Polytechnic University of Puerto Rico

San Juan, Puerto Rico

Electrical and Computer Engineering Department

CECS 4202-81 Database Systems

FA 2023

Assignment 1

Due Date: September 17, 2023

Total Points: 100

Student Name: Carlos Surillo

Student ID: 138098

Prof. Dr. Alfredo Cruz

Part 1: Entity Relationship Diagrams (ERD) (40 points)

Chapter 1

Checkpoint 1.2

1. **What are the three main types of data models?**
   → The Hierarchical Model, The Network Model and The Relational Model

Checkpoint 1.3

1. **What are functional dependencies? Give examples.**
   → A functional relationship is a relationship of one attribute or field in a record to another. An example would be that Social Security Number (SSN) defines a name. If I have a database with SSNs and names, and if I know someone's SSN, then I can find their name.

2. **What does the augmentative rule state? Give examples**
   → It states that If X→Y, then XZ→Y. For example, if Name → City and we add a column named eye color then Name + eye color → City. Name plus more information will always identify the unique City for that individual.

Chapter 2

Checkpoint 2.3

1. **How do you map multi-valued attributes?**
   → Form a separate table for the multi-valued attribute. Record a row for each value of the multi-valued attribute, together with the key from the original table. The key of the new table will be the concatenation of the multi-valued attribute plus the key of the owner entity. Remove the multi-valued attribute from the original table.

2. **How do you map composite attributes?**
   → Map entities to a table by forming columns from the elementary parts of the composite attributes.

3. **What is a unique identifier? Is it a candidate key? Is it the primary key? Discuss.**

→ A unique identifier can be an attribute or a combination of attributes. If an attribute can be thought of as a unique identifier for an entity, it is called a candidate key. When a candidate key is chosen to be the unique identifier, it becomes the primary key for the entity.

Chapter 3

Checkpoint 3.1

1. **Can the nature of an entity change over time? Explain.**

   → Yes, the nature of an entity can change over time because as time changes the nature of the entity should change to satisfy the relationship between other tables and entities.

2. **What is a relationship?**

   → A relationship denotes the connection between entities.

3. **What are the differences between an entity and a relationship?**

   → An entity's attributes express qualities in terms of properties or characteristics. Relationships express associations with other entities.

Chapter 4

Checkpoint 4.1

1. **What are structural constraints?**

   → Structural constraints are information about how two (or more) entities are related to one another. There are two types of structural constraints: cardinality and participation.

2. **What kind of information does the cardinality ratio give us?**

   → Cardinality is a rough measure of the number of entities (one or more) that will be related to another entity (or entities)

3. **In how many different ways can two entities be involved in a cardinality relationship? Give examples.**

   → There are four ways in which the entities can be "numerically involved" in a relationship: one-to-one (1:1), many-to-one (M:1), one-to-many (1:M), and many-to-many (M: N).

Checkpoint 4.3

1. **Give an example of a 1(full):1 relationship? Does such a relationship always have to be mandatory? Explain with examples.**
   - → An example of a 1:1 relationship would be a student and their student Id; each student can only have one Id number and each Id number can only be assigned to one student. This relationship is mandatory because each instance of the entity must participate in the relationship

2. **Give an example of a 1(partial):1 relationship? Does such a relationship always have to be optional? Explain with examples.**
   - → An example would be students and automobiles in a database. The automobile is driven by one student and one student drives one automobile, making it a 1:1 relationship. For every automobile one student will be driving it, but not every student will drive an automobile (maybe they don't own one). The sense of partial, optional participation is that there could be students who do not have a relationship to an automobile.

3. **Give an example of an M(full): N relationship? Would such a relationship always be optional or mandatory? Explain with examples.**
   - → An example would be students and teachers, as a student in school is taught by many teachers and teachers teach many students. The relationship would be optional since the students may choose not to be part of the relationship at any given time.

Part 2: SQL

A. Review Questions

Chapter 2

1. **What is an entity?**
   - → An entity is like a noun; it is a person, place, thing, or even
2. **What is an attribute?**
   - → An attribute is a property of an entity
3. **What is a relationship? What is a one-to-many relationship?**
   - → A relationship is the association between entities. The relationship between sales reps and customers is an example of a one-to-many relationship because one sales

rep is associated with many customers, but each customer is associated with only one sales rep.

4. **What is a repeating group?**
   → An entry that is repeated.
5. **What is a relation?**
   → A relation is a two-dimensional table in which the entries in the table are single-valued, each column has a distinct name, all values in the column match this name, the order of the rows and columns is immaterial, and each row contains unique values.
6. **What is a relational database?**
   → A relational database is a collection of relations.
7. **Describe the shorthand representation of the structure of a relational database. Why is it important to be able to represent the structure of a database in a shorthand fashion?**
   → For each table, you write the name of the table and then within parentheses list all of the columns in the table. In this representation, each table appears on its own line. It is important to be able to represent the structure in a shorthand fashion because it makes it easier to identify patterns, relations and repeating groups in the database.
8. **How do you qualify the name of a field, and when do you need to do this?**
   → You qualify the name of a field by writing both the table name and the column name, separated by a period. It is always acceptable to qualify field names, even when there is no confusion, but if confusion might arise, it is essential.
9. **What does it mean for a column to be functionally dependent on another column?**
   → In a relational database, a column (X) is functionally dependent on another column (Y), if at any point in time a value for Y determines a single value for X.
10. **What is a primary key? Why is a primary key required for proper database design?**
    → The primary key is the unique identifier for a table. It is required because it allows us to properly identify unique records in a column.
11. **A database at a college must support the following requirements:**
    a. **For a department, store its number and name.**
       □ Create Table DEPT (DEPT_NUM DECIMAL (2,0), DEPT_NAME CHAR (15));
    b. **For an advisor, store his or her number, last name, first name, and the department number to which the advisor is assigned.**
       □ Create Table ADV(ADV_NUM DECIMAL (2,0), FIRST_NAME CHAR (15), LAST_NAME CHAR(15), DEPT_NUM DECIMAL(2,0));
    c. **For a course, store its code and description (for example, DBA210, SQL Programming).**
       □ Create Table COURSE (COURSE_CODE CHAR(10), DESCRIPTION CHAR (60));
    d. **For a student, store his or her number, first name, and last name. For each course the student takes, store the course code, course description, and grade earned. Also, store the number and name of the student's advisor. Assume that**

an advisor might advise any number of students but that each student has just one advisor.

- Create Table STUDENT(STU_NUM DECIMAL (6,0), FIRST_NAME CHAR (15), LAST_NAME CHAR(15), COURSE_CODE CHAR(10), DESCRIPTION CHAR (60), GRADE CHAR(2));

Design the database for the preceding set of requirements. Use your own experience as a student to determine any functional dependencies. List the tables, columns, and relationships. In addition, represent your design with an E-R diagram.

→ Functional Dependencies:

- DEPT: DEPT_NUM → DEPT_NAME
- ADVISOR: ADV_NUM → FIRST_NAME, LAST_NAME, DEPT_NUM
- COURSE: COURSE_CODE → DESCRIPTION
- STUDENT: STU_NUM → FIRST_NAME, LAST_NAME, COURSE_CODE

→ Relationships:

- The DEPT and ADVISOR tables are related by the DEPT_NUM column. This is a one to many relationship because of the possibility of there being many advisors per department.
- The COURSE and STUDENT table are be related by the COURSE_CODE column. This is also a one to many relationship because of the possibility of there being many students per course.

12. Define first normal form. What types of problems might you encounter using tables that are not in first normal form?

→ A table is in first normal form when it does not contain a repeating group**.** When you convert an unnormalized table to a table in first normal form, the primary key of the table in first normal form is usually the primary key of the unnormalized table concatenated with the key for the repeating group.

13. Define second normal form. What types of problems might you encounter using tables that are not in second normal form?

→ Second normal form represents an improvement over first normal form because it eliminates update anomalies. A table is in second normal form when it is in first normal form and a column that is not part of the primary key is dependent on only a portion of the primary key.

14. Define third normal form. What types of problems might you encounter using tables that are not in third normal form?

→ A table is in third normal form when it is in second normal form and the only determinants it contains are candidate keys.

15. Using the functional dependencies that you determined in Question 11, convert the following table to an equivalent collection of tables that are in third normal form.

→ STUDENT (STUDENT _ NUM, LAST _ NAME, FIRST _ NAME)

→ ADVISOR (ADVISOR _ NUM, LAST _ NAME, FIRST _ NAME)
→ COURSE (COURSE _ CODE, DESCRIPTION, GRADE)

Chapter 3

1. **How do you create a table using SQL?**
   → You use the CREATE TABLE command to describe the layout of a table. The word TABLE is followed by the name of the table to be created and then by the names and data types of the columns that the table contains.
2. **How do you delete a table using SQL?**
   → You can delete the entire table using the DROP TABLE command followed by the name of the table you want to delete and a semicolon.
3. **What are common data types used to define columns using SQL?**
   → CHAR(n), VARCHAR(n), DATE, DECIMAL (p, q), INT, SMALLINT.
4. **Identify the best data type to use to store the following data in Oracle, in SQL Server, and in MySQL:**
   a. The month, day, and year that an employee was hired
   → DATE
   b. An employee's Social Security number
   → CHAR(n) or VARCHAR(n)
   c. The department in which an employee works
   → CHAR(n) or VARCHAR(n)
   d. An employee's hourly pay rate
   → DECIMAL (p, q)
5. **Identify the following column names as valid or invalid in MySQL:**
   a. COMMISSIONRATE
   → Valid
   b. POSTAL_CODE_5CHAR
   → Valid
   c. SHIP TO ADDRESS
   → Invalid
   d. INVOICE-NUMBER
   → Invalid
6. **What is a null value? How do you use SQL to identify columns that cannot accept null values?**
   → A null value is a special value used to represent cases in which an actual value is unknown, unavailable, or not applicable. When creating a table, you can specify whether to allow nulls in the individual columns, you use the NOT NULL clause in a CREATE TABLE command to indicate columns that cannot contain null values. The default is to allow nulls; columns for which you do not specify NOT NULL can accept null values.

