OREGON DEPARTMENT OF TRANSPORTATION

ITS Data Warehouse

Contents

1.0 ITS DW Overview Diagram 2

2.0 Definitions 2

3.0 Sources of ITS Warehouse Data 3

4.0 ITS DW JOBS 4

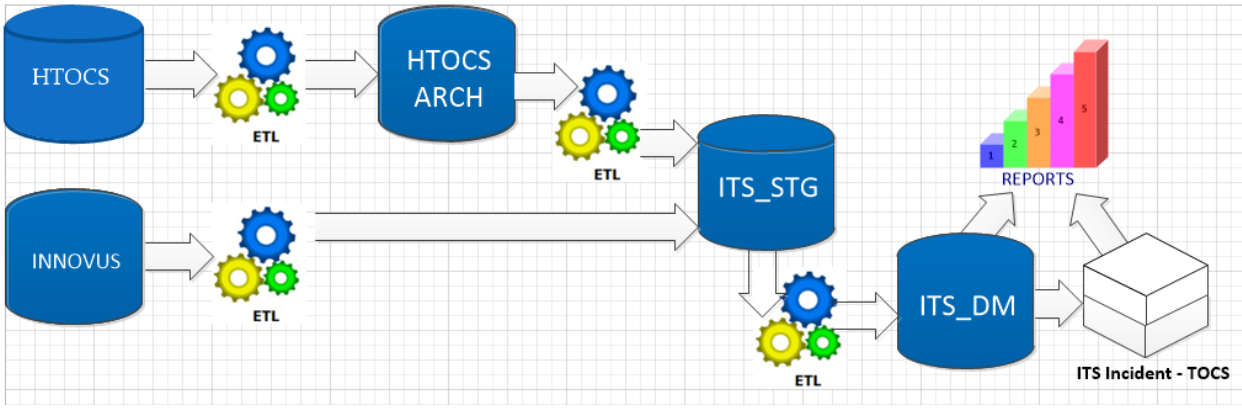
5.0 Destinations of ITS Data 4

6.0 Staging Database Standards 5

7.0 Staging SSIS Projects/Packages 9

8.0 ITS Incident – TOCS CUBE 10

1.0 ITS DW Overview Diagram



2.0 Definitions

- **Staging Database** – used to house data pulled from the source database.
- **Cube** – Similar to a database in that users connect to it, but the data in a cube is further aggregated and compressed for faster reporting.
- **Data Mart** – This is just a database that receives its information from various staging databases. SSIS packages are used to ETL (extract transform load) the data into the DataMart. The bulk of our SSIS packages truncate and load the data each night, but many incrementally load only the new data.
- **Dimension** – "dimension tables contain descriptive attributes (or fields) that are typically textual fields (or discrete numbers that behave like text)" (Wikipedia).
- **Fact** – "a Fact table consists of the measurements, metrics or facts of a business process. It is located at the center of a star schema or a snowflake schema surrounded by dimension tables" (Wikipedia).
- **Measure** – Measures are the core of the dimensional model and are data elements that can be summed, averaged, or mathematically manipulated. Fact tables are at the center of the star schema (which is a physically implemented dimensional model), and data marts are made up of multiple fact tables. Jun 17, 2007 (SQLMAG.com)

- **Star schema** – "In computing, the star schema is the simplest style of data mart schema and is the approach most widely used to develop data warehouses and dimensional data marts. The star schema consists of one or more fact tables referencing any number of dimension tables" (Wikipedia)
- **Snowflake schema** – "In computing, a snowflake schema is a logical arrangement of tables in a multidimensional database such that the entity relationship diagram resembles a snowflake shape. The snowflake schema is represented by centralized fact tables which are connected to multiple dimensions" (Wikipedia).
- **Bridge table** – Dimensional designs often need to accommodate multivalued dimensions. Patients can have multiple diagnoses. Students can have multiple majors. Consumers can have multiple hobbies or interests. Commercial customers can have multiple industry classifications. Employees can have multiple skills or certifications. Products can have multiple optional features. Bank accounts can have multiple customers. The multivalued dimension challenge is a natural and unavoidable cross-industry dilemma.http://www.kimballgroup.com/2014/05/design-tip-166-potential-bridge-table-detours/

3.0 Sources of ITS Warehouse Data

The source information listed below is based upon tables listed in the ITS_STG database and the supporting schemas the tables reside in, with the understanding that the schema should be a strong reference to the source database.

