CSE 3241/5241: Database Systems Project

Computer Science & Engineering - The Ohio State University

Equipment Renting *Database System* for Local Communities

General requirements Overview

A non profit organization implemented a service to help communities in a region, renting at a low cost, equipment for community and family projects. Any person in the community can become a member of the service and rent out any item available in stock and be charged the fees. The elements rented are delivered and picked up by a drone. The institution has a drone fleet composed of different drone capabilities to perform the delivery service to members.

The institution needs to build an application software to manage their drone and equipment collection and needs a simple database management system to support their inventory and renting operations. This project will require to design the database as well to develop a Java application to integrate with the database.

You and your team will work on this project independently during the semester to implement a running database and create a small application as a proof of concept.

This document contains the specification of the basic set of user requirements for the system. In addition, to receive full credit for your project you will be required to come up with extensions that expand on the requirements here and provide new functionality beyond the scope of the basic requirements.

Basic description of the renting operation

The institution has on each local community a local warehouse with an inventory of drones (different sizes and capabilities) and stock of equipment for diverse types of activities (e.g. house renovations/fixes, plumbing, painting, watering systems. gardening, electrical systems, siding and windows, roof, computer and internet devices, sensors and controllers, etc). When a registered user rents an item(s) or needs to return it, a drone is dispatched to deliver/pick up the item and then return to the warehouse. A fee is charged to the user account according to the item renting fee and the number of days. Drones can transport multiple items, but without exceeding its capacity and time autonomy.

The following list of features include internal and external use cases. A non-exhaustive list of requirements that need to be supported are the following:

- Store basic information of local warehouse as: city, address, phone number, manager name, storage capacity, drone capacity.
- Registering a person as a community member and assigning a user id, deactivating a member when moving out.
- Each member needs to have stored as minimum: first name, last name, address, phone, email, start date, warehouse distance (Note: the value for warehouse distance could be calculated using a free webpage application)

- Entering new drones as the inventory is expanded, storing at least into the database: description/name, model number, serial number, fleet id, manufacturer, year, weight capacity, volume capacity, distance autonomy, max speed, warranty expiration date. When drone is lost/decommissioned, the drone status is change to inactive.
- Entering new equipment into the stock, storing in the database at least the following: equipment type, description, model number, year, serial number, inventory id, arrival date, warranty expiration, manufacturer, weight, size.
- Entering and keeping track of new elements ordered (new drones or equipment needed), storing at least: description, element type, number of items ordered, value, and an estimated date of arrival. Order history must be preserved.
- Adding/activating new ordered items that arrive to the current inventory (drones or equipment); so the equipment can be rented or the drone can be use to deliver.
- Store equipment rating and reviews by members. Members can add reviews about equipment rented as well as the service, providing comments and ratings
- Keeping track of each inventory item and its location: drones, equipment.
- Editing existing entries in the database to correct errors
- Deleting entries in the database in the case of incorrect data
- Displaying entries from the database based on user entered criteria
- Searching the database for members, drones, equipment, rental history, manufacturers, ratings, expiration dates
- Listing all rentals for a member
- Check out instances of items rented by a member, keeping track of the due date.
- Check in instances of items returned by a member, fees charged

More use cases will be provided and become clear as the project progresses. You will be required to come up with extensions beyond the provided criteria.

It is highly recommended that you read through this document and all project documents in their entirety before starting the project to understand the requirements as early as possible.

Project reports (Checkpoints and Final Report)

The project requires to design and implement a database system, each team must deliver a final project report that contains the documentation of the design and the files with the database and software implementation.

The design and implementation of the project will be done in four intermediate (4) stages and a final project report. For each stage a checkpoint report will be submitted according to guidelines given for each one.

The final report contains different sections, most of those sections correspond to the material elaborated during the four checkpoints throughout the semester (see calendar for each checkpoint due date). There is a document with the description and instructions for each one of the checkpoints (Go to Canvas to download the blank checkpoint documents).

You will start working on the project immediately and work on it incrementally during the semester. Since this a team project, communication is essential to coordinate efforts and reach a balance among all the members and get to enjoy the experience. Here is an overview of the checkpoints:

- Checkpoint 1 asks you to analyze the use cases described above and provide an ER/EER model of the database as well as the mapping into a Relational Schema. You also need to provide a high level design of the application user interface to provide to the users. Define a basic functionality for your application, e.g. add, modify, or search any of the important data items in the data base. Use a text-based user interface to navigate the menus and submenus (graphical UX is optional)
- Checkpoint 2 asks you to revise your diagram and model, based on feedback. You will come up with additional useful queries and you will then identify the functional dependencies and normalize your relational model. Finally, you will start writing and testing an initial version of a Java application, that on a later Checkpoint you will integrate with the database (exact requirements are in that document).
- Checkpoint 3 asks you to write the SQL that creates your database, prepare and populate the database. You will then write SQL to perform the queries from Checkpoint 2, the additional queries you came up with, and some more advanced queries.
- In checkpoint 4 you need to connect your program to use the database, to perform queries and transactions. You may use the initial program version developed on Checkpoint 2.

The four checkpoint reports will be submitted during the semester so that you can get feedback and work on any necessary corrections. You will receive feedback (but no grading) for submitting four checkpoint documents.

A score will be assigned to each Checkpoint to indicate your progress, that will be used to evaluate how was the performance during the project (see rubric). NO grades will be entered/counted on Carmen, only the final report.

