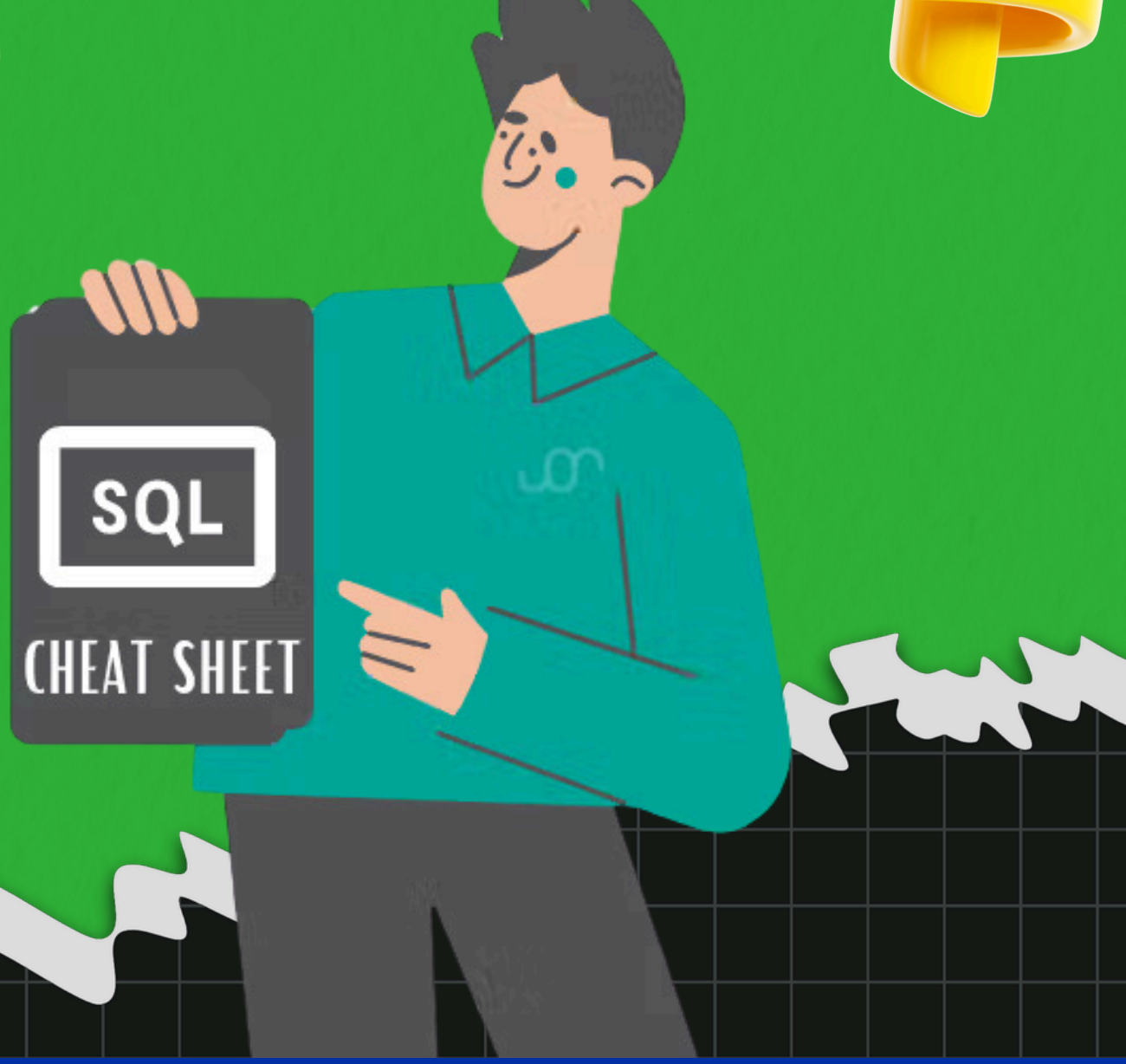

Yogesh Tyagi
@ytyagi782

All-In-one SQL Cheatsheet

Commands, Syntax, and Examples for Every Scenario



https://www.linkedin.com/in/ytyagi782/

Yogesh Tyagi
@ytyagi782

DML Commands
Data Manipulation Language

Command	Description	Syntax	Example
SELECT	The SELECT command retrieves data from a database.	SELECT column1, column2 FROM table_name ;	SELECT first_name, last_name FROM customers;
INSERT	The INSERT command adds new records to a table.	INSERT INTO table_nane (column, column2) VALUES (value1, value2);	INSERT INTO customers (first_name, last_name VALUES (‘Mary’, ‘Doe’);
UPDATE	The UPDATE Command is Used to modify existing records in a table.	UPDATE table_name SET column1 = value1, column2=value2 WHERE condition;	UPDATE employees SET employee_name ='John Doe', department = 'Marketing’ ;
DELETE	The DELETE command removes records from a table.	DELETE FROM tabte_name WHERE condition;	DELETE FROM employees WHERE employee_name = 'John Doe' ;


Yogesh Tyagi
@ytyagi782

DDL Commands

Data Definition Language

Command	Description	Syntax	Example
CREATE	The CREATE command creates a new database and objects, such as a table, index, view, or stored procedure.	CREATE TABLE table_name (column1 datatype1, column2 datatype2,	CREATE TABLE emptoyees (employee_id INT PRIMARY KEY, first_name VARCHAR(59) , last _ name VARCHAR(59) , age INT);
ALTER	The ALTER command adds, deletes, or modifies columns in an existing table.	ALTER TABLE tane_name ADD column _ name datatype;	ALTER TABLE customers ADO email VARCHAR(100);
DROP	The DROP command is used to drop an existing table in a database.	DROP TABLE table_name;	DROP TABLE customers;
TRUNCATE	The TRUNCATE command iS used to delete the data inside a table, but not the table itself.	TRUNCATE TABLE table_name;	TRUNCATE TABLE customers;


Yogesh Tyagi
@ytyagi782

DCL Commands
Data Control Language

Command	Description	Syntax	Example
GRANT	The GRANT command is used to give specific privileges to users or roles.	GRANT SELECT, INSERT ON table_name TO user_nane;	GRANT SELECT, INSERT ON employees TO 'John Doe' ;
REVOKE	The REVOKE command is used to take away privileges previously granted to used or roles	REVOKE SELECT, INSERT ON table_name FROM user_name ;	REVOKE SELECT, INSERT ON employees FROM ‘John Doe' ;


Yogesh Tyagi
@ytyagi782

Querying Data Commands

Command	Description	Syntax	Example
SELECT Statement	The SELECT statement is the primary command used to retrieve data from a database	SELECT column1, column2 FROM table_name;	SELECT first_name, last_name FROM customers;
WHERE Clause	The WHERE clause is used to filter rows based on a specified condition.	SELECT * FROM table_name WHERE condition;	SELECT * FROM customers WHERE age > 30;
ORDER BY Clause	The ORDER BY clause is used to sort the result set in ascending or descending order based on a specified column.	SELECT * FROM table_name ORDER BY column_name ASC| DESC;	, SELECT * FROM products ORDER BY price DESC;
GROUP BY Clause	The GROUP BY clause groups rows based on the values in a specified column. It is often used with aggregate functions like COUNT, SUM, AVG. etc.	SELECT column_name, COUNT(*)FROM tabte_name GