CSE 3241/5241: Database Systems

Computer Science & Engineering - The Ohio State University

Media Manager Database System for a Local Library

Overview

A local library is building an application to manage their music and video collection and needs a simple database management system to support their inventory and circulation operations. This 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 all semester. You will receive feedback (but no grading) for submitting five checkpoint documents.

This document contains the specification with the basic set of user requirements as well as the description of some sample data provided to be used to populate some tables in your database.

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.

Lists features will include internal and external use cases. A non-exhaustive list of features:

- Entering new artists, albums, tracks, genres, length, year into the database as the music collection is expanded
- Entering new movies, start role actors, director, genres, length, year, content rating into the database as the video collection is expanded
- Entering new author, audiobooks, chapters, genres, length, year into the database as the audiobook collection is expanded
- Entering and keeping track of new media items ordered (but still not received): albums, videos and audiobooks, with number of copies purchase, price and an estimated date of arrival.
- Transferring new media items that arrive to the current media inventory (removing them from the ordered items inventory)
- There could be several physical copies of albums/videos/audiobooks as well as electronic copies
- Keeping track of inventory: physical copies of albums, videos and audiobooks with their location, and licenses for digital copies of albums/videos.
- Registering a person for a library card, deactivating old or lost library cards. A patron can have only one card. First name, last name, address of the patrons are needed as well as the email address.
- Editing existing entries in the database to correct errors
- Deleting entries in the database in the case of lost items
- Displaying entries in the database based on user entered criteria

- Searching the database for artists, actors, authors, albums, audiobooks, movies and music tracks
- Listing all checkouts (physical, digital, or both) for a patron
- Check out instances of physical media to a patron using a library card, keeping track of the due date.
- Check out instances of digital media to a patron using a library card, keeping track of the due date
- Registering into the system the items returned by patrons, and sending an email to confirm and notify any fees due if the item was returned late

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.

Partial project reports: Project Checkpoints

There will be five checkpoints (see calendar for checkpoint due dates). The Checkpoint documents (1-5) will provide a complete list of required use cases and directions to add extended functionality (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 finally get to enjoy the experience. Here is an overview of the checkpoints:

- Checkpoint 1 will consist of a conceptual analysis of the project. You will be asked to reason about the data being modeled and propose ways of representing the data in the database. You will then be asked if common and useful tasks can be completed using your representation, allowing you to revise your representation. Finally, you will compose an ER diagram for your database.
- Checkpoint 2 asks you to revise your ER diagram to reflect the feedback you will receive on Checkpoint 1. You will then map your revised diagram to a relational schema and write relational algebra to perform a list of queries, again, allowing you to revise your schema if needed. Finally, you will think of three additional queries you think would be useful or interesting.
- Checkpoint 3 asks you to revise your diagram and model based on feedback. You will then provide the functional dependencies of your model. You will then find the highest normal form for each relation schema in your model. If the relation is not in 3rd normal form, you will rewrite it so that it is. For each relation that is in the 3rd normal form but not Boyce-Codd normal form, you will either rewrite it or justify why it should be an exception. You will construct two interesting views that can be built with your relations (exact requirements are in the document). 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 the document).

- Checkpoint 4 and 5 asks you to revise your diagram and model again based on feedback. Then you will write the SQL that creates your database with SQLite and populate the database with the data provided with the project along with some additional data you must create. You will then write SQL to perform the queries from Checkpoint 2, the additional queries you came up with, and some more advanced queries.
- Checkpoint 5 will ask you to write a program to use the database, to perform some queries and transactions. You will use the initial version developed on Checkpoint 4.

The five checkpoint reports will be submitted during the semester so that your instructor can give you feedback and any necessary corrections. A score will be assigned to each Checkpoint that will be used to evaluate how your progress/performance was during the project (see rubric). NO grades will be entered/counted on Carmen, only the final report.

(Go to Modules on Canvas to download the document with the blank checkpoint descriptions).

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 Canvas or SQLite

Part I – The Final Report

Your final 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, as indicated below.

Note: It's recommended that you create a final project template document and then start feeding that document with the designs (after corrections are made) that you deliver for each checkpoint. This could save you time by the end of the semester and allow you to identify 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.

c. For each table, give a brief description of the level of normalization achieved for that table. If the table is not in BCNF, explain why.

d. A description of each of the indexes that you have chosen to implement on your database, along with rationale for each.

- 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 taken as a unit – this can be reads and writes together or just a sequence of writes. One example of a sample transaction you might want to consider is the user making changes to an order – what might need to be considered a transaction in that case?

