SQL

SQL - Part 24 - concat() MySQL String Function

1. **Overview**:
   - The concat() function in MySQL is used to concatenate two or more strings together into a single string. It's a versatile function for combining strings from various columns or expressions.
2. **Example**:
   - Suppose you have two columns, first_name and last_name, in a users table, and you want to combine them into a full name.

Code Example:
sql
Copy code

```
SELECT CONCAT(first_name, ' ', last_name) AS full_name
FROM users;
```

3.
4. **Explanation**:
   - In this example, the concat() function combines first_name and last_name with a space in between. The result is aliased as full_name, which can be used in further queries or displayed in results.
5. **Summary**:
   - The concat() function is essential for string operations in MySQL, enabling you to merge multiple strings into one seamlessly.

https://drive.google.com/file/d/1J3LWLwtdz6LILNS41RQAFezYJmSjOlTk/preview

---

SQL - Part 25 - trim() MySQL String Function

1. **Overview**:
   - The trim() function in MySQL is used to remove unwanted leading and trailing spaces from a string. It's particularly useful for cleaning up data before processing.
2. **Example**:
   - Suppose you have a column username in a table that contains spaces at the beginning or end, and you want to remove these spaces.

Code Example:
sql

Copy code
```
SELECT TRIM(username) AS cleaned_username
FROM users;
```

3.
4. **Explanation**:
   - The trim() function takes the username field and removes any spaces before or after the string. This is especially helpful when comparing or displaying data.
5. **Summary**:
   - The trim() function is a simple yet powerful tool for ensuring data cleanliness by removing unnecessary spaces from strings.

https://drive.google.com/file/d/1OwO8aW8h7LS01nUeR1tY78YLwtmHMo1h/preview

---

SQL - Part 26 - abs() MySQL Numeric Function

1. **Overview**:
   - The abs() function in MySQL returns the absolute value of a number. It's commonly used to ensure non-negative results from calculations.
2. **Example**:
   - If you have a column profit_loss that contains both positive and negative values, you might want to find the magnitude of the values.

Code Example:
sql
Copy code
```
SELECT ABS(profit_loss) AS magnitude
FROM financials;
```

3.
4. **Explanation**:
   - The abs() function converts any negative values in the profit_loss column to their positive counterparts, providing a measure of the magnitude of losses or profits.
5. **Summary**:
   - The abs() function is crucial for handling numeric data, ensuring that you can work with absolute values regardless of the original sign.

https://drive.google.com/file/d/1H6U_IfmSXnQL8v66ydqcPs0uy3C6sgr9/preview

SQL - Part 27 - mod() MySQL Numeric Function

1. **Overview**:
   - The mod() function in MySQL returns the remainder of a division operation between two numbers. It's useful for tasks like determining even or odd numbers, cycles, and more.
2. **Example**:
   - If you have a column quantity and want to find out which entries represent even numbers, you can use the mod() function.

Code Example:
sql
Copy code

```
SELECT quantity, MOD(quantity, 2) AS remainder
FROM products;
```

3.
4. **Explanation**:
   - The mod() function divides quantity by 2 and returns the remainder. A remainder of 0 indicates that the number is even, while a remainder of 1 indicates it's odd.
5. **Summary**:
   - The mod() function is a handy tool for division operations where the remainder is significant, helping you perform various numeric checks and calculations.

https://drive.google.com/file/d/1c0wtc2DanMNoR7TNleNT4H7OaAddPwAH/preview

SQL - Part 28 - greatest() and least() MySQL Numeric Functions

1. **Overview**:
   - The greatest() and least() functions in MySQL are used to find the largest and smallest values from a list of arguments, respectively. They are valuable for comparisons across multiple columns or values.
2. **Example**:
   - Suppose you have columns score1, score2, and score3 in a results table, and you want to find the highest and lowest scores for each row.

Code Example:
sql

Copy code
```
SELECT GREATEST(score1, score2, score3) AS highest_score,
       LEAST(score1, score2, score3) AS lowest_score
FROM results;
```

3.
4. **Explanation**:
    - The greatest() function returns the highest value among score1, score2, and score3, while the least() function returns the lowest. These results are then aliased as highest_score and lowest_score.
5. **Summary**:
    - The greatest() and least() functions are powerful for comparative analysis across multiple values, making them essential in scenarios where you need to determine the extreme values in a dataset.

https://drive.google.com/file/d/1QCcP47KYtV0q_5Q9RQCKreeeRstCROzO/preview

---

SQL - Part 29 - truncate() MySQL Numeric Function

1. **Overview**:
    - The truncate() function in MySQL is used to shorten a number to a specified number of decimal places without rounding. It's useful for controlling the precision of numeric outputs.
2. **Example**:
    - If you have a column price and want to limit the number of decimal places to two for display purposes.

