_cd,product_type_cd,produced_date,
    language_cd,tenant_cd
T1,Turbine,WT2444,Type Turbine,2010-01-01,EN,PMQ
T2,Turbine,WT2444,Type Turbine,2011-01-01,EN,PMQ
PB 1,Production Batch 1,Prd_No_1,Type1,2011-12-08,EN,PMQ
PB 2,Production Batch 2,Prd_No_2,Type2,2011-03-18,EN,PMQ
PB 3,Production Batch 3,Prd_No_3,Type3,2012-01-04,EN,PMQ
PB 4,Production Batch 4,Prd_No_4,Type4,2012-06-06,EN,PMQ
```

```
PB 12,Production Batch 12,Prd_No_4,Type4,2012-06-06,EN,PMQ
PB 5,Production Batch 5,Prd_No_5,Type5,2012-10-26,EN,PMQ
PB 6,Production Batch 6,Prd_No_6,Type6,2013-07-07,EN,PMQ
PB 7,Production Batch 7,Prd_No_7,Type7,2011-11-28,EN,PMQ
PB 8,Production Batch 8,Prd_No_8,Type8,2011-12-19,EN,PMQ
PB 9,Production Batch 9,Prd_No_9,Type9,2012-08-17,EN,PMQ
```

Event data loading

The events for inspection can be in the form of runtime or batch data. Runtime data is raw time series data and batch data is data that is aggregated by day, month, and other units of time. The events can be stored in time series tables.

EVENT table
Contains information for master entities that is related to the event, for example, production batch, process, material, and resource.

EVENT_OBSERVATION table
Contains information that is related to the core event, for example, measurement, time of occurrence, and type of event.

Event format for batch mode loading

Batch mode uses aggregated or cumulated quantity by day or hour for a product. The cumulated quantity that was inspected, and the outcome, for example, quantity failed, are loaded through the batch mode event processing.

The main parts of the batch mode event processing are the product_code, product_type_code information, the date on which inspection is done, the quantity that was inspected, and the quantity that failed.

If information for a particular product_cd and product_type_cd repeats within a day, then the data for the entire day is accumulated and analyzed. For example, if there is hourly batch testing, then all data for the day is accumulated and analyzed.

The following text is an example of a CSV file that is used in batch mode loading:

```
product_cd,product_type_cd,inspection_date,qty_produced,inspected,failed,langauge_cd,tenant_cd
WT2444,Type Turbine,2012-11-01,295,295,23,EN,PMQ
WT2444,Type Turbine,2012-11-02,1273,1273,15,EN,PMQ
WT2444,Type Turbine,2012-11-03,1244,1244,13,EN,PMQ
WT2444,Type Turbine,2012-11-04,1313,1313,18,EN,PMQ
WT2444,Type Turbine,2012-11-05,608,608,9,EN,PMQ
WT2444,Type Turbine,2012-11-06,1148,1148,6,EN,PMQ
WT2444,Type Turbine,2012-11-07,1180,1180,16,EN,PMQ
WT2444,Type Turbine,2012-11-08,607,607,16,EN,PMQ
WT2444,Type Turbine,2012-11-09,707,707,6,EN,PMQ
WT2444,Type Turbine,2012-11-10,227,227,17,EN,PMQ
WT2444,Type Turbine,2012-11-11,1256,1256,3,EN,PMQ
WT2444,Type Turbine,2012-11-12,1325,1325,24,EN,PMQ
```

Event format for real-time loading

The classification is based on the event type and the measurement. For a PRODUCTION event type, the measurement type must be quantity (QTY). The quantity is always 1. For an INSPECTION event type, the measurement type must be either INSPECT or FAIL. The INSPECT measurement type contains “Y” in `observation_text`. The outcome of the INSPECT measurement type is indicated with a “Y” or “N” in `observation_text`. If `observation_text` is “Y”, then it is a failure case. If it is “N”, then it is a pass case. The event_type and measurement

type must be the key. Other columns that are used are production_batch_code, the location code, the event_start_time, observation_timestamp, and value_type_code. The event_start_time and observation_timestamp indicates the date and time of inspection.

Each PRODUCTION event is followed by two INSPECTION events. Each INSPECTION event has the value 1 and 2 for multirow_no. The INSPECTION events must be in sequence and are not considered as a complete event unless both are included. An INSPECT measurement type must have one more INSPECTION event with FAIL measurement type to complete the action.

The following text is an example of a CSV file that is used in real-time loading:

```
incoming_event_cd,event_type_cd,source_system,process_cd,prod_batch_cd,location_cd,
    event_start_time,event_end_time,event_planned_end_time,tenant_cd,operator_cd,model,serial_no,
    measurement_type_cd,observation_timestamp,value_type_cd,observation_text,measurement,material_code,multirow_no
1,PRODUCTION,,,T1,CA,2013-12-19T11:05:00,,,PMQ,,,,QTY,2013-12-19T11:05:00,ACTUAL,,,,1
2,INSPECTION,,,T1,CA,2013-12-19T11:05:00,,,PMQ,,,,INSPECT,2013-12-19T11:05:00,ACTUAL,Y,,,1
3,INSPECTION,,,T1,CA,2013-12-19T11:05:00,,,PMQ,,,,FAIL,2013-12-19T11:05:00,ACTUAL,Y,,,2
4,PRODUCTION,,,T1,CA,2013-12-19T11:07:00,,,PMQ,,,,QTY,2013-12-19T11:07:00,ACTUAL,,,,1
5,INSPECTION,,,T1,CA,2013-12-19T11:07:00,,,PMQ,,,,INSPECT,2013-12-19T11:07:00,ACTUAL,Y,,,1
6,INSPECTION,,,T1,CA,2013-12-19T11:07:00,,,PMQ,,,,FAIL,2013-12-19T11:07:00,ACTUAL,N,,,2
7,PRODUCTION,,,T1,CA,2013-12-19T11:09:00,,,PMQ,,,,QTY,2013-12-19T11:09:00,ACTUAL,,,,1
8,INSPECTION,,,T1,CA,2013-12-19T11:09:00,,,PMQ,,,,INSPECT,2013-12-19T11:09:00,ACTUAL,Y,,,1
9,INSPECTION,,,T1,CA,2013-12-19T11:09:00,,,PMQ,,,,FAIL,2013-12-19T11:09:00,ACTUAL,Y,,,2
```

Inspection message flow and triggering mechanism

QEWS invocation is done only for batch or aggregated data. Real-time data is aggregated by the IBM Integration Bus flow on a day level and stored in the KPI table of the data model. After the batch processing starts, IBM Predictive Maintenance and Quality performs the next level computation by generating the analysis charts and outputs.

Predictive Maintenance and Quality has the following types of flows:

Real-time event flow

The File Input node performs the following steps:

1. Reads inspection real-time data.
2. Converts it into MQ messages.
3. Aggregates at day level per product.
4. Performs CUSUM, CUW, and failure rate calculations as part of the intraday calculations.

The event .csv file is picked from the file input node. Each record is converted into a standard event format and pushed into the event input queue. The event processor code reads the messages and performs aggregation at the day level. The event processor code inserts the aggregation into the KPI table, where it is available for the next level of processing by the QEWS algorithm.

Batch flow

The batch data is in aggregated form. The QEWS algorithm computes the failure rate and threshold values to identify the progress of the process flow.

The event .csv file is picked from the file input node. The QEWS invocation algorithm performs the fault detection analysis. The fault detection analysis generates the charts and outputs to report the results.

The timer notification initiates the QEWS call at scheduled intervals. The QEWS call checks for records in the product_kpi table whose batch flag is not set. Those records are input into the QEWS invocation algorithm.

Timer-based flow

A timer-based flow is similar to a batch flow except that it contains a timer notification trigger to start the batch mode processing. The batch mode processing reads the historic data records in the KPI table and sets the batch_flag to Y. The inspection flow starts the QEWS algorithm and updates the KPI and profile tables for the records whose batch_flag is set to Y.

Output and reporting

The output from the inspection flow is put on the NFS share location at `/var/mqsi/shared-classes/loc.properties`. The location of the inspection analysis output is stored in the `Location` variable in the `loc.properties` file.

The Location value is the base path of the folder for each product_id. The folder is named by a combination of Product_cd and Product_type_cd and it contains the list of output files of the QEWS analysis.

IBM Cognos Business Intelligence loads the images from the NFS shared location into report.

Results and benefits

The quality early warning system (QEWS) in IBM Predictive Maintenance and Quality reduces cost by detecting problems and issues earlier and more accurately.

Results

Predictive Maintenance and Quality QEWS provides the following results:

- Improves production yields in the manufacturing line.
- Helps you to gain better understanding of root causes of manufacturing issues.
- Provides faster detection of manufacturing quality problems.

Benefits

Subtle changes in failure rates that indicate potential emerging quality problems are detected earlier. Early detection means quicker problem identification, faster problem resolution, and lower total costs.

The definitive nature of QEWS alerts eliminates the need for subjective judgment of statistical process control charts and other traditional tools, providing you with consistent and accurate direction.

QEWS can deliver insightful early warning signals even under variable lot size scenarios.

Warranty

Various conditions can lead to accelerated wear and replacement of manufactured products that are under warranty. Such conditions might include variations in the manufacturing process of the product, variations in the quality of vendors' materials that are used in the product, or the ways in which the product is used.

A small delay in detecting the conditions that lead to accelerated wear can cause more warranty claims and related losses. By understanding the factors that lead to warranty claims, you can take corrective actions such as the following actions:

- Improve manufacturing processes to prevent warranty claims.
- Set pricing for warranties and extended warranties.
- Evaluate vendors of the materials that are used in the product.

The quality early warning system (QEWS) warranty use case in IBM Predictive Maintenance and Quality provides detection that is based on excessive replacement rate and evidence of wear-out.

Replacement rate

QEWS alerts you when the product's random failure rate exceeds a computed threshold. The threshold can reflect product reliability goals (for example, the product population in the field must not exceed a specified failure rate) or financial liability goals (for example, the cost of reimbursing product warranty claims must not exceed a specified total amount).

Wear-out

QEWS alerts you when it finds evidence that product failures are not random in time, but are indicative of wear-out. Wear-out means that products that are in customer use for a longer time fail more often than products that are in customer use for a shorter time. Because wear-out can have serious consequences, QEWS alerts you when it detects evidence of wear-out regardless of how many product units contributed to the detection.

QEWS enables warranty models that are based on sales, production, and manufacturing dates.

Sales model

The Sales model identifies variations in product wear and replacement rates according to the date of sale. The date of sale might correlate with in-service conditions, seasonal climatic conditions, a particular customer, or other important similarities.

For example, a product carries a one-year warranty. In cold conditions, the product becomes brittle and wears prematurely. In certain geographies, products that are sold and enter service in winter initially suffer rapid wear, followed by slower wear during the latter part of the warranty period. The opposite is true for products that are sold and enter service in summer. These seasonal variations affect the product wear rates and weighted replacement rates, which are detected early by QEWS.

Production model

The Production model identifies variations in product wear and replacement rates according to the production date of the product, not the resource in which the product is used. The production date of the product might correlate with the manufacturing equipment operator, the manufacturing process, or other important similarities.

For example, a faulty batch of products is produced during a particular period. The products are installed in resources that have different manufacturing dates. Although the resource manufacturing dates and the product production dates are unrelated, QEWS makes it easier to identify and understand the real cause of the warranty claims.

Manufacturing model

The Manufacturing model identifies variations in product wear and replacement rates according to the manufacturing date of the resource in which the product is used. The resource manufacturing date might correlate with assembly problems that occurred during a particular period.

For example, due to a short-term problem with the manufacturing process of a resource, some of the products that are used in the resource fail prematurely. Although the resource manufacturing dates and the product production dates are unrelated, QEWS makes it easier to identify and understand the real cause of the warranty claims.

You can adjust the frequency at which data is captured and input to QEWS, and the frequency at which QEWS analyses are run, according to the requirements of each situation. For example, monitoring data from a network of field service personnel might best be done on a monthly basis.

Business and technical challenges

Fast product cycles, high product volumes, and increasing cost pressure can all lead to increasing numbers of released defective products. The quality early warning system uses IBM technology to detect warranty claim trends earlier than they would otherwise be detected so that you can intervene with corrective action.

Business challenges

Statistical process control methods often overlook cumulative evidence that indicates a worsening quality problem. Better analytical methods are often difficult to implement due to complex computational challenges and constraints in software implementation.

Technical challenges

Premature product wear can have non-obvious causes such as source material variations, seasonal climatic conditions, or temporary manufacturing problems either with the product or with the resource in which the product is used. A small delay in detecting the conditions that lead to accelerated wear can cause more warranty claims and related losses.

Defining the warranty solution

To define the warranty solution, you must load master data, load event data, define message flows, and define the output location of the warranty analysis.

Procedure

1. Load master data. For more information about loading master data, see Chapter 4, “Master data,” on page 13.
2. Load event data. You can load event data in batch mode or in real time. For more information about loading event data, see Chapter 5, “Event data,” on page 45.
3. Define message flows. For more information about message flows, see “Message flows” on page 9.
4. Define the output location of warranty analysis and reports. For more information about the warranty reports, see “QEWSL - Warranty Chart” on page 121.

Warranty solution details

There are requirements that you must consider when loading the master data and event data tables.

Master data tables are loaded by the master flows. The following tables are required to implement a warranty use case:

Master_Location

The Master_Location table contains information that is specific to the geography of the location where the event is produced, or the resource that produces the events.

The following text is an example of a CSV file that is used to load the Master_Location table:

```
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
Tokyo,Tokyo,AP,Asia Pacfic,JP,Japan,TY,Tokyo,TokyoCity,35.41,139.45,
EN,PMQ,1
```

Master_Resource_Type

The Master_Resource_Type table maintains the Resource type classification. It supports two classification types: ASSET and AGENT. ASSET is a machine or machine part that is used in production. AGENT is the one who operates the machine or the system to ensure that the production procedure is carried out properly.

The following text is an example of a CSV file that is used to load the Master_Resource_Type table:

```
resource_type_cd,resource_type_name,language_cd,tenant_cd
ASSET,Asset,EN,PMQ
AGENT,Agent,EN,PMQ
```

Master_Resource

The Master_Resource table maintains all of the details pertaining to a Resource (ASSET or AGENT). The table maintains information such as which organization hierarchy the resource is lined to, the location at which the resource is installed, the tenant to which the resource is attached or leased, production rate, maintenance interval, and the manufactured date of the resource.

The following text is an example of a CSV file that is used to load the Master_Resource table:

```
resource_cd1,resource_cd2,resource_name,resource_type_cd,
resource_sub_type,parent_resource_cd1,parent_resource_cd2,
standard_production_rate,production_rate_uom,
preventive_maintenance_interval,group_type_cd_1,
group_member_cd_1,group_type_cd_2,group_member_cd_2,
group_type_cd_3,group_member_cd_3,group_type_cd_4,
group_member_cd_4,group_type_cd_5,group_member_cd_5,
location_cd,mfg_date,language_cd,tenant_cd,Is_active
-NA-,-NA-,Not Applicable,ASSET,,,,,,,-NA-,-NA-,-NA-,-NA-,-NA-,
-NA-,-NA-,-NA-,-NA-,-NA-,TK,2014-01-01,EN,PMQ,1
RCD1,MOD1,RCMOD1,ASSET,,,,,,,,,,,,,,,,TK,,,,1
RCD2,MOD2,RCMOD2,ASSET,,,,,,,-NA-,-NA-,-NA-,-NA-,-NA-,-NA-,
-NA-,-NA-,-NA-,-NA-,-NA-,TK,,,,1
RCD3,MOD3,RCMOD3,ASSET,,,,,,,-NA-,-NA-,-NA-,-NA-,-NA-,-NA-,
-NA-,-NA-,-NA-,-NA-,-NA-,TK,,,,1
```

Master_Product

The Master_Product table contains the core data for the inspection and warranty use cases. This table stores information that is related to a product and the product_type.

The following text is an example of a CSV file that is used to load the Master_Product table:

```
product_cd,product_name,product_type_cd,product_type_name,
language_cd,tenant_cd,Is_active
AAA,TRUNK,B005,Body,EN,PMQ,1
AAB,TRUNK,B005,Body,EN,PMQ,
AAC,TRUNK,B006,Body,EN,PMQ,
AAD,TRUNK,B006,Body,EN,,
AAE,TRUNK,B006,Body,,,
```

Master_Production_Batch

The Master_Production_Batch table contains information about each production batch that is used to produce a product. The information includes the product that is produced, the date it is produced, and the batch information.

The following text is an example of a CSV file that is used to load the Master_Production_Batch table:

```
production_batch_cd,production_batch_name,product_cd,
product_type_cd,produced_date,language_cd,tenant_cd
B1001,FrameBatch,AAA,B005,2012-03-01,EN,PMQ
B1002,FrameBatch,AAB,B005,2012-03-01,EN,PMQ
B1003,FrameBatch,AAC,B006,2012-03-01,EN,PMQ
B1004,FrameBatch,AAA,B006,,,
```

Master_Product_Parameters

The Master_Product_Parameters table is specifically for the inspection and warranty use cases. This table stores information about the parameters that are used in computation of the fault detection. The parameters that are used are acceptable limits, unacceptable limits, and the confidence probability value.

The following text is an example of a CSV file that is used to load the Master_Product_Parameters table:

```
product_cd,product_type_cd,parameter_name,parameter_value,
language_cd,tenant_cd
XYY672B-F,Engine,CW0,0.00007,EN,PMQ
XYY672B-F,Engine,CW1,0.00023,EN,PMQ
XYY672B-F,Engine,PROBW0,0.99,EN,PMQ
XYY672B-D,Gearbox,CW0,0.00007,EN,PMQ
XYY672B-D,Gearbox,CW1,0.00023,EN,PMQ
XYY672B-D,Gearbox,PROBW0,0.99,EN,PMQ
```

Master_Resource_Production_Batch

The Master_Resource_Production_Batch table contains information about each production batch that is used to produce a resource.

The following text is an example of a CSV file that is used to load the Master_Resource_Production_Batch table:

```
resource_cd1,resource_cd2,production_batch_cd,qty,language_cd
RCD1,MOD1,B005,3,EN
RCD2,MOD2,B006,3,EN
RCD3,MOD3,B005,3,EN
```

Tip:

- If a product can have different parameters (such as LAM0, LAM1, PROB0, CW0, CW1, PROBW0), then you can assign a separate product code and production

batch to each product variation. Reference each production batch in the Master_Resource_Production_Batch table.

- If a product has the same parameters but different manufacturing or production dates, then you can assign a separate production batch for each manufacturing or production date. Reference each production batch in the Master_Resource_Production_Batch table.

Master data in the Sales model

The following statements apply to the Sales model:

- When a resource is sold, the warranty is tracked from the date of sale until the end of the warranty period. Resources are tracked because, unlike products, in IBM Predictive Maintenance and Quality resources are serialized and can form a hierarchy.
- Each resource contains a number of products. Each product is tracked by a Master_Production_Batch table record.
- The Master_Resource_Production_Batch table handles the mapping between the Master_Resource and Master_Production_Batch tables and also maintains the quantity of products that go into a resource.

Master data in the Production model

The following statements apply to the Production model:

- The warranty for a product extends from the date of production until the end of the warranty period.
- Products are tracked by produced_date.
- The product produced_date is stored in the Master_Production_Batch table and is used as the vintage date.

Master data in the Manufacturing model

The following statements apply to the Manufacturing model:

- The warranty for a resource extends from the date of manufacture until the end of the warranty period.
- Resources are tracked by mfg_date.
- The mfg_date is stored in the Master_Resource table.

Event data loading

After the Master data load flows are completed, you must load the event flows. Event data is loaded on an event basis, where each event is associated with a number of observations. Each observation indicates a measurement type (for example, pressure in kilopascals) and a measurement reading.

The event flows load events such as SALES and WARRANTY that are predefined in the Master_Event_Type table. Every event is related to a particular Resource and the Production_Batch details.

The following text is an example of a CSV file that is used for loading:

```
incoming_event_cd,event_type_cd,source_system,process_cd,
prod_batch_cd,location_cd,event_start_time,event_end_time,
event_planned_end_time,tenant_cd,operator_cd,resource_cd2,
resource_cd1,measurement_type_cd,observation_timestamp,
value_type_cd,observation_text,measurement,material_code,multirow_no
```

```
1,SALES,,,B1001,Tokyo,2006-12-19T12:00:00,,,PMQ,,MOD1,RCD1,
SALESDATE,2006-12-19T12:00:00,ACTUAL,12/19/2009,35.9344262295082,,1
1,WARRANTY,,,B1001,Tokyo,2013-06-17T12:00:00,,,PMQ,,MOD1,RCD1,
WARRANTYINDICATOR,2013-06-17T12:00:00,ACTUAL,N,,,1
1,SALES,,,B1002,Tokyo,2006-11-20T12:00:00,,,PMQ,,MOD2,RCD2,
SALESDATE,2006-11-20T12:00:00,ACTUAL,11/20/2009,35.9344262295082,,1
1,WARRANTY,,,B1002,Tokyo,2009-05-04T12:00:00,,,PMQ,,MOD2,RCD2,
WARRANTYINDICATOR,2009-05-04T12:00:00,ACTUAL,Y,,,1
1,SALES,,,B1003,Tokyo,2006-10-31T12:00:00,,,PMQ,,MOD3,RCD3,
SALESDATE,2006-10-31T12:00:00,ACTUAL,10/31/2009,35.9344262295082,,1
```

Event data loading in the Sales model

The Sales model event data is loaded in the following order:

1. The SALES event is loaded.
   - The measurement_type_cd field contains SALESDATE.
   - The event_start_time field and the observation_timestamp field contain the date of sale.
   - The observation_text field contains the warranty end date. By default, the value is three years but it can be changed as required.
   - The measurement field contains the number of warranty months.
2. Any number of WARRANTY events are loaded.
   - The measurement_type_cd field contains WARRANTYINDICATOR.
   - The event_start_time field and the observation_timestamp field contain the date when the claim was made.
   - The observation_text field and the measurement field are blank.

Event data loading in the Production model

The Production model event data is loaded in the following order:

1. The SALES event is loaded.
   - The measurement_type_cd field contains SALESDATE.
   - The event_start_time field and the observation_timestamp field contain the Produced Date from the Master_Production_Batch table.
   - The observation_text field contains the warranty end date. By default, the value is 3 years but it can be changed as required.
   - The measurement field contains the number of warranty months.
2. Any number of WARRANTY events are loaded.
   - The measurement_type_cd field contains WARRANTYINDICATOR.
   - The event_start_time field and the observation_timestamp field contain the date when the claim was made.
   - The observation_text field and the measurement field are blank.

Event data loading in the Manufacturing model

The Manufacturing model event data is loaded in the following order:

1. The SALES event is loaded.
   - The measurement_type_cd field contains SALESDATE.
   - The event_start_time field and the observation_timestamp field contain mfg_date from the Master_Resource table.
   - The observation_text field contains the warranty end date. By default, the value is 3 years but it can be changed as required.

- The measurement field contains the number of warranty months.

2. Any number of WARRANTY events are loaded.
   - The measurement_type_cd field contains WARRANTYINDICATOR.
   - The event_start_time field and the observation_timestamp field contain the date when the claim was made.
   - The observation_text field and the measurement field are blank.

SPSS Modeler Streams

Data in the event and event observation tables must be processed so that it can be provided to QEWS. Processing the tables involves calling the SPSS modeler stream, which picks data from Event, Event_Observation, Resource, Product, and Production_Batch and prepares the data in the following format:

```
Product_code | Produced Date | Service_Month | Parts under Warranty | Parts replaced | tenant_cd
```

A Service table holds these records and forms as input for QEWS.

There are two SPSS Modeler streams and corresponding Collaboration & Deployment Services (C&DS) jobs for Warranty. The first stream is for the Manufacturing and Production models, in which the specific use case can be controlled by toggling across a parameter from MFG (Manufacturing) to PROD (Production). The second stream is for the Sales model.

The streams differ in the transformation logic for producing the Service table (for more information, see "Service tables" on page 75). The SPSS Modeling layer provides special logic for each of the models; all other processing and treatment is the same for all of the models.

The main difference between the models is in the aggregation and tracking of vintages. A vintage is a combination of the product ID (numbered product type) and a date (date of sale, date of production, or date of manufacture). The date at which the product was put in service is assumed to be the same as the date of sale of the resource in which the product is used. The models do take into account the differential tracking and treatment of products that are sold or shipped as replacements of other products that were shipped separately. Replacement products can either be excluded from the event structure or they can be included as a separate vintage.

You can choose between the Production and Manufacturing models by changing the IsMFG_OR_PROD variable of the IBM_QEWSL_JOB C&DS job to PROD or MFG. You can change the variable from either SPSS C&DS (during ad-hoc one time trigger) or IIB (during automated triggers).

The Sales model is controlled by a separate job named IBMPMQ_QEWSL_SALES_JOB. The job can be run from IIB by using its job URI.

Customizable Parameters and Special Scenarios

Both SPSS Modeler streams contain some common parameters that can be used while running the SPSS models under special scenarios and requirements. These options can be altered from the SPSS C&DS Job Variable or from IIB. The preferred way of altering these parameters is through IIB. The description and uses of these parameters is as follows:

IsRunDateEqServerDate
This parameter determines whether the SPSS server system date (value = 1) or a custom run date (value = 0) is used in computation logic requiring a run date. The default value is 0 and it uses the custom run date that is supplied by IIB (corresponding to the IIB server system date during default runs).

RunDateInFormatYYYYMMDDHyphenSeparated
This parameter is used only if the value of IsRunDateEqServerDate parameter is 0. The parameter sets the custom run date. The required date format is YYYY-MM-DD.

ServiceTablQtyMultiplier
For performance reasons, sometimes you might need to run the QEWSL warranty engine on a sample of the complete data. QEWSL is a weighted algorithm, so by default it does not produce the same graphs or alerts for a sample as it would for the complete data. If the sample is a good true representative sample, this parameter helps to correct the scale of the weighted results or plots to give a representative output. The parameter is set with a value of multiplier as 1/*number*.

Service tables

When the SPSS stream runs, it populates a DB2® table named SYSREC.SERVICE (referred to as the Service table). After the table is populated, the processing is the same for all models.

The structure of the Service table is the same for all models. What changes is the computation and aggregation logic for the table fields by the different SPSS streams and models.

The Service table contains the following fields:

PRODUCED_DATE
This field contains the vintage date of the Sales or Manufacturing model. Together with the PRODUCT_ID field, this field represents the vintage of the record. Together with PRODUCT_ID and SVC_MNTHS fields, this field represents the composite unique key for the table.

PRODUCT_ID
This field represents the non-serialized product identifier (numeric product type) for the product whose replacement needs to be tracked.

SVC_MNTHS

This field represents the number of months that any of the products of that vintage (PRODUCED_DATE + PRODUCT_ID) were in service during their warranty period. For example, a three-year warranty period can contain up to 36 service months.

To have a consistent number of maximum service months across vintages in a computation lot, products with shorter warranty periods (for example, two years) can be given more SVC_MNTHS to match products with longer warranty periods (for example, 36 months). In this case, during the SVC_MNTHS that are outside of the warranty period, WPARTS and WREPL are both 0.

WPARTS
This field represents the number of products of that vintage

(PRODUCED_DATE + PRODUCT_ID) that were in service without any warranty claims during the service month (SVC_MNTHS).

WREPL
This field represents the number of products of that vintage (PRODUCED_DATE + PRODUCT_ID) that failed (received a warranty claim) during the service month (SVC_MNTHS).

TENANT_ID
This field is an identifier to differentiate between tenant data in a multi-tenant environment.

Warranty message flow and triggering mechanism

When the SPSS modeler stream runs successfully, it invokes the warranty flow. A status message that is embedded with a date value is placed in the PMQ.QEWS.WARRANTY.IN queue. When the broker interface detects a message in the queue, it triggers the QEWSL algorithm. The embedded date value in the message is the rundate, which becomes the date of reference for the warranty flow. The Service table records and the parameters are passed to the QEWSL algorithm.

The same message flow is used to trigger all of the warranty models.

Output and reporting

The warranty flow output is saved to the NFS share location that is specified in the `/var/mqsi/shared-classes/loc.properties` file. In the file, the Location1 variable specifies the location where the warranty analysis output is saved.

In the location that is specified by the Location1 variable, a folder is created and named by the rundate value in the format YYYY_MM_DD. Within the folder, for each product_id (a combination of Product_cd and Product_type_cd), a folder is created that contains the list of output files from the QEWSL analysis.

IBM Cognos Business Intelligence loads the images from the NFS shared location into the report.

Results and benefits

The quality early warning system (QEWS) warranty use case in IBM Predictive Maintenance and Quality reduces cost by detecting problems and issues earlier than they would otherwise be detected, and more accurately.

Results

IBM Predictive Maintenance and Quality QEWS delivers the following results:

- Shows you where to improve manufacturing processes to prevent warranty claims.
- Helps you to set pricing for warranties and extended warranties.
- Helps you to evaluate vendors of the materials that are used in products.

Benefits

Subtle changes in warranty claim rates indicative of potential emerging quality problems are detected earlier. This allows for quicker problem identification, faster problem resolution, and lower total costs.

The definitive nature of QEWS alerts eliminates the need for subjective judgment of statistical process control charts and other traditional tools, providing you with consistent and accurate direction.

QEWS can deliver insightful early warning signals even under variable lot size scenarios.

Chapter 7. Predictive models

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

This section describes the steps that are needed to build predictive models in the predictive maintenance area by using IBM Predictive Maintenance and Quality (PMQ). It also covers some sample use cases in the manufacturing field. Later, it highlights the steps involved, starting from the business/data understanding to deploying the predictive models built for a given use case.

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

- The Maintenance predictive model
- The Sensor Health predictive model
- The Top Failure Reason predictive model
- The Integrated Health predictive model

Sample predictive models are provided. For more information, see "IBM SPSS artifacts" on page 191.

The training and scoring process

The steps for training and scoring the predictive models are as follows:

1. The modeling node estimates the model by studying records for which the outcome is known and creates a model nugget. This is referred to as training the model.
2. The model nugget can be added to any stream with the expected fields to score records. By scoring the records for which you already know the outcome (such as existing customers), you can evaluate how well it performs.
3. After you are satisfied that the model performs acceptably well, you can score new data (such as health score of an asset or life time of an asset) to predict how they will perform.

Optimized recommended actions

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) that you use, the recommendation may be acted on automatically. You can also predict the success rate of the corrective action that is based on previous actions.

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.

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 the template `OptimizedAssetMaintenance.xml` to prioritize your business objective that is based on, for example, profit maximization, or downtime minimization.

The Maintenance predictive model

The Maintenance predictive model analyzes helps you optimize your Preventive Maintenance System.

In the past, a scheduler would optimize a plant's Preventive Maintenance System (PMS) by carefully altering the days that were allotted for maintenance in the OEM's default schedule. The IBM Predictive Maintenance and Quality Maintenance predictive model helps you to optimize your maintenance schedule using predictive analysis.

Often, in a new setup of PMQ/ sensors in the plant, even if sensor data has not gained optimal maturity for effective predictions, there may be enough data in the plant's maintenance system (Maximo/ SAP-PM etc.) to initiate a Predictive Maintenance regime. IBM PMQ's Maintenance Analytics can work on such maintenance work orders alone and does not depend on any sensor data. Therefore, the Maintenance model may help to expedite the ROI of any Predictive Analytics system before any useful sensor data is obtained.

For some resources or instances, sensor analytics alone might not provide the most accurate predictions. In this case, you can combine the results of both Maintenance Analytics and Sensor Analytics (via Integration Analytics module) to produce more optimal end results.

Data understanding

The performance indicator table `RESOURCE_KPI` contains aggregated values for each day. You can use it to prepare for model training and scoring.

The following figure shows the counts of various profiles in the dataset for a given resource and their percentages of the dataset.

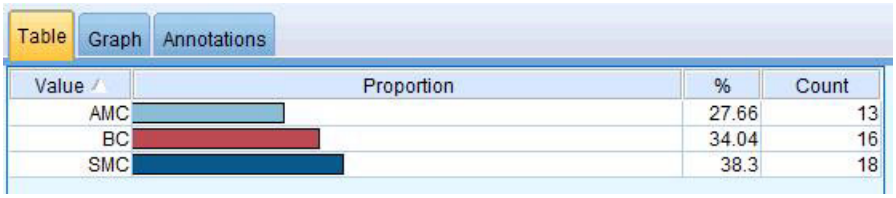
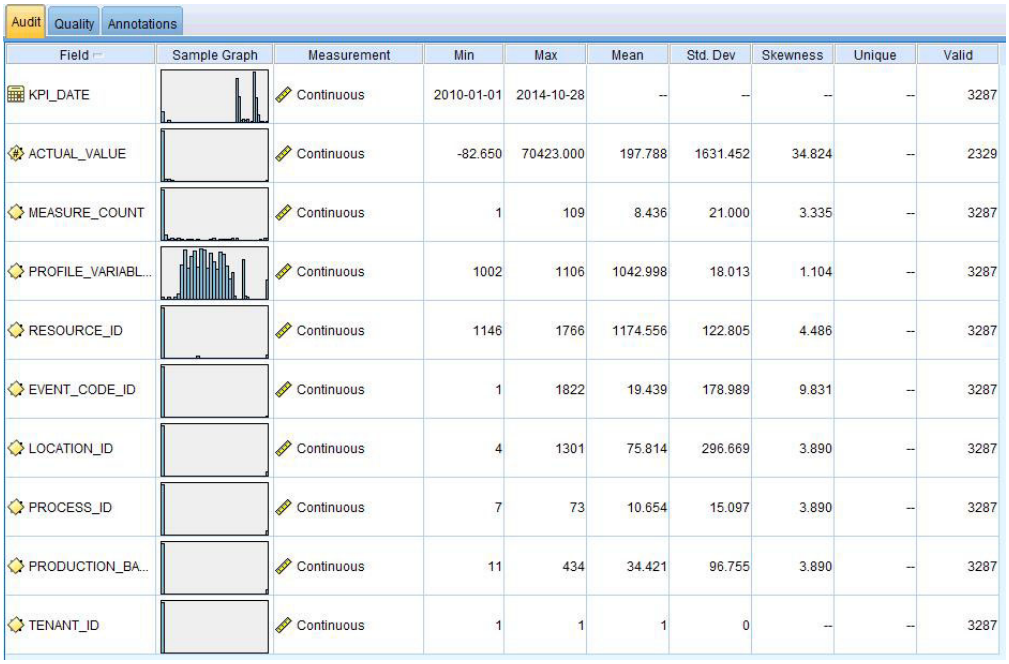


Figure 26. Percentage and counts of each profile

Additionally, the `MASTER_PROFILE_VARIABLE` and the `MASTER_MEASUREMENT_TYPE` tables help to define the appropriate codes, names, and other generic or static data.

The following figure shows a data audit node.



Field	Sample Graph	Measurement	Min	Max	Mean	Std. Dev	Skewness	Unique	Valid
KPI_DATE		Continuous	2010-01-01	2014-10-28	--	--	--	--	3287
ACTUAL_VALUE		Continuous	-82.650	70423.000	197.788	1631.452	34.824	--	2329
MEASURE_COUNT		Continuous	1	109	8.436	21.000	3.335	--	3287
PROFILE_VARIABL...		Continuous	1002	1106	1042.998	18.013	1.104	--	3287
RESOURCE_ID		Continuous	1146	1766	1174.556	122.805	4.486	--	3287
EVENT_CODE_ID		Continuous	1	1822	19.439	178.989	9.831	--	3287
LOCATION_ID		Continuous	4	1301	75.814	296.669	3.890	--	3287
PROCESS_ID		Continuous	7	73	10.654	15.097	3.890	--	3287
PRODUCTION_BA...		Continuous	11	434	34.421	96.755	3.890	--	3287
TENANT_ID		Continuous	1	1	1	0	--	--	3287

Figure 27. Data audit node

The data audit node provides summary statistics, histograms, and distribution graphs that can help you to better understand the data. The report also displays the storage icon (data type) before the field name.

Pre-modeling the data

All pre-modeling required by Maintenance Analytics is done during the `MAINTENANCE.str` modeling stream.

For information about both the modeling and pre-modeling data preparation, see "Modeling the data."

Modeling the data

Modeling for the Maintenance model occurs during the MAINTENANCE.str stream.

See the following table for information about MAINTENANCE.str.

Table 14. The MAINTENANCE.str stream

Stream name	Purpose	Input	Target	Output
MAINTENANCE.str	Predicts forecasted days to maintenance interval of equipment based on Maximo work orders, and then converts these predictions into continuous health scores.	The Maximo (or other Plant Maintenance Systems') work orders converted into profiles for the actual, planned, and scheduled maintenance dates for Breakdown and Planned Maintenance.	1. Custom Target as obtained using pre-data preparation within the stream itself. 2. IsFail	1. Forecasted days until the next maintenance for each resource and each historic and current day 2. Health score of the equipment for each day

There are some limitations that affect the Maintenance model:

- Limitations exist in the Breakdown + Planned maintenance work orders that are extracted from Maximo. As a result, these work orders are not optimal for forecasting directly. Breakdown + Planned maintenance work orders represent intermittent events, for which the default SPSS Time Series Modeling node cannot be used directly.
- Both types of the maintenance series contain censured data at either limit (left and right respectively). For example, for the Breakdown series, we cannot identify from the given work orders what the optimal maintenance day would be to prevent a breakdown or irreversible wear. Similarly, for the Planned Maintenance work orders, we cannot identify the day on which a breakdown or irreversible wear would occur if we choose not to perform machine maintenance on the day identified by the Breakdown work orders.
- The series that we want to predict, that is, an ideal maintenance period, either does not exist or is divided into two series of planned and unplanned maintenance. Direct application of time series models, even with transfer function or multi-variate ARIMA models, might not help to solve the problem.

To overcome these limitations, IBM PMQ uses a custom application of Croston's forecasting methods for intermittent demand (patent pending). Using this method, the two work orders' dates series are converted to the difference of days and then combined into a single series (using censoring adjustments). This single series can be subsequently forecasted using available time series nodes in SPSS. In the current application, a simple method of global user-defined multiplicative factors is used. However, other more sophisticated, optimal, and customized methods can also be used.

The resulting value of the number of days until the next forecast can then be used to predict the machine's failure. Health scores can then be obtained using the raw propensity scores or the adj raw propensity/ or the confidence of the obtained predictions. These Health scores can be used directly or with standardization at each resource level. The present implementation uses standardization to get a uniform scale/ level of health scores for each resource.

Post modeling data manipulation

Post modeling for the Maintenance model occurs during the MAINTENANCE_DAILY.str and MAINTENANCE_EVENTS.str streams.

See the following table for more information.

Table 15. The MAINTENANCE_DAILY.str and MAINTENANCE_EVENTS.str streams

Stream name	Purpose	Input	Output
`MAINTENANCE_DAILY.str`	This is a post modeling data preparation stream for the purpose of preparing data for BI plots. This stream converts the predictions from the `MAINTENANCE TRENDS` table into a format required by the Maintenance Overview Dashboard. The results are entered into the `Maintenance Daily` table in the DB.	The input data source contains all the records present in the Maintenance Trends table in the DB across all the days.	Only the current day's data with some transformations into the `Maintenance Daily` Table

Table 15. The MAINTENANCE_DAILY.str and MAINTENANCE_EVENTS.str streams (continued)

Stream name	Purpose	Input	Output
MAINTENANCE_EVENTS.str	This is a post modeling data preparation stream for the purpose of preparing data for BI plots. This stream converts the data from the MAINTENANCE DAILY table into a format required by the by IIB flows. The results are used to populate the IBM PMQ Events in the Event Observation table in the DB.	The input data source contains all the records present in the Maintenance Daily table in the DB.	A csv file (uploaded on the IIB integration in folder on the Analytics server) with the Maintenance Daily data in a format that can be used by IIB flows to populate the Events Table.

To improve the performance at the BI end and ensure a fast refresh and optimal user experience, all static computations and data manipulations (calculations and data manipulations not affected by user selection of prompts/ filters on the dashboards) were transferred to the SPSS batch jobs. These batch jobs can be run at an off-peak hour.

The later part of Maintenance.strand Maintenance_daily.str run the batch jobs and prepare the Maintenance Trends and Maintenance Daily tables.

The data form maintenance daily can be transferred back as events in a IBM PMQ acceptable event format. External applications can then access the events. Dashboards can also consume structure events efficiently, as the Overview dashboard does currently. The Maintenance_Events.str stream helps achieve this objective.

Evaluation of the model

An example application used the Maintenance predictive model very effectively.

The following figure shows a time series plot with both predicted values and actual values. In this case, the predictions were accurate.

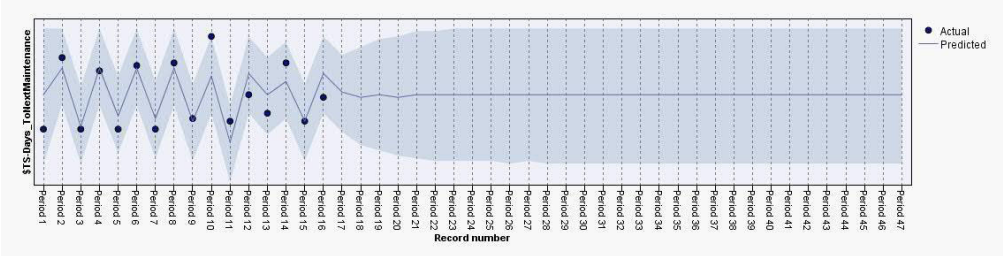


Figure 28. Time series plot

The Analysis node in the output tab helps with the evaluation of a particular model output. In this example, the predicted IsFAIL is compared on with the existing/actual values and arrived at a best-fitting training model. See the following table.

Table 16. Comparing $L-IsFAIL with IsFAIL

Category	Value
Minimum Error	0.0
Maximum Error	1.0
Mean Error	0.032
Mean Absolute Error	0.032
Standard Deviation	0.177
Linear Correlation	
Occurrences	495

Deployment of the model

The Maintenance predictive model uses parameters from SPSS.

The model is developed using parameters which should also be used during deployment. Some parameters are configured in the downstream applications. If the parameter values are passed when the stream is executed, these values are used. Otherwise, default values are used.

The following figure shows parameters used for deployment.

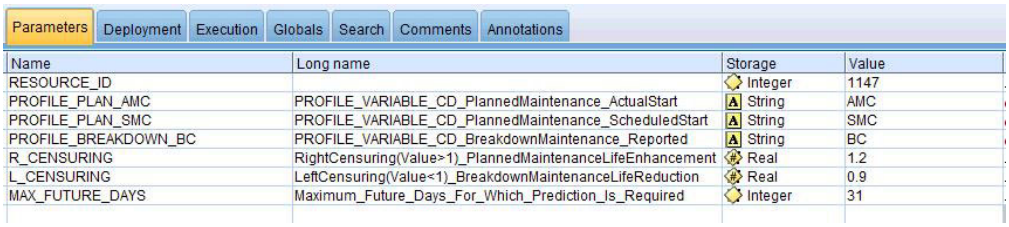


Figure 29. Parameters used for deployment

You can find all of these parameters using SPSS. However, only the `RESOURCE_ID` is exposed from the IIB end out of the box. This is because the stream has multiple execution branches that use scripts to sequence the parameters. You can see the scripts being referenced in the Execution tab.

Recommendations from ADM

The Maintenance predictive model provides scores and data that allow you to adjust maintenance dates optimally.

The deployed model, once invoked, helps to produce probability and propensity scores. However, probability and propensity scores may not be very useful to an end business user. Therefore, the results are consumed by IBM SPSS Decision Management, which then provides a more useful, text-based result.

The following figure shows the probability and propensity scores.

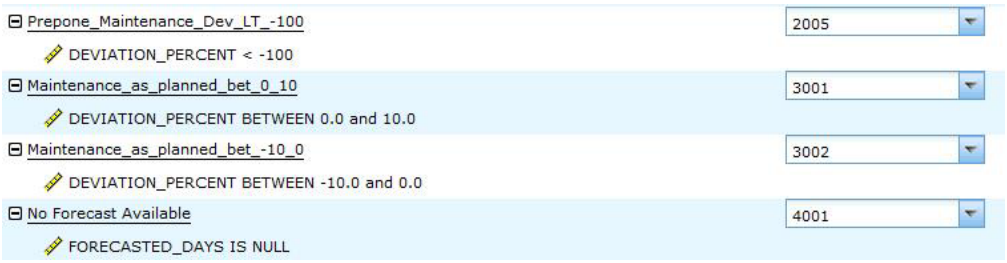


Figure 30. Probability and propensity scores

Based on the scores and the data received from the modeler stream, we can determine whether specific maintenance tasks should be rescheduled.

The Sensor Health predictive model

The Sensor Health predictive model analyzes an asset's sensor readings to help determine the likelihood that the asset will fail. If the likelihood of failure is high, you can schedule an urgent inspection of the machine.

The Sensor Health model continuously monitors the health of a machine or an asset and predicts potential machine faults in real time. The model uses historical sensor data profile values stored in the KPI tables and keeps a running status to determine the current health of an asset. The Sensor Health model can also be used to predict the future health of an asset.

Tip: If there are too many failures (for example, more than 30% of the days, or multiple times in a day), then instead of using the KPI tables for training, a user could use raw events from the event table for training with appropriate filtering or treatment of noise, if any.

Data understanding

The Sensor Health predictive model uses the `RESOURCE_KPI` and `MASTER_PROFILE_VARIABLE` tables.

The performance indicator table `RESOURCE_KPI` is used to hold aggregated values for each day. The table can be further used to prepare for model training and scoring. The `MASTER_PROFILE_VARIABLE` is used to help identify the specific profiles and select only the profiles that require further analysis.

The following diagram shows an example of the source data stream for the Sensor Health predictive model.

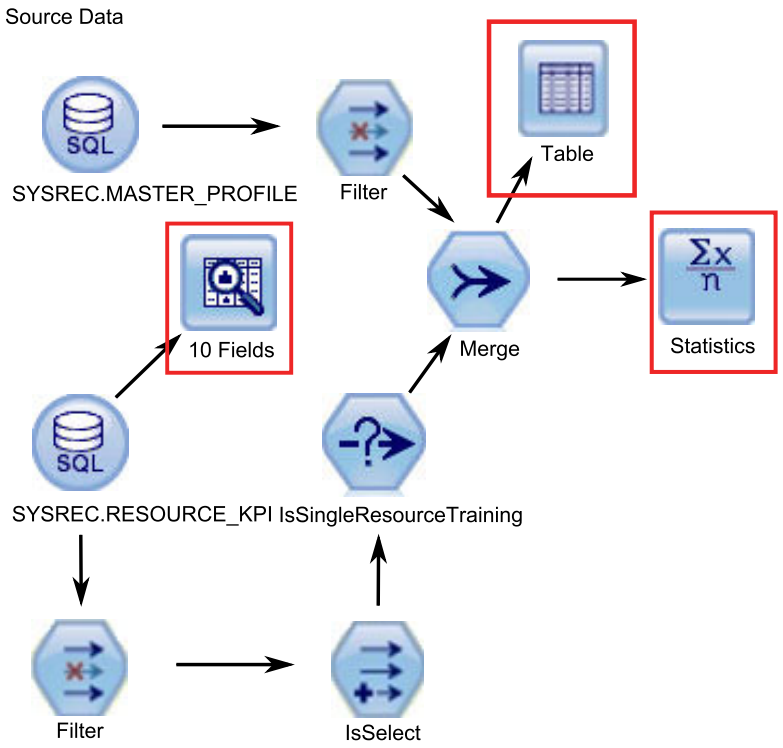


Figure 31. Example source data stream

In the diagram, the highlighted red boxes indicate possible ways that we can interpret the data. For example, the Statistics node addresses the summary statistics for individual fields and the correlations between fields. The Data Audit node provides a comprehensive first look at the data and is presented in an easy-to-read matrix. This matrix can be sorted and used to generate full-size graphs and a variety of data preparation nodes.

Data preparation

Data preparation for the Sensor Health predictive model occurs during execution of the SENSOR_HEALTH_DATA_PREP.str stream.

See the following table.

Table 17. The SENSOR_HEALTH_DATA_PREP.str stream

Stream name	Purpose	Input	Output
SENSOR_HEALTH_DATA_PREP.str	Data preparation stream extracts the data from IBM PMQ tables and prepares the data to be used in the modeling. The eligible data is exported to a csv file for the modeling.	The input data source contains the measurement type actual reading information of machines	A list of machines for which there is enough data and that are eligible for training to identify the patterns.

To prepare for analysis of the health score based on the measurement types, only the machine's measurement type attributes are considered. Each measurement type has a value. The number of times that the value exceeds the upper and lower limits is taken into account. Further, to train the model to identify failure patterns, a sufficient amount of failure data must be available. Machines that do not have sufficient failure data are not eligible for further modelling. Machine names are

logged in the file `Training_Eligibility_SensorAnalytics_Report.csv`. In this file, resources are indicated either with 1 (eligible) or 0 (not eligible).

The following diagrams show an example of a data preparation stream for the Sensor Health predictive model.

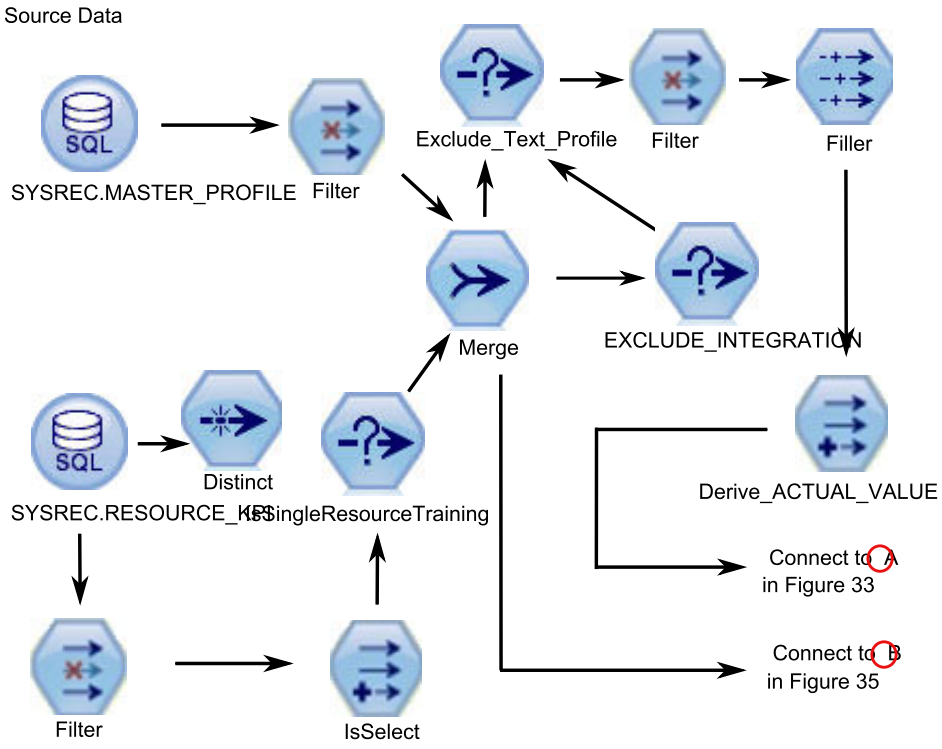


Figure 32. Example data preparation stream for the Sensor Health predictive model - Part 1

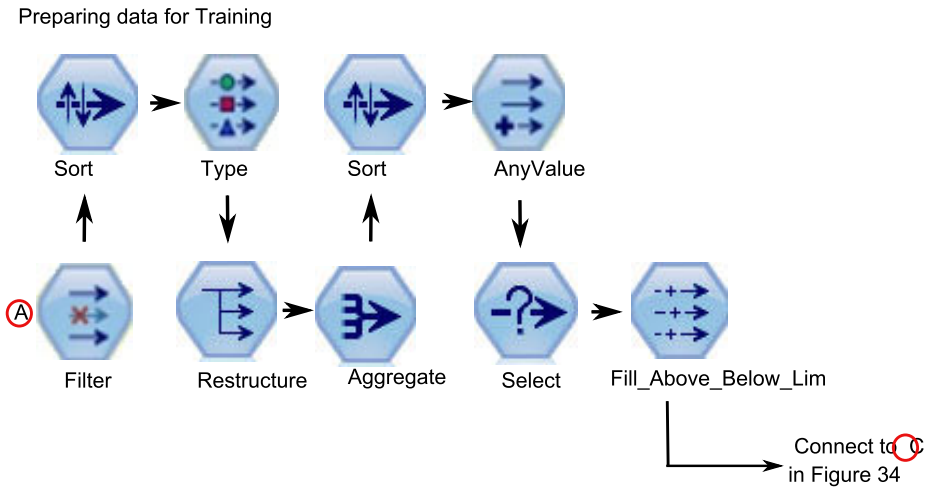


Figure 33. Example data preparation stream for the Sensor Health predictive model - Part 2

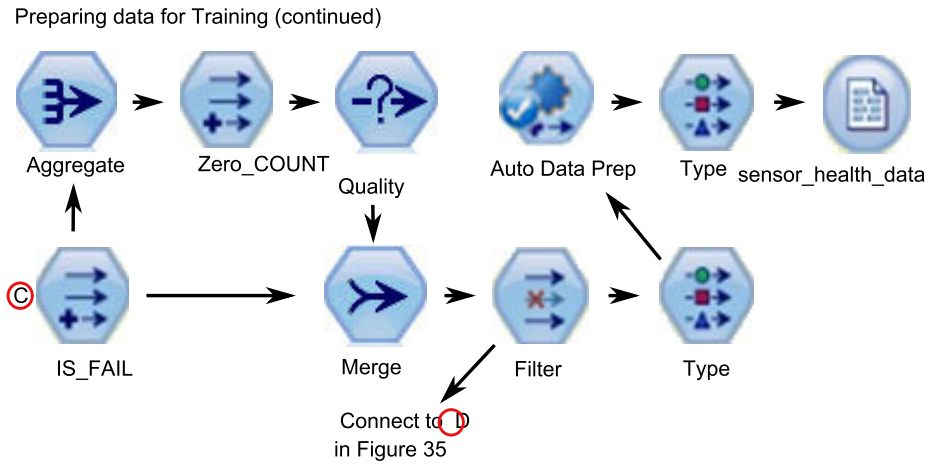


Figure 34. Example data preparation stream for the Sensor Health predictive model - Part 3

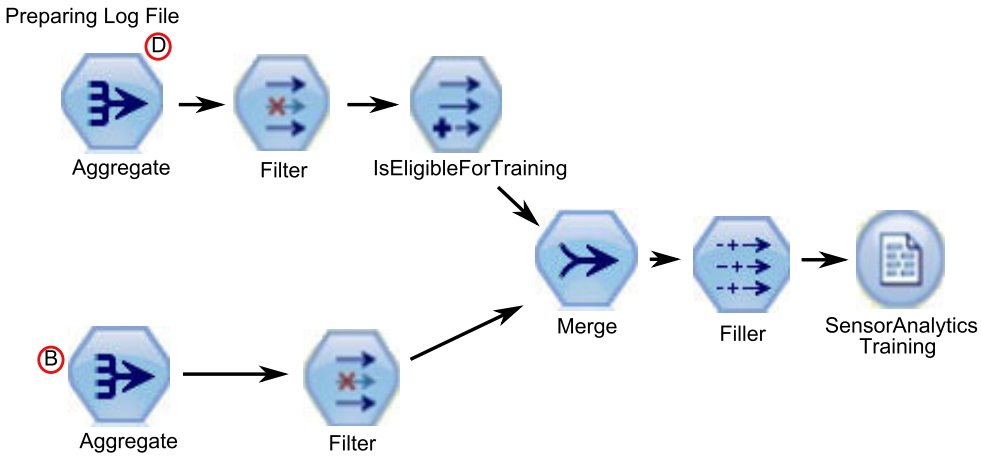


Figure 35. Example data preparation stream for the Sensor Health predictive model - Part 4

Data modeling

The Sensor Health predictive model uses the SENSOR_HEALTH_COMBINED.str stream.

See the following table.

Table 18. The SENSOR_HEALTH_COMBINED.str stream

Stream name	Purpose	Input	Target	Output
SENSOR_HEALTH_ COMBINED.str	Predicts failure of equipment based on the measurement types received thru the sensor details, train the models and also refresh them for the scoring service	The machine levels measurement type data received thru the sensor reading systems	`IS_FAIL`	Health score of the equipment

The following figures show an example of a modeling stream for the Sensor Health predictive model.

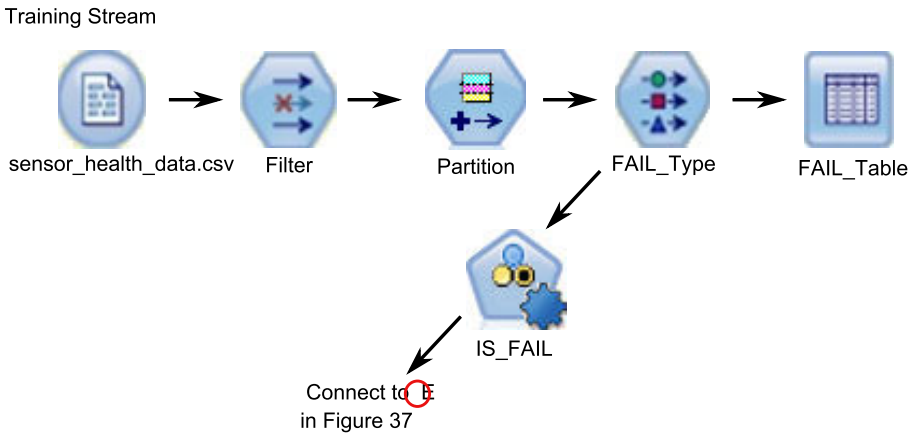


Figure 36. Example modeling stream for the Sensor Health predictive model - Part 1

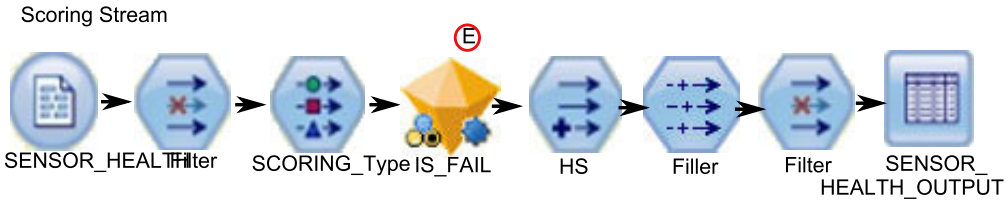


Figure 37. Example modeling stream for the Sensor Health predictive model - Part 2

Depending upon the input data, you might need to consider a different approach for the health score modeling. In addition, the concept of Splits at a resource id level (in Type Node) is introduced because, for each resource, the trained model would be unique.

The value of the health score of an asset is between 0 and 1. The higher the value of the health scores, the better the health of the asset. If the input data model and 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 auto classification model's confidence. Alternatively, raw and adjusted raw propensity scores can be used to generate such scores. In the model node, there are options to modify the costs, revenue, and weights. This setting depends on the requirements and the available data. Similarly, the data in this case is not balanced. Depending on the data and requirements, balancing might give better results.

Note: Not all of the models support propensity score outputs, especially when splits are enabled.

Evaluation of the model

The Sensor Health predictive model

At this point, most of the data mining activities are complete. However, there is a need to verify the model across the business success criteria that were established at the beginning of the project. We asked the following questions:

- Did the Health scores that were generated from sensor readings provide any helpful insights?
- What new insights or surprises were discovered?

- Were there any issues that were caused by inadequate data preparation or misinterpretation of the data? If there was an issue, we returned to the appropriate stage and rectified the problem.

Deployment

The Sensor Health predictive model uses a combined stream that performs several functions.

The model is developed using parameters which should also be used during deployment. Some parameters are configured in the downstream applications. If the parameter values are passed when the stream is executed, these values are used. Otherwise, default values are used. See the following figure.

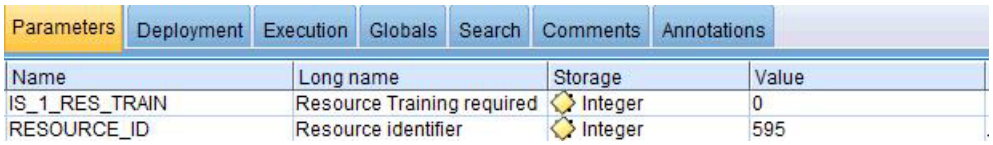


Figure 38. Parameters used for Deployment

If there is a provision to train one resource at a time, the resource id is passed along with the flag value.

This combined stream performs the following functions:

- helps to train the models
- refreshes data for the scoring service
- uses auto-modeling to identify the best suitable model
- produces health score output that measures the probability of machine failure

The stream has multiple execution branches that use scripts to sequence the parameters. Note that the scripts being referenced appear in the Execution tab. See the following figure.

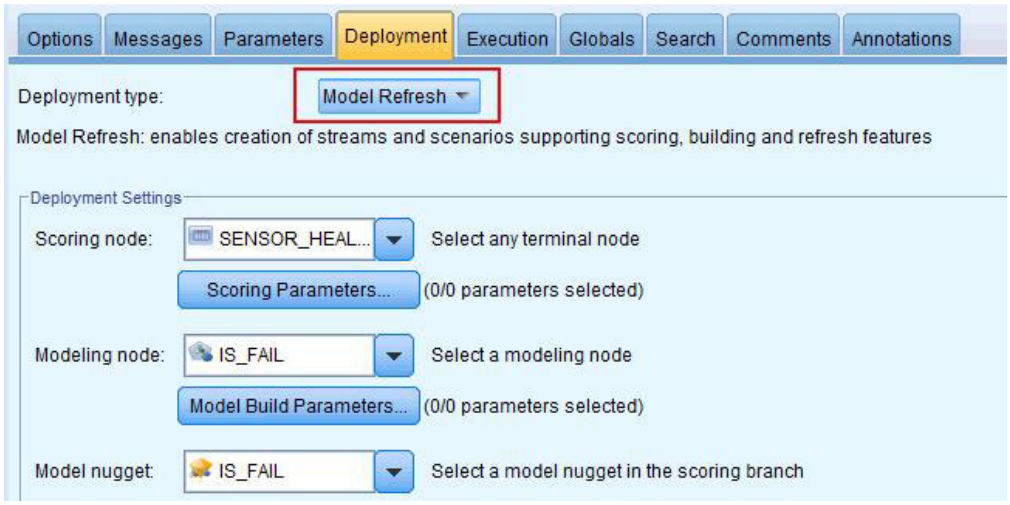


Figure 39. Refreshing the data for the scoring service

The stream is auto-generated when a training instance occurs and, for the real time scoring, in the `SENSOR_HEALTH_SCORE` service which is invoked by the IIB flows.

Recommendations

The Sensor Health predictive model provides recommendations for each asset.

Sensor analytics recommendations are produced using the real-time mode of invocation. In the invocation mode, the stream is developed using the ADM and a SENSOR_RECOMMENDATION service is configured for scoring services. The service is invoked to receive a recommendation for each asset. See the following figure.

Figure 40. Recommendation settings

Depending on the Heath score calculated from the Modeler, a recommendation of an Urgent Inspection (HS101) may be produced. For each HS101 code, a trigger is sent to Maximo to create the work order.

The Top Failure Reason predictive model

The Top Failure Reason predictive model helps you identify the top failure predictors for a given asset in order of their importance. You can then further analyze the identified reasons or parameters to assist in a guided trail from cause or root cause analysis to its respective pattern discovery.

This model is used to analyze and discover the top percentile and number of parameters which are influential in predicting a machine's failure (or optimal health) and their relative importance.

Understanding the data

The Top Failure Reason predictive model uses the event and master tables from the IBM PMQ database to get the sensor data available for each resource at a given point in time. It also gathers the defective and failure information.

The performance indicator table RESOURCE_KPI contains aggregated values for each day. You can use it to prepare for model training and scoring. The MASTER_PROFILE_VARIABLE and the MASTER_MEASUREMENT tables are used to help identify the specific profiles which are considered as parameters and will be considered for further analysis.

Preparing the data

Preparation for the Top Failure Reason predictive model includes merging data, selecting a sample subset, deriving new attributes, and removing the unwanted fields.

Depending on the data and identified goals, in this phase of data preparation the following tasks are performed:

- Merging data sets and records of the master and the events data
- Selecting a sample subset of data, identifying only the specified resource and profiles
- Deriving new attributes for each of the selected profiles based on the parameters

- Removing unwanted fields that are not required for further analysis

The measurements used as parameters are based on the data understanding. They are kept as parameters so they can be modified later, based on the data set. In the IIB layer, only the resource id is available.

Modeling the data

The prepared data is now considered for the modeling exercise. The target is set as the `IS_FAIL` variable and uses the Logistic regression model to obtain a percentile or probability value.

See the following figure.

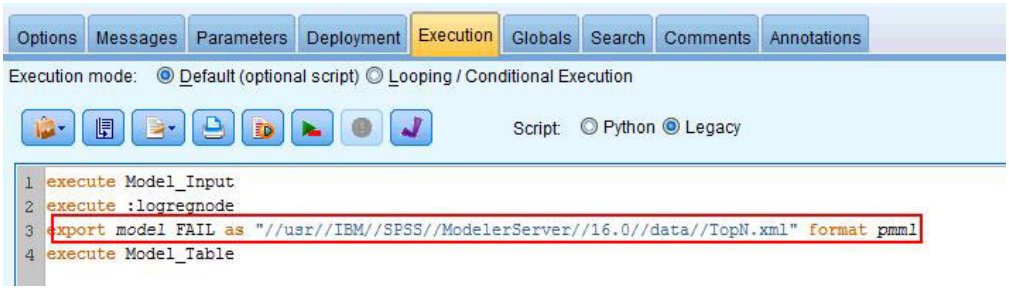


Figure 41. The Execution tab for the modeling stream

The stream has multiple execution branches that use scripts to sequence the parameters. You can see the scripts being referenced in the Execution tab. The important point here is to have exported the model `FAIL` in the pmml format. This is consumed in the `TopN_XML` stream to obtain the appropriate predictive importance of each profile.

Evaluation

The Top Failure Reason predictive model should be verified against the business success criteria established at the beginning of the project.

The Cumulative Gains chart shows the advantage of using a predictive model instead of a default, random model. The random model (depicted by a red line in the following figure) shows an equal ratio of percent gain (that is, the selection of entities of interest) to the percent of the total number of entities processed. Therefore, the red line has a 45 degree slope and the gains percentage is equal to the population percentile.

Cumulative Gains charts always start at 0 percent and end at 100 percent as you go from left to right. In the following Cumulative Gains chart, the percentage gain rises from zero percent to one hundred percent as the percentage of failures rises from zero percent to forty percent. Continuing after the forty percent failure rate, there are no gains until one hundred percent of the assets have failed.

A good predictive model has a steeper slope than the random model. When using a predictive model, the goal is to categorize and predict more entities of interest than would be done randomly. The model shown in the following figure can predict all the entities of interest by incorporating only 40 percent of the total population.

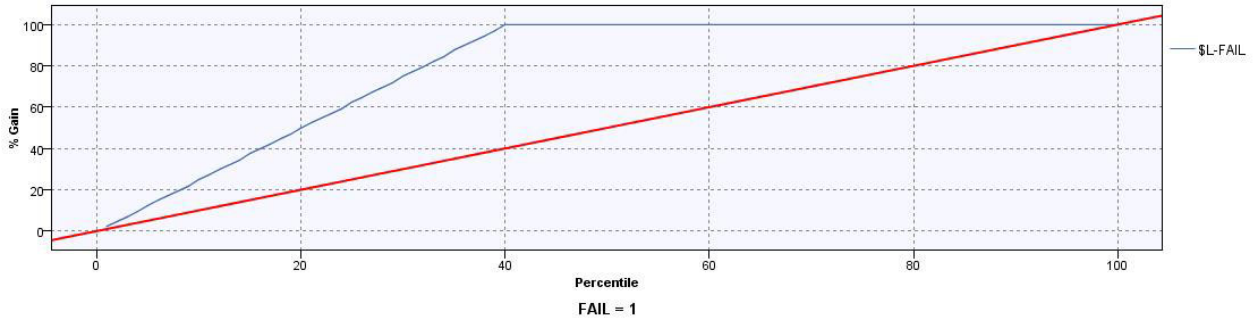


Figure 42. A Cumulative Gains chart

For example, a training and validation set contains only 2 percent of defective parts. Using a random selection model, we would have to select 100 percent of the parts to identify the intended 2 per cent of the faults. However, using the model in the preceding figure, we only need to select the top 40 percent of the parts that are most likely defective. This will ensure that all 2 percent of the intended defective parts (equivalent to 100 percent of gain) are covered in our target set.

Deployment

The output of the model calculates the cumulative sum of all the predictive importance values. These values are exported to a csv file. The IIB flow loads the csv file into the profiles table that will be consumed in the Cognos charts.

Note: You can alter the values at each resource level by exposing them to IIB and making a mechanism for picking the right parameters for each resource. Otherwise, for ad-hoc purposes, the parameters may be changed and triggered manually for each required resource. Also, the output table contents must be deleted manually for this operation, in case the data for the same resource exists from an earlier run.

The Integrated Health predictive model

The Integrated Health predictive model produces a predicted health score for each asset or process at a site. The health score is used to determine an asset's performance.

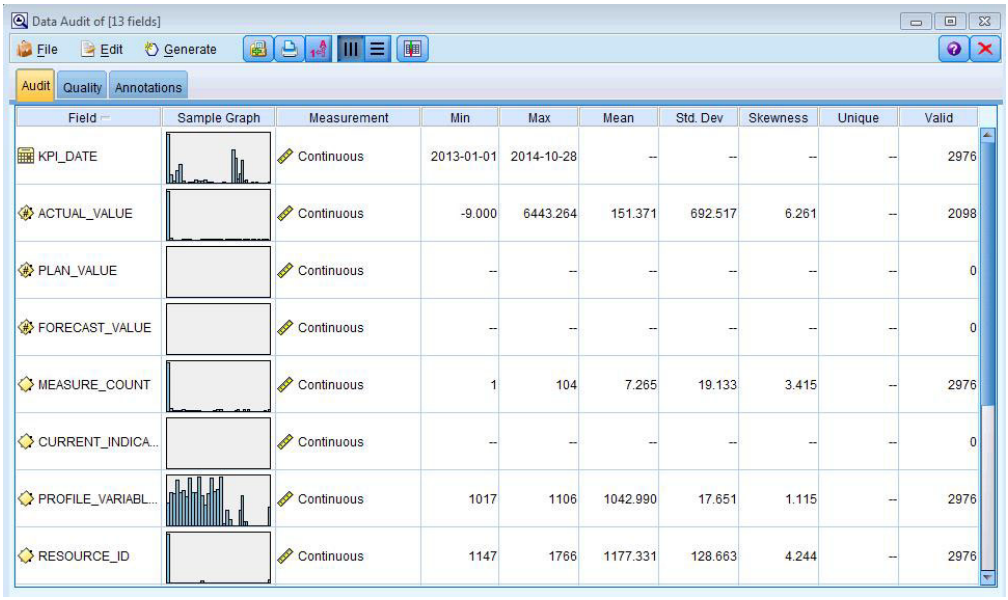
The health score determines how likely an asset is to fail. This health score model can continuously monitor machine or asset health and predict potential machine faults in real time. It uses historical defect data, operational information, and environmental sensor data to determine the current health of an asset. The health score model can also be used to predict the future health of an asset.

Data Understanding

From the `RESOURCE_ KPI` and the `MASTER_PROFILE_VARIABLE` tables, specific profiles like the maintenance score, sensor based health score, and the scheduled and forecasted maintenance days are considered for further analysis.

Tip: Analyze the output from the Data Audit node of the `RESOURCE_ KPI` to better understand the different types of data and their behavior. Ensure that you spend enough time during this stage so that you can perform the data preparation steps most efficiently.

See the following figure.



Field	Sample Graph	Measurement	Min	Max	Mean	Std. Dev	Skewness	Unique	Valid
KPI_DATE		Continuous	2013-01-01	2014-10-28	--	--	--	--	2976
ACTUAL_VALUE		Continuous	-9.000	6443.264	151.371	692.517	6.261	--	2098
PLAN_VALUE		Continuous	--	--	--	--	--	--	0
FORECAST_VALUE		Continuous	--	--	--	--	--	--	0
MEASURE_COUNT		Continuous	1	104	7.265	19.133	3.415	--	2976
CURRENT_INDICA...		Continuous	--	--	--	--	--	--	0
PROFILE_VARIABL...		Continuous	1017	1106	1042.990	17.651	1.115	--	2976
RESOURCE_ID		Continuous	1147	1766	1177.331	128.663	4.244	--	2976

Figure 43. The Data Audit node of the RESOURCE_ KPI table

Data Preparation

The Integrated Health predictive model uses the INTEGRATION_HEALTH_DATA_PREPARATION.str stream for data preparation.

See the following table.

Table 19. The INTEGRATION_HEALTH_DATA_PREPARATION.str stream

Stream name	Purpose	Input	Output
INTEGRATION_HEALTH_ DATA_PREPARATION.str	A Data preparation stream extracts the data from IBM PMQ tables and prepares the data to be used in the modeling. The eligible data is exported to a csv file for the modeling.	The input data source contains the sensor and maintenance-based health score information of machines. It also contains the scheduled and forecasted maintenance details.	Machines where there are enough data and is eligible for training to identify the patterns

The following diagrams show an example of a data preparation stream for the Integrated Health predictive model.

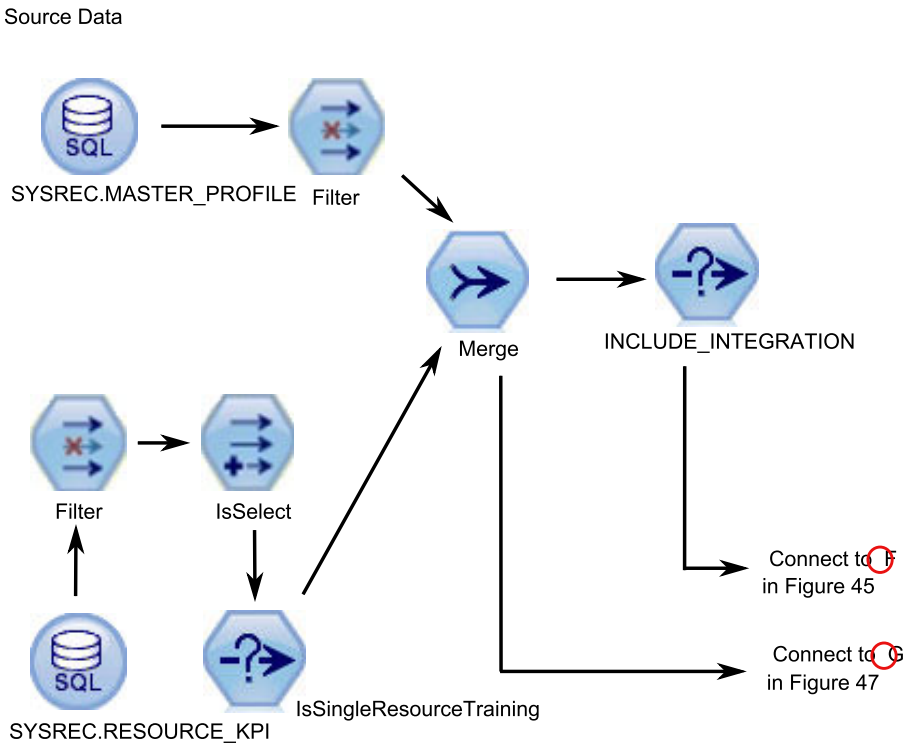


Figure 44. Example data preparation stream for the Integrated Health predictive model - Part 1

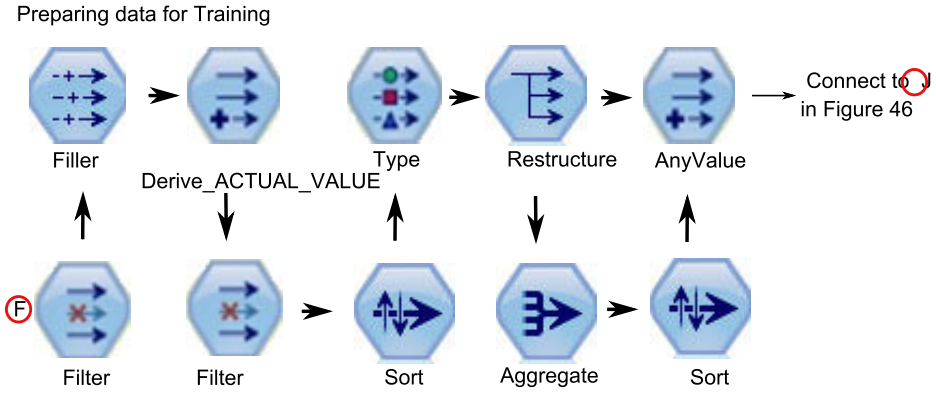


Figure 45. Example data preparation stream for the Integrated Health predictive model - Part 2

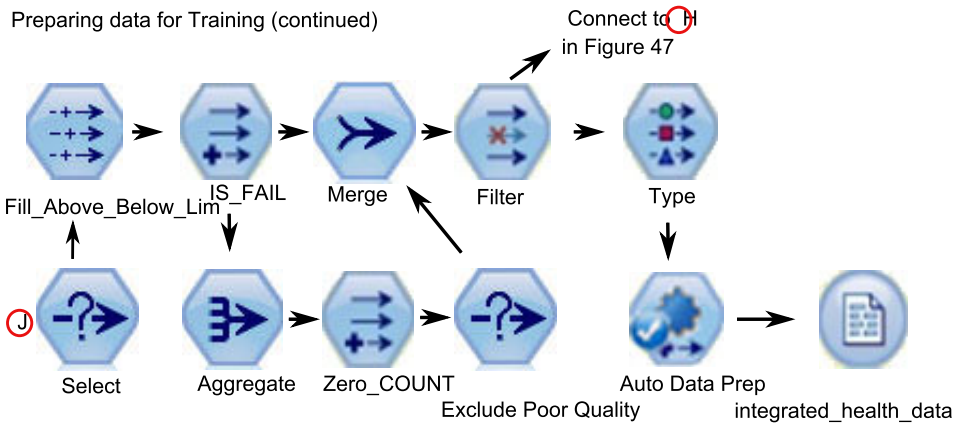


Figure 46. Example data preparation stream for the Integrated Health predictive model - Part 3

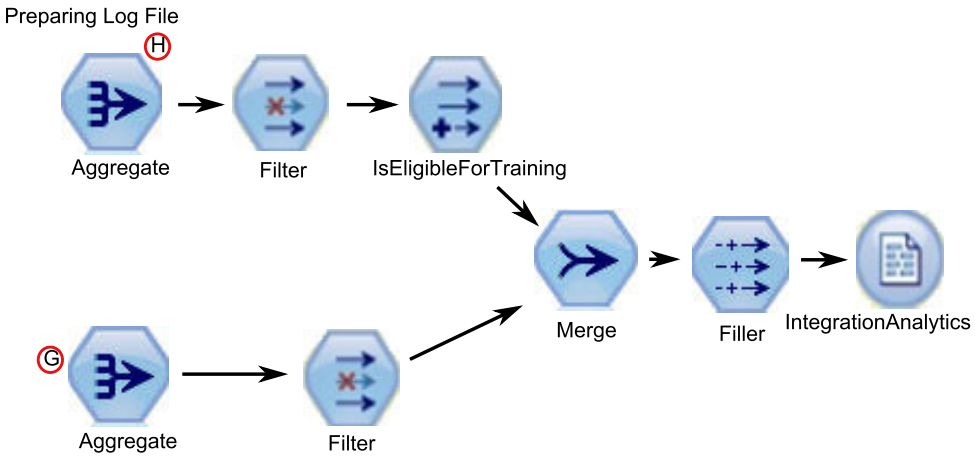


Figure 47. Example data preparation stream for the Integrated Health predictive model - Part 4

The integrated health score is executed using data provided by the maintenance and sensor health score Models. Further, to train the model to identify failure patterns, a sufficient amount of failure data must be available. Machines that do not have sufficient failure data are not eligible for further modelling. Machine names are logged in the file Training_Eligibility_IntegratedAnalytics_Report.csv. In this file, resources are indicated either with 1 (eligible) or 0 (not eligible).

Modeling

The Integrated Health predictive model uses the INTEGRATION_HEALTH_COMBINED.str stream for the modeling phase.

See the following table.

Table 20. The INTEGRATION_HEALTH_COMBINED.str stream

Stream name	Purpose	Input	Target	Output
INTEGRATION_HEALTH_ COMBINED.str	To Train the models and refresh them for the scoring service	The input data source contains the sensor and maintenance-based health score information of machines. It also contains the scheduled and forecasted maintenance details.	IS_FAIL	Integrated Health score of the equipment

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

Alternatively, raw and adjusted raw propensity scores can be used to generate such scores. In the model node, there are options to modify the costs, revenue, and weights. This setting depends on the requirements and the available data. Similarly, the data in this case is not balanced. Depending on the data and requirements, balancing might give better results.

Note: Not all of the models support propensity score outputs, especially when splits are enabled.

The following diagrams show an example of a modeling stream for the Integrated Health predictive model.

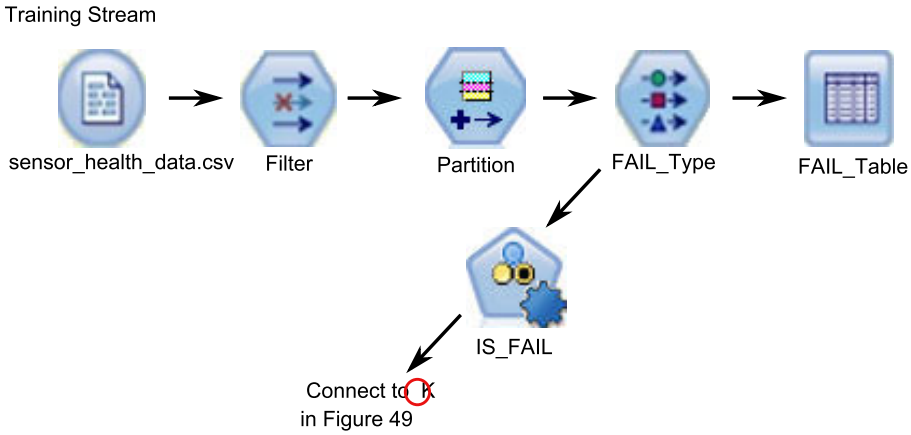


Figure 48. Example modeling stream for the Integrated Health predictive model - Part 1

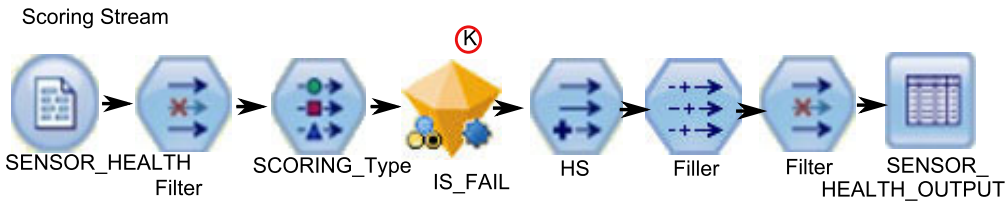


Figure 49. Example modeling stream for the Integrated Health predictive model - Part 2

Evaluation

The model should be verified against the business success criteria established at the beginning of the project.

At this point, most of the data mining activities are complete. However, there is a need to verify the model across the business success criteria that were established at the beginning of the project. We asked the following questions:

- Did the Health scores that were generated from sensor readings provide any helpful insights?
- What new insights or surprises were discovered?
- Were there any issues that were caused by inadequate data preparation or misinterpretation of the data? If there was an issue, we returned to the appropriate stage and rectified the problem.

Deployment

This stream is auto generated when a training happens and for the real time scoring -- `INTEGRATED_HEALTH_SCORE _HEALTH_SCORE` service configured which would be invoked by the IIB flows.

This combined stream performs the following functions:

- helps to train the models
- refreshes data for the scoring service
- uses auto-modeling to identify the best suitable model
- produces health score output that measures the probability of machine failure

The stream has multiple execution branches that use scripts to sequence the parameters. You can see the scripts being referenced in the Execution tab. See the following figure.

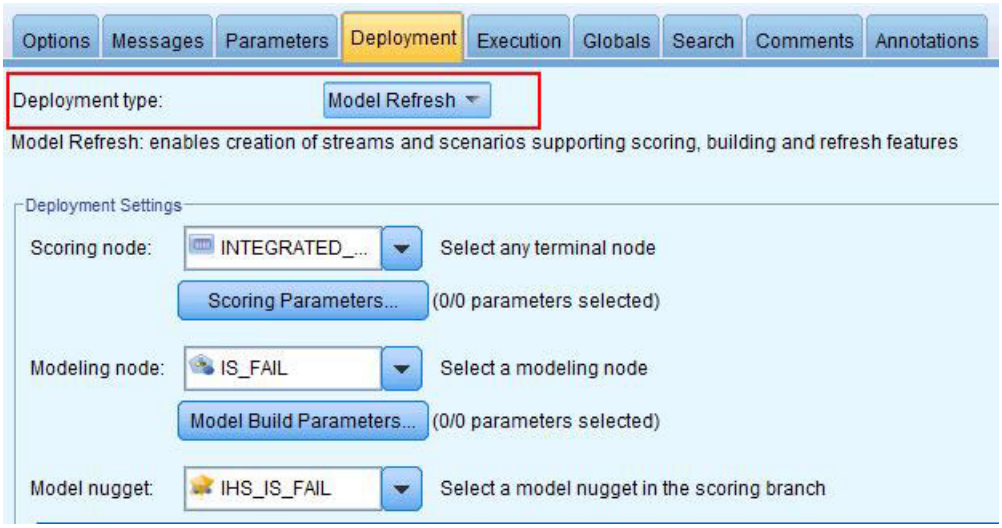


Figure 50. Refreshing the data for the scoring service

Recommendations

Integration analytics recommendations are produced using the real-time mode of invocation. In the invocation mode, the stream is developed using the ADM and an `INTEGRATION_HEALTH_RECOMMENDATION` service is configured for scoring services. The service is invoked to receive a recommendation for each asset.

See the following figure.

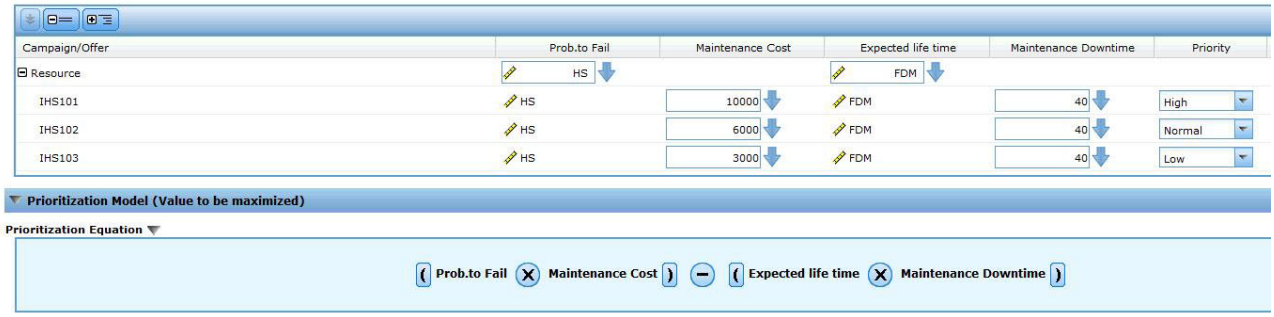


Figure 51. Recommendation settings

The recommendations seems similar to sensor but here in the Integrated Health score is being used, further we have the recommendations prioritized based on the equation which considers the costs and the health scores calculated thru sensor and maintenance.

Chapter 8. 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 IBM Analytical Decision Management Knowledge Center (http://www.ibm.com/support/knowledgecenter/SS6A3P).

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 or 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 9. Reports and dashboards

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

You can use IBM Cognos Report Studio to create scorecards and reports. Before you run the reports, familiarize yourself with the behavior of reports in Report Studio. For example, a star beside a prompt indicates that it is required. For information about how to use Report Studio, see *IBM Cognos Report Studio User Guide*. You can obtain this user guide from IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter/).

You can modify the data model for these reports by using IBM Cognos Framework Manager. For more information, see Appendix D, "IBM Cognos Framework Manager model description," on page 171.

The following table describes the reports available from the Site Overview Dashboard.

Table 21. 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.
List Of 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 dashboard.

Table 22. Reports available from the Equipment 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 23. 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.

Site Overview Dashboard

The Site Overview Dashboard 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. It contains the Site Summary Report, the Health Score Trend Bar Chart, the Health Score Contributors Pie Chart, and the Incident and Recommendation Analysis Bar Chart.

You can use the following prompt filters in the dashboard. The filters are applied to all reports and charts on the dashboard:

- From Date
- To Date
- Location
- Resource sub type

Site Summary Report

The following table describes the measures for the Site Summary Report.

Table 24. Site summary measures

Measures	Description
Health Score	The evaluation of the health of a resource that is based on predictive models.

Table 24. Site summary measures (continued)

Measures	Description
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 generated by resources.
Recommendation Count	A recommendation is generated by the predictive model when a resource approaches a failure. This measure counts the number of recommendations generated.
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 location.

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 date.

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 resource.

Incident and Recommendation Analysis Bar Chart

You can use this report to analyze incidents and recommendations.

You can access the Drill Through - Incident and Recommendation Event List drill through report from the Incident and Recommendation Analysis Bar Chart:

Note: The drill through reports are stored in the **Drill Through Reports** folder. The reports in this folder are intended to be run from the main report with which they are associated. Do not run the drill through reports on their own.

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

Table 25. Incident analysis bar chart

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

Recommendation and Incident Count Accessibility List

This chart provides the same information as the Incident and Recommendation Analysis Bar Chart in an accessible format.

The Recommendation and Incident Count Accessibility List contains the following drill through reports:

Drill Through - Incident Event List
This report shows the incident event list in tabular form.

Drill Through - Recommendation Event List
This report shows the recommendation event list in tabular form.

Note: The drill through reports are stored in the **Drill Through Reports** folder. The reports in this folder are intended to be run from the main report with which they are associated. Do not run the drill through reports on their own.

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 26. 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 27. 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 28. 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 by 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 by 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 29. Measures in the Outliers report

Measures	Measure description	Measure source
Life to Date Average	The daily average measurement for 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

Table 29. Measures in the Outliers report (continued)

Measures	Measure description	Measure source
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:

- 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 dashboard

The Equipment 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 30. 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.

Table 30. Measures in the Equipment profile report (continued)

Measures	Measure description
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 code
- 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 31. 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 code
- Location
- Event code
- Calendar date
- Start time
- End time
- Measurement type

- Profile variable
- Sigma level

Equipment run chart

The 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 code
- Location
- Event code
- Calendar date
- Start time
- End time
- Measurement type

Equipment outliers

The equipment outliers report 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 32. Measures in the Equipment outliers report

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`
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 code

- Location
- Event code

Event type history report

The event type history 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 code
- 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 33. Measures in the Defect summary report

Measures	Measure Description
Defect Count	The number of defects that are reported.
Quantity Produced	The quantity that is produced.
Defect Rate	`Defect Count` divided by `Quantity Produced`.
Quantity Planned	The quantity that is expected to be produced.
Defect Target	The acceptable number of defects.
Test Failure Rate	`Test Failures` divided by `Number of Tests`.
Target Defect Rate	`Defect Target` divided by `Quantity Planned`.
Inspection Time	The amount of time that is spent inspecting the product.

Table 33. Measures in the Defect summary report (continued)

Measures	Measure Description
Assembly Time	The amount of time that is spent producing the product.
Inspection Time Rate	`Inspection Time` divided by the `Assembly Time`.
Inspection Count	The number of inspections performed.
Inspection Rate	`Inspection Count` divided by `Quantity Produced`.
Average Assembly Time	.`Assembly Time` divided by `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 34. Measures in the Defect plan vs actual report

Measures	Measure description
Defect Rate	Defect Count divided by Qty Produced.
Target Defect Rate	The rate of the Defect Target divided by Quantity Planned measure.

The following prompt filters are applied to this report:

- Process hierarchy
- Calendar from and 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 35. Measures in the Defect rate vs inspection rate line chart

Measures	Measure description
Defect Rate	Defect Count divided by Quantity Produced.
Inspection Rate	Inspection Count divided by 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 is applied to this report.

Audit Report

The Audit Report shows the count of rows in the major master data tables.

Note: The `Asset` count is shown in the Audit Report.

The Audit Report contains the following drill through reports:

Drill Through - Resource List
Lists the resources for a resource type.

Example

For example, the Audit Report shows the count for the `Asset` resource type. Click on this count number to open the Drill Through - Resource List report that lists all of the assets.

Drill Through - Profile Variables
Lists all measures and key performance indicators that are being tracked in daily profiles and historical snapshots.

Drill Through - Process List
Lists all production processes.

Drill Through - Material List
Lists materials that are used in the production process.

Drill Through - Production Batch List
Lists production batches for a product.

Drill Through - Measurement Type List
Lists measurement types. For each measurement type, the report shows unit of measure and aggregation type.

Note: The drill through reports are stored in the **Drill Through Reports** folder. The reports in this folder are intended to be run from the main report with which they are associated. Do not run the drill through reports on their own.

The following table describes the measures in the Audit Report. The measure source is the report calculation.

Table 36. Measures in the Audit Report

Measures	Measure Description	Hierarchies
Count of Resources by Type	A count of rows in the dimension	Resource by Type

Table 36. Measures in the Audit Report (continued)

Measures	Measure Description	Hierarchies
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 Measurement Types	A count of measurement types in the dimension	Measurement Type
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

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

Maintenance Overview Report

The Maintenance Overview Report provides insights by using existing maintenance data and can include sensor data when your organization's data matures. The Maintenance Overview Report also gives you insight into stable life and rapid wear scenarios.

This report shows the sensor health score, maintenance health score, and the integrated health score by location and resource. The sensor health score is a near real-time value calculated from the sensor reading. The maintenance health score is calculated from the maintenance logs. The sensor health score and the maintenance health score are combined to give the integrated health score.

You can set the following prompt filters in this chart:

- Location
- Health Score
- Recommendation
- Absolute Deviation %
- Forecasted Days to Next Maintenance
- Scheduled Days to Next Maintenance
- Event Code

The following measures are reported in this chart.

Table 37. Measures in the Maintenance Overview Report

Measure	Description
Location	Location of the resource.
Resource Sub Type	Sub type of the resource.
Resource	Identifies the resource.
Health Score	The sensor health score, maintenance health score, and the integrated health score are values between 0.00 and 1.00. The higher the score, the better the performance of the resource.
Days To Next Forecasted & Scheduled Maintenance	The forecast number of days to next maintenance and number of days to next scheduled maintenance. The maximum positive deviation, minimum positive deviation, maximum negative deviation, and minimum negative deviation are also indicated.
Forecast - Schedule Deviation	The difference between the forecast days and the scheduled days.
Recommendation	The recommended action as indicated by the health scores.

Click **Summary** to see a summary of the number of resources, total counts, and percentage of total count for each recommendation.

Maintenance Advance Sorting

Click **Advance Sorting** to drill through to the Maintenance Advance Sorting report. This report displays the same measures as the main report in a tabular format. You can sort on a column by clicking the column header. The prompt values from the main report are used in the Maintenance Advance Sorting report. You can change the prompt values in the Maintenance Advance Sorting report and run it with the new values.

Maintenance Health and Failure Detail Report

Click a resource in the **Resource** column to drill through to the Maintenance Health and Failure Detail Report for the resource.

The prompt values from the main report are used in this chart. You can change the following prompt filters in this chart and run it with the new values:

- From date
- To date
- Location
- Resource

You can include or exclude the following events:

- Breakdown maintenance
- Planned maintenance
- Forecasted maintenance
- Scheduled maintenance

Each event that you include appears as a bar on the chart. The bar indicates the date on which the event occurs. The health score, which is a value between zero and one, is indicated on the y axis. The x axis indicates the date of the health

score. Health scores that occur before the current date are historical health scores. Health scores that occur after the current date are forecast health scores. The current health score is shown for the current date.

Click **Top N Failure Analysis** to drill through to the Top N Failure Analysis Report. For more information, see "TopN Failure Analysis Report" on page 123.

Note: It is possible that the location of a resource in the Maintenance Health and Failure Detail Report is different from the location of the same resource in the Top N Failure Analysis Report. If this happens, the **Location** field in the TopN Failure Analysis Report will be empty and you must select a location from the list and then run the report.

Statistical process control reports

The statistical process control (SPC) reports monitor the stability of a process. The charts in the reports show the data points in relation to the mean value, and upper and lower control limits.

SPC - Histogram

This bar chart is an overview of the frequency of an event or observation across a set of ranges or bins. The y axis shows the frequency. The x axis shows the bins. The height of the bar in a bin indicates the frequency of the event that falls into the range.

You can set the following prompt filters in this chart:

- From Date
- To Date
- Location
- Resource
- Event Type
- Measurement Type
- Event Code
- Number of Bins: Select **Number of Bins** to set the number of bins to display in the chart. The value that you select from the **User Selected Value** list is the number of bins that appears on the x axis.
- Bin Interval: Select **Bin Interval** to set the range for each bin. Enter the range in the **User Selected Value** field.
- Minimum: The minimum value for the bin range limit. Use this filter to set the lowest data point to be included in the data set.
- Maximum: The maximum value for the bin range limit. Use this filter to set the highest data point to be included in the data set.

The following measures are reported in the SPC Histogram chart.

Table 38. Measures in the SPC Histogram chart

Measure	Description
Frequency	Number of events that fall into a bin. The height of the bar indicates the frequency. Displayed on the y axis.
Bin Range	The bin interval. Displayed on the bins on the x axis.

Table 38. Measures in the SPC Histogram chart (continued)

Measure	Description
Frequency of Bin Containing Mean Value	Frequency of the bin that contains the mean value of the events in the chart.
Count of Observations	Total number of events in the chart.
Mean	Mean value of the data in the chart.
Median	Median value of the data in the chart.
Minimum	Minimum value of the data in the chart.
Maximum	Maximum value of the data in the chart.
Range	The difference between the minimum and maximum values.
Standard Deviation	The standard deviation of the data in the chart.
Skewness	Indicates how symmetrical or asymmetrical the data is.
Kurtosis	Indicates if the data is peaked or flat, relative to a normal situation.
Start Date	The date of the earliest event in the chart.
End Date	The date of the latest event in the chart.

The **Fitted Distribution** line shows the trend in the data.

Click **X Bar R/S Chart** to run the SPC - X Bar R/S Chart.

SPC - X Bar R/S Chart

The SPC - X Bar R/S chart shows variations in a process. You can use this chart to evaluate the stability of a process over a set of day ranges.

The SPC - X Bar chart shows how the average process changes over time. The median control limit is indicated by a dotted line. The solid lines on the chart indicate the upper and lower control limits. Data points that occur outside of the control limits indicate that the process is unstable.

The SPC - R/S chart shows how the average within a sub group changes over time. The SPC - R (range) chart is displayed when you enter a sub group value of 10 or less. The SPC - S (standard deviation) chart is displayed when you enter a sub group value that is greater than 10. The sub group size prompt controls the ranges that display on the x axis of both charts. For example, if you set the sub group prompt to 11 and the charts contain data from Jan 1 to Mar 9 (68 days), the x axis displays six ranges in 11-day increments. The seventh range contains a 2-day increment. The y axis on both charts indicates the control limit value.

The following prompts apply to this chart:

- From date
- To date
- Location
- Resource sub type
- Resource
- Measurement type
- Event code
- Resource code

- Profile variable type
- Sub group

Advanced KPI Trend Chart

This chart compares multiple key performance indicators (KPIs) across multiple resources. You can use this chart to analyze variations in a resource against a set of profiles. The main chart shows monthly data and you can drill down to a daily chart.

You can set the following prompt filters in this chart:

- From Date
- To Date
- Location
- Resource Sub Type
- Resource
- Profiles
- Event Code

Each chart displays data for one profile and all resources that you select in the prompt lists. By default, the chart displays all resources and all profiles but for clarity, select a few related profiles to analyze across a set of resources. Each data point in the chart represents one month of data for the profile. Click a data point or on the month on the x axis to see one month of data by day.

The following measures are reported in this chart.

Table 39. Measures in the Advanced KPI Trend Chart

Measure	Description
Actual Value	The value of the profile or measure for the resource for the month. Shown on the y axis.
Date	The year and month. Shown on the x axis. The month does not display if there is no data for the month.

QEWS - Inspection Chart

The quality early warning system inspection chart reports the failure rates and values of evidence that the underlying failure rate process is unacceptably high.

You can report on a specific product type or a group of products. The analysis is based on data for a specified time period.

The chart shows performance of parts by vintage, where vintage is the day that the part was shipped. However, the analysis can be done for other vintages, such as the day of part manufacturing, or the day of part testing.

This chart is generated by IBM Predictive Maintenance and Quality daily. If the daily chart was not generated for the date that you select, the report is empty.

You can set the following prompt filters in this chart:

- Product Type
- Product Code

The chart heading contains the following information:

- Product code
- Last run date of the chart
- Period during which the product was shipped (start date and end date)
- Number of parts that were shipped over the period
- Number of parts that failed over the period
- Failure rate per 100 units over the period

Note: This chart is not an IBM Cognos Report Studio report so you cannot modify it in Report Studio.

Failure rate chart

This chart has a dual x axis that shows vintage number and Cumulative N_Tested. Vintage number is the day number that the part was shipped during the period. Cumulative N_Tested is the number of parts that were tested. The y axis shows the failure rate of the product per 100 units. A data point in the chart indicates the failure rate for a vintage number. The Acceptable Level is a horizontal line that indicates the acceptable failure rate.

Evidence chart

This chart has a dual x axis that shows vintage number and Cumulative N_Tested. Vintage number is the day number that the part was shipped during the period. Cumulative N_Tested is the number of parts that were tested. The y-axis shows the level of evidence that the underlying process failure rate is unacceptable, which is computed by using a weighted cumulative sum (CUSUM) formula.

The H Value is a horizontal line on the chart that shows the failure rate threshold value. The CUSUM values that are higher than H Value are displayed as triangles on the chart. The triangles indicate unacceptable process levels in the data. The vertical dotted line indicates the last time that the vintage number had an unacceptable failure rate. The forgiveness marker is the point in time when the process accumulated enough statistical evidence to suggest that its underlying failure rate was acceptable.

Summary list

The summary list header contains the same information as the chart heading. The summary list shows detailed information by vintage. It includes date, failure rate, total quantity that failed, and other data.

QEWSL - Warranty Chart

The quality early warning system life time (QEWSL) warranty chart reports the replacement rates for a specific product type and product code over a time period.

This chart is generated by IBM Predictive Maintenance and Quality daily. If the daily chart was not generated for the date that you select, the report is empty.

You can set the following prompt filters in this chart:

- Run Date
- Product Type
- Product Code

The chart heading contains the following information:

- Product code
- Last run date of the chart
- Period during which the product was shipped (start date and end date)
- Number of parts that were shipped over the period
- Number of parts that failed over the period
- Replacements per machine month of service over the period

Note: This chart is not an IBM Cognos Report Studio report so you cannot modify it in Report Studio.

Replacement rate chart

This chart has a dual x axis that shows vintage number and cumulative number of machine months of service. Vintage number is the day number that the part was shipped during the period. Cumulative number of machine months is the total number of machine months of service that is accrued by the population of machines that have the parts installed. The y axis shows the replacement rate of the product per machine month. A data point on the chart indicates the replacement rate for a vintage. The Acceptable Level is a horizontal line on the chart that shows the acceptable replacement rate.

If the severity of wear out conditions is greater than zero, the chart contains a curve corresponding to monitoring wear out conditions. The levels of wear out index that is based on vintages summarized monthly corresponds to the y-axis on the right side of the chart.

Evidence chart

This chart monitors the reliability or characteristics of the lifetime of a part. The chart has a dual x axis that shows vintage number and cumulative number of machine months of service. Vintage number is the day number that the part was shipped as part of a machine. Cumulative machine months is the number of machine months of service. Cumulative machine month is shown on the x-axis. The y-axis shows the level of evidence that the underlying process replacement rate is unacceptable. It is computed by using a weighted cumulative sum (CUSUM) formula.

Threshold H is a horizontal line that shows the replacement rate threshold value. The CUSUM values that are higher than Threshold H are displayed as triangles on the chart. The triangles indicate unacceptable process levels in the data. The vertical dotted line indicates the last time that the vintage number had an unacceptable replacement rate per machine month.

If the severity of wear out conditions is greater than zero, the chart contains a curve corresponding to monitoring wear out conditions. The wear out curve is shown together with the corresponding threshold.

Summary list

The summary list header contains the same information as the chart heading. The summary list shows detailed information by vintage number. It includes date, number of parts that were tested, total quantity, and other data.

TopN Failure Analysis Report

This report shows the profiles that contribute to the failure of a resource. Each profile has an importance value that is expressed as a percentage. The total of the importance values displayed on the report is 100%.

The profile is indicated by the x axis. The importance value is indicated on the y axis. Each profile is represented by a bar on the chart. The higher the importance value, the more that the profile contributes to the failure of the resource. The profile importance values that display in blue contribute the most (top 80%) to whether the resource fails. The profile importance values that display in yellow contribute the least (bottom 20 %) to whether the resource fails.

The curved line on the chart indicates the cumulative importance value.

You can set the following prompt filters in this chart:

- Location
- Resource Sub Type
- Resource
- Resource Code
- Event Code

You can also access this report from the Maintenance Health and Failure Detail Report. For more information, see "Maintenance Overview Report" on page 116.

Drill through to the statistical process control reports

Select a profile from the **Analyze Profile Variable** list. Click a link to one of the statistical process control (SPC) reports.

Note: The raw measurement type for the profile is passed to the SPC report.

Appendix A. Accessibility features

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

For information about the commitment that IBM has to accessibility, see the IBM Accessibility Center (www.ibm.com/able).

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

Report output

In IBM Cognos Administration, you can enable system-wide settings to create accessible report output. For more information, see the *IBM Cognos Business Intelligence Administration and Security Guide*. In IBM Cognos Report Studio, you can enable settings to create accessible output for individual reports. For more information, see the *IBM Cognos Report Studio User Guide*. You can access the previously mentioned documents at IBM Knowledge Center (http://www.ibm.com/support/knowledgecenter).

Appendix B. Analytics Solutions Foundation

Use IBM Analytics Solutions Foundation to extend or modify IBM Predictive Maintenance and Quality.

The IBM Predictive Maintenance and Quality data model can be extended by modifying the solution definition in IBM Analytics Solutions Foundation. After modifying the solution definition, the data model must be regenerated. Changes may also be required to the IBM Integration Bus flows that depend on the model.

IBM Predictive Maintenance and Quality event processing can also be extended by modifying orchestration definitions in IBM Analytics Solutions Foundation. These orchestration definitions determine the event processing behavior of IBM Analytics Solutions Foundation within the event processing flows implemented in IBM Integration Bus.

Analytics Solutions Foundation is an alternative to using the flat file application programming interface (API) to extend the Predictive Maintenance and Quality solution. Analytics Solutions Foundation helps you to define orchestrations without writing code. You must be familiar with data modeling techniques and database design to use Analytics Solutions Foundation.

Analytics Solutions Foundation consists of one or more orchestration definition files and a solution definition file. A common pattern is to use one orchestration definition file for each type of event to be processed. The solution definition file contains common elements that are referenced by the orchestration definition files. For more information, see Appendix E, “IBM Predictive Maintenance and Quality Artifacts,” on page 187.

The following list explains some important concepts for using and understanding Analytics Solutions Foundation.

Business key versus surrogate key

Business keys are human-readable values that are inputs provided to the application. Surrogate keys are guid (generated unique identifier) values that are used by the application as primary keys on dimensional tables to facilitate multi-language support. The database tables show the surrogate keys, not the business keys.

Support tables

The following tables are defined and populated by Analytics Solutions Foundation:

- LANGUAGE
- TENANT
- CALENDAR
- EVENT_TIME
- KEY_LOOKUP

Orchestration definition

An orchestration definition consists of a list of event orchestration mappings and a list of orchestrations.

Operational systems and devices generate events. Orchestration is the process of mapping these events to a series of steps. Each step has an adapter that performs the step and a configuration that tells the adapter how to process the event. The data from the event is calculated into a value or score. The calculation is performed using event data to update key performance indicators (KPIs) and profiles. Business rules define the action to take, depending on the score.

Master data definition

Master data describes the type of resource that you want to manage. A master data definition defines a table that stores master data. Master data is defined in the solution definition file.

Use the master data definition components to define the master data tables that meet your business requirements.

The following diagram shows the master data definition schema:

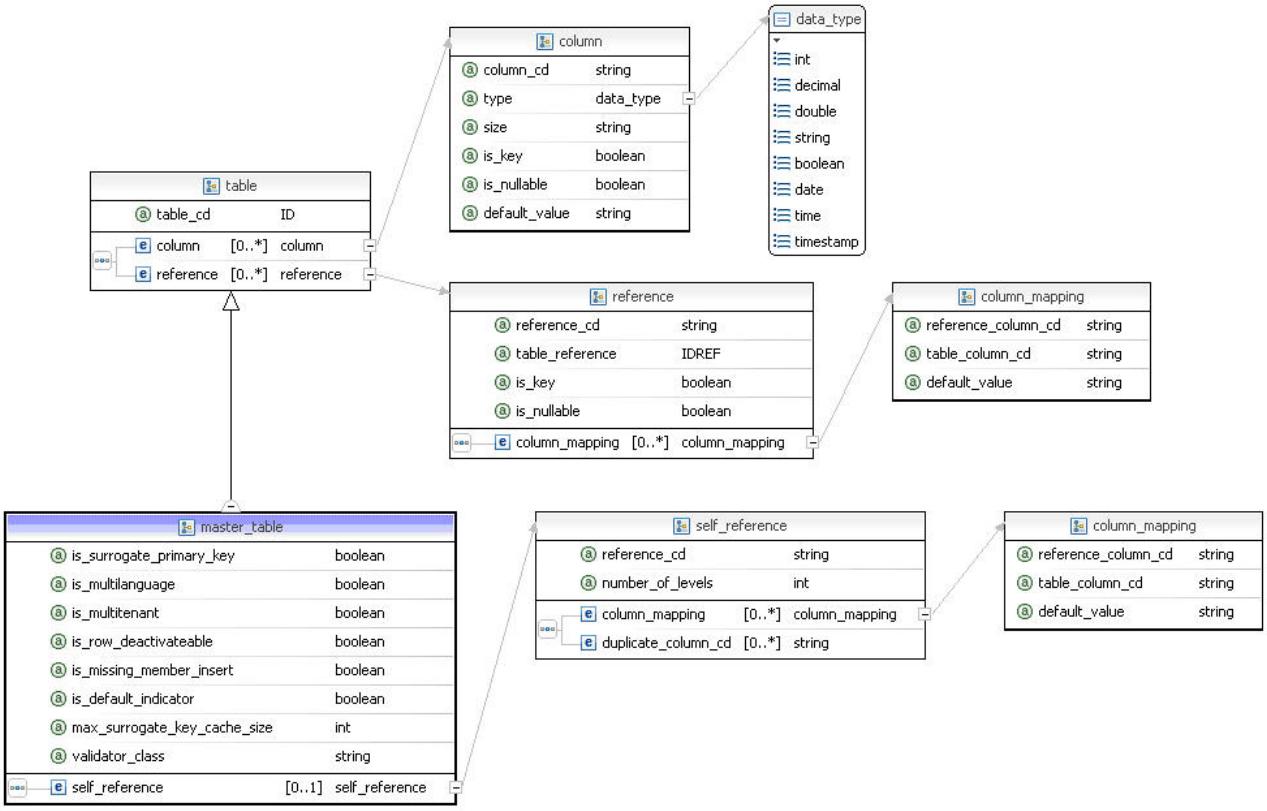


Figure 52. Master data definition schema

The following table lists the components in the master data definition. Use these components to create a master data definition in the solution definition file:

Table 40. Master data definition components

Type	Attribute or element	Description
table	table_cd	A unique code that identifies the table and will be used as the table name in the database. Required.

Table 40. Master data definition components (continued)

Type	Attribute or element	Description
table	is_surrogate_primary _key	An indicator to automatically create a column to hold an integer surrogate primary key for the table. The name of the column is the table_cd with "_ID" appended. This column is populated automatically. Optional. Default is false.
table	is_multilanguage	An indicator for adding a column to identify the language for a row. The name of the column is LANGUAGE_ID. Optional. Default is false.
table	is_multitenant	An indicator for adding a column to identify the tenant for a row. The name of the column is TENANT_ID. Optional. Default is false.
table	is_row_deactivateable	An indicator for adding an IS_ACTIVE column to identify if a row is still active. Optional. Default is false.
table	is_missing_member _insert	An indicator to automatically create a new row in the table when looking up a reference. If the row in this table being referenced does not exist, it will be created. Columns in the table that are not nullable have a default value of false.
table	max_surrogate_key_ cache_size	The maximum number of entries in the surrogate key cache for this table. Optional. Default is 1000.
table	validator_class	The fully qualified class name of a validator for rows in this table. The class must implement com.ibm.analytics.foundation.validation.Validation.
column	column_cd	A unique code in the table definition that identifies the column and will be used as the column name in the database. Required.
column	type	The data type of the column. One of int, decimal, double, string, boolean, date, time, or timestamp. Required.
column	size	Size for string (the number of characters) and decimal (the number of digits before and after the decimal. For example, 9,2). Optional. Default for string size is 50 and decimal is 9,2.
column	is_key	An indicator that this column is part of the business key of a row. Key columns may not have null values. Optional. Default is false. A master data table with isSurrogateKey = true can have a maximum of 5 keys.
column	is_nullable	An indicator that a non-key column may contain null values. Optional. Default is false.
column	default_value	A default value that is assigned to the column.
reference	reference_cd	A unique code in the table definition that identifies the reference. It is used as the name of the column in the database used to hold the foreign key reference.
reference	table_reference	The table_cd of the table being referenced. The table being referenced must have a surrogate primary key.

Table 40. Master data definition components (continued)

Type	Attribute or element	Description
reference	is_key	Indicates that this reference is a part of the business key of a row. Key references can not have null values. Optional. Default is false.
reference	is_nullable	Indicates that the column used for a non-key reference can contain null values. Optional. Default is false.
reference	column_mapping	Used when more than one column has the same identifier. It maps a custom column identifier for the reference to a column in the referenced table. A column mapping can also be used to provide a default value for the reference to a column.
self reference	reference_cd	A unique code that identifies the reference and will be used as the name of the column in the database used to hold the foreign key reference. The existence of a self_reference element indicates that a hierarchy table must be created for this master data table. The reference_cd indicates the name of the column that contains the reference to the parent of a given row.
self reference	number_of_levels	The number of levels in the hierarchy table created for the self reference.
self reference	column_mapping	This element is required to provide column names that differ from the key columns of the table. It maps a custom column identifier for the self reference to a column in the table.
self reference	duplicate_column_cd	A column in the table to duplicate into the hierarchy table. The column is duplicated at every level in the hierarchy.

The following XML code is an example of a master data definition:

```
<table table_cd="EMPLOYEE">
   <column column_cd="EMPLOYEE_CODE" is_key="true" type="string" size="10"/>
   <column column_cd="EMPLOYEE_NAME" type="string" size="10"/>
   <reference reference_cd="DEPARTMENT_ID" table_reference="DEPARTMENT"
is_nullable="false"/>
</table>
```

Profile definition

A profile definition defines key performance indicator (KPI) and profile tables. A profile defines how to aggregate observations from events. An example of a profile is the aggregation of values from observations which can be used as the input for a predictive score.

Use the profile definition components to define the KPI and profile tables that meet your business requirements. KPI tables are a type of profile table that includes an interval column. The values are aggregated over the interval. For example, daily.

The profile adapter updates profile tables using event data.

The following diagram shows the profile definition schema:

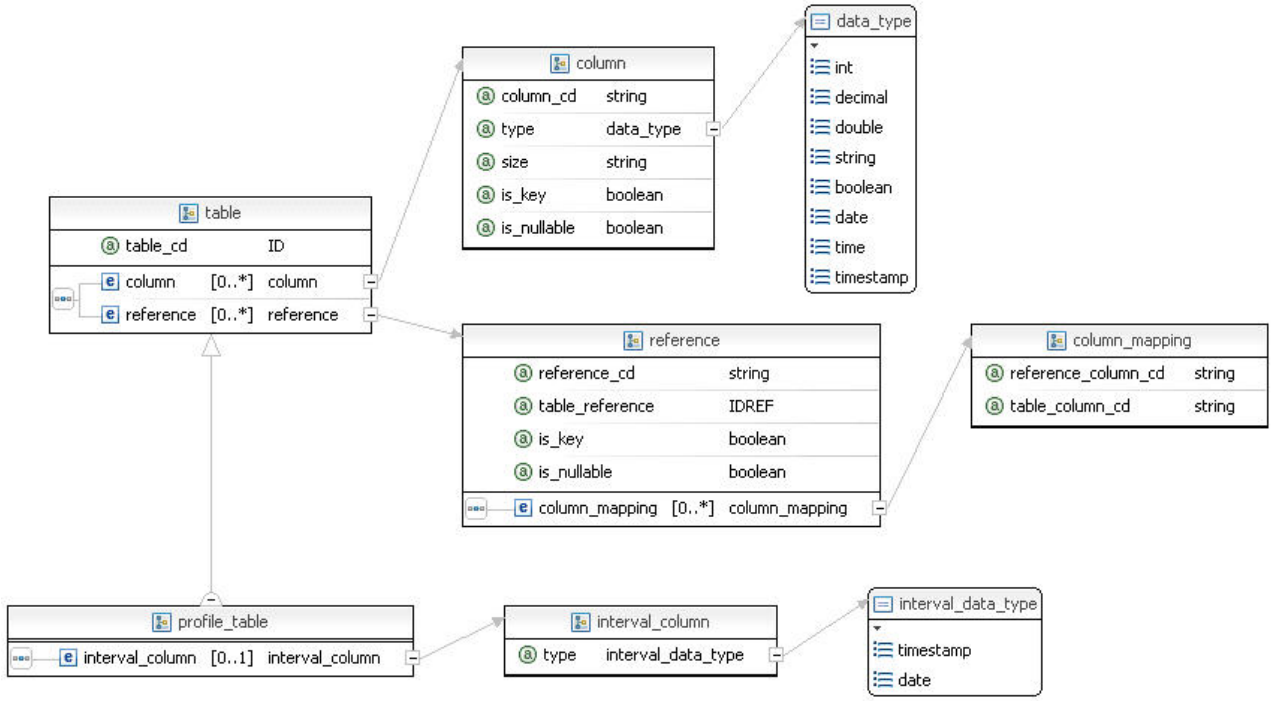


Figure 53. Profile definition schema

The following table lists the components in the profile definition. Use these components to create a profile definition in the solution definition file:

Table 41. Profile definition components

Type	Attribute or element	Description
table	table_cd	A unique code that identifies the table. It is used as the table name in the database. Required.
column	column_cd	A unique code that identifies the column. It is used as the column name in the database. Required.
column	type	The data type of the column. Must be one of int, decimal, double, string, boolean, date, time, or timestamp. Required.
column	size	For string, the number of characters in the string. For decimal, the number of digits before and after the decimal. Optional. Default for string size is 50 and decimal is 9,2.
column	is_key	An indicator that this column is part of the business key of a row. Key columns can not have null values. Optional. Default is false.
reference	reference_cd	A unique code that identifies the reference. It is used as the name of the column in the database that holds the foreign key reference. Required.
reference	table_reference	The table_cd of the table being referenced. The table being referenced must have a surrogate primary key. Required.

Table 41. Profile definition components (continued)

Type	Attribute or element	Description
reference	is_key	An indicator that this reference is a part of the business key of a row. Key references can not have null values. Optional. Default is false.
reference	column_mapping	A column mapping is used when more than one column has the same identifier. It maps a custom column identifier for the reference to a column in the referenced table.
interval column	column_cd	A unique code that identifies the column. It is used as the column name in the database. Required.
interval column	type	The data type of the column. Must be one of date, time, or timestamp. Required.

The following XML code is an example of a profile definition:

```
<profile_definition>
   <table table_cd="AUDIO_PROFILE">
      <column column_cd="PROFILE_DATE" is_nullable="false" type="timestamp"/>
      <column column_cd="LAST_PROFILE_DATE" type="timestamp"/>
      <column column_cd="PERIOD_AVERAGE" type="double"/>
      <column column_cd="PERIOD_MIN" type="double"/>
      <column column_cd="PERIOD_MAX" type="double"/>
      <reference reference_cd="AUDIO_ID" table_reference="AUDIO"
         is_nullable="false" is_key="true"/>
      <interval_column column_cd="KPI_DATE" type="date"/>
   </table>
</profile_definition>
```

Profile adapter

The profile adapter updates profile tables by using the event data.

A database connection must be present in the orchestration context The database connection is accessed by using the `OrchestrationEngineConstants.ORCHESTRATION_DATABASE_CONNECTION_KEY` key.

The configuration for this adapter is a list of profile updates that are run when an event is received. A profile update specifies when the list is triggered, the profile rows that are updated, and the calculations that perform the update.

Profile updates

There are two types of profile updates: event_profile_update and observation_profile_update.

event_profile_update is used when the event contains the data to update the profile. observation_profile_update is used when the event has observations that contain the data to update the profile.

Observation profile update

The observation profile update contains three components.

The following components are in the observation profile update:

- An event or observation selector that determines whether the update applies to a particular event or observation.
- One or more profile row selectors that determine the set of profile rows that are in the update.
- Calculation invocations that perform the update.

Event selector

An event selector determines whether the update applies to a particular event. The event selector contains a list of fields and values that can be used to select a particular event.

The following example shows a selector that matches an event row that has the EVENT_TYPE_CD value set to "MEASURMENT":

