MAPÚA UNIVERSITY
SCHOOL OF ELECTRICAL, ELECTRONICS, AND COMPUTER ENGINEERING

Experiment 5:
Data Modeling and Database Systems

CPE106L (Software Design Laboratory)

PALOY, EURO GABRIEL
TAMSON, JOMARI ANGELO
ZALAMEDA, ANDREA KAYLA

Group No.: **10**
Section: **E01**

PreLab

Readings, Insights, and Reflection

A Guide to SQL, VBID: 9780357419830

(TAMSON)

Chapter 1: This chapter covers the basics of storing and managing structured data, showing how the relational model keeps data accurate and efficient. The various case studies show real-world uses of SQL databases. These examples highlight how good data management improves efficiency, decision-making, and customer service, proving that databases are key to business success.

Chapter 2: This chapter covers database design, focusing on structure, relationships, and normalization to improve accuracy and efficiency. It explains key concepts like primary keys, functional dependencies, and reducing redundancy. A well-designed database prevents errors and ensures smooth data management, highlighting the importance of mastering SQL.

(PALOY)

Chapter 1: This chapter presents the use of SQL by highlighting two distinct organizations in Chapter 1. These companies use databases to handle their data, and exercises showing how to access and modify data in those databases are included throughout the chapter.

Chapter 2: The fundamentals of database design are covered in chapter 2. It defines important words and describes the relationships between various database objects. Additionally, the chapter offers best practices and crucial recommendations for building and organizing a well-designed database.

(ZALAMEDA)

Chapter 1: I gained knowledge about the basic idea of a database in this chapter. I learned that a database is an ordered set of structured data that has been arranged for effective storing, retrieval, and manipulation. I also learned more about the many kinds of databases, including NoSQL databases, which grow horizontally and provide flexibility in handling unstructured data, and relational databases, which manage data using tables and SQL. Building reliable applications and systems that need effective data management and retrieval requires an understanding of databases.

Chapter 2: This chapter introduced me to the idea of normalization as well as other crucial database topics. I found that developing effective databases depends heavily on database principles like entities, properties, relationships, and keys. I also learned more about normalization, which is the act of arranging data to reduce dependencies and redundancies, enhancing data integrity and lowering anomalies.

Core Python Programming, VBID: 9789351198918

(TAMSON)

Chapter 24: This chapter of the book covers databases used with Python, including MySQL, SQLite, PostgreSQL, and NoSQL options like MongoDB. It explains database connectivity using libraries like MySQL-connector and PyMySQL, showing how Python simplifies database interaction. This highlights Python's value in data-driven applications.

(PALOY)

Chapter 24: The fundamentals of using databases to handle and visualize data were covered in Chapter 24. We looked at how to easily import SQL databases into MongoDB Compass after converting them to JSON format. The chapter also discussed the various ways to work with MongoDB, including using a user-friendly GUI or the command line. We practiced executing CRUD operations and grouping the data into collections in addition to storing it. Lastly, we tried using Python to automate database operations, which improved the efficiency of data changes and retrieval.

(ZALAMEDA)

Chapter 24: This chapter covered data visualization, database conversion to JSON files using SQLite Browser, and importation into MongoDB Compass. We learnt how to manage MongoDB using the command line and graphical user interface (GUI), construct collections, and execute CRUD operations. Additionally, we showed how to use Python to view and modify the database's contents.

Python Project, 9781118908891

(TAMSON)

Chapter 3: This chapter covers relational databases, explaining how data is structured in tables with relationships. It introduces SQL, focusing on DML for data handling and DDL for database structure. The chapter highlights SQL's key role in database management and its integration with Python for data processing.

(PALOY)

Chapter 3: Chapter 3 is devoted to data management with Python. It looks at how data can be efficiently organized and retrieved by storing it in databases and files. Additionally, the chapter discusses basic methods for using the Python programming environment to search, sort, and access stored data.

(ZALAMEDA)

Chapter 3: This chapter taught me how to use Python for data analysis and SQL for data management. I learned how well SQL can manage operations like building databases, specifying tables, and utilizing DML and DDL statements to manipulate data. I also

learned how Python's data analysis packages, such as pandas and NumPy, enhance SQL's data management skills by offering strong tools for processing, visualizing, and extracting insights from big datasets.