Code Example:
sql
Copy code
```
SELECT TRUNCATE(price, 2) AS truncated_price
FROM products;
```

3.
4. **Explanation**:
    - The truncate() function cuts off the number of decimal places in the price field to two, without rounding the value. This is helpful for ensuring consistent numeric formatting.
5. **Summary**:

- The truncate() function is essential for managing numeric precision, particularly when you need to present data in a specific format without altering the actual values.

https://drive.google.com/file/d/18TKFQWXe31c8dXAIfxYe1kqzRMZGIMec/preview

---

SQL - Part 30 - power() and sqrt() MySQL Numeric Functions

1. **Overview**:
   - The power() function in MySQL raises a number to the power of another number, while the sqrt() function returns the square root of a number. Both are fundamental in mathematical operations.
2. **Example**:
   - Suppose you want to calculate the square of a value in a measurements table and also find the square root of another value.

Code Example:
sql
Copy code
```
SELECT POWER(value, 2) AS squared_value,
       SQRT(value) AS square_root_value
FROM measurements;
```

3.
4. **Explanation**:
   - The power() function raises value to the power of 2, effectively squaring it. The sqrt() function finds the square root of value. These results provide both the squared and square root values for further analysis.
5. **Summary**:
   - The power() and sqrt() functions are key for performing mathematical operations in MySQL, enabling complex calculations directly within your queries

https://drive.google.com/file/d/1znnoMcBeeX5052UX6rwC6EeQxxR6Xr9L/preview

—----------------------------

SQL - Part 31 - current_date(), curdate(), current_time(), curtime(), now(), and sysdate() MySQL Date Time Functions

1. **Overview**:

- MySQL provides various functions to retrieve the current date and time. These include current_date(), curdate(), current_time(), curtime(), now(), and sysdate(). Each of these functions serves a slightly different purpose and is used to handle date and time data effectively.

2. **Example**:
   - You want to log the current date and time when a new record is added to a logs table.

Code Example:
sql
Copy code
```
INSERT INTO logs (log_date, log_time, full_timestamp)
VALUES (CURRENT_DATE(), CURRENT_TIME(), NOW());
```

3.
4. **Explanation**:
   - current_date() and curdate() both return the current date in YYYY-MM-DD format.
   - current_time() and curtime() return the current time in HH:MM:SS format.
   - now() returns the current date and time.
   - sysdate() is similar to now(), but it returns the actual time of execution, which can differ from now() if there is a delay between invocation and execution.
5. **Summary**:
   - These MySQL date and time functions are essential for managing time-sensitive data, allowing precise logging, scheduling, and time-based operations.

https://drive.google.com/file/d/1cau6hxjW8OOWMJEc3JxT1cnltO7YtZJJ/preview

---

SQL - Part 32 - year(), month(), day(), monthname(), dayname() MySQL Date Time Functions

1. **Overview**:
   - MySQL provides several functions to extract specific components from date values, such as the year, month, day, month name, and day name. These functions are invaluable for analyzing and reporting date-based data.
2. **Example**:
   - Suppose you need to generate a report showing the year, month, and day for orders placed in an orders table.

Code Example:
sql

Copy code

```
SELECT YEAR(order_date) AS order_year,
      MONTH(order_date) AS order_month,
      DAY(order_date) AS order_day,
      MONTHNAME(order_date) AS order_month_name,
      DAYNAME(order_date) AS order_day_name
FROM orders;
```

3.
4. **Explanation**:
    - year(), month(), and day() extract the year, month, and day from a date value, respectively.
    - monthname() returns the full name of the month.
    - dayname() returns the name of the day of the week.
    - These functions allow for detailed analysis and reporting of date data, enabling insights into trends based on time periods.
5. **Summary**:
    - MySQL's date extraction functions are crucial for breaking down and analyzing date values, making them essential for generating detailed reports and conducting time-based analyses.

https://drive.google.com/file/d/1Tr4E7h9nZmycj-v_hCktYCfMcWeAZG3_/preview

---

SQL - Part 33 - avg(), max(), min(), count(), and sum() MySQL Aggregate Functions

1. **Overview**:
    - MySQL aggregate functions are used to perform calculations on a set of values and return a single value. Commonly used aggregate functions include avg() for average, max() for maximum, min() for minimum, count() for counting rows, and sum() for summing values.
2. **Example**:
    - Imagine you have a sales table and want to calculate the average sale amount, the total number of sales, and the sum of all sales.

