Business Intelligence and Data Warehousing (ANL408)

- By Sabarish Nair

Recap from last week....

- DWH Common Use cases
- Optimizing a Data Warehouse
- Indexes
- B-Tree Indexes
- Bitmap Indexes
- Guidelines for Indexes
- Setting Indexes
- Visualization and Reporting
- Practical: Create Dimension Tables
- Practical: Create Fact Tables
- Practical: Insert Data in Dimension Tables
- Practical: Insert Data in Fact Tables
- Practical: Connect Postgre SQL to Power BI

Typical Roles in DWH Project

- Project Sponsor
- Project Manager
- Functional/Business Analyst
- DW BI Architect
- Data Modeler
- BI Developers
- DB Admins
- Data Architect/Data Warehouse Architect/BI Architect/Solution Architect

Question

- You have been provided with a sample CSV file containing data related to sales transactions. Your task is to perform the following tasks using PostgreSQL:
  - Extract (Overall 10 marks)
    - Create a database in named "DataWarehouseANL408". (Share the script for database creation) [2 marks]
    - Create a schema named "Staging" within the database created. (Share the script for schema creation) [2 marks]
    - Create a table in staging schema to store the data imported from CSV. (Share the script for table creation) [3 marks]
    - Load the CSV data into the table created in the staging schema. State the total number of rows and columns in the table. [3 marks]

Question [cont…]

- Transform (Overall 10 Marks)
  - Copy all the data from the staging table to a new temporary table within the staging schema (Share the script created for the same) [1 mark]
  - Write an SQL Query to get the total count of records with negative unit price. (Share the SQL Query) [1 mark]
  - Write an SQL Query to get the total count of records with NULL or Blank product names. (Share the SQL Query) [1 marks]
  - Write an SQL Query to get the total count of records with NULL or Blank Customer names. (Share the SQL Query) [1 marks]
  - Write an SQL Query to replace negative price values by its absolute values. (Share the SQL Query) [2 marks]
  - Write an SQL Query to replace all the product names with NULL or BLANK to 'UNKNOWN'. (Share the SQL Query) [2 marks]
  - Write an SQL Query to replace all the customer names with NULL or BLANK to 'UNKNOWN'. (Share the SQL Query) [2 marks]

Question [cont…]

- Identify the fact and dimension tables (Overall 10 Marks)
  - Identify the fact table (Share the table name and its associated columns)[4 mark]
  - Identify the dimension tables (Share the table names and its associated columns)[6 mark]
- Load (20 Marks)
  - Create the fact table in "public" schema within the same database. (Share the table creation script) [4 marks]
  - Create the dimension tables in "public" schema within the same database. (Share the table creation scripts) [6 marks]
  - Populate the dimension tables from the staging schema. (Share the SQL script) [6 marks]
  - Populate the fact tables. (Share the SQL script) [4 marks]

Solution

- Database creation

```
CREATE DATABASE "DataWarehouseANL408"
    WITH
    OWNER = postgres
    ENCODING = 'UTF8'
    LOCALE_PROVIDER = 'libc'
    CONNECTION LIMIT = -1
    IS_TEMPLATE = False;
```

Solution [cont…]

- Creation of Schema

```
CREATE SCHEMA "Staging"
    AUTHORIZATION postgres;
```

Solution [cont…]

- Create staging table

```
CREATE TABLE "Staging".transaction_sales
(
    transaction_id integer NOT NULL,
    product_id character varying,
    product_name character varying,
    quantity integer,
    unit_price numeric(5, 2),
    customer_id character varying,
    customer_name character varying,
    transaction_date date,
    PRIMARY KEY (transaction_id)
);
```

Solution [cont...]

- Load CSV data into table

```
SELECT transaction_id, product_id, product_name, quantity, unit_price,
  customer_id, customer_name, transaction_date
   FROM "Staging".transaction_sales;
```

Columns: 8

Rows: 10

Solution [cont…]

- Copy data from staging table to a temporary table.

```
CREATE TABLE temp_transactionsales  AS
SELECT * FROM "Staging"."transaction_sales";
```

Solution [cont...]