QUESTION AND ANSWERS

1. **What are DML and DDL statements in Structured Query Language? Give examples of each.**

- DML (Data Manipulation Language) is used to manage and modify data in a database, including commands like SELECT (retrieve data), INSERT (add data), UPDATE (modify data), and DELETE (remove data). DDL (Data Definition Language) is used to define and manage database structures, with commands like CREATE (make a new table or database), ALTER (modify an existing table), and DROP (delete a table or database). DML works with the data itself, while DDL focuses on the database structure.

2. **What are the categories of SQLite Functions? Give 3 examples of each category.**

- Scalar, Aggregate, and Window functions are the three primary categories into which SQLite functions fall. Each time they are called, scalar functions act on discrete values and deliver a single result. After operating on a set of values, aggregate functions produce a single output that summarizes the set. Last but not least, Window Functions compute a value over a range of table rows that are connected to the current row.

3. **How do you check if you have SQLite installed in system using the Linux terminal?**

- Use the command "sqlite3 --version" in the terminal to see if SQLite is installed on your Linux machine. If SQLite is installed on your machine, running sqlite3 --version in the terminal will either reveal the version number of SQLite or an error message stating that the command could not be found.

InLab

- **Objectives**
- **Tools Used**
    - Visual Studio Code
    - SQLite
    - DB Browser
- **Procedure**

In Figure l.1, The DB Browser table creation process is shown in the screenshot. The interface displays a table creation dialog box where you may define the columns of the table and configure the name, data type, and any constraints (primary key, not null, etc.) for the table. The screenshot also shows how to add a row of data into the newly made table and how to enter values into the designated columns straight from the DB Browser graphical user interface.