Code Example:
sql
Copy code

```
SELECT AVG(amount) AS average_sale,
      MAX(amount) AS max_sale,
      MIN(amount) AS min_sale,
      COUNT(*) AS total_sales,
      SUM(amount) AS total_revenue
FROM sales;
```

3.
4. **Explanation**:
    - avg() calculates the average of a numeric column.
    - max() finds the maximum value in a column.
    - min() finds the minimum value in a column.
    - count() counts the number of rows in a table or those meeting certain conditions.
    - sum() adds up all the values in a numeric column.
    - These functions are critical for summarizing large datasets and providing key metrics for business decision-making.
5. **Summary**:
    - Aggregate functions in MySQL are powerful tools for data analysis, allowing you to compute summaries and key statistics from your data with ease.

https://drive.google.com/file/d/1rYe7drZVP_aN-lwWpP7gsUOQW6C-tQJq/preview

---

SQL - Part 34 - Arithmetic Operators

1. **Overview**:
    - Arithmetic operators in MySQL are used to perform mathematical operations on numeric data. These include + (addition), - (subtraction), * (multiplication), / (division), and % (modulus).
2. **Example**:
    - Suppose you have a products table and want to calculate the total cost of a product by multiplying the price by the quantity.

Code Example:
sql
Copy code

```
SELECT product_name,
    price,
    quantity,
    (price * quantity) AS total_cost
FROM products;
```

3.

4. **Explanation**:
   - + adds two values.
   - - subtracts one value from another.
   - * multiplies two values.
   - / divides one value by another.
   - % returns the remainder of a division operation.
   - Arithmetic operators are fundamental to data manipulation, allowing for a wide range of calculations directly within SQL queries.
5. **Summary**:
   - MySQL arithmetic operators are essential for performing basic mathematical operations, making them indispensable for calculations within your SQL queries.

https://drive.google.com/file/d/1qwXh2FtY-fK0St2CZSXdaQTEIvJnPjWw/preview

SQL - Part 35 - SQL for Beginners: Installing MySQL Server and Workbench Client for Practicing SQL (Latest Way)

1. **Overview**:
   - This part covers the latest methods for installing MySQL Server and Workbench Client on your system, which are essential tools for practicing SQL. MySQL Server is the database engine, while Workbench is a graphical user interface (GUI) that allows you to manage databases, write SQL queries, and more.
2. **Example**:
   - Walkthrough of downloading MySQL Server and Workbench from the official MySQL website.
3. **Code Example**:
   - N/A (This part involves software installation, so code examples are not applicable.)
4. **Explanation**:
   - Detailed instructions on how to download, install, and configure MySQL Server and Workbench, including choosing the appropriate version for your operating system and setting up the initial configurations, such as the root password.
5. **Summary**:
   - Installing MySQL Server and Workbench Client is the first step in setting up your environment for SQL practice. Following the latest installation methods ensures compatibility and access to the newest features.

https://drive.google.com/file/d/1LM08TR5ZYnvb_DnJp9NSgh9c54SHPKD8/preview

---

SQL - Part 36 - Installing MySQL Server and Workbench Client for Practicing SQL

1. **Overview**:
   - This part provides an alternative method for installing MySQL Server and Workbench Client, offering a step-by-step guide that might be preferred by some users based on their system requirements or personal preferences.
2. **Example**:
   - Example of using a package manager (e.g., Homebrew for macOS, apt-get for Linux) to install MySQL Server and Workbench.
3. **Code Example**:
   - N/A (This part involves software installation, so code examples are not applicable.)
4. **Explanation**:
   - Instructions on using different installation methods, such as package managers or installers, and any additional configurations required, such as setting up environment variables or network access.
5. **Summary**:
   - Understanding different methods of installing MySQL Server and Workbench Client allows you to choose the best approach based on your specific environment and needs.

https://drive.google.com/file/d/1YQ_oKc2SZj294CYBwkdc4kyqSQOMKmKR/preview

---

SQL - Part 37 - Creating, Deleting, Viewing, and Using Databases

1. **Overview**:
   - This part covers the essential SQL commands for managing databases, including creating new databases, deleting existing ones, viewing available databases, and selecting a database to use for your SQL queries.
2. **Example**:
   - Example of creating a new database named test_db, selecting it for use, and then deleting it.