- Get count of records with negative unit price.

```
SELECT COUNT(1)
FROM temp_transactionsales
WHERE unit_price < 0;
```

- Get Count of Records with blank or null product names

```
SELECT COUNT(1)
FROM temp_transactionsales
WHERE product_name is NULL or product_name = 'NULL'
```

Solution [cont...]

- Get Count of Records with blank or null customer names

```
SELECT COUNT(1)
FROM temp_transactionsales
WHERE customer_name is NULL or customer_name = 'NULL'
```

- Replace negative price with its absolute value

```
UPDATE temp_transactionsales
SET unit_price = ABS(unit_price)
WHERE unit_price < 0;
```

Solution [cont…]

- Replace NULL or BLANK product names with 'UNKNOWN'

```
UPDATE temp_transactionsales
SET product_name = 'UNKNOWN'
WHERE product_name is NULL or product_name = 'NULL'
```

- Replace NULL or BLANK customer names with 'UNKNOWN'

```
UPDATE temp_transactionsales
SET customer_name = 'UNKNOWN'
WHERE customer_name is NULL or customer_name = 'NULL'
```

Fact Tables

- Sales_Fact
  - Transaction_id (PK)
  - Product_id (FK)
  - Quantity
  - Unit_Price
  - Customer_id (FK)
  - Date_id (FK)

Dimension Tables

- Date_Dim
  - Date_id (PK)
  - Date
  - Day
  - Month
  - Year
- Product_Dim
  - Product_id (PK)
  - Product_Name
- Customer_Dim
  - Customer_Id (PK)
  - Customer_Name

Create Dimension Tables in "Public Schema"

- Date_Dim

```
CREATE TABLE Date_Dim (
    date_id SERIAL PRIMARY KEY,
    date DATE,
    day INT,
    month INT,
    year INT
);
```

- Product_Dim

```
CREATE TABLE Product_Dim (
product_pk SERIAL PRIMARY KEY,
product_id VARCHAR(100),
product_name VARCHAR(100)
);
```

Create Dimension Tables in "Public Schema"

- Customer Dim Table

```
CREATE TABLE Customer_Dim (
customer_pk SERIAL PRIMARY KEY,
customer_id VARCHAR(100),
customer_name VARCHAR(100)
);
```

Create Sales Fact Table

- Sales Fact Table

```
CREATE TABLE Sales_Fact (
    transaction_id INT PRIMARY KEY,
    date_id INT REFERENCES Date_Dim(date_id),
    product_pk INT REFERENCES Product_Dim(product_pk),
    customer_pk INT REFERENCES Customer_Dim(customer_pk),
    price NUMERIC(10, 2),
    quantity INT
);
```

Populate Dimension Tables

- Populate Date_Dim

```
INSERT INTO Date_Dim (date, day, month, year)
SELECT DISTINCT transaction_date, EXTRACT(day
FROM transaction_date), EXTRACT(month FROM
transaction_date), EXTRACT(year FROM transaction_date)
FROM "Staging"."temp_transactionsales";
```

Populate Dimension Tables

- Populate Product_Dim

  INSERT INTO Product_Dim (product_id,product_name)

  SELECT DISTINCT product_id,product_name

  FROM "Staging"."temp_transactionsales";

- Populate Customer Dim

  INSERT INTO Customer_Dim (customer_id, customer_name)

  SELECT DISTINCT customer_id, customer_name

  FROM "Staging"."temp_transactionsales";

Populate Fact Table

```
INSERT INTO Sales_Fact (transaction_id, date_id, product_pk, customer_pk, price, quantity)
SELECT d.date_id,
    p.product_id,
    c.customer_id,
    s.unit_price,
    s.quantity
FROM "Staging"."temp_transactionsales" s
JOIN Date_Dim d
    ON s.transaction_date = d.date
JOIN Product_Dim p
    ON UPPER(s.product_name) = UPPER(p.product_name)
JOIN Customer_Dim c
    ON UPPER(s.customer_name) = UPPER(c.customer_name)
```


thank you