- DBO – 1 table – numbers(0 rows)
- HHAT – 3 tables – Phone call information (0 rows)
- HRAMP – 3 tables – Ramp information and 1 fact(0 rows)
- HTOCS/INCIDENT – most HTOCS database tables(rows range from 2.7million to 0)
- INNOVUS – 215 tables – the bulk of the tables have no rows.
- MOBILITY – 1 table – looks to contain UserStationData(0 rows)
- TELEMATICS – 3 tables – eventlog,inbound_log,vehicles(0 rows for all)
- TRIPCHECK – 3 tables – visit information – all tables have 0 rows

4.0 ITS DW JOBS

The jobs listed below are what kick off the ETL's to refresh/populate the ITS_STG database from its various sources.

- DW_STG_ITS_HTOCS
  - 14 steps
  - Scheduled daily at 12:01am
  - Kicks off
    - DW_DM_ITS_ITS_HTOCS
- DW_STG_ITS_Innovus
  - 28 steps
  - Scheduled daily at 10pm
  - Kicks off
    - DW_DM_ITS_ITS_INNOVUS
- DW_DM_ITS_ITS_HTOCS
  - 26 steps
  - No schedule – kicked off by DW_STG_ITS_HTOCS
  - Kicks off
    - DW_Cube_HTOCS
- DW_DM_ITS_ITS_INNOVUS
  - 28 steps
  - No schedule – kicked off by DW_STG_ITS_Innovus
- DW_Cube_HTOCS
  - 5 steps
  - No schedule – kicked off by DW_DM_ITS_ITS_HTOCS
  - Kicks off
    - Copy DW_HTOCS_OLAP job to copy db to query server
    - Copy HTOCS_ARCH job to copy db to query server

5.0 Destinations of ITS Data

Data from ITS_STG feeds into the ITS_DM database, which houses the following schemas/table counts (see below):

- ITS_DM – 2.3GB (1 data file / 1 log file)
- Security
    - DW_DEVELOPER_RO – read/view defn
    - ISB_ITS_PROD – read/view defn
    - QP_DW_ITS_MAINTENANCE_RO – read/**write**/view defn
    - QP_DW_ITS_RO – read/view defn/exec on 2 procs/update/delete on 1 tbl
- SCHEMA: INCIDENT
    - 42 tables – some dim and fact representation
- SCHEMA: INNOVUS
    - 215 tables – looks to be a 1-1 copy from the stg tables
- SCHEMA: MICROMAIN
    - 20 tables – many dim/fact tables, but most are empty – 1 backlogreport table
- SCHEMA: MOBILITY
    - 6 tables – 1 facts – 1 dim – 2 stg* tables – all row counts 0(zero)

6.0 Staging Database Standards

The information in this subject is derived from the Data Warehouse development standards document and future inquiries as to staging database standards should reference that document.

STAGING DATABASES

Staging databases should match their source databases as closely as possible. This allows a single Power Designer model to support the source system definitions and the staging definitions and allows for ease of troubleshooting in the warehouse.

When creating a new staging database the following are the expected standards to follow:

- No other objects will exist in staging databases except for tables derived from the source system and:
    - A stored procedure used to truncate table
        - Allows for ease of truncation in etl pkgs.
    - A stored procedure to drop indexes
        - Allows for faster inserts by not contending with indexes.
    - A stored procedure to create indexes
        - Allows for the re-creation of indexes that may have been dropped.

If the source system has large number of tables to be brought in, the use of BIML to derive the ETL to support source to stage data transfers can greatly speed up the process. Please note: BIML will not write any transformations so if there are any that are needed from the source to stage then manual intervention will need to take place. However, transformations from source to stage should be limited, if any at all, so as to alleviate troubleshooting source to stage issues.

