TECHNISCHE UNIVERSITÄT WIEN

TU Wien

Business Intelligence VU

188.429

Assignment 2: ETL, Cube Modeling Analytics (SQL/MDX/Atoti)

Group: 006

Student A: Kerim Halilović, 12434665
Student B: Nikola Lukić, 12127674

Date of Submission: November 29, 2025

1 ETL Summary

In comparison to the first assignment, this phase was quite straightforward to implement. Since the star schema was already defined for us, we did not have the creative burden of designing the model and could focus entirely on implementing the pipeline.

The logic of populating the dimensions was provided. It used the `generate_series` function to create the time dimension and simple joins to populate the City and Parameter tables. I then mplemented a `CASE` statement to map countries to their respective regions (e.g., mapping Austria to "Central Europe") directly in the SQL transformation.

The Fact table (`ft_param_city_month`) required more complex logic. We used CTEs to break the problem down:

- First, we mapped the text-based alert levels (Yellow, Red, etc.) to numeric ranks (1-4).
- We then aggregated readings to the daily grain to find the peak daily alert level.
- Finally, we rolled everything up to the monthly grain. This allowed us to calculate complex measures like `exceed_days_any` correctly.
- We generated the primary key using `ROW_NUMBER()` to ensure uniqueness.

2 Answers to Business Questions

We decided to select the first 10 questions for both SQL and MDX. By keeping the business requirements constant, we were able to directly compare the syntax and logic between the relational approach (SQL) and the multidimensional approach (MDX).

Selected Questions

- **SQL – Student A:** Q01, Q03, Q05, Q07, Q09
- **MDX – Student A:** Q02, Q04, Q06, Q08, Q10
- **SQL – Student B:** Q02, Q04, Q06, Q08, Q10
- **MDX – Student B:** Q01, Q03, Q05, Q07, Q09

Observations

In SQL, we found that calculating pivots (like showing months as columns) required repetitive `CASE WHEN` statements with aggregations. In contrast, MDX handled this naturally by simply selecting the Time hierarchy on the columns axis. However, filtering in MDX was sometimes tricky; we learned that we had to use sub-selects to avoid "hierarchy appearing in multiple axes" errors, whereas SQL 'WHERE' clauses are generally more forgiving.

3 Reflection and Lessons Learned

3.1 Student A: Kerim Halilović

Working on Assignment 2 provided a great opportunity to see how the theoretical design from Assignment 1 translates into actual code. The ETL process reinforced the importance of cleaning data before loading it into the warehouse, especially regarding the region mappings.

I particularly enjoyed working with Atoti. Moving from writing raw SQL queries to dragging and dropping measures in a dashboard made the data feel much more ”alive.” It was interesting to see how the semi-additive measures (like averages) were handled automatically by the cube, whereas in SQL we had to be very careful about how we grouped the data to avoid calculating averages of averages.

3.2 Student B: Nikola Lukić

For me, the most significant challenge in this assignment was learning MDX. Coming from a background where I am comfortable with SQL, shifting my mindset to multi-dimensional thinking was difficult. In SQL, I am used to thinking in rows and joins, but MDX required thinking in tuples, sets, and axes.

Understanding the difference between slicing a cube (using `WHERE`) and selecting axes took some trial and error. The error messages in MDX were often cryptic compared to PostgreSQL. However, once I understood the concept of `CrossJoin` and `TopCount`, I realized how powerful the language is for analytics. Writing a ”Top 10” query in MDX turned out to be much shorter than the equivalent SQL query, which was a nice takeaway.