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 provided in the different Checkpoints (8 queries from Chkp4 and 8 queries from Chkp 5, See sections CP4: 4.3, 4.4 and CP5 5.4). 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

c. INSERT statement syntax for adding new tracks, albums, movies/videos, audiobooks, artists and patrons to your system. If there are dependencies in your system that require 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.

d. DELETE statement syntax for removing tracks, albums, movies/videos, audiobooks, artists and patrons from your system. Again, indicate any dependencies that exist on the order that the steps in your DELETE must take. In addition, provide an example set 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 either the sqlite command, SQLite studio.

Section 2: SQL scripts (to recreate database)

1. SQL CREATE. A text file containing all of the 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.

3. SQL QUERIES. A text file containing all of the SQL queries used in your final report from Part I. All of these queries must be in a form where they can be run over your database through a simple cut and paste into the admin tool or the sqlite3 command line tool. In addition, make sure that these queries are completely commented so that it is clear where the query comes from in your final report write-up.

4. SQL INSERT/ DELETE. A text file containing all of the sample INSERT and DELETE statements provided in your user manual, suitable for pasting into a command prompt and testing the result on your database.

Section 3: Program (software code)

1. Java Application. The code of the Java application developed 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.

Part III

Appendix - Graded Checkpoint Documents

An appendix to the final report that MUST contain all of your original, graded checkpoint documents organized in sequence and in a neat manner with the feedback provided at the end of each checkpoint.

For each one of the checkpoint that required a revision you MUST include a revision for that checkpoint. Usually the revisions are performed in the next checkpoint. Please indicate/point to where the "fixed" version of the checkpoint resides.

END OF FINAL REPORT

Test Data (real music data, use it according to your table structures to populate them)

Real test data files for music can be found in the provided project files. Use this data to prepare your music data according to your schema (NOTE: use tools like Excel to prepare the data needed according to you table structures, save it in csv format to import it).

Test data sample. The first line of the CSV file consists of column names. Each row after that is a record.

```
artist_name, album_title, track, length_seconds, size_bytes, genre, release_year
AC/DC,For Those About To Rock We Salute You,Breaking The Rules,263,8596840,Rock,1981
AC/DC,For Those About To Rock We Salute You,C.O.D.,199,6566314,Rock,1981
AC/DC,For Those About To Rock We Salute You,Evil Walks,263,8611245,Rock,1981
AC/DC,For Those About To Rock We Salute You,For Those About To Rock (We Salute You),343,11170334,Rock,1981
AC/DC,For Those About To Rock We Salute You,Inject The Venom,210,6852860,Rock,1981
AC/DC,For Those About To Rock We Salute You,Let's Get It Up,233,7636561,Rock,1981
AC/DC,For Those About To Rock We Salute You,Night Of The Long Knives,205,6706347,Rock,1981
AC/DC,For Those About To Rock We Salute You,Put The Finger On You,205,6713451,Rock,1981
AC/DC,For Those About To Rock We Salute You,Snowballed,203,6599424,Rock,1981
AC/DC,For Those About To Rock We Salute You,Spellbound,270,8817038,Rock,1981
AC/DC,Let There Be Rock,Bad Boy Boogie,267,8776140,Rock,1977
AC/DC,Let There Be Rock,Dog Eat Dog,215,7032162,Rock,1977
AC/DC,Let There Be Rock,Go Down,331,10847611,Rock,1977
AC/DC,Let There Be Rock,Hell Ain't A Bad Place To Be,254,8331286,Rock,1977
AC/DC,Let There Be Rock,Let There Be Rock,366,12021261,Rock,1977
AC/DC,Let There Be Rock,Overdose,369,12066294,Rock,1977
AC/DC,Let There Be Rock,Problem Child,325,10617116,Rock,1977
AC/DC,Let There Be Rock,Whole Lotta Rosie,323,10547154,Rock,1977
Accept,Balls to the Wall,Balls to the Wall,342,5510424,Rock,1984
Accept,Restless and Wild,Fast As a Shark,230,3990994,Rock,1982
Accept,Restless and Wild,Princess of the Dawn,375,6290521,Rock,1982
Accept,Restless and Wild,Restless and Wild,252,4331779,Rock,1982
Aerosmith,Big Ones,Amazing,356,11616195,Rock,1994
Aerosmith,Big Ones,Angel,307,9989331,Rock,1994
Aerosmith,Big Ones,Blind Man,240,7877453,Rock,1994
Aerosmith,Big Ones,Crazy,316,10402398,Rock,1994
Aerosmith,Big Ones,Cryin',309,10056995,Rock,1994
```

Test data sample as displayed in Microsoft Excel.

