Computer Science

Coral S. Schmidt Montilla

#148830

Database Systems

FA 2024 CECS4202-81

Contents

[Part I. Review Question and Exercises from the book Database Systems: Design, Implementation, & Management by Carlos Coronel and Steven Morris (Use Edition 12 that is not so expensive) (50 points). 3](#_Toc177729576)

[Chapter 1. 3](#_Toc177729577)

[Chapter 2. 8](#_Toc177729578)

[Chapter 3. 12](#_Toc177729579)

[Chapter 4. 18](#_Toc177729580)

[Part II: SQL (60 points) 24](#_Toc177729581)

[A. Review Questions (Module Quiz) Chapter 2-4 of the book A Guide to SQL (2015): Odd Review Questions. Copy the question in bold face above the answer.  Do not use bold face for the answers. 24](#_Toc177729582)

[Chapter 2: Odd Numbers: pg. 46-47 24](#_Toc177729583)

[Chapter 3: Odd Numbers: pg. 92-93 30](#_Toc177729584)

[Chapter 4: Odd Numbers: pg. 131-132 32](#_Toc177729585)

[B. Except for Chapters 1 and 2, please show the complete question, query used, and print the query results. 35](#_Toc177729586)

[Chapter 1 (A Guide to SQL) Use the Staywell student Accommodation database only. 35](#_Toc177729587)

[Chapter 2 (A Guide to SQL) Use the Staywell student Accommodation database only. 39](#_Toc177729588)

[For Chapters 3 and 4 please show the screen print out of the Oracle or MySql (among others) output. **Use the Staywell student Accommodation database only.** 44](#_Toc177729589)

[Chapter 3 (A Guide to SQL) Use the Staywell student Accommodation database only. 44](#_Toc177729590)

[Chapter 4 (A Guide to SQL) Use the Staywell student Accommodation database only. 56](#_Toc177729591)

# Part I. Review Question and Exercises from the book Database Systems: Design, Implementation, & Management by Carlos Coronel and Steven Morris (Use Edition 12 that is not so expensive) (50 points).

## Chapter 1.

Review Questions: 1-6: Page 32

**1. Define each of the following terms:**

**a. data:** Raw facts, such as a phone number or a birth date.

**b. field:** A character or group of characters (alphabetic or numeric) that has a specific meaning.

**c. record:** A logically connected set of one or more fields that describes a person, place, or thing.

**d. file:** A collection of related records.

**2. What is data redundancy, and which characteristics of the file system can lead to it?**

Data redundancy occurs when the same data is stored in multiple places within a system. Characteristics of file systems that lead to redundancy include a lack of data sharing and the absence of a central database, resulting in duplicated data across different files.

**3. What is data independence, and why is it lacking in file systems?**

Data independence refers to the separation of data descriptions from the application programs that use the data. File systems lack data independence because changes in the file structure require modifications to the application programs.

**4. What is a DBMS, and what are its functions?**

A Database Management System (DBMS) is a collection of programs that manage the database structure and control access to the data stored in the database. Its functions include data storage, retrieval, update, and management of data integrity and security.

**5. What is structural independence, and why is it important?**

Structural independence exists when changes to the database schema do not affect application programs. It is important because it allows the database structure to be altered without requiring changes to applications.

**6. Explain the differences among data, information, and a database.**

Data are raw facts. Information is processed data that conveys meaning. A database is a collection of organized data.

**Problems: 1-4: Page 33**

A close-up of a document

Description automatically generated

**Given the file structure shown in Figure P1.1, answer Problems 1–4.**

**1. How many records does the file contain? How many fields are there per record?**

The file contains 7 records. Each record contains 5 fields: PROJECT\_CODE, PROJECT\_MANAGER, MANAGER\_PHONE, MANAGER\_ADDRESS, and PROJECT\_BID\_PRICE.

**2. What problem would you encounter if you wanted to produce a listing by city? How would you solve this problem by altering the file structure?**

The problem is that the MANAGER\_ADDRESS field contains the full address, including street, city, state, and ZIP code in a single field. This makes it difficult to sort or filter by city. To solve this, you could split the MANAGER\_ADDRESS field into separate fields for STREET, CITY, STATE, and ZIP\_CODE. This would allow you to easily produce listings sorted by the city.

**3. If you wanted to produce a listing of the file contents by last name, area code, city, state, or zip code, how would you alter the file structure?**

For last name you would need to separate the PROJECT\_MANAGER field into two fields: MANAGER\_FIRST\_NAME and MANAGER\_LAST\_NAME. For area code you need to separate the MANAGER\_PHONE field into AREA\_CODE and PHONE\_NUMBER. Finaly for city, state, or zip code: As mentioned in Problem 2, split the MANAGER\_ADDRESS field into STREET, CITY, STATE, and ZIP\_CODE.

**4. What data redundancies do you detect? How could those redundancies lead to anomalies?**

The current file structure exhibits redundancies where `PROJECT\_MANAGER`, `MANAGER\_PHONE`, and `MANAGER\_ADDRESS` information are repeated for managers who oversee more than one project, such as Holly B. Parker and George F. Dorts. This repetition can lead to several potential anomalies. An update anomaly occurs if a project manager's phone number changes; you would need to update multiple records, and any oversight could result in inconsistent data. A deletion anomaly arises when deleting a project record, such as `21-5Z`, which could inadvertently remove all contact information for a project manager if no other records exist with their details. An insertion anomaly could also occur when adding a new project manager, as it would require entering redundant data into the table. To address these issues, the database should be normalized by creating a separate `MANAGERS` table that contains the `PROJECT\_MANAGER`, `MANAGER\_PHONE`, and `MANAGER\_ADDRESS` fields. The `PROJECTS` table would then include a foreign key to reference the `MANAGERS` table, effectively reducing redundancy and minimizing the risk of anomalies.

## Chapter 2.

Review Questions: 1-6: Pages 65-66

**1. Discuss the importance of data models.**

Data models are essential because they provide a structured way to organize data, ensuring consistency and accuracy in how data is stored, retrieved, and used within a database. They also serve as a blueprint for database design, guiding the creation of databases that reflect real-world entities and relationships.

**2. What is a business rule, and what is its purpose in data modeling?**

Data models are essential because they provide a structured way to organize data, ensuring consistency and accuracy in how data is stored, retrieved, and used within a database. They also serve as a blueprint for database design, guiding the creation of databases that reflect real-world entities and relationships.

**3. How do you translate business rules into data model components?**

Business rules are translated into data model components by identifying the entities, attributes, relationships, and constraints they imply. For example, a rule like "a customer can place many orders" would lead to creating a Customer entity, an Order entity, and a one-to-many relationship between them.

**4. Describe the basic features of the relational data model and discuss their importance to the end user and the designer**

The relational data model organizes data into tables (relations) where each table represents an entity. The model allows for powerful querying capabilities using SQL and ensures data integrity through primary and foreign keys. For the end user, this model provides a straightforward way to interact with data, while for the designer, it offers flexibility and a strong theoretical foundation for database design.

**5. Explain how the entity relationship (ER) model helped produce a more structured relational database design environment.**

The ER model provides a visual representation of entities and their relationships, which helps designers structure a database in a more organized way. It enables the identification of all necessary entities and relationships upfront, reducing the risk of errors and omissions in the database design process.

**6. Consider the scenario described by the statement “A customer can make many payments, but each payment is made by only one customer.” Use this scenario as the basis for an entity relationship diagram (ERD) representation.**

The scenario can be represented in an ERD with two entities: Customer and Payment. There would be a one-to-many (1) relationship from Customer to Payment, indicating that each customer can make multiple payments, but each payment is associated with only one customer.

**A diagram of a college

Description automatically generatedProblems 1-2: Page 66**

**Use the contents of Figure 2.1 to work Problems 1–3.**

**1. Write the business rule(s) that govern the relationship between AGENT and**

**CUSTOMER.**

The figure provided represents an internal model for Tiny College, illustrating the relationships between various entities such as PROFESSOR, CLASS, ROOM, and COURSE. Although the original question asked about the relationship between AGENT and CUSTOMER, which are not depicted in this figure, we can instead analyze the business rules relevant to Tiny College. According to the model, a PROFESSOR teaches multiple CLASSES, but each CLASS is taught by only one PROFESSOR. Additionally, a CLASS is scheduled in one ROOM, though a ROOM can host many CLASSES over time. Lastly, each COURSE generates multiple CLASSES, and each CLASS is associated with only one COURSE. These relationships reflect the structure and operations of Tiny College as represented in the provided diagram.

**2. Given the business rule(s) you wrote in Problem 1, create the basic Crow’s Foot**

**ERD.**

Based on the analysis of the Tiny College internal model, we can create a basic Crow’s Foot Entity-Relationship Diagram (ERD). The entities involved are PROFESSOR, CLASS, ROOM, and COURSE. The relationships between these entities are as follows: A PROFESSOR can teach many CLASSES, but each CLASS is taught by only one PROFESSOR, indicating a one-to-many (1) relationship. A CLASS must be scheduled in one ROOM, but a ROOM can host many CLASSES over time, also forming a one-to-many (1) relationship. Additionally, each COURSE generates multiple CLASSES, creating a one-to-many (1) relationship. The cardinality and optionality indicate that each PROFESSOR must teach at least one CLASS, each CLASS must be assigned to a ROOM, and each COURSE must generate at least one CLASS. This ERD provides a clear and structured visualization of how these entities interact within the context of Tiny College.

## Chapter 3.

Review Questions: 1-6: Page 107

**1. What is the difference between a database and a table?**

A database is a structured collection of data stored in a way that facilitates easy access, management, and updating. It can contain multiple tables. A table, on the other hand, is a specific structure within a database that organizes data into rows and columns, where each row represents a record, and each column represents an attribute of the data.

**2. What does it mean to say that a database displays both entity integrity and**

**referential integrity?**

Entity integrity ensures that each table has a primary key and that this key is unique and not null for all records in the table. Referential integrity means that foreign keys in a table must match primary keys in another table, ensuring that relationships between tables remain consistent.

**3. Why are entity integrity and referential integrity important in a database?**

Entity integrity is crucial because it ensures that each record in a table can be uniquely identified, preventing duplication and ensuring data accuracy. Referential integrity is important because it maintains consistency across related tables, ensuring that data referenced in one table has a corresponding, valid entry in another table.

**4. What are the requirements that two relations must satisfy to be considered**

**union-compatible?**

For two relations (tables) to be union-compatible, they must have the same number of attributes (columns), and corresponding attributes must have the same domain (i.e., the same data type and meaning).

**5. Which relational algebra operators can be applied to a pair of tables that are**

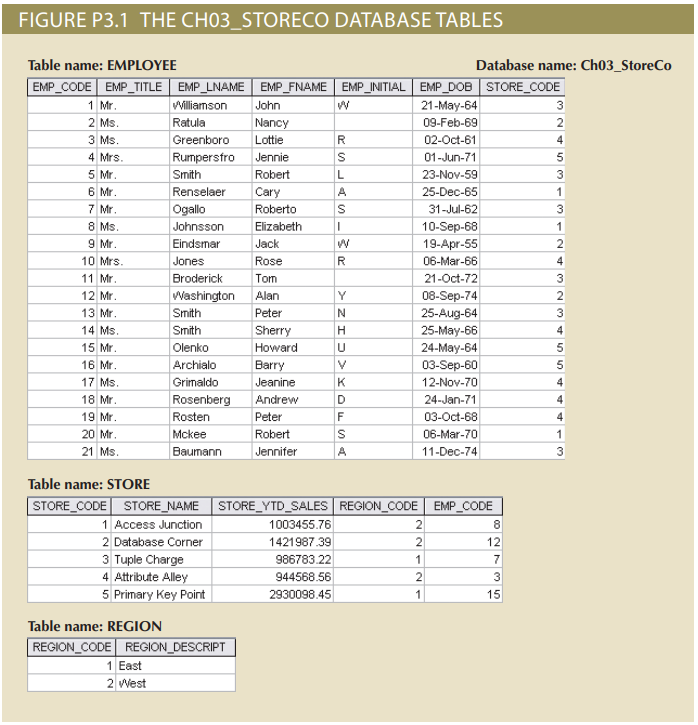
**not union-compatible?**

For tables that are not union-compatible, the following relational algebra operators can be applied: Cartesian Product (or Cross Join), Theta Join, and Division.

**6. Explain why the data dictionary is sometimes called “the database designer’s**

**database.”**

The data dictionary is often referred to as “the database designer’s database” because it contains metadata—information about the structure, organization, and constraints of the database itself. This includes details about tables, columns, data types, and relationships, which are essential for designing and maintaining the database.



**Problems: 1-6: Page 110**

**Use the database shown in Figure P3.1 to answer Problems 1–9.**

**1. For each table, identify the primary key and the foreign key(s). If a table does not**

**have a foreign key, write None.**

EMPLOYEE Table:

Primary Key: EMP\_CODE

Foreign Key: STORE\_CODE

STORE Table:

Primary Key: STORE\_CODE

Foreign Key: REGION\_CODE, EMP\_CODE

REGION Table:

Primary Key: REGION\_CODE

Foreign Key: None

**2. Do the tables exhibit entity integrity? Answer yes or no, and then explain your**

**answer.**

Yes, the tables exhibit entity integrity. Each table has a primary key (EMP\_CODE for EMPLOYEE, STORE\_CODE for STORE, and REGION\_CODE for REGION), and all the primary key fields have unique, non-null values.

**3. Do the tables exhibit referential integrity? Answer yes or no, and then explain your answer. Write NA (Not Applicable) if the table does not have a foreign key.**

EMPLOYEE Table:

NA (No foreign key in EMPLOYEE).

STORE Table:

Yes, the STORE table exhibits referential integrity because the REGION\_CODE and EMP\_CODE foreign keys refer to valid entries in the REGION and EMPLOYEE tables, respectively.

REGION Table:

NA (No foreign key in REGION).

**4. Describe the type(s) of relationship(s) between STORE and REGION.**

There is a one-to-many (1) relationship between REGION and STORE, where each REGION can have multiple STORE locations, but each STORE is associated with only one REGION.

**5. Create the ERD to show the relationship between STORE and REGION.**

A diagram of a diagram

Description automatically generated

**6. Create the relational diagram to show the relationship between STORE and REGION.**

A diagram of a diagram

Description automatically generated

## Chapter 4.

Review Questions: 1-6: Page 153-154

**1. What two conditions must be met before an entity can be classified as a weak entity? Give an example of a weak entity.**

A weak entity is an entity that cannot be uniquely identified by its own attributes alone. Two conditions must be met for an entity to be classified as a weak entity:

It must have a mandatory relationship with a strong entity (one that can be uniquely identified by its own attributes).

It must have a partial key or discriminator that, in combination with the primary key of the strong entity, uniquely identifies its instances.

Example: An example of a weak entity could be a "Dependent" entity in a database where the "Employee" entity is strong. Dependents rely on the employee entity for identification since they are identified through a combination of their attributes (like Dependent\_Name) and the employee's attributes (like Employee\_ID).

**2. What is a strong (or identifying) relationship, and how is it depicted in a Crow’s Foot**

**ERD?**

A strong (or identifying) relationship is a relationship where the primary key of the parent entity is part of the primary key in the child entity. In Crow’s Foot notation, it is depicted by a solid line connecting the entities, with a key symbol on the parent entity’s side and a dashed underline in the child entity to denote the foreign key.

**3. Given the business rule “an employee may have many degrees,” discuss its effect on**

**attributes, entities, and relationships. (Hint: Remember what a multivalued attribute**

**is and how it might be implemented.)**

The business rule implies that an "Employee" entity and a "Degree" entity should be created with a one-to-many relationship. The "Degree" entity would likely have attributes like Degree\_Type, Degree\_Field, and Graduation\_Year. Each degree would be related to one employee, but each employee could have multiple related degree entries.

**4. What is a composite entity, and when is it used?**

A composite entity (or associative entity) is used to capture a many-to-many relationship between two or more entities. It typically includes foreign keys from the entities it connects and may also contain additional attributes specific to the relationship. For example, a StudentCourse composite entity might connect Student and Course entities, with foreign keys from both and additional attributes like Enrollment\_Date.

A diagram of a car

Description automatically generated

**5. Suppose you are working within the framework of the conceptual model in**

**Figure Q4.5.**

**Given the conceptual model in Figure Q4.5:**

1. **Write the business rules that are reflected in it.**

The conceptual model presented reflects several key business rules that define the relationships between entities in a maintenance management system. Firstly, the model shows that a customer can own one or more cars, indicating a one-to-many relationship between customers and cars. Each car, however, is owned by exactly one customer, ensuring exclusivity in this relationship. Secondly, the model indicates that each car can undergo one or more maintenance procedures, emphasizing that maintenance activities are directly tied to specific cars. This relationship ensures that every maintenance record is associated with a single car.

Furthermore, the model details that a maintenance procedure can include one or more maintenance lines, which represent the detailed tasks or operations performed during the maintenance. Each of these maintenance lines is specific to one maintenance procedure, reinforcing the one-to-many relationship. Lastly, the model captures the fact that maintenance lines may require parts, and this relationship is many-to-many. This means that a single part can be used in multiple maintenance lines, and each maintenance line can involve several parts. These business rules ensure that the systemcan accurately track ownership, maintenance history, and the parts used in each maintenance task.

1. **Identify all of the cardinalities.**

The conceptual model illustrates several important cardinalities that define the relationships between entities. The relationship between Customer and Car is defined as a one-to-many (1) relationship. This indicates that while a customer can own multiple cars, each car is associated with exactly one customer. This ensures that the ownership of cars is uniquely tied to individual customers within the database.

Similarly, the relationship between Car and Maintenance is also a one-to-many (1) relationship, meaning that a single car can have multiple maintenance records over time, but each maintenance record pertains to one specific car. This relationship helps to maintain a clear history of maintenance activities for each car.

The relationship between Maintenance and Maintenance Line is again one-to-many (1). This shows that each maintenance procedure can consist of multiple detailed tasks, or maintenance lines, but each of these lines is associated with a specific maintenance procedure. Finally, the relationship between Maintenance Line and Part is a many-to-many (M) relationship, indicating that parts can be used in various maintenance lines, and each maintenance line may involve multiple parts. This many-to-many relationship allows the system to track the usage and allocation of parts across different maintenance tasks comprehensively.

These cardinalities are essential for defining how the entities interact within the database, ensuring that the system can accurately represent the relationships and constraints necessary for effective maintenance management.

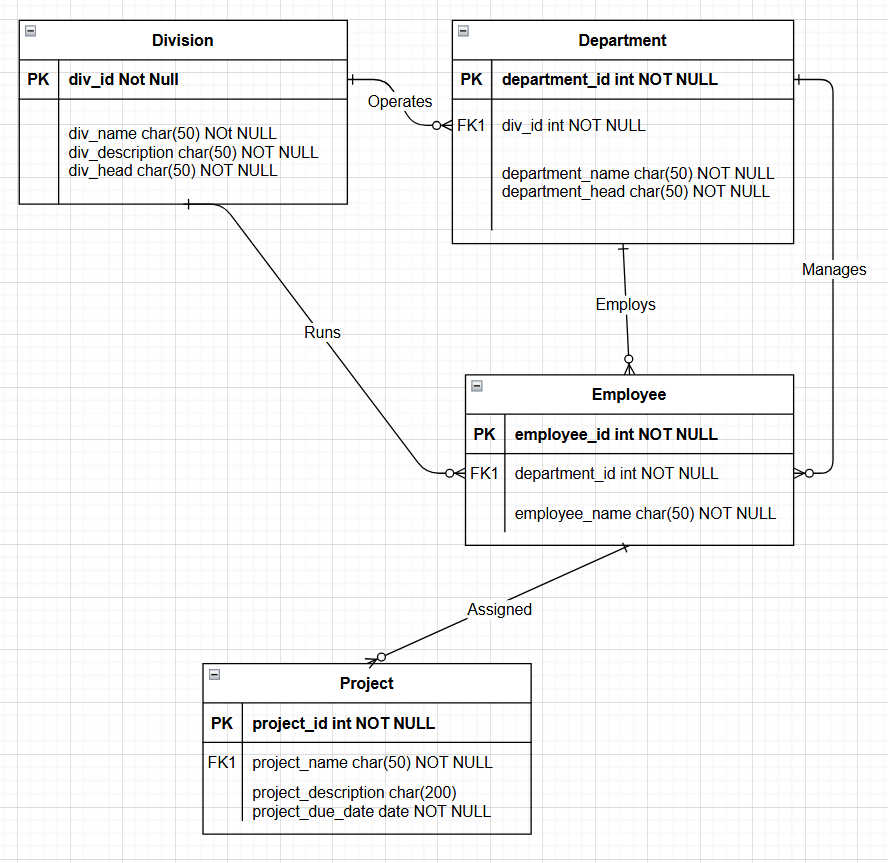
**6. What is a recursive relationship? Give an example**.

A recursive relationship occurs when an entity is related to itself. For example, in an employee database, a recursive relationship might model a situation where one employee (a manager) supervises other employees. In this case, the "Employee" entity might have a relationship with itself, representing the "supervises" relationship.

**Problems: 1: Page 156**

**1. Use the following business rules to create a Crow’s Foot ERD. Write all appropriate**

**connectivities and cardinalities in the ERD.**

* **A department employs many employees, but each employee is employed by only one department.**
* **Some employees, known as “rovers,” are not assigned to any department.**
* **A division operates many departments, but each department is operated by only one division.**
* **An employee may be assigned many projects, and a project may have many employees assigned to it.**
* **A project must have at least one employee assigned to it.**
* **One of the employees manages each department, and each department is managed by only one employee.**
* **One of the employees runs each division, and each division is run by only one employee.**

# **Part II: SQL (60 points**)

## A. Review Questions (Module Quiz) Chapter 2-4 of the book A Guide to SQL (2015): Odd Review Questions. Copy the question in bold face above the answer.  Do not use bold face for the answers.

### Chapter 2: Odd Numbers: pg. 46-47

1. **What is an entity?**

An entity in the context of databases is a distinct object or concept about which data is stored.

**3. What is a relationship? What is a one-to-many relationship?**

In a database, a relationship is an association between entities. A One-to-Many relationship means that one entity is related to multiple entities.

**5. What is a relation?**

A relation is a two-dimensional table in a relational database.

**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?**

The shorthand representation of the structure of a relational database involves listing each table's name followed by its columns within parentheses. For example, a table named CUSTOMER with columns CustomerID, Name, and Address would be represented as CUSTOMER (CustomerID, Name, Address). This method is crucial because it provides a clear, concise overview of the database's structure, making it easier to understand and communicate the relationships and dependencies between different tables. It simplifies the design and documentation process, ensuring that all stakeholders have a consistent understanding of the database schema. Additionally, this representation aids in identifying primary keys, foreign keys, and other constraints, which are essential for maintaining data integrity and optimizing database performance.

**9. What does it mean for a column to be functionally dependent on another column?**

In a relational database, functional dependence refers to a relationship between two columns. Column B is functionally dependent on column A if, at any given time, a value for column A determines a single value for column B. This means that knowing the value of column A allows you to uniquely identify the value of column B. For example, in a table of sales representatives, if each representative has a unique ID (REP\_ID), then the last name (LAST\_NAME) of the representative is functionally dependent on REP\_ID. Given a specific REP\_ID, you can determine the corresponding LAST\_NAME. This concept is crucial for database design as it helps in organizing data efficiently and ensuring data integrity.

**11. A database at a college must support the following requirements:**

**a. For a department, store its number and name.**

* DEPARTMENT\_NUMBER (Primary Key)
* DEPARTMENT\_NAME

**b. For an advisor, store his or her number, last name, first name, and the department number to which the advisor is assigned.**

* ADVISOR\_NUMBER (Primary Key)
* ADVISOR\_LAST\_NAME
* ADVISOR\_FIRST\_NAME
* ADVISOR\_DEPARTMENT\_NUMBER (Foreign Key referencing Department)

**c. For a course, store its code and description (for example, DBA210, SQL Programming).**

* COURSE\_CODE (Primary Key)
* COURSE\_DESCRIPTION

**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.**

* STUDENTS (STUDENT\_NUMBER, STUDENT\_FIRST\_NAME, STUDENT\_LAST\_NAME, ADVISOR\_NUMBER)
  + STUDENT\_NUMBER (Primary Key)
  + STUDENT\_FIRST\_NAME
  + STUDENT\_LAST\_NAME
  + ADVISOR\_NUMBER
* ENROLLMENT (STUDENT\_NUMBER, COURSE\_CODE, GRADE\_EARNED)
  + STUDENT\_NUMBER (Foreign Key referencing STUDENTS)
  + COURSE\_CODE (Foreign Key referencing COURSES)
  + GRADE\_EARNED

A diagram of a student number

Description automatically generated**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.**

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

Second Normal Form (2NF) is a stage in the normalization process of database design. A table is in 2NF if it is in First Normal Form (1NF) and all non-key columns are fully dependent on the entire primary key, not just part of it. This means that there should be no partial dependency of any column on the primary key.

Problems with tables not in 2NF include:

* Redundancy: Data may be duplicated, wasting storage space.
* Update Anomalies: Changes in data may require multiple updates, increasing the risk of inconsistencies.
* Insertion Anomalies: Adding new data may be problematic if certain fields are missing.
* Deletion Anomalies: Deleting data can inadvertently remove other valuable information.

These issues can lead to inefficiencies and inaccuracies in database management.

