Xetra ETL

Overview

The main goal of this project was to implement an ETL pipeline to generate the “biggest winner”, “most traded”, and “highest volume” for each day from the Xetra Trade data set.

Another important goal was to implement good design and coding practices while keeping in mind the short timeline and small scope.

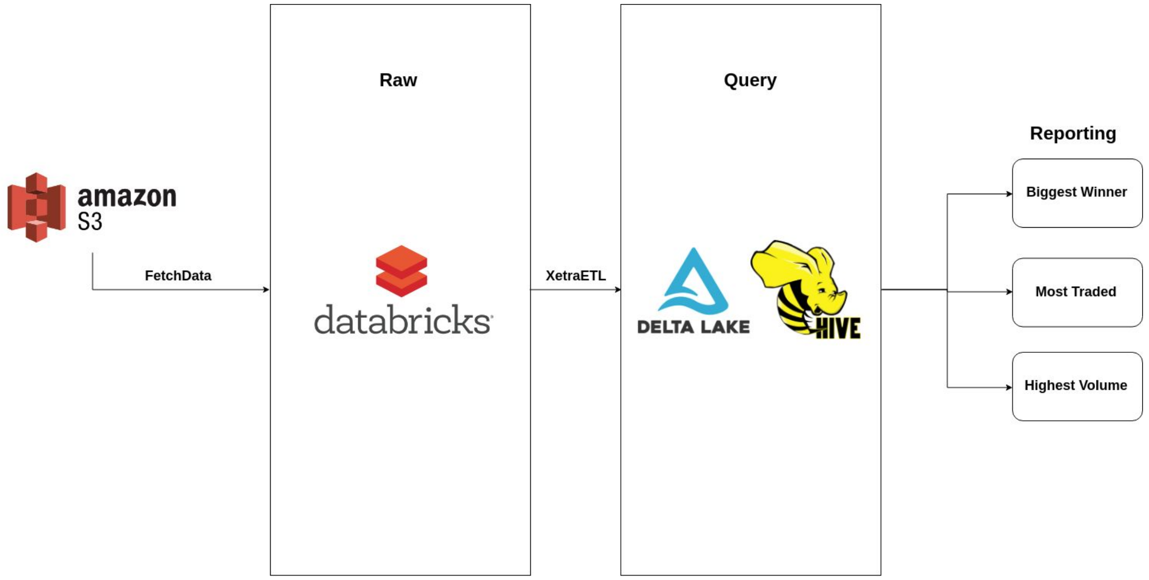
Architecture

- Scala
- Spark
- Databricks
- Delta Lake
- Hive

The code was written in Scala because it is highly performant and naturally compatible with Spark. The functional paradigm also makes it great for data manipulation.

Spark on Databricks was used because of how simple it is to share code and visualizations through notebooks. It also gave easy access to Delta Lake, which allowed for optimizations on storage and queries with partitioning.

Finally, Hive was used to create tables on top of the Databricks File System and Delta Lake.

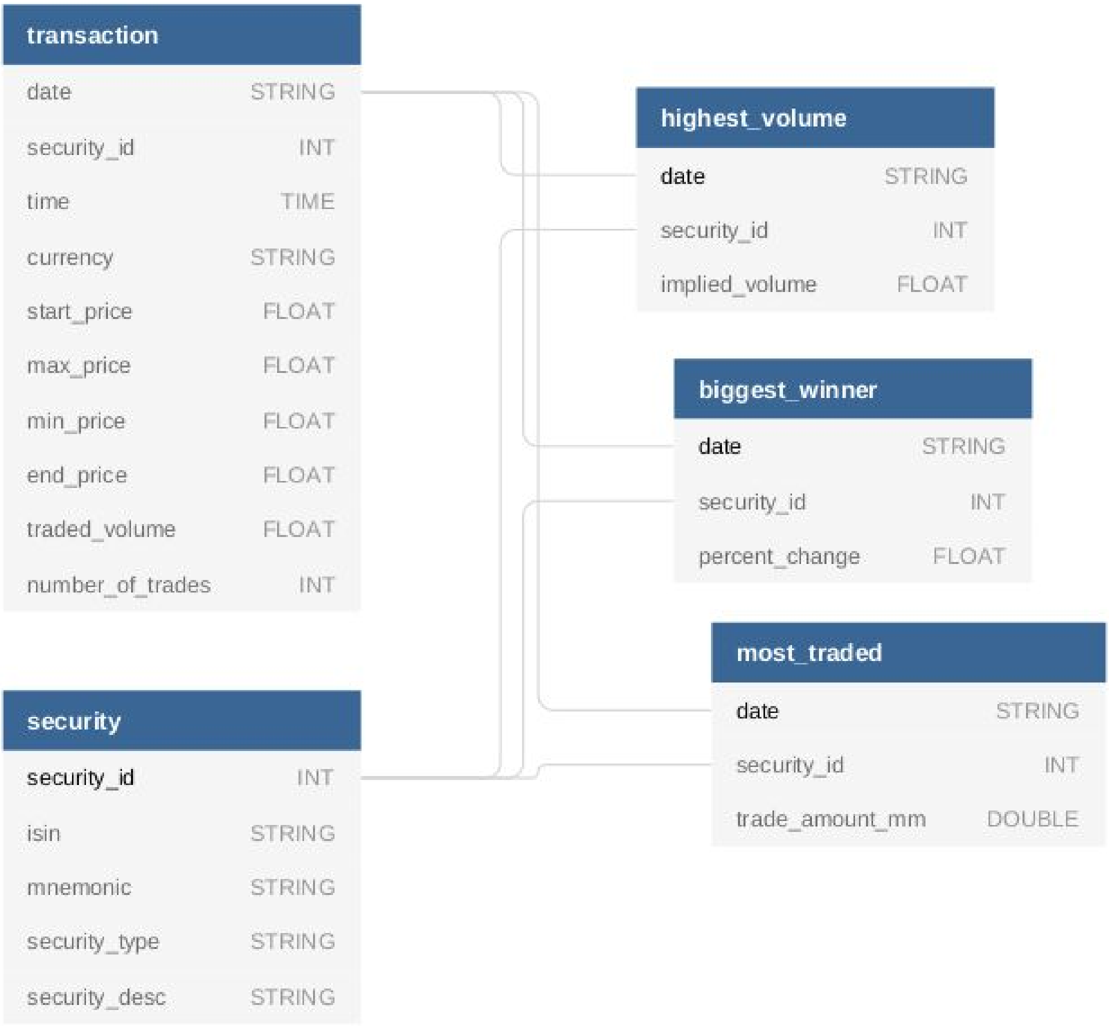


Data Schema

After extracting the raw data, it is stored into delta tables partitioned by date. This forms the raw data layer.

The data is then separated into transaction and security relation tables and defined a strict schema. This is the query layer, where data is well cataloged and analytics can be performed.

Finally, the query tables are used to create the biggest winner, most traded, and highest volume tables. These tables form the report layer, which is targeted towards business analytics and visualizations.



transaction	
date	STRING
security_id	INT
time	TIME
currency	STRING
start_price	FLOAT
max_price	FLOAT
min_price	FLOAT
end_price	FLOAT
traded_volume	FLOAT
number_of_trades	INT

security	
security_id	INT
isin	STRING
mnemonic	STRING
security_type	STRING
security_desc	STRING

highest_volume	
date	STRING
security_id	INT
implied_volume	FLOAT

biggest_winner	
date	STRING
security_id	INT
percent_change	FLOAT

most_traded	
date	STRING
security_id	INT
trade_amount_mm	DOUBLE