**BUS 440 Database Management**

**Project 1: Uptown Rentals Musical Instrument Rental System**

## Use Case:

*As you learned from Assignment 1, Uptown Rentals is a new startup business that rents musical instruments to individuals. You have already designed a relational database management system for this company, which categorizes musical instruments by type (e.g., Strings, Woodwinds, Brass, etc.) and rental tier (e.g., Basic, Premium, etc.).*

*You also learned that the store has multiple employees with a primary role (e.g., cashier, consultant, admin clerk, manager) and email address. Store staff manage the rentals and track the returns. Customers must register with the store, providing their full name, age, address, and contact information. Each customer can rent more than one instrument at a time. Each rental includes the serial number of the rented instrument, daily rental fee, rental date, due date, and return date. Late returns incur a daily overdue fee based on the instrument's rental tier and the days overdue of the rented instrument. Fines are calculated per day late. Customers cannot rent new instruments until outstanding fines are paid. Uptown Rentals also tracks the condition of each instrument it rents. If the instrument is damaged upon return, or if the instrument needs periodic maintenance, the employee initiates a repair (maintenance), and the business tracks the repair cost and the maintenance date.*

*The business has been capturing data related to its inventory and rentals on spreadsheets but wants to migrate to a database solution. Figure 1 shows a sample of some of the current data:*

*Figure 1*

| Serial Num | Customer Name | Rental Date | Instrument Type | Rental Tier | Contact Email | Staff Name | Return Date | Due Date | Daily Rental Fee | Daily Overdue Fee |
|---|---|---|---|---|---|---|---|---|---|---|
| 12878 | Joseph Dow | 12/4/2024 | Flute | Basic | jdow@gmail.com | Liz Conners | 12/19/2024 | 12/20/2024 | 35.00 | 0 |
| 76887 | Ric Martin | 1/5/2015 | Trumpet | Premium | Rm5@nc.rr.com | Liz Conners | 1/12/2025 | 1/10/2015 | 40.00 | 5.00 |
| 76657 | Ric Martin | 1/5/2015 | Bass Guitar | Premium | Rm5@nc.rr.com | Liz Conners | 1/12/2025 | 1/10/2015 | 43.00 | 5.00 |
| 98223 | Lauren Cox | 1/5/2015 | Flute | Basic | Lc49@gmail.com | Tom Lindel | 1/12/2025 | 1/12/2015 | 35.00 | 0 |
| 12878 | Luke Diago | 12/20/2024 | Flute | Basic | jdow@gmail.com | Liz Conners | 12/29/2024 | 12/30/2024 | 35.00 | 0 |
| 98223 | Sue Mann | 1/13/2015 | Flute | Basic | sm3@gmail.com | Tom Lindel | 1/22/2025 | 1/22/2015 | 35.00 | 0 |
| 98223 | Lauren Cox | 1/25/2015 | Flute | Basic | Lc49@gmail.com | Tom Lindel | 1/30/2025 | 1/30/2015 | 35.00 | 0 |

## Instructions:

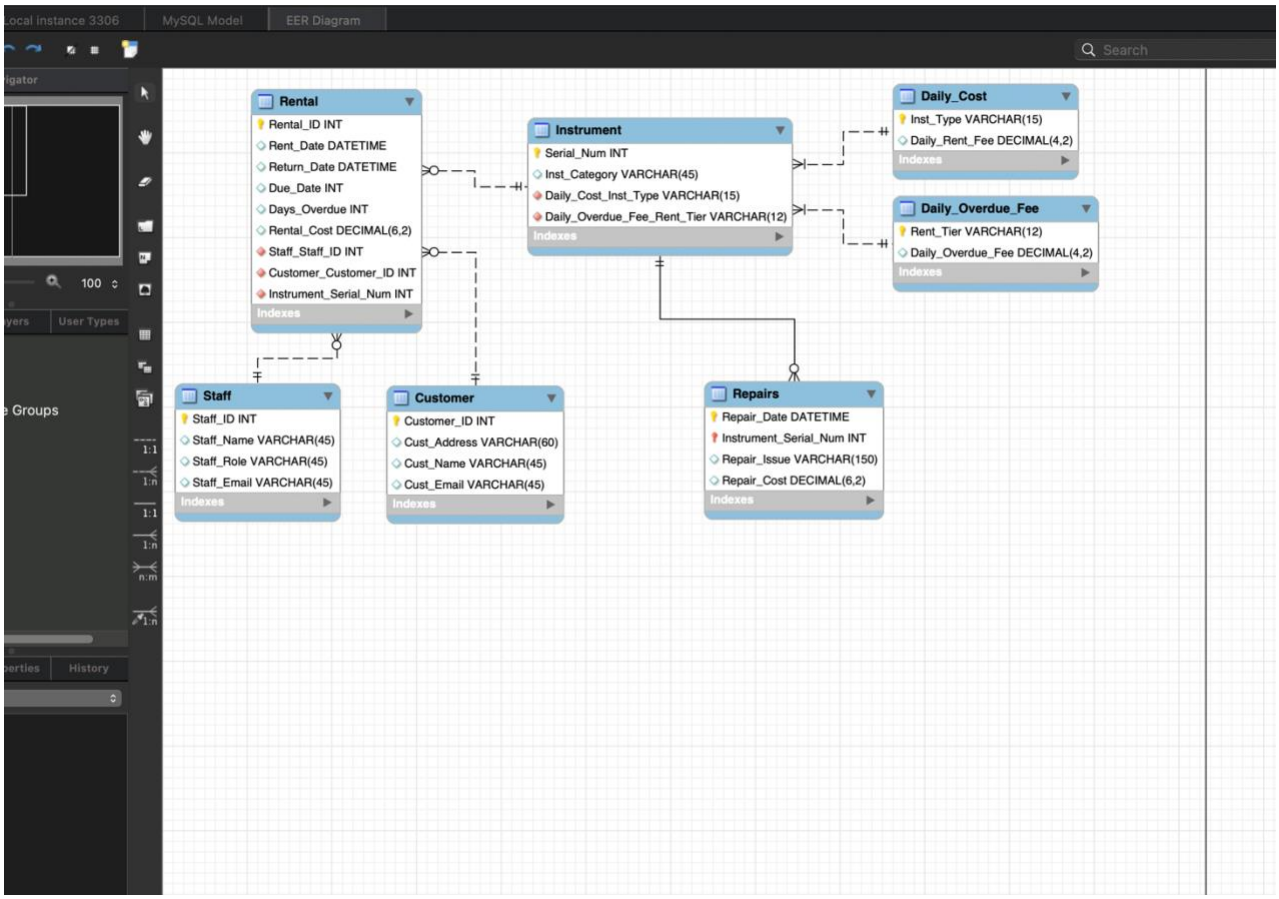
**From Assignment 1, take the feedback and modify your design. Then, build the schema and database from your modified design:**

1. Build your EER model in MySQL Workbench and name it **uptown**. Include a screenshot of the model here.

2. Synchronize the EER to the build. Note that this process will create an empty schema.

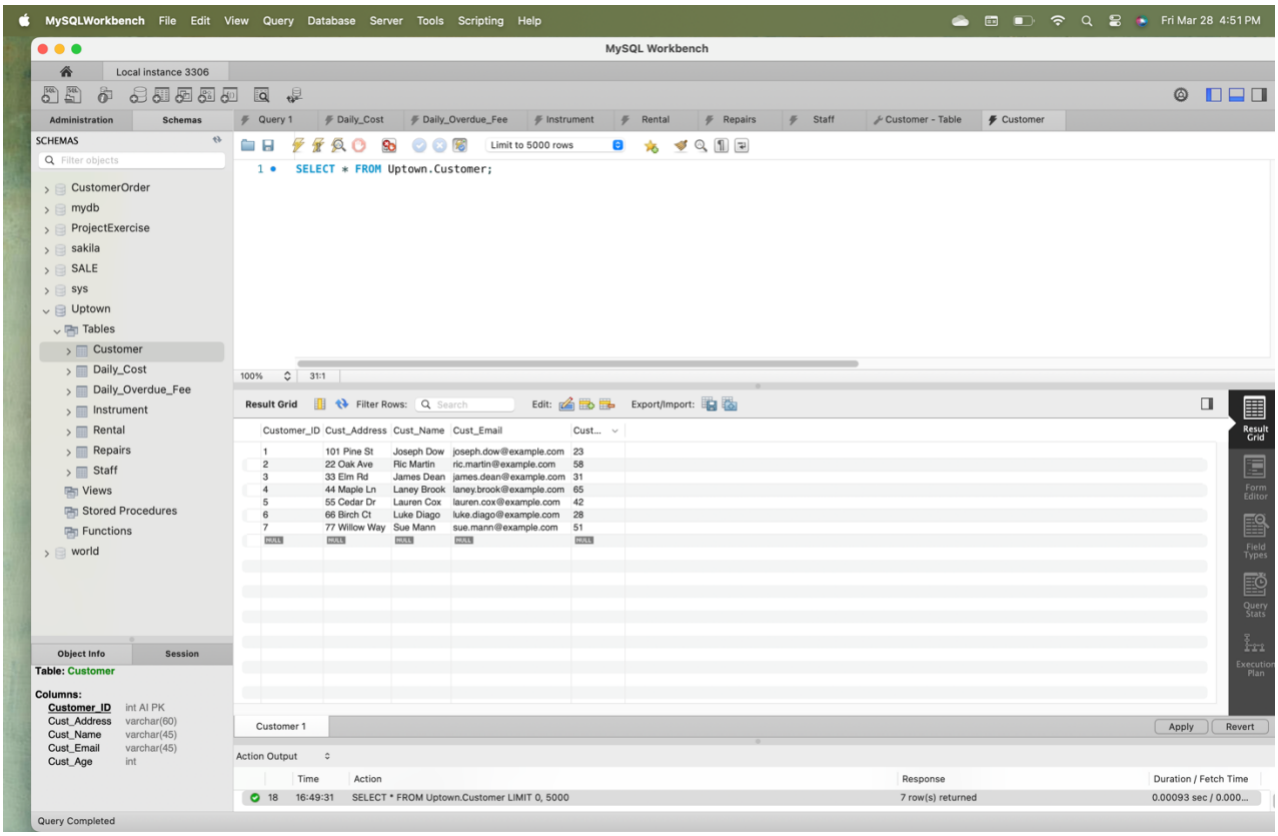
3. Verify the build.

