FADW Metric Repository

This is the design document for the Metrics Repository within the Financial and Administrative Data Warehouse. This is being developed as part of the Health Finance Integration Program.

12/19/2017

Table of Contents

Change Log

#	Date	Person	Description
V1	3/8/2017	D Dobbs	Draft Version
V1.1	3/12/2017	D Dobbs	1. Can categories of metrics on reports have multiple levels? **Answer: Yes. 2 Max.** 2. Will grouping of Categorized Metrics get reused across multiple Reports? **Answer: Yes** 3. Will all Metrics on a Report have the same dimensions (e.g., a report on a particular business unit will only have metrics for that business unit)? **Answer: Yes** Agreed that implementation of Metric Repository will be phased in.
V1.2	3/15/2017	D Dobbs	1. Updated Table Definitions section to reflect model changes made in V1.1 2. Fleshed out Metric to encompass all metric metadata needed for Metadata Repository, Metric Repository and Dashboard Metric Catalogue.
V1.3	3/28/2017	D Meyer	1. Updated document after initial discussion w/Claudia to include template to receive metrics from business owners & deformalized dimensions to receive 2. Updated document after initial discussion w/PD-Cognos team
V1.4	4/4/2017	D Meyer	1. Updated Model – added fields, & included estimates for base metric store build & Financial Highlights data mart builds 2. Updated to include assumptions (receiving lowest level metric only & rolling up along dim. Hier. in BI tool) (ETL JOB sourcing measures from Financial DW into repository for one location to go to for BI reporting 3. ETL JOB: Sourcing Budget data also from Financial DW or business owner 4. Will still need to update clinical measures based on what business owners can supply 5. Will need to capture original reported metric and adjusted or revised updated amounts 6. TBD – determine how to create calculation routines 7. Need Lori's spreadsheet of metrics shown during webEx
V1.5	11/6/2017	Jeff	8. Updated to reflect current implementation.
V1.6	11/12/2017	D Meyer	1. Updated excel template & received EPSI data from Miguel 2. Updated reporting data model for Financial Highlights report to include period & BU dimension objects with Metric dimension populated with additive measures only & Fact table. Then, the other metrics will be calculated

			with database functions leveraging the base Fact_Metric table.
V1.7	12/12/17	D Meyer	1. Updated Clinical Metric sourcing strategy on pg4 2. Updated Calculated metric section on pg4 3. Updated data model diagram on pg8
V1.8	12/18/17	D Meyer	Added Calculated Metric Process
V1.9	1/8/18	D Meyer	Added flow chart & updated Metric process for design review

Metric Repository Overview

This document describes the logical data model and high level data population and report generation process design for the Metric Repository, which will store metrics that are displayed on Health System financial reports. The Repository consists of:

- Metrics Facts that are sliced by dimensions in a dimension set
- Reports that have one or more categories of metrics on them

The logical data model design for the Metric Repository has primary, surrogate primary and foreign keys identified for each logical table. The technical team will take this logical data model, flesh out the data model for additional attributes for each logical table and then translate that into a physical data model that gets implemented.

The Metric Repository will be implemented in phases. See the Phase Implementation Plan section for more details.

Assumptions, Risks, Constraints

1. Team will need to create a job / process to populate financial measures, at the lowest level required, into the metrics repository.
    a. Budget / Actual
    b. Reported / Adjusted (only for Actuals)
    c. From excel sources & Financial Data Warehouse
2. For clinical measures, discuss with data providers best approach to source data –
    a. Standard excel template is provided for some and rest are
    b. Sourced from UCAll database loaded by Adam Joe’s team
3. Calculate the metrics will use the stored measures in the metrics repository.
    a. Receive the root elements and store for each calculation (and also calculate the metric)
    b. Budget / Actual
    c. Reported / Adjusted (only for Actuals)
        i. Will need to recalculate metric if received adjusted root measure elements & add an indicator this was modified.
4. In addition to base metric repository data store, create and populate Financial Highlights data mart objects.
5. Create calculation routines – Reporting team created stored procedures for all calculated metrics for Financial Highlights report (first use case) and ETL team after sourcing all root measures from financial DW, Excel, UCAll will also monthly run these stored procs to calculate & store all the calculated metrics for this report.

Metric Repository Set Operation

Step 1 - Set up the Metrics and Dimensions

A. Add Metrics
B. Add Period Types
C. Add Dimension and Dimension Hierarchy
    i. Add Dim Type
    ii. Add Dim Value
    iii. Add Dim Hierarchy
    iv. Add Dim Hier Node (bridge between Dim Value and Dim Hierarchy)
D. Add Dimension Set
    i. Add Dim Set
    ii. Add Dim Set Dim (bridge between Dim Set and Dim)

Step 2 – Create Metric Sets

A. Add Category and Hierarchy
    i. Add Category
    ii. Add Category Hierarchy and give it a name
    iii. Add Category Hier Nodes (bridge between Category and Category Hierarchy)
B. Create a Metric Set
    - Add a Metric Set and give it a name
    - Add Categories and Metrics into Metric Set Itemk
        a. Select Category Hierarchy
        b. Select one or more Metrics for the Category(ies) in the Category Hierarchy
        c. Assign a Period Type to each Metric (e.g., daily, monthly, FYTD)
        d. Assign a CatSequence for the metric to order of the metric in the category.

Step 3 - Add Report and Report Definition

A. Add a Report - Enter the report name, report owner and any additional information needed about the report
B. Add a Report Dim Set - Select the Dimension Set(s) that that will be on the Report's filter page and put them into Report Dim Set
C. Add Metric Set(s) to the Report - Select the Metric Set(s) for the Report and assign a sequence number to indicate the order in which the Metric Set(s) will be on the Report.

Step 4 - Populate the Metric Values in Metric Fact

A. Populate Metric Fact with the metric ID, metric version, dimension set ID, period type ID, period end date, reported version, the metric value, metric units of measure, reported date and other audits fields as need (such as source).
B. Populate Metric Dimension with the values of the dimensions for the metric in Metric Fact. These will be the dimension values for the dimensions in the Dim Set identified by the DimSetID in the Metric Fact.

Step 5 - Run the Report

A. The Report Filter Page reads the Report table to get the Report Dim Set(s) possible for the Report. The user will enter ONE Report Dim Set on the Filter Page. This user will enter the values for each dimension in the Report Dim Set. Whatever the user enters will be the dimensions for ALL metrics on the Report.

B. The Report filter page will pass the information in the Report Execution and Report Filter Value tables (e.g., report ID, report version, dimension set ID, period end date, and dimension types and values) to the report creation program.

C. The report creation program uses the information passed to it to get the report's list of categories and metrics out of a combination of tables: Report Definition plus the Categorized Metric Set plus the Metric Set Item.

