# A. Title Page

Lewis University  
CPSC 50900: Database Systems   
Spring 2025 Term Project

**E-Commerce Order Management System**

Saba Tahmeen

sabatahmeen@lewisu.edu

Work products stored in the Github repository https://github.com/rsabatahmeen/Data-models.git

Table of Contents

[A. Title Page 1](#_Toc186278180)

[Schedule of Milestones 1](#_Toc186278181)

[B. Initial Proposal 2](#_Toc186278182)

[C. Data Sources 2](#_Toc186278183)

[D. Alternative Ways to Store the Data 3](#_Toc186278184)

[E. Conceptual and Logical Models 3](#_Toc186278185)

[F. Physical Model 4](#_Toc186278186)

[G. Populate the database with data 5](#_Toc186278187)

[H. Data Manipulation Language (DML) Scripts 5](#_Toc186278188)

[I. Indexes 6](#_Toc186278189)

[J. Views 6](#_Toc186278190)

[K. Stored Programs (Stored Procedures, Stored Functions, Triggers) 6](#_Toc186278191)

[L. Transactions 7](#_Toc186278192)

[M. Database Security 7](#_Toc186278193)

[N. Locking and Concurrent Access 7](#_Toc186278194)

[O. Backing Up Your Database 8](#_Toc186278195)

[P. Programming 8](#_Toc186278196)

[Q. Suggested Future Work 8](#_Toc186278197)

[R. Activity Log 9](#_Toc186278198)

# Schedule of Milestones

Here is a schedule that shows when each milestone is due and what sections comprise it.

|  |  |
| --- | --- |
| Deadline | Sections for which you must demonstrate significant progress |
| February 4 at 11:59pm | a. Title page  b. Initial proposal  c. Data sources  d. Alternative ways to store the data  r. Activity Log – at least six entries covering the first two weeks |
| February 18  at 11:59pm | e. Conceptual and logical models  f. Physical model  g. Populate the database with data  r. Activity Log – at least six entries covering the past two weeks |
| March 4 at 11:59pm | h. Data manipulation language (DML) scripts  i. Indexes  j. Views  l. Transactions  m. Security  r. Activity Log – at least six entries covering the past two weeks |

The remaining sections – Triggers, Locking and Concurrency, Backup, and Programming, will be turned in with the final report, which is due March 16 at 11:59pm.

# B. Initial Proposal

*Description: You will describe the data you aim to store. What data will be storing? Why are you interested in this data? Why is it important? Where will the data come from? Who will use this data? What kind of application do you plan to build with it?*

*Rubric: Your response to each of these six questions will be graded out of 3 points.*

* *3 points: clear, complete descriptions that convey the importance and meaning of your data*
* *2 points: mostly clear descriptions, although some additional data would have helped in some sections*
* *1 point: necessary details are lacking in many of your responses.*

*You will also earn 2 additional points for coming up with a descriptive title for your project.*

*As you consider various ideas for your project, keep in mind that your database is going to have to store data for at least 8 different types of things. Each of these different “types of things” will become a table in the database you design and build. So, the idea can’t be so narrow that you can’t identify at least eight different types of things in it that you’d store data about.*

*Total points possible: 20*

**Introduction**

Customers now have a smooth online shopping experience thanks to e-commerce, which has changed how firms operate. Dealing with a range of entities, such as customers, goods, orders, payments, delivery, and reviews, is necessary for managing an online marketplace. These crucial elements are meant to be easily handled and stored by the E-Commerce Order Management System, which will facilitate smooth transactions and enhance business insights.

A relational database that can manage the complete order lifecycle—from product listings to order processing, payments, deliveries, and customer feedback—is what this project aims to provide. The system is perfect for practical implementation since it will use the real-world Olist Brazilian E-Commerce Public Dataset, which includes historical e-commerce transactions and business interactions.

**Data to Be Stored**

To provide effective order management, the database will store structured data in a number of linked tables. Among the essential data elements are:

* **Clients:** Individual client identification and geographic information (zip code, city, and state).
* **Products:** Product information, including ID, name, category, weight, size, and stock level.
* **Orders:** Order transactions that include timestamps, order status, customer ID, and order ID.
* **Order Items:** Details on the particular products that are part of each order, such as the product ID, order-item ID, and prices.
* **Payments:** Order-related payment transactions that include the method, amount, and installments of payment.
* **Shipping & Geolocation:** Shipping information, including anticipated arrival timeframes, delivery tracking, and zip code.
* **merchants (Vendors):** Details about how merchants handle product listings and order fulfillment.
* **Product Reviews:** Consumer evaluations and comments on their purchasing experiences.

Together, these data points aid in the comprehension of financial transactions, vendor performance, customer behavior, and logistics while preserving a strong and organized database for effective e-commerce operations.

**Significance and Relevance of the Data**

This information is essential for making sure an e-commerce platform runs well by:

* Effectively tracking fulfillment and managing orders.
* Managing financial activities and processing safe payments.
* Making the most of goods inventory to satisfy consumer needs.
* Improving client happiness via evaluations and comments.utilizing geolocation tracking to enhance delivery and logistics management.
* Producing information on business performance and sales patterns

Businesses may improve user experience, optimize operations, and make data-driven decisions to increase profitability by keeping and evaluating this data.

# C. Data Sources

*Description:* *Gather your data in text files. The text files may be csv, tab-delimited, xml, json, or some other custom format. Not all the files need be of the same type. Identify what each file contains by indicating where it came from, explaining in detail how it is structured, and describing how you will reorganize the data into a relational database. Post your data files to your GitHub repository, and provide samples of the data in your Word doc.*

*Rubric: Your work will be graded as follows:*

* *5 points: you gathered multiple data files that contain the data that will populate your databases. If you do not use multiple data files, you will not receive credit.*
* *5 points: you described the contents of the data files in detail, including referencing their origin and explaining how they were structured.*
* *3 points: you identify which fields you plan to include in your database, including their data types and any constraints you expect to impose on the data or steps you'll have to take to clean up the data.*
* *2 points: you post the data files to your GitHub account and make it possible for me to see them.*

*Total points possible: 15*

**Data Source**

The Olist Brazilian E-Commerce Public Dataset, a thorough and authentic dataset with millions of online transactions, is the dataset's original source. The dataset consists of:

* Order histories and customer demographics
* Product details, reviews, and seller information
* Tracking geolocation and shipping information
* Financial summaries and payment transactions

The project will use this dataset to model an actual e-commerce marketplace, enabling the use of normalization, indexing, query optimization, and database architecture concepts in a real-world setting.

**Dataset Breakdown**

Each dataset file contains structured data that maps directly to the key components of the database:

|  |  |  |
| --- | --- | --- |
| **Entity** | **Dataset File** | **Description** |
| Customers | olist\_customers\_dataset.csv | Stores customer details, including unique customer IDs, location (city, state, and zip code). |
| Orders | olist\_orders\_dataset.csv | Contains information on order transactions, including order ID, timestamps, order status, and estimated delivery date. |
| Order Items | olist\_order\_items\_dataset.csv | Links orders to specific products, detailing order-item ID, product ID, price, and freight cost. |
| Payments | olist\_order\_payments\_dataset.csv | Tracks payment transactions, including order ID, payment type (credit card, boleto, etc.), and total payment amount. |
| Products | olist\_products\_dataset.csv | Stores product details, including product ID, category, name, dimensions, and weight. |
| Sellers | olist\_sellers\_dataset.csv | Contains vendor details, including seller ID, location (city, state, zip code). |
| Shipping & Geolocation | olist\_geolocation\_dataset.csv | Stores delivery tracking details, linking customer locations with geographic coordinates (latitude, longitude). |
| Product Reviews | olist\_order\_reviews\_dataset.csv | Contains customer feedback, including review scores, review comments, and timestamps. |
| Product Categories | product\_category\_name\_translation.csv | Translates product categories from Portuguese to English for better accessibility. |

**Data integration and transformation:**

* The dataset will be normalized and handled so that it fits a relational database architecture.
* Foreign keys will be developed to preserve data integrity between tables.
* Indexes will be used to maximize query performance, especially in highly frequented tables such payments and orders.

# D. Alternative Ways to Store the Data

*Description: We will study alternatives to storing data in a relational database. Some of the alternatives come from several decades ago, including the hierarchical and network models. Some are newer options, such as NoSQL databases that use JSON or some other encoding. Describe in detail how to store the data using two alternatives to relational databases. Be sure to describe how you would implement the alternatives and the advantages and disadvantages of each.*

*Rubric: Your work will be graded as follows*

* *5 points for clearly describing how your data could be stored using one alternative to relational databases and what the advantages and disadvantages of that approach would be.*
* *5 points for clearly describing how your data could be stored using another alternative to relational databases and what the advantages and disadvantages of that approach would be.*

*Total points possible: 10*

**Alternative Ways to Store Data**

**1. NoSQL Databases (Document-Based Storage - MongoDB)**

E-commerce data may be stored as JSON-like documents rather than relational tables using a NoSQL document-oriented database, like MongoDB.

**Advantages:**

* The capacity to scale horizontally is a feature of NoSQL databases.
* Flexible Schema: This kind of schema does not need a set schema and permits many characteristics.
* Nested data queries are faster, requiring less intricate JOIN procedures.

**Disadvantages:**

* Data Redundancy – Updates can be inefficient, leading to duplication.
* Lack of ACID Compliance – Transactions involving multiple documents may not be as reliable as relational databases.
* Complex Queries – NoSQL requires aggregation pipelines instead of SQL queries.

**2. Hierarchical Model (XML-Based Storage)**

The Hierarchical model arranges data in a tree-like fashion wherein records have a parent-child link.

**Advantages:**

* Faster Navigation for Structured Data - Applied for one-to- many connections.
* Data Integrity: Provides referential integrity devoid of foreign keys.
* Perfect for XML-Based Data Exchange- a lot of e-commerce systems conduct transactions using XML.

**Disadvantages:**

* Not scalable for complex interactions- many-to- many relationships are challenging to execute.
* Rigid Structure: Schema revisions call for significant alterations.
* Limited Query Capabilities- Searching slows down searches by means of hierarchical traversal.

**Conclusion**

Although NoSQL and Hierarchical models provide good substitutes, relational databases like MySQL or Postgres remain the preferred option since:

* Integrity and consistent data.
* Help with sophisticated searches and indexing.
* ACID transactions monitoring orders and for financial security.

# E. Conceptual and Logical Models

*Description: First, come up with a conceptual model. The conceptual model identifies the entity sets and the relationships among them. For each relationship, identify the connectivity and the participation (optional or mandatory).*

*Now that you know the entity sets, the next step is to develop the logical model by adding attributes. For each entity set, identify the attributes that describe the entity set. This may include references to other entity sets that are involved in relationships. Then, identify the functional dependencies that exist among them. For each functional dependency, identify the determinants and the fields they determine, like this:*

*determinant, or, determinants  attributes, they, determine*

*This becomes the basis for identifying your entity sets, which will become your tables when we move to the physical model in the next section. The attributes listed on the left of the arrows are candidates to become your primary key attributes. Attributes that are references to other entity sets are candidates to become the foreign keys.*

*For entity sets that have multi-attribute determinants, replace them with surrogate keys. This makes it easier to identify each entity in the set and to define foreign keys.*

*Then apply normalization to make sure that your design satisfies First, Second, and Third Normal forms. For 1st Normal Form, make sure that all attributes are indivisible. This may require adding an entity set that lists values that appear in comma-separated lists as individual entities. For 2nd Normal Form, make sure there are no partial dependencies (this won’t be a problem if all your entity sets have single-attribute determinants). Finally, make sure all your entity sets are in 3rd Normal Form. This means that you have to split transitive dependencies into separate entity sets and add relationships between the original entity set and the new ones.*

*Finally, draw the logical model as an ERD. At this point, your design will have entity sets, their relationships, and their attributes. M:N relationships are acceptable at this point, as we’ll remove them in the physical model.*

*Rubric: Your work will be graded as follows:*

* *5 points for identifying all entity sets*
* *5 points for writing each relationship between entity sets as two sentences and correctly identifying their connectivity and participation.*
* *5 points for adding attributes to entity sets and writing the functional dependencies correctly. Replace multi-attribute determinants with surrogate keys.*
* *4 points for performing the normalization steps. Make sure your design is in 3rd Normal Form.*
* *5 points for drawing the ERD for the logical model. At this point, the ERD will show entity sets, relationships, attributes, and primary identifiers. The design may include M:N relationships at this point. We’ll get rid of those in the physical model.*

*Total points possible: 24*

ENTER YOUR RELATIONAL DATABASE DESIGN DESCRIPTION HERE. INCLUDE A PICTURE OF YOUR ERD.

# F. Physical Model

*Description: This is where you will complete your database design. Add data types, including size constraints, uniqueness constraints, and auto-incrementing for all attributes. Replace many-to-many relationships with two one-to-many relationships using bridge entity sets. Add additional entity sets that you think could be helpful for storing the acceptable values of particular attributes. (For example, if you were storing student data, valid student statuses might include Good Standing, Graduated, On Probation, Expelled. Put those in a table and create a relationship back to the student table). Draw the ERD for the physical model.*

*Using the final ERD, write the SQL DDL statements needed to create the database, its tables, and the relationships among them. Run these statements in MySQL to build your database. Provide screen shots that show the database you built in MySQL, including its tables and descriptions of some of the tables. To show a list of databases and a list of the tables in a particular database, use the show command. To see a description for a table, use the describe command.*

*Rubric: Your work will be graded as follows:*

* *3 points for introducing bridge entity sets (if necessary)*
* *3 points for adding data types and other constraints on the data.*
* *3 points for introducing other entity sets and their relationships that help enforce what values can be assigned to particular attributes (if necessary)*
* *5 points for drawing the ERD for the physical model. If you used Vertabelo, the resulting ERD must be free of errors and warnings*
* *6 points for generating the SQL scripts that build the database and then running the script in mysql. Demonstrate that the script built the database and its tables with screenshots that show that you ran the show and describe commands.*

*You will be penalized 4 points if your database doesn’t have at least 8 appropriately defined tables.*

*Total points possible: 20*

DESCRIBE THE STEPS YOU TOOK TO COMPLETE THE PHYSICAL MODEL. THEN SHOW THE ERD FOR THE PHYSICAL MODEL. THEN SHOW THE SQL COMMANDS THAT BUILD THE DATABASE. (FOR THIS, YOU MAY REFER TO A PARTICULAR FILE IN YOUR GITHUB REPOSITORY. MAKE SURE YOU INVITE ME AS A COLLABORATOR ON YOUR REPOSITORY SO THAT I CAN ACCESS THE SCRIPT.) FINALLY, SHOW SCREEN SHOTS THAT PROVE THAT YOU BUILT THE DATABASE AND ITS STRUCTURES IN MYSQL.

# G. Populate the database with data

*Description: You built the database in section F, and it now exists in mysql. Now populate it with your data. Take your original data source or sources and generate insert statements from them. Store the insert statements in a text file, and then use the mysql source command to run these insert statements to populate the various table structures. Generating the necessary insert statements may require writing Python scripts or manipulating Excel databases to convert the data from your original data sources.*

*Rubric: Your work will be grades as follows:*

* *Explain step-by-step and very clearly how you created the required SQL statements from your initial data. Write it as a set of instructions. 5 points*
* *Show the file of insert statements that you ran in MySQL. You may do this either by including the listing in this report or by identifying the file in your GitHub that contains the insert statements. Make sure I have access to your GitHub repository. 4 points*
* *Show screenshots of the data in your MySQL database. To do this, run select statements for each table and show screen shots of what is displayed: 5 points*

*Total points possible: 14*

ENTER YOUR DDL WORK HERE

# H. Data Manipulation Language (DML) Scripts

*Description: Write the SQL commands for twelve queries. Two queries should be insert statements, two should update statements, one should be a delete statement, one should be a simple select statement that selects a subset of the rows and columns from one table, two should be a select statements that select data from a joining of two tables, two should use summary functions to generate statistics about the data, one should be a multi-table query, and one should be another query of your choice. Show the queries and screenshots of the results in your Word document, and save your queries in a commented sql script to GitHub.*

*Rubric: Your work will be graded as follows:*

* *1 point each for the two insert statements*
* *1 point each for the two update statements*
* *1 point for the delete statement*
* *1 point for the simple select statement*
* *2 points each for the 2 join statements*
* *2 points each for the two that use summary statements*
* *2 points for the multi-table query*
* *2 points for the query of your choice.*
* *6 points for showing the query and a screenshot of the corresponding result set back-to-back for each of these queries in your Word document.*

*Total points possible: 24*

ENTER DML WORK HERE

# I. Indexes

*Description: Improve the performance of your design by adding indexes to various tables. Show the SQL needed to add the indexes. Explain why you chose the ones you added. Explain how you would demonstrate the impact the indexes had on the performance of various queries.*

*Rubric: Your work will be graded as follows:*

* *3 points for clearly defining at least three indexes and explaining why you chose them.*
* *3 points for showing the sql needed to generate the indexes*
* *2 points for explaining how you would demonstrate the performance improvement afforded by the indexes.*

*Total points possible: 8*

ENTER YOUR INDEX WORK HERE

# J. Views

*Description: Add two views to your database to provide easy access to combinations of data from multiple tables.*

*Rubric: Your work will be graded as follows:*

* *3 points for including the SQL for generating the two views in your Word document*
* *3 points for including screenshots for the data contained in each view in your Word document*
* *3 points for explaining why each view is a valuable addition to your database*

*Total points possible: 9*

ENTER YOUR WORK WITH VIEWS HERE

# K. Stored Programs (Stored Procedures, Stored Functions, Triggers)

*Description: Add a stored procedure, stored function or trigger to a table and demonstrate using it.*

*Rubric: Your work will be graded as follows:*

* *3 points for including the SQL for the stored program (procedure, function, or trigger in your Word document*
* *3 points for clearly explaining the purpose of the stored program*
* *3 points for a screenshot and explanation that shows the stored program in action.*

*Total points possible: 9*

ENTER YOUR WORK WITH STORED PROGRAMS HERE

# L. Transactions

*Description: Demonstrate that you know how to define and use a transaction. Why are transactions important for ensuring ACID behavior?*

*Rubric: Your work will be graded as follows:*

* *5 points for clearly explaining the importance of transactions to ensuring ACID behavior*
* *3 points for including a screenshot and accompanying explanation of a MySQL transaction.*

*Total points possible: 8*

ENTER YOUR WORK WITH TRANSACTIONS HERE

# M. Database Security

*Description: Identify the different kinds of users who will use your database. Write GRANT statements to define the privileges for these different kinds of users.*

*Rubric: Your work will be graded as follows:*

* *4 points for clearly identifying and describing the various kinds of users who will use the databases and identifying and justifying what privileges each should have.*
* *4 points for writing GRANT statements that assign privileges to these different kinds of users.*
* *4 points for demonstrating with screenshots that your GRANT statements do distinguish among different kinds of users in regard to what they can do with the database.*

*Total points possible: 12*

ENTER YOUR WORK WITH DATABASE SECURITY HERE

# N. Locking and Concurrent Access

*Description: Explain the purpose of locking tables and show how to do that to prevent inconsistencies that may arise in your data when concurrent transactions take place.*

*Rubric: Your work will be graded as follows:*

* *3 points for clearly explaining an example that shows why you should lock tables to prevent inconsistencies.*
* *3 points for providing a screenshot and accompanying explanation of locking tables.*

*Total points possible: 5*

ENTER YOUR WORK WITH LOCKING AND CONCURRENT ACCESS HERE

# O. Backing Up Your Database

*Description: How you will back up your database. What commands will you issue? How frequently will the commands run? How can they be automated? Where will the backups be stored?*

*Rubric: Your work will be graded as follows:*

* *6 points for clearly explaining and justifying your database backup strategy, including the frequency with which you will back up the database, how you will automate backups, where you will store them, and how you will secure them. You will earn three points for addressing each factor (frequency, location, automation, and security)*
* *2 points for providing a screenshot of the command you would issue to back up the database and for including a portion of the resulting file.*

*Total points possible: 8*

ENTER YOUR WORK ON DATABASE BACKUPS HERE

# P. Programming

*Description: Write a Python, Java, or PHP program that generates a report that contains a subset of the data from your database. Include the code for your Python program in your Word document, and also post the program to your GitHub repository.*

*Rubric: Your work will be graded as follows:*

* *10 points for writing a Python script (and including its code in the Word doc) that will pull data from a database and store it to a text file and present it to the screen. Your code must have comments in it that explain how it works. You will be awarded 3 points for successfully connecting to the database, 3 points for successfully querying it, and 4 points for presenting the data to the screen and to a file. Internal comments count for 2 points.*
* *2 points for posting the code to GitHub*
* *6 points for showing a screenshot of your running the script and showing the results it produces on the screen.*

*Total points possible: 18*

ENTER YOUR PYTHON, PHP, or JAVA DATABASE PROGRAMMING WORK HERE

# Q. Suggested Future Work

*Description: Describe the limitations of your current database and explain how you or someone else could improve the design to address these shortcomings. Also describe how you might take advantage of leverage cloud services to increase the performance and availability of your database. Finally, explain the advantages and disadvantages of storing your data in a NoSQL format instead.*

*Rubric: Your work will be graded as follows:*

* *3 points for clearly describing the limitations of your databases*
* *3 points for explaining how you would address these shortcomings*
* *3 points for explaining how you might migrate the database to the cloud and describing what advantages you might gain from doing that.*
* *3 points for explaining the advantages and disadvantages of storing your data in a document-based NoSQL format instead.*

*Total points possible: 12*

ENTER YOUR SUGGESTED FUTURE WORK IDEAS HERE

# R. Activity Log

*Description: As an appendix, the team will keep a frequently updated diary or log of their activity. What did you or your team study in this class each day? What did you learn? What did you accomplish or build or design? You don't have to enter something every day, but there should be at least three entries each week. Since we have eight weeks, that means you should make 3 posts to the Activity Log each week, for a total of at least 24 posts. Each post will be worth 1 point.*

*If you are working as part of a team, make sure you clearly identify which team member worked on which tasks. The Activity Log should help me figure out how each team member contributed to the project. If I cannot discern who worked on what aspects of the project from the activity log, no points will be awarded for it.*

*Total points possible: 24*

|  |  |
| --- | --- |
| **Date** | **Activity Log** |
| January 19, 2025 | Learned about Data vs. Information, storage models, and why relational databases are preferred over file systems. |
| January 22, 2025 | Studied Conceptual Data Modeling, defining entities and relationships. |
| January 25, 2025 | Explored NoSQL and Hierarchical Storage Models, comparing them with relational databases. |
| January 27, 2025 | Reviewed Logical Data Models, focusing on keys, attributes, and normalization. |
| January 29, 2025 | Learned about Physical Data Models and SQL implementation (DDL statements). |
| January 31, 2025 | Tried Entity Relationship Diagrams (ERDs) using Crow’s Foot notation. |