Making of the SQL Airbnb Datamart IU project.

DLBDSPBDM01

1) Conception

The primary idea of the first phase was to design and create an Entity Relationship Model of the project using either MySQL, Postgres or DBeaver. After carefully analyzing the three applications, I decided to try DBeaver, where the initial blueprint for the ERD was created, but in the later development a decision to eventually switch to MySQL was made. The much user-friendlier interface as well as an opportunity to edit the tables using the reverse engineer wizard in MySQL that allowed to adjust the tables both via the modelling interface as well as via the tables interface, proved to be an irreplaceable tool during the development, as I spent most of the time in this stage, planning how the final result should look like and how to connect the foreign keys. Therefore, a conclusion on making three main tables was reached: User Account, Booking and Property. The rest of the tables were derived from the and facilitated the efficient communication between them. The chosen methodology involved the following steps:

- Creation of the ERM to identify entities, attributes and relationships. Adjusting the relationship between them and defining them using the crow-foot notation.
- Normalization of the database schema following the received feedback, as well as reducing the complexity of the design.
- Implementation of the database using MySQL Reverse Engineer functionality.
- Execution of sample queries to validate the database's capabilities.

2) Development

The second phase proved to be more complicated than the first one, since I also encountered several issues that appeared due to poor planning of the first stage:

- Some of the tables used a simple INT data type, while others had INT UNSIGNED data type. This became an issue when it was necessary to link foreign keys.
- Populating the tables with data was a challenging task, because I didn't know that there is a specific sequence of doing it, so creating the file with the data to be inserted was not as straightforward as was initially anticipated. The result is a specific sequence of tables to be filled first. Each dummy data had to be first synthetized using Mockaroo website and then handled manually to make sure that the input was accepted correctly and sometimes text queries had to be appended fully manually.

- Figuring out the relationships between the users (host, guest, admin) was simple at first, but became a lot harder when the idea to assign an admin to a ticket was considered. Ultimately, this feature was scrapped, because it proved to alter the design of the user_account table and complicate the normalization.
- Sample queries were executed and documented in this phase to demonstrate the full functionality of the technical solution. Such queries that either retrieve all the available properties for a specific attribute or the whole tables are documented in the file.
- The implemented database meets the functional requirements and fulfills the roles that were defined in the conception phase.

3) Finalization

During the finalization phase, a retrospective analysis on the overall development procedure as well as what could've been done differently was conducted with the following conclusions:

- Normalization was done to a degree that was possible. Further normalization proved to be complicated and ended up messing with the overall design and simplicity of the final solution. As an example, special tables with composite primary keys were defined for property amenities, property services and property policies to solve some of the issues.
- One of the most important features that was missed during the design was definition of what should happen to the columns on deletion. Since the cascading rules were not defined with the creation of the tables, it became really complicated to experiment with the dummy data and manipulations with it. Therefore, the tables don't support deleting data and were created with the goal of storing information long term for the sake of traceability.
- Ensuring data integrity via foreign keys also proved to be a challenge, especially when older tables had an integer that was signed, and the newer tables had an unsigned integer. This was adjusted for all the tables.
- The process of designing and implementing the database improved my understanding of ER modelling and SQL. The most important step of the design process is the initial concept and rigorous normalization, which was achieved with the help and feedback of the professor. Testing filling out the data with dummy data, as well as writing queries showed possible design flaws that were adjusted for by changing the data in the tables and foreign keys.