D. The report creation program will then have enough information to look into the Metric Dimension table to MetricFactID for the set of dimensions that match the filter criteria entered by the user on the report filter page. It does this by selecting distinct the combination of MetricFactID, and ReportedVersion for the fields listed as part of the Surrogate Key of the Metric Dimension (e.g., MetricID, MetricVer, DimSetID, PeriodTypeID, PeriodEndDt, DimTypeID, DimValue), where:

   - Metric_Set_Item.MetricID = Metric_Dimension.MetricID AND
   - Metric_Set_Item.MetricVer = Metric_Dimension.MetricVer AND
   - Report_Execution.DimSetID = Metric_Dimension.DimSetID AND
   - Report_Definition.PeriodTypeID = Metric_Dimension.PeriodTypeID AND
   - Report_Execution.PeriodEndDt = Metric_Dimension.PeriodEndDt AND
   - For each entry in Report Filter value:
     e. Report_Filter_Value.DimTypeID = Metric_Dimension.DimTypeID AND
     f. Report_Filter_Value.DimValue = Metric_Dimension.DimValue
   - The report creation uses program will then use the MetricFactID (a unique number for each entry in the Metric Fact table) with the HIGHEST ReportedVersion to read the Metric Fact table to get the MetricValue and any other needed information.

Phase Implementation Plan

The Metric Repository will be implemented in phases. In the initial phase only those items that are needed to support the storage of metrics and dimensions in the FADW will be implemented as physical tables. These logical entities are listed below.

- Metric
- Metric Fact
- Metric Dimension
- Dim Type
- Dim Value
- Dim Hierarchy
- Dim Hierarchy Node
- Dim Set
- Dim Set Dim
- Period Type

Note: The technical staff will need to evaluate and make recommendations for aligning (and possibly integrating) the dimension tables listed above with the existing FADW dimension tables.

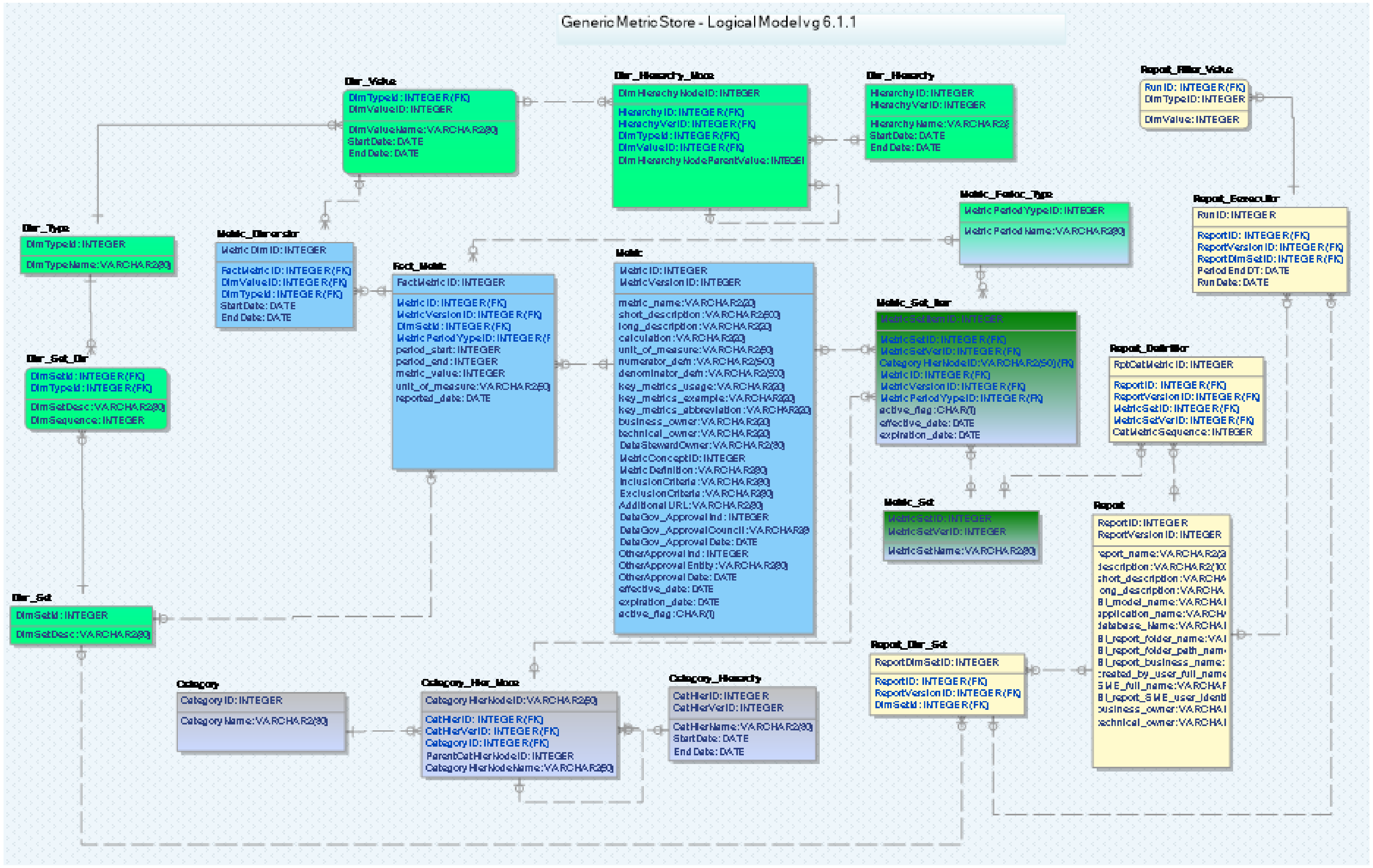
The Report Filter Page will need to collect the information in the following logical tables so that the report program can look up the values Metric Fact / Metric Dimension:

- Report Execution – the period end date
- Report Filter Value – the filter values (dimensions) for which the report is being run

The report program will also need to have available the information in the following tables to be able to look up the Metric Fact. The report development staff will determine the best way to make this information available to the report program.

