Hussein Younes
Mikhail Gudkov
Valentin Sergeev
Daniel Atonge

PROJECT PHASE 3

Design decisions

- Table Creation:
  All Entities which are present in the ER diagram are mapped into Tables in our database. One-to-one relations aren't present in our ER diagram. For one-to-many relations, the primary key in the many-side is used as an attribute in the one's-side as a foreign key, a similar action is taken for relations containing attributes. The attribute is moved to the many's side and added to the many's Table. For many-to-many relations, a new table is created containing the primary keys from both tables (foreign keys in the current relationship table) and if existent, the attributes contained in the relationship. Enumerations are used to represent attributes that assume predefined values like the days of the week, staff roles and more.
  Tables representing weak entities are assigned unique identifiers which is the composition of their partial key and the primary key from the corresponding strong entity they are in relation with.

- Normalization:
  Implementing table creation as described above, we had all our created tables in 1's NF (normal form here and after will be denoted as NF) i.e each table had a primary key, the values in each column of the tables are atomic and there are no repeating groups. Furthermore, we satisfied the requirements for our tables to be in 2nd NF (i.e removing partial dependencies in all tables). Finally, our tables were optimized to satisfy the 3rd NF requirement by removing transitive dependencies. This sometimes leads to removing some irrelevant attributes hence optimizing our database design.

  An interesting table to consider is the PATIENT'S TABLE:
  We realized data duplication and dependencies in the patient's table and using normalization techniques, we were able to overcome these shortcomings. (Dropping attributes which weren't needed and creating new tables)

- Changes in ER Diagram
  After Project phase 2, the feedback provided implied major changes in our ER diagram. These included the removal of some attributes (e.g login)

- How to Create Database without script using Postgres:
  1)Log in to Postgres
  2)Create the database hospital
  3)Execute DataBase_PostgreSQL.sql to create tables and views for the database
  4)Execute Dump_PostgreSQL.sql to publish data in the database

- How to Create Database without script using MySQL:
  1)Log in to MySQL
  2)Create the database hospital
  3)Execute DataBase_MySQL.sql to create tables and views for the database
  4)Execute Dump_MySQL.sql to publish data in the database

- **NOTE:** Data contained in the dump file (which includes the INSERT statements) is not random, it is already prepared data to serve the purpose of the queries.

- How to Run the python Script:
  To use Postgres script you need:
  1) Create the database hospital
  2) Execute DataBase_PostgreSQL.sql to create tables and views
  3) Download 'psycopg2' and 'names' python libraries
  4) Open the Postgres script code and change the database name, user and password
  5) Run the Data_Generator_PostgreSQL.py script

  To use MySQL script you need:
  1) Log in to MySql server and create the database hospital
  2) Execute DataBase_MySQL.sql
  3) Download 'pymysql' and 'names' external libraries
  4) Open the mysql script code and change the database name, user and password
  5) Run Data_Generator_MySQL.py script

- How to Run the GUI Application:
  Requirements (Running JavaFx application)
  After creating the database and filling it with the required data (i.e using the dump file or running the python script), follow the steps below in order to set up the GUI and execute required queries.

1) Check https://openjfx.io/openjfx-docs/#IDE-Intellij
2) In project settings, add library “org.postgresql:postgresql:42.2.52” using maven
3) Run application
4) You will be presented with a login interface to enter Database Name, user and password.
5) Click Connect and move to the Postgresql tab to execute the queries