**A white rectangular sign with blue text

Description automatically generated15. 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.**

1. Student

* Columns: STUDENT\_NUM (PK), STUDENT\_LAST\_NAME, STUDENT\_FIRST\_NAME, ADVISOR\_NUM

1. Advisor

* Columns: ADVISOR\_NUM (PK), ADVISOR\_LAST\_NAME, ADVISOR\_FIRST\_NAME

1. Course

* Columns: COURSE\_CODE (PK), DESCRIPTION

1. Enrollment

* Columns: STUDENT\_NUM, COURSE\_CODE, GRADE
* Primary Key: Composite (STUDENT\_NUM, COURSE\_CODE)

### Chapter 3: Odd Numbers: pg. 92-93

**1. How do you create a table using SQL?**

To create a table using SQL, you use the CREATE TABLE command1. This command is followed by the table name and a list of columns, each with its specified data type, enclosed within parentheses.

**3. What are common data types used to define columns using SQL?**

In SQL, several common data types are used to define columns in a table. CHAR(n) is used to store a fixed-length character string, where 'n' specifies the length. VARCHAR(n) is similar but stores variable-length strings, using only as much space as needed. DATE stores date values, with formats varying by SQL implementation. DECIMAL(p,q) stores decimal numbers, where 'p' is the total number of digits and 'q' is the number of digits to the right of the decimal point. INT and SMALLINT store integer values, with SMALLINT using less space. These data types help ensure that the data stored in a database is accurate and efficiently managed.

**5. Identify the following column names as valid or invalid in MySQL:**

**a. COMMISSIONRATE -** Valid - This name follows MySQL’s rules for column names.

**b. POSTAL\_CODE\_5CHAR -** Valid - This name is valid as it uses underscores instead of spaces.

**c. SHIP TO ADDRESS -** Invalid - Column names cannot contain spaces.

**d. INVOICE-NUMBER -** Invalid - Column names cannot contain hyphens.

**7. Which SQL command do you use to add a row to a table?**

To add a row to a table in SQL, you use the INSERT INTO command. This command allows you to specify the table into which you want to insert the new row and the values for each column in that row.

**9. Which SQL command do you use to change the value in a column in a table?**

To change the value in a column in a table using SQL, you use the UPDATE command. This command allows you to modify existing data within a table.

**11. How do you display the columns in a table and their characteristics in MySQL?**

To display the columns in a table and their characteristics in MySQL, you can use the DESCRIBE command. This command provides a detailed description of the table’s structure, including the column names, data types, and any constraints such as whether a column can accept null values.

### Chapter 4: Odd Numbers: pg. 131-132

**1. Describe the basic form of the SQL SELECT command.**

The basic form of the SQL SELECT command is structured as SELECT-FROM-WHERE. This command is used to query a database and retrieve specific data. The SELECT clause specifies the columns to be included in the query results. Following this, the FROM clause identifies the table that contains these columns. Optionally, the WHERE clause can be included to list any conditions or restrictions that apply to the data being retrieved. This structure allows users to filter and sort data efficiently, making it a fundamental aspect of SQL querying.

**3. How do you form a compound condition?**

In SQL, a compound condition is created by combining two or more simple conditions using logical operators such as AND, OR, and NOT. For example, to find records where a customer’s balance is greater than 100 and their credit limit is less than 500, you would use: WHERE balance > 100 AND credit\_limit < 500. This ensures that both conditions must be true for a record to be included in the results.

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

A computed column in SQL is a column that is derived from an expression involving other columns in the table. For instance, to calculate the total price of items ordered, you might use: SELECT item\_id, quantity \* price AS total\_price FROM orders. Here, quantity \* price is the computed column, and AS total\_price names the computed column total\_price.

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

MySQL supports several wildcards for pattern matching in queries. The percent sign (%) represents zero or more characters, while the underscore (\_) represents a single character. For example, LIKE 'a%' matches any string starting with ‘a’, and LIKE '\_b%' matches any string with ‘b’ as the second character.

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

To sort data in SQL, you use the ORDER BY clause followed by the column name(s) you want to sort by. By default, the sorting is in ascending order. For example, SELECT \* FROM customers ORDER BY last\_name sorts the customers by their last names in ascending order.

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

To sort data in descending order, you use the ORDER BY clause followed by the column name and the keyword DESC. For example, SELECT \* FROM customers ORDER BY last\_name DESC sorts the customers by their last names in descending order.

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

To avoid including duplicate values in a query’s results, you use the DISTINCT keyword. For example, SELECT DISTINCT last\_name FROM customers retrieves unique last names from the customers table, eliminating any duplicates.

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

To group data in SQL, you use the GROUP BY clause followed by the column name(s) you want to group by. This is often used with aggregate functions like COUNT, SUM, AVG, etc. For example, SELECT department, COUNT(\*) FROM employees GROUP BY department groups the employees by department and counts the number of employees in each department.

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

To find rows where a particular column contains a null value, you use the IS NULL condition. For example, SELECT \* FROM customers WHERE email IS NULL retrieves all customers who do not have an email address listed.

## B. Except for Chapters 1 and 2, please show the complete question, query used, and print the query results.

### Chapter 1 (A Guide to SQL) Use the Staywell student Accommodation database only.

Case Exercises: Odd number Case Exercises, pg. 13.

**A purple and black text

Description automatically generatedA close-up of a purple box

Description automatically generatedA screenshot of a data table

Description automatically generatedAnswer each of the following questions using the StayWell data shown in Figures 1-4 through 1–9. No computer work is required.**

**A screenshot of a service request

Description automatically generated**

**A table of names with numbers

Description automatically generated**

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

* PA101: Patel, Makesh
* AK102: Aksoy, Ceyda
* CO103: Cole, Meerab
* KO104: Kowalczyk, Jakub
* SI105: Sims, Haydon
* BU106: Burke, Ernest
* RE107: Redman, Seth
* LO108: Lopez, Janine
* BI109: Bianchi, Nicole
* JO110: Jones, Ammarah

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

* 3: 9800 Sunbeam Ave., 1,005 square feet
* 5: 887 Vine Rd., 1,125 square feet
* 9: 891 Alton Dr., 1,300 square feet
* 11: 9531 Sherwood Rd., 1,075 square feet

**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.**

* Moore, Elle-May from Reno, NV
* Aksoy, Ceyda from Seattle, WA
* Bianchi, Nicole from New York, NY

**7. How many properties have two floors?**

There are 4 properties with two floors (Properties with IDs: 2, 6, 7, and 8).