7. **Which SQL command do you use to add a row to a table?**
    - → The INSERT command adds rows to a table. You type INSERT INTO followed by the name of the table into which you are adding data. Then you type the word VALUES followed by the specific values to be inserted in parentheses.
8. **Which SQL command do you use to view the data in a table?**
    - → You can use a simple version of the SELECT command to display all the rows and columns in a table by typing the word SELECT, followed by an asterisk (*), followed by the key- word FROM and the name of the table containing the data you want to view
9. **Which SQL command do you use to change the value in a column in a table?**
    - → You can use the UPDATE command followed by the table name. Then you use SET command followed by the column name equal to the value you want updated. Lastly you use WHERE command to implicate a condition for the update to execute.
10. **Which SQL command do you use to delete rows from a table?**
    - → When you need to delete a row from a table, you can use the DELETE command. Then you use FROM command to choose table, then set the condition for the deletion to execute using the WHERE command.
11. **How do you display the columns in a table and their characteristics in MySQL?**
    - → You can use the DESCRIBE command followed by table name to list all the columns in a table and their properties

Chapter 4

1. **Describe the basic form of the SQL SELECT command.**
    - → The basic form of the SELECT command is SELECT-FROM-WHERE. After you type the word SELECT, you list the columns that you want to include in the query results (SELECT clause). Next, you type the word FROM followed by the name of the table that contains the data you need to query (FROM clause). Finally, after the word WHERE, you list any conditions that apply to the data you want to retrieve (WHERE clause).
2. **How do you form a simple condition?**
    - → A simple condition has the form column name, comparison operator, and then either another column name or a value.
3. **How do you form a compound condition?**
    - → You form a compound condition by connecting two or more simple conditions with the AND, OR, and NOT operators.
4. **In SQL, which operator do you use to determine whether a value is between two other values without using an AND condition?**

→ An alternative approach uses the BETWEEN operator which lets you specify a range of values in a condition. The BETWEEN operator is inclusive, meaning that the query selects a value equal to either value in the condition and in the range of the values.

5. **How do you use a computed column in SQL? How do you name the computed column?**

   → You can perform computations using SQL queries. A computed column does not exist in the database but can be computed using data in the existing columns. Computations can involve any arithmetic operator (+,-, *, /). You also can assign a name, or alias, to a computed column by following the compu- tation with the word AS and the desired name.

6. **In which clause would you use a wildcard in a condition?**

   → The LIKE operator uses one or more wildcard characters to test for a pattern match.

7. **What wildcards are available in MySQL, and what do they represent?**

   → The percent sign (%) is used as a wildcard to represent any collection of characters, and the underscore(_) to represent any individual character.

8. **How do you determine whether a column contains one of a particular set of values without using an AND condition?**

   → An IN clause, which consists of the IN operator followed by a collection of values, provides a concise way of phrasing certain conditions, the IN clause contains a collection of values. The condition is true for those rows in which the value in the column is in this collection.

9. **How do you sort data?**

   → You use the ORDER BY clause to list data in a specific order. The column on which to sort data is called a sort key or simply a key. If you do not specify a sort order, the default is ascending

10. **How do you sort data on more than one sort key? What is the more important key called? What is the less important key called?**

    → When you need to sort data on two columns, the more important column is called the major sort key, or the primary sort key, and the less important column is called the minor sort key, or the secondary sort key. To sort on multiple keys, you list the keys in order of importance in the ORDER BY clause.

11. **How do you sort data in descending order?**

    → To sort in descending order, you fol- low the name of the sort key with the DESC operator.

12. **What are the SQL aggregate functions?**

    → They are special functions used to calculate sums, averages, counts, maximum values, and minimum values. These functions apply to groups of rows. They could apply to all the rows in a table or to those rows satisfying some particular condition.

13. **How do you avoid including duplicate values in a query’s results?**

→ The DISTINCT operator is useful when used in conjunction with the COUNT function because it eliminates duplicate values in the query results

14. **What is a subquery?**

    → When you have nested queries, the inner query is called a subquery. The subquery is evaluated first. After the subquery has been evaluated, the outer query can use the results of the subquery to find its results.

15. **How do you group data in an SQL query?**

    → The GROUP BY clause lets you group data on a particular column, and then calculate statistics, when desired. The GROUP BY clause does not sort the data in a particular order; you must use the ORDER BY clause to sort data.

16. **When grouping data in a query, how do you restrict the output to only those groups satisfying some condition?**

    → The HAVING clause is used to restrict the groups that are included. This restriction does not apply to individual rows but rather to groups. the HAVING clause does for groups what the WHERE clause does for rows. The HAVING clause limits the groups that are included in the results.

17. **How do you find rows in which a particular column contains a null value?**

    → The correct format uses the IS NULL operator. To select a column whose value is not null, use the IS NOT NULL operator.

B. Using MySQL

I will show how I have configured the StayWell Student Accommodation Database using MySQL:

```
CREATE DATABASE StayWell_Students;
```

```
CREATE TABLE OFFICE(OFFICE_NUM DECIMAL(2,0) NOT NULL PRIMARY KEY,
OFFICE_NAME char(25), ADDRESS char(25), AREA char(25), CITY char(25) ,STATE
CHAR(2), ZIPCODE char(5));
```

```
INSERT INTO OFFICE VALUES('1','StayWell-Columbia City', '1135 N. Wells Avenue',
'Columbia City', 'Seattle', 'WA', '98118');
```

```
INSERT INTO OFFICE VALUES('2', 'StayWell-Georgetown',  '986 S. Madison Rd',
'Georgetown', 'Seattle','WA', '98108');

CREATE TABLE OWNER(OWNER_NUM CHAR(5) NOT NULL PRIMARY KEY,
LAST_NAME CHAR(25), FIRST_NAME CHAR(25), ADDRESS CHAR(25), CITY
VARCHAR(25), STATE CHAR(2), ZIPCODE CHAR(5));

INSERT INTO OWNER VALUES('MO100','Moore','Elle-May','8006 W. Newport Ave.','Reno',

'NV','89508');

INSERT INTO OWNER VALUES('PA101','Patel','Makesh','7337 Sheffield St.','Seattle',

'WA','98119');

INSERT INTO OWNER VALUES('AK102','Aksoy','Ceyda','411 Griffin Rd.','Seattle',

'WA','98131');

INSERT INTO OWNER VALUES('CO103','Cole','Meerab','9486 Circle Ave.','Olympia',

'WA','98506');

INSERT INTO OWNER VALUES('KO104','Kowalczyk','Jakub','7431 S. Bishop
St.','Bellingham',  'WA','98226');

INSERT INTO OWNER VALUES('SI105','Sims','Haydon','527 Primrose Rd.','Portland',

'OR','97203');

INSERT INTO OWNER VALUES('BU106','Burke','Ernest','613 Old Pleasant St.','Twin Falls',

'ID','83303');

INSERT INTO OWNER VALUES('RE107','Redman','Seth','7681 Fordham St.','Seattle',

'WA','98119');
```

```
INSERT INTO OWNER VALUES('LO108','Lopez','Janine','9856 Pumpkin Hill Ln.','Everett',
'WA','98213');

INSERT INTO OWNER VALUES('BI109','Bianchi','Nicole','7990 Willow Dr.','New York',
'NY','10005');

INSERT INTO OWNER VALUES('JO110','Jones','Ammarah','730 Military Ave.','Seattle',
'WA','98126');

CREATE TABLE PROPERTY(PROPERTY_ID DECIMAL(2,0) NOT NULL PRIMARY
KEY, OFFICE_NUM DECIMAL(2,0), ADDRESS CHAR(25), SQR_FT
DECIMAL(5,0),BDRMS DECIMAL(2,0),FLOORS DECIMAL(2,0), MONTHLY_RENT
DECIMAL(6,2), OWNER_NUM CHAR(5));

INSERT INTO PROPERTY VALUES(1,'1','30 West Thomas Rd.',1600,3,1,1400,'BU106');
INSERT INTO PROPERTY VALUES(2,'1','782 Queen Ln.',2100,4,2,1900,'AK102');
INSERT INTO PROPERTY VALUES(3,'1','9800 Sunbeam Ave.',1005,2,1,1200,'BI109');
INSERT INTO PROPERTY VALUES(4,'1','105 North Illinois Rd.',1750,3,1,1650,'KO104');
INSERT INTO PROPERTY VALUES(5,'1','887 Vine Rd.',1125,2,1,1160,'SI105');
INSERT INTO PROPERTY VALUES(6,'1','8 Laurel Dr.',2125,4,2,2050,'MO100');
INSERT INTO PROPERTY VALUES(7,'2','447 Goldfield St.',1675,3,2,1700,'CO103');
INSERT INTO PROPERTY VALUES(8,'2','594 Leatherwood Dr.',2700,5,2,2750,'KO104');
INSERT INTO PROPERTY VALUES(9,'2','504 Windsor Ave.',700,2,1,1050,'PA101');
INSERT INTO PROPERTY VALUES(10,'2','891 Alton Dr.',1300,3,1,1600,'LO108');
INSERT INTO PROPERTY VALUES(11,'2','9531 Sherwood Rd.',1075,2,1,1100,'JO110');
INSERT INTO PROPERTY VALUES(12,'2','2 Bow Ridge Ave.',1400,3,2,1700,'RE107');

CREATE TABLE SERVICE_CATEGORY(CATEGORY_NUM DECIMAL(2,0) NOT NULL
PRIMARY KEY, CATEGORY_DESCRIPTION CHAR(35));
```

```
INSERT INTO SERVICE_CATEGORY VALUES('1', 'Plumbing');
INSERT INTO SERVICE_CATEGORY VALUES('2', 'Heating');
INSERT INTO SERVICE_CATEGORY VALUES('3', 'Painting');
INSERT INTO SERVICE_CATEGORY VALUES('4', 'Electrical Systems');
INSERT INTO SERVICE_CATEGORY VALUES('5', 'Carpentry');
INSERT INTO SERVICE_CATEGORY VALUES('6', 'Furniture Replacement');
```

```
CREATE TABLE SERVICE_REQUEST(SERVICE_ID DECIMAL(2,0) NOT NULL
PRIMARY KEY, PROPERTY_ID DECIMAL(2,0), CATEGORY_NUM DECIMAL(2,0),
OFFICE_NUM DECIMAL (2,0), DESCRIPTION CHAR(255), STATUS CHAR(255),
EST_HOURS DECIMAL(4,0), SPENT_HOURS DECIMAL(4,0), NEXT_SERVICE_DATE
DATE);
```

```
INSERT INTO SERVICE_REQUEST VALUES(1,11,2,2,'The second bedroom upstairs is not
heating up at night.', 'Problem has been confirmed. Central heating engineer has been
scheduled',2,1,'2019-11-01');
```

```
INSERT INTO SERVICE_REQUEST VALUES(2,1,4,1,'A new strip light is needed for the
kitchen', 'Scheduled',1,0,'2019-10-02');
```

```
INSERT INTO SERVICE_REQUEST VALUES(3,6,5,1,'The bathroom door does not close
properly.', 'Service rep has confirmed issue. Scheduled to be refitted',3,1,'2019-11-09');
```

```
INSERT INTO SERVICE_REQUEST VALUES(4,2,4,1,'New outlet has been requested for
upstairs bedroom.', 'Scheduled',1,0,'2019-10-02');
```

