Data Manipulation Language (DML) Commands

Command	Description	Syntax
SELECT	Retrieve data from a database	SELECT column1, column2 FROM table_name;
INSERT INTO	Insert new records	INSERT INTO table_name (column1,column2) VALUES (value1, value2);
UPDATE	Modify existing records	UPDATE table_name SET column1 = value1 WHERE condition;
DELETE	Delete records	DELETE FROM table_name WHERE condition;

Data Definition Language (DDL) Commands

Command	Description	Syntax
CREATE	Create database and table	CREATE DATABASE database_name; USE database_name; CREATE TABLE table_name (column1data_type, column2 data_type, ...);
DROP	Remove table from the database	DROP TABLE table_name;
ALTER	Alter the structure of database	ALTER TABLE table_name ADD COLUMN column_name data_type;
TRUNCATE	Remove all records froma table, including all spaces allocated for the records are removed	TRUNCATE TABLE table_name;

Data Query Language (DQL) Commands

Command	Description	Syntax
SELECT	retrieve data from the database	SELECT column1, column2 FROM table_name;
UNION	Combine result of 2 or more SELECT statements and remove duplicates	SELECT column1, column2 FROM table1 UNION SELECT column1, column2 FROM table2;
UNION ALL	Combines the result sets of two or more SELECT statements without removing duplicates	SELECT column1, column2 FROM table1 UNION ALL SELECT column1, column2 FROM table2;

Data Control Language (DCL) Commands

Command	Description	Syntax
GRANT	Assigns new privileges to a user account	GRANT privilege_type ON object TO user [WITH GRANT OPTION];
REVOKE	Removes previously granted privileges from a user account	REVOKE privilege_type ON object FROM user;

Transaction Control Language (TCL) Commands

Command	Description	Syntax
BEGIN TRANSACTION	Allows users to start a new transaction	BEGIN TRANSACTION;
COMMIT	To save the changes made in the current transaction	COMMIT;

ROLLBACK	To undo the changes made in the current transaction	ROLLBACK;
SAVEPOINT	To create a point within a transaction to which you can later roll back	SAVEPOINT savepoint_name;

Joins

Join Type	Description	Syntax Example
INNER JOIN	Returns records that have matching values in both tables.	SELECT * FROM table1 INNER JOIN table2 ON table1.column = table2.column;
LEFT JOIN	Returns all records from the left table (table1), and the matched records from the right table (table2).	SELECT * FROM table1 LEFT JOIN table2 ON table1.column = table2.column;
RIGHT JOIN	Returns all records from the right table (table2), and the matched records from the left table (table1).	SELECT * FROM table1 RIGHT JOIN table2 ON table1.column =table2.column;
FULL OUTER JOIN	Returns all records when there is a match in either left or right table.	SELECT * FROM table1 FULL OUTER JOIN table2 ON table1.column = table2.column;
CROSS JOIN	Returns the Cartesian product of the two tables, i.e., all possible combinations of rows.	SELECT * FROM table1 CROSS JOIN table2;
SELF JOIN	Joins a table to itself using aliases to comparerows within the same table.	SELECT e1.employee_id, e1.first_name, e2.first_name AS manager_name FROM employees e1 INNER JOIN employees e2 ON e1.manager_id = e2.employee_id;
NATURAL JOIN	Joins two tables based onall columns with the same name and data types, implicitly.	SELECT * FROM table1 NATURAL JOIN table2;

Aggregate Functions

Command	Description	Syntax
COUNT()	Count the number of rows	SELECT COUNT(*) FROM table_name;
SUM()	Calculate the sum of values	SELECT SUM(column1) FROM table_name;
AVG()	Calculate the average of values	SELECT AVG(column1) FROM table_name;
MIN()	Find the minimum value	SELECT MIN(column1) FROM table_name;
MAX()	Find the maximum value	SELECT MAX(column1) FROM table_name;

Window Functions

Command	Description	Syntax
ROW_NUMBER()	Assigns a unique sequential integer to each row within a partition of result set.	ROW_NUMBER() OVER (PARTITION BY partition_column ORDER BY sort_column DESC)
RANK()	Assigns a rank to each row within a partition of a result set, with gaps in the ranking if there are ties.	RANK() OVER (PARTITION BY partition_column ORDER BY sort_column DESC)
DENSE_RANK()	Similar to RANK(), but without gaps in the ranking sequence. Useful for continuous ranking without gaps.	DENSE_RANK() OVER (PARTITION BY partition_column ORDER BY sort_column DESC)
NTILE()	Divides the result set into specified number of groups (buckets), assigning each row a group number.	NTILE(number_of_buckets) OVER (PARTITION BY partition_column ORDER BY sort_column)
LEAD()	Retrieves the value from row that is physically next to current row by offset number within a result set.	LEAD (column, offset) OVER (PARTITION BY partition_column ORDER BY sort_column)
LAG()	Retrieves the value from row that is physically previous to the current row by offset number within a result set.	LAG(column, offset) OVER (PARTITION BY partition_column ORDER BY sort_column)
FIRST_VALUE()	Retrieves the value from the first row in a partition of a result set	FIRST_VALUE(column) OVER (PARTITION BY partition_column ORDER BY order_column)