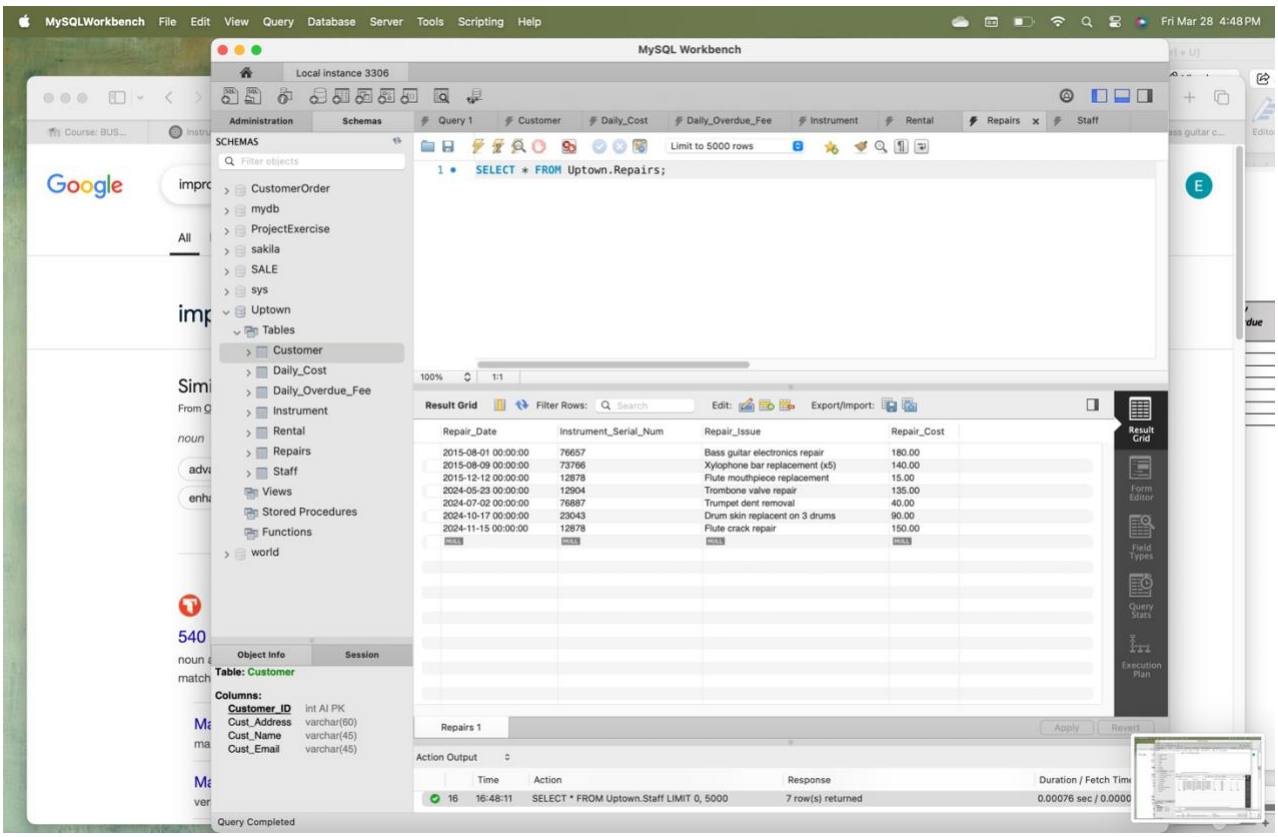
a. Verify all tables and their attributes and PK. For example, after synchronization, you could create a ***rental*** table that would include attributes such as the *serialNum, rentalDate, customerID, etc.*
b. Verify the FKs representing each relationship in your EER.

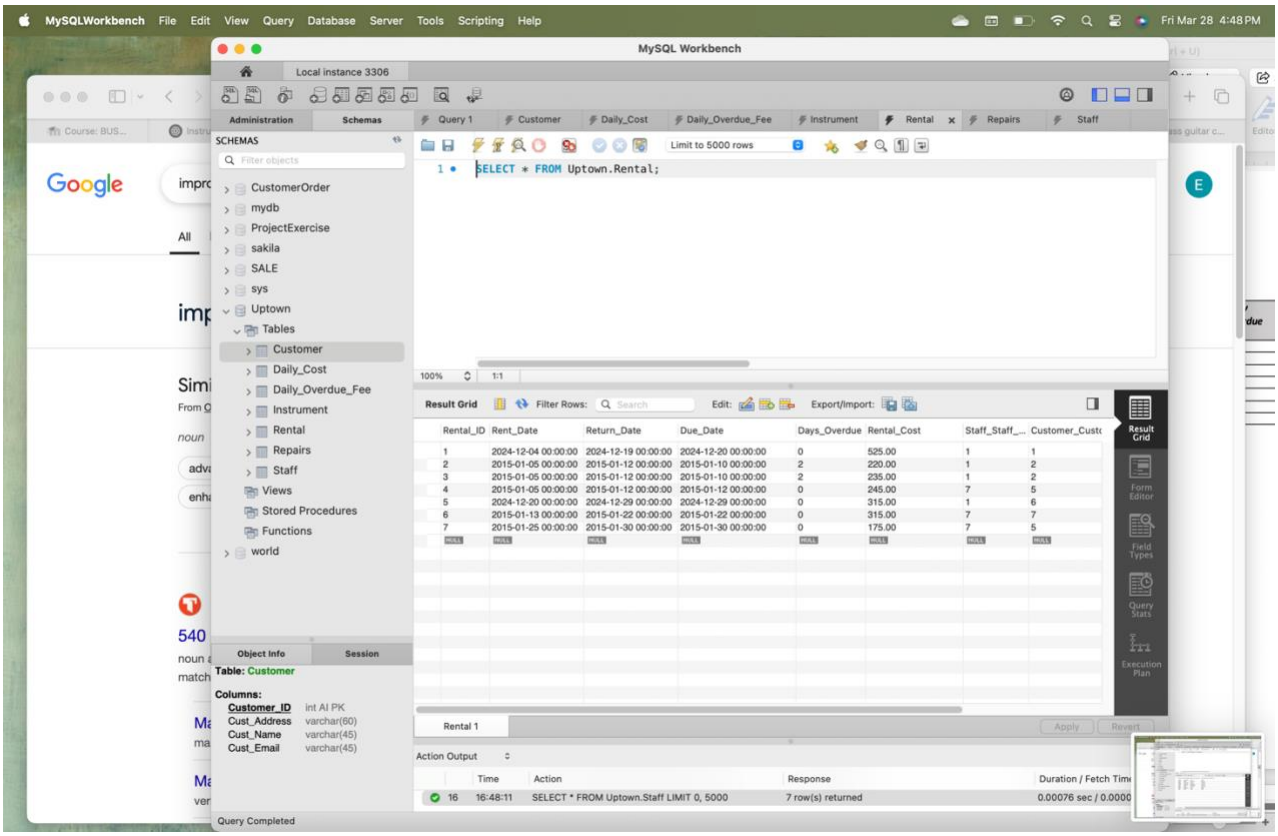


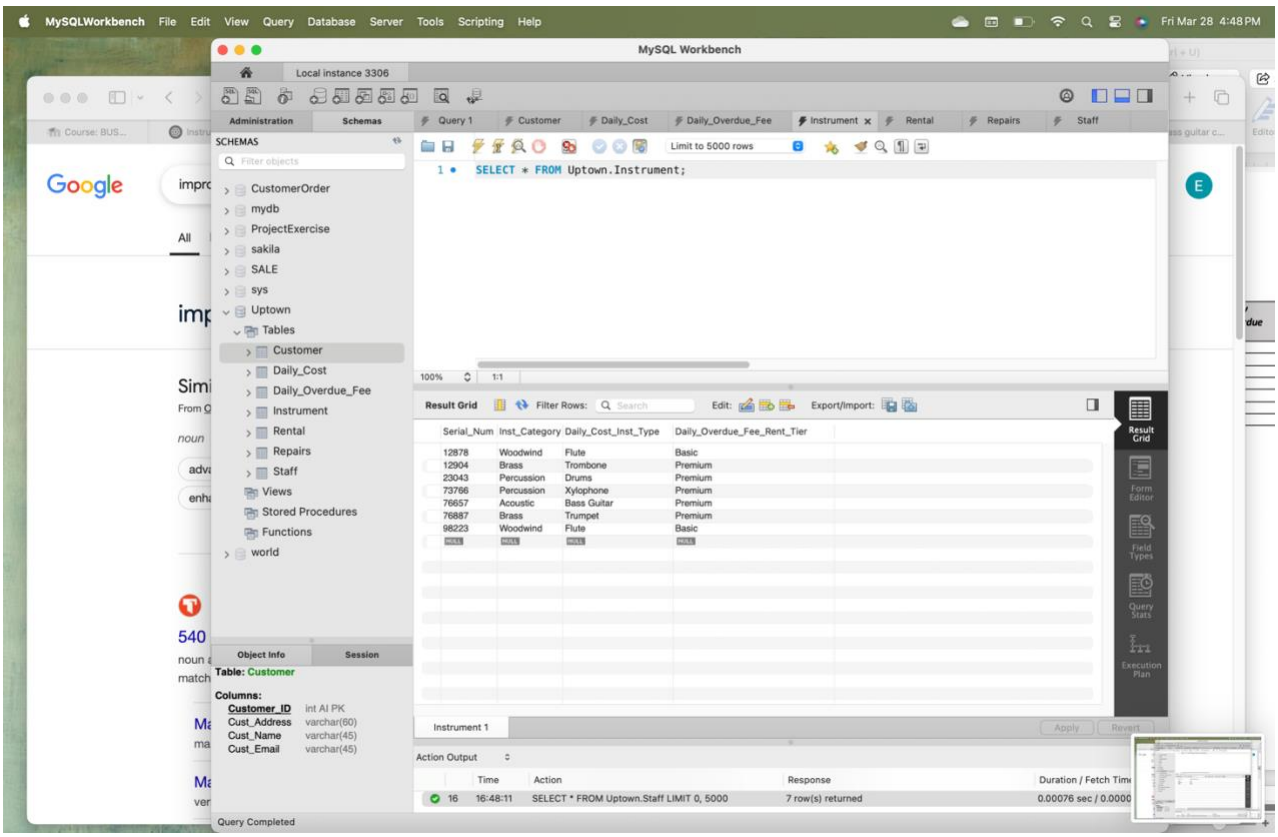

**Populate the tables with data:**

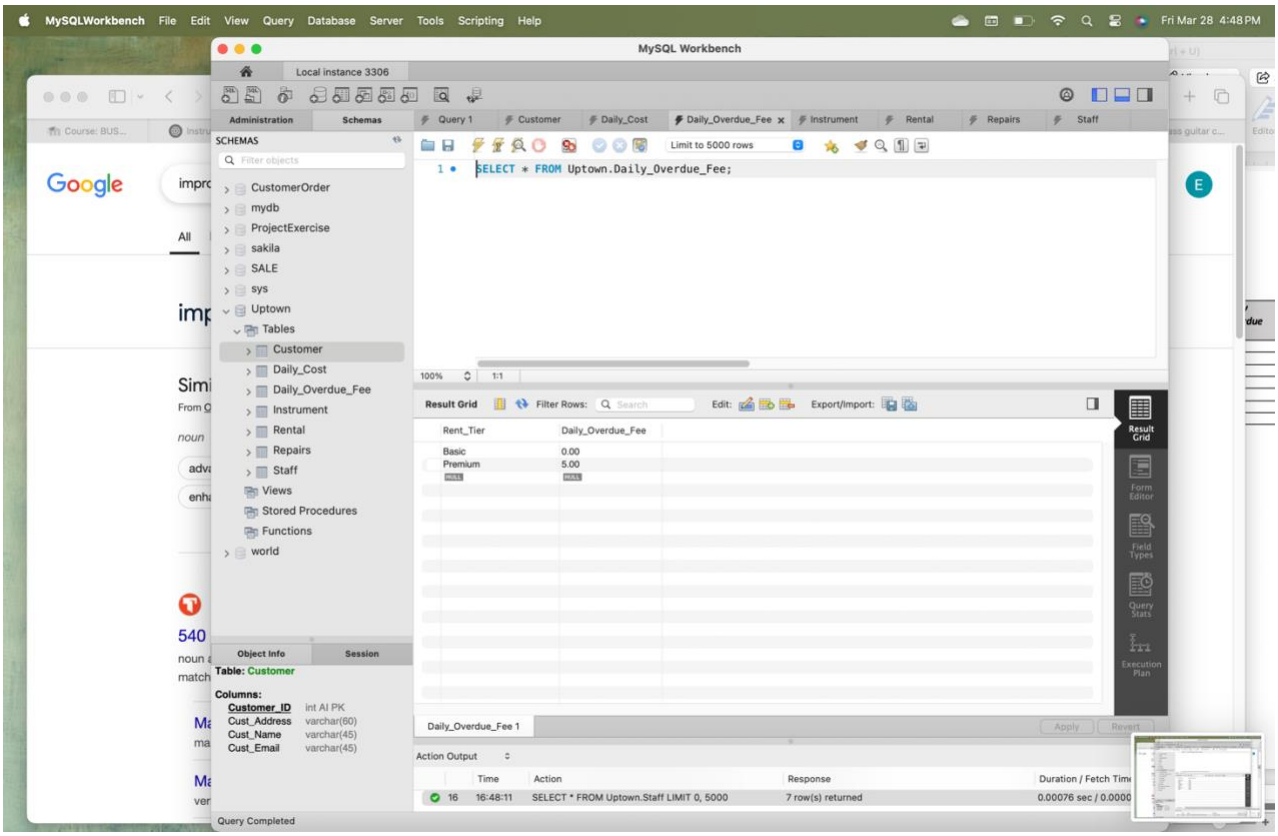
4. Enter the data into the appropriate tables according to your data model design (at least 7 rows per table). (Note: Everyone should type the data from Figure 1 into the appropriate tables. However, the remaining data should be your own and different from others – each person should derive their own hypothetical data.)
    a. Populate each table of the database by typing the data directly into each table, by importing it, or by using SQL code to insert it
    b. Populate the parent tables first, then the child tables.

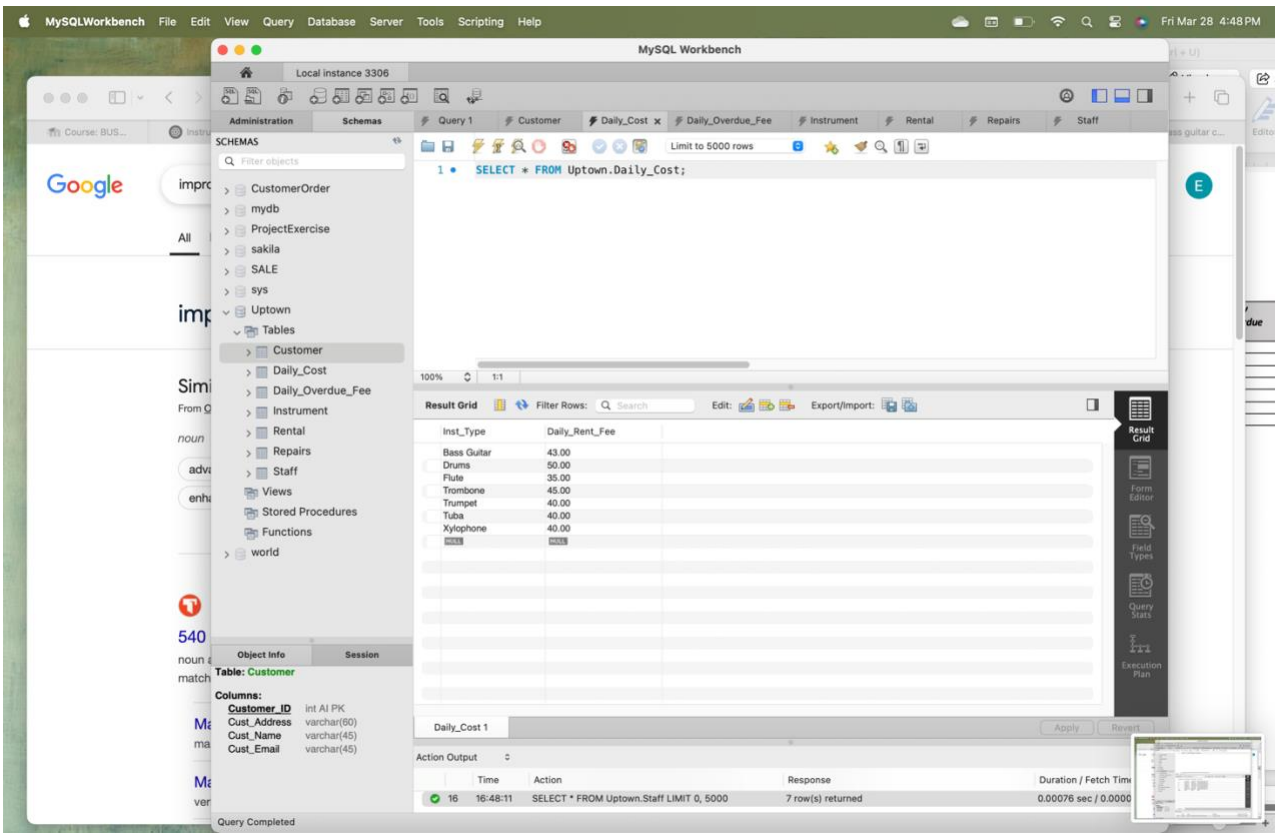

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Fri Mar 28 4:51 PM
MySQL Workbench
Local instance 3306
SELECT * FROM Uptown.Customer;
Result Grid
Customer_ID Cust_Address Cust_Name Cust_Email
1 101 Pine St Joseph Dow joseph.dow@example.com 23
2 22 Oak Ave Ric Martin ric.martin@example.com 58
3 33 Elm Rd James Dean james.dean@example.com 31
4 44 Maple Ln Laney Brook laney.brook@example.com 65
5 55 Cedar Dr Lauren Cox lauren.cox@example.com 42
6 66 Birch Ct Luke Diago luke.diago@example.com 28
7 77 Willow Way Sue Mann sue.mann@example.com 51
Table: Customer
Columns:
Customer_ID int AI PK
Cust_Address varchar(60)
Cust_Name varchar(45)
Cust_Email varchar(45)
Cust_Age int
Action Output
18 16:49:31 SELECT * FROM Uptown.Customer LIMIT 0, 5000 7 row(s) returned 0.00093 sec / 0.000...
Query Completed