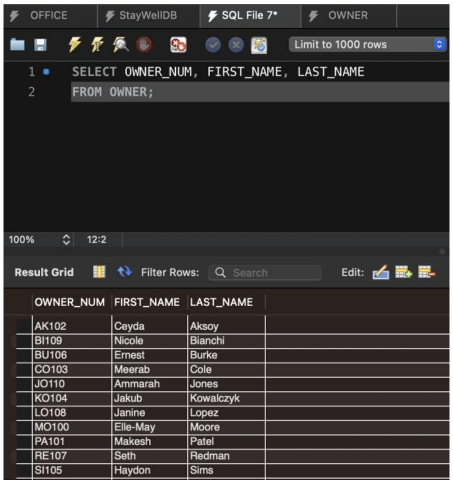
```
INSERT INTO SERVICE_REQUEST VALUES(5,8,3,2,'New paint job requested for common
area.', 'Open',10,0, null);
```

```
INSERT INTO SERVICE_REQUEST VALUES(6,4,1,1,'Shower is dripping when not in use.',

'Problem has been confirmed. Plumber scheduled',4,2,'2019-10-07');
```

```
INSERT INTO SERVICE_REQUEST VALUES(7,2,2,1,'Heating unit in entrance smells like its burning', 'Service rep confirmed the issue to be dust in heating unit.To be cleaned',1,0,'2019-10-09');

INSERT INTO SERVICE_REQUEST VALUES(8,9,1,2,'Kitchen sink does not drain properly.', 'Problem has been confirmed. Plumber scheduled',6,2,'2019-11-12');

INSERT INTO SERVICE_REQUEST VALUES(9,12,6,2,'New sofa requested.', 'Open',2,0, null);

CREATE TABLE RESIDENTS(RESIDENT_ID DECIMAL(2,0) NOT NULL PRIMARY KEY, FIRST_NAME CHAR(25),  SURNAME CHAR(25),PROPERTY_ID DECIMAL(2,0));

INSERT INTO RESIDENTS Values(1,'Albie',"O'Ryan",1);
INSERT INTO RESIDENTS Values(2,'Tariq',"Khan",1);
INSERT INTO RESIDENTS Values(3,'Ismail',"Salib",1);
INSERT INTO RESIDENTS Values(4,'Callen',"Beek",2);
INSERT INTO RESIDENTS Values(5,'Milosz',"Polansky",2);
INSERT INTO RESIDENTS Values(6,'Ashanti',"Lucas",2);
INSERT INTO RESIDENTS Values(7,'Randy',"Woodrue",2);
INSERT INTO RESIDENTS Values(8,'Aislinn',"Lawrence",3);
INSERT INTO RESIDENTS Values(9,'Monique',"French",3);
INSERT INTO RESIDENTS Values(10,'Amara',"Dejsuwan",4);
INSERT INTO RESIDENTS Values(11,'Rosalie',"Blackmore",4);
INSERT INTO RESIDENTS Values(12,'Carina',"Britton",4);
INSERT INTO RESIDENTS Values(13,'Valentino',"Ortega",5);
INSERT INTO RESIDENTS Values(14,'Kaylem',"Kent",5);
INSERT INTO RESIDENTS Values(15,'Alessia',"Wagner",6);
INSERT INTO RESIDENTS Values(16,'Tyrone',"Galvan",6);
INSERT INTO RESIDENTS Values(17,'Constance',"Fleming",6);

INSERT INTO RESIDENTS Values(18,'Eamonn',"Bain",6);

INSERT INTO RESIDENTS Values(19,'Misbah',"Yacob",7);

INSERT INTO RESIDENTS Values(20,'Gianluca',"Esposito",7);

INSERT INTO RESIDENTS Values(21,'Elinor',"Lake",7);

INSERT INTO RESIDENTS Values(22,'Ray',"Rosas",8);

INSERT INTO RESIDENTS Values(23,'Damon',"Caldwell",8);

INSERT INTO RESIDENTS Values(24,'Dawood',"Busby",8);

INSERT INTO RESIDENTS Values(25,'Dora',"Harries",8);

INSERT INTO RESIDENTS Values(26,'Leroy',"Stokes",8);

INSERT INTO RESIDENTS Values(27,'Tamia',"Hess",9);

INSERT INTO RESIDENTS Values(28,'Amelia',"Sanders",9);

INSERT INTO RESIDENTS Values(29,'Zarah',"Byers",10);

INSERT INTO RESIDENTS Values(30,'Sara',"Farrow",10);

INSERT INTO RESIDENTS Values(31,'Delilah',"Roy",10);

INSERT INTO RESIDENTS Values(32,'Dougie',"McDaniel",11);

INSERT INTO RESIDENTS Values(33,'Tahir',"Halabi",11);

INSERT INTO RESIDENTS Values(34,'Mila',"Zhikin",12);

INSERT INTO RESIDENTS Values(35,'Glenn',"Donovan",12);

INSERT INTO RESIDENTS Values(36,'Zayn',"Fowler",12);
```

Chapter 1

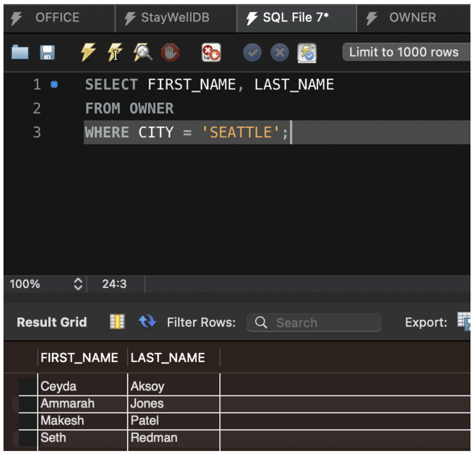
1. **List the owner number, last name, and first name of every property owner.**



```sql
SELECT OWNER_NUM, FIRST_NAME, LAST_NAME
FROM OWNER;
```

OWNER_NUM	FIRST_NAME	LAST_NAME
AK102	Ceyda	Aksoy
BI109	Nicole	Bianchi
BU106	Ernest	Burke
CO103	Meerab	Cole
JO110	Ammarah	Jones
KO104	Jakub	Kowalczyk
LO108	Janine	Lopez
MO100	Elle-May	Moore
PA101	Makesh	Patel
RE107	Seth	Redman
SI105	Haydon	Sims