- Report
- Report Definition
- Report Dim Set
- Category
- Category Hierarchy
- Category Hier Node
- Metric Set
- Metric Set Item

Data Model



Generic Metric Store - Logical Model vg 6.1.1
Dim_Value
DimTypeId : INTEGER (FK)
DimValueID : INTEGER
DimValueName : VARCHAR2(80)
StartDate : DATE
End Date : DATE
Dim_Hierarchy_Node
Dim Hierarchy NodeID : INTEGER
Hierarchy ID : INTEGER (FK)
HierachyVerID : INTEGER (FK)
DimTypeId : INTEGER (FK)
DimValueID : INTEGER (FK)
Dim Hierarchy NodeParentValue : INTEGEI
Dim_Hierarchy
Hierarchy ID : INTEGER
HierachyVerID : INTEGER
Hierarchy Name : VARCHAR2(
StartDate : DATE
End Date : DATE
Report_Filter_Value
Run ID : INTEGER (FK)
Dim TypeID : INTEGER
DimValue : INTEGER
Dim_Type
DimTypeId : INTEGER
DimTypeName : VARCHAR2(80)
Metric_Dimension
Metric Dim ID : INTEGER
FactMetric ID : INTEGER (FK)
DimValueID : INTEGER (FK)
DimTypeId : INTEGER (FK)
StartDate : DATE
End Date : DATE
Fact_Metric
FactMetric ID : INTEGER
Metric ID : INTEGER (FK)
MetricVersion ID : INTEGER (FK)
DimSetId : INTEGER (FK)
Metric Period YypeID : INTEGER (F
period_start : INTEGER
period_end : INTEGER
metric_value : INTEGER
unit_of_measure : VARCHAR2(50)
reported_date : DATE
Metric_Period_Type
Metric Period YypeID : INTEGER
Metric Period Name : VARCHAR2(80)
Report_Execution
Run ID : INTEGER
ReportID : INTEGER (FK)
ReportVersion ID : INTEGER (FK)
Report DimSetID : INTEGER (FK)
Period End DT : DATE
Run Date : DATE
Metric
Metric ID : INTEGER
MetricVersion ID : INTEGER
metric_name : VARCHAR2(20)
short_description : VARCHAR2(500)
long_description : VARCHAR2(20)
calculation : VARCHAR2(20)
unit_of_measure : VARCHAR2(50)
numerator_defn : VARCHAR2(500)
denominator_defn : VARCHAR2(500)
key_metrics_usage : VARCHAR2(20)
key_metrics_example : VARCHAR2(20)
key_metrics_abbreviation : VARCHAR2(20)
business_owner : VARCHAR2(20)
technical_owner : VARCHAR2(20)
DataStewardOwner : VARCHAR2(30)
MetricConceptID : INTEGER
Metric Definition : VARCHAR2(80)
InclusionCriteria : VARCHAR2(80)
ExclusionCriteria : VARCHAR2(80)
Additional URL : VARCHAR2(80)
DataGov_Approval Ind : INTEGER
DataGov_ApprovalCouncil : VARCHAR2(8
DataGov_Approval Date : DATE
OtherApproval Ind : INTEGER
OtherApproval Entity : VARCHAR2(80)
OtherApproval Date : DATE
effective_date : DATE
expiration_date : DATE
active_flag : CHAR(1)
Metric_Set_Item
MetricSetItem ID : INTEGER
MetricSetID : INTEGER (FK)
MetricSetVerID : INTEGER (FK)
Category HierNodeID : VARCHAR2(50) (FK)
Metric ID : INTEGER (FK)
MetricVersion ID : INTEGER (FK)
Metric Period YypeID : INTEGER (FK)
active_flag : CHAR(1)
effective_date : DATE
expiration_date : DATE
Report_Definition
RptCatMetric ID : INTEGER
ReportID : INTEGER (FK)
ReportVersion ID : INTEGER (FK)
MetricSetID : INTEGER (FK)
MetricSetVerID : INTEGER (FK)
CatMetricSequence : INTEGER
Dim_Set_Dim
DimSetId : INTEGER (FK)
DimTypeId : INTEGER (FK)
DimSetDesc : VARCHAR2(80)
DimSequence : INTEGER
Metric_Set
MetricSetID : INTEGER
MetricSetVerID : INTEGER
MetricSetName : VARCHAR2(80)
Report
ReportID : INTEGER
ReportVersion ID : INTEGER
report_name : VARCHAR2(3
description : VARCHAR2(10
short_description : VARCHA
long_description : VARCHA
BI_model_name : VARCHAI
application_name : VARCHA
database_Name : VARCHAI
BI_report_folder_name : VAI
BI_report_folder_path_nam
BI_report_business_name :
reated_by_user_full_name
SME_full_name : VARCHAF
BI_report_SME_user_identi
business_owner : VARCHAI
technical_owner : VARCHAI
Dim_Set
DimSetId : INTEGER
DimSetDesc : VARCHAR2(80)
Report_Dim_Set
ReportDimSetID : INTEGER
ReportID : INTEGER (FK)
ReportVersion ID : INTEGER (FK)
DimSetId : INTEGER (FK)
Category
Category ID : INTEGER
Category Name : VARCHAR2(80)
Category_Hier_Node
Category HierNodeID : VARCHAR2(80)
CatHierID : INTEGER (FK)
CatHierVerID : INTEGER (FK)
Category ID : INTEGER (FK)
ParentCatHierNodeID : INTEGER
Category HierNodeName : VARCHAR2(80)
Category_Hierarchy
CatHierID : INTEGER
CatHierVerID : INTEGER
CatHierName : VARCHAR2(80)
StartDate : DATE
End Date : DATE

Table Definitions

Tables implemented for Financial Highlights are in yellow.

Model Section	Table Name	Description
Category	Category	Stores the list of categories that can be used to group metrics on a report (Report). For example, a report may have 3 categories of metrics: Financial, Clinical and Other. There would 3 rows in this table, one each for Financial, Clinical and Other.
	Category Hier Node	Each row in this table stores a node in a category hierarchy (Category Hierarchy). Each node (i.e., row) may have a parent. This is a bridge table between Category Hierarchy and Category. The functional team will need to determine if there is the need to allow categories on reports to have a hierarchy. If this not needed, this table can be removed from the design and Category will be directly linked to Report Definition and to Report through a bridge table (new name of Report Category) that replaces the Report Category Hier bridge table.
	Category Hierarchy	Used to allow categories (Category) on report to have a hierarchy. For example, there may be a report (Report) that has 2 Categories, Financial and Clinical, and the Clinical category has subcategories of ED, Outpatient and Inpatient. The functional team will need to determine if there is the need to allow categories on reports to have a hierarchy. If this not needed, this table can be removed from the design.
Dimension	Dim Hier Node	Each row in this table stores a node in a dimension hierarchy (Dim Hierarchy). Each node may have a parent. This is a bridge table between Dim Hierarchy and Dim Value.
	Dim Hierarchy	Stores the list of dimension hierarchies. There will likely be one (or possibly more) dimension hierarchies for each dimension type (Dim Type). For example, ther may be one or more department hierarchies. A dimension hierarchy can have a version. Each version has a start and end date. This allows a history to be kept of dimension hierarchy changes.
	DIM_TYPE	Stores the types of dimensions that metrics can be sliced by. Examples include: business unit, business line and department.
	DIM_VALUE	Stores the values and names for a dimension type. Examples values for the business unit dimension type are: SFMED, SFFPO and SFBCH with corresponding names of UCSF Medical Center, UCSF Faculty Practice and UCSF Benioff Children's Hospital.

Model Section	Table Name	Description
	METRIC_PERIOD_TYPE	List the types of periods for a Metric Fact or a Report Definition. Examples are: monthly, quarterly, FTYD, Calendar YTD, daily.
Dimension Set	DIM_SET	Stores the list of dimension sets that can be used to slice a metric (Metric Fact) or that can be used to indicate the types of dimensions for a report (Report). The dimensions within a dimension set are stored in the Dim Set Dim table.
	METRIC_DIMENSION	This stores the dimensions that are part of a Dimension Set. For example, there may be a dimension set (Dim Set) that needs to allow a metric to be slice by business unit and department. In this instance the Dim Set Dim table will have 2 rows, one for business unit and the 2nd for department. Each row in Dim Set Dim has a sequence number so that the dimensions in the dimension sets can be ordered. The DimSequence is used to sequence the display of the dimensions on a report's (Report) filter page. The DimSequence will not be needed if the report's filter page is going to hard-code the list of the dimensions that a report can be run by.
Metric	METRIC	Stores the list of metrics and information about them, such as their definition and how the metrics is calculated. A metric can have multiple versions so that history can be maintained about definitions, calculations or other attributes of the metric that change over time.
	FACT_METRIC	"Stores calculated values for a metric for the dimension set (Dim Set), period type (Period Type), and period end date. The ReportedVer field allows a metric's value to be published multiple times in those instances where updates need to be made to the metric's reported value. The value(s) for each of the dimension(s) in the dimension set are stored in the Metric Fact Dim Value table.
	FACT_DIM_METRIC	Stores the dimension value(s) for the Metric Fact. For example, if the Metric Fact had a entry (row) that indicated that the discharges for the month ending December 31, 2016 was 2,000; the Metric Fact Dim Value would tell you that this was for business unit SFMED. There may be one or more rows in Metric Fact Dim Value for each row in Metric Fact.
	Metric Set	Used to define a set of metrics that will be part of a report. This set can be used by multiple reports. For example, there may be consistent set of metrics on all business unit and business line segment reports. If there are a set of metrics at the top of the report and a different set of metrics at the bottom of the report there will 2 different Metric Sets.

Model Section	Table Name	Description
	Metric Set Item	Used to identify metrics within a Metric Set and how those metrics are categorized. It also is used to identify the PeriodTypes (e.g., monthly, FYTD) for the metrics. The MetricSequence is used to sequence the order of the metrics that are within a category. The non-time-based dimensions for the metrics on the report are determined at the time the report is run by the user's filter page selections
Report	Report	Stores a list of the reports that display metrics. For example, the Financial Highlights Report or the Business Unit Segment Report.
	Report Definition	Stores the information that is needed to define the report including: metrics that are in the report's categories and the period types for the metrics. The MetricSequence is used to sequence the order of the metrics that are within a category.
	Report Dim Set	Stores the list of dimensions that can be set by the users on the report's filter page.
	Report Execution	This table, in combination with the Report Filter Value table, depicsts the information that is generated by the filter page of the report when it is run. The information in these two tables, combined with the information in the Report Defintion table, will allow the program that generates the report to find the right entry in the Metric Fact (and its associated Metric Fact Dim Value) table.
	Report Filter Value	This table, in combination with the Report Execution table, depicts the information that is generated by the filter page of the report when it is run. The information in these two tables, combined with the information in the Report Defintion table, will allow the program that generates the report to find the right entry in the Metric Fact (and its associated Metric Fact Dim Value) table.

DDL to Generate Repository

```
CREATE TABLE DIM_SET
(
 DIMSETID                              INTEGER CONSTRAINT SYS_C001759737 NOT NULL,
 DIMSETDESC                            VARCHAR2(80 BYTE) NULL
);

CREATE TABLE DIM_SET_DIM
(
 DIMTYPEID                             INTEGER CONSTRAINT SYS_C001759738 NOT NULL,
 DIMSETID                              INTEGER CONSTRAINT SYS_C001759739 NOT NULL,
 DIMSETDESC                            VARCHAR2(80 BYTE) NULL,
 DIMSEQUENCE                           INTEGER NULL
);

CREATE TABLE DIM_TYPE
(
 DIMTYPEID                             INTEGER CONSTRAINT SYS_C001759740 NOT NULL,
 DIMTYPENAME                           VARCHAR2(80 BYTE) NULL
);

CREATE TABLE DIM_VALUE
(
 DIMVALUEID                            INTEGER CONSTRAINT SYS_C001759741 NOT NULL,
 DIMVALUENAME                          VARCHAR2(80 BYTE) NULL,
 FACILITYID                            VARCHAR2(30 BYTE) NULL,
 DIMTYPEID                             INTEGER CONSTRAINT SYS_C001759742 NOT NULL,
 STARTDATE                             DATE NULL,
 ENDDATE                               DATE NULL
);

CREATE TABLE FACT_METRIC
(
 BUS_UNIT                              VARCHAR2(5 BYTE) NULL,
 METRIC_VALUE                          INTEGER NULL,
 UNIT_OF_MEASURE                       VARCHAR2(50 BYTE) NULL,
 REPORTED_DATE                         DATE NULL,
 PERIOD_START                          DATE NULL,
 PERIOD_END                            DATE NULL,
 FACTMETRICID                          INTEGER CONSTRAINT SYS_C001759743 NOT NULL,
 METRICID                              INTEGER NULL,
 METRICVERSIONID                    INTEGER NULL,
 DIMSETID                           INTEGER NULL,
 METRICPERIODTYPEID                 INTEGER NULL,
 REPORTEDVERSIONID                  NUMBER(38) NULL
);

CREATE INDEX FCT_METRIC_IDX1 ON FACT_METRIC
(BUS_UNIT ASC,PERIOD_START ASC,PERIOD_END ASC);

CREATE TABLE FACT_METRIC_RENUMBER
(
 BUS_UNIT                           VARCHAR2(5 BYTE) NULL,
 METRIC_VALUE                       INTEGER NULL,
 UNIT_OF_MEASURE                    VARCHAR2(50 BYTE) NULL,
 REPORTED_DATE                      DATE NULL,
 PERIOD_START                       DATE NULL,
 PERIOD_END                         DATE NULL,
 FACTMETRICID                       INTEGER CONSTRAINT SYS_C001760492 NOT NULL,
 METRICID                           INTEGER NULL,
 METRICVERSIONID                    INTEGER NULL,
 DIMSETID                           INTEGER NULL,
 METRICPERIODTYPEID                 INTEGER NULL,
 REPORTEDVERSIONID                  NUMBER(38) NULL
);

CREATE TABLE FACT_METRIC_STG
(
 BUS_UNIT                           VARCHAR2(5 BYTE) NULL,
 METRIC_VALUE                       INTEGER NULL,
 UNIT_OF_MEASURE                    VARCHAR2(50 BYTE) NULL,
 REPORTED_DATE                      DATE NULL,
 PERIOD_START                       DATE NULL,
 PERIOD_END                         DATE NULL,
 FACTMETRICID                       INTEGER CONSTRAINT SYS_C001759744 NOT NULL,
 METRICID                           INTEGER NULL,
 METRICVERSIONID                    INTEGER NULL,
 DIMSETID                           INTEGER NULL,
 METRICPERIODTYPEID                 INTEGER NULL,
 REPORTEDVERSIONID                  NUMBER(38) NULL
);
```