Code Example:
sql

Copy code
```
-- Creating a new database
CREATE DATABASE test_db;

-- Viewing all databases
SHOW DATABASES;

-- Selecting a database to use
USE test_db;

-- Deleting a database
DROP DATABASE test_db;
```

3.
4. **Explanation**:
   - `CREATE DATABASE` is used to create a new database.
   - `SHOW DATABASES` lists all the databases on the MySQL server.
   - `USE` selects a database to work with in your session.
   - `DROP DATABASE` deletes an existing database.
   - These commands are fundamental for database management and are often the first steps in any SQL project.
5. **Summary**:
   - Mastering the basic commands for database management in MySQL is crucial for any SQL practitioner, enabling you to effectively create, manage, and organize your databases.

https://drive.google.com/file/d/12t2y_WImeKnW8W81Jvtmcw9xUj9ZiXl7/preview

---

SQL - Part 38 - Creating, Viewing, Describing, and Deleting Tables

1. **Overview**:
   - This part delves into the SQL commands necessary for creating tables within a database, viewing their structure, describing their columns and data types, and deleting them when they are no longer needed.
2. **Example**:
   - Example of creating a table named `employees`, viewing its structure, describing its columns, and then deleting the table.

Code Example:
sql

Copy code

```
-- Creating a new table
CREATE TABLE employees (
    id INT AUTO_INCREMENT PRIMARY KEY,
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    hire_date DATE
);

-- Viewing all tables in the current database
SHOW TABLES;

-- Describing the structure of a table
DESCRIBE employees;

-- Deleting a table
DROP TABLE employees;
```

3.
4. **Explanation**:
    - CREATE TABLE defines a new table with specified columns and data types.
    - SHOW TABLES lists all tables in the current database.
    - DESCRIBE provides details about the columns, including their data types and constraints.
    - DROP TABLE removes a table from the database.
    - Understanding these commands is essential for managing the structure of your database and organizing data effectively.
5. **Summary**:
    - Knowing how to create, view, describe, and delete tables in MySQL allows you to manage the structure and integrity of your database, forming the foundation for more complex SQL operations.

https://drive.google.com/file/d/1ONLhc9_8UxF462S_DVe4ScZzvruOHmmQ/preview

---

SQL - Part 39 - Insert Into Statements (For Inserting Data into Tables)

1. **Overview**:
    - This part focuses on the INSERT INTO statement, which is used to add new rows of data to a table. It's one of the most frequently used commands in SQL for populating tables with data.

2. **Example**:
    - Example of inserting a new employee into the employees table.

Code Example:
sql
Copy code

```
-- Inserting a single row into the table
INSERT INTO employees (first_name, last_name, hire_date)
VALUES ('John', 'Doe', '2024-09-02');

-- Inserting multiple rows into the table
INSERT INTO employees (first_name, last_name, hire_date)
VALUES
('Jane', 'Smith', '2024-08-15'),
('Emily', 'Jones', '2024-07-20');
```

3.
4. **Explanation**:
    - The INSERT INTO statement adds new rows of data to a table, specifying the columns and the values to be inserted.
    - Multiple rows can be inserted in a single query by separating each set of values with a comma.
    - This command is essential for populating your tables with data, allowing for further analysis and query operations.
5. **Summary**:
    - The INSERT INTO statement is a fundamental SQL command that enables you to populate tables with data, making it a critical skill for anyone working with databases.

https://drive.google.com/file/d/167RregfeAYbBM44ZNvPQQxzqHvYTX6Nj/preview

SQL - Part 40 - Data Types

1. **Overview**:
    - Data types in SQL define the kind of data that can be stored in a column. MySQL supports various data types, including numeric, string, date/time, and spatial types. Understanding these data types is crucial for designing efficient databases and ensuring data integrity.
2. **Example**:
    - Example of defining a table with different data types for each column.

Code Example:
sql
Copy code

```
CREATE TABLE products (
    product_id INT AUTO_INCREMENT PRIMARY KEY,
    product_name VARCHAR(100),
    price DECIMAL(10, 2),
    release_date DATE
);
```

3.
4. **Explanation**:
   - INT is a numeric data type used for integers.
   - VARCHAR is a string data type with a specified maximum length.
   - DECIMAL is used for precise decimal values, often used in financial calculations.
   - DATE stores date values in YYYY-MM-DD format.
   - Choosing the correct data type is essential for optimizing storage and ensuring that the data stored is appropriate for the operations you intend to perform.
5. **Summary**:
   - Understanding and correctly using data types in MySQL is foundational for database design, ensuring data accuracy and performance efficiency.

https://drive.google.com/file/d/1VUr6Dzv0Sfy_W3yzuSTLseOacrzUPVKk/preview

---

SQL - Part 41 - Null Value, Is Null Operator, and Is Not Null Operator

1. **Overview**:
   - In SQL, NULL represents a missing or undefined value. The IS NULL and IS NOT NULL operators are used to check for the presence or absence of NULL values in a column. Handling NULL values correctly is important for accurate data analysis.
2. **Example**:
   - Example of querying a table to find records with NULL or non-NULL values in a specific column.

Code Example:
sql

Copy code
```sql
-- Finding rows with a NULL value in the last_name column
SELECT * FROM employees WHERE last_name IS NULL;

-- Finding rows where the last_name column is not NULL
SELECT * FROM employees WHERE last_name IS NOT NULL;
```

3.
4. **Explanation**:
    - NULL is used to represent missing data in a database.
    - IS NULL checks if a column's value is NULL.
    - IS NOT NULL checks if a column's value is not NULL.
    - Proper handling of NULL values ensures that queries return accurate results, especially when filtering or aggregating data.
5. **Summary**:
    - Dealing with NULL values using the IS NULL and IS NOT NULL operators is crucial for maintaining data accuracy and consistency in SQL operations.

https://drive.google.com/file/d/1qOsECMNo62exiIC2gog9SPQfuCNQgzVx/preview

---

SQL - Part 42 - Delete Statement (For Deleting the Records from Table)

1. **Overview**:
    - The DELETE statement in SQL is used to remove one or more records from a table. Care must be taken when using DELETE, especially without a WHERE clause, as it can result in the removal of all records from a table.
2. **Example**:
    - Example of deleting a specific employee from the employees table based on their ID.

Code Example:
sql
Copy code
```sql
-- Deleting a single record by ID
DELETE FROM employees WHERE id = 10;

-- Deleting all records from the table
DELETE FROM employees;
```

3.

4. **Explanation**:
    - DELETE FROM specifies the table from which records are to be deleted.
    - The WHERE clause is used to specify the condition for which records to delete. Without a WHERE clause, all records in the table will be deleted.
    - Understanding the impact of the DELETE statement is critical to avoid accidental data loss.
5. **Summary**:
    - The DELETE statement is a powerful tool for removing records from a table, but it must be used with caution to prevent unintentional data loss.

https://drive.google.com/file/d/17RtCXshZKeiZa5n04sD6fuw7TRlgYRp7/preview

---

SQL - Part 43 - Update Statement and Set Keyword (For Updating the Table Records)

1. **Overview**:
    - The UPDATE statement is used to modify existing records in a table. The SET keyword specifies the columns and new values to be updated. Using the WHERE clause is crucial to ensure only specific records are updated.
2. **Example**:
    - Example of updating an employee's last name in the employees table based on their ID.

Code Example:
sql
Copy code
```
-- Updating the last name of an employee
UPDATE employees
SET last_name = 'Smith'
WHERE id = 10;
```

3.
4. **Explanation**:
    - UPDATE specifies the table to update.
    - SET specifies the column(s) to be updated and the new value(s).
    - The WHERE clause limits the update to specific records. Without it, all records will be updated, which can be dangerous.
    - Mastery of the UPDATE statement is essential for maintaining and modifying data in a database.
5. **Summary**:

- The UPDATE statement, combined with the SET keyword, allows for precise modifications of data in a table. Proper use of the WHERE clause is crucial to avoid unintended updates.

https://drive.google.com/file/d/1vP-rBhdLJllrywJsrhcMAZWQK0z7cTd6/preview

---

SQL - Part 44 - Rename Statement and To Keyword (For Renaming Table Name)

1. **Overview**:
   - The RENAME statement is used to change the name of an existing table in a database. This can be useful when the table's purpose changes, or a more descriptive name is needed.
2. **Example**:
   - Example of renaming a table from employees to staff.

