Building ETL Pipe line to populate DWH

Index

1. Scope of the Project
2. Source Scripts:
3. Source Flow
4. Star Schema
5. Job Flow
6. Airflow
7. High Level Overview of the System
8. DB’s info
9. Process Steps
10. Queries
11. Scope of the Project

Create a Data warehouse and build ETL pipe line to load the data from source tables to Data warehouse tables.

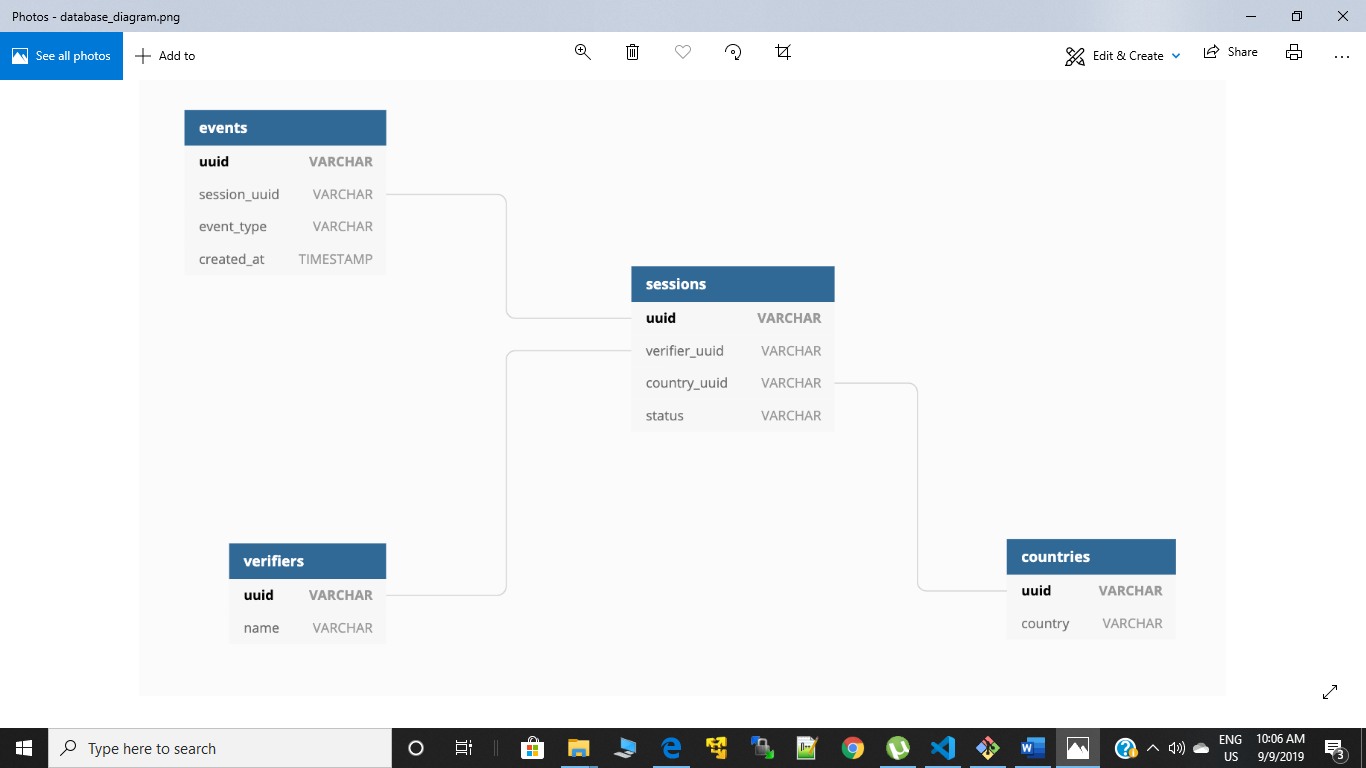
1. Source Scripts

Used below Source scripts to create DB, Tables and to populate Transactional DB. DB which is used Trans\_DB.

* create\_database.sql
* create\_tables.sql
* sessions.sql
* events.sql
* countries.sql
* verifiers.sql

1. Source Flow

Diagram provided for Mock Transactional Database:



1. Star Schema

Transformed above flow into Star Schema as below:

COUNTRIES\_DIM

UUID VARCHAR(PK)  
COUNTRY VARCHAR

EVENTS\_DIM

UUID VARCHAR(PK)  
SESSION\_UUID VARCHAR  
EVENT\_TYPE VARCHAR  
CREATED\_AT TIMESTAMP

SESSIONS\_DIM

UUID VARCHAR(PK)  
STATUS VARCHAR  
VERIFIER\_UUID VARCHAR  
COUNTRY\_UUID VARCHAR

TIME\_FACT

COUNTRY\_UUID(FK)  
VERIFIER\_UUID(FK)  
SESSION\_UUID(FK)  
EVENTS\_UUID(FK)  
TIME\_INTERVAL

VERIFIER\_DIM

UUID VARCHAR(PK)  
VERIFIER\_NAME VARCHAR

1. Job Flow:

Countries\_dim\_load.py

Countries\_data\_export.py

Verifiers\_dim\_load.py

Verifiers\_data\_export.py

Time\_Fact\_table\_Data\_load.py

Sessions\_dim\_load.py

Sessions\_data\_export.py

Events\_dim\_load.py

Events\_data\_export.py

1. Airflow

Used Airflow to execute the Data pipeline.