•

No temporary and global temporary tables will be used for processing source to stage data/objects.

For staging loads that are incremental, there will be a need for an ETL schema and possibly _wrk tables to house the rows that will need to be updated in the staging table. In this case, the ETL schema will be created and the necessary _wrk table will be created to house the values that will be used to update the staging DB table before it's loaded into the data mart. The reason we use the _wrk is so that it easily differentiates between the actual table in the staging database and the transient data table in the ETL

schema. Although it is in a separate schema, we want to ensure its use is clearly delineated from the actual table.

For source to stage loads that are incremental there are various ways that this can be handled. Please refer to the SSIS Development Standards document on the Enterprise Data Management SharePoint site.

For standard reports to determine successful and failed runs, duration, and date/time of runs, refer to the SSISDB catalog, or the BIXpress database. Data stored therein can provide all information needed.

For non-database sources being loaded into the warehouse that we can easily determine the last refresh date from, we will use a date stored in our DW_CTL database/ SSIS_PROCCS_PARM table so that we can look up the last loaded file's refresh date, and only re-load the data if the date has changed between the files and the database table.

Views in the source system that are used to derive data will be made into tables in the staging database, with all definitions for all columns stored in Power Designer and TFS, and added as extended attributes.

Data types may shift from the source system to the staging database if by doing so SSIS development easier.

The rule is simple in one sense, but not so in the other -

- varchars should be storing variable length character data,
- chars should be storing non-variable length data.

- We see in lots of source systems where many char fields being used to store variable length strings, which can cause issues for SSIS. Therefore, if we see a character field doing this, in a source system, then we should change the datatype to a varchar. SSIS is the reason we may do conversion because it is case sensitive and white space sensitive, whereas sql server default collation is case insensitive.

Staging databases will not be deployed to the query server. They only exist on the DW processing server so that we ensure we maintain control of our staging environment.

Security in staging databases will consist of the following:

- ODOT\QP_DW_JOBCONTROL
  - Reader/writer/execute on the 3 procs listed above
- ODOT\DW_DEVELOPER_ADM
  - This account will only be used in development and given db_owner.
- ODOT\DW_DEVELOPER_CHG
  - This account will only be used in system test and given db_datareader, db_datawriter, view definition, and execute on the procs listed above for staging databases.
- ODOT\DW_DEVELOPER_RO
  - This account will only be used in production and given db_datareader and view definition.

All staging databases will adhere to the following naming conventions:

STG_<PROGRAM AREA>_<TYPE OF SOURCE>_<SOURCE DB NAME>

Example: STG_HR_SQL_DEHR – This database serves as a staging database for Human Resources data, which is housed in DEHR SQL database.

STG_HR_SQL_DEHR

HR = program area

SQL = type of source

DEHR = source database name

Standard TYPE OF SOURCE naming conventions are as follows

- SQL – SQL Server
- MYS – My SQL
- DB2 – DB2
- ORA – Oracle
- ACS – Access
- FF – Fixed width flat txt files
- XML – XML
- CSV – CSV
- XLS – Excel source

7.0 Staging SSIS Projects/Packages

All staging ssis projects and sql agent jobs will follow the same naming convention as the staging database. This will allow for a better understanding of what refreshes what. When more than one dtsx package is used to pull data from the source system the name should be suffixed with an _<number> in support of how the packages should be run, utilizing more descript naming conventions as possible.

There should be no complexity in the staging SSIS projects. All the ETL's should be doing are moving data from the source database into the staging database. This will rule out having to troubleshoot that area, and we can focus any problem tracking efforts in the DataMart etl's or the source database data. In addition, this ensures the SSIS extracts are as fast as possible.

Staging SSIS projects should be deployed under SSISDB\BIDW_STG, deriving its connection/configuration information from the BIDW_CONNECTIONS project. This will allow for the reuse of connections in staging etl's, data mart etl's, and outgoing etl's.

All staging SSIS projects should exist in TFS under TAD_MAINTENANCE/ODOT.BI.ETL/STAGING. Check-in and out process for this is the same as other code bases.

8.0 ITS Incident – TOCS CUBE

Data source: ITS_DM