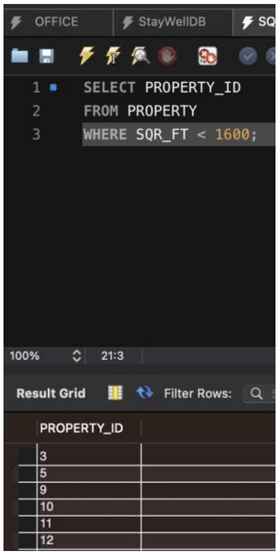
2. **List the last name and first name of every owner located in Seattle.**



```sql
SELECT FIRST_NAME, LAST_NAME
FROM OWNER
WHERE CITY = 'SEATTLE';
```

FIRST_NAME	LAST_NAME
Ceyda	Aksoy
Ammarah	Jones
Makesh	Patel
Seth	Redman

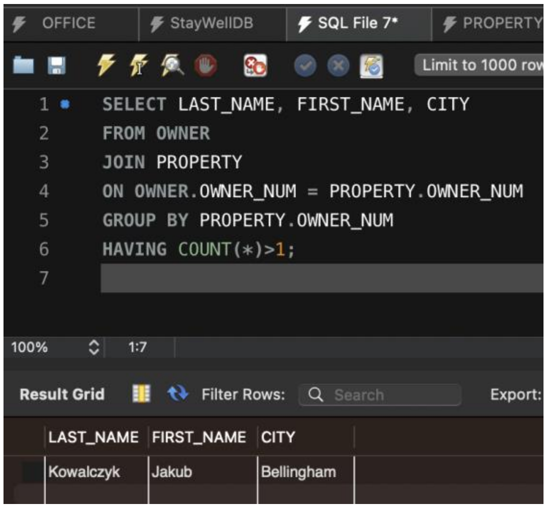
3. **List the property ID for each condo that is smaller than 1,600 square feet.**



```
SELECT PROPERTY_ID
FROM PROPERTY
WHERE SQR_FT < 1600;
```

PROPERTY_ID
3
5
9
10
11
12

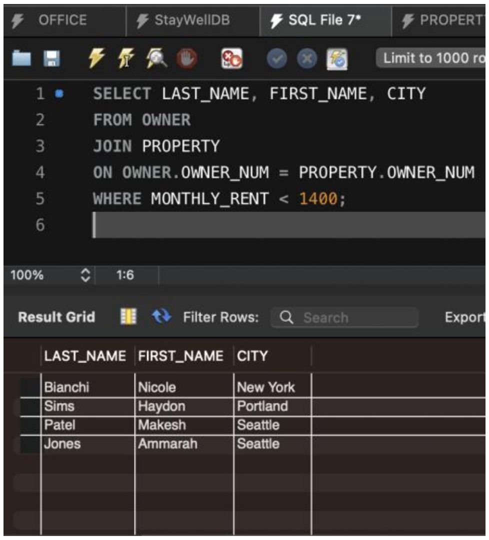
4. **List the last name, first name, and city of every owner who owns more than one property in the database.**



```
SELECT LAST_NAME, FIRST_NAME, CITY
FROM OWNER
JOIN PROPERTY
ON OWNER.OWNER_NUM = PROPERTY.OWNER_NUM
GROUP BY PROPERTY.OWNER_NUM
HAVING COUNT(*)>1;
```

LAST_NAME	FIRST_NAME	CITY
Kowalczyk	Jakub	Bellingham

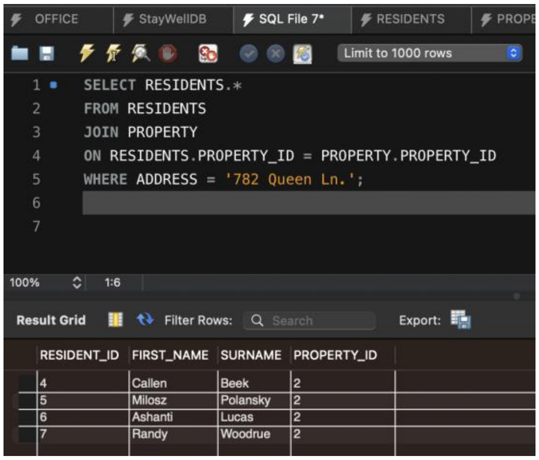
5. **List the last name, first name, and city of every owner with a property that has a monthly rent of less than $1,400 per month.**



```
SELECT LAST_NAME, FIRST_NAME, CITY
FROM OWNER
JOIN PROPERTY
ON OWNER.OWNER_NUM = PROPERTY.OWNER_NUM
WHERE MONTHLY_RENT < 1400;
```

LAST_NAME	FIRST_NAME	CITY
Bianchi	Nicole	New York
Sims	Haydon	Portland
Patel	Makesh	Seattle
Jones	Ammarah	Seattle

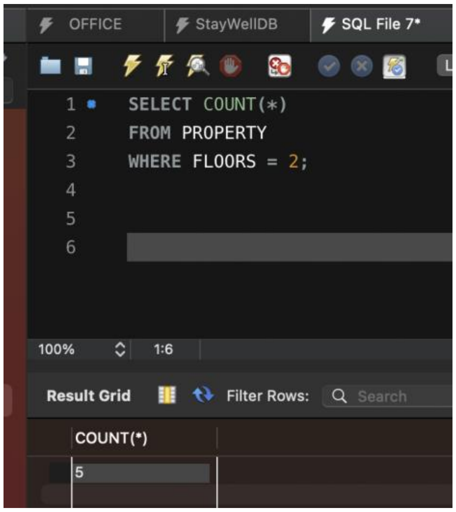
6. List all the residents staying at 782 Queen Ln.



```
SELECT RESIDENTS.*
FROM RESIDENTS
JOIN PROPERTY
ON RESIDENTS.PROPERTY_ID = PROPERTY.PROPERTY_ID
WHERE ADDRESS = '782 Queen Ln.';
```

RESIDENT_ID	FIRST_NAME	SURNAME	PROPERTY_ID
4	Callen	Beek	2
5	Milosz	Polansky	2
6	Ashanti	Lucas	2
7	Randy	Woodrue	2

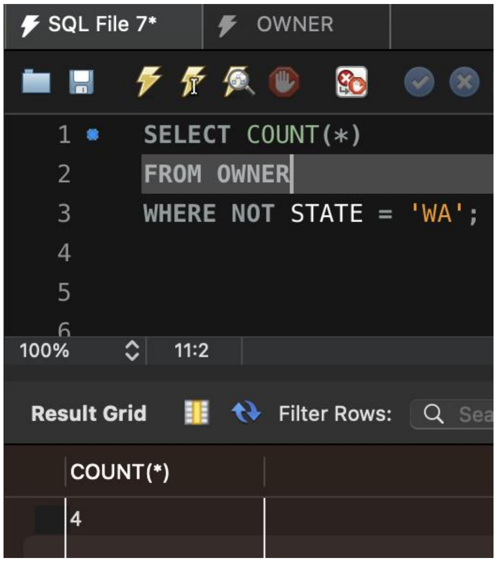
7. How many properties have two floors?



```sql
SELECT COUNT(*)
FROM PROPERTY
WHERE FLOORS = 2;
```

```
COUNT(*)
5
```

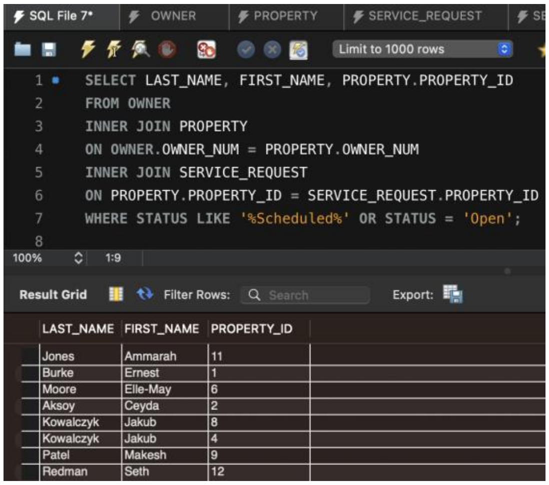
8. **How many owners live outside of Washington state (WA)?**



```sql
SELECT COUNT(*)
FROM OWNER
WHERE NOT STATE = 'WA';
```

```
COUNT(*)
4
```