Code Example:
sql
Copy code

```
-- Renaming the table employees to staff
RENAME TABLE employees TO staff;
```

3.
4. **Explanation**:
   - RENAME TABLE specifies the table to rename.
   - The TO keyword is used to specify the new name of the table.
   - Renaming tables can be important for maintaining clarity and consistency in your database schema as the project evolves.
5. **Summary**:
   - The RENAME statement is a straightforward way to change table names, helping to keep your database structure clear and organized as needs change.

https://drive.google.com/file/d/1HayxXVshIVlfrpqIk9ajJ0YN0mgzMjD3/preview

---

SQL - Part 45 - Alter Statement, Add, Modify Column, Rename Column, and Drop Column

1. **Overview**:

- The ALTER statement is a versatile command used to modify the structure of an existing table. It can add, modify, rename, or drop columns, allowing for dynamic adjustments to a table's design as requirements change.

2. **Example**:
   - Example of adding a new column, modifying an existing column, renaming a column, and dropping a column in the staff table.

Code Example:
sql
Copy code

```sql
-- Adding a new column
ALTER TABLE staff ADD COLUMN department VARCHAR(50);

-- Modifying an existing column
ALTER TABLE staff MODIFY COLUMN hire_date DATETIME;

-- Renaming a column
ALTER TABLE staff RENAME COLUMN last_name TO surname;

-- Dropping a column
ALTER TABLE staff DROP COLUMN department;
```

3.
4. **Explanation**:
   - ALTER TABLE specifies the table to be altered.
   - ADD COLUMN adds a new column to the table.
   - MODIFY COLUMN changes the data type or attributes of an existing column.
   - RENAME COLUMN changes the name of a column.
   - DROP COLUMN removes a column from the table.
   - These operations are essential for adapting the table structure to evolving data needs.
5. **Summary**:
   - The ALTER statement provides powerful tools for modifying table structures, making it critical for database maintenance and updates.

https://drive.google.com/file/d/1NtLyoujBXZHznegZv1gSC0mKRQE6-W0m/preview

---

SQL - Part 46 - Set Autocommit

1. **Overview**:

- The SET AUTOCOMMIT statement in SQL is used to control the autocommit mode of a session. In autocommit mode, each SQL statement is committed to the database as soon as it is executed. Turning off autocommit allows for transactions to be managed manually.

2. **Example**:
   - Example of turning off autocommit mode to manually manage transactions.

Code Example:
sql
Copy code

```
-- Turning off autocommit
SET AUTOCOMMIT = 0;
```

3.
4. **Explanation**:
   - SET AUTOCOMMIT = 0 disables autocommit, requiring explicit COMMIT or ROLLBACK statements to finalize or undo transactions.
   - When AUTOCOMMIT is enabled (the default setting), changes are committed automatically after each statement.
   - Controlling autocommit is important for managing transactions, ensuring data integrity during complex operations.
5. **Summary**:
   - Understanding how to control autocommit mode is essential for effective transaction management in SQL, particularly in scenarios requiring multiple related operations.

https://drive.google.com/file/d/1aDG0G3Z5Wq4HFcFR_SxKDCeuZPBf5TVx/preview

SQL - Part 47 - Commit Statement

1. **Overview**:
   - The COMMIT statement in SQL is used to save all the changes made during the current transaction. When you execute a COMMIT, all changes made to the database during the transaction are permanently applied. This ensures data integrity and consistency in database operations.
2. **Example**:
   - Example of using COMMIT to save changes after inserting multiple rows into a table.

