Database Design Final Report

Ryan Wu, Emily Dolan, Urvi Bhojani

Group: WuRDolanEBhojaniU

CS3200: Database Design

June 21st, 2023

README/TECHNICAL SPECIFICATIONS:

Steps to run:

- Install and import pymysql[1]
- Add password to connection (replace '----' in password = —- with password)
- Follow program prompts

For our Database Design Final Project, we wanted to create a hospital office database. This would allow a user (ideally a receptionist at a medical office) to access physician records, patient records, appointment details, equipment details, insurance details, prescription details, as well as medicinal details. As a result, our objective with this project was to allow a user to be able to use or interface seamlessly to perform specific tasks, such as updating patient insurance details, deleting appointments, or adding new prescriptions.

To create our database and programming objects (functions, procedures, triggers, and events), we utilized MySQL[2]. To create our user-interactive interface, we utilized python. Specifically, we worked in PyCharm[3] and Jupyter Notebook and utilized the pymysql library in order to create connections and cursors to connect to our created database (doc_office) and call functions from MySQL.

More specifically, our database doc_office includes 11 tables: appoint_equipment, appointment, billing, department, equipment, insurance, medication, patient, physician, prescription, and room. From here we created approximately 30 programming objects in MySQL (which also covered the CRUD operations). These objects were vital in our user interface.

Additionally, our user interface includes approximately 30 functions which all include prompts for the user to navigate through the hospital database in order to create, read, update, or delete information. Following these functions, there are a series of if/else statements which allow the user to maneuver through the tables and perform operations.

