DS 206: GROUP PROJECT #2

PART 1. RELATIONAL DATABASE CREATION (40)

1. Create an empty script file named **database_creation.sql.** Add a DDL query to create an empty database named **Orders_RELATIONAL_DB** and run it (within SQL Server).
2. Store the database connection configurations in the **sql_server_config.cfg** file (feel free to name the config file differently).
3. Create parametrized SQL scripts named **create_table_{}.sql** for creating empty tables for each sheet in the **raw_data_source.xlsx** data file.
4. Establish the primary and foreign key constraints based on Table 1 and Table 2 respectively.
5. Create parametrized SQL scripts named **insert_into_{}.sql** to ingest data into the empty tables from **raw_data_source.xlsx.**
6. Write Python utilities (utils.py), tasks (tasks.py) and flow ({}_flow.py) to create a unified process for creating the tables for the relational database and ingesting data into them. Ensure the atomicity and the reproducibility of Your Python scripts.
7. **IMPORTANT: Make sure to have informative logging in place. You may use formatted print() statements or the logging module of Python to achieve this.**

Table 1. Primary Key Constraints

schema_name	table_name	columns
dbo	Categories	CategoryID
dbo	Customers	CustomerID
dbo	Employees	EmployeeID
dbo	Order Details	OrderID, ProductID
dbo	Orders	OrderID
dbo	Products	ProductID
dbo	Region	RegionID
dbo	Shippers	ShipperID
dbo	Suppliers	SupplierID
dbo	Territories	TerritoryID

Table 2. Foreign Key Constraints

FK_Table	FK_Column	PK_Table	PK_Column
Employees	ReportsTo	Employees	EmployeeID
Order Details	OrderID	Orders	OrderID
Order Details	ProductID	Products	ProductID
Orders	CustomerID	Customers	CustomerID
Orders	EmployeeID	Employees	EmployeeID
Orders	ShipVia	Shippers	ShipperID
Orders	TerritoryID	Territories	TerritoryID
Products	CategoryID	Categories	CategoryID
Products	SupplierID	Suppliers	SupplierID
Territories	RegionID	Region	RegionID

PART 2. DIMENSIONAL DATABASE (DATA WAREHOUSE) CREATION & POPULATION (60)

1. Add a new DDL query in the **database_creation.sql** file to create an empty database named **Orders_DIMENSIONAL_DW** and run it (within SQL Server).
2. Store the database connection configurations in the **sql_server_config.cfg** file (feel free to name it differently).
3. Create parametrized SQL scripts named **create_table_dim_{}.sql** for creating empty dimension and fact tables for each table in the relational db. The dimension tables types for each group are allocated in **Table 1** (see at the end of this description file).Make sure to have all necessary surrogate (SK) columns (both PK and FK).
4. Establish the primary and foreign key constraints respectively.
5. Create a SQL script to create a date dimension table that is consistent with the date format provided by the relational table.
6. Create parametrized SQL scripts named **update_dim_{}.sql** to ingest data into the empty dim tables from respective tables in the relational db.
7. Create a parametrized SQL script named **updated_fact_{}.sql** to ingest data into the empty fact tables.
8. Make all the necessary changes to the Python utilities (utils.py), tasks (tasks.py) and flow ({}_flow.py) files in order to add more steps to the unified process for

creating tables in the dimensional database and ingesting data into them. Keep in mind that your tasks have to be generic enough to be used for updating data in both relational and dimensional databases. In other words, you will need to have one function (but multiple queries) for creating tables, one task for updating dimension tables etc.

9. **IMPORTANT: Make sure to have informative logging in place. You may use formatted print() statements or the logging module of Python to achieve this.**

PART 3. BI DASHBOARD CREATION USING POWER BI (up to 5%, depending on the output quality)

1. Your solution should be based on the dimensional data store.
2. Your solution should be **at least 2 pages** long (please refer to Power BI pages).
3. Your solution should contain
    1. **at least 3-4 topic-consistent visualizations** on each page,
    2. **at least 2 slicers** on each page.
4. Your solution should have **at least 5 distinct DAX measures**:
    1. at least 8 Date Intelligence measures
    2. at least 5 CALCULATE() + FILTER() combination,
5. Your solution should have **at least 2 tabular**, at least **2 categorical**, and **at least 1 trend** (time series) visualizations.
6. Your dashboard must contain **at least 1 tooltip**.
7. Each page on Your solution should have a separate **Reset All Slicers button**.
8. The information provided on each page should be consistent and holistic.

YOUR SUBMISSION SHOULD INCLUDE THE FOLLOWING COMPONENTS:

- **queries** folder with all the necessary SQL scripts (including **database_creation.sql**),
- **utils.py** file for storing all the necessary utilities,
- **config.py** file for storing all the flow-level configurations,
- **tasks.py** file for storing all the distinct tasks,
- **sql_server_config.cfg** for storing the database connection configurations,
- **flow.py** file for storing the unified process for populating the relational and dimensional databases,
- a PBIX file containing the dashboard created using the dimensional data store (**extra-credit**).

GROUP ID	DIMENIONS
GROUP 1	DimCategories SCD1 with delete DimCustomers SCD2 DimEmployees SCD1 with delete DimProducts SCD4 DimRegion SCD3 (one attribute, current and prior) DimShippers SCD1 DimSuppliers SCD4 DimTerritories SCD3
GROUP 2	DimCategories SCD1 DimCustomers SCD2 DimEmployees SCD1 with delete DimProducts SCD1 DimRegion SCD4 DimShippers SCD1 with delete DimSuppliers SCD3 (one attribute, current and prior) DimTerritories SCD4
GROUP 3	DimCategories SCD1 with delete DimCustomers SCD4 DimEmployees SCD1 DimProducts SCD4 DimRegion SCD2 DimShippers SCD3 (one attribute, current and prior) DimSuppliers SCD3 (one attribute, current and prior) DimTerritories SCD2
GROUP 4	DimCategories SCD1 with delete DimCustomers SCD3 (one attribute, current and prior) DimEmployees SCD2 DimProducts SCD1 with delete DimRegion SCD1 DimShippers SCD1 DimSuppliers SCD4 DimTerritories SCD2

GROUP 5	DimCategories SCD1 DimCustomers SCD2 DimEmployees SCD3 (one attribute, current and prior) DimProducts SCD3 (one attribute, current and prior) DimRegion SCD3 (one attribute, current and prior) DimShippers SCD1 with delete DimSuppliers SCD4 DimTerritories SCD1
GROUP 6	DimCategories SCD3 (one attribute, current and prior) DimCustomers SCD3 (one attribute, current and prior) DimEmployees SCD2 DimProducts SCD4 DimRegion SCD1 with delete DimShippers SCD3 (one attribute, current and prior) DimSuppliers SCD1 with delete DimTerritories SCD3 (one attribute, current and prior)