```
<event_selector><event_field_value>
    <field_name> EVENT_TYPE_CD </field_name>
    <value> MEASUREMENT </value>
</event_field_value></event_selector>
```

Observation selector

An observation_selector determines whether the update applies to a particular observation on an incoming event. The observation_selector specifies the table_cd of the observation. It can also specify a list of fields and values that can be used to select a particular observation or group of observations.

The following example shows a selector that selects an observation with table_cd = EVENT_OBSERVATION and value "RPM" for MEASUREMENT_TYPE_CD and the value "ACTUAL" for VALUE_TYPE_CD:

```
<observation_selector table_cd="EVENT_OBSERVATION">
    <observation_field_value>
        <field_name>MEASUREMENT_TYPE_CD</field_name>
        <value>RPM</value>
    </observation_field_value>
    <observation_field_value>
        <field_name>VALUE_TYPE_CD</field_name>
        <value>ACTUAL</value>
    </observation_field_value>
</observation_selector>
```

Profile row selector

A list of profile row selectors determines the profile rows that are updated. Each profile row selector specifies the table_cd or shared_selector_cd of the profile or shared_profile_row_selector and a list of key_field_value fields to specify a single profile row. A shared_profile_row_selector can be defined in the solution definition file and referenced by an observation profile update. Each key_field_value identifies a profile_field_name and a value for that field. The value can be a literal value or it can be determined by referring to a field in the event or the selected observation.

Normally all the selected profile rows are updated. If a profile row is selected to provide only input to a calculation, you must set an alias attribute to a unique value in the scope of the update.

A profile row selector can be shared between different updates. In the following example, profile_row_selector selects the profile rows in RESOURCE_KPI that have the value "OPHR Delta" for PROFILE_VARIABLE_CD:

```
<profile_row_selector>
    <shared_selector_cd>RESOURCE_KPI</shared_selector_cd>
    <key_field_value>
        <profile_field_name>PROFILE_VARIABLE_CD</profile_field_name>
    <value>OPHR Delta</value>
    </key_field_value>
</profile_row_selector>
```

Read only profile row selector

A list of read only profile row selectors determines the profile rows that are read. Each read only profile row selector specifies the table_cd or shared_selector_cd of the profile and a list of key_field_value fields to specify a single profile row. Each key_field_value identifies a profile_field_name and a value for that field. The value can be a literal value or it can be determined by referring to a field in the event or the selected observation.

A read only profile row is selected to provide only input to a calculation. The following read only selector selects the rows in RESOURCE_PROFILE that have the value of "OPHR Delta" for PROFILE_VARIABLE_CD:

```
<read_only_profile_row_selector alias="RESOURCE_PROFILE_LAST_VALUE">
    <shared_selector_cd>RESOURCE_PROFILE</shared_selector_cd>
    <key_field_value>
        <profile_field_name>PROFILE_VARIABLE_CD</profile_field_name>
        <value>OPHR Delta</value>
    </key_field_value>
</read_only_profile_row_selector>
```

Calculation invocation

A list of calculation_invocations updates the values of a profile row. Each invocation specifies a calculation_cd to identify the calculation to perform. An input_field_value identifies the field_name of a calculation input and a value for that field. The value can be a literal value or it can be determined by referring to a field in the event, the selected observation, or another observation. The value can also be obtained from a field in the profile row that is being updated, another selected profile row (by using the alias for that row), or from an orchestration context variable. The following value references are examples:

```
<value_ref>
    <event_field_name>RPM</event_field_name>
</value_ref>

<value_ref>
    <context_variable_name>PROFILE_VARIABLE_CD</context_variable_name>
</value_ref>

<value_ref>
    <selected_observation_field_name>OBSERVATION_TIMESTAMP
</selected_observation_field_name>
</value_ref>

<value_ref>
    <observation_field>
        <table_cd>EVENT_OBSERVATION</table_cd>
        <key_field_value>
            <field_name>MEASUREMENT_TYPE_CD</field_name>
            <value>TEMP</value>
        </key_field_value>
        <field_name>VALUE_TYPE_CD</field_name>
    </observation_field>
</value_ref>
```

The update_field_value specifies the fields to update after a calculation is performed. A field in the profile row or an orchestration context variable can be updated. The value comes from one of the output fields of the calculation. For example, the following update_field_value specifies the FORECAST_VALUE row in the profile table to be updated:

```
<update_field_value>
    <profile_field_name>FORECAST_VALUE</profile_field_name>
    <value>1.99</value>
</update_field_value>
```

A list of calculation_invocations can be grouped and shared for reuse. A shared_calculation_invocation_group can be defined in the solution definition file and referenced by an observation profile update.

Event profile update

The configuration of an event _profile update is similar to the configuration of an observation profile_update.

The event profile update is used when all of the data to update a profile row is part of the event itself rather than in an observation. The following example shows an event profile update:

```
<event_profile_update>
    <event_selector>
        <event_field_value>
            <field_name>RPM</field_name>
            <value>*</value>
        </event_field_value>
    </event_selector>
    <profile_update_action>
        <profile_row_selector>
        <shared_selector_cd>RESOURCE_KPI_INLINE</shared_selector_cd>
            <key_field_value>
                <profile_field_name>PROFILE_VARIABLE_CD</profile_field_name>
                <value>RPM</value>
            </key_field_value>
        </profile_row_selector>
<shared_calculation_invocation_group_cd>KPI.MEASUREMENT_ABOVE_LIMIT.ACTUAL.INLINE
</shared_calculation_invocation_group_cd>
        <calculation_invocation_invocation_cd=
"KPI.MEASUREMENT_ABOVE_LIMIT.ACTUAL.VALUE">
             <input_field_value>
                 <field_name>MEASURE_VALUE</field_name>
                    <value_ref>
                        <event_field_name>RPM</event_field_name>
                    </value_ref>
             </input_field_value>
             <input_field_value>
                  <field_name>THRESHOLD</field_name>
                  <value>100</value>
             </input_field_value>
         </calculation_invocation>
    </profile_update_action>
</event_profile_update>
```

Type conversions

Normally values are obtained from fields of the same type. However, IBM Predictive Maintenance and Quality provides some data type conversions.

Predictive Maintenance and Quality provides the following automatic data type conversions:

Table 42. Type conversions

Data type	Converted to
int	decimal, double, string, Boolean
decimal	int, double, string
double	int, decimal, string
string	int, decimal, double, Boolean, date, time, timestamp
Boolean	int, string
date	string, timestamp
time	string
timestamp	string, date, time

Data mapping

Data is mapped to various settings when the profile adapter runs.

Profile row selection

The profile row selection provides the input to select a profile row. The event and any observations are available. If a specific observation was selected for the update, then that observation is available for input.

The following mappings apply to the profile row selection setting:

Destination

profile field

Source

literal

event field

selected observation field (only valid in an observation profile update)

other observation field (must select an observation)

Input to calculation

This setting provides the required input to the calculation. The event and any observations are available. If a specific observation was selected for the update, then that observation is available for input. Selection of the observation is not required.

The following mappings apply to the input to calculation setting:

Destination

calculation input field

Source

literal

event field

selected observation field (only valid in an observation_profile_update)

other observation field (must select an observation)

profile field

other profile field (using an alias)

context variable

Update after calculation

This setting uses the results of the calculation to update the profile row or a context variable. The output of the calculation is available.

The following mappings apply to the update after calculation setting:

Destination

profile field

context variable

Source

literal

calculation output field

Input to service

This setting provides the input for the service call. The incoming event and the selected profile rows are available, along with any context variables.

The following mappings apply to the input to service setting:

Destination

service input field

Source

literal

event field

other observation field (must select an observation)

profile field

other profile field (using an alias)

context variable

Update after service

This setting manages the results of the service call. A service event in the context can be used to hold the result. Results can also be recorded directly in the orchestration context.

The output of the service call is available. The incoming event is also available, along with any observations it contains. All of the master data references can be copied from the incoming event to the service event. A shared update group can be used to do the copy.

The following mappings apply to the update after service setting:

Destination

event field

observation field (must specify the observation)

context variable

Source

literal

event field

service output field

incoming event field

Service adapter

The service adapter calls a service and optionally updates an event with the result of the service.

The configuration for this adapter provides a list of service invocations. A service invocation specifies when the list is triggered, the inputs to use for the service, and how to process the result of the event.

Service invocation configuration

The event selector determines if a particular event should trigger a service invocation action.

The following code is an example of how the event selector is used:

```
<event_selector>
   <event_field_value>
      <field_name>EVENT_TYPE_CD</field_name>
 <value>MEASUREMENT</value>
   </event_field_value>
</event_selector>
```

Service profile row selector

A list of service_profile_row_selectors determines what profile rows will be selected as inputs for the service. Each service_profile_row_selector specifies the table_cd or shared_selector_cd and a list of key_field_value to specify a single profile row. Each key_field_value identifies a profile_field_name and a value for that field. The value may be a literal value or it may be determined by referring to a field in the event or an observation.

Because the profile rows are being selected from multiple profile tables and are not being used for updates, it is required that all the profile_row_selectors provide an alias. This alias is used when defining an input for the service.

A profile_row_selector can be shared rather than defining the same selector many times for various updates. A shared_profile_row_selector can be defined in the solution definition file and referenced by a service_profile_row_selector.

```
<profile_row_selector alias="RPM">
   <shared_selector_cd>RESOURCE_KPI</shared_selector_cd>
   <key_field_value>
      <profile_field_name>PROFILE_VARIABLE_CD</profile_field_name>
 <value>RPM</value>
   </key_field_value>
</profile_row_selector>
```

Service invocation

A service invocation determines the input values for the service. The input_field_value is a list of input fields that can be either values or field references from a profile, observation, or event row

The following example shows a service invocation:

```
<input_field_value>
   <field_name>RPM_BELOW_LIMIT</field_name>
   <value_ref>
```

```
      <profile_field>
         <alias>RPM_BELOW_LIMIT</alias>
  <field_name>ACTUAL_VALUE</field_name>
      </profile_field>
   </value_ref>
</input_field_value>
<input_field_value>
   <field_name>CURRENT_VALUE</field_name>
   <value_ref>
      <observation_field>
         <table_cd>EVENT_OBSERVATION</table_cd>
         <key_field_value>
     <field_name>MEASUREMENT_TYPE_CD</field_name>
     <value>RPM</value>
  </key_field_value>
     <field_name>MEASUREMENT</field_name>
      </observation_field>
   </value_ref>
</input_field_value>
```

The update_field_value are a list of fields from either the service event or the context_variable_name that will be updated by the service output.