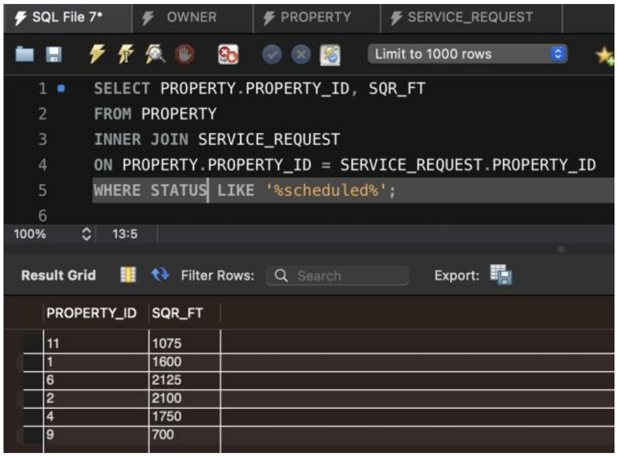
9. **List the owner's last and first names and property IDs for each property that has a scheduled or open service request.**



```sql
SELECT LAST_NAME, FIRST_NAME, PROPERTY.PROPERTY_ID
FROM OWNER
INNER JOIN PROPERTY
ON OWNER.OWNER_NUM = PROPERTY.OWNER_NUM
INNER JOIN SERVICE_REQUEST
ON PROPERTY.PROPERTY_ID = SERVICE_REQUEST.PROPERTY_ID
WHERE STATUS LIKE '%Scheduled%' OR STATUS = 'Open';
```

LAST_NAME	FIRST_NAME	PROPERTY_ID
Jones	Ammarah	11
Burke	Ernest	1
Moore	Elle-May	6
Aksoy	Ceyda	2
Kowalczyk	Jakub	8
Kowalczyk	Jakub	4
Patel	Makesh	9
Redman	Seth	12

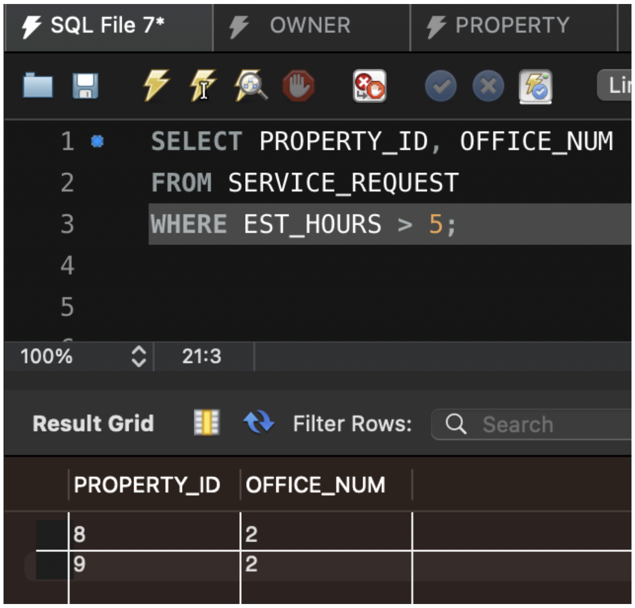
10. **List the property ID and square footage for each property that has a maintenance service request.**



```sql
SELECT PROPERTY.PROPERTY_ID, SQR_FT
FROM PROPERTY
INNER JOIN SERVICE_REQUEST
ON PROPERTY.PROPERTY_ID = SERVICE_REQUEST.PROPERTY_ID
WHERE STATUS LIKE '%scheduled%';
```

PROPERTY_ID	SQR_FT
11	1075
1	1600
6	2125
2	2100
4	1750
9	700

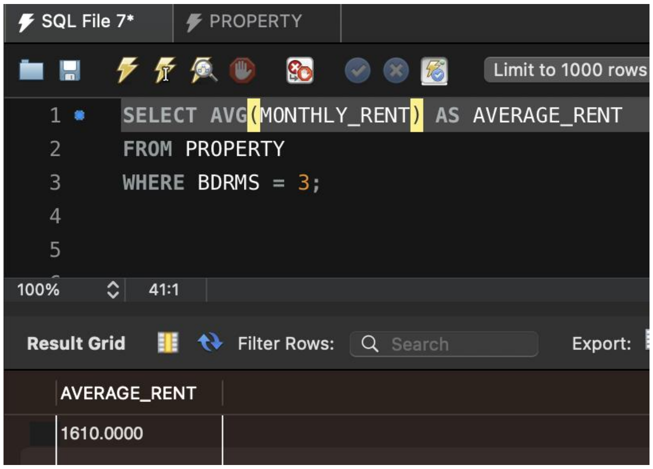
11. **List the property ID and office number for all service requests for which the estimated number of hours is greater than 5.**



```sql
SELECT PROPERTY_ID, OFFICE_NUM
FROM SERVICE_REQUEST
WHERE EST_HOURS > 5;
```

PROPERTY_ID	OFFICE_NUM
8	2
9	2

12. What is the average rent for all three-bedroom properties?



```sql
SELECT AVG(MONTHLY_RENT) AS AVERAGE_RENT
FROM PROPERTY
WHERE BDRMS = 3;
```

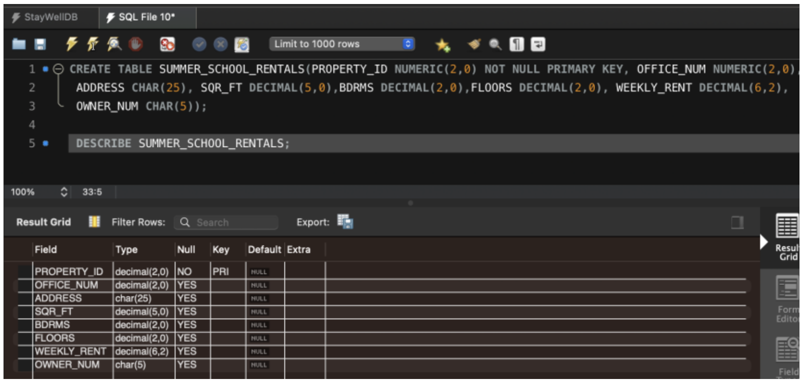
AVERAGE_RENT
1610.0000

Chapter 2

1. **Determine the functional dependencies that exist in the following table and then convert this table to an equivalent collection of tables that are in third normal form.**
   - □ Functional Dependencies:
     - o OFFICE_NUM → OFFICE_NAME
     - o ADDRESS → SQR_FT, BDRMS, FLOORS, MONTHLY_RENT, OWNER_NUM, OFFICE_NUM
   - □ Tables in 3NF:
     - o OFFICE(OFFICE_NUM, OFFICE_NAME)
     - o OFFICE_RENTED(ADDRESS,OFFICE_NUM, SQR_FT, BDRMS, FLOORS, MONTHLY_RENT,OWNER_NUM)
2. **Determine the functional dependencies that exist in the following table and then convert this table to an equivalent collection of tables that are in third normal form.**
   - □ Functional Dependencies:
     - o PROPERTY_ID → OFFICE_NUM, ADDRESS, SQR_FT, BDRMS, FLOORS, MONTHLY_RENT, OWNER_NUM
     - o OWNER_NUM → LAST_NAME, FIRST_NAME
   - □ Tables in 3NF:
     - o PROPERTY(PROPERTY_ID, OFFICE_NUM, ADDRESS, SQR_FT, BDRMS, FLOORS, MONTHLY_RENT, OWNER_NUM)
     - o OWNER(OWNER_NUM, LAST_NAME, FIRST_NAME)

Chapter 3

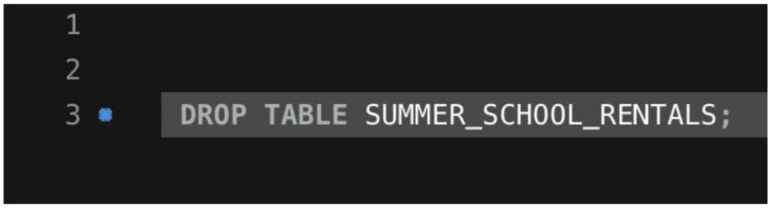
1. **Create a table named SUMMER_SCHOOL_RENTALS. The table has the same structure as the PROPERTY table shown in Figure 3-48 except the PROPERTY_ID and OFFICE_ NUMBER columns should use the NUMBER data type and the MONTHLY_RENT column should be changed to WEEKLY_RENT. Execute the command to describe the layout and characteristics of the SUMMER_SCHOOL_RENTALS table.**



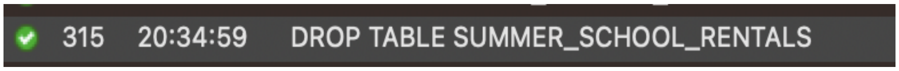
```
CREATE TABLE SUMMER_SCHOOL_RENTALS(PROPERTY_ID NUMERIC(2,0) NOT NULL PRIMARY KEY, OFFICE_NUM NUMERIC(2,0),
 ADDRESS CHAR(25), SQR_FT DECIMAL(5,0),BDRMS DECIMAL(2,0),FLOORS DECIMAL(2,0), WEEKLY_RENT DECIMAL(6,2),
 OWNER_NUM CHAR(5));

DESCRIBE SUMMER_SCHOOL_RENTALS;
```

Field	Type	Null	Key	Default	Extra
PROPERTY_ID	decimal(2,0)	NO	PRI	NULL	
OFFICE_NUM	decimal(2,0)	YES		NULL	
ADDRESS	char(25)	YES		NULL	
SQR_FT	decimal(5,0)	YES		NULL	
BDRMS	decimal(2,0)	YES		NULL	
FLOORS	decimal(2,0)	YES		NULL	
WEEKLY_RENT	decimal(6,2)	YES		NULL	
OWNER_NUM	char(5)	YES		NULL	