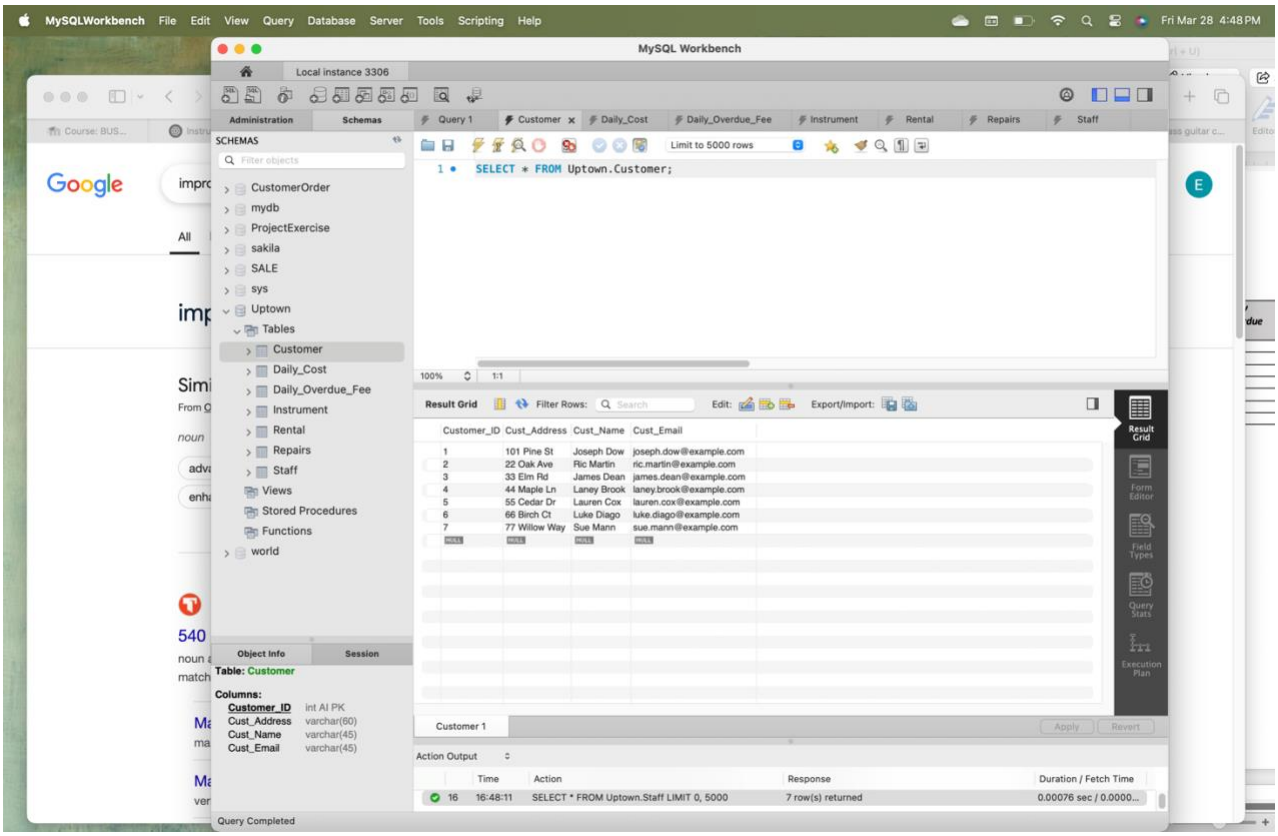

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Fri Mar 28 4:48 PM
MySQL Workbench
Local instance 3306
SELECT * FROM Uptown.Repairs;
Result Grid
Repair_Date Instrument_Serial_Num Repair_Issue Repair_Cost
2015-08-01 00:00:00 76657 Bass guitar electronics repair 180.00
2015-08-09 00:00:00 73766 Xylophone bar replacement (x5) 140.00
2015-12-12 00:00:00 12878 Flute mouthpiece replacement 15.00
2024-05-23 00:00:00 12904 Trombone valve repair 135.00
2024-07-02 00:00:00 76887 Trumpet dent removal 40.00
2024-10-17 00:00:00 23043 Drum skin replacent on 3 drums 90.00
2024-11-15 00:00:00 12878 Flute crack repair 150.00
Table: Customer
Columns:
Customer_ID int AI PK
Cust_Address varchar(60)
Cust_Name varchar(45)
Cust_Email varchar(45)
Action Output
16 16:48:11 SELECT * FROM Uptown.Staff LIMIT 0, 5000 7 row(s) returned 0.00076 sec / 0.0000
Query Completed


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Fri Mar 28 4:48 PM
MySQL Workbench
Local instance 3306
Administration Schemas
SCHEMAS
CustomerOrder
mydb
ProjectExercise
sakila
SALE
sys
Uptown
Tables
Customer
Daily_Cost
Daily_Overdue_Fee
Instrument
Rental
Repairs
Staff
Views
Stored Procedures
Functions
world
Query 1 Customer Daily_Cost Daily_Overdue_Fee Instrument Rental Repairs Staff
Limit to 5000 rows
1 SELECT * FROM Uptown.Rental;
Result Grid Filter Rows: Edit: Export/Import:
Rental_ID Rent_Date Return_Date Due_Date Days_Overdue Rental_Cost Staff_Staff_... Customer_Cust
1 2024-12-04 00:00:00 2024-12-19 00:00:00 2024-12-20 00:00:00 0 525.00 1 1
2 2015-01-05 00:00:00 2015-01-12 00:00:00 2015-01-10 00:00:00 2 220.00 1 2
3 2015-01-05 00:00:00 2015-01-12 00:00:00 2015-01-10 00:00:00 2 235.00 1 2
4 2015-01-05 00:00:00 2015-01-12 00:00:00 2015-01-12 00:00:00 0 245.00 7 5
5 2024-12-20 00:00:00 2024-12-29 00:00:00 2024-12-29 00:00:00 0 315.00 1 6
6 2015-01-13 00:00:00 2015-01-22 00:00:00 2015-01-22 00:00:00 0 315.00 7 7
7 2015-01-25 00:00:00 2015-01-30 00:00:00 2015-01-30 00:00:00 0 175.00 7 5
Rental 1
Object Info Session
Table: Customer
Columns:
Customer_ID int AI PK
Cust_Address varchar(60)
Cust_Name varchar(45)
Cust_Email varchar(45)
Action Output
Time Action Response Duration / Fetch Time
16 16:48:11 SELECT * FROM Uptown.Staff LIMIT 0, 5000 7 row(s) returned 0.00076 sec / 0.0000
Query Completed


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Fri Mar 28 4:48 PM
MySQL Workbench
Local instance 3306
Administration Schemas
SCHEMAS
CustomerOrder
mydb
ProjectExercise
sakila
SALE
sys
Uptown
Tables
Customer
Daily_Cost
Daily_Overdue_Fee
Instrument
Rental
Repairs
Staff
Views
Stored Procedures
Functions
world
Query 1 Customer Daily_Cost Daily_Overdue_Fee Instrument Rental Repairs Staff
Limit to 5000 rows
1 SELECT * FROM Uptown.Instrument;
Result Grid Filter Rows: Edit: Export/Import:
Serial_Num Inst_Category Daily_Cost_Inst_Type Daily_Overdue_Fee_Rent_Tier
12878 Woodwind Flute Basic
12904 Brass Trombone Premium
23043 Percussion Drums Premium
73766 Percussion Xylophone Premium
76657 Acoustic Bass Guitar Premium
76887 Brass Trumpet Premium
98223 Woodwind Flute Basic
Instrument 1
Object Info Session
Table: Customer
Columns:
Customer_ID int AI PK
Cust_Address varchar(60)
Cust_Name varchar(45)
Cust_Email varchar(45)
Action Output
Time Action Response Duration / Fetch Time
16 16:48:11 SELECT * FROM Uptown.Staff LIMIT 0, 5000 7 row(s) returned 0.00076 sec / 0.0000
Query Completed


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Fri Mar 28 4:48 PM
MySQL Workbench
Local instance 3306
Administration Schemas
SCHEMAS
CustomerOrder
mydb
ProjectExercise
sakila
SALE
sys
Uptown
Tables
Customer
Daily_Cost
Daily_Overdue_Fee
Instrument
Rental
Repairs
Staff
Views
Stored Procedures
Functions
world
Query 1 Customer Daily_Cost Daily_Overdue_Fee Instrument Rental Repairs Staff
Limit to 5000 rows
1 SELECT * FROM Uptown.Daily_Overdue_Fee;
Result Grid Filter Rows: Edit: Export/Import:
Rent_Tier Daily_Overdue_Fee
Basic 0.00
Premium 5.00
Daily_Overdue_Fee 1
Action Output
Time Action Response Duration / Fetch Time
16 16:48:11 SELECT * FROM Uptown.Staff LIMIT 0, 5000 7 row(s) returned 0.00076 sec / 0.0000
Object Info Session
Table: Customer
Columns:
Customer_ID int AI PK
Cust_Address varchar(60)
Cust_Name varchar(45)
Cust_Email varchar(45)
Query Completed

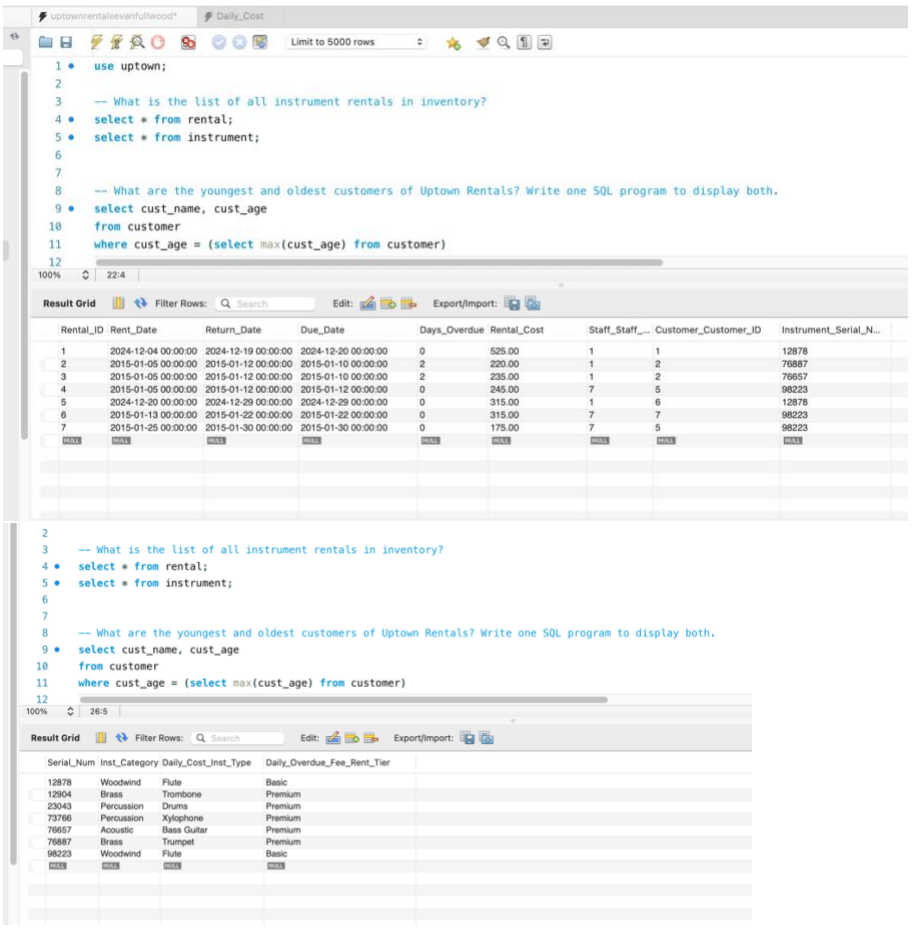

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Fri Mar 28 4:48 PM
MySQL Workbench
Local instance 3306
Administration Schemas
SCHEMAS
CustomerOrder
mydb
ProjectExercise
sakila
SALE
sys
Uptown
Tables
Customer
Daily_Cost
Daily_Overdue_Fee
Instrument
Rental
Repairs
Staff
Views
Stored Procedures
Functions
world
Query 1 Customer Daily_Cost Daily_Overdue_Fee Instrument Rental Repairs Staff
Limit to 5000 rows
1 SELECT * FROM Uptown.Daily_Cost;
Result Grid Filter Rows: Edit: Export/Import:
Inst_Type Daily_Rent_Fee
Bass Guitar 43.00
Drums 50.00
Flute 35.00
Trombone 45.00
Trumpet 40.00
Tuba 40.00
Xylophone 40.00
Daily_Cost 1
Action Output
Time Action Response Duration / Fetch Time
16 16:48:11 SELECT * FROM Uptown.Staff LIMIT 0, 5000 7 row(s) returned 0.00076 sec / 0.0000
Object Info Session
Table: Customer
Columns:
Customer_ID int AI PK
Cust_Address varchar(60)
Cust_Name varchar(45)
Cust_Email varchar(45)
Query Completed




**Create SQL programs to answer questions of the data:**

5. Open a new query tab and save it with the name, **uptownRentalsxy.sql** (where x is your first name and y is your last name).
6. Add a comment to the top of your script with your name, the purpose of the script, and the current date.
7. Type the "*use*" statement for uptown and run it: **use uptown;**
8. Create and run the following SQL programs, each with a preceding comment stating the query in English. Be sure to validate your results using an alternate method. **Screenshot your code and answer in this document for each of the following.**

    a. What is the list of all instrument rentals in inventory? (Show the list displayed in Figure 1, along with any other rentals in your database.)



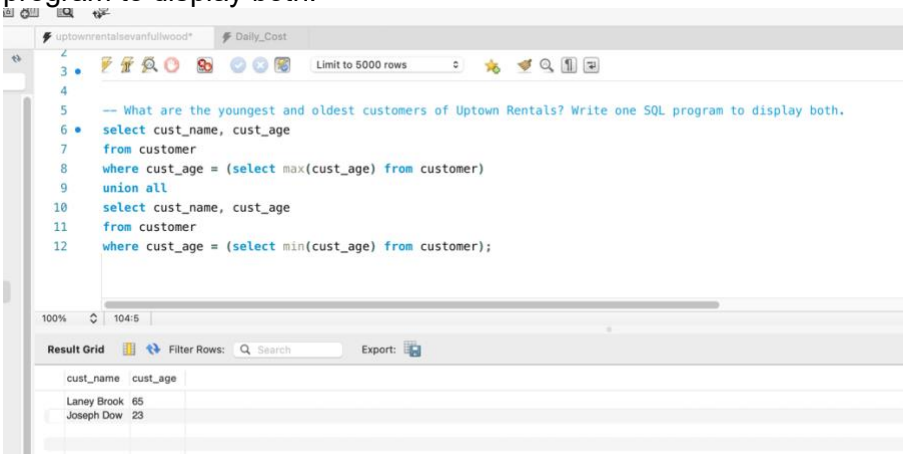
```
use uptown;

-- What is the list of all instrument rentals in inventory?
select * from rental;
select * from instrument;


-- What are the youngest and oldest customers of Uptown Rentals? Write one SQL program to display both.
select cust_name, cust_age
from customer
where cust_age = (select max(cust_age) from customer)
```

| Rental_ID | Rent_Date | Return_Date | Due_Date | Days_Overdue | Rental_Cost | Staff_Staff_... | Customer_Customer_ID | Instrument_Serial_N... |
|---|---|---|---|---|---|---|---|---|
| 1 | 2024-12-04 00:00:00 | 2024-12-19 00:00:00 | 2024-12-20 00:00:00 | 0 | 525.00 | 1 | 1 | 12878 |
| 2 | 2015-01-05 00:00:00 | 2015-01-12 00:00:00 | 2015-01-10 00:00:00 | 2 | 220.00 | 1 | 2 | 76887 |
| 3 | 2015-01-05 00:00:00 | 2015-01-12 00:00:00 | 2015-01-10 00:00:00 | 2 | 235.00 | 1 | 2 | 76657 |
| 4 | 2015-01-05 00:00:00 | 2015-01-12 00:00:00 | 2015-01-12 00:00:00 | 0 | 245.00 | 7 | 5 | 98223 |
| 5 | 2024-12-20 00:00:00 | 2024-12-29 00:00:00 | 2024-12-29 00:00:00 | 0 | 315.00 | 1 | 6 | 12878 |
| 6 | 2015-01-13 00:00:00 | 2015-01-22 00:00:00 | 2015-01-22 00:00:00 | 0 | 315.00 | 7 | 7 | 98223 |
| 7 | 2015-01-25 00:00:00 | 2015-01-30 00:00:00 | 2015-01-30 00:00:00 | 0 | 175.00 | 7 | 5 | 98223 |

```
-- What is the list of all instrument rentals in inventory?
select * from rental;
select * from instrument;


-- What are the youngest and oldest customers of Uptown Rentals? Write one SQL program to display both.
select cust_name, cust_age
from customer
where cust_age = (select max(cust_age) from customer)
```

| Serial_Num | Inst_Category | Daily_Cost_Inst_Type | Daily_Overdue_Fee_Rent_Tier |
|---|---|---|---|
| 12878 | Woodwind | Flute | Basic |
| 12904 | Brass | Trombone | Premium |
| 23043 | Percussion | Drums | Premium |
| 73766 | Percussion | Xylophone | Premium |
| 76657 | Acoustic | Bass Guitar | Premium |
| 76887 | Brass | Trumpet | Premium |
| 98223 | Woodwind | Flute | Basic |

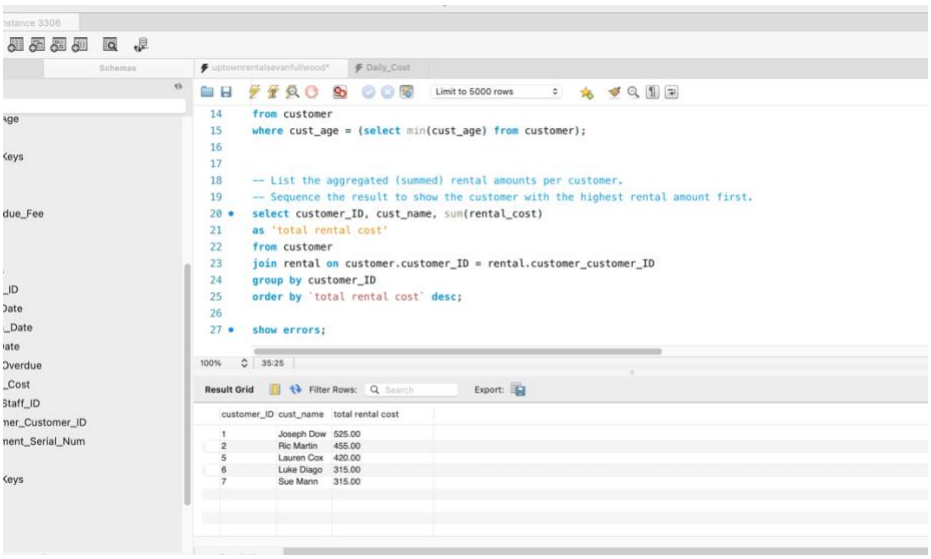
b. What are the youngest and oldest customers of Uptown Rentals? Write one SQL program to display both.



```
-- What are the youngest and oldest customers of Uptown Rentals? Write one SQL program to display both.
select cust_name, cust_age
from customer
where cust_age = (select max(cust_age) from customer)
union all
select cust_name, cust_age
from customer
where cust_age = (select min(cust_age) from customer);
```

| cust_name | cust_age |
|---|---|
| Laney Brook | 65 |
| Joseph Dow | 23 |

c. List the aggregated (summed) rental amounts per customer. Sequence the result to show the customer with the highest rental amount first.

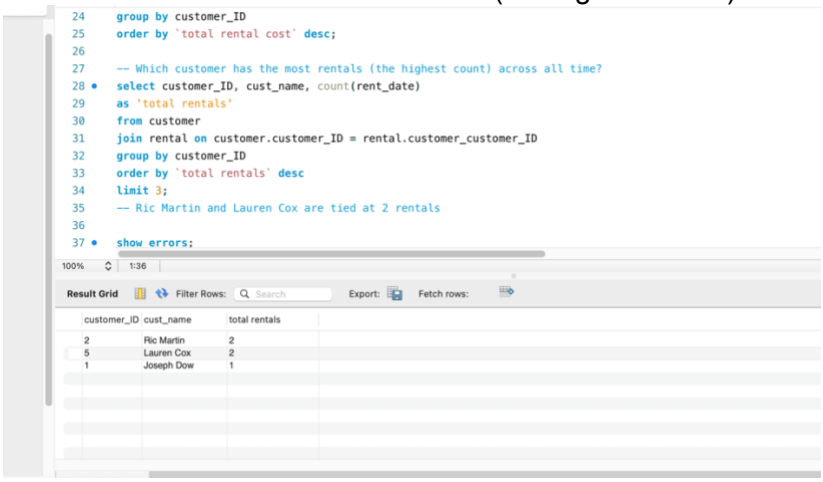


```
from customer
where cust_age = (select min(cust_age) from customer);


-- List the aggregated (summed) rental amounts per customer.
-- Sequence the result to show the customer with the highest rental amount first.
select customer_ID, cust_name, sum(rental_cost)
as 'total rental cost'
from customer
join rental on customer.customer_ID = rental.customer_customer_ID
group by customer_ID
order by `total rental cost` desc;

show errors;
```

Result Grid

| customer_ID | cust_name | total rental cost |
|---|---|---|
| 1 | Joseph Dow | 525.00 |
| 2 | Ric Martin | 455.00 |
| 5 | Lauren Cox | 420.00 |
| 6 | Luke Diago | 315.00 |
| 7 | Sue Mann | 315.00 |

d. Which customer has the most rentals (the highest count) across all time?

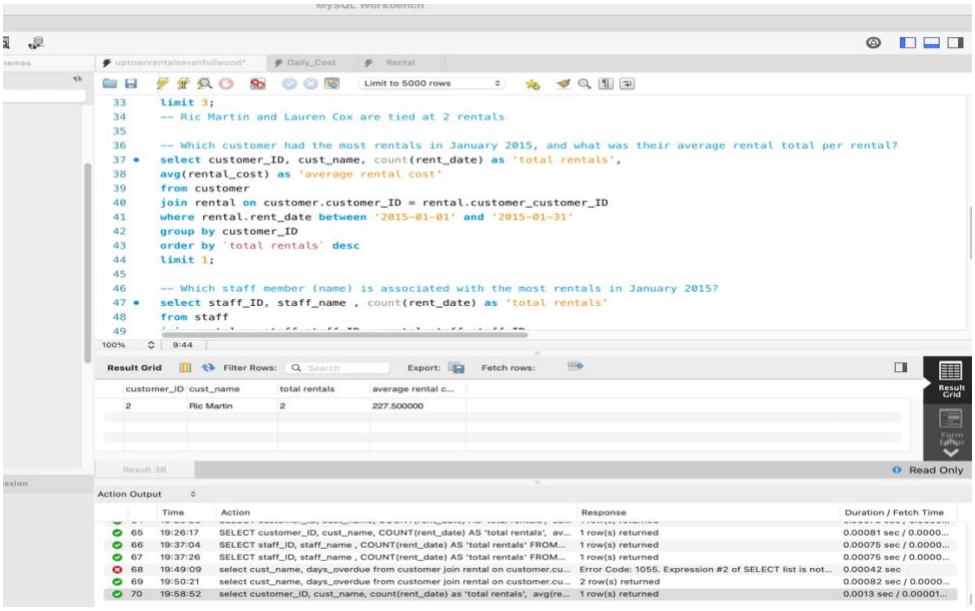


```
group by customer_ID
order by `total rental cost` desc;

-- Which customer has the most rentals (the highest count) across all time?
select customer_ID, cust_name, count(rent_date)
as 'total rentals'
from customer
join rental on customer.customer_ID = rental.customer_customer_ID
group by customer_ID
order by `total rentals` desc
limit 3;
-- Ric Martin and Lauren Cox are tied at 2 rentals

show errors;
```

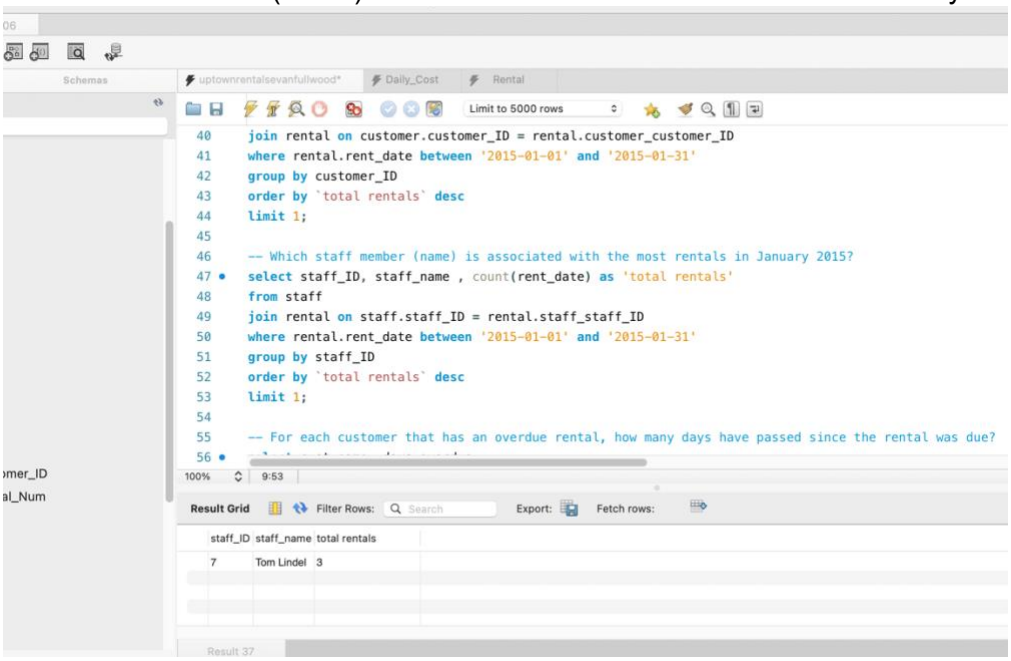
Result Grid

| customer_ID | cust_name | total rentals |
|---|---|---|
| 2 | Ric Martin | 2 |
| 5 | Lauren Cox | 2 |
| 1 | Joseph Dow | 1 |

e. Which customer had the most rentals in January 2015, and what was their average rental total per rental?




f. Which staff member (name) is associated with the most rentals in January 2015?




g. For each customer that has an overdue rental, how many days have passed since the rental was due?

```
52    order by  total rentals  desc
53    limit 1;
54
55    -- For each customer that has an overdue rental, how many days have passed since the rental was due?
56 •  select cust_name, days_overdue
57    from customer
58    join rental on customer.customer_ID = rental.customer_customer_ID
59    where rental.days_overdue > 0;
60    -- Ric Martin has had two instances of days overdue, each for two days.
61
62
63
64
65
```

100% 1:61

Result Grid | Filter Rows: Search | Export:

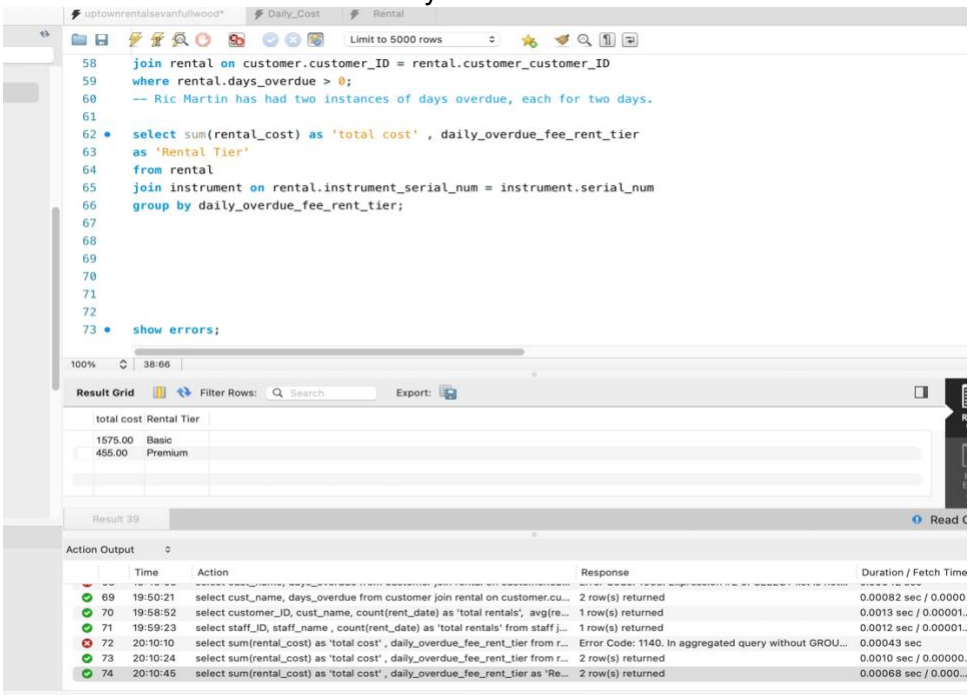
| cust_name | days_overdue |
|---|---|
| Ric Martin | 2 |
| Ric Martin | 2 |

Result 35

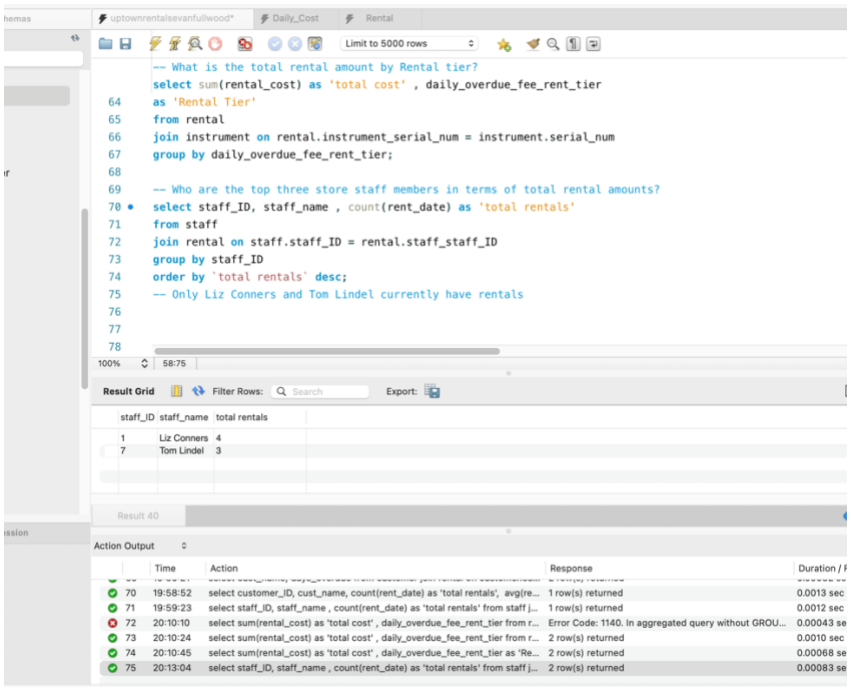
Action Output

| | Time | Action | Response | Duration / Fetch Ti |
|---|---|---|---|---|
| 64 | 19:25:26 | SELECT customer_ID, cust_name, COUNT(rent_date) AS 'total rentals', su... | 1 row(s) returned | 0.00079 sec / 0.00 |
| 65 | 19:26:17 | SELECT customer_ID, cust_name, COUNT(rent_date) AS 'total rentals', av... | 1 row(s) returned | 0.00081 sec / 0.00 |
| 66 | 19:37:04 | SELECT staff_ID, staff_name , COUNT(rent_date) AS 'total rentals' FROM... | 1 row(s) returned | 0.00075 sec / 0.00 |
| 67 | 19:37:26 | SELECT staff_ID, staff_name , COUNT(rent_date) AS 'total rentals' FROM... | 1 row(s) returned | 0.00075 sec / 0.00 |
| 68 | 19:49:09 | select cust_name, days_overdue from customer join rental on customer.cu... | Error Code: 1055. Expression #2 of SELECT list is not... | 0.00042 sec |
| 69 | 19:50:21 | select cust_name, days_overdue from customer join rental on customer.cu... | 2 row(s) returned | 0.00082 sec / 0.00 |

h. What is the total rental amount by Rental tier?



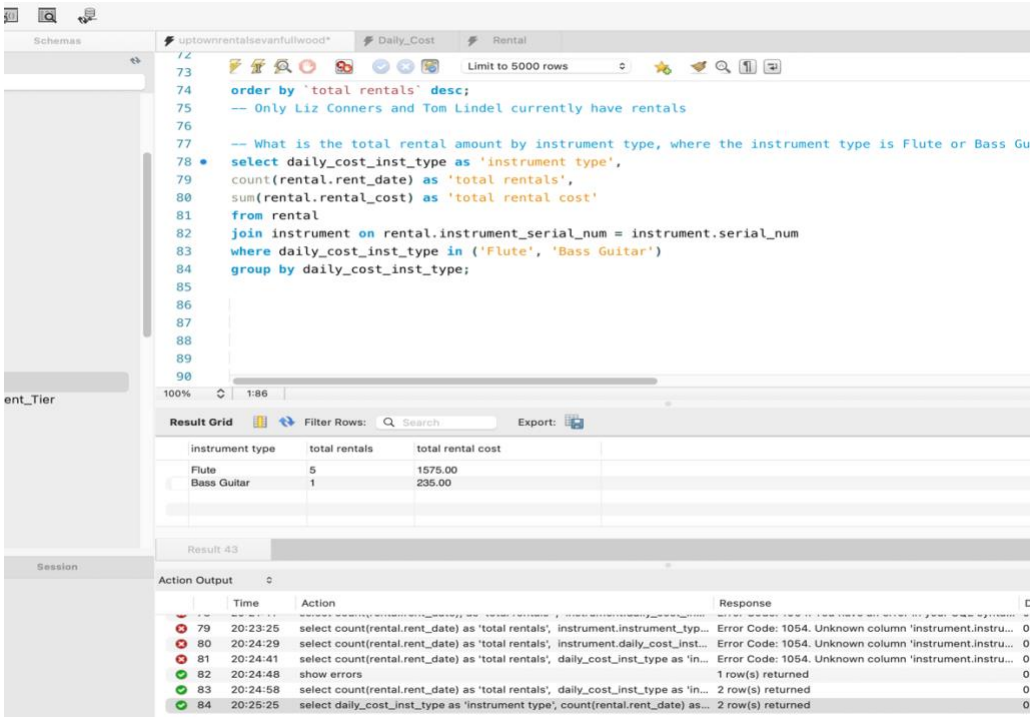

i. Who are the top three store staff members in terms of total rental amounts?



```
-- What is the total rental amount by Rental tier?
select sum(rental_cost) as 'total cost' , daily_overdue_fee_rent_tier
as 'Rental Tier'
from rental
join instrument on rental.instrument_serial_num = instrument.serial_num
group by daily_overdue_fee_rent_tier;

-- Who are the top three store staff members in terms of total rental amounts?
select staff_ID, staff_name , count(rent_date) as 'total rentals'
from staff
join rental on staff.staff_ID = rental.staff_staff_ID
group by staff_ID
order by `total rentals` desc;
-- Only Liz Conners and Tom Lindel currently have rentals
```

| staff_ID | staff_name | total rentals |
|---|---|---|
| 1 | Liz Conners | 4 |
| 7 | Tom Lindel | 3 |

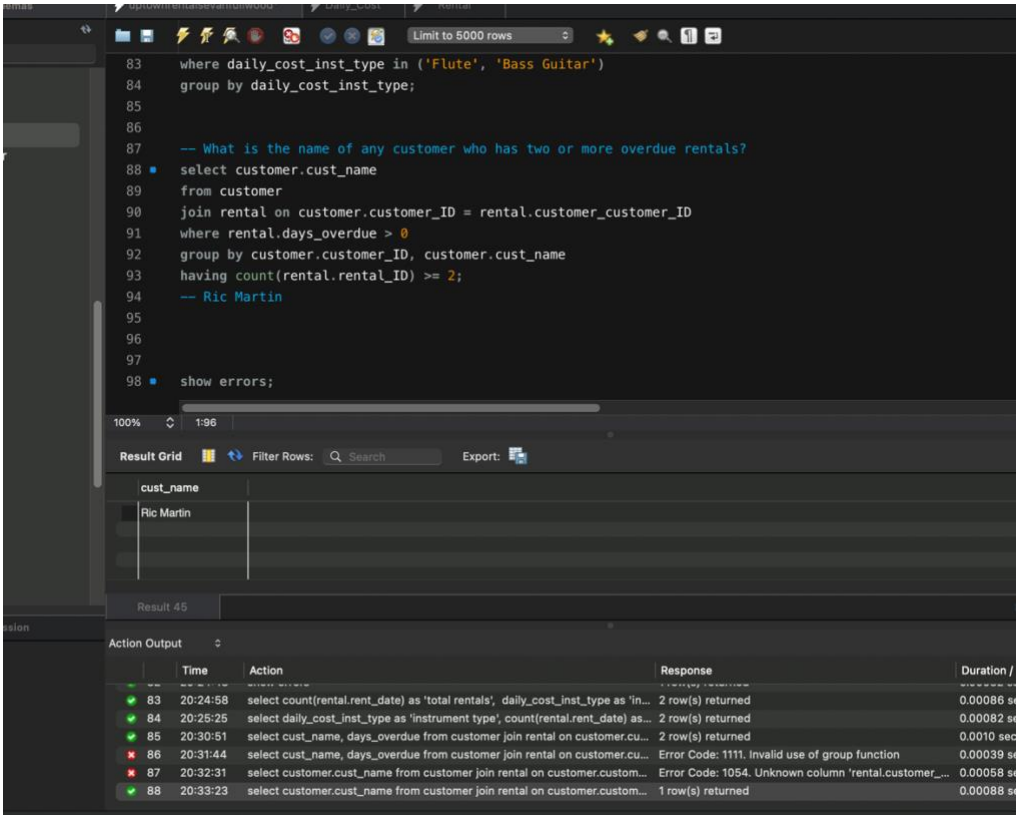
j. What is the total rental amount by instrument type, where the instrument type is *Flute* or *Bass Guitar*?



```
order by `total rentals` desc;
-- Only Liz Conners and Tom Lindel currently have rentals

-- What is the total rental amount by instrument type, where the instrument type is Flute or Bass Gu
select daily_cost_inst_type as 'instrument type',
count(rental.rent_date) as 'total rentals',
sum(rental.rental_cost) as 'total rental cost'
from rental
join instrument on rental.instrument_serial_num = instrument.serial_num
where daily_cost_inst_type in ('Flute', 'Bass Guitar')
group by daily_cost_inst_type;
```

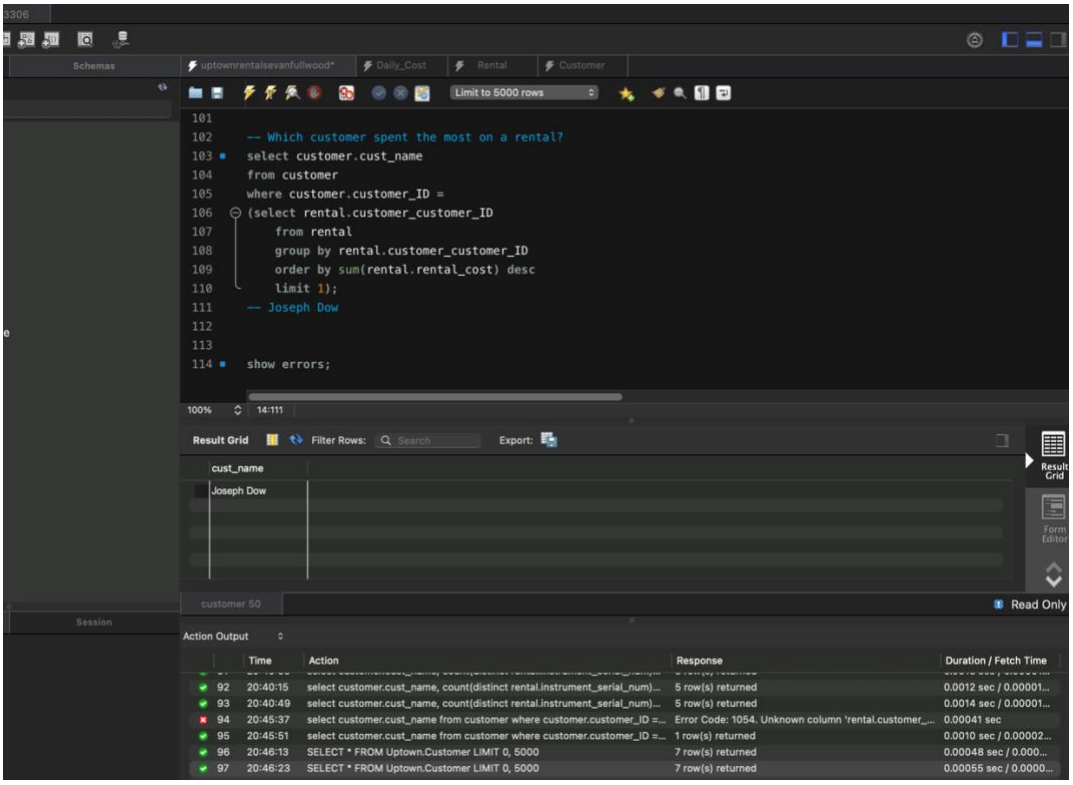
| instrument type | total rentals | total rental cost |
|---|---|---|
| Flute | 5 | 1575.00 |
| Bass Guitar | 1 | 235.00 |

k. What is the name of any customer who has two or more overdue rentals?

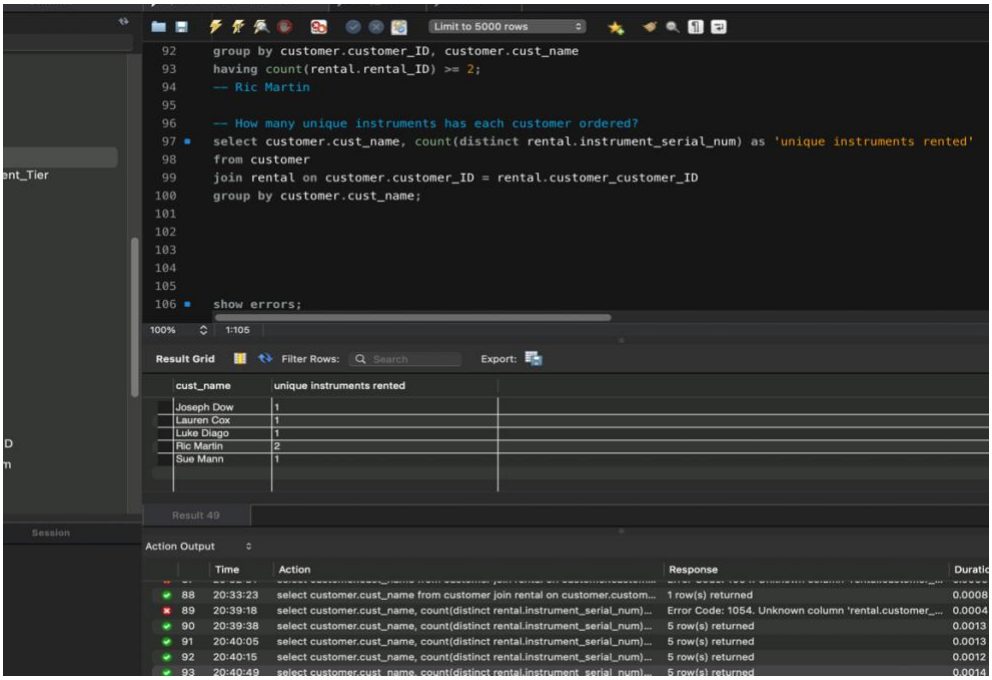



l. List all of the instruments in inventory in 2015 that were damaged upon return or needed maintenance. Include the employee that handled the rental, the repair cost, and the maintenance date.

m. Create a query of your choice that includes a subquery.




n. Add another <u>meaningful</u> query of your choice. For example, you could create a query that answers the following question: What is the name of any customer who has rented 3 or more Woodwind instruments?




**Submit your project:** Two files are required:

1. This exact .docx file with all query answers and screenshots;
2. Screenshots of the data in each table; and
3. The .sql script.