DS-2002 – Data Project 2 (Course Capstone)

25 points
The goal of the second data project, building upon the first project, is to further demonstrate (1) an understanding of and (2) competence creating and implementing basic data science systems such as pipelines, scripts, data transformations, APIs, databases and cloud services. Submit your project in your GitHub Repo or file drop on Collab.

Data Projects must be done individually.

Putting it All Together: Data Integration & Analysis

Deliverable: Using Azure Databricks, design and populate a dimensional Data Lakehouse that represents a simple business process of your choosing. Examples might include retail sales, inventory, procurement, order management, transportation or hospitality bookings, medical appointments, student registration and/or attendance. You may select any business process that interests you, but remember that a dimensional Data Lakehouse provides for the post hoc summarization and historic analysis of business transactions that reflect the interaction between various entities (e.g., patients & doctors, retailers & customers, students & schools/classes, travelers & airlines/hotels).

The most straight-forward approach is to identify an existing OLTP example database wherein all required data relationships already exist; however, you may choose to populate your Data Lakehouse using data from multiple sources as long as you can successfully use their business keys (e.g., customer code, product code) to establish the appropriate relationships between the Fact and Dimension tables.

Your project should demonstrate your understanding of the differing types of relational data systems (OLTP/OLAP), and how data can be **extracted** from various source systems (structured, semi-structured, unstructured), **transformed** (cleansed, integrated), and then **loaded** into a destination system that's optimized for post hoc diagnostic analysis. Your project should also demonstrate your knowledge of data integration patterns like ETL, ELT and ELTL, and architectures (e.g., lambda or kappa) for integrating batch and real-time (streaming) data sources.

Design Requirements:
Your solution (database schema) needn't be complex, but should meet the following requirements:

- A **Date dimension** to enable the analysis of the business process over various intervals of time *(the code for creating this in MySQL and Microsoft SQL Server has already been provided for you).*
- At least 3 additional dimension tables (e.g., customers, employees, products)
- At least 1 fact table that models the business process (e.g., sales, reservations, bookings)
- Your solution must populate its dimensions using data originating from multiple sources:
    - A relational database like Azure MySQL, or Azure SQL Server
    - A NoSQL database like MongoDB Atlas, or Azure Cosmos DB
    - Files (e.g., CSV) from a cloud-based file system; like the Databricks File System (DBFS)

- Your solution must integrate datum of differing granularity (i.e., static and near real-time)
- Your solution must include results that demonstrate the business value of your solution. For example, a query (SELECT statement) that summarizes transaction details by customer, product, etc.

Functional Requirements:

1. Your solution must demonstrate at least one batch execution (i.e., use sample source data [SQL, NoSQL and file system] to demonstrate loading at least one incremental data load).
2. Your solution must demonstrate accumulating data that originates from a real-time (streaming) data source for a predetermined interval (mini-batch), integrating it with reference data, and then using the product as a source for populating your dimensional Data Lakehouse. (i.e., implement the Databricks bronze, silver, gold architecture).
    a. Use the Spark AutoLoader to demonstrate integrating *streaming* data (using separate JSON files) for at least 3 intervals. *This is most easily accomplished by segmenting the Fact table source data into 3 ranges and exporting them into 3 separate JSON files.*
    b. Illustrate the relationships between the "real-time" fact data and the static reference data. This is accomplished by joining fact and dimension tables at the Silver table phase.
3. Use a Databricks Notebook to execute all data integration, object creation and query execution.
4. Please submit all code, and other artifacts, in a GitHub repository in your account.

Note: You may utilize any combination of Cloud service technologies. For example, you can collect streaming data from a source API on the Internet, integrate it with reference data that's stored in another Cloud hosted database service (e.g., Mongo DB Atlas) using Databricks, and then load it into your dimensional Data Lakehouse that's hosted in an Azure MySQL or Azure SQL database.

Grading:

- Successful execution – 40%.
- Functionality that meets all benchmarks – 50%.
- Documentation – Describe your process and code using inline Markup in the Notebook – 10%.

Publicly-available sample databases:

- https://dataedo.com/kb/databases/mysql/sample-databases (Sample MySQL databases)
- https://docs.microsoft.com/en-us/sql/samples/sql-samples-where-are?view=sql-server-ver15 (Microsoft SQL samples)

Publicly-available datasets:

- https://www.kaggle.com/datasets
- https://data.world/
- https://www.data.gov/
- https://opendata.charlottesville.org/

Publicly-available APIs:

- https://docs.github.com/en/rest
- https://developer.twitter.com/en/docs/twitter-api
- HUGE LIST: https://github.com/public-apis/public-apis