# 1. Python SQLite Game Shop Point of Sale - Design

## Contents



Time required: 60 minutes

**NOTE:** This project builds on your SQLite project from last semester.

- Follow all directions carefully and accurately.
- Think of the directions as minimum requirements.

## SQL Tutorial

- https://www.w3schools.com/sql/sql_create_table.asp

## SQLite with Python Tutorials

- SQLite in Python - (Humorous short video introduction to Python SQLite)
- SQLite Databases with Python - Full Course – FreeCodeCamp.org
- https://www.sqlitetutorial.net

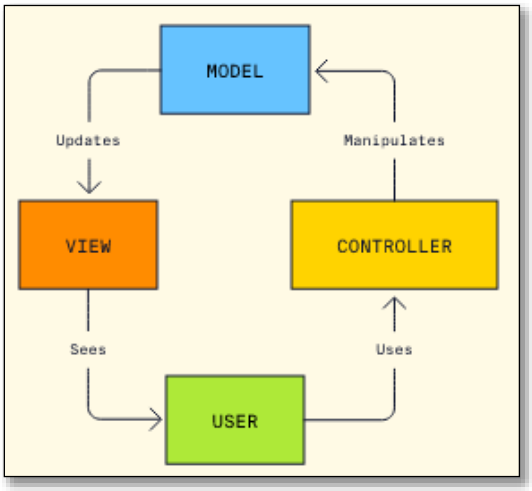
## MVC (Model View Controller)

MVC is short for Model, View, and Controller. MVC is a popular way of organizing your code. The big idea behind MVC is that each section of your code has a purpose, and those purposes are different. Some of your code holds the data of your app, some of your code makes your app look nice, and some of your code controls how your app functions.

**Model:** Model code typically reflects real-world things. This code can hold raw data, or it will define the essential components of your app. For instance, if you were building a To-do app, the model code would define what a "task" is and what a "list" is – since those are the main components of a todo app.

**View:** View code is made up of all the functions that directly interact with the user. This is the code that makes your app look nice, and otherwise defines how your user sees and interacts with it.

**Controller:** Controller code acts as a liaison between the Model and the View, receiving user input and deciding what to do with it. It's the brains of the application, and ties together the model and the view.




## SQLite Relational Database

SQLite is a relational database. We can create tables related by primary keys. We will design our databases using an ERD (Entity Relationship Diagram). www.lucidchart.com is free web-based diagram site used in these SQLite tutorials.

In this tutorial, we will create a games database with related tables, game, customer and publisher along with a bridge or transaction table.