Code Example:
sql

Copy code

```
-- Start of a transaction
START TRANSACTION;

-- Insert statements
INSERT INTO staff (first_name, surname, hire_date)
VALUES ('Alice', 'Johnson', '2024-09-01'),
       ('Bob', 'Williams', '2024-09-02');

-- Commit the transaction
COMMIT;
```

3.
4. **Explanation**:
    - START TRANSACTION initiates a new transaction.
    - The INSERT INTO statements add new rows to the staff table.
    - COMMIT finalizes the transaction, making the inserted data permanent.
    - Using COMMIT is crucial for ensuring that all operations within a transaction are applied together, maintaining data consistency.
5. **Summary**:
    - The COMMIT statement is essential for making transaction changes permanent in a database, ensuring data integrity and consistency.

https://drive.google.com/file/d/14QSn-tLtuCj_i49ns8KCfIIcS8aYFgKv/preview

---

SQL - Part 48 - Rollback Statement

1. **Overview**:
    - The ROLLBACK statement is used to undo changes made during the current transaction. If an error occurs or if you decide not to proceed with the changes, ROLLBACK reverts the database to its state before the transaction began.
2. **Example**:
    - Example of using ROLLBACK to undo changes if an error occurs during a transaction.

Code Example:
sql
Copy code

```
-- Start of a transaction
START TRANSACTION;
```

```
-- Insert statements
INSERT INTO staff (first_name, surname, hire_date)
VALUES ('Carol', 'Smith', '2024-09-03');

-- If something goes wrong, rollback the transaction
ROLLBACK;
```

3.
4. **Explanation**:
   - START TRANSACTION initiates a new transaction.
   - The INSERT INTO statement attempts to add a new row to the staff table.
   - ROLLBACK reverts any changes made during the transaction, ensuring the database remains in a consistent state.
   - Using ROLLBACK is critical for handling errors and ensuring that incomplete or incorrect transactions do not affect the database.
5. **Summary**:
   - The ROLLBACK statement is crucial for undoing changes in a transaction, providing a way to handle errors and maintain database integrity.

https://drive.google.com/file/d/1Gp4MxNeBg0LWtxzSgFWeAtgwcMePrXrL/preview

MCQ

Topic: concat() MySQL String Function

Question 1:
What is the primary use of the concat() function in MySQL?
A) To find the length of a string
B) To concatenate two or more strings into one
C) To replace characters in a string
D) To split a string into multiple parts

Answer: B) To concatenate two or more strings into one

Topic: trim() MySQL String Function

Question 2:
Which of the following is removed by default when using the trim() function in MySQL?
A) Commas
B) Numbers
C) Leading and trailing spaces
D) Special characters

Answer: C) Leading and trailing spaces

Topic: abs() MySQL Numeric Function

Question 3:
What does the abs() function return in MySQL?
A) The sum of a number
B) The absolute value of a number
C) The factorial of a number
D) The rounded value of a number

Answer: B) The absolute value of a number

Topic: mod() MySQL Numeric Function

Question 4:
What does the mod() function in MySQL return?
A) The remainder of a division
B) The quotient of a division
C) The largest number
D) The smallest number

Answer: A) The remainder of a division

Topic: greatest() and least() MySQL Numeric Functions

Question 5:
Which of the following correctly describes the greatest() function in MySQL?
A) It returns the largest value from a list of values

B) It returns the smallest value from a list of values
C) It returns the sum of the list of values
D) It returns the product of the list of values

Answer: A) It returns the largest value from a list of values

---

Topic: truncate() MySQL Numeric Function

Question 6:
The truncate() function in MySQL is used to:
A) Round a number to the nearest whole number
B) Remove decimal places from a number
C) Increase a number's precision
D) Multiply a number by a fixed factor

Answer: B) Remove decimal places from a number

Topic: power() and sqrt() MySQL Numeric Functions

Question 7:
What is the result of the power(3, 4) function in MySQL?
A) 7
B) 12
C) 64
D) 81

Answer: D) 81

Topic: current_date(), curdate(), current_time(), curtime(), now(), sysdate()

Question 8:
Which function in MySQL returns the current date and time?
A) current_date()
B) curtime()
C) now()
D) sysdate()

Answer: C) now()

Topic: year(), month(), day(), monthname(), dayname() MySQL Date Time Functions

Question 9:
What does the monthname() function in MySQL return?
A) The number of the month
B) The full name of the month
C) The number of days in the month
D) The first three letters of the month

Answer: B) The full name of the month

Topic: avg(), max(), min(), count() and sum() MySQL Aggregate Functions

Question 10:
Which MySQL aggregate function is used to count the number of rows in a table?
A) sum()
B) count()
C) avg()
D) max()

Answer: B) count()