Final Project Report

The final report must be a **professionally presented**, **well-organized**, **typed document**, and a **complete SQL database**, submitted electronically to the Carmen Dropbox in a single ZIP file. This ZIP file needs to be neatly and professionally organized. With all filenames appropriately chosen, and all files suitably organized into subdirectories. Include a Table of Contents file named README.txt that explains the layout of your files, including where to find each of the following files in your file structure. USE ONLY pdf, doc, pptx, sqlite file formats that can be open easily on Carmen or SQLite

Part I – The Final Report

Your final report document will include the documentation that you generated during the different stages of your design (Checkpoints): a relational database schema, entity relationship diagram, SQL queries, and reports, Java application as indicated below.

Note: It's recommended that you create a final project template document at the beginning of the semester and then start feeding into that document with the specific designs (after corrections are made) that you deliver for each checkpoint. This will save you time by the end of the semester and allow you to identify/focus some additional things needed in the final report.

Section 1 - Database Description

A database description document that contains the following information about your database (compiled from your completed and revised checkpoints);

a. A professionally presented, well-formatted ER-model that reflects the updates you have made during the semester. **Do not submit a hand-drawn diagram**.

b. A professionally presented, well-formatted relational schema for the database. This schema must be annotated with the primary key for each table, all foreign keys on all tables, and all functional dependencies on all tables. Make sure that connections between FKs and PKs are clear. **Do not submit a hand-drawn diagram**.

c. For each table, provided a summary and justification of the FDs and level of normalization achieved for that table. If the table is not in BCNF, explain why.

d. A description of at least three indexes that you choose to implement on your database, along with rationale for selecting each one. The indexes provided must be non-primary key attributes as databases already index on the primary key.

- You need to identify which indexes to define for your tables. To identify them, analyze the most frequent queries to see which ones are needed/useful, check tables with high volume of data (now or that will grow in the future) that could be used frequently. This analysis is the rationale behind it.

e. For each view that you have implemented, provide the following:

i. A brief description in English of what this view produces, and why it would be useful.
ii. Relational algebra expression to produce this view.
iii. SQL statements to produce the view.
iv. Sample output from the view, with 5-10 lines of data records shown.

f. A professionally presented description of three sample transactions useful for your database. This should include the sample SQL code for each transaction as well as an English language description of what “unit of work” the transaction represents. Remember – a transaction is a sequence of SQL statements executed as a unit. Write the SQL transaction following the example on the slides (no need to write/embed the transaction on Java). Look for use cases where some data items could be accessed concurrently and we want to ensure consistency on the database. A transaction with multiple R/W operations are expected.

Section 2 - User Manual

A user manual describing the usage of your database, for use by developers who are going to be writing code to use your database. Your manual should include:

a. For each table, explain what real world entity it represents. Provide a description of each entity and each attribute, including its data type and any constraints you have built-in.

b. The sample SQL queries that you worked in the different Checkpoints. These are all the queries including the ones made up by each group: Queries (6), ExtraQueries (3), AdvanceQueries (8) . These queries should be organized and presented neatly and professionally. Each query should include:

- i. An English language description of what the query should be returning
- ii. The correct relational algebra syntax of the query
- iii. The equivalent SQL query
- iv. A screenshot (SQLiteStudio) of the execution/results of the query

c. INSERT statement syntax for adding new data items to your system: members, drones, equipment, rentals, etc. If there are dependencies in your system that require one/multiple records to be added to tables in a specific order to add one of these items, make sure you clearly indicate what those restrictions are. You need to provide an example of INSERT statements for each entity in your database.

d. DELETE statement syntax for removing data items from your system: members, drones, equipment, rentals, etc.. Again, indicate any dependencies that exist on the order that the steps in your DELETE must take. You need to provide an example of DELETE statements for each entity in your database.

Part II – The SQL Database

Section 1: Database File(s)

1. A binary version of your database, suitable for opening using SQLiteStudio

Section 2: SQL scripts (to recreate database)

1. SQL CREATE. A text file containing all of the sql scripts needed to create your database schema on an empty database. This file should be properly commented and should execute properly if pasted into a SQLite command prompt (or loaded from the command line tool). These scripts should include all indexes and views created on your database.

2. DATA FILES. A set of text files containing the data to be loaded into your database. These files, when used with the table creation scripts above, should be able to recreate your database from scratch if your binary file is corrupted or lost. Make sure you provide instructions on how to use these scripts and files in a separate text file.

Note: you could generate an export (sql format), including the both the database creation and data insertion sql script.

Section 3: Program (software code)

1. Java Application. The code of the Java application developed on Checkpoint 4 must be included, with the required corrections. To get points your program must be running, integrated with the database and executing the queries required by the user options provided in the functionality description.

- Besides the Java application project files, include a screenshot (Eclipse) running each program option implemented (you could copy/paste them into word/powerpoint file to ease the process).

We will use Eclipse to import and run the Java application. You need to export your project and provide specific instructions on how to import the project and execute it. To be sure that the project file could be imported without problem, test it before submitting it , following your own instructions.

END OF FINAL REPORT

Test data requirements:
You will need to create your own data. You should put enough test data into your database in order to run the sample queries and test the scenarios. Study on the Project Module section on Carmen guidelines for data preparation.

It is important that you have enough test data to:
(1) generate reasonable results for each of the scenarios described on the different queries and be able to verify they are correctly implemented; and
(2) generate good reports. For example, some of the reports require computing total; in that case you should return multiple data rows.

Test scenarios requirements:

The database should be able to support the usual operations of the renting inventory. That is, that users may browse for equipment, type, checkout and review their accounts. An administrator should be able to review the renting and see what items need to be ordered from the provider.

Complete list of documents (Go to Modules on Canvas to download them)**:**

- Project Complete description (this document)
- Project Rubric
- Checkpoints 1 through 4
- Peer-evaluation form