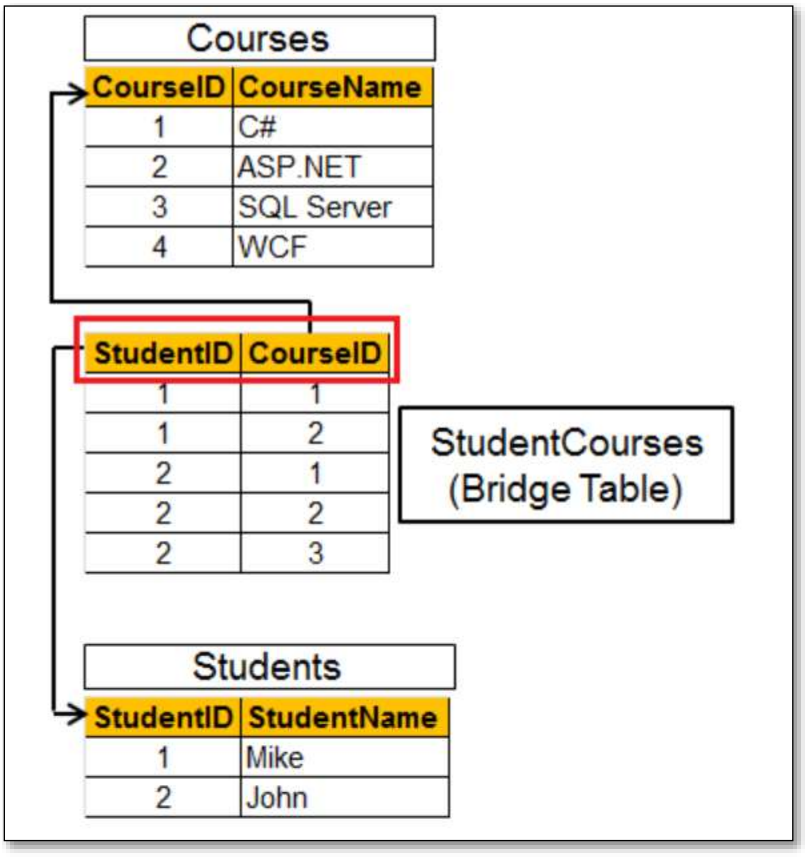
## Bridge Table

A **bridge table** (also known as an associative or junction table) is used in database design to handle many-to-many relationships between two tables. It acts as a link between two tables by holding foreign keys that reference the primary keys of the related tables. This allows for the association of multiple records from one table with multiple records from another.

For example, consider a database with Students and Courses tables. A student can enroll in multiple courses, and a course can have multiple students. To model this many-to-many relationship, you would create a bridge table, say Enrollments, with the following structure:

- student_id (foreign key referencing Students)
- course_id (foreign key referencing Courses)

This way, each record in the Enrollments table represents a unique combination of a student and a course.



## Business Rules

**NOTE:** Modify the business rules of your database from last semester.

Business rules establish how the entities interact and their relationship with each other. For example:

- A publisher can produce many games.
- A game can only have one publisher.
- A customer can purchase many games.
- A game can be purchased by many customers.

## Game Shop Database ERD

www.lucidchart.com is free web-based diagram site used in these SQLite tutorials.

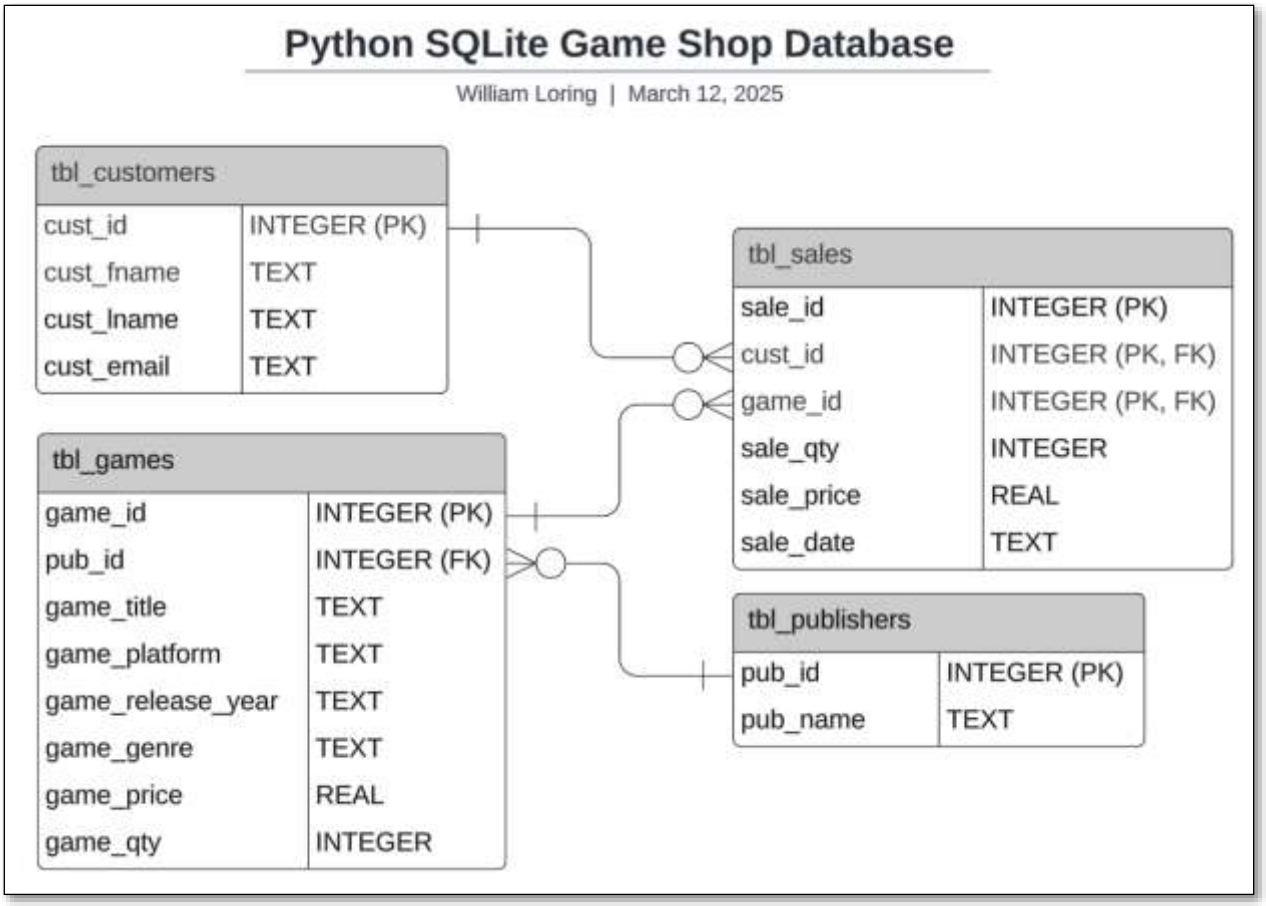
**NOTE:** Modify the ERD of your database from last semester.