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

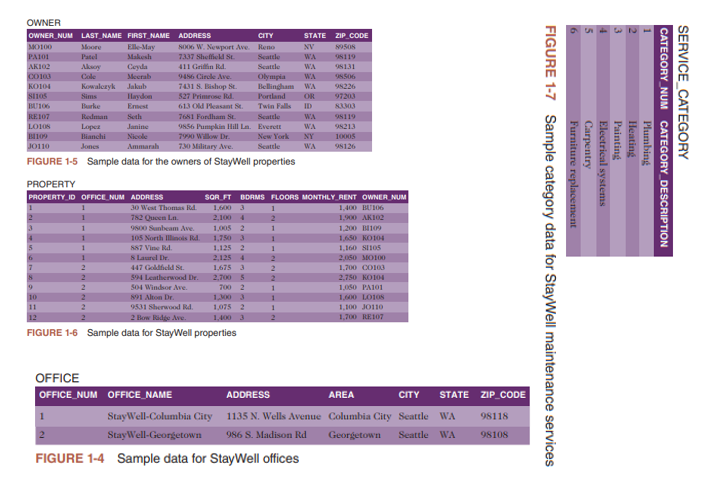
* Sims, Haydon (Property ID: 5)
* Jones, Ammarah (Property ID: 12)

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

* Property ID: 8, Office Number: 2
* Property ID: 12, Office Number: 2

### Chapter 2 (A Guide to SQL) Use the Staywell student Accommodation database only.

Case Exercises: Odd Number Case Exercises, pg. 48-49.



**A screenshot of a service request

Description automatically generated**

**A table of names with numbers

Description automatically generated**

**Answer each of the following questions using the StayWell Student Accommodation data shown in Figures 1-4 through 1-9 in Module 1. No computer work is required.**

**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.**



1. OFFICE

* OFFICE\_NUM (PK): Primary key, unique identifier for each office.
* ADDRESS: Address of the office.
* SQR\_FT: Square footage of the office.
* BDRMS: Number of bedrooms in the office.
* FLOORS: Number of floors in the office.
* MONTHLY\_RENT: Monthly rent price.
* OWNER\_NUM (FK): Foreign key that links to the OWNER table.

1. OWNER

* OWNER\_NUM (PK): Primary key, unique identifier for each owner.
* OWNER\_NAME: Name of the owner.

1. Relationship

* Each OFFICE belongs to one OWNER, with the foreign key in OFFICE referencing the primary key in OWNER.

**3. StayWell also rents out properties on a weekly basis to students attending summer school in the Seattle area. Design a database to meet the following requirements, using the shorthand representation and a diagram of your choice.**

**a. For each student renter, list his or her number, first name, middle initial, last name, address, city, state, postal code, telephone number, and e-mail address.**

Student\_Renter:

* Renter\_ID (PK): Primary key.
* First\_Name
* Middle\_Initial
* Last\_Name
* Address
* City
* State
* Postal\_Code
* Telephone\_Number
* Email

**b. For each property, list the office number, property address, city, state, postal code, square footage, number of bedrooms, number of floors, maximum number of persons that can sleep in the unit, and the base weekly rate.**

Property:

* Property\_ID (PK): Primary key.
* Office\_Num (FK): Foreign key.
* Address
* City
* State
* Postal\_Code
* Sqr\_Ft
* Bedrooms
* Floors
* Max\_Occupancy
* 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.**

Rental Agreement:

* Rental\_ID (PK): Primary key.
* Renter\_ID (FK): Foreign key.
* Property\_ID (FK): Foreign key.
* Start\_Date
* End\_Date
* Weekly\_Rental\_Amount

A screen shot of a computer

Description automatically generated

## For Chapters 3 and 4 please show the screen print out of the Oracle or MySql (among others) output.Use the Staywell student Accommodation database only.

### Chapter 3 (A Guide to SQL) Use the Staywell student Accommodation database only.

Review Questions: Odd Numbers: pgs. 92-93

**1. How do you create a table using SQL?**

To create a table using SQL, you use the CREATE TABLE command. Ex:

CREATE TABLE table\_name (

column1 datatype,

column2 datatype,

...

);

**3. What are common data types used to define columns using SQL?**

* CHAR(n): Stores a fixed-length character string. Useful for columns containing letters, special characters, or numbers not used in calculations.
* VARCHAR(n): Stores a variable-length character string. More space-efficient than CHAR but slower in processing.
* DATE: Stores date data. The format varies by SQL implementation (e.g., ‘YYYY-MM-DD’ in MySQL and SQL Server).
* INT: Stores integers, which are numbers without a decimal part. Suitable for calculations.
* DECIMAL(p,q): Stores a decimal number with a specified number of digits. Useful for precise calculations involving decimal points.

**5. Identify the following column names as valid or invalid in MySQL:**

**a. COMMISSIONRATE** - Valid. It follows the rules for column names, which can include letters and numbers without spaces.

**b. POSTAL\_CODE\_5CHAR** - Valid. It uses underscores to separate words, which is acceptable in MySQL.

**c.** **SHIP TO ADDRESS** - Invalid. Column names cannot contain spaces. You could use underscores instead, like SHIP\_TO\_ADDRESS.

**d. INVOICE-NUMBER** - Invalid. Column names cannot contain hyphens. You could use underscores instead, like INVOICE\_NUMBER.

**7. Which SQL command do you use to add a row to a table?**

To add a row to a table in SQL, you use the INSERT command. Ex:

INSERT INTO table\_name (column1, column2, column3, ...)

VALUES (value1, value2, value3, ...);

**9. Which SQL command do you use to change the value in a column in a table?**

To change the value in a column in a table, you use the UPDATE command. Ex:

UPDATE table\_name

SET column\_name = new\_value

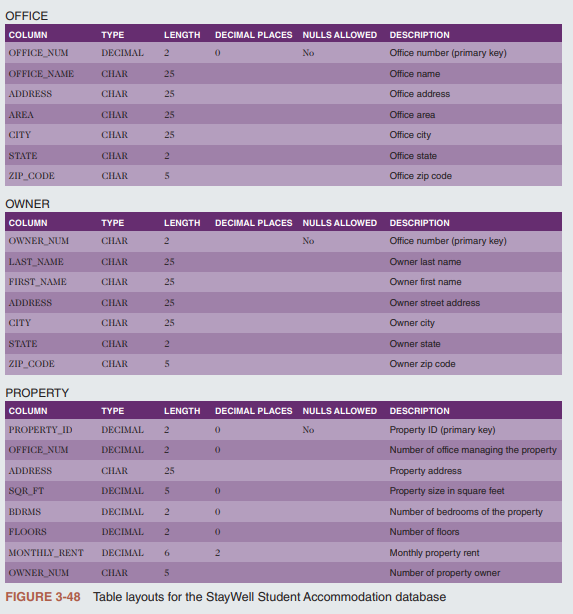
WHERE condition;

