Joining Model Repository and Profiling
Warehouse Database Views
Abstract

Regulatory reporting might require you to consolidate information available in multiple Model repository views and profiling warehouse views. You can join the views based on your requirements using key columns of the views. You can run a join query on the views to retrieve the required information from the Model and profiling warehouse repositories.

Supported Versions

- Data Quality 9.6.1
- Data Services 9.6.1

Table of Contents

Overview .................................................. 2
Business Example 1 ....................................... 2
  Database View Queries ................................. 3
Business Example 2 ....................................... 6
  Database View Query ................................... 7

Overview

Informatica provides a set of relational views that allow SQL access to the Model repository and profiling warehouse. Each view exposes different types of data stored in the Model repository and profiling warehouse. The Model repository stores the metadata for projects created in Informatica Analyst and Informatica Developer.

The Model repository also stores run-time and configuration information for applications that are deployed to a Data Integration Service. The profiling warehouse contains the results of the profiles that you run from the Analyst tool and Developer tool. You can combine multiple Model repository and profiling warehouse views to retrieve the required information from the repositories.

Note: The information in this document applies to deployment scenarios where both Model repository and profiling warehouse are on the same relational database. For information on how to run queries when the Model repository and profiling warehouse are on different databases, contact Informatica Global Customer Support.

Business Example 1

This example describes how you can combine Model repository and profiling warehouse views to retrieve the required information for a scorecard titled SC_Sales.

You need to find out specific scorecard details, such as the scorecard run details and information on metrics, metric groups, and cost. You performed the following steps when you created the scorecard:

1. You ran a column profile with multiple rules in it.
2. You created a scorecard titled SC_Sales from the profile results.
3. You edited the scorecard to move a few metrics to different metric groups in the scorecard.
Database View Queries

You can run the following query to retrieve the scorecard run details, such as start time, end time, and scorecard object name from the MRX_SCORECARD_RUN_REQ view:

Oracle

```sql
SELECT stat_id, object_name, parent_stat_id, created_time, last_update_time,
       service_name, node_name, MESSAGE, log_file_name, request_type,
       error_count, output_row_count, c_size, last_accessed_time, security_domain, user_name,
       request_state,
       processed_columns, total_columns, profile_id, start_time, end_time
FROM (SELECT stat_id, object_name, parent_stat_id, created_time, last_update_time,
       service_name, node_name, MESSAGE, log_file_name, request_type,
       error_count, output_row_count, c_size, last_accessed_time, security_domain, user_name,
       request_state,
       processed_columns, total_columns, profile_id,
       NEW_TIME (TO_DATE('1970/01/01', 'YYYY/MM/DD HH24:MI:SS')
       + NUMTODSINTERVAL (aa.start_time / 1000, 'SECOND'),
       'GMT',
       'PDT') AS start_time,
       NEW_TIME (TO_DATE('1970/01/01', 'YYYY/MM/DD HH24:MI:SS')
       + NUMTODSINTERVAL (aa.end_time / 1000, 'SECOND'),
       'GMT',
       'PDT') AS end_time,
       DENSE_RANK() OVER (PARTITION BY profile_id ORDER BY
       TO_DATE('1970/01/01', 'YYYY/MM/DD HH24:MI:SS')
       + NUMTODSINTERVAL (aa.end_time / 1000, 'SECOND'),
       'GMT',
       'PDT') desc AS rank_data
FROM mrx_scorecard_run_req aa)
WHERE rank_data = 1
```

IBM DB2

```sql
SELECT stat_id, object_name, parent_stat_id, created_time, last_update_time,
       service_name, node_name, MESSAGE, log_file_name, request_type,
       error_count, output_row_count, c_size, last_accessed_time,
       security_domain, user_name, request_state, processed_columns,
       total_columns, profile_id, start_time, end_time
FROM (SELECT stat_id, object_name, parent_stat_id, created_time,
       last_update_time, service_name, node_name, MESSAGE,
       log_file_name, request_type, error_count, output_row_count,
       c_size, last_accessed_time, security_domain, user_name,
       request_state, processed_columns, total_columns, profile_id,
       TIMESTAMP('1970-01-01-00.00.00') +
       CAST(aa.start_time as DECIMAL(31,0))/1000 seconds
       - 7 hour AS start_time,
       TIMESTAMP('1970-01-01-00.00.00') +
       CAST(aa.end_time as DECIMAL(31,0))/1000 seconds
       - 7 hour AS end_time,
       DENSE_RANK() OVER (PARTITION BY profile_id ORDER BY
       TIMESTAMP('1970-01-01-00.00.00') +
       CAST(aa.end_time as DECIMAL(31,0))/1000 seconds
       - 7 hour desc) AS rank_data
FROM mrx_scorecard_run_req aa) as RESULT
WHERE RESULT.rank_data = 1
```

Microsoft SQL Server

```sql
SELECT stat_id, object_name, parent_stat_id, created_time, last_update_time,
       service_name, node_name, MESSAGE, log_file_name, request_type,
```
error_count, output_row_count, c_size, last_accessed_time,
security_domain, user_name, request_state, processed_columns,
total_columns, profile_id, start_time, end_time
FROM (SELECT stat_id, object_name, parent_stat_id, created_time,
last_update_time, service_name, node_name, MESSAGE,
log_file_name, request_type, error_count, output_row_count,
c_size, last_accessed_time, security_domain, user_name,
request_state, processed_columns, total_columns, profile_id,
DATEADD(
    HOUR, -7,
    DATEADD(    SECOND, CAST(aa.start_time as BIGINT)/1000, '1970-01-01'
        ) AS start_time,
    DATEADD(    SECOND, CAST(aa.end_time as BIGINT)/1000, '1970-01-01'
            ) AS end_time,
DENSE_RANK ()  OVER ( PARTITION BY profile_id ORDER BY
DATEADD(
    HOUR, -7,
    DATEADD(    SECOND, CAST(aa.end_time as BIGINT)/1000, '1970-01-01')
        ) desc ) AS rank_data
FROM mrx_scorecard_run_req aa)as RESULT
WHERE RESULT.rank_data = 1

Next, you can join the following views to find information on scorecard metrics, metric groups, cost, and cost type:

- **MRX_SCORECARD_INFO**
- **MRX_SC_METRIC_GROUPS**
- **MRX_SC_NONRULE_METRIC**
- **MRX_SC_RULE_METRICS**

Run the following query to join the views and retrieve the required scorecard information:

**Oracle**

```sql
SELECT info.pid, info.nsid, info.sc_id, info.cid, info.sc_name,
info.cost_unit, info.sc_project_name, info.sc_path,
grp.metric_grp_name, aa.metric_name, aa.metric_weight, aa.cost_type,
aa.invalid_row_cost, aa.variable_cost_field, aa.data_obj_name,
aa.data_obj_type, aa.data_obj_project_name, aa.data_obj_path,
NULL rule_type, NULL "rule_expr"
FROM mrx_scorecard_info info,
(SELECT   pid, nsid, metric_grp_id, cid, sc_pid, metric_grp_name
FROM mrx_sc_metric_groups
GROUP BY pid, nsid, metric_grp_id, cid, sc_pid, metric_grp_name) grp,
(SELECT DISTINCT pid, nsid, metric_id, cid, metric_grp_id, metric_name,
metric_weight, cost_type, invalid_row_cost,
variable_cost_field, data_obj_pid, data_obj_nsid,
data_obj_id, data_obj_cid, data_obj_name,
data_obj_type, data_obj_project_name, data_obj_path
FROM mrx_sc_nonrule_metric) aa
WHERE sc_name = 'SC_Sales' AND sc_pid = info.pid
AND grp.pid = aa.metric_grp_id
UNION
/* To get information of rule metric */
SELECT info.pid, info.nsid, info.sc_id, info.cid, info.sc_name,
info.cost_unit, info.sc_project_name, info.sc_path,
grp.metric_grp_name, rulemet.metric_name, rulemet.metric_weight,
rulemet.cost_type, rulemet.invalid_row_cost,
rulemet.variable_cost_field, rulemet.rule_name, NULL,
rulemet.rule_project, rulemet.rule_path, rulemet.rule_type,
rulemet.rule_expression
FROM mrx_scorecard_info info,
(SELECT DISTINCT pid, nsid, metric_grp_id, cid, sc_pid,
```
metric_grp_name
FROM mrx_sc_metric_groups) grp,
(SELECT DISTINCT rulemet1.pid, rulemet1.nsid, rulemet1.metric_id,
rulemet1.cid, rulemet1.metric_grp_id,
rulemet1.metric_name, rulemet1.metric_weight,
rulemet1.cost_type, rulemet1.invalid_row_cost,
rulemet1.variable_cost_field, rulemet1.rule_name,
NULL, rulemet1.rule_project, rulemet1.rule_path,
rulemet1.rule_type,
DBMS_LOB.SUBSTR (rulemet1.rule_expr,
4000,
1) rule_expression
FROM mrx_sc_rule_metrics rulemet1) rulemet
WHERE sc_name = 'SC_Sales'
AND sc_pid = info.pid
AND grp.pid = rulemet.metric_grp_id