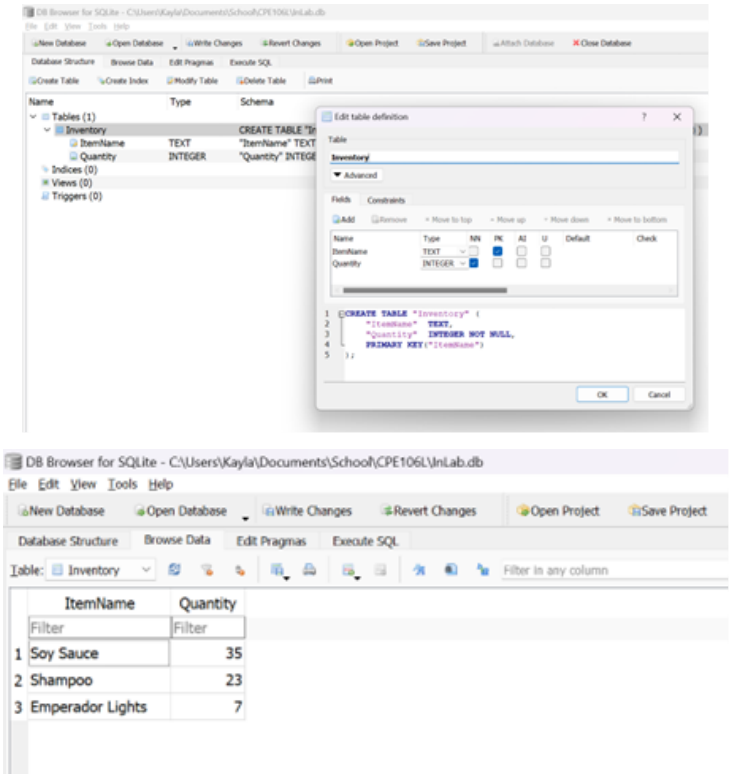

Figure I.1 *Creating a table in DB Browser and inserting data*

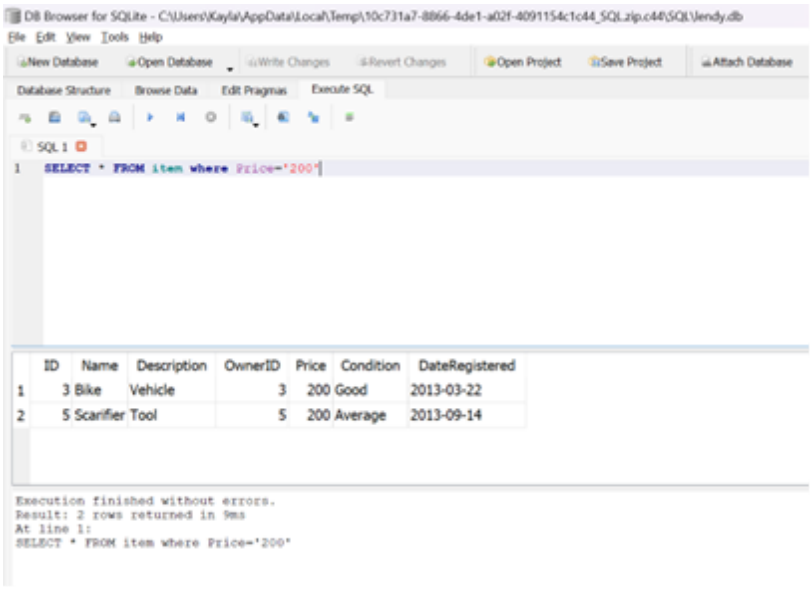

Figure I.2.1 *Getting to know the DB Browser's "SELECT" function with constraints.*

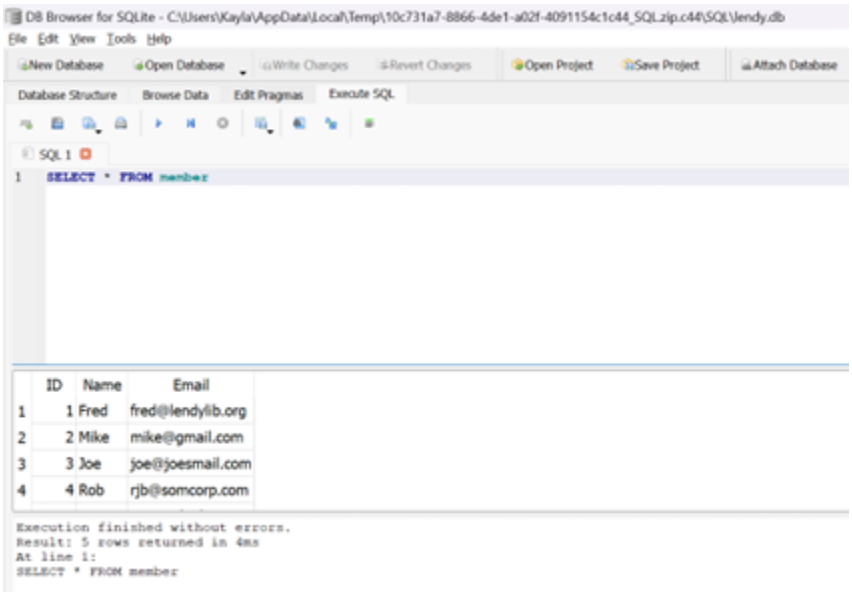

Figure I.2.2 *Using the DB Browser, try the "SELECT" function in "cassel" both with and without constraints.*

In Figures I.2.1 and I.2.2, The screenshot shows how to use the SELECT function in DB Browser on the "cassel" database. It presents two cases: one in which SELECT is used without restrictions, showing all of the data in a given table or set of columns, and another in which SELECT is used with restrictions, such as WHERE clauses, filtering the results according to predefined criteria like matching particular values or ranges.