An Entity Relationship Diagram, also known as ERD, is a diagram that displays the relationship of entities stored in a database. ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities (tables), attributes (fields), and relationships.

An ERD of our game Point of Sale program is based on the following business rules:

- A publisher can produce many games.
- A game can only have one publisher.
- A customer can purchase many games.
- A game can be purchased by many customers.

This is a crow's foot diagram illustrating a one to many relationship between the entities.




Reference: https://vertabelo.com/blog/crow-s-foot-notation/

**PK (Primary Key):** A primary key is a column or a set of columns in a table whose values uniquely identify a row in the table. A primary key typically has no meaning other than to uniquely identify each record.

**FK (Foreign Key):** A foreign key is a column or a set of columns in a table whose values correspond to the values of the primary key in another table.

Crows foot notation indicates the record relationship between the tables.

| | |
|---|---|
|  | One or many |
|  | One |

## Assignment 1: Draw ERD

1. Take your ERD from your previous database project.
2. Add the necessary tables and relationships to your Crow's Foot ERD for a relational table version of your project similar to the example.
3. Create a Word Documentation file to store a screenshot of your ERD and your other planning items.

## Assignment 2: Data Dictionary

**NOTE:** Modify the data dictionary of your database from last semester.

Let's create your database and tables. Fill in your names in your planning document.

This is the structure of our sample database. This is called a data dictionary.

| Table Name | Field Name | Field Format | Description |
|---|---|---|---|
| tbl_publishers | pub_id | INTEGER – PK | Game publisher unique identifier |
| tbl_publishers | pub_name | TEXT | Game publisher name |
| tbl_games | game_id | INTEGER – PK | Game unique identifier |
| tbl_games | pub_id | INTEGER – FK | Foreign key referencing the publisher associated with the game |
| tbl_games | game_title | TEXT | Game title |
| tbl_games | game_platform | TEXT | Game platform |
| tbl_games | game_release_year | INTEGER | Game release year |
| tbl_games | game_genre | TEXT | Game genre |
| tbl_games | game_price | REAL | Price of game |
| tbl_games | game_qty | INTEGER | Quantity of games on hand |
| tbl_sales | sale_id | INTEGER (PK) | Sale unique identifier |
| tbl_sales | sale_qty | INTEGER | Sale quantity |
| tbl_sales | sale_date | TEXT | Date of sale |
| tbl_sales | cust_id | (PK, FK) | Composite key for sales and customers |
| tbl_sales | game_id | (PK, FK) | Composite key for sales and games |
| tbl_customers | cust_id | INTEGER - PK | Customer unique identifier |

| | | | |
|---|---|---|---|
| tbl_customers | cust_f_name | TEXT | First name |
| tbl_customers | cust_l_name | TEXT | Last name |
| tbl_customers | cust_email | TEXT | Email Address |

**Assignment:**

1. Update your data dictionary for your database in your planning document.
2. Add a Customer Email address to the customers table.

## Assignment Submission

1. Attach the program files.
2. Attach screenshots showing the successful operation of the program.
3. Submit in Blackboard.