Script for Airflow: Airflow\_code.py

Steps in Airflow file:

1. Imported required libraries
2. Define Default Arguments.
3. Instantiate DAG
4. Layout all the tasks in the workflow.
5. Setting up the Dependency.

Note: Developed above process in Autosys scheduler also, Prepared Autosys JIL’s to executed above process their as well.

1. High Level Overview of the System

TRANS\_DB

Export data to a CSV file

CSV File

Load Dimension and Fact table from CSV file

Reconciliation Step after load

VERIFF\_DWH\_DB

1. DB’s INFO

Transactional DB: Trans\_DB  
Data Warehouse DB: veriff\_dwh\_db

Tables Created under Trans\_DB:

1. Countries
2. Verifiers
3. Sessions
4. Events

Dimension Tables Created under veriff\_dwh\_db:

1. Countries\_dim
2. Verifiers\_dim
3. Sessions\_dim
4. Events\_dim

Fact Table Created under Veriff\_dwh\_db:

* Time\_fact

Temporary table created to populate amount of time to process a session:

* Time\_temp

CSV Files created for Data Export:

1. Countries.csv
2. Verifiers.csv
3. Sessions.csv
4. Events.csv
5. Process Steps

Process followed to develop ETL pipe line:

* Created a Transactional DB called **Trans\_DB** in PostgreSQLL**,** created given Tables under that DB and Populated those tables with the Data provided.
* Created a DWH DB called veriff\_dwh\_db, and created Required Dimension tables and Fact tables under it.
* Created ETL jobs as below.
  + - Exporting data from source DB[**Trans\_DB**] to a CSV file  
       1. Countries\_data\_export.py: Connect to the Trans\_DB in PostgreSQL

Export data into Countries.csv file from countries table.

2. Verifiers\_data\_export.py: Connect to the Trans\_DB in PostgreSQL  
 Export data into Verifiers.csv file from Verifiers table.

3. Sessions\_data\_export.py: Connect to the Trans\_DB in PostgreSQL  
 Export data into Sessions.csv file from Sessions table.

4. Events\_data\_export.py: Connect to the Trans\_DB in PostgreSQL  
 Export data into Events.csv file from Events table.

* Loaded data into Dim Tables table which are created under DWH DB[**veriff\_dwh\_db**]

1. Countries\_dim\_load.py: Connect to the veriff\_dwh\_db in PostgreSQL, Populate Countries\_dim table under veriff\_dwh\_db from Countries.csv file.

Validation step to ensure that the number of rows in the table is matching with the number of records in file [Reconciliation Step].

1. Verifiers\_dim\_load.py: Connect to the veriff\_dwh\_db in PostgreSQL, Populate Verifiers\_dim table under veriff\_dwh\_db from Verifiers.csv file.

Validation step to ensure that the number of rows in the table is matching with the number of records in file [Reconciliation Step].

1. Sessions\_dim\_load.py: Connect to the veriff\_dwh\_db in PostgreSQL, Populate Sessions\_dim table under veriff\_dwh\_db from Sessions.csv file.

Validation step to ensure that the number of rows in the table is matching with the number of records in file [Reconciliation Step].

1. Events\_dim\_load.py: Connect to the veriff\_dwh\_db in PostgreSQL, Populate Events\_dim table under veriff\_dwh\_db from Events.csv file.

Validation step to ensure that the number of rows in the table is matching with the number of records in file [Reconciliation Step].

* Implemented star schema with the fact table focusing on the amount of time it takes to process a session. [**Time\_Fact\_table\_Data\_load.py**]

1. Connect to the veriff\_dwh\_db in PostgreSQL.
2. Create a Temporary table called time\_temp.
3. populate it with amount of time taken to process a session.
4. Populate the FACT Table[**time\_fact**]
5. Drop the Temporary table called time\_temp
6. Queries
7. What was the average processing time per country?

select c.country,to\_timestamp(avg(18000 + cast(extract(epoch from t.time\_interval::time) as integer)))::time from countries\_dim c inner join time\_fact t on t.country\_uuid=c.uuid group by 1;

1. Which verifier processed the sessions quickest for New Zealand?

select v.verifier\_name,min(t.time\_interval) from verifiers\_dim v inner join time\_fact t on v.uuid=t.verifier\_uuid inner join countries\_dim c on c.uuid = t.country\_uuid where c.country='New Zealand' group by 1;