PostLab

A. Machine Problems

1. Colonial Adventure Tours is considering offering outdoor adventure classes to prepare people to participate in hiking, biking, and paddling adventures. Only one class is taught on any given day. Participants can enroll in one or more classes. Classes are taught by the guides that Colonial Adventure employs. Participants do not know who the instructor for a particular class will be until the day of the class. Colonial Adventure Tours needs your help with the database design for this new venture. In each step, represent your answer using the shorthand representation and a diagram. Use crow's foot notation for the diagram. Follow the sample SQLite chinook database ERD (Download it from Blackboard Course Materials)

a) For each participant, list his or her number, last name, first name, address, city, state, postal code, telephone number, and date of birth.

b) For each adventure class, list the class number, class description, maximum number of people in the class, and class fee.

c) For each participant, list his or her number, last name, first name, and the class number, class description, and date of the class for each class in which the participant is enrolled.

d) For each class, list the class date, class number, and class description; and the number, last name, and first name of each participant in the class

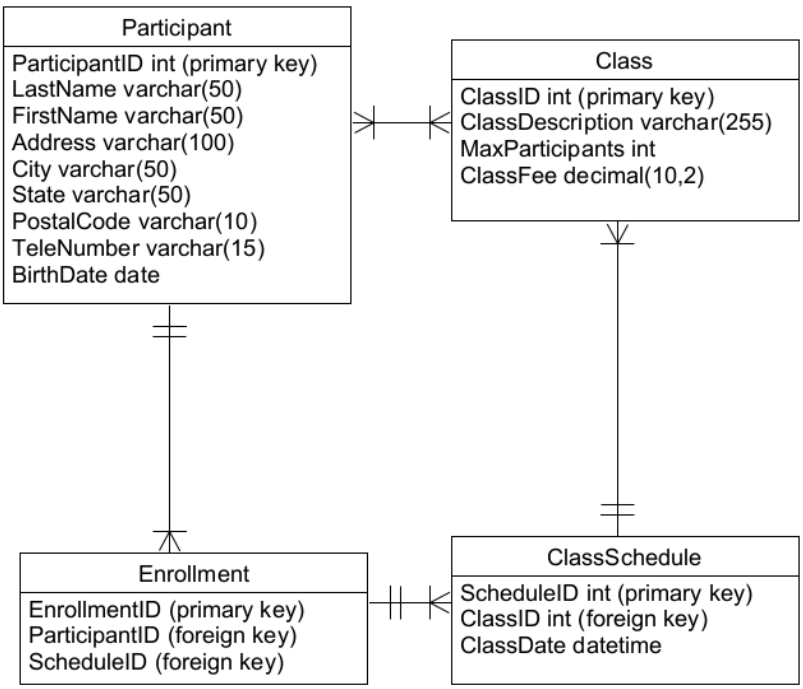

Figure 1.3 ERD Diagram of Post Lab A1

The **Colonial Adventure Tours** database manages class scheduling and participant enrollments efficiently. The **Particant** table stores personal details like name, address, and phone number. Since participants can join multiple classes, the **Enrollment** table tracks their registrations for specific sessions. The **Class** table holds details like description, capacity, and cost, while the **ClassSchedule** table records when each class takes place, allowing the same class to be scheduled multiple times.

2. Solmaris Condominium Group has many condos that are available as weekly vacation rentals. Design a database to meet the following requirements:

a) For each renter, list his or her number, first name, middle initial, last name, address, city, state, postal code, telephone number, and email address.

b) For each property, list the condo location number, condo location name, address, city, state, postal code, condo unit number, square footage, number of bedrooms, number of bathrooms, maximum number of persons that can sleep in the unit, and the base weekly rate.

c) For each rental agreement, list the renter number, first name, middle initial, last name, address, city, state, postal code, telephone number, start date of the rental, end date of the rental, and the weekly rental amount. The rental period is one or more weeks.

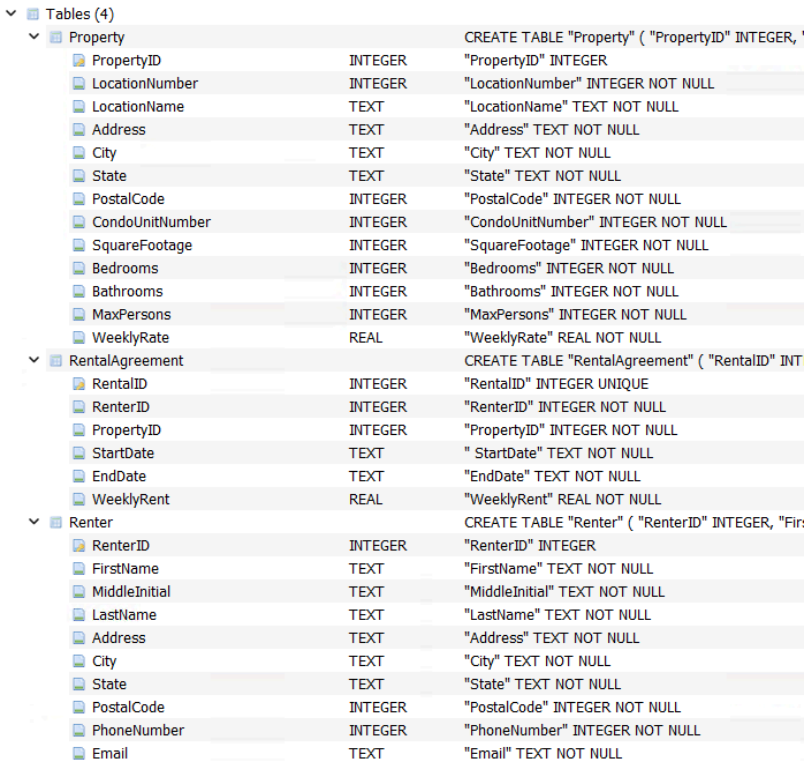

Figure 1.4 Solaris Condominium Group Database Tables in DB Browser

The database was created through DB Browser for SQLite, providing easy instructions on creating a database based on SQLite.

3. Use SQLite commands to complete the following exercises.

a) Create a table named ADVENTURE_TRIP. The table has the same structure as the TRIP table shown in Figure 3-2 below except the TRIP_NAME column should use the VARCHAR data type and the DISTANCE and MAX_GRP_SIZE columns should use the NUMBER data type. Execute the command to describe the layout and characteristics of the ADVENTURE_TRIP table.

b) Add the following row to the ADVENTURE_TRIP table: trip ID: 45; trip name: Jay Peak; start location: Jay; state: VT; distance: 8; maximum group size: 8; type: Hiking and sea- son: Summer. Display the contents of the ADVENTURE_TRIP table.

c) Delete the ADVENTURE_TRIP table.

d) Open the script file (SQLServerColonial.sql) to create the six tables and add records to the tables. Revise the script file so that it can be run in the DB Browser.

e) Confirm that you have created the tables correctly by describing each table and comparing the results to the figures shown below. Confirm that you have added all data correctly by viewing the data in each table and comparing the results to Figures 1-4 through 1-8 shown below.

```
1  PRAGMA table_info(ADVENTURE_TRIP);
2
3
```

	cid	name	type	notnull	dflt_value	pk
1	0	TRIP_ID	REAL	0	NULL	0
2	1	TRIP_NAME	TEXT	1	NULL	0
3	2	START_LOCATION	TEXT	1	NULL	0
4	3	STATE	TEXT	1	NULL	0
5	4	DISTANCE	REAL	1	NULL	0
6	5	MAX_GRP_SIZE	REAL	1	NULL	0
7	6	TYPE	TEXT	1	NULL	0
8	7	SEASON	TEXT	1	NULL	0

Figure 1.5 ADVENTURE_TRIP ran through SQLite

```
2   VALUES (45, 'Jay Peak', 'Jay', 'VT', 8, 8, 'Hiking', 'Summer');
3   SELECT * FROM ADVENTURE_TRIP;
4
```

	TRIP_ID	TRIP_NAME	START_LOCATION	STATE	DISTANCE	MAX_GRP_SIZE	TYPE	SEASON
1	45.0	Jay Peak	Jay	VT	8.0	8.0	Hiking	Summer