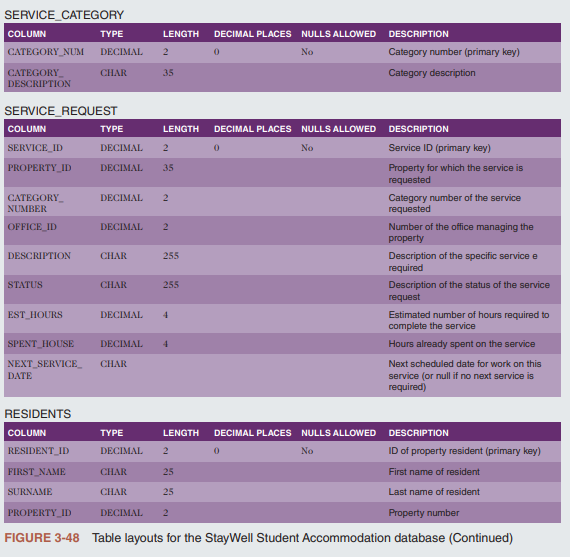
**11. How do you display the columns in a table and their characteristics in MySQL?**

To display the columns in a table and their characteristics in MySQL, you can use the DESCRIBE command. Ex:

DESCRIBE table\_name;

Case Exercises: Odd Numbers: pgs. 94-95.

