**Creating an ETL Pipeline**

**Introduction**

This assignment involved the implementation and population of a data warehouse using Oracle Cloud and the integration of Apache Hop as an ETL (Extract, Transform, Load) tool. The focus was on setting up the data warehouse schema, loading dimension tables, and preparing a pipeline to load a fact table using lookup operations and automated data processing.

**Activities Performed in the ETL Pipeline Creation Process**

**Preparation and Setup**:

The assignment began by configuring Oracle Cloud’s Autonomous Data Warehouse and setting up the necessary tables using provided DDL (Data Definition Language) scripts. These included dimensions such as Dim\_Product, Dim\_Customer, Dim\_Date, and a FactSales table.

Existing tables from a prior lab exercise were either removed or renamed to avoid conflicts during the re-creation of the data warehouse.

**Loading Dimension Tables:**

The Dim\_Date table was populated first, as it serves as a “conformed dimension” with a fixed range of dates. The original date script was modified to ensure that the dimension spanned from January 2016 to December 2026. Screenshots of both the SQL script and query results were included in the report.

Data for Dim\_Customer and Dim\_Product was manually inserted to simulate real-world scenarios of data addition. The Dim\_Customer table was populated with sample customers using INSERT statements, and five products were added to Dim\_Product with specific details like category and brand.

**Apache Hop ETL Configuration**:

Apache Hop was downloaded and configured to interact with the Oracle Cloud Data Warehouse. Key steps included the setup of JDBC drivers and security files from Oracle’s client credentials.

A simple ETL pipeline was built to automate data extraction and loading. This involved using nodes such as "Table Input" for retrieving data from the data warehouse and "Text File Output" for exporting results.

**Advanced ETL Operations:**

The assignment required handling slowly changing dimensions (SCDs). Using Apache Hop’s "Dimension Lookup/Update" node, the Dim\_Product table was updated based on changes in product information, implementing Type 1 and Type 2 SCD techniques where necessary.

Another pipeline was created to manage updates to the Dim\_Customer table, involving the specification of business and surrogate keys and ensuring historical attributes were accurately maintained.

**Loading Fact Tables:**

The SalesUpdate.csv file was used to load the FactSales table. This process required lookups for foreign keys in the Dim\_Product and Dim\_Customer tables. Apache Hop’s "Stream Lookup" node was employed to map business keys from the CSV to the corresponding surrogate keys in the dimensions, ensuring referential integrity.

A "Filter Rows" node was utilized to remove records with missing lookup values, preventing incomplete or erroneous data from entering the fact table.

**Quality Assurance and Final Steps:**

The successful execution of ETL pipelines was confirmed by running queries in Oracle Cloud and validating the integrity of the loaded data. Screenshots of the output from these queries are included below to demonstrate the correctness of the data transformations and load processes.

**Database Diagram**

A screenshot of a computer

Description automatically generated

**Based on the diagram generated, what is this database missing that you’d expect to see? Why might it be missing this component?**

Answer - Based on the generated diagram, the database appears to be missing a well-defined relationship or foreign key constraint between the fact table and the dimension tables. This might be missing because the focus of the assignment is on ETL operations rather than enforcing referential integrity within the database schema itself. Additionally, the design may prioritize flexibility or performance considerations for the ETL process.

**Loading the Dimensions Table**

A screenshot of a computer

Description automatically generated

Fig: Results for ‘SELECT \* FROM DIM\_DATE’ before updating the script.

A screenshot of a computer

Description automatically generatedFig: Results for ‘SELECT \* FROM DIM\_DATE’ after updating the script.

A screenshot of a computer

Description automatically generatedFig: Results for ‘SELECT \* FROM DIM\_PRODUCT’.

A screenshot of a computer

Description automatically generatedFig: Dimensional Loader Pipeline

A screenshot of a computer

Description automatically generatedFig: Results for ‘SELECT \* FROM DIM\_PRODUCT’ (Slowly Changing Dimension)

A screenshot of a computer

Description automatically generatedFig: Pipeline for Product Update Input

A screenshot of a computer

Description automatically generatedFig: Pipeline for Customer Update Input

A screenshot of a computer

Description automatically generated

Fig: Results for “SELECT \* FROM DIM\_CUSTOMER”

**Loading Facts Table**

A screenshot of a computer

Description automatically generatedFig: Fact Table

A screenshot of a computer

Description automatically generatedFig: Table Input

A screenshot of a computer

Description automatically generatedFig: Stream Lookup 2

**Write 1-2 sentences detailing why we don’t need to. You’ll be able to figure this out likely by looking at the data in the fact CSV and the date dimension.**

Answer - We don’t require a lookup for the date dimension because the fact table already contains date values that directly correspond to entries in the pre-populated date dimension. Since the date dimension encompasses the complete date range, each record in the fact table will automatically align with an existing entry in the date dimension, eliminating the need for additional lookups.

A screenshot of a computer

Description automatically generatedFig: Results for “SELECT \* FROM FACT\_SALES”

**Final Pipeline**

A screenshot of a computer

Description automatically generatedFig: Completed Pipeline

References:

* ChatGPT.com
* YouTube.com