	A	B	C	D	E	F	G
1	artist_name	album_title	track	length_seconds	size_bytes	genre	release_year
2	AC/DC	For Those About To R	Breaking The Rules	263	8596840	Rock	1981
3	AC/DC	For Those About To R	C.O.D.	199	6566314	Rock	1981
4	AC/DC	For Those About To R	Evil Walks	263	8611245	Rock	1981
5	AC/DC	For Those About To R	For Those About To Rc	343	11170334	Rock	1981
6	AC/DC	For Those About To R	Inject The Venom	210	6852860	Rock	1981
7	AC/DC	For Those About To R	Let's Get It Up	233	7636561	Rock	1981
8	AC/DC	For Those About To R	Night Of The Long Kni	205	6706347	Rock	1981
9	AC/DC	For Those About To R	Put The Finger On You	205	6713451	Rock	1981
10	AC/DC	For Those About To R	Snowballed	203	6599424	Rock	1981
11	AC/DC	For Those About To R	Spellbound	270	8817038	Rock	1981
12	AC/DC	Let There Be Rock	Bad Boy Boogie	267	8776140	Rock	1977
13	AC/DC	Let There Be Rock	Dog Eat Dog	215	7032162	Rock	1977
14	AC/DC	Let There Be Rock	Go Down	331	10847611	Rock	1977
15	AC/DC	Let There Be Rock	Hell Ain't A Bad Place	254	8331286	Rock	1977
16	AC/DC	Let There Be Rock	Let There Be Rock	366	12021261	Rock	1977
17	AC/DC	Let There Be Rock	Overdose	369	12066294	Rock	1977
18	AC/DC	Let There Be Rock	Problem Child	325	10617116	Rock	1977
19	AC/DC	Let There Be Rock	Whole Lotta Rosie	323	10547154	Rock	1977
20	Accept	Balls to the Wall	Balls to the Wall	342	5510424	Rock	1984
21	Accept	Restless and Wild	Fast As a Shark	230	3990994	Rock	1982
22	Accept	Restless and Wild	Princess of the Dawn	375	6290521	Rock	1982
23	Accept	Restless and Wild	Restless and Wild	252	4331779	Rock	1982
24	Aerosmith	Big Ones	Amazing	356	11616195	Rock	1994
25	Aerosmith	Big Ones	Angel	307	9989331	Rock	1994
26	Aerosmith	Big Ones	Blind Man	240	7877453	Rock	1994
27	Aerosmith	Big Ones	Crazy	316	10402398	Rock	1994
28	Aerosmith	Big Ones	Cryin'	309	10056995	Rock	1994

It can be useful to open the CSV file in Excel to get a feel for the data but Excel is not a good way to interact with the data. Saving CSV files with Excel will add formatting

information that will make the file difficult to work with. Instead, create a copy of the CSV file to keep open in Excel for viewing while you write something to parse the file and enter it into your database. Java and Python (and many other languages) have libraries that will allow you to interact (insert, delete, update records, etc.) with an SQLite database. After implementing a schema, consider writing a program that reads the CSV file line-by-line, splits the record apart based on the commas, and inserts it into the database.

Test data requirements:
You will need to create your own patron and other data. You should put enough test data into your database in order to run the sample queries and test the scenarios. It is important that you have enough test data to (1) generate reasonable results for each of the scenarios described at the end of this document; and (2) generate good reports. For example, some of the reports require computing total; in that case you should not only return one data row.

Test scenarios requirements:
The database should be able to support the usual operations of an online music and video library. That is, that users may browse for artist, actors, checkout music/video and review their accounts. An administrator should be able to review the circulation and see what albums 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 5
- Project Worksheets 00 through 10
- Test data (CSV)

Project Worksheets (optional material to work/prepare Checkpoint discussions)

The project worksheet documents consist of questions applying lecture material to the project, almost week-by-week. You may find these worksheets to be useful guides in helping you develop some of the project ideas in order to complete each one of the Checkpoints.

DO NOT upload these worksheets to Carmen, they're optional (but recommended).

These worksheets will serve as a guideline when meeting and preparing the checkpoints with your project team members.

Here is an overview of the worksheets:

- Worksheet 00: database concepts and how they apply to the project.

- Worksheet 01: composing an Entity-Relationship diagram for the project database
- Worksheet 02: adding hierarchical relationships to your ER diagram to create an Enhanced Entity-Relationship diagram.
- Worksheet 03: using the EER diagram to create a schema for the database.
- Worksheet 04: creating update operations for the database and considering their impact on the constraints of the database.
- Worksheet 05: writing queries to your schema using relational algebra.
- Worksheet 06: show whether your database is fully normalized or not. If necessary, normalize your database to the third normal form.
- Worksheet 07: considering constraints that should be in the SQL schema along with tables and fields to write the SQL that will create your database.
- Worksheet 08: Install SQLite in your own computer and play around
- Worksheet 09: writing queries to your database in SQL
- Worksheet 10: Indexing and Transactions.

(Go to Canvas to download the document with the blank worksheets).