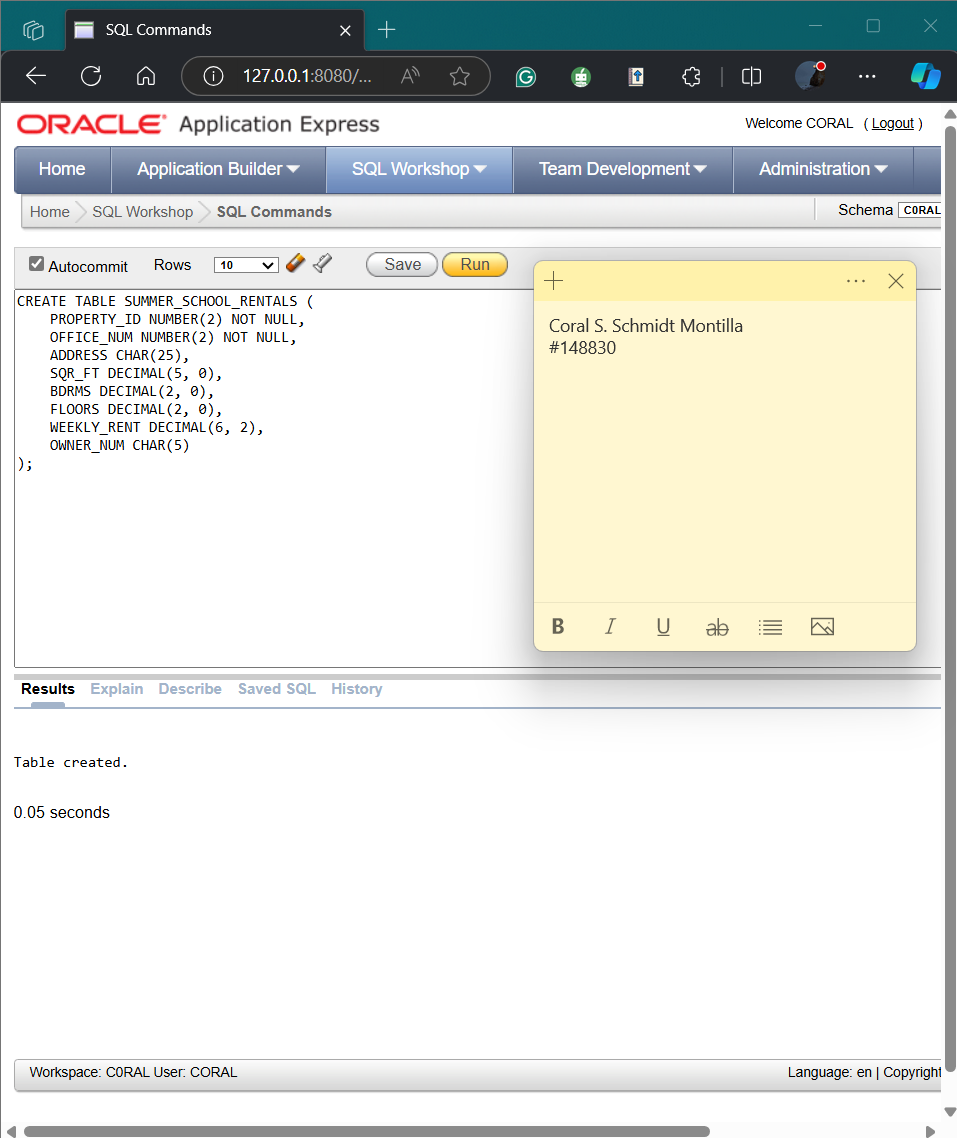




**StayWell Student Accommodation**

**Use SQL to complete the following exercises.**

**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.**



A screenshot of a computer

Description automatically generated

3. Delete the SUMMER\_SCHOOL\_RENTALS table.

A screenshot of a computer

Description automatically generated

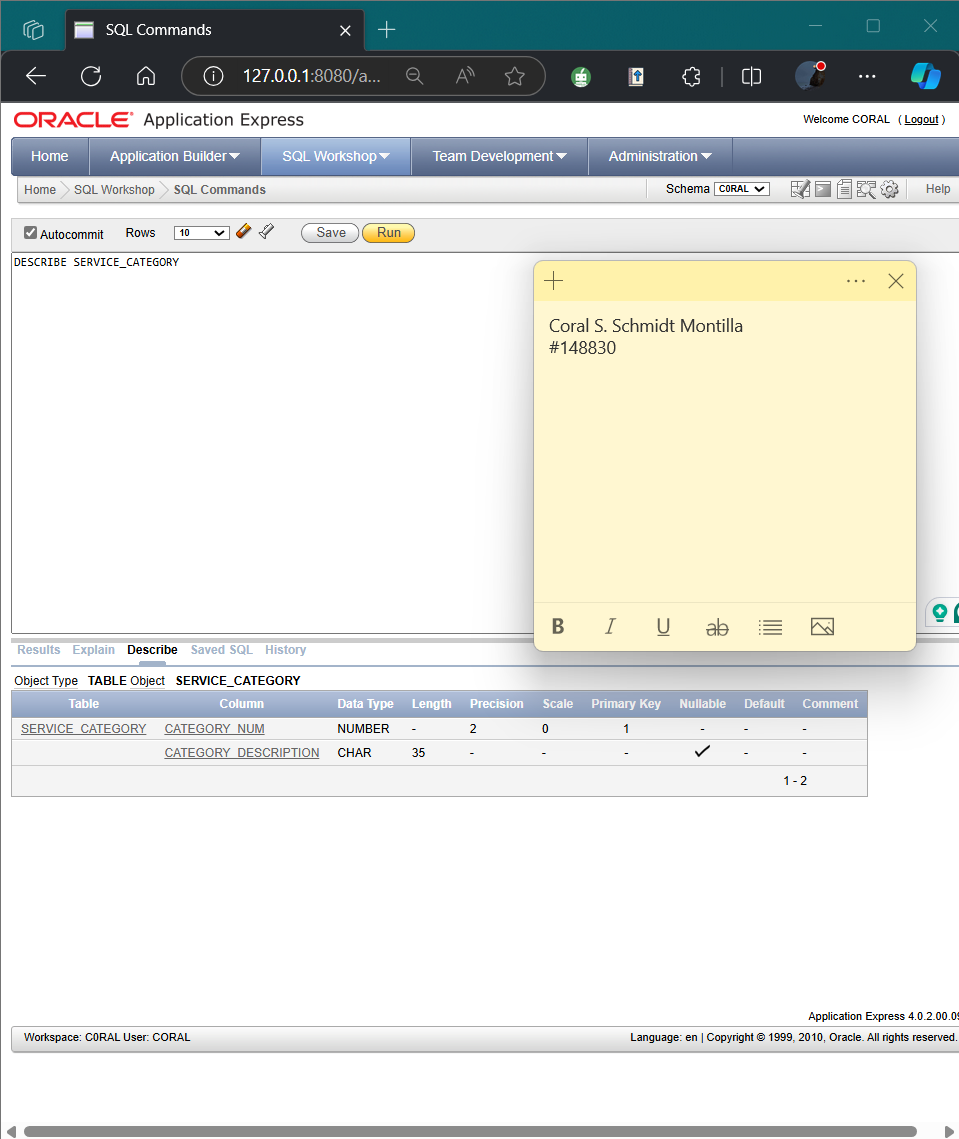
5. Confirm that you have created the tables correctly by describing each table and comparing the results to Figures 3-48.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

### Chapter 4 (A Guide to SQL) Use the Staywell student Accommodation database only.

Review Questions: Odd Numbers: pgs. 101-102

**1. Describe the basic form of the SQL SELECT command.**

The basic form of the SQL SELECT command is SELECT-FROM-WHERE:12

* SELECT: Specify the columns you want to retrieve.
* FROM: Indicate the table containing these columns.
* WHERE: (Optional) List any conditions to filter the data.

Ex:

SELECT column1, column2

FROM table\_name

WHERE condition;

**3. How do you form a compound condition?**

To form a compound condition in SQL, you connect two or more simple conditions using the AND, OR, and NOT operators:

* AND: All simple conditions must be true for the compound condition to be true.
* OR: The compound condition is true if any one of the simple conditions is true.
* NOT: Reverses the truth of the original condition.

Ex:

SELECT DESCRIPTION

FROM ITEM

WHERE (LOCATION = 'B') AND (ON\_HAND > 15);[^5^][5][^6^][6]

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

Using a Computed Column in SQL:

* Performing Computations: 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. For example, to compute available credit, you can use the expression `CREDIT\_LIMIT - BALANCE`.

Naming a Computed Column:

* Assigning a Name: You can assign a name, or alias, to a computed column by following the computation with the word `AS` and the desired name. For example, `CREDIT\_LIMIT - BALANCE AS AVAILABLE\_CREDIT`.

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

In MySQL, there are two main wildcard characters used in SQL queries:

* Percent Sign (%): Represents any collection of characters. For example, `LIKE '%Rock%'` matches any string containing "Rock".
* Underscore (\_): Represents any single character. For example, `LIKE 'T\_m'` matches "Tim", "Tom", or "T3m".

These wildcards are typically used with the `LIKE` operator to search for patterns within text columns.

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

To sort data in SQL, you use the ORDER BY clause. Here are the key points:

* Basic Sorting: Use ORDER BY column\_name to sort data in ascending order by default.
* Descending Order: Add DESC after the column name to sort in descending order, e.g., ORDER BY column\_name DESC.
* Multiple Keys: List multiple columns separated by commas to sort by multiple keys, e.g., ORDER BY column1, column2 DESC.
* Major and Minor Keys: The first column listed is the major sort key, and the subsequent columns are minor sort keys.

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

To sort data in descending order in SQL, you use the ORDER BY clause followed by the column name and the DESC keyword. Ex:

SELECT column\_name

FROM table\_name

ORDER BY column\_name DESC;

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

To avoid including duplicate values in a query’s results, you can use the DISTINCT operator in your SQL command12. This operator ensures that only unique values are returned in the query results. Ex:

SELECT DISTINCT column\_name

FROM table\_name;

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

To group data in an SQL query, you use the GROUP BY clause. This clause allows you to group rows that have the same values in specified columns. Ex:

SELECT column\_name, COUNT(\*)

FROM table\_name

GROUP BY column\_name;

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

To find rows where a particular column contains a null value in SQL, you can use the IS NULL operator in a WHERE clause. Ex:

SELECT column1, column2, ...

FROM table\_name

WHERE column\_name IS NULL;

A table of names with numbers

Description automatically generatedA screenshot of a table

Description automatically generatedCase Exercises: Odd Numbers: pgs. 133-134.A close-up of a address

Description automatically generated

A purple and white rectangular box with text

Description automatically generated

A screenshot of a service request

Description automatically generated

**Use SQL and the StayWell Student Accommodation database (Figures 1-4 through 1-9 in Module 1) to complete the following exercises. If directed to do so by your instructor, use the information provided with the Module 3 Exercises to print your output or save it to a document.**

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

A screenshot of a computer

Description automatically generated

3. List the last name and first name of every owner who lives in Seattle.

A screenshot of a computer

Description automatically generated

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

A screenshot of a computer

Description automatically generated

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

A screenshot of a computer

Description automatically generated

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

A screenshot of a computer

Description automatically generated

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

A screenshot of a computer

Description automatically generated

13. How many three-bedroom properties are managed by each office?

A screenshot of a computer

Description automatically generated