IBM DB2

SELECT info.pid, info.nsid, info.sc_id, info.cid, info.sc_name,
info.cost_unit, info.sc_project_name, info.sc_path,
grp.metric_grp_name, aa.metric_name, aa.metric_weight, aa.cost_type,
aa.invalid_row_cost, aa.variable_cost_field, aa.data_obj_name,
aa.data_obj_type, aa.data_obj_project_name, aa.data_obj_path,
'' rule_type, '' "rule_expr"
FROM mrx_scorecard_info info,
(SELECT pid, nsid, metric_grp_id, cid, sc_pid, metric_grp_name
FROM mrx_sc_metric_groups
GROUP BY pid, nsid, metric_grp_id, cid, sc_pid, metric_grp_name) grp,
(SELECT DISTINCT pid, nsid, metric_id, cid, metric_grp_id, metric_name,
metric_weight, cost_type, invalid_row_cost,
variable_cost_field, data_obj_pid, data_obj_nsid,
data_obj_id, data_obj_cid, data_obj_name,
data_obj_type, data_obj_project_name, data_obj_path
FROM mrx_sc_nonrule_metric) aa
WHERE sc_name = 'SC_Sales' AND sc_pid = info.pid
AND grp.pid = aa.metric_grp_id
UNION
SELECT info.pid, info.nsid, info.sc_id, info.cid, info.sc_name,
info.cost_unit, info.sc_project_name, info.sc_path,
grp.metric_grp_name, rulemet.metric_name, rulemet.metric_weight,
rulemet.cost_type, rulemet.invalid_row_cost,
rulemet.variable_cost_field, rulemet.rule_name, '',
rulemet.rule_project, rulemet.rule_path, rulemet.rule_type,
rulemet.rule_expression
FROM mrx_scorecard_info info,
(SELECT DISTINCT pid, nsid, metric_grp_id, cid, sc_pid,
metric_grp_name
FROM mrx_sc_metric_groups) grp,
(SELECT DISTINCT rulemet1.pid, rulemet1.nsid, rulemet1.metric_id,
rulemet1.cid, rulemet1.metric_grp_id,
rulemet1.metric_name, rulemet1.metric_weight,
rulemet1.cost_type, rulemet1.invalid_row_cost,
rulemet1.variable_cost_field, rulemet1.rule_name, '',
rulemet1.rule_project, rulemet1.rule_path,
rulemet1.rule_type,
substr(
cast(
rulemet1.rule_expr as varchar(4000)),4000,1
) rule_expression
FROM mrx_sc_rule_metrics rulemet1) rulemet
WHERE sc_name = 'SC_Sales'
AND sc_pid = info.pid
AND grp.pid = rulemet.metric_grp_id