Figure 1.6 Adding a row to the database of ADVENTURE_TRIP

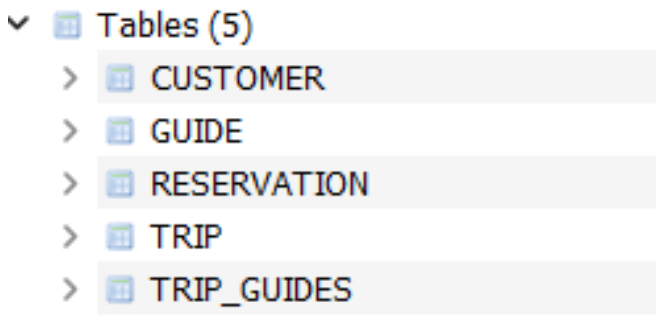

Figure 1.7 SQLColonial.sql file revised to run in DB Browser

Table: CUSTOMER — Filter in any column

	CUSTOMER_NUM	LAST_NAME	FIRST_NAME	ADDRESS	CITY	STATE	POSTAL_CODE	PHONE
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	101	Northfold	Liam	9 Old Mill Rd.	Londonderry	NH	03053	603-555-7563
2	102	Ocean	Arnold	2332 South St. Apt 3	Springfield	MA	01101	413-555-3212
3	103	Kasuma	Sujata	132 Main St. #1	East Hartford	CT	06108	860-555-0703
4	104	Goff	Ryan	164A South Bend Rd.	Lowell	MA	01854	781-555-8423
5	105	McLean	Kyle	345 Lower Ave.	Wolcott	NY	14590	585-555-5321
6	106	Morontoia	Joseph	156 Scholar St.	Johnston	RI	02919	401-555-4848
7	107	Marchand	Quinn	76 Cross Rd.	Bath	NH	03740	603-555-0456
8	108	Rulf	Uschi	32 Sheep Stop St.	Edinboro	PA	16412	814-555-5521
9	109	Caron	Jean Luc	10 Greenfield St.	Rome	ME	04963	207-555-9643
10	110	Bers	Martha	65 Granite St.	York	NY	14592	585-555-0111
11	112	Jones	Laura	373 Highland Ave.	Somerville	MA	02143	857-555-6258
12	115	Vaccari	Adam	1282 Ocean Walk	Ocean CITY	NJ	08226	609-555-5231
13	116	Murakami	Iris	7 Cherry Blossom St.	Weymouth	MA	02188	617-555-6665
14	119	Chau	Clement	18 Ark Ledge Ln.	Londonderry	VT	05148	802-555-3096
15	120	Gernowski	Sadie	24 Stump Rd.	Athens	ME	04912	207-555-4507
16	121	Bretton-Borak	Siam	10 Old Main St.	Cambridge	VT	05444	802-555-3443
17	122	Hefferson	Orlagh	132 South St. Apt 27	Manchester	NH	03101	603-555-3476
18	123	Barnett	Larry	25 Stag Rd.	Fairfield	CT	06824	860-555-9876
19	124	Busa	Karen	12 Foster St.	South Windsor	CT	06074	857-555-5532
20	125	Peterson	Becca	51 Fredrick St.	Albion	NY	14411	585-555-0900
21	126	Brown	Brianne	154 Central St.	Vernon	CT	06066	860-555-3234

Figure 1.8 CUSTOMER table from Colonial Database

Table: GUIDE

Filter in any column

	GUIDE_NUM	LAST_NAME	FIRST_NAME	ADDRESS	CITY	STATE	POSTAL_CODE	PHONE_NUM	HIRE_DATE
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	AM01	Abrams	Miles	54 Quest Ave.	Williamsburg	MA	01096	617-555-6032	6-3-2012
2	BR01	Boyers	Rita	140 Oakton Rd.	Jaffrey	NH	03452	603-555-2134	3-4-2012
3	DH01	Devon	Harley	25 Old Ranch Rd.	Sunderland	MA	01375	781-555-7767	1-8-2012
4	GZ01	Gregory	Zach	7 Moose Head Rd.	Dummer	NH	03588	603-555-8765	11-4-2012
5	KS01	Kiley	Susan	943 Oakton Rd.	Jaffrey	NH	03452	603-555-1230	4-8-2013
6	KS02	Kelly	Sam	9 Congaree Ave.	Fraconia	NH	03580	603-555-0003	6-10-2013
7	MR01	Marston	Ray	24 Shenandoah Rd.	Springfield	MA	01101	781-555-2323	9-14-2015
8	RH01	Rowan	Hal	12 Heather Rd.	Mount Desert	ME	04660	207-555-9009	6-2-2014
9	SL01	Stevens	Lori	15 Riverton Rd.	Coventry	VT	05825	802-555-3339	9-5-2014
10	UG01	Unser	Glory	342 Pineview St.	Danbury	CT	06810	203-555-8534	2-2-2015