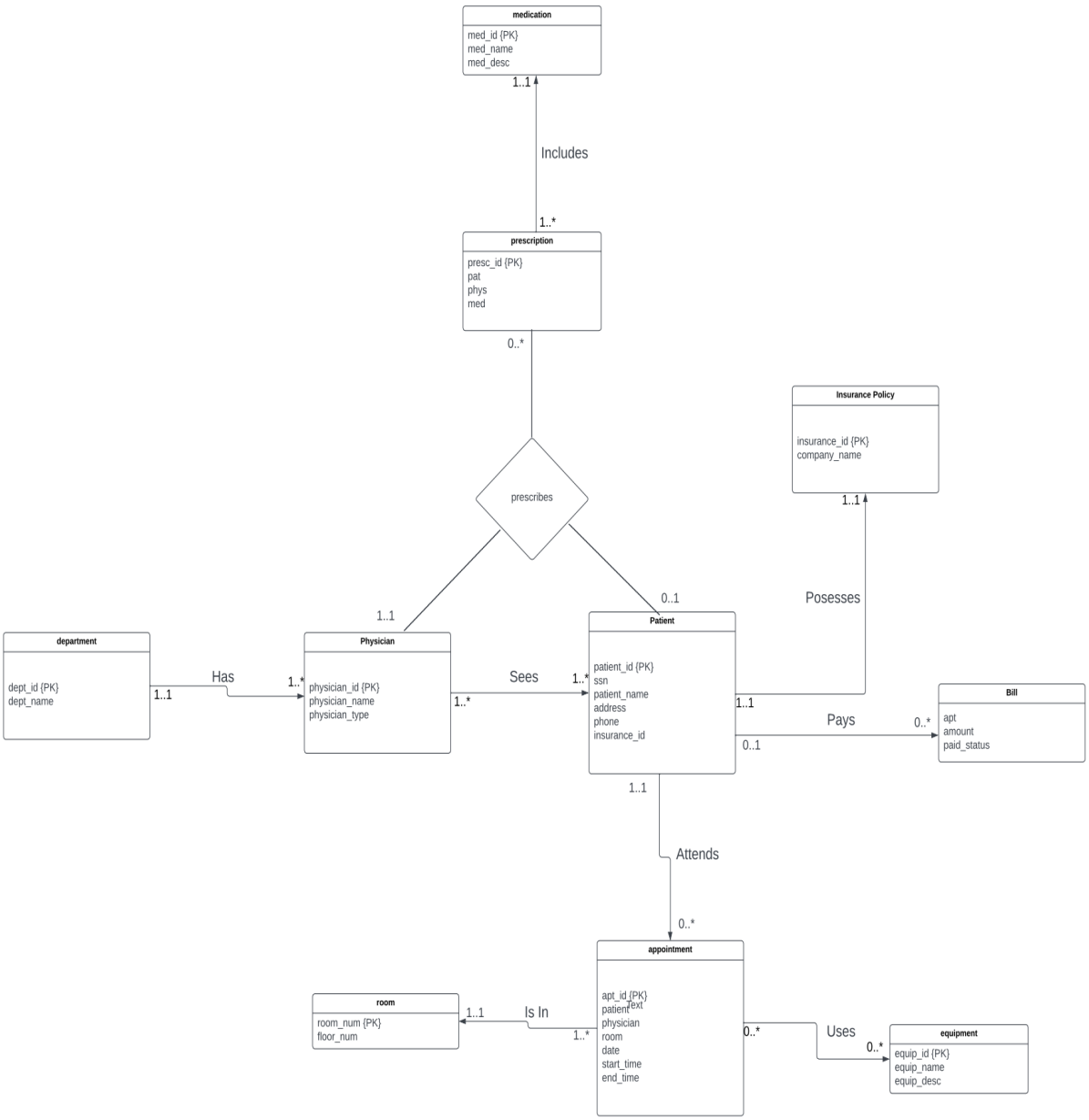
---

[1] http://https//pypi.org/project/pymysql/

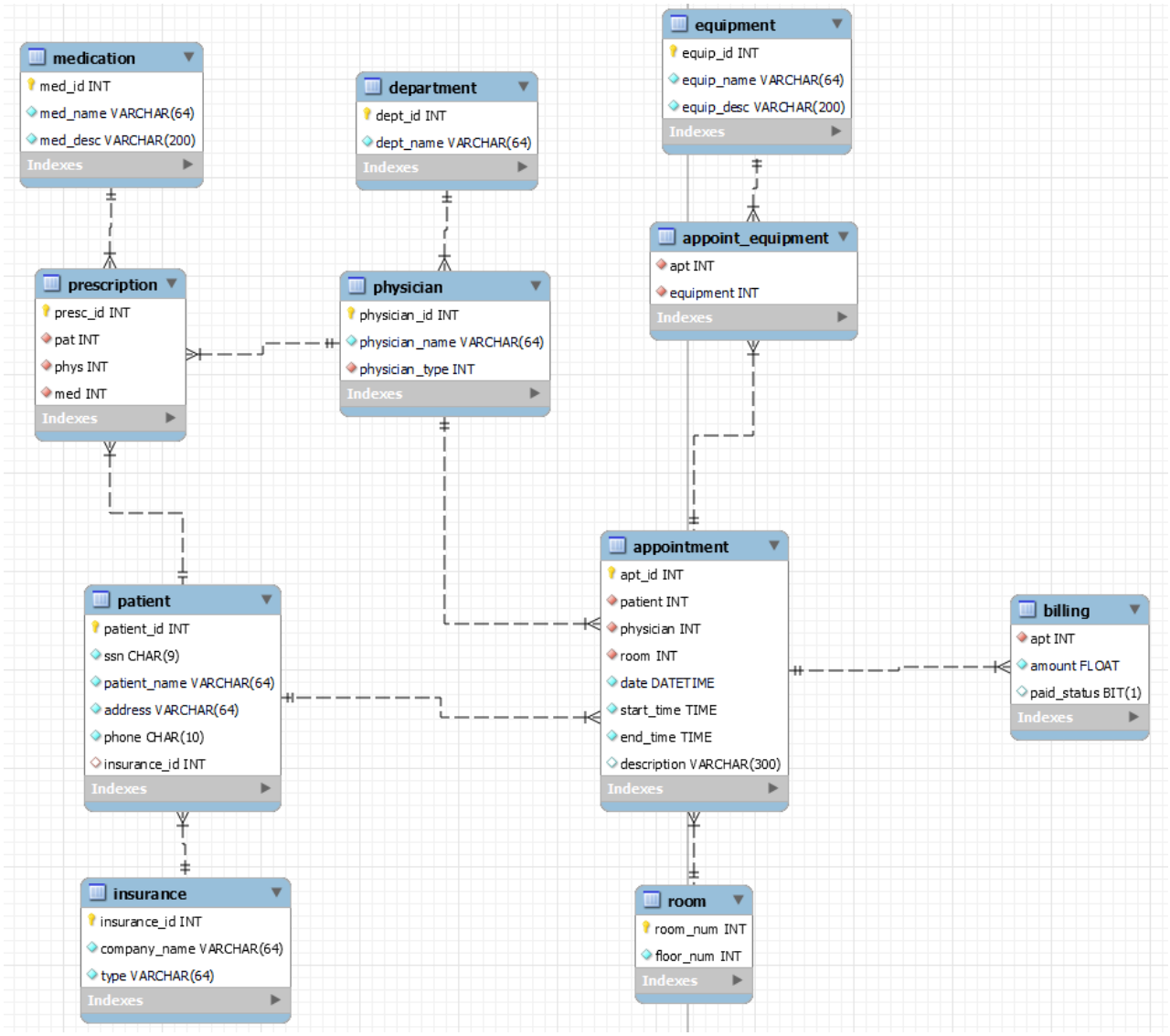
[2] https://dev.mysql.com/downloads/workbench/