```
<update_field_value>
   <event_field_name>EVENT_START_TIME</event_field_name>
   <value_ref>
      <service_output_field_name>CURRENT_TIMESTAMP</service_output_field_name>
   </value_ref>
</update_field_value>
<update_field_value>
   <observation_field>
      <table_cd>EVENT_OBSERVATION</table_cd>
      <key_field_value>
         <field_name>MEASUREMENT_TYPE_CD</field_name>
  <value>HS</value>
      </key_field_value>
      <field_name>MEASUREMENT</field_name>
   </observation_field>
   <value_ref>
      <service_output_field_name>SCORE</service_output_field_name>
   </value_ref>
</update_field_value>
```

Service invocation handler

The service invocation handler calls the service. The role of the adapter is to provide the required input for this call and to store the result.

In IBM Integration Bus, the orchestration engine can be called from a JavaCompute node to perform an orchestration that includes a call to a web service. IBM Integration Bus can perform the call to the web service with a dedicated node.

The following steps describe how the orchestration engine can use capabilities of IBM Integration Bus to perform the web service call:

1. The orchestration node is a JavaCompute node that performs the call to the `processEvent()` method of the orchestration engine.

   The process event method takes an event and a map of orchestration context values. Each of the values is associated with a key that identifies how it will be used by the orchestration. For example, a key of *OrchestrationEngineConstants.ORCHESTRATION_DATABASE_CONNECTION_KEY* supplies a JDBC database connection.

2. The orchestration includes an instance of ServiceInvocationHandler (typically an anonymous inner class) as one of the orchestration context values.

   The ServiceInvocationHandler implements the invokeService method to perform the web service call.
3. In IBM Integration Bus, the ServiceInvocationHandler creates the message structure and explicitly propagates it to the SOAP request node.
4. When the SOAP request returns, the Process Result node records the result in the IBM Integration Bus execution context.
5. Control returns to the ServiceInvocationHandler. The ServiceInvocationHandler retrieves the results from the IBM Integration Bus execution context and returns them as the result of the invokeService method.

Handlers and scoring events

The attributes of the service_invocation element in an orchestration definition are described in this section.

service_cd

The attribute service_cd identifies the service being invoked. Services are defined in the solution definition file.

handler_context_variable_name

This is the name used to access the service invocation handler in the orchestration context. Clients must populate the orchestration context with this handler and pass it to the processEvent API. The handler must implement the ServiceInvocationHandler interface. The example shows the declaration of a service invocation handler and a reference to it in a service invocation element:

```
// Create and populate the context
Map<String, Object> map = new HashMap<String, Object>();
map.put(OrchestrationEngineConstants.ORCHESTRATION_DATABASE_CONNECTION_KEY,
dbConnect.getConnection());
map.put("RPM_SCORE", new ServiceInvocationHandler() {
     @Override
     public void invokeService(ServiceRow input,ServiceRow output,
Map<String,String>configProperties) throws ServiceInvocationException {
          try {
             // Demo implementation - a real implementation would use the input
             // and call scoring
             output.setTimestamp("CURRENT_TIMESTAMP",
new Timestamp(System.currentTimeMillis()));
             output.setDouble("SCORE", 10.0);
          } catch (FieldAccessException e) {
           throw new ServiceInvocationException("Exception setting score value",e);
          }
     }
});
engine.processEvent(eventRow, map);

<service_invocation service_cd="SPSS" handler_context_variable_name="RPM_SCORE">
```

event_table_cd

The result of a service invocation can be stored in an event by the service adapter. The service adapter assigns values to the event after scoring by specifying the event_table_cd attribute.

event_context_variable_name

This name accesses the event used to hold the result of scoring in the orchestration context. Clients must populate the context using this name with an event row matching the `event_table_cd`.

The following example shows supplying an event and a reference to it in a service invocation element:

```
// Create and populate the context
Map<String, Object>map = new HashMap<String,Object>();
map.put(OrchestrationEngineConstants.ORCHESTRATION_DATABASE_CONNECTION_KEY,
dbConnect.getConnection());
EventRow scoreEvent = eventManager.prepareEventRow("EVENT");
EventObservationRow observation = eventManager.prepareEventObservationRow
("EVENT_OBSERVATION");
scoreEvent.addObservation(observation);
observation.setString("MEASUREMENT_TYPE_CD","HS");
scoreEvent.setOrchestrationKeyCd(score_event_orchestration_key_cd);
map.put("SCORE_EVENT",scoreEvent);
...
engine.processEvent(eventRow,map);

<service_invocation service_cd="SPSS" handler_context_variable_name=
"RPM_SCORE"
 event_table_cd="EVENT" event_context_variable_name="SCORE_EVENT">
```

event_handler_context_variable_name

This is the name used to access the event handler in the orchestration context. The event handler is used to process the event that holds the result of a service invocation. Clients are responsible for populating the orchestration context with this handler and passing it to the `processEvent` API. The handler must implement the `ServiceEventHandler` interface. Normally this handler is only used if the event being processed has a content provider. Using this handler allows the service event to be processed after each scoring invocation.

The following is an example of the declaration of an event handler and a reference to it in a service invocation element:

```
map.put("SCORE_EVENT_HANDLER", new ServiceEventHandler() {
     @Override
     public void handleEvent(EventRow scoreEvent,
Map<String,Object>orchestrationContext) throws ServiceEventException {
     try {
          engine.processEvent(scoreEvent,orchestrationContext);
     } catch (OrchestrationEngineException e) {
          throw new ServiceEventException("Exception processing
score value",e);
     }
  }
});

<service_invocation service_cd="SPSS" handler_context_variable_name="RPM_SCORE"
 event_table_cd="EVENT" event_context_variable_name=
"SCORE_EVENT" event_handler_context_variable_name="SCORE_EVENT_HANDLER">
```

Event definition

An event definition defines the tables that store event data.

The following types of events are processed in a typical orchestration:

- Events generated by a device.
- Events generated by the orchestration engine to record the results of scoring and recommendations.

Use the event definition components to define the event tables that meet your business requirements.

The event store adapter is added to an orchestration to insert the event being processed into the database. The adapter stores the event and any observations that the event contains. The event store adapter automatically stores all the events that it receives.

The following diagram shows the event definition schema:

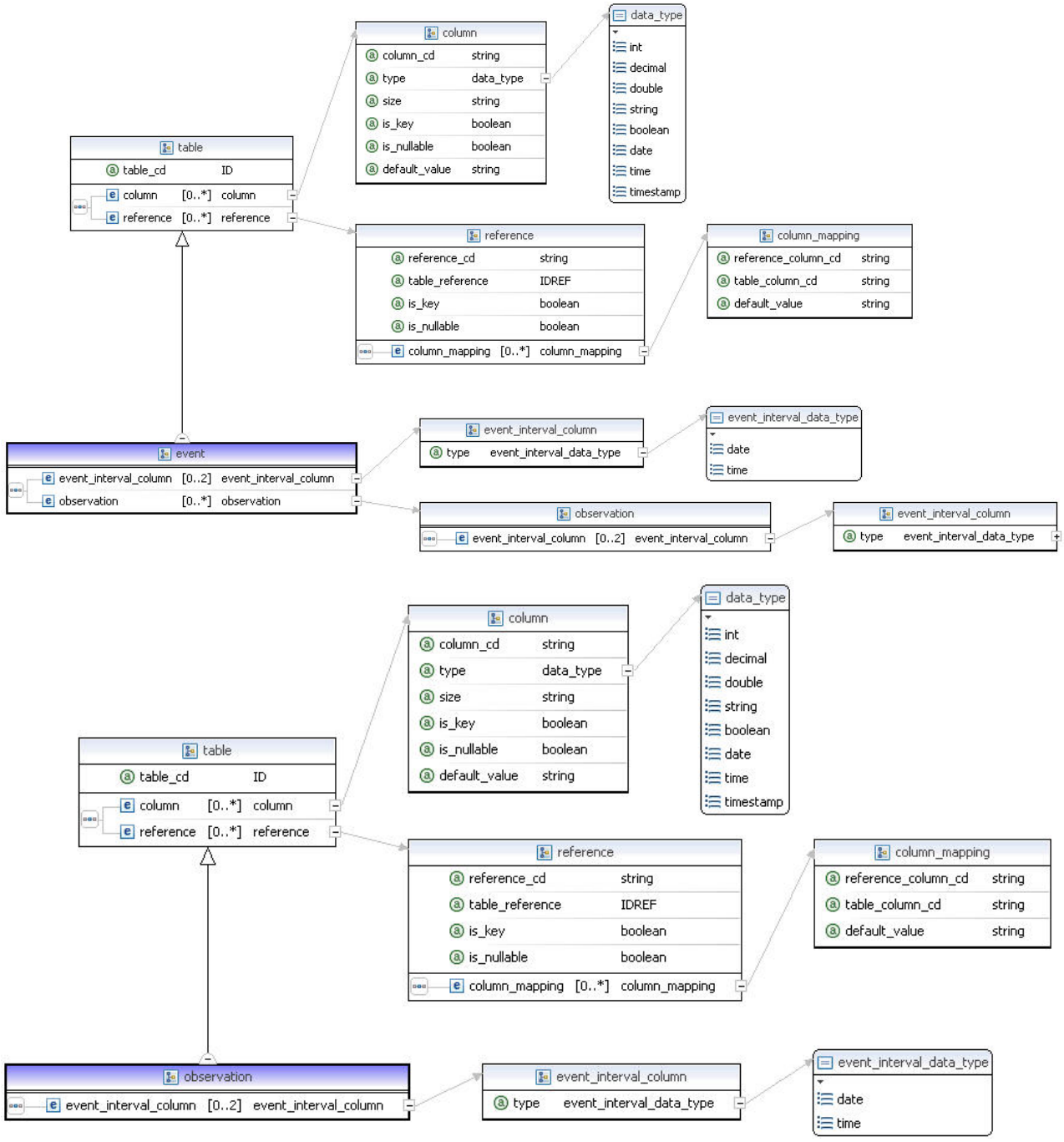


Figure 54. Event definition schema

The following table lists the components in the event definition. Use these components to create an event definition in the solution definition file:

Table 43. Event definition components

Type	Attribute or element	Description
table	table_cd	A unique code that identifies the table. It is used as the table name in the database. Required.

Table 43. Event definition components (continued)

Type	Attribute or element	Description
column	column_cd	A unique code that identifies the column. It is used as the column name in the database. Required.
column	type	The data type of the column. Must be one of int, decimal, double, string, boolean, date, time, or timestamp. Required.
column	size	For string, the number of characters in the string. For decimal, the number of digits before and after the decimal. Optional. Default for string size is 50 and decimal is 9,2.
column	is_key	An indicator that this column is part of the business key of a row. Key columns can not have null values. Optional. Default is false.
reference	reference_cd	A unique code that identifies the reference. It is used as the name of the column in the database that holds the foreign key reference.
reference	table_reference	The table_cd of the table being referenced. The table being referenced must have a surrogate primary key.
reference	is_key	An indicator that this reference is a part of the business key of a row. Key references can not have null values. Optional. Default is false.
reference	column_mapping	A column mapping is used when more than one column has the same identifier. It maps a custom column identifier for the reference to a column in the referenced table. Define a default value.
event interval column	column_cd	A unique code that identifies the column. It is used as the column name in the database. Required.
event interval column	type	The data type of the column. Must be one of date or time. Required.
observation	table_cd	A unique code that identifies the observation table. It is used as the table name in the database. Required.
observation	column	A unique code that identifies the observation column. It is used as the column name in the database. Required.
observation	reference	A unique code that identifies the reference. It is used as the reference name in the database. Required.
observation	event_interval_ column	A unique code that identifies the reference. It is used as the event interval column name in the database. Required.

The following XML code is an example of an event definition:

```
<event_definition>
   <table table_cd="EVENT">
      <column column_cd="EVENT_START_TIME" type="timestamp"/>
   <column column_cd="EVENT_END_TIME" type="timestamp"
    is_nullable="true"/>
```

```
   <column column_cd="EVENT_PLANNED_END_TIME" type="timestamp"
    is_nullable="true"/>
   <column column_cd="INCOMING_EVENT_CD" type="string" size="200"
    is_nullable="true"/>
   <reference reference_cd="ASSET_ID" table_reference="MASTER_RESOURCE">
     <column_mapping reference_column_cd="SERIAL_NO"
    table_column_cd="RESOURCE_CD1" default_value="-NA-"/>
     <column_mapping reference_column_cd="MODEL"
    table_column_cd="RESOURCE_CD2" default_value="-NA-"/>
   </reference>
   <reference reference_cd="AGENT_ID" table_reference="MASTER_RESOURCE">
    <column_mapping reference_column_cd="OPERATOR_CD"
     table_column_cd="RESOURCE_CD1" default_value="-NA-"/>
    <column_mapping reference_column_cd="OPERATOR_NA"
     table_column_cd="RESOURCE_CD2" default_value="-NA-"/>
   </reference>
   <reference reference_cd="EVENT_TYPE_ID" table_reference="MASTER_EVENT_TYPE"/>
   <reference reference_cd="SOURCE_SYSTEM_ID" table_reference=
"MASTER_SOURCE_SYSTEM">
     <column_mapping table_column_cd="SOURCE_SYSTEM_CD" default_value="-NA-"/>
   </reference>
   <reference reference_cd="PROCESS_ID" table_reference="MASTER_PROCESS">
  <column_mapping table_column_cd="PROCESS_CD" default_value="-NA-"/>
   </reference>
   <reference reference_cd="PRODUCTION_BATCH_ID" table_reference=
"MASTER_PRODUCTION_BATCH">
  <column_mapping table_column_cd="PRODUCTION_BATCH_CD" default_value="-NA-"/>
   </reference>
   <reference reference_cd="LOCATION_ID" table_reference="MASTER_LOCATION">
  <column_mapping table_column_cd="LOCATION_CD" default_value="-NA-"/>
   </reference>
      <observation table_cd="EVENT_OBSERVATION">
     <column column_cd="OBSERVATION_TIMESTAMP" is_key="true" type="timestamp"/>
  <column column_cd="OBSERVATION_TEXT" type="string" size="800"
     is_nullable="true" />
  <column column_cd="MEASUREMENT" type="double" is_nullable="true"/>
  <reference reference_cd="MEASUREMENT_TYPE_ID" is_key="true"
     table_reference="MASTER_MEASUREMENT_TYPE"/>
  <reference reference_cd="VALUE_TYPE_ID" is_key="true"
     table_reference="MASTER_VALUE_TYPE">
    <column_mapping table_column_cd="VALUE_TYPE_CD" default_value="ACTUAL"/>
  </reference>
  <reference reference_cd="EVENT_CODE_ID" is_key="true"
     table_reference="MASTER_EVENT_CODE">
    <column_mapping table_column_cd="EVENT_CODE" default_value="-NA-"/>
  </reference>
  <reference reference_cd="MATERIAL_ID" table_reference="MASTER_MATERIAL">
    <column_mapping table_column_cd="MATERIAL_CD" default_value="-NA-"/>
  </reference>
  <event_interval_column column_cd="OBSERVATION_DATE"
     type="date"/>
  <event_interval_column column_cd="OBSERVATION_TIME"
     type="time"/>
      </observation>
   </table>
</event_definition>
```

Solution definition file

The solution definition file contains common elements that are referenced by the orchestration files.

Calculation definition

A calculation definition defines the input and output of a calculation. The profile adapter uses a calculation to update values in a profile row.

A calculation must implement the calculate method. This method returns true if the result of the calculation should be used to update a profile row and returns false otherwise. Returning false provides faster performance by avoiding an update to a profile row that does not result in a change.

The following table shows the components in the calculation definition. Use these components to create a calculation definition in the solution definition file:

Table 44. Components in a calculation definition

Attribute	Description
calculation_cd	Uniquely identifies the calculation.
is_increment	Indicates if the calculation can be performed without having any current values. For example, the calculations *Count* and *Total* should have is_increment set to true.
calculation_class	Provides the fully qualified name of the class that implements the calculation. This class must implement the com.ibm.analytics.foundation.calculation.api.Calculation interface.

The following XML code is an example of a calculation definition:

```
<calculation calculation_cd="MINIMUM"
         calculation_class="com.ibm.analytics.foundation.calculation.calculations.Minimum"
         is_increment="false">
         <input>
                <field_name>MEASURE_VALUE</field_name>
                <type>double</type>
         </input>
         <input>
                <field_name>CURRENT_MIN</field_name>
                <type>double</type>
         </input>
         <output>
                <field_name>UPDATED_MIN</field_name>
                <type>double</type>
         </output>
</calculation>
```

Service definition

A service definition defines the input and output of a service.

The service adapter uses a service definition to perform a call to the service. The definition of a service includes the service_cd attribute which uniquely identifies the service. Create a service definition in the solution definition file.

The following XML code is an example of a service definition:

```
<service_definition>
 <service service_cd="RPM" >
  <input>
   <field_name>RPM</field_name>
   <type>double</type>
  </input>
  <input>
   <field_name>RPM_ABOVE_LIMIT</field_name>
   <type>int</type>
  </input>
  <input>
   <field_name>RPM_BELOW_LIMIT</field_name>
   <type>int</type>
```

```
  </input>
  <output>
   <field_name>SCORE</field_name>
   <type>double</type>
  </output>
 </service>
</service_definition>
```

Modifying the Analytics Solutions Foundation data model

The `solution.xml file` contains definitions for master data, profiles, and events. These definitions represent the data model in the IBM Predictive Maintenance and Quality solution. You can create or modify the database tables in the data model by adding definitions to or modifying the definition in this file.

About this task

The master data manager uses `solution.xml` to generate SQL scripts to create the tables. The scripts create the master data, profile, and event tables in the database. The `solution.xml` file is in the *installation_location*`/var/mqsi/shared-classes` directory.

Procedure

1. Open `solution.xml` in an XML editor.
2. Add or modify the definitions as required.
3. Save the file.
4. Run the following command to generate the DDL file:
   `MasterDDLGenerator.java <solution.xml_path> <ddl_file_output_path>`
5. Run the following command to update the database:
   `db2 -tvf <ddl_file_output_path>`

Other databases

IBM Analytics Solutions Foundation is configured for IBM DB2 but can be configured to support other databases.

To configure Analytics Solutions Foundation to support a database other than IBM DB2, you must customize the following artifacts:

`sql.properties` file

The `sql.properties` file contains vendor-specific SQL that is used in IBM Analytics Solutions Foundation. Use the `db2.sql.properties` file as a template and replace the DB2 SQL with SQL specific to your database.

`xml2ddl_transformer` specification

The `xml2ddl_transformer` XSL transform specification generates the DDL script to create the Analytics Solutions Foundation database tables. Use the `xml2ddl_transformer.xsl` as a template, and modify the XSL specification to generate DDL specific to your database. The output DDL syntax includes the following commands and concepts:

- CREATE TABLE
- ALTERNATE TABLE to add primary key and unique key constraints

- ALTERNATE TABLE to add foreign key constraints
- The concept of an identity column

`naRowInserts` specification

The `naRowInserts` XSL transform specification generates a stored procedure that populates the Master Data tables with NA rows. The output creates a `New_NA_LG` stored procedure.

Stored procedures

The `populate_calendar_and_event_time.sql` creates the `Calendar_pop` stored procedure to load the `Calendar` and `Event_Time` support tables. You must modify the stored procedure syntax for your database.

Add custom XSL transform specifications to the class path for Analytics Solutions Foundation to use. You can add the `sql.properties` to the class path or you can use the `dbPropFile` system property.

System properties

Set the following system properties so that Analytics Solutions Foundation can locate the `sql.properties` file:

dbVendor

Is a prefix that identifies the `sql.properties` file for your database. For example, set dbVendor system property to "ora" for Analytics Solutions Foundation to use `ora.sql.properties` from the class path.

dbPropFile

Is an absolute path to the custom `sql.properties` file. It is an alternative to having an `.sql.properties` name prefixed with `dbVendor` system property in the class path.

Appendix C. 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 `IS_ACTIVE` indicator is used to mark records as no longer in use `(IS_ACTIVE = 0)`.

The `IS_ACTIVE` 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: `IS_ACTIVE=0`, that resource is loaded and associated with that location. Similarly if the event is reported by the resource with `IS_ACTIVE=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 45. 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) **Note:** No circular relationship is allowed. That is, a process_code cannot be a parent to itself.
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 46. 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 47. 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.LANGUAGE L ON
M.LANGUAGE_ID = L.LANGUAGE_ID JOIN SYSREC.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 48. 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 48. 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.LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID
JOIN SYSREC.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 49. Fields in the **language** *table*

Field	Type	Comments
language_cd	string(50)	Required. For example, EN
language_name	string(200)	Required. For example, English.
DEFAULT_IND	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.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 50. 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.
IS_ACTIVE	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.LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.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 51. 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.
IS_ACTIVE	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.LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.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 52. 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.LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.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 53. 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.LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.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 54. 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.
IS_ACTIVE	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.LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.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 55. 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
product_type_cd	string(50)	Required

Table 55. Fields in the **production_batch** *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.

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.LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.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 56. 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.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 57. 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

Table 57. Fields in the **resource** *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
IS_ACTIVE	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.LANGUAGE
L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN SYSREC.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 58. 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.
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.LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.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 59. 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.
IS_ACTIVE	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.LANGUAGE L ON
M.LANGUAGE_ID = L.LANGUAGE_ID JOIN SYSREC.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 60. 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.
IS_ACTIVE	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.LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.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 61. Fields in the **tenant** *table*

Field	Type	Comments
tenant_cd	string(50)	Required.
tenant_name	string(200)	Required.
DEFAULT_IND	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.TENANT;
```

For information on adding new languages and tenants, see the following information: "New languages and tenants" on page 152.

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 62. 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.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 records are supported by the metadata section of the application programming interface (API). The records 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 63. 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.LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.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 64. 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.LANGUAGE L ON M.LANGUAGE_ID = L.LANGUAGE_ID JOIN
SYSREC.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 65. 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

Table 65. Fields in the **profile_variable** *table (continued)*

Field	Type	Comments
profile_units	string(100)	Optional
comparison_string	string(200)	Optional
low_value_date	datetime	Optional
high_value_date	datetime	Optional
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.LANGUAGE L ON M.LANGUAGE_ID =
L.LANGUAGE_ID JOIN SYSREC.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.

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, the following profile variables and corresponding measurement types must be defined for the field `profile_variable_cd`:

- `AC`
- `ATIME`
- `CELLLDX`
- `CELLLDXX`
- `CLTX`
- `CLTXX`
- `FAIL`
- `HS`
- `INSP`
- `ITIME`
- `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.

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 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)