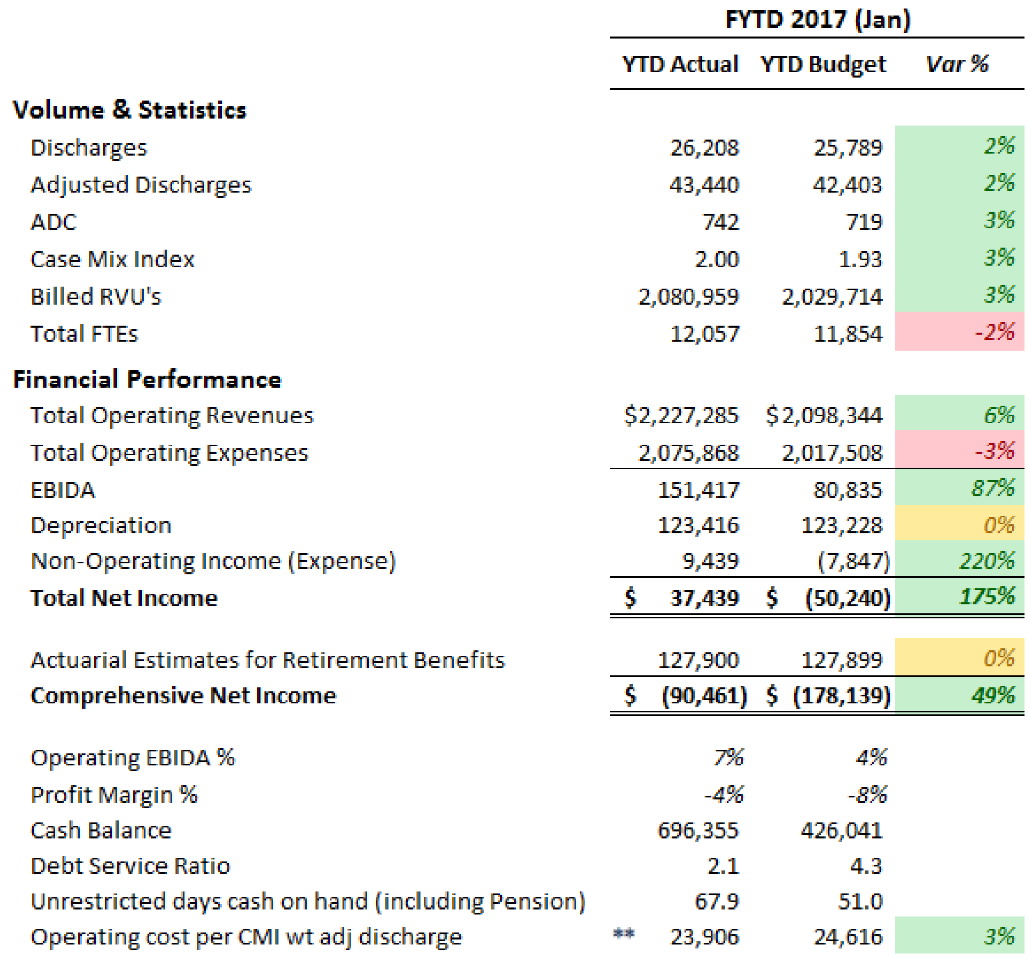
```
CREATE TABLE METRIC
(
  METRICID                          INTEGER NULL,
  EFFECTIVE_DATE                    DATE NULL,
  EXPIRATION_DATE                   DATE NULL,
  ACTIVE_FLAG                       CHAR(1 BYTE) NULL,
  NUMERATOR_DEFN                    VARCHAR2(500 BYTE) NULL,
  DENOMINATOR_DEFN                  VARCHAR2(500 BYTE) NULL,
  KEY_METRICS_USAGE                 VARCHAR2(20 BYTE) NULL,
  KEY_METRICS_EXAMPLE               VARCHAR2(20 BYTE) NULL,
  KEY_METRICS_ABBREVIATION          VARCHAR2(20 BYTE) NULL,
  BUSINESS_OWNER                    VARCHAR2(20 BYTE) NULL,
  TECHNICAL_OWNER                   VARCHAR2(20 BYTE) NULL,
  CALCULATION                       VARCHAR2(20 BYTE) NULL,
  UNIT_OF_MEASURE_ID                VARCHAR2(50 BYTE) NULL,
  SHORT_DESCRIPTION                 VARCHAR2(500 BYTE) NULL,
  LONG_DESCRIPTION                  VARCHAR2(20 BYTE) NULL,
  METRICVERSIONID                   INTEGER NULL,
  METRIC_NAME                       VARCHAR2(200 BYTE) NULL,
  METRIC_IND                        VARCHAR2(1 BYTE) NULL
);

CREATE TABLE METRIC_DIMENSION
(
  METRICDIMID                       INTEGER CONSTRAINT SYS_C001759745 NOT NULL,
  DIMVALUEID                        INTEGER NULL,
  DIMTYPEID                         INTEGER NULL,
  FACTMETRICID                      INTEGER NULL,
  STARTDATE                         DATE NULL,
  ENDDATE                           DATE NULL
);

CREATE TABLE METRIC_PERIOD_TYPE
(
  METRICPERIODTYPEID                INTEGER CONSTRAINT SYS_C001759746 NOT NULL,
  METRICPERIODNAME                  VARCHAR2(80 BYTE) NULL
);

CREATE TABLE METRICID
(
  METRICID                          INTEGER CONSTRAINT SYS_C001759747 NOT NULL,
  METRIC_NAME                          VARCHAR2(200 BYTE) NULL
);

CREATE TABLE MR_REJECT_CM_DIMVALUE
(
  METRIC_FACT_ID                       VARCHAR2(50 BYTE) NULL,
  METRIC_NAME                          VARCHAR2(50 BYTE) CONSTRAINT SYS_C001759776 NOT NULL,
  METRIC_VERSION_ID                    VARCHAR2(50 BYTE) NULL,
  DIMENSION_SET_ID                     VARCHAR2(50 BYTE) NULL,
  DIMTYPENAME                          VARCHAR2(50 BYTE) NULL,
  FACILITYID                           VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_ID                       VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_ID                       VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_ID                       VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_ID                       VARCHAR2(50 BYTE) NULL,
  METRICPERIODNAME                     VARCHAR2(50 BYTE) NULL,
  FIRST_DAY_OF_MONTH_DATE              DATE CONSTRAINT SYS_C001759777 NOT NULL,
  LAST_DAY_OF_MONTH_DATE               DATE NULL,
  METRIC_VALUE                         VARCHAR2(50 BYTE) NULL,
  REPORTED_DATE                        DATE NULL,
  REPORTED_SOURCE                      VARCHAR2(50 BYTE) NULL,
  ROWGEN                               INTEGER CONSTRAINT SYS_C001759778 NOT NULL
);

CREATE TABLE MR_REJECT_CM_METRIC
(
  METRICPERIODTYPEID                   INTEGER CONSTRAINT SYS_C001759748 NOT NULL,
  METRICPERIODNAME                     VARCHAR2(80 BYTE) NULL,
  DIMTYPENAME                          VARCHAR2(80 BYTE) CONSTRAINT SYS_C001759749 NOT NULL,
  DIMTYPEID                            INTEGER NULL,
  DIMVALUEID                           INTEGER CONSTRAINT SYS_C001759750 NOT NULL,
  DIMVALUENAME                         VARCHAR2(80 BYTE) CONSTRAINT SYS_C001759751 NOT NULL,
  FACILITYID                           VARCHAR2(30 BYTE) CONSTRAINT SYS_C001759752 NOT NULL,
  METRIC_FACT_ID                       VARCHAR2(50 BYTE) NULL,
  METRIC_NAME                          VARCHAR2(50 BYTE) CONSTRAINT SYS_C001759753 NOT NULL,
  METRIC_VERSION_ID                    VARCHAR2(50 BYTE) NULL,
  DIMENSION_SET_ID                     VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_ID                   VARCHAR2(50 BYTE) NULL,
  FIRST_DAY_OF_MONTH_DATE          DATE CONSTRAINT SYS_C001759754 NOT NULL,
  LAST_DAY_OF_MONTH_DATE           DATE NULL,
  METRIC_VALUE                     VARCHAR2(50 BYTE) NULL,
  REPORTED_DATE                    DATE NULL,
  REPORTED_SOURCE                  VARCHAR2(50 BYTE) NULL,
  ROWGEN                           INTEGER CONSTRAINT SYS_C001759755 NOT NULL
);

CREATE TABLE MR_REJECT_CM_PERIODTYPE
(
  DIMTYPENAME                      VARCHAR2(80 BYTE) CONSTRAINT SYS_C001759756 NOT NULL,
  DIMTYPEID                        INTEGER NULL,
  DIMVALUEID                       INTEGER CONSTRAINT SYS_C001759757 NOT NULL,
  DIMVALUENAME                     VARCHAR2(80 BYTE) CONSTRAINT SYS_C001759758 NOT NULL,
  FACILITYID                       VARCHAR2(30 BYTE) CONSTRAINT SYS_C001759759 NOT NULL,
  METRIC_FACT_ID                   VARCHAR2(50 BYTE) NULL,
  METRIC_NAME                      VARCHAR2(50 BYTE) CONSTRAINT SYS_C001759760 NOT NULL,
  METRIC_VERSION_ID                VARCHAR2(50 BYTE) NULL,
  DIMENSION_SET_ID                 VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_TYPE_ID              VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_ID                   VARCHAR2(50 BYTE) NULL,
  METRICPERIODNAME                 VARCHAR2(50 BYTE) NULL,
  FIRST_DAY_OF_MONTH_DATE          DATE CONSTRAINT SYS_C001759761 NOT NULL,
  LAST_DAY_OF_MONTH_DATE           DATE NULL,
  METRIC_VALUE                     VARCHAR2(50 BYTE) NULL,
  REPORTED_DATE                    DATE NULL,
  REPORTED_SOURCE                  VARCHAR2(50 BYTE) NULL,
  ROWGEN                                INTEGER CONSTRAINT SYS_C001759762 NOT NULL
);

CREATE TABLE MR_REJECT_DIM_VALUE
(
  METRICID                              INTEGER CONSTRAINT SYS_C001759763 NOT NULL,
  DIMVALUENAME                          VARCHAR2(5 BYTE) CONSTRAINT SYS_C001759764 NOT NULL,
  PERIOD_START                          TIMESTAMP(6) NULL,
  PERIOD_END                            TIMESTAMP(6) NULL,
  METRIC_VALUE                          NUMBER(26,3) NULL,
  FACTMETRICID                          INTEGER NULL
);

CREATE TABLE MR_REJECT_METRIC
(
  DIMVALUEID                            INTEGER CONSTRAINT SYS_C001759765 NOT NULL,
  DIMVALUENAME                          VARCHAR2(80 BYTE) NULL,
  DIMTYPEID                             INTEGER CONSTRAINT SYS_C001759766 NOT NULL,
  METRICID                              INTEGER CONSTRAINT SYS_C001759767 NOT NULL,
  PERIOD_START                          TIMESTAMP(6) NULL,
  PERIOD_END                            TIMESTAMP(6) NULL,
  METRIC_VALUE                          NUMBER(26,3) NULL,
  FACTMETRICID                          INTEGER NULL
);

CREATE TABLE MR_REJECT_WRVU_DIMVALUE
(
  METRIC_FACT_ID                        VARCHAR2(50 BYTE) NULL,
  METRICID                              VARCHAR2(50 BYTE) NULL,
  METRIC_VERSION_ID                     VARCHAR2(50 BYTE) NULL,
  DIMENSION_SET_ID                      VARCHAR2(50 BYTE) NULL,
  FACILITYID                            VARCHAR2(50 BYTE) NULL,
  DIMVALUENAME                          VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_TYPE_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_ID                        VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_TYPE_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_ID                        VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_TYPE_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_ID                        VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_TYPE_ID                   VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_ID                        VARCHAR2(50 BYTE) NULL,
  METRICPERIODTYPEID                   VARCHAR2(50 BYTE) NULL,
  FIRST_DAY_OF_MONTH_DATE              DATE NULL,
  LAST_DAY_OF_MONTH_DATE               DATE NULL,
  METRIC_VALUE                         VARCHAR2(50 BYTE) NULL,
  REPORTED_DATE                        DATE NULL,
  REPORTED_SOURCE                      VARCHAR2(50 BYTE) NULL,
  ROWGEN                               INTEGER CONSTRAINT SYS_C001759768 NOT NULL
);

CREATE TABLE MR_REJECT_WRVU_METRIC
(
  DIMTYPENAME                          VARCHAR2(80 BYTE) CONSTRAINT SYS_C001759769 NOT NULL,
  DIMTYPEID                            INTEGER NULL,
  DIMVALUEID                           INTEGER CONSTRAINT SYS_C001759770 NOT NULL,
  DIMVALUENAME                         VARCHAR2(80 BYTE) CONSTRAINT SYS_C001759771 NOT NULL,
  FACILITYID                           VARCHAR2(50 BYTE) NULL,
  METRIC_FACT_ID                       VARCHAR2(50 BYTE) NULL,
  METRICID                             VARCHAR2(50 BYTE) CONSTRAINT SYS_C001759772 NOT NULL,
  METRIC_VERSION_ID                    VARCHAR2(50 BYTE) NULL,
  DIMENSION_SET_ID                     VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_2_ID                       VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_3_ID                       VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_4_ID                       VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_TYPE_ID                  VARCHAR2(50 BYTE) NULL,
  DIMENSION_5_ID                       VARCHAR2(50 BYTE) NULL,
  METRICPERIODTYPEID                   VARCHAR2(50 BYTE) NULL,
  FIRST_DAY_OF_MONTH_DATE              DATE CONSTRAINT SYS_C001759773 NOT NULL,
  LAST_DAY_OF_MONTH_DATE               DATE NULL,
  METRIC_VALUE                         VARCHAR2(50 BYTE) NULL,
  REPORTED_DATE                        DATE NULL,
  REPORTED_SOURCE                      VARCHAR2(50 BYTE) NULL,
  ROWGEN                               INTEGER CONSTRAINT SYS_C001759774 NOT NULL
);

CREATE TABLE UNIT_OF_MEASURE
(
  UNIT_OF_MEASURE_ID                   INTEGER CONSTRAINT SYS_C001759775 NOT NULL,
  UNIT_OF_MEASURE_VALUE                VARCHAR2(80 BYTE) NULL
```

```
);

CREATE OR REPLACE PROCEDURE SP_FACT_METRIC_RENUMBER
AS
 TYPE C_REF IS REF CURSOR;

 FACTMETRICS                         C_REF;
 CURR_FACT                           FACT_METRIC_RENUMBER%ROWTYPE;
 PREV_FACT                           FACT_METRIC_RENUMBER%ROWTYPE;
 PREV_START_DATE                     DATE;
 CURR_END_DATE                       DATE;
 RVID                                NUMBER;

BEGIN
 OPEN FACTMETRICS FOR
 SELECT FCT.*
 FROM MRDATA.FACT_METRIC_RENUMBER FCT
 ORDER BY METRICID, BUS_UNIT, METRICPERIODTYPEID, PERIOD_START, REPORTED_DATE;

 -- Initialize report version id
 RVID := 1;

 LOOP
 -- Get current fact metric instance
 FETCH FACTMETRICS INTO CURR_FACT;

 -- Check if metric fact is the same
 IF (                                PREV_FACT.METRICID = CURR_FACT.METRICID
 AND PREV_FACT.BUS_UNIT = CURR_FACT.BUS_UNIT
 AND PREV_FACT.METRICPERIODTYPEID = CURR_FACT.METRICPERIODTYPEID
 AND PREV_FACT.PERIOD_START = CURR_FACT.PERIOD_START )
 THEN
 -- Still on the same metric fact

 -- Check if metric value are the same
 IF PREV_FACT.METRIC_VALUE = CURR_FACT.METRIC_VALUE
 THEN
 -- Fact metric instance is the same

 -- Set the report version id to negative value, mark for deletion
 UPDATE MRDATA.FACT_METRIC_RENUMBER FM
```

```
SET FM.REPORTEDVERSIONID = - REPORTEDVERSIONID
WHERE FM.FACTMETRICID = CURR_FACT.FACTMETRICID;

ELSE
-- metric value is not the same

-- Set the report version id to negative value
UPDATE MRDATA.FACT_METRIC_RENUMBER FM
SET FM.REPORTEDVERSIONID = RVID
WHERE FM.FACTMETRICID = CURR_FACT.FACTMETRICID;

-- Increment report version id
RVID := RVID + 1;

-- Track metric value
PREV_FACT.METRIC_VALUE := CURR_FACT.METRIC_VALUE;

END IF;

ELSE
-- Evaluating a different fact metric

-- Get the new previous project id
PREV_FACT.METRICID := CURR_FACT.METRICID;
PREV_FACT.BUS_UNIT := CURR_FACT.BUS_UNIT;
PREV_FACT.METRICPERIODTYPEID := CURR_FACT.METRICPERIODTYPEID;
PREV_FACT.PERIOD_START := CURR_FACT.PERIOD_START;

-- Reset report version id
RVID := 1;

-- Set the report version id to negative value
UPDATE MRDATA.FACT_METRIC_RENUMBER FM
SET FM.REPORTEDVERSIONID = RVID
WHERE FM.FACTMETRICID = CURR_FACT.FACTMETRICID;

-- Increment report version id
RVID := RVID + 1;

-- Track metric value
PREV_FACT.METRIC_VALUE := CURR_FACT.METRIC_VALUE;
```

```
  END IF;

  EXIT WHEN FACTMETRICS%NOTFOUND;

  END LOOP;

  -- Save the fact metrics
  COMMIT;

  -- Close the cursor
  CLOSE FACTMETRICS;

END;

/
```