Figure 1.9 GUIDE table from Colonial Database

	RESERVATION_ID	TRIP_ID	TRIP_DATE	NUM_PERSONS	TRIP_PRICE	OTHER_FEES	CUSTOMER_NUM
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1600001	40.0	3-26-2016	2.0	55.0	0.0	101
2	1600002	21.0	6-8-2016	2.0	95.0	0.0	101
3	1600003	28.0	9-12-2016	1.0	35.0	0.0	103
4	1600004	26.0	10-16-2016	4.0	45.0	15.0	104
5	1600005	39.0	6-25-2016	5.0	55.0	0.0	105
6	1600006	32.0	6-18-2016	1.0	80.0	20.0	106
7	1600007	22.0	7-9-2016	8.0	75.0	10.0	107
8	1600008	28.0	9-12-2016	2.0	35.0	0.0	108
9	1600009	38.0	9-11-2016	2.0	90.0	40.0	109
10	1600010	2.0	5-14-2016	3.0	25.0	0.0	102
11	1600011	3.0	9-15-2016	3.0	25.0	0.0	102
12	1600012	1.0	6-12-2016	4.0	15.0	0.0	115
13	1600013	8.0	7-9-2016	1.0	20.0	5.0	116
14	1600014	12.0	10-1-2016	2.0	40.0	5.0	119
15	1600015	10.0	7-23-2016	1.0	20.0	0.0	120
16	1600016	11.0	7-23-2016	6.0	75.0	15.0	121
17	1600017	39.0	6-18-2016	3.0	20.0	5.0	122
18	1600018	38.0	9-18-2016	4.0	85.0	15.0	126
19	1600019	25.0	8-29-2016	2.0	110.0	25.0	124
20	1600020	28.0	8-27-2016	2.0	35.0	10.0	124
21	1600021	32.0	6-11-2016	3.0	90.0	20.0	112
22	1600022	21.0	6-8-2016	1.0	95.0	25.0	119
23	1600024	38.0	9-11-2016	1.0	70.0	30.0	121
24	1600025	38.0	9-11-2016	2.0	70.0	45.0	125
25	1600026	12.0	10-1-2016	2.0	40.0	0.0	126
26	1600029	4.0	9-19-2016	4.0	105.0	25.0	120
27	1600030	15.0	7-25-2016	6.0	60.0	15.0	104

Figure 1.10 RESERVATION table from Colonial Database

Table: TRIP

Filter in any column

	TRIP_ID	TRIP_NAME	START_LOCATION	STATE	DISTANCE	MAX_GRP_SIZE	TYPE	SEASON
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1.0	Arethusa Falls	Harts Location	NH	5.0	10.0	Hiking	Summer
2	2.0	Mt Ascutney - North Peak	Weathersfield	VT	5.0	6.0	Hiking	Late Spring
3	3.0	Mt Ascutney - West Peak	Weathersfield	VT	6.0	10.0	Hiking	Early Fall
4	4.0	Bradbury Mountain Ride	Lewiston-Auburn	ME	25.0	8.0	Biking	Early Fall
5	5.0	Baldpate Mountain	North Newry	ME	6.0	10.0	Hiking	Late Spring
6	6.0	Blueberry Mountain	Batchelders Grant	ME	8.0	8.0	Hiking	Early Fall
7	7.0	Bloomfield - Maidstone	Bloomfield	CT	10.0	6.0	Paddling	Late Spring
8	8.0	Black Pond	Lincoln	NH	8.0	12.0	Hiking	Summer
9	9.0	Big Rock Cave	Tamworth	NH	6.0	10.0	Hiking	Summer
10	10.0	Mt. Cardigan - Firescrew	Orange	NH	7.0	8.0	Hiking	Summer
11	11.0	Chocorua Lake Tour	Tamworth	NH	12.0	15.0	Paddling	Summer
12	12.0	Cadillac Mountain Ride	Bar Harbor	ME	8.0	16.0	Biking	Early Fall
13	13.0	Cadillac Mountain	Bar Harbor	ME	7.0	8.0	Hiking	Late Spring
14	14.0	Cannon Mtn	Franconia	NH	6.0	6.0	Hiking	Early Fall
15	15.0	Crawford Path Presidentials Hike	Crawford Notch	NH	16.0	4.0	Hiking	Summer
16	16.0	Cherry Pond	Whitefield	NH	6.0	16.0	Hiking	Spring
17	17.0	Huguenot Head Hike	Bar Harbor	ME	5.0	10.0	Hiking	Early Fall
18	18.0	Low Bald Spot Hike	Pinkam Notch	NH	8.0	6.0	Hiking	Early Fall
19	19.0	Mason's Farm	North Stratford	CT	12.0	7.0	Paddling	Late Spring

Figure 1.11 TRIP table from Colonial Database

	TRIP_ID	GUIDE_NUM
	Filter	Filter
1	1.0	GZ01
2	1.0	RH01
3	2.0	AM01
4	2.0	SL01
5	3.0	SL01
6	4.0	BR01
7	4.0	GZ01
8	5.0	KS01
9	5.0	UG01
10	6.0	RH01
11	7.0	SL01
12	8.0	BR01

Figure 1.12 TRIP_GUIDES table from Colonial Database

Two tasks were in this activity, create ADVENTURE_TRIP and revising the .sql file into an importable DB file for DB Browser. The ADVENTURE_TRIP table was created to sort classes and demonstrate SQL data.

While the Colonial SQL file revises the file format of .sql into .db to make it open on DB Browser for SQLite.

B. Debugging and Sample Run of Python program connection to your created SQLite database (with edited screengrabs and discussion)

IMPORTANT: Include figure numbers and labels. Edit your screengrabs

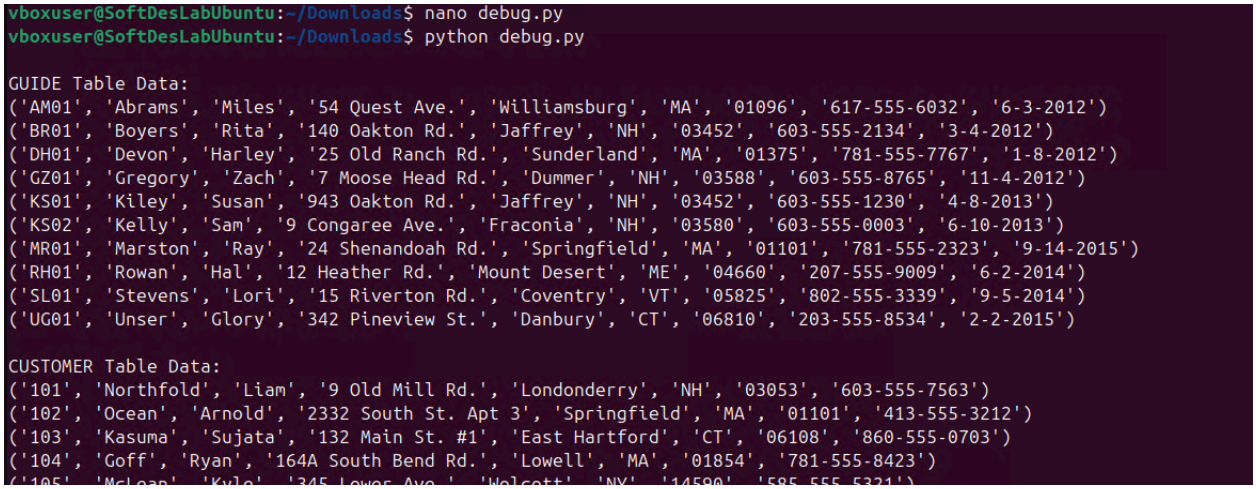

Figure 1.13 Python Code Output of the SQLite Database

The Python Code can be found in the GitHub Group Repo and the OneDrive Folder displaying all tables. The code was able to display all the tables in the database in an organized matter.