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.LANGUAGE L ON PB1.LANGUAGE_ID = L.LANGUAGE_ID
JOIN SYSREC.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.LANGUAGE L ON PB2.LANGUAGE_ID = L.LANGUAGE_ID
JOIN SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.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.LANGUAGE L WHERE
 L.LANGUAGE_CD = 'EN') AND
M.TENANT_ID = (SELECT T.TENANT_ID FROM SYSREC.TENANT T WHERE
 T.TENANT_CD = 'PMQ');
```

Note:

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

Appendix D. 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*. These documents are available at IBM Cognos Business Intelligence Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSEP7J).

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. The database layer also contains alias shortcuts, which behave as if they were a copy of the original object with completely independent behavior.

The alias shortcuts 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. Language is based on the used locale settings. Localization is implemented for the FM model as well. Users can select the language of their choice from the Active Language drop down menu and change the model 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.

A parameter map for language supports the localization of report data. Language codes for English (EN), Simplified Chinese (SC), Traditional Chinese (TC), French (FR), Japanese (JP), and Portuguese (Brazil)(PT) are configured in the parameter map.

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 following diagram shows the star schema for the `event_observation` table.

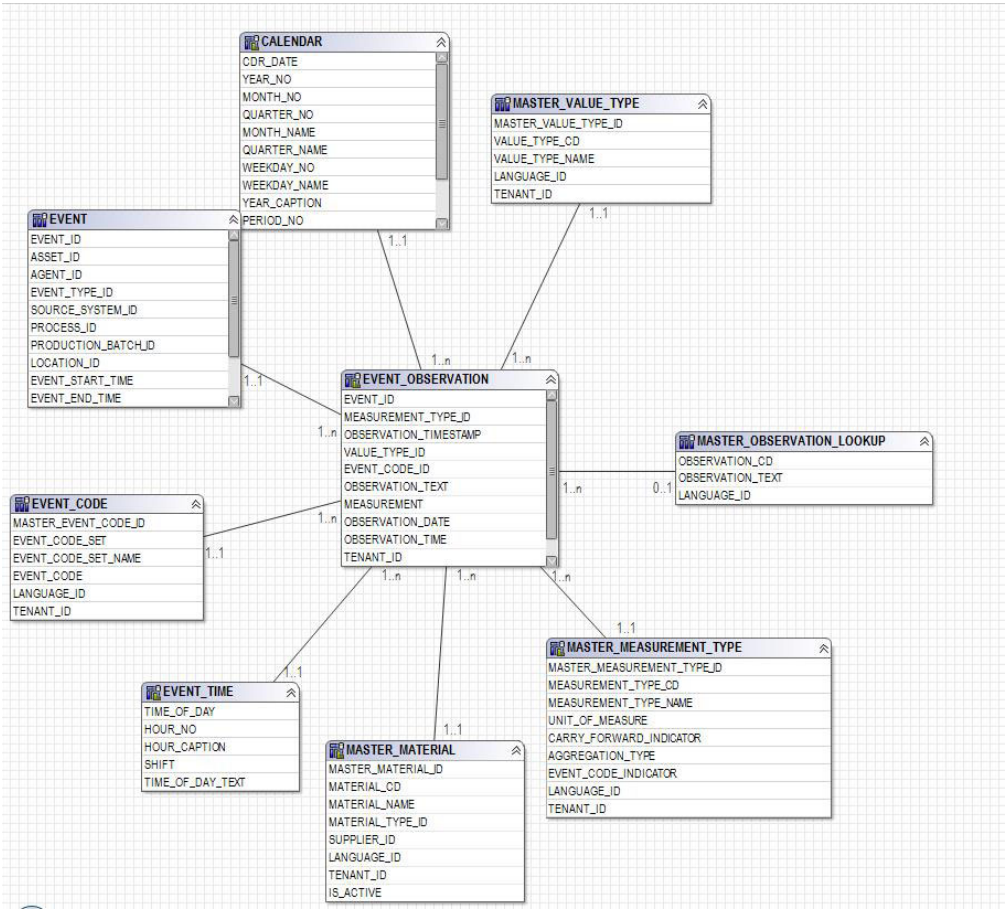


Figure 55. The `event_observation` star schema

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

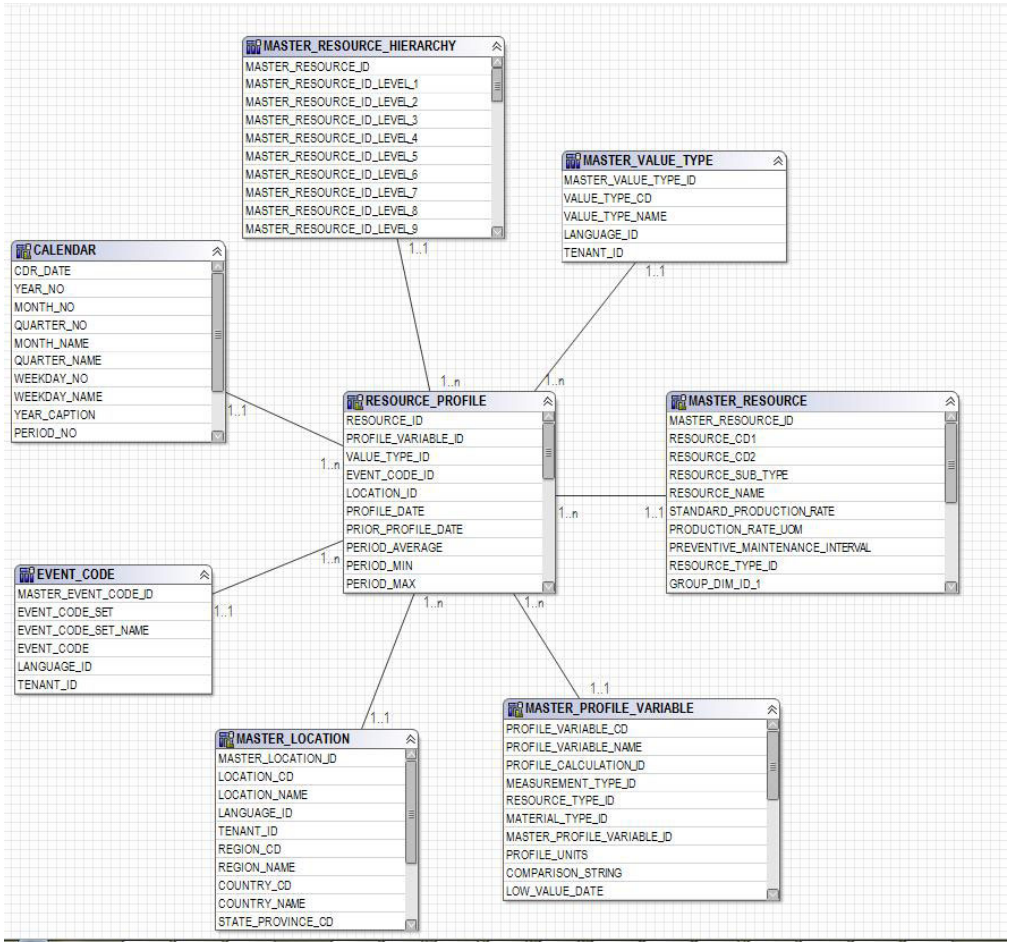


Figure 56. The resource_profile star schema

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

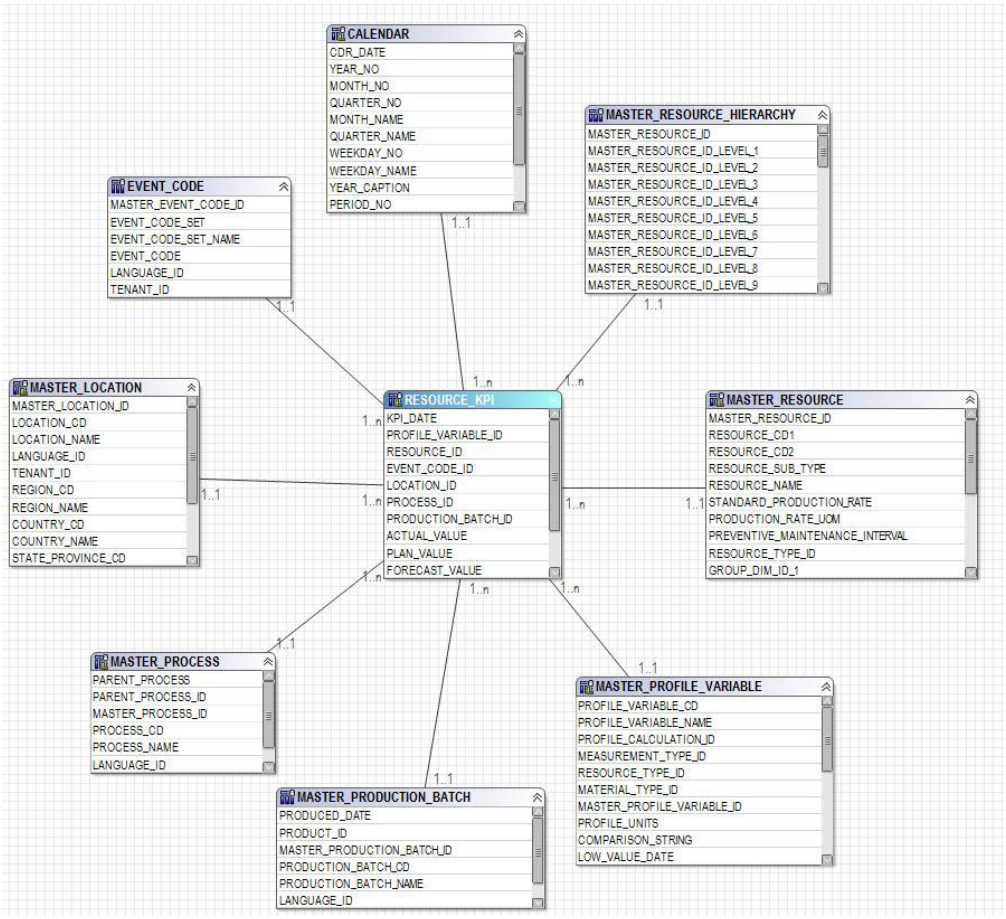


Figure 57. The resource_kpi star schema

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

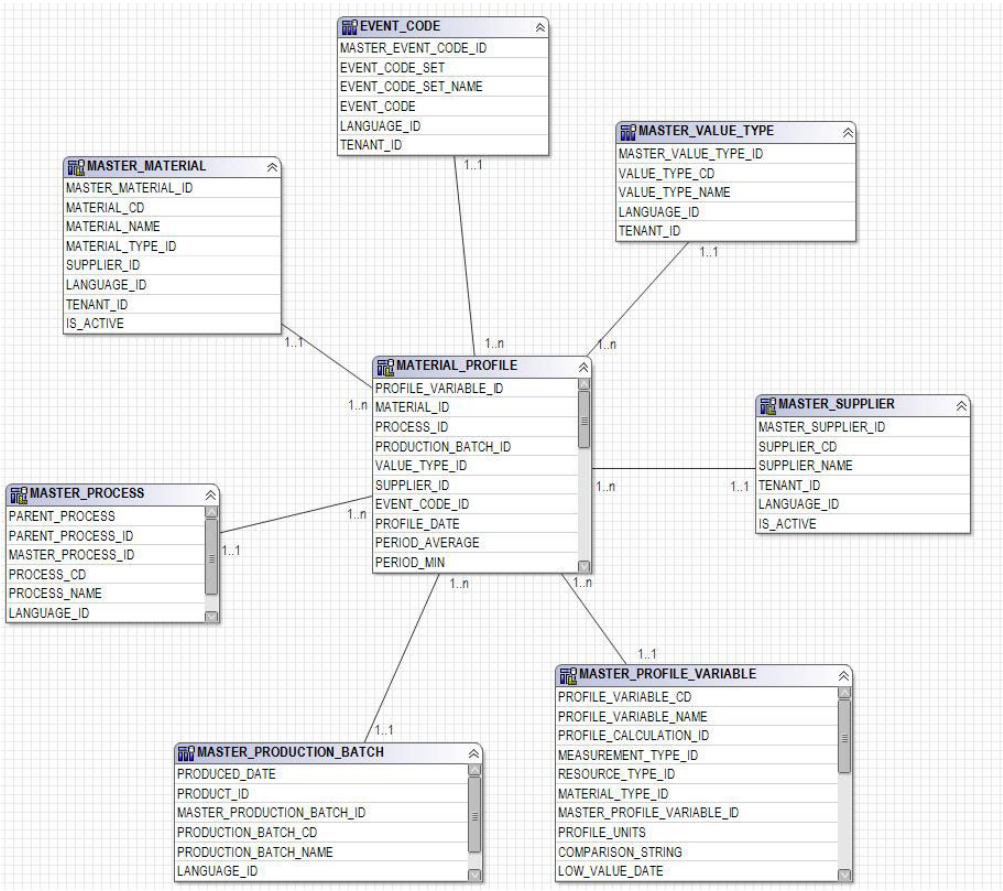


Figure 58. The material_profile star schema

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

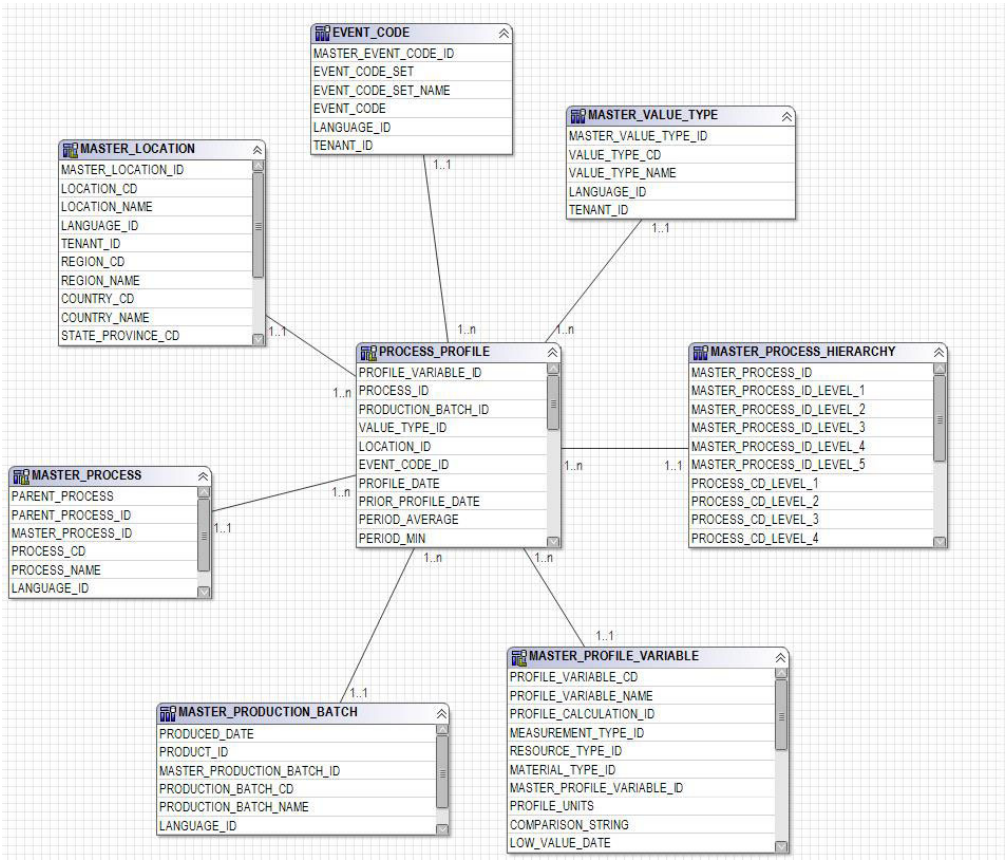


Figure 59. The process_profile star schema

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

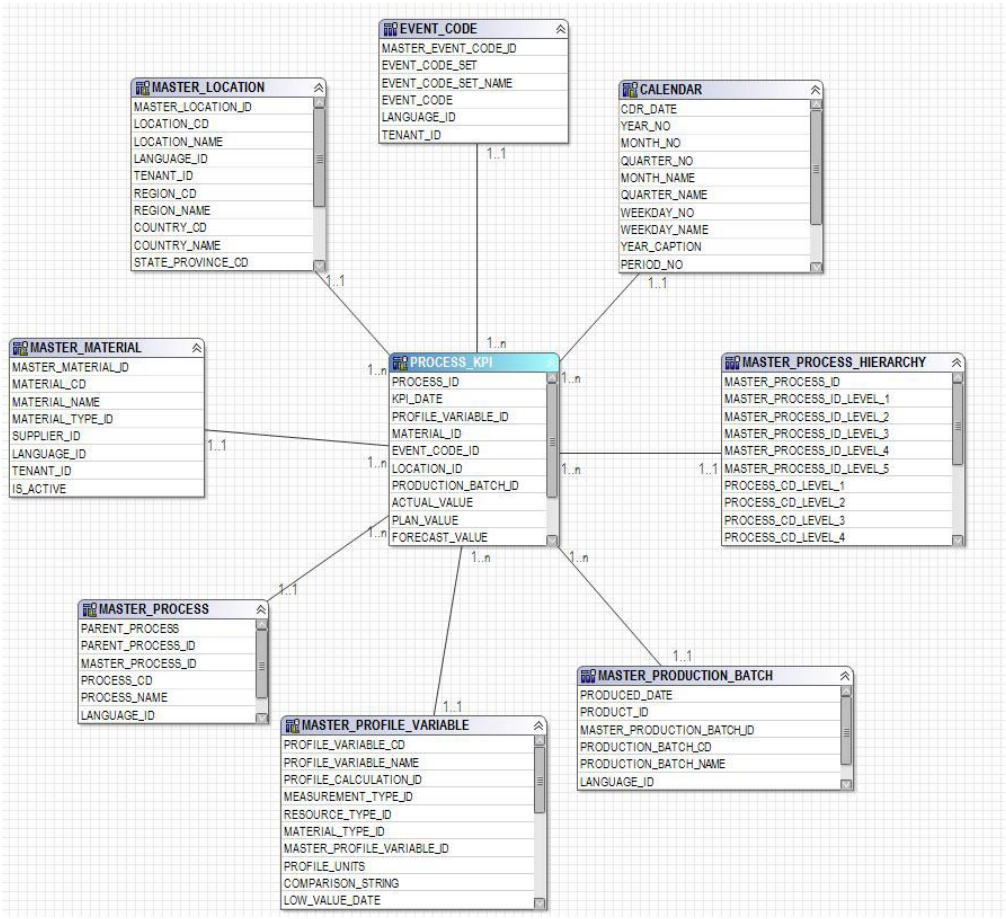


Figure 60. The process_kpi star schema

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

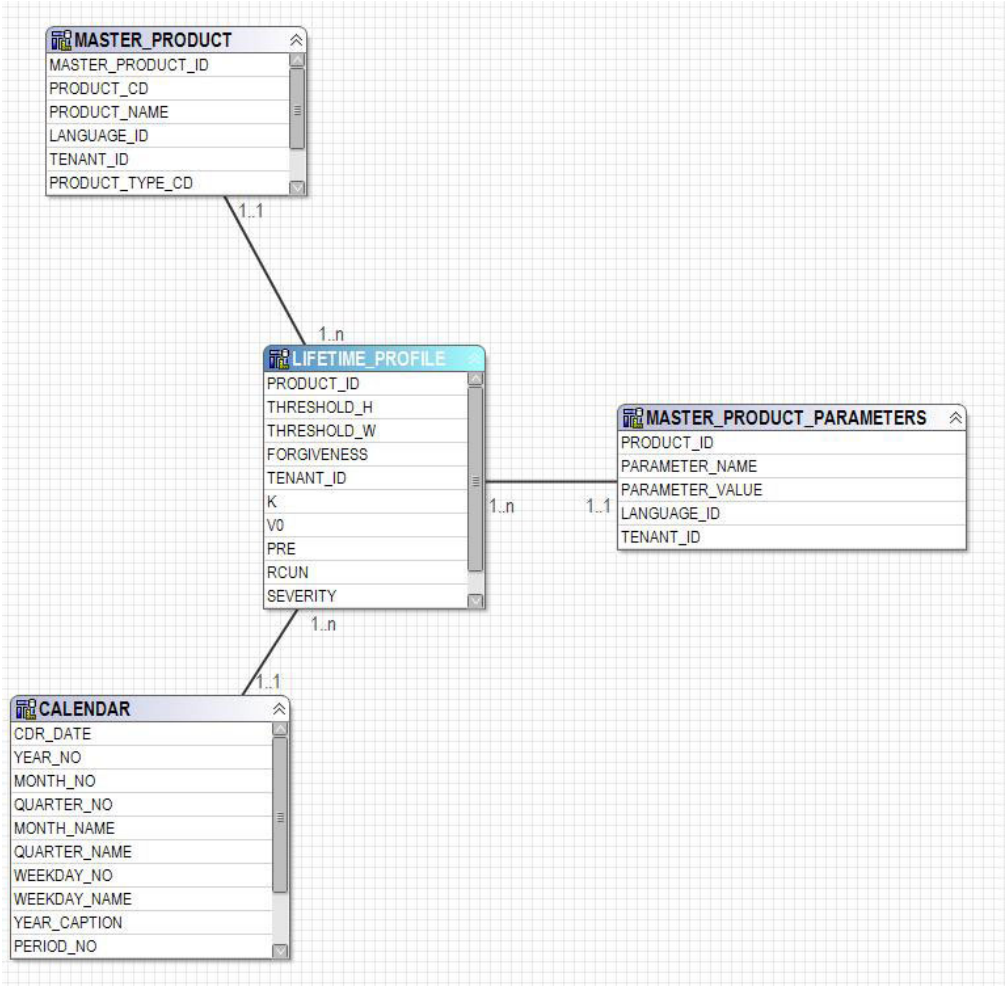


Figure 61. The lifetime_profile star schema

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

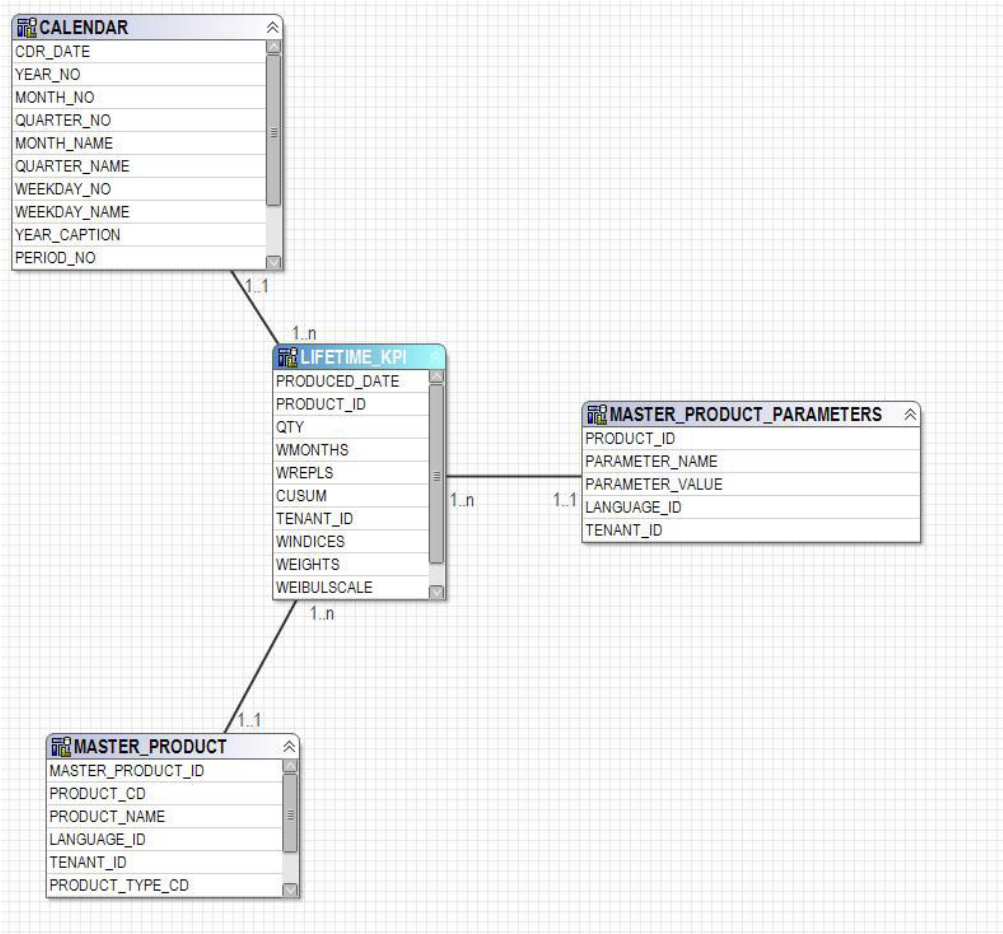


Figure 62. The lifetime_kpi star schema

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

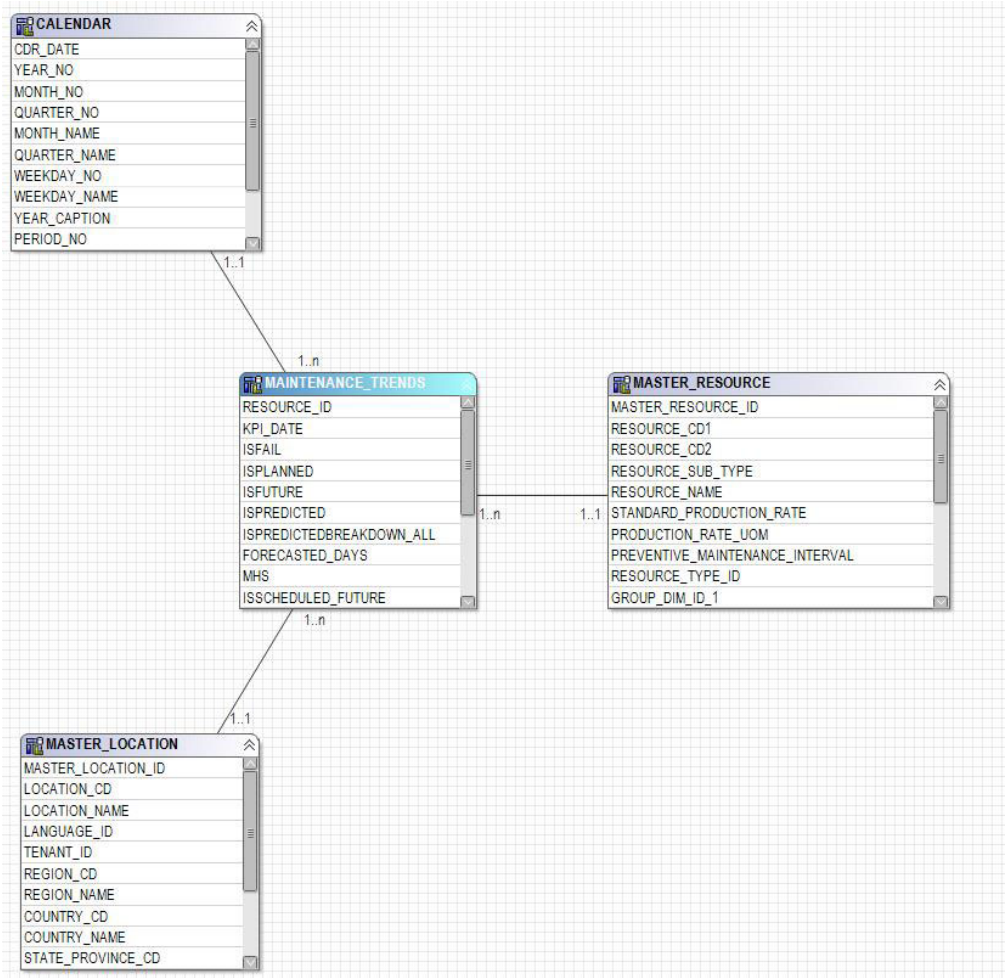


Figure 63. The maintenance_trends star schema

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

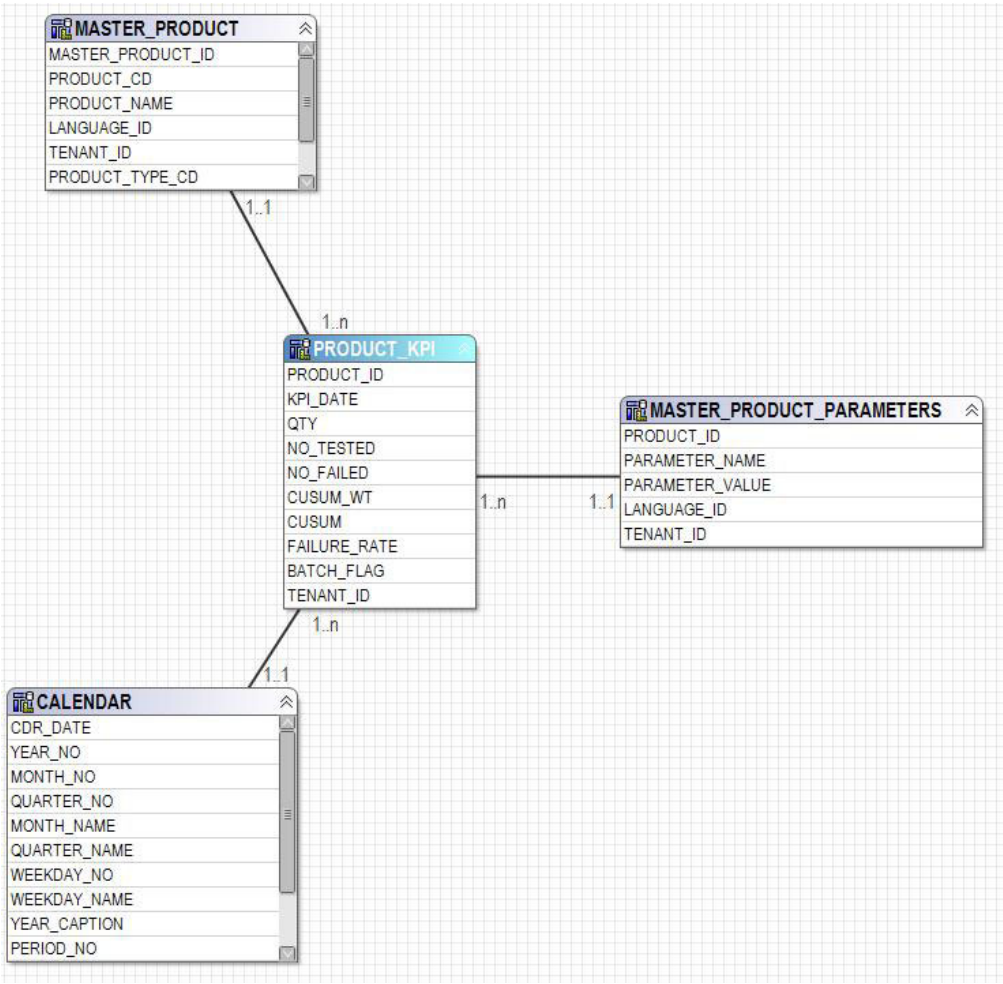


Figure 64. The product_kpi star schema

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

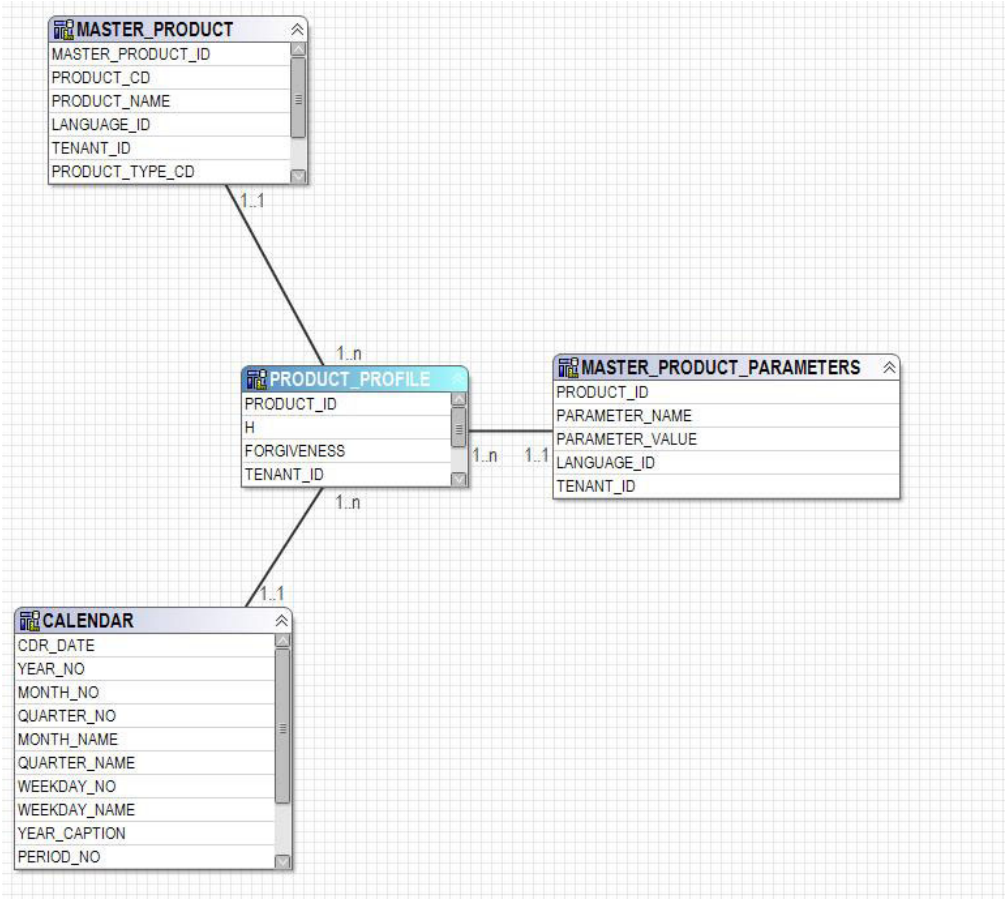


Figure 65. The product_profile star schema

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

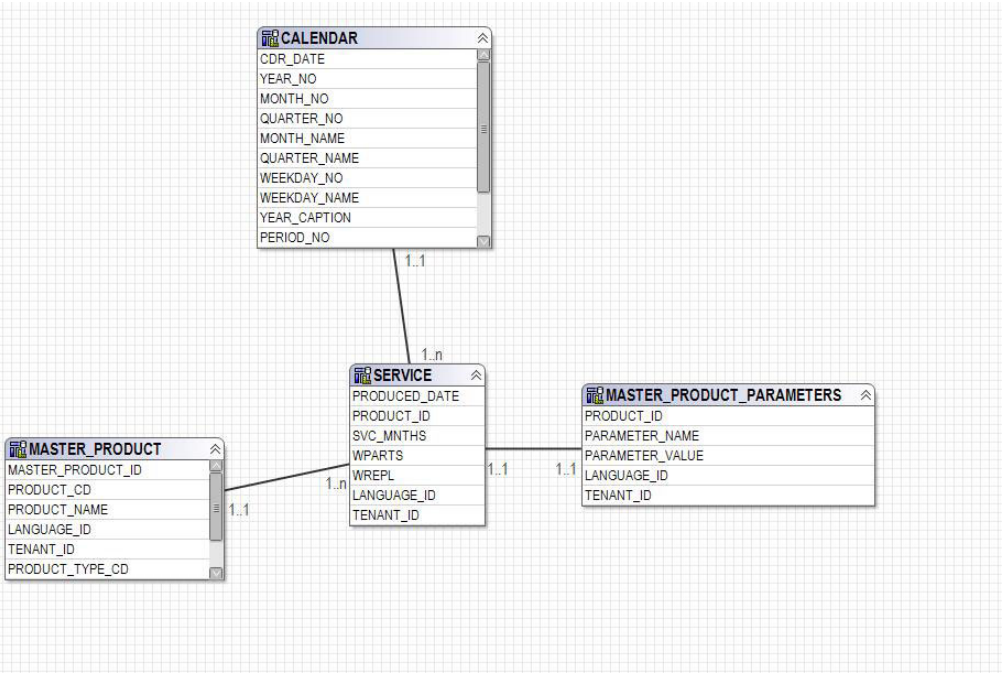


Figure 66. The service star schema

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 and avoid cross product result sets:

- 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 E. IBM Predictive Maintenance and Quality Artifacts

IBM Predictive Maintenance and Quality (PMQ) artifacts contain the configuration files that provide connections to customer data, predictive models, rules, dashboards, reports, and external systems.

PMQ artifacts also contain sample data to aid in the understanding of how PMQ connects, manages, and analyzes data to produce business tools in the form of reports, dasboards, or maintenance work orders. These artifacts can be modified, as explained in this solution guide, for additional asset model requirements, event types, custom reports, or connections to other external data sources or systems of engagement.

Data model

The data model file name is `IBMPMQ.sql`. This DDL contains scripts to create all the tables that form the PMQ Master / Event / Profile Data mart. It contains stored procedures for the initial set up of Language and Tenant Data to perform basic operations required by PMQ functions.

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 all the templates, reports, and data of the MDM CE data model specific to PMQ Master Data.

IBM Integration Bus and ESB artifacts

IBM Integration Bus (IIB) and Enterprise Service Bus (ESB) artifacts are provided.

IBM Integration Bus archive files

IBM Integration Bus archive files are shown in the following table:

Table 66. IBM Integration Bus archive files

Sl. No.	BAR Files	Description
1.	PMQMasterDataLoad	onboard Master Data information on to the PMQ Data mart.
2.	PMQEventDataLoad	Onboard and process event Data information on to the PMQ Event Store Integrate with SPSS scoring services (sensor health score and integrated health score) and process score results
3.	PMQMaintenance	Perform data preparations and invoke SPSS Maintenance Job as per the schedule
4.	PMQTopNFailure	Perform data preparations and invoke SPSS TopN Failure Job as per the schedule
5.	PMQQEWSInspection	Prepares data and invokes the QEWS algorithm to perform inspection early warning analysis and loads the results back to the Profile data Mart of PMQ.

Table 66. IBM Integration Bus archive files (continued)

Sl. No.	BAR Files	Description
6.	PMQQEWSWarranty	Gathers data from the Service tables of PMQ data mart and passes as input to the QEWSL analysis and loads the results to the Profile data mart of PMQ.
7.	PMQMaximoIntegration	load master data and workorder from Maximo in PMQ and also supports creation/updating of Maximo workorders
8.	PMQQEWSIntegration	provide integration support to call Inspection and Warranty flows as per the required sequence or schedule and to invoke SPSS warranty stream
9.	PMQModelTraining	Invoke SPSS Job for training SPSS streams for sensor health score and integrated health score

Supported JAR files

Supported JAR files are shown in the following table:

Table 67. Supported JAR files

Sl. No.	JAR / Properties / XML files	Description
1.	foundation-engine-api-1.0.0.0-SNAPSHOT.jar	APIs provided by Analytic Solution Foundation 1.0
2.	foundation-engine-core-1.0.0.0-SNAPSHOT	Implementation jar of Analytics Solution Foundation 1.0
3.	commons-collections-3.2.1.jar	This jar provides utility methods for most of the collection interfaces.
4.	commons-io-1.4.jar	This is a library of utilities to assist with developing IO functionality
5	commons-lang-2.4.jar	Provides a host of helper utilities for the java.lang API, notably String manipulation methods
6	commons-pool-1.6.jar	This open source software library provides an object-pooling API and a number of object pool implementations.
7	hamcrest-core-1.3.jar	Provides a library of matcher objects allowing 'match' rules to be defined declaratively, to be used in other frameworks.
8.	log4j-1.2.16.jar	Serves methods for logging purpose.
9.	icu4j.52.1.jar	Serves for internationalization ,
10.	pmq-foundation.jar	PMQ custom calculations over what is supported by Foundation
11.	ews.jar	The early warning system java module to analyze inspection and warranty usecases.

Supported property files and XML files

Supported property files and XML files are shown in the following table:

Table 68. Supported property files and XML files

Sl. No.	JAR / Properties / XML files
1	SetPerm.sh - Used to set 755 on the folder structure containing Warranty and Inspection charts
2	credentials.properties - Used to store SPSS credentials and joblocation urls
3	loc.properties - This is a properties file which maintains the location information of where the outputs to Warranty and Inspection is to be rendered.
4	log4j.properties - This is to set the logging levels and paths for the logs to be persisted.
5	orchestration_definition.xsd - foundation orchestration schema
6	solution_definition.xsd - foundation solution schema
7	PMQ_orchestration_definition_inspection.xml PMQ_orchestration_definition_maintenance.xml PMQ_orchestration_definition_measurement.xml PMQ_orchestration_definition_topnfailure.xml PMQ_orchestration_definition_warranty.xml - These Foundation specific orchestration XML contains the orchestration mapping definitions to carry out the sequence of adapter calls to fulfill an operation. We have a separate XML for each of the use case/event type
8	PMQ_solution_definition.xml - This Foundation specific XML contains the table definitions and the relations to carry the DML and DDL operations.

Table 68. Supported property files and XML files (continued)

Sl. No.	JAR / Properties / XML files
13	PMQEventLoad.properties PMQMaintenance.properties PMQMaximoIntegration.properties PMQModelTraining.properties PMQQEWSIntegration.properties PMQTopNFailure.properties - These properties files will contain the webservice endpoint urls and are used to override the bar files with the right endpoint urls as per customer needs
14	Queues.txt - Contains all supporting queue definitions' and this is executed to create queues

Sample master data, event data, and QEWS data files

Sample master data files, event data files, and QEWS data files are provided.

The sample master data files are shown in the following list:

- `language_upsert.csv`
- `tenant_upsert.csv`
- `event_code_upsert.csv`
- `event_type_upsert.csv`
- `group_dim_upsert.csv`
- `location_upsert.csv`
- `material_type_upsert.csv`
- `measurement_type_upsert.csv`
- `observation_lookup_upsert.csv`
- `process_upsert.csv`
- `product_upsert.csv`
- `profile_calculation_upsert.csv`
- `resource_type_upsert.csv`
- `source_system_upsert.csv`
- `supplier_upsert.csv`
- `value_type_upsert.csv`
- `material_upsert.csv`
- `production_batch_upsert.csv`
- `profile_variable_upsert.csv`
- `resource_upsert.csv`

The sample event data files are shown in the following list:

- `event_observation_maintenance_training.csv`
- `event_observation_maintenance_training_recommendation.csv`
- `event_observation_sensor_training.csv`
- `event_observation_process_material.csv`

- event_observation_spc.csv
- event_observation_sensor.csv

The QEWS data files are shown in the following list:

- parameter_upsert.csv
- resource_production_batch_upsert.csv
- batchdata_inspection.csv
- event_observation_warranty.csv
- qewsrundate.txt

IBM SPSS artifacts

IBM SPSS streams and jobs for Warranty, Maintenance, TopN failure predictors, SENSOR-based health analytics, and Integrated health analytics are provided as artifacts.

Warranty - Streams and Jobs

Warranty artifacts are shown in the following table:

Table 69. Warranty - Streams and Jobs

.pes file	Modeler Stream / ADM Streams / CaDS jobs	Description
IBMPMQ_QEWSL	IBMPMQ_QEWSL_WARR.str	The manufacturing or production warranty stream built to do a ETL sort of processing. No modeling activity involved here
	IBMPMQ_QEWSL_JOB	CaDS job used to invoke IBMPMQ_QEWSL_WARR.str for the manufacturing (MFG) or production (PROD)use cases
	IBMPMQ_QEWSL_SALES.str	CaDS job used to invoke IBMPMQ_QEWSL_JOB for the sales (SALES) use case
	IBMPMQ_QEWSL_ SALES_JOB	CaDS job used to invoke IBMPMQ_QEWSL_SALES.str for the SALES use cases

Maintenance - Stream and Jobs

Maintenance artifacts are shown in the following table:

Table 70. Maintenance - Stream and Jobs

.pes file	Modeler Stream / ADM Streams / CaDS jobs	Description
IBMPMQ_MAINTENANCE_ ANALYTICS	MAINTENANCE.str	Main stream in maintenance to identify and forecast the forecasted days to next maintenance and calculate the Maintenance Health score value .

Table 70. Maintenance - Stream and Jobs (continued)

.pes file	Modeler Stream / ADM Streams / CaDS jobs	Description
	MAINTENANCE_DAILY.str	Gives the Maintains details for a specific day
	MAINTENANCE_ RECOMMENDATIONS.str	The ADM Stream to give the Maintenance recommendations
	IBMPMQ_MAINTENANCE_ ANALYTICS_JOB	CaDS job used to invoke MAINTENANCE.str, MAINTENANCE_DAILY.str, MAINTENANCE_ RECOMMENDATIONS.str, and IBMPMQ_MAINTENANCE_ ANALYTICS_JOB

TopN failure predictors - Stream and Jobs

TopN failure predictors artifacts are shown in the following table:

Table 71. TopN failure predictors - Stream and Jobs

.pes file	Modeler Stream / ADM Streams / CaDS jobs	Description
IBMPMQ_TOP_FAILURE_ PREDICTORS	TopN_MODEL.str	Modeling stream to extract and store the PMML giving the predictor importance of various configured parameters in predicting a resource's failure.
	TopN_XML.str	This stream uses the PMML generated by the TopN_MODEL.str stream and extracts the necessary information from it and does essential transformation such that the output can be consumed by Cognos
	IBMPMQ_TOP_FAILURE_ PREDICTORS_JOB	CaDS job used to invoke the TopN_MODEL.str and the TopN_XML.str streams
	TOPN_EVENTS.str	Create csv with of the Top N data in a format that can be loaded into the PMQ event table using the IIB flows

SENSOR-based health analytics - Stream and Jobs

SENSOR-based health analytics artifacts are shown in the following table:

Table 72. SENSOR-based health analytics - Stream and Jobs

.pes file	Modeler Stream / ADM Streams / CaDS jobs	Description
IBMPMQ_SENSOR_ ANALYTICS	SENSOR_HEALTH_DATA_ PREP.str	A Data preparation stream which retrieves the data from IBM PMQ tables and prepares the data to be used in the modeling, the eligible data is exported to a csv file for the modeling
	SENSOR_HEALTH_ COMBINED.str	The combined stream helps in training the models and also refresh them for the scoring service
	SENSOR_HEALTH_ ANALYTICS_JOB	CaDS job used to invoke the SENSOR_HEALTH_ COMBINED.str stream
	IBMPMQ_SENSOR_ ANALYTICS.str	This stream is auto generated when a training happens and for the real time scoring -- SENSOR_HEALTH_SCORE service configured to be used

Integrated health analytics - Stream and Jobs

Integrated health analytics artifacts are shown in the following table:

Table 73. Integrated health analytics - Stream and Jobs

.pes file	Modeler Stream / ADM Streams / CaDS jobs	Description
IBMPMQ_INTEGRATED_ ANALYTICS	INTEGRATION_HEALTH_ DATA_PREPARATION.str	A Data preparation stream which retrieves the data from IBM PMQ tables and prepares the data to be used in the modeling, the eligible data is exported to a csv file for the modeling
	INTEGRATION_HEALTH_ COMBINED.str	The combined stream helps in training the models and also refresh them for the scoring service
	INTEGRATION_HEALTH_ ANALYTICS_JOB	CaDS job used to invoke the INTEGRATION_HEALTH_ COMBINED.str stream

Table 73. Integrated health analytics - Stream and Jobs (continued)

.pes file	Modeler Stream / ADM Streams / CaDS jobs	Description
	IBMPMQ_INTEGRATED_ ANALYTICS.str	This stream is auto generated when a training happens and for the real time scoring -- INTEGRATED_HEALTH_ SCORE service configured to be used

IBM Cognos Business Intelligence Artifacts

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

Framework Manager model

The Framework Manager model is described in the following table:

Table 74. Framework Manager model

Sl. No.	FM Model	Purpose
1.	IBMPMQ	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.

Site Overview dashboard

The Site Overview dashboard is described in the following table:

Table 75. Site Overview dashboard

Sl. No.	Report/Dashboard	Purpose
1.	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.
2.	Top 10 Contributors	Identifies the equipment, locations, and operators responsible for the most failures.
3.	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?
4.	Actual vs Plan	You can monitor how closely the metrics track against the plan. Variances are highlighted.
5.	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.

Table 75. Site Overview dashboard (continued)

Sl. No.	Report/Dashboard	Purpose
6.	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.
7.	Recommended actions	A summary of all recommended actions for each piece of equipment, for the health score measurement.

Equipment Reports dashboard

The Equipment Reports dashboard is described in the following table:

Table 76. Equipment Reports dashboard

Sl. No.	Report/Dashboard	Purpose
1.	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.
2.	Equipment Control Chart	Shows the upper and lower control limits and the average limits for selected measures.
3.	Equipment Run Chart	Shows the measures for a particular piece of equipment.
4.	Equipment Outliers	Shows detailed measures for a piece of equipment that shows anomalies.
5.	Event Type History	Lists the events for a device.

Product Quality dashboard

The Product Quality dashboard is described in the following table:

Table 77. Product Quality dashboard

Sl. No.	Report/Dashboard	Purpose
1.	Defect Analysis	Shows product defects and inspection rates.

Table 77. Product Quality dashboard (continued)

Sl. No.	Report/Dashboard	Purpose
2.	Inspection Rate Analysis	Examines the relationship between inspections and defects over time to find the optimal rate of inspection.
3.	Material Usage By Process	Provides an overview of material usage in the production processes.

SPC reports

SPC reports are described in the following table:

Table 78. SPC reports

Sl. No.	Report/Dashboard	Purpose
1.	SPC - Histogram	This report allows a visual interpretation of data by indicating the number of data points (events) that lie within a range of values, called a class or a bin. The frequency of the data that falls in each bin is depicted by the use of a bar.
2.	SPC - X Bar and S / R Charts	To track instantaneous variations and to evaluate the stability of the variability with in the process for smaller sample sizes (R Chart) and for bigger sample sizes (S Chart)

Other reports

Other reports are described in the following table:

Table 79. Other reports

Sl. No.	Reports/Dashboard	Purpose
1.	Advance KPI Trend Report	This chart compares multiple key performance indicators (KPIs) across multiple resources. You can use this chart to analyze variations in a resource against a set of profiles.

Table 79. Other reports (continued)

Sl. No.	Reports/Dashboard	Purpose
2.	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.
3.	Audit Report	Shows the counts of rows in the major master data tables.

Drill Through reports from the Audit report

The following table lists Drill Through reports from the Audit report.

Table 80. Drill Through reports from the Audit report

Sl. No.	Reports/Dashboard	Purpose
1.	Resource List	Lists the resources by resource type.
2.	Profile Variables	Lists all measures and key performance indicators that are being tracked in daily profiles and historical snapshots.
3.	Process List	Lists all production processes.
4.	Material List	Lists materials that are used in the production process.
5.	Production Batch List	Lists production batches.
6.	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.
7.	Measurement Type List	Lists measurement types. For each measurement type, the report shows unit of measure and aggregation type.

Maintenance dashboard and Top N Failure reports

The Maintenance dashboard and Top N Failure reports are described in the following table:

Table 81. Maintenance dashboard and Top N Failure reports

Sl. No.	Reports/Dashboard	Purpose
1.	Maintenance Overview Dashboard	This dashboard provides an overview of health score for the last current day in the record. Along with maintenance health score, report also shows a comparative view with sensor health score and integrated health score.
2.	Maintenance Advance Sorting Report	This chart displays the same measures as the main report (Maintenance Overview Dashboard) in a tabular format. Users can sort on a column by clicking the column header.
3.	Maintenance Health Score and Failure Detail Report	This report will help user to see the historical and forecasted health scores of a machine along with Historical Breakdown, Forecasted Breakdown, Planned Maintenance Schedules.
4.	Top N Failure Report	The plot shows the UNSIGNED predictor importance, indicating the absolute importance of any predictor in predicting a failure or non-failure condition.

Inspection and Warranty reports

Inspection and Warranty reports are described in the following table:

Table 82. Inspection and Warranty reports

Sl. No.	Reports/Dashboard	Purpose
1.	QEWS - Inspection Chart	This chart reports the failure rates and CUSUM values for a specific product type and product code over a time period.
2.	QEWSL - Warranty Chart	This chart reports the replacement rates for a specific product type and product code over a time period.

Appendix F. Troubleshooting

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 83. 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 Integration Bus log files

Errors that occur within IBM Integration Bus 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.

Performance tuning guidelines

You can tune the performance of your IBM Predictive Maintenance and Quality environment.

Deadlock errors happen when parallel processing is enabled

Deadlock errors in IBM Predictive Maintenance and Quality typically happen when parallel processing is enabled by increasing extra instances, and all messages are routed to single folders and queues.

About this task

The error message is named `EventError.txt` and is found in the `\error` folder in the IBM Integration Bus node, location that is defined by the **MQSI_FILENODES_ROOT_DIRECTORY** environment variable.

The error message is as follows:

```
"Error:Label:StdEventLoad_1.LoadEvent:TransactionId:fbc6b4c0-b434-11e2-8336
-09762ee50000TransactionTime:2013-05-04 02:34:022322:Child SQL exception:[unixODBC]
[IBM][CLI Driver][DB2/LINUXX8664] SQL0911N  The current transaction has been rolled
back because of a deadlock or timeout.  Reason code "2". SQLSTATE=40001"
```

For more information, see “Parallel processing” on page 58.

Procedure

1. Connect to the database with the following command: `db2 connect to db` <*dbname* `[IBMPMQ]>`
2. Set the isolation level to RR with the following command: `db2 set isolation level to RR`
3. Check the value of the dead lock check time setting with the following command: `db2 get db cfg |grep DL`

   The suggested values are:

   Interval for checking deadlock (ms)
   `(DLCHKTIME) = 20000`

Deadlock events
(MON_DEADLOCK) = WITHOUT_HIST

4. If the value for the **DLCHKTIME** property is less than 2000, then set the value with the following command: `db2 update db cfg for <dbname> using DLCHKTIME 20000 immediate`
5. Check the value of Lock list and percentage of Locks that are allowed per application `db2 get db cfg |grep LOCK`

   The suggested values are:

   Maximum storage for lock list (4 KB)
   (LOCKLIST) = 100000

   Percentage of lock lists per application
   (MAXLOCKS) = 97

   Lock timeout (sec)
   (LOCKTIMEOUT) = -1

   Block non logged operations
   (BLOCKNONLOGGED) = NO

   Lock timeout events
   (MON_LOCKTIMEOUT) = NONE

   Deadlock events
   (MON_DEADLOCK) = WITHOUT_HIST

   Lock wait events
   (MON_LOCKWAIT) = NONE
6. If the value for the **LOCKLIST** property is less 1000, then set the value with the following command: `db2 update db cfg for <dbname> using LOCKLIST 100000 immediate`
7. If the value for the **MAXLOCKS** property is less than 97, then set the value with the following command: db2 `update db cfg for <dbname> using MAXLOCKS 97 immediate`

Event processing performance

There are two approaches for increasing the performance of event processing. Events can be processed in multiple threads and events can be processed as a batch.

The event processing flow StdEventLoad processes messages that contain a single event or that contain a collection of events. The flow MultiRowEventLoad is an example of a flow that loads events and sends them for processing as a collection.

Processing events as collections has the best performance improvement when the events in the collection update the same profile rows. Sort the events so that similar events are processed together. For example, sorting them by device, time, and measurement.

Events that are processed as a collection can be processed only by a single thread. The exception is when the collections that are processed in separate threads do not update any of the same profile rows.

Processing single events by using multiple threads improves performance when the events are updating different profile rows. If the events are all updating the same profile rows, then there is little advantage in using multiple threads. A thread

locks the profile rows that it is updating and the other threads must wait until the lock is released. The lock is released when the transaction is committed.

Calculations that are identified as is_increment also have improved performance because they can update a profile row in the database without first having to retrieve it and lock it.

Troubleshooting reports

Reports in IBM Predictive Maintenance and Quality are created in IBM Cognos Report Studio. You may encounter problems when using some of the reports included with IBM Predictive Maintenance and Quality.

For further information about troubleshooting reports, see the *IBM Cognos Business Intelligence Troubleshooting Guide*, and the *IBM Cognos Report Studio User Guide*. These documents are available at IBM Cognos Business Intelligence Knowledge Center (http://www.ibm.com/support/knowledgecenter/SSEP7J).

Audit Report fails with error DMB-ECB-0088 A DMB cube build limit has been exceeded

This error can occur in any report when the master table contains more than 1 million resources but occurs most commonly in the Audit Report.

About this task

To fix the problem, you must increase the **MaxCacheSize** and **MaxNumberOfRecordRows** parameter values in the `qfs_config.xml` file.

Procedure

1. Go to the following IBM Cognos Business Intelligence configuration folder path: `/opt/ibm/cognos/c10_64/configuration`.
2. Open the `qfs_config.xml` file and increase the value of the following parameters:
   - MaxCacheSize
   - MaxNumberOfRecordRows
3. Save the `qfs_config.xml` file and run the report.

Notices

This information was developed for products and services offered worldwide.

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

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

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

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

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

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

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

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

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

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

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

IBM Software Group
Attention: Licensing
200 W. Madison St.
Chicago, IL
60606
U.S.A.

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

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

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

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

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

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

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

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

Trademarks

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

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

- 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

K

L

M

O

P

Q

R

S

T

U

V

W