[3] https://www.jetbrains.com/pycharm/download/#section=mac

CONCEPTUAL DESIGN - UML DIAGRAM:



LOGICAL DESIGN:



USER FLOW:

Appointment, physician, patient, medication

1. User adds password to connection and connects to database
2. User inputs the number corresponding to:
    a. 1. Physician Records
        i. If chosen, the user selects the number corresponding to
            1. ‘Enter new Physician Details’
                a. User enters information for the Physician being added to the database
            2. ‘Delete old Physician Details’
                a. User enters the information of a physician to be deleted from the database
            3. ‘Read Physician Details’
                a. User is presented with information for each physician in the database
            4. ‘Fetch Physician Details’
                a. User enters a physician ID and is presented with the details for that physician
    b. 2. Patient Records
        i. If chosen, the user inputs the number corresponding with
            1. ‘Enter new Patient Details’
                a. User enters information for the Patient being added to the database
            2. ‘Delete old Patient Details’
                a. User enters information for the Patient to delete from the database
            3. ‘Read Patient Details’
                a. User is presented with all patients in the database
            4. ‘Update Patient Details’
                a. User enters information for the Patient to be updated
            5. ‘Fetch Patient Details’
                a. User enters a patient id and is presented with the details for that patient
    c. 3. Appointment Details
        i. If chosen, the user selects the number that corresponds with
            1. ‘Enter new Appointment Details’
                a. User enters information for the new appointment
            2. ‘Delete old Appointment Details’
                a. User enters the information for the old Appointment to be deleted

3. ‘Update Appointment Details’
    a. User chooses if medical equipment is involved and then enters the new information for the existing appointment to be updated
4. ‘Read appointment details’
    a. User is presented with all appointments in the database
5. ‘Fetch appointment details’
    a. User enters a appointment ID and is presented with the details for that appointment