Report Example



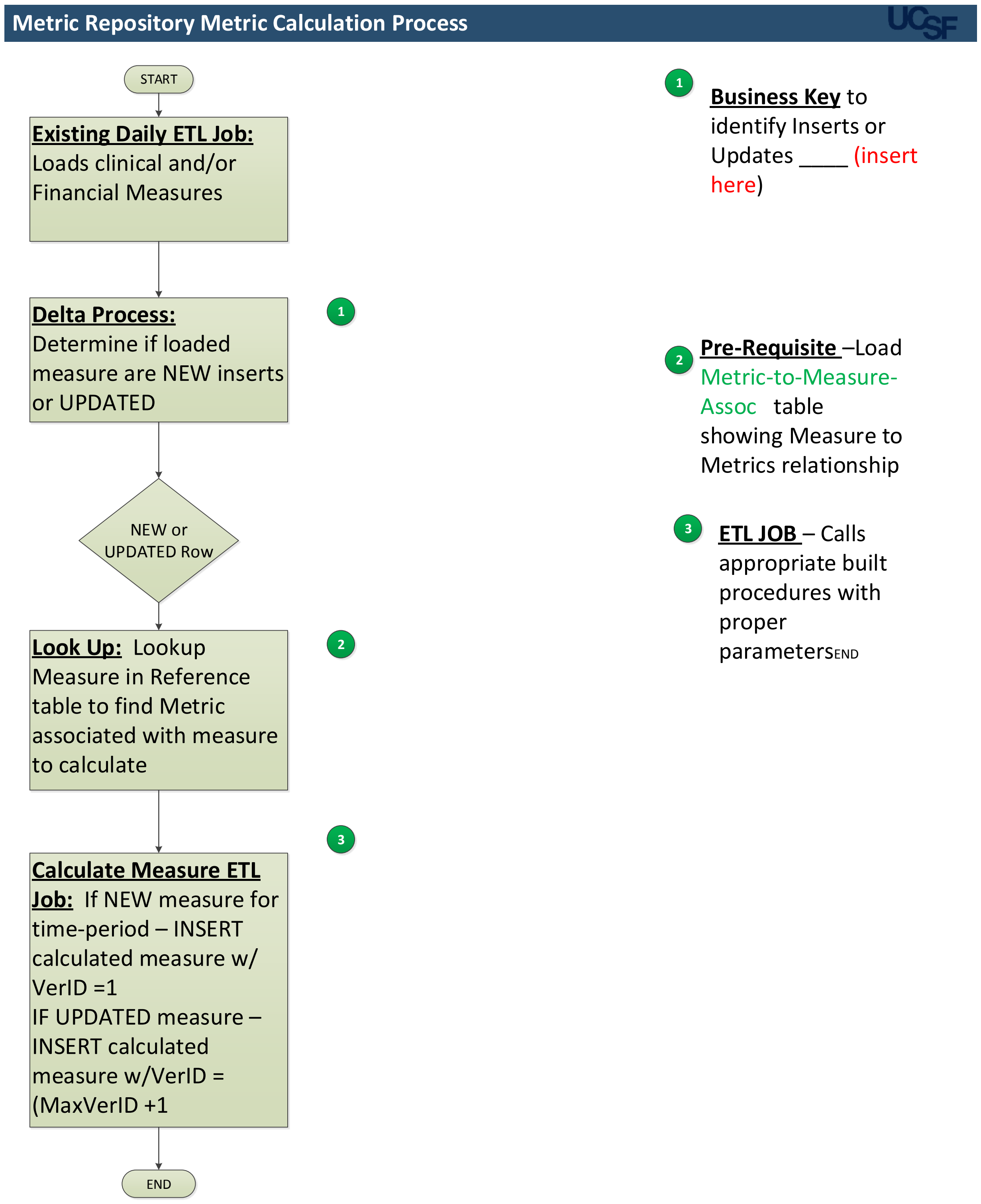
	FYTD 2017 (Jan)		
	YTD Actual	YTD Budget	*Var %*
Volume & Statistics			
Discharges	26,208	25,789	*2%*
Adjusted Discharges	43,440	42,403	*2%*
ADC	742	719	*3%*
Case Mix Index	2.00	1.93	*3%*
Billed RVU's	2,080,959	2,029,714	*3%*
Total FTEs	12,057	11,854	*-2%*
Financial Performance			
Total Operating Revenues	$2,227,285	$2,098,344	*6%*
Total Operating Expenses	2,075,868	2,017,508	*-3%*
EBIDA	151,417	80,835	*87%*
Depreciation	123,416	123,228	*0%*
Non-Operating Income (Expense)	9,439	(7,847)	*220%*
Total Net Income	**$ 37,439**	**$ (50,240)**	***175%***
Actuarial Estimates for Retirement Benefits	127,900	127,899	*0%*
Comprehensive Net Income	**$ (90,461)**	**$ (178,139)**	***49%***
Operating EBIDA %	*7%*	*4%*	
Profit Margin %	*-4%*	*-8%*	
Cash Balance	696,355	426,041	
Debt Service Ratio	2.1	4.3	
Unrestricted days cash on hand (including Pension)	67.9	51.0	
Operating cost per CMI wt adj discharge	** 23,906	24,616	*3%*

Calculated Metric process from Measures

Process Overview

Daily Run ETL Job as dependency to root measure load that kicks off MRDATA.MR_METRIC_CALC – function *to be built* that kicks off the individual functions (listed below) with proper parameters to generate the calculated Metrics and store calculated Measure results into MRDATA.FACT_METRIC. The process will also be able to re-build corrected or “updated” Metrics if one of the root Measures is modified.

Flow Chart


Metric Repository Metric Calculation Process
START
Existing Daily ETL Job: Loads clinical and/or Financial Measures
Delta Process: Determine if loaded measure are NEW inserts or UPDATED
NEW or UPDATED Row
Look Up: Lookup Measure in Reference table to find Metric associated with measure to calculate
Calculate Measure ETL Job: If NEW measure for time-period – INSERT calculated measure w/ VerID =1
IF UPDATED measure – INSERT calculated measure w/VerID = (MaxVerID +1
END
1
2
3
1 Business Key to identify Inserts or Updates ____ (insert here)
2 Pre-Requisite –Load Metric-to-Measure-Assoc table showing Measure to Metrics relationship
3 ETL JOB – Calls appropriate built procedures with proper parametersEND

Summary

Daily new ETL job runs and kicks off new function (MRDATA.MR_METRIC_CALC which calls all detail functions and store calculated Metrics in FACT_METRIC for the period being run by Business Unit.

Pre-Requisite – (1) *Seed* the MRDATA.METRIC_to_MEASURE_ASSOC table identifying root MEASURE that are associated with calculating a METRIC (see list of functions below)

Flagging root measures for update

In order to identify which previous run measures have been modified – (2) a new process needs to be developed to compare newly loaded measures into stageing with what was previously loaded into metric repository (Based on Busines Key (update here) to flag changed records –vs- insert records

Final Measure Job/Function

(3) This MRDATA.MR_METRIC_CALC Function will look up in the METRIC_to_MEASURE_ASSOC table to identify which functions need to be run. And, for NEW records do INSERT w/VERID = to 1 and with MODIFIED records do INSERT w/VERID = to MAX(CUR VERID) +1.

List of Current Detail Functions for each individual metric

MRDATA.MR_ADJ_DISCHRG

MRDATA.MR_ADC

MRDATA.MR_WRVU

MRDATA.MR_CMI

MRDATA.MR_FTES

MRDATA.MR_VISITS

MRDATA.MR_SURGERIES

MRDATA.MR_TOR

MRDATA.MR_MODOPEXD

MRDATA.MR_MODOPEBIDA

MRDATA.MR_DEPAMOR_EXP

MRDATA.MR_NONOPINLOSS

MRDATA.MR_MODNETINLOSS

MRDATA.MR_INCRETBENEXP

MRDATA.MR_TOTNETINLOSS

MRDATA.MR_OTHERCHGS

MRDATA.MR_NETPOSBEG

MRDATA.MR_NETPOSEND

MRDATA.MR_OPEREBIDA

MRDATA.MR_OPEREBIDAPCT

MRDATA.MR_PROFITMARGIN

MRDATA.MR_UNRCASHINV

MRDATA.MR_COREDAYSCASH

MRDATA.MR_OPEREXPCMI

MRDATA.MR_DEBTSERVRATIO