LAST_VALUE()	Retrieves the value from the last row in a partition of the result set	LAST_VALUE(column) OVER (PARTITION BY partition_column ORDER BY order_column)

Date-Time Functions

CURDATE()	Returns the current date.	SELECT CURDATE();	2024-07-29
CURRENT_DATE()	Returns the current date.	SELECT CURDATE();	2024-07-29
CURTIME()	Returns the current time.	SELECT CURTIME();	14:35:21
CURRENT_TIME()	Returns the current time.	SELECT CURRENT_TIME ();	14:35:21
NOW()	Returns the current date and time.	SELECT NOW();	2024-07-29 14:35:21
CURRENT_TIMESTAMP()	Returns the current date and time.	SELECT CURRENT_TIMESTAMP();	2024-07-29 14:35:21
DATE()	Extracts the date part of a date or datetime expression.	SELECT DATE('2024-07-29 10:30:00');	2024-07-29

DATE_ADD()	Adds a time interval to a date.	SELECT DATE_ADD('2024-07-29', INTERVAL 1 DAY)	2024-07-30
DATE_SUB()	Subtracts a time interval from a date.	SELECT DATE_SUB('2024-07-29', INTERVAL 1 DAY);	2024-07-28
DATEDIFF()	Returns the number of days between two dates.	SELECT DATEDIFF('2024-07-29', '2024-07-01');	28
DAY()	Returns the day of the month for a date.	SELECT DAY('2024-07-29');	29
DAYNAME()	Returns the name of the weekday for a date.	SELECT DAYNAME('2024-07-29');	Monday
DAYOFWEEK()	Returns the weekday index for a date (1 = Sunday, 7 = Saturday)	SELECT DAYOFWEEK('2024-07-29');	2
DAYOFMONTH()	Returns the day of the month for a date.	SELECT DAYOFMONTH('2024-07-29');	29
DAYOFYEAR()	Returns the day of the year for a date.	SELECT DAYOFYEAR('2024-07-29');	211
EXTRACT()	Extracts parts of a date.	SELECT EXTRACT(YEAR FROM '2024-07-29');	2024
MONTH()	Returns the month for a date.	SELECT MONTH('2024-07-29');	7
MONTHNAME()	Returns the name of the month for a date.	SELECT MONTHNAME('2024-07-29');	July
YEAR()	Returns the year for a date.	SELECT YEAR('2024-07-29');	2024
WEEK()	Returns the week number for a date.	SELECT WEEK('2024-07-29');	31
WEEKDAY()	Returns the weekday index for a date (0 =	SELECT WEEKDAY('2024-07-29');	0

	Monday, 6 = Sunday).		
WEEKOFYEAR()	Returns the calendar week of the date.	SELECT WEEKOFYEAR('2024-07-29');	31
PERIOD_ADD()	Adds a period to a year-month.	SELECT PERIOD_ADD(202407, 2);	202409
PERIOD_DIFF()	Returns the difference between two year-month periods.	SELECT PERIOD_DIFF(202407, 202401);	6
SEC_TO_TIME()	Converts seconds to 'HH:MM:SS' format.	SELECT SEC_TO_TIME(3600);	01:00:00
STR_TO_DATE()	Converts a string to a date based on a specified format.	SELECT STR_TO_DATE('29-07-2024', '%d-%m-%Y');	2024-07-29
SUBDATE()	Same as DATE_SUB()	SELECT SUBDATE('2024-07-29', INTERVAL 1 DAY);	2024-07-28
SUBTIME()	Subtracts a time interval from a time/datetime.	SELECT SUBTIME('10:30:00', '01:00:00');	09:30:00
TIME()	Extracts the time part of a time/datetime expression.	SELECT TIME('2024-07-29 10:30:00');	10:30:00
TIMEDIFF()	Returns the difference between two times.	SELECT TIMEDIFF('10:30:00', '09:00:00');	01:30:00
TIMESTAMP()	Returns a datetime value from a date or datetime expression.	SELECT TIMESTAMP('2024-07-29');	2024-07-29 00:00:00