SQL Notes

Pro Tips for MySQL Workbench

Export/Import

Use the "Data Export" and "Data Import" tabs under the "Server" menu for backups or restoring data.

ER Diagrams

Navigate to Database > Reverse Engineer to visualize your schema as an Entity-Relationship (ER) diagram.

Shortcuts

Ctrl+Enter: Execute queries.

Ctrl+Shift+Z: Redo.

Ctrl+/: Comment or uncomment lines.

Triggers in SQL

What is a Trigger?

A trigger is a special type of stored procedure that is automatically executed (or "triggered") in response to certain events on a particular table or view in a database.

Basic Concepts

Trigger Events

INSERT: Triggered when a new row is added.

UPDATE: Triggered when an existing row is modified.

DELETE: Triggered when a row is deleted.

Trigger Timing

BEFORE: Executes before the event (useful for validation or modifications).

AFTER: Executes after the event (useful for logging or cascading actions).

Trigger Action

The SQL logic executed when the trigger is activated. This could involve:

Validating data consistency.

Automatically inserting/updating related data.

Enforcing business rules.

Example of a Trigger

DELIMITER $$

CREATE TRIGGER check\_phone\_number

BEFORE INSERT ON Employees

FOR EACH ROW

BEGIN

IF NEW.PhoneNumber IS NULL THEN

SIGNAL SQLSTATE '45000' SET MESSAGE\_TEXT = 'PhoneNumber cannot be null';

END IF;

END $$

DELIMITER ;

Explanation:

BEFORE INSERT: Runs before inserting a new record.

FOR EACH ROW: Ensures each row is checked individually.

NEW.PhoneNumber: Refers to the PhoneNumber in the new row being inserted.

SIGNAL SQLSTATE: Raises an error to prevent invalid data.

UNION Clause

Combines results from two queries.

All corresponding columns in the two queries must have the same data type or be implicitly convertible.

Example:

SELECT Name, Age FROM Students

UNION

SELECT Name, Age FROM Teachers;

Ensures data types of Name and Age are consistent across both queries.

GROUP BY Clause

Purpose

The GROUP BY clause is used to group rows sharing a common attribute and apply aggregate functions to each group.

Example

Find the total sales for each seller:

SELECT Seller, SUM(SaleAmount)

FROM Sales

GROUP BY Seller;

Key Points

Grouping Splits Rows:

Groups rows by unique values in the specified column.

Aggregate functions (e.g., SUM, COUNT) apply to each group.

Columns in SELECT Clause:

Columns not part of an aggregate function must be included in the GROUP BY clause.

Multiple Groupings:

SELECT Seller, Product, SUM(SaleAmount)

FROM Sales

GROUP BY Seller, Product;

Groups by both Seller and Product.

JOINS in SQL

What are Joins?

Joins are used to combine rows from two or more tables based on a related column.

Types of Joins

INNER JOIN:

Returns rows where there is a match in both tables.

SELECT Employees.Name, Departments.Name

FROM Employees

INNER JOIN Departments ON Employees.DeptID = Departments.ID;

LEFT JOIN (OUTER JOIN):

Returns all rows from the left table and matched rows from the right table.

SELECT Employees.Name, Departments.Name

FROM Employees

LEFT JOIN Departments ON Employees.DeptID = Departments.ID;

RIGHT JOIN (OUTER JOIN):

Returns all rows from the right table and matched rows from the left table.

SELECT Employees.Name, Departments.Name

FROM Employees

RIGHT JOIN Departments ON Employees.DeptID = Departments.ID;

FULL OUTER JOIN:

Returns all rows from both tables, with NULL in unmatched rows.

SELECT Employees.Name, Departments.Name

FROM Employees

FULL OUTER JOIN Departments ON Employees.DeptID = Departments.ID;

CROSS JOIN:

Returns the Cartesian product of two tables.

SELECT A.Name, B.Name

FROM TableA A

CROSS JOIN TableB B;

SELF JOIN:

Joins a table to itself.

SELECT A.EmployeeName, B.EmployeeName

FROM Employees A

INNER JOIN Employees B ON A.ManagerID = B.EmployeeID;

Additional Topics

Indexes

Speed up the retrieval of rows.

Types:

Primary Index: Automatically created with the primary key.

Unique Index: Prevents duplicate values in a column.

Views

Virtual tables created using a SQL query.

Example:

CREATE VIEW EmployeeDetails AS

SELECT Name, DeptID

FROM Employees;

Stored Procedures

Reusable SQL code stored in the database.

Example:

DELIMITER $$

CREATE PROCEDURE GetEmployee(IN EmpID INT)

BEGIN

SELECT \* FROM Employees WHERE ID = EmpID;

END $$

DELIMITER ;

Transactions

Ensure data consistency.

Example:

START TRANSACTION;

UPDATE Accounts SET Balance = Balance - 100 WHERE ID = 1;

UPDATE Accounts SET Balance = Balance + 100 WHERE ID = 2;

COMMIT;

Conclusion

SQL is a powerful language for managing and querying relational databases. Understanding concepts like Triggers, Joins, GROUP BY, and Transactions helps streamline database operations and maintain data consistency.

**Original Query Breakdown:**

sql

Copy code

SELECT e.FirstName, s.ServiceType, s.ServingDate, SUM(s.Price \* s.Quantity) AS Amount

FROM ServiceLogs s

INNER JOIN Employees e ON s.EmployeeId = e.Id

WHERE s.ServiceType IN ("Beverages", "Water")

AND s.PaymentStatus = "Pending"

GROUP BY e.FirstName, s.ServiceType;