d. 4. Equipment Details
    i. If chosen, the user selects the number that corresponds with
        1. ‘Enter new Equipment Details’
            a. User enters information for the new Equipment to be added to the database
        2. ‘Delete old Equipment Details’
            a. User enters information for the existing Equipment to be deleted
        3. ‘Read Equipment Details’
            a. User is presented with every equipment in the database
e. 5. Insurance Details
    i. If chosen, the user selects the number that corresponds with
        1. ‘Enter new Insurance Details’
            a. User enters the information for the new Insurance plan to be added to the database
        2. ‘Read through insurance details’
            a. User is presented with every insurance plan in the database
f. 6. Prescription Details
    i. If chosen, the user selects the number that corresponds with
        1. ‘Enter new Prescription Details’
            a. User enters the information for the new Prescription to be added to the database
        2. ‘Update Prescription Details’
            a. User enters the information to be updated for a specific prescription
        3. ‘Read Prescription Details’
            a. User is presented with every prescription in the database
g. 7. Medication Details
    i. If chosen, the user selects the number that corresponds with
        1. ‘Enter new Medication Details’

    a. User enters information for the new medication to be added to the database
2. ‘Read Prescription Details’
    a. User is presented with every prescription in the database
3. ‘Fetch Medication Details’
    a. User enters a medication ID and is presented with the details for that medication

After performing the selected operations, the application closes the connection.

LESSONS LEARNED:

1. Starting with the end in mind
    a. One of the prominent lessons learned from this project is the importance of understanding what functions you want your database to provide your user before beginning construction. In a ‘real world’ situation this would involve detailed discussions with a client, but for us it meant brainstorming our final product before we began around the question: what information would doctors and admin want from a doctors office database? Once we lead from this question, deciding on tables and fields became much easier.
2. Creating a Quality Conceptual Design
    a. Though the conceptual design may seem like an ‘easier’ portion of the project to move through quickly, this assignment taught us the value of investing time on the front end into designing a quality conceptual design. Spending extra time here gave us confidence in our conceptual design as well as an increased understanding of our data overall, which made it easier to integrate into SQL and develop our front end application.
3. Normalization and Data Integrity
    a. When designing our database, one of our biggest challenges and concerns was normalization and data integrity. We learned how to eliminate transitive functional dependencies in order to break our tables down into the third normal form. For example, we added a procedure_equipment table to account for the possibility of multiple pieces of equipment being used in one procedure.
4. Collaboration and Communication
    a. Collaboration and communication were both key to integrating all of the pieces of our final project. We met frequently via zoom and communicated via a group chat to discuss our progress on each component of the project. This consistent communication both kept us on track with time, as well as allowed us to work on portions of the project individually that integrated successful

FUTURE WORK:

Some planned uses and potential areas for added functionality of our database and project include:

1. Automated Appointment Reminders: implementing automated appointment reminder via SMS or send reminders to patients and physicians with corresponding appointment details, in order to ensure that both parties are well-prepared
2. Integration with Electronic Health Records: this database can be integrated with an EHR system to provide a comprehensive patient record that includes medical history, patient allergies, as well as previous treatments and medications. This integration would enable healthcare providers a complete overview of the patient's health information, which would result in better informed decision making and more coordinated care.
3. Patient Portal: Currently employees have access to a similar feature, but creating a secure patient portal integrated in the system on the patient end would effectively allow them to access their medical records, schedule appointments, view test results, communicate with healthcare professionals, and manage their healthcare preferences on their own. This type of self-service functionality can be super useful for the patient because it can promote patient engagement, convenience, and active participation in their own healthcare journey.
4. Telemedicine Integration: In addition to a patient portal, integrating the database with a telemedicine platform would be a powerful thing as it would allow patients to schedule and conduct virtual appointments for those who may not be physically able to visit the hospital.
5. Advanced Analytics and Reporting: Based on patient and employee engagement, developing a corresponding comprehensive analytics and reporting creature can help healthcare professionals and administrators to gain insights into patient outcomes, resource utilization, as well as performance metrics. This approach can revolutionize the decision-making, improve resource allocation, and overall help the facilitation of quality improvement initiatives within a hospital.