Microsoft SQL Server

SELECT info.pid, info.nsid, info.sc_id, info.cid, info.sc_name,
info.cost_unit, info.sc_project_name, info.sc_path,
grp.metric_grp_name, aa.metric_name, aa.metric_weight, aa.cost_type,
aa.invalid_row_cost, aa.variable_cost_field, aa.data_obj_name,
aa.data_obj_type, aa.data_obj_project_name, aa.data_obj_path,
" rule_type", " *rule_expr"
FROM mrx_scorecard_info info,
(SELECT pid, nsid, metric_grp_id, cid, sc_pid, metric_grp_name
FROM mrx_sc_metric_groups
GROUP BY pid, nsid, metric_grp_id, cid, sc_pid, metric_grp_name) grp,
(SELECT DISTINCT pid, nsid, metric_id, cid, metric_grp_id, metric_name,
metric_weight, cost_type, invalid_row_cost,
variable_cost_field, data_obj_pid, data_obj_nsid,
data_obj_id, data_obj_cid, data_obj_name,
data_obj_type, data_obj_project_name, data_obj_path
FROM mrx_sc_nonrule_metric) aa
WHERE sc_name = 'SC_Sales' AND sc_pid = info.pid
AND grp.pid = aa.metric_grp_id
UNION
SELECT info.pid, info.nsid, info.sc_id, info.cid, info.sc_name,
info.cost_unit, info.sc_project_name, info.sc_path,
grp.metric_grp_name, rulemet.metric_name, rulemet.metric_weight,
rulemet.cost_type, rulemet.invalid_row_cost,
rulemet.variable_cost_field, rulemet.rule_name, rulemet.cost1,
rulemet.rule_project, rulemet.rule_path, rulemet.rule_type,
rulemet.rule_expression
FROM mrx_scorecard_info info,
(SELECT DISTINCT pid, nsid, metric_grp_id, cid, sc_pid,
metric_grp_name
FROM mrx_sc_metric_groups) grp,
(SELECT DISTINCT rulemet1.pid, rulemet1.nsid, rulemet1.metric_id,
rulemet1.cid, rulemet1.metric_grp_id,
rulemet1.metric_name, rulemet1.metric_weight,
rulemet1.cost_type, rulemet1.invalid_row_cost,
rulemet1.variable_cost_field, rulemet1.rule_name,
' cost1, rulemet1.rule_project, rulemet1.rule_path,
rulemet1.rule_type,
substring(
  cast(
    rulemet1.rule_expr as varchar(4000)),4000,1
) rule_expression
FROM mrx_sc_rule_metrics rulemet1) rulemet
WHERE info.sc_name = 'SC_Sales'
AND sc_pid = info.pid
AND grp.pid = rulemet.metric_grp_id

Finally, you can join the IDPV_SCORE_SMRY view and MRX_SCORECARD_INFO view to know more about the score summary:

```sql
select * from idpv_score_smry smry,mrx_scorecard_info info
where smry.scorecard_identifier=info.sc_id
AND sc_name = 'SC_Sales'
```

### Business Example 2

This example describes how you can combine Model repository and profiling warehouse views to retrieve profile information, such as the folder path, profile names, and the profile results.

You have multiple profiles that have the same name and are located in different folders. When you view the profile results from the profiling warehouse, you cannot differentiate between the profiles without the folder path name. To identify each of the profiles along with the results from the profiling warehouse, you need to view a list of profiles from the Model repository that includes the folder path, time of creation, and last run date of each profile.

You need to view information from the following views:

- mrx_profile_summary
- mrx_object_summary
- idpv_profile_details
Database View Query

You can run the following query to view the list of profiles and results including the file path names, date of creation, and date of last run:

Oracle

```sql
SELECT prof_def.profile_id, object_sumry_parent.object_name, prof_def.profile_name, object_sumry_child.project_name, object_sumry_child.object_path, NEW_TIME
   (TO_DATE ('1970/01/01', 'YYYY/MM/DD HH24:MI:SS') + NUMTODSINTERVAL (object_sumry_child.creation_time / 1000, 'SECOND')
   , 'GMT', 'PDT') AS time_created, prof_res.field_name, prof_res.null_count, prof_res.total_sum, prof_res.documented_datatype, prof_res.inferred_datatype, prof_res.minimum_value, prof_res.maximum_value, prof_res.last_time_run last_run_date
FROM mrx_profile_summary prof_summry, mrx_object_summary object_sumry_child, mrx_object_summary object_sumry_parent, idpv_profile_details prof_def, idpv_col_profile_results prof_res
WHERE prof_summry.cid = object_sumry_child.cid AND object_sumry_child.parent_cid = object_sumry_parent.cid AND prof_summry.eid = prof_def.profile_id AND prof_res.profile_id = prof_def.profile_id
GROUP BY object_sumry_child.project_name, object_sumry_child.creation_time, object_sumry_child.object_path, object_sumry_parent.object_name, prof_def.profile_id, prof_def.profile_name, prof_res.field_name, prof_res.documented_datatype, prof_res.inferred_datatype, prof_res.minimum_value, prof_res.maximum_value, prof_res.last_time_run, prof_res.null_count, prof_res.total_sum
```

IBM DB2

```sql
SELECT prof_def.profile_id, object_sumry_parent.object_name, prof_def.profile_name, object_sumry_child.project_name, object_sumry_child.object_path, TIMESTAMP(TIMESTAMP('1970-01-01-00.00.00') +
   CAST(
       object_sumry_child.creation_time as DECIMAL(31,0))/1000 ) seconds
   - 7 hour AS time_created, prof_res.field_name, prof_res.null_count, prof_res.total_sum, prof_res.documented_datatype, prof_res.inferred_datatype, prof_res.minimum_value, prof_res.maximum_value, prof_res.last_time_run last_run_date
FROM mrx_profile_summary prof_summry, mrx_object_summary object_sumry_child, mrx_object_summary object_sumry_parent,
```
idpv_profile_details prof_def,
idpv_col_profile_results prof_res
WHERE prof_summry.cid = object_sumry_child.cid
AND object_sumry_child.parent_cid = object_sumry_parent.cid
AND prof_summry.eid = prof_def.profile_id
AND prof_res.profile_id = prof_def.profile_id
GROUP BY object_sumry_child.project_name,
    object_sumry_child.creation_time,
    object_sumry_child.object_path,
    object_sumry_parent.object_name,
    prof_def.profile_id,
    prof_def.profile_name,
    prof_res.field_name,
    prof_res.documented_datatype,
    prof_res.inferred_datatype,
    prof_res.minimum_value,
    prof_res.maximum_value,
    prof_res.last_time_run,
    prof_res.null_count,
    prof_res.total_sum

Microsoft SQL Server

SELECT prof_def.profile_id, object_sumry_parent.object_name,
    prof_def.profile_name, object_sumry_child.project_name,
    DATEADD(HOUR, -7,
            DATEADD(SECOND,
                    CAST(object_sumry_child.creation_time as BIGINT)/1000,
                    '1970-01-01')) AS time_created,
    prof_res.field_name,
    prof_res.null_count, prof_res.total_sum,
    prof_res.documented_datatype, prof_res.inferred_datatype,
    prof_res.minimum_value, prof_res.maximum_value,
    prof_res.last_time_run last_run_date
FROM mrx_profile_summary prof_summry,
    mrx_object_summary object_sumry_child,
    mrx_object_summary object_sumry_parent,
    idpv_profile_details prof_def,
    idpv_col_profile_results prof_res
WHERE prof_summry.cid = object_sumry_child.cid
AND object_sumry_child.parent_cid = object_sumry_parent.cid
AND prof_summry.eid = prof_def.profile_id
AND prof_res.profile_id = prof_def.profile_id
GROUP BY object_sumry_child.project_name,
    object_sumry_child.creation_time,
    object_sumry_child.object_path,
    object_sumry_parent.object_name,
    prof_def.profile_id,
    prof_def.profile_name,
    prof_res.field_name,
    prof_res.documented_datatype,
    prof_res.inferred_datatype,
    prof_res.minimum_value,
    prof_res.maximum_value,
    prof_res.last_time_run,
    prof_res.null_count,
    prof_res.total_sum

Authors

Vikas Kumar
Senior Software Engineer