2. **(3) Delete the SUMMER_SCHOOL_RENTALS table.**

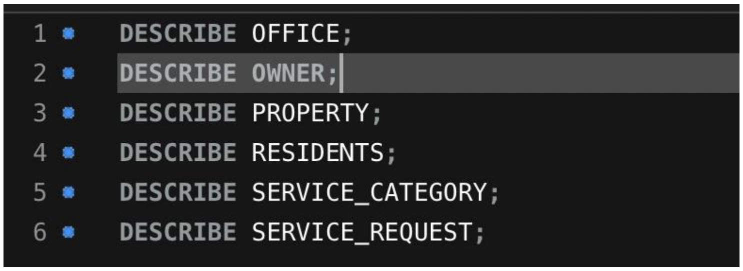


```sql


DROP TABLE SUMMER_SCHOOL_RENTALS;
```

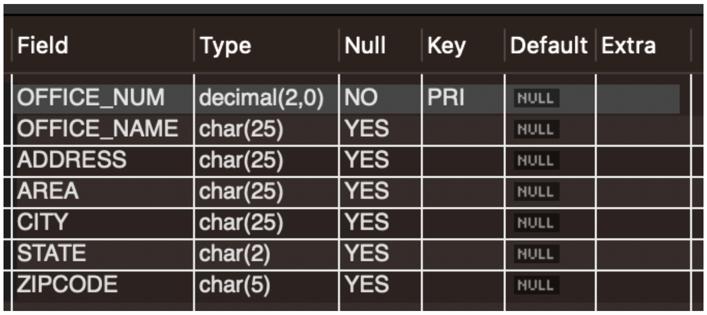


```
✓  315   20:34:59   DROP TABLE SUMMER_SCHOOL_RENTALS
```

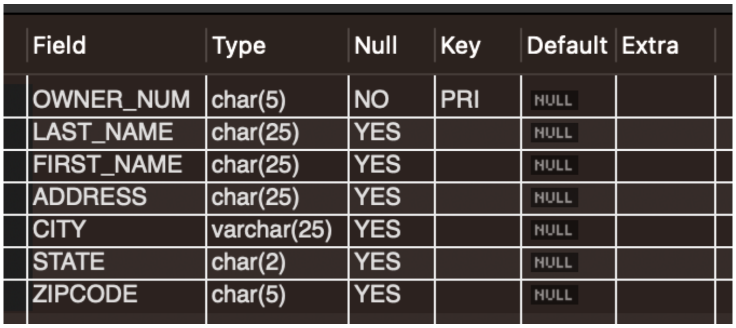
3. **(5) Confirm that you have created the tables correctly by describing each table and comparing the results to Figures 3-48.**



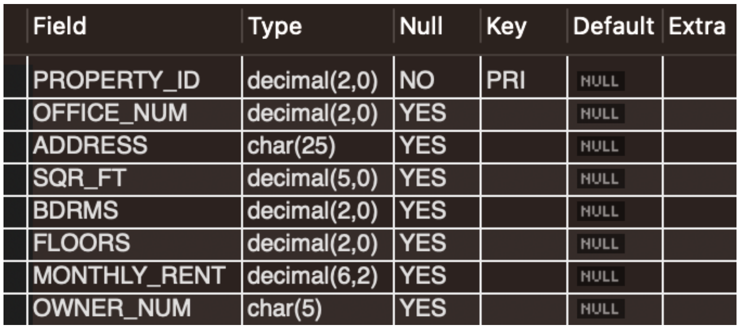
```sql
DESCRIBE OFFICE;
DESCRIBE OWNER;
DESCRIBE PROPERTY;
DESCRIBE RESIDENTS;
DESCRIBE SERVICE_CATEGORY;
DESCRIBE SERVICE_REQUEST;
```



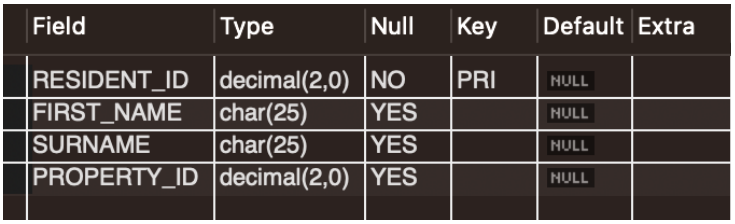
Field	Type	Null	Key	Default	Extra
OFFICE_NUM	decimal(2,0)	NO	PRI	NULL	
OFFICE_NAME	char(25)	YES		NULL	
ADDRESS	char(25)	YES		NULL	
AREA	char(25)	YES		NULL	
CITY	char(25)	YES		NULL	
STATE	char(2)	YES		NULL	
ZIPCODE	char(5)	YES		NULL	



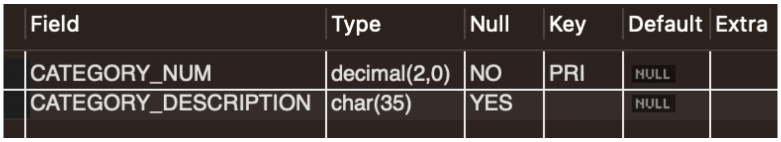
Field	Type	Null	Key	Default	Extra
OWNER_NUM	char(5)	NO	PRI	NULL	
LAST_NAME	char(25)	YES		NULL	
FIRST_NAME	char(25)	YES		NULL	
ADDRESS	char(25)	YES		NULL	
CITY	varchar(25)	YES		NULL	
STATE	char(2)	YES		NULL	
ZIPCODE	char(5)	YES		NULL	



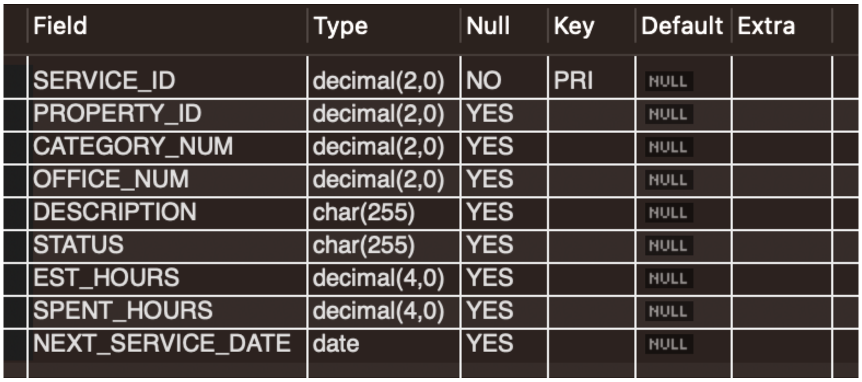
Field	Type	Null	Key	Default	Extra
PROPERTY_ID	decimal(2,0)	NO	PRI	NULL	
OFFICE_NUM	decimal(2,0)	YES		NULL	
ADDRESS	char(25)	YES		NULL	
SQR_FT	decimal(5,0)	YES		NULL	
BDRMS	decimal(2,0)	YES		NULL	
FLOORS	decimal(2,0)	YES		NULL	
MONTHLY_RENT	decimal(6,2)	YES		NULL	
OWNER_NUM	char(5)	YES		NULL	



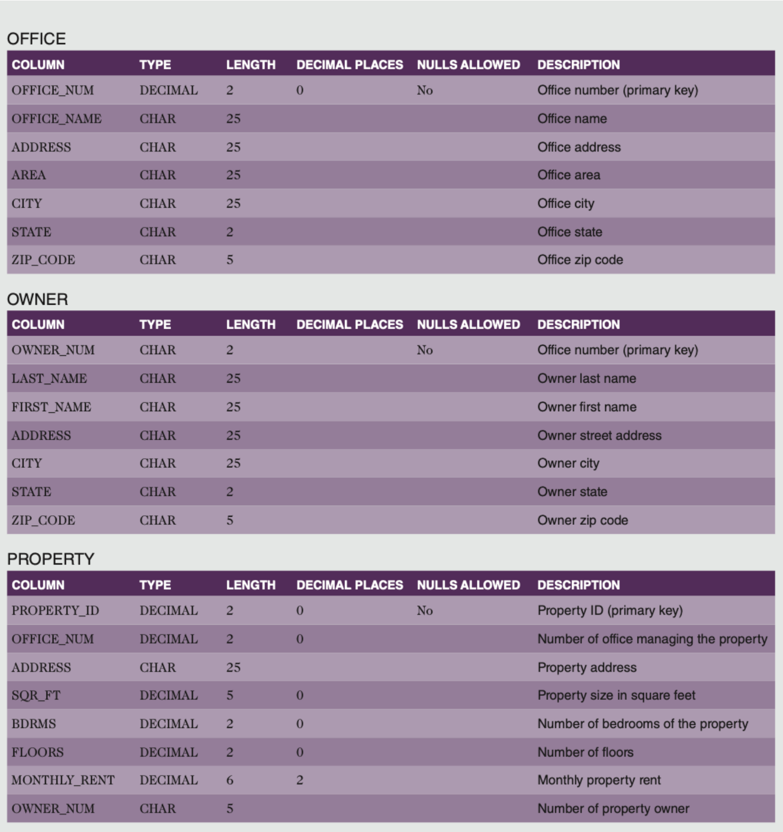
Field	Type	Null	Key	Default	Extra
RESIDENT_ID	decimal(2,0)	NO	PRI	NULL	
FIRST_NAME	char(25)	YES		NULL	
SURNAME	char(25)	YES		NULL	
PROPERTY_ID	decimal(2,0)	YES		NULL	



Field	Type	Null	Key	Default	Extra
CATEGORY_NUM	decimal(2,0)	NO	PRI	NULL	
CATEGORY_DESCRIPTION	char(35)	YES		NULL	



Field	Type	Null	Key	Default	Extra
SERVICE_ID	decimal(2,0)	NO	PRI	NULL	
PROPERTY_ID	decimal(2,0)	YES		NULL	
CATEGORY_NUM	decimal(2,0)	YES		NULL	
OFFICE_NUM	decimal(2,0)	YES		NULL	
DESCRIPTION	char(255)	YES		NULL	
STATUS	char(255)	YES		NULL	
EST_HOURS	decimal(4,0)	YES		NULL	
SPENT_HOURS	decimal(4,0)	YES		NULL	
NEXT_SERVICE_DATE	date	YES		NULL	



OFFICE

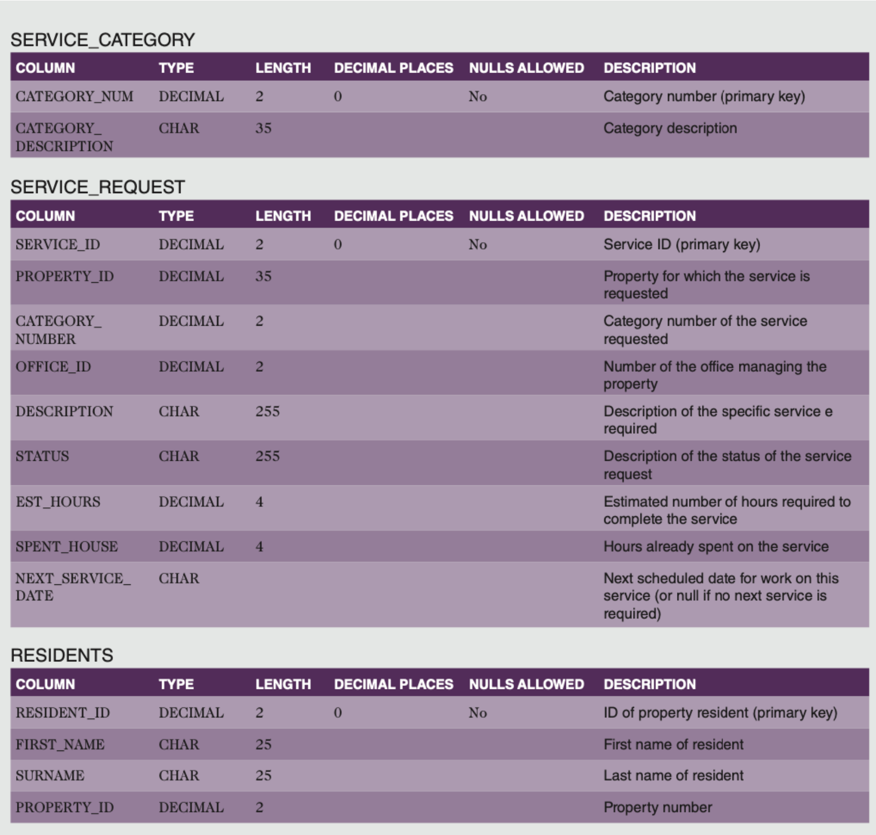
COLUMN	TYPE	LENGTH	DECIMAL PLACES	NULLS ALLOWED	DESCRIPTION
OFFICE_NUM	DECIMAL	2	0	No	Office number (primary key)
OFFICE_NAME	CHAR	25			Office name
ADDRESS	CHAR	25			Office address
AREA	CHAR	25			Office area
CITY	CHAR	25			Office city
STATE	CHAR	2			Office state
ZIP_CODE	CHAR	5			Office zip code

OWNER

COLUMN	TYPE	LENGTH	DECIMAL PLACES	NULLS ALLOWED	DESCRIPTION
OWNER_NUM	CHAR	2		No	Office number (primary key)
LAST_NAME	CHAR	25			Owner last name
FIRST_NAME	CHAR	25			Owner first name
ADDRESS	CHAR	25			Owner street address
CITY	CHAR	25			Owner city
STATE	CHAR	2			Owner state
ZIP_CODE	CHAR	5			Owner zip code

PROPERTY

COLUMN	TYPE	LENGTH	DECIMAL PLACES	NULLS ALLOWED	DESCRIPTION
PROPERTY_ID	DECIMAL	2	0	No	Property ID (primary key)
OFFICE_NUM	DECIMAL	2	0		Number of office managing the property
ADDRESS	CHAR	25			Property address
SQR_FT	DECIMAL	5	0		Property size in square feet
BDRMS	DECIMAL	2	0		Number of bedrooms of the property
FLOORS	DECIMAL	2	0		Number of floors
MONTHLY_RENT	DECIMAL	6	2		Monthly property rent
OWNER_NUM	CHAR	5			Number of property owner



SERVICE_CATEGORY

COLUMN	TYPE	LENGTH	DECIMAL PLACES	NULLS ALLOWED	DESCRIPTION
CATEGORY_NUM	DECIMAL	2	0	No	Category number (primary key)
CATEGORY_DESCRIPTION	CHAR	35			Category description

SERVICE_REQUEST

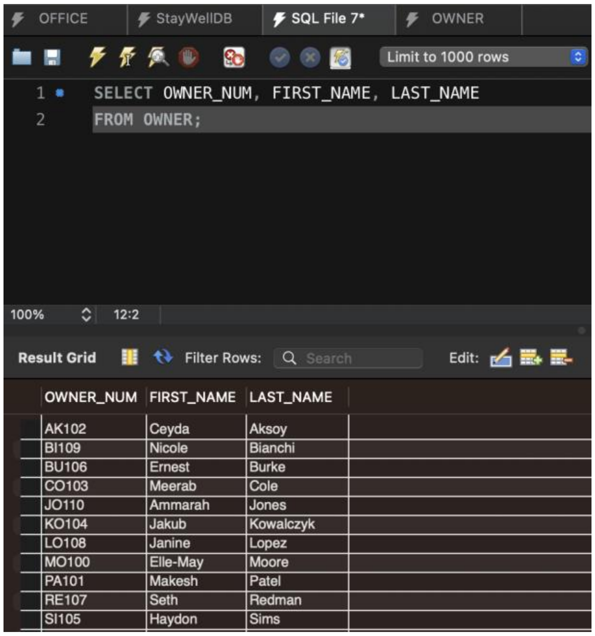
COLUMN	TYPE	LENGTH	DECIMAL PLACES	NULLS ALLOWED	DESCRIPTION
SERVICE_ID	DECIMAL	2	0	No	Service ID (primary key)
PROPERTY_ID	DECIMAL	35			Property for which the service is requested
CATEGORY_NUMBER	DECIMAL	2			Category number of the service requested
OFFICE_ID	DECIMAL	2			Number of the office managing the property
DESCRIPTION	CHAR	255			Description of the specific service e required
STATUS	CHAR	255			Description of the status of the service request
EST_HOURS	DECIMAL	4			Estimated number of hours required to complete the service
SPENT_HOUSE	DECIMAL	4			Hours already spent on the service
NEXT_SERVICE_DATE	CHAR				Next scheduled date for work on this service (or null if no next service is required)

RESIDENTS

COLUMN	TYPE	LENGTH	DECIMAL PLACES	NULLS ALLOWED	DESCRIPTION
RESIDENT_ID	DECIMAL	2	0	No	ID of property resident (primary key)
FIRST_NAME	CHAR	25			First name of resident
SURNAME	CHAR	25			Last name of resident
PROPERTY_ID	DECIMAL	2			Property number

Chapter 4

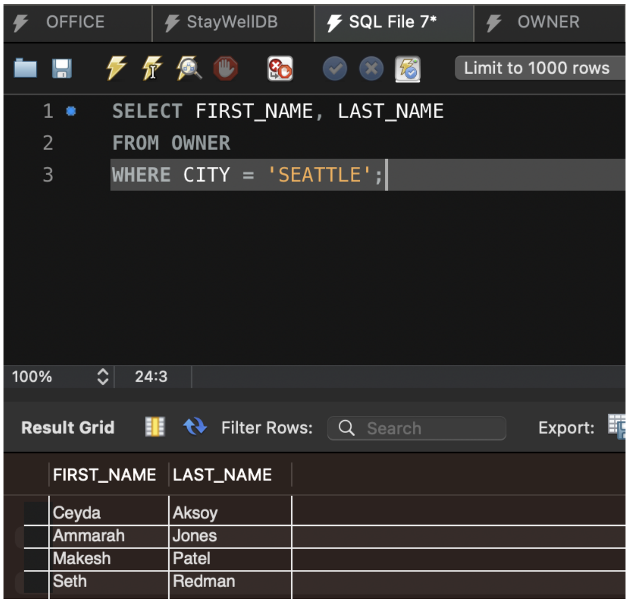
1. **List the owner number, last name, and first name of every property owner.**



```
SELECT OWNER_NUM, FIRST_NAME, LAST_NAME
FROM OWNER;
```

OWNER_NUM	FIRST_NAME	LAST_NAME
AK102	Ceyda	Aksoy
BI109	Nicole	Bianchi
BU106	Ernest	Burke
CO103	Meerab	Cole
JO110	Ammarah	Jones
KO104	Jakub	Kowalczyk
LO108	Janine	Lopez
MO100	Elle-May	Moore
PA101	Makesh	Patel
RE107	Seth	Redman
SI105	Haydon	Sims

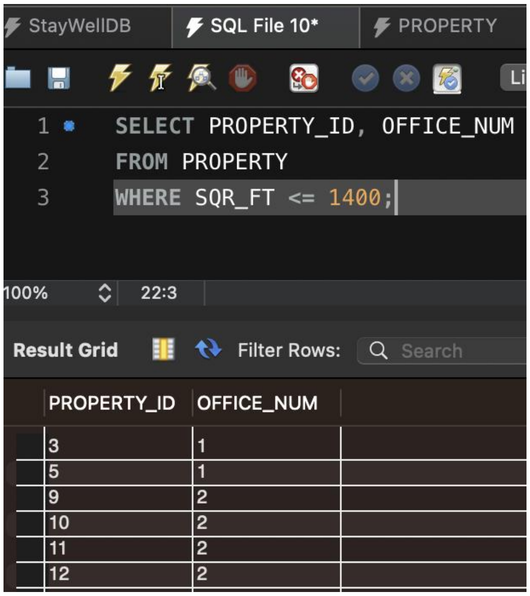
2. **(3)List the last name and first name of every owner who lives in Seattle.**



```
SELECT FIRST_NAME, LAST_NAME
FROM OWNER
WHERE CITY = 'SEATTLE';
```

FIRST_NAME	LAST_NAME
Ceyda	Aksoy
Ammarah	Jones
Makesh	Patel
Seth	Redman

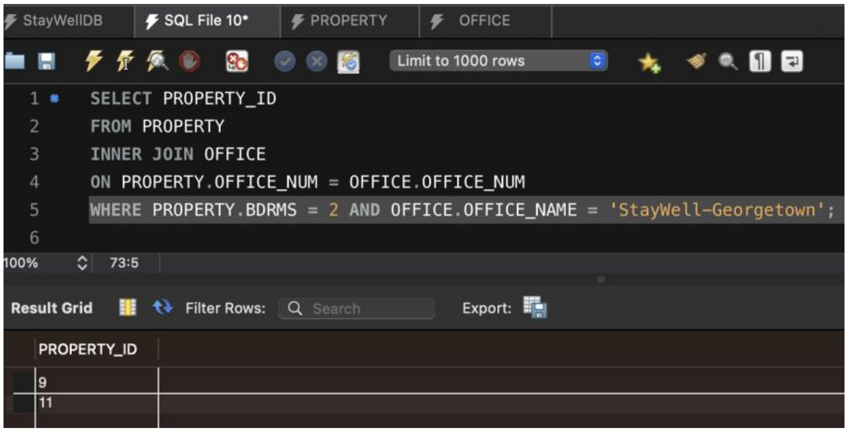
3. **(5)List the property ID and office number for every property whose square footage is equal to or less than 1,400 square feet.**



```sql
SELECT PROPERTY_ID, OFFICE_NUM
FROM PROPERTY
WHERE SQR_FT <= 1400;
```

PROPERTY_ID	OFFICE_NUM
3	1
5	1
9	2
10	2
11	2
12	2

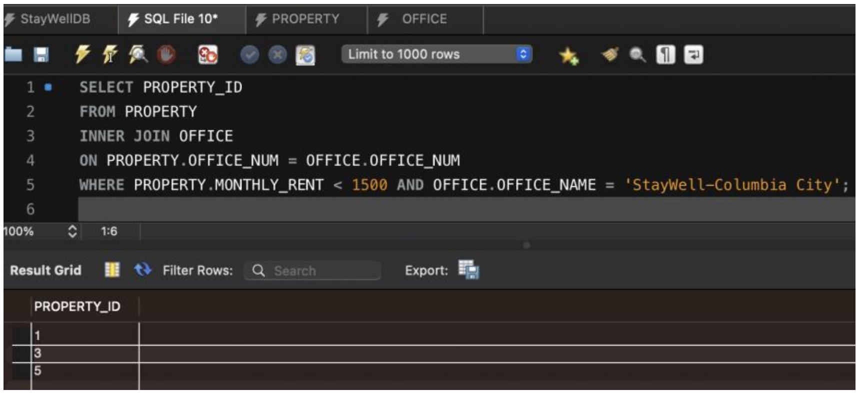
4. **(7)List the property ID for every property with two bedrooms that is managed by StayWell-Georgetown.**



```sql
SELECT PROPERTY_ID
FROM PROPERTY
INNER JOIN OFFICE
ON PROPERTY.OFFICE_NUM = OFFICE.OFFICE_NUM
WHERE PROPERTY.BDRMS = 2 AND OFFICE.OFFICE_NAME = 'StayWell-Georgetown';
```

PROPERTY_ID
9
11

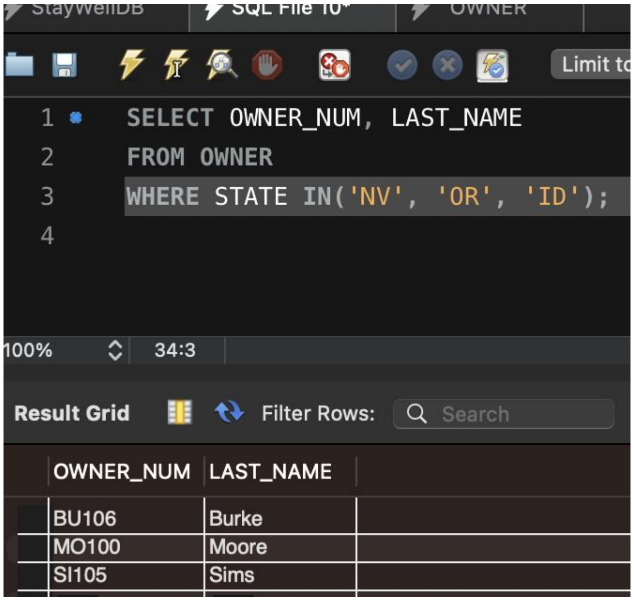
5. **(9)List the property ID for every property managed by StayWell-Columbia City whose monthly rent is less than $1,500.**



```sql
SELECT PROPERTY_ID
FROM PROPERTY
INNER JOIN OFFICE
ON PROPERTY.OFFICE_NUM = OFFICE.OFFICE_NUM
WHERE PROPERTY.MONTHLY_RENT < 1500 AND OFFICE.OFFICE_NAME = 'StayWell-Columbia City';
```

```
PROPERTY_ID
1
3
5
```

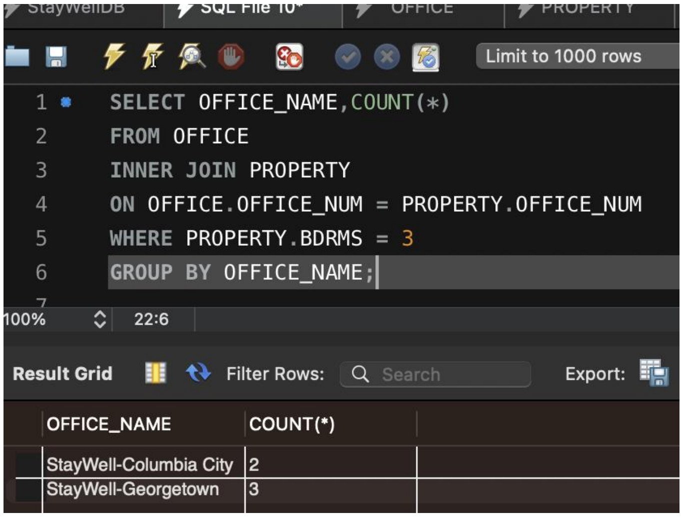
6. **(11)List the owner number and last name for all owners who live in Nevada (NV), Oregon (OR), or Idaho (ID).**



```sql
SELECT OWNER_NUM, LAST_NAME
FROM OWNER
WHERE STATE IN('NV', 'OR', 'ID');
```

```
OWNER_NUM  LAST_NAME
BU106      Burke
MO100      Moore
SI105      Sims
```

7. **(13)How many three-bedroom properties are managed by each office?**



```sql
SELECT OFFICE_NAME,COUNT(*)
FROM OFFICE
INNER JOIN PROPERTY
ON OFFICE.OFFICE_NUM = PROPERTY.OFFICE_NUM
WHERE PROPERTY.BDRMS = 3
GROUP BY OFFICE_NAME;
```

OFFICE_NAME	COUNT(*)
StayWell-Columbia City	2
StayWell-Georgetown	3

REFERENCES

“MySQL Tutorial for Beginners [Full Course].” *YouTube*, YouTube, 19 Mar. 2019, www.youtube.com/watch?v=7S_tz1z_5bA.

Shellman, M., Afyouni, H. A., Pratt, P. J., & Last, M. Z. (2021). *A guide to SQL*. Cengage.

Bagui, S., & Earp, R. (2003). *Database design using entity-relationship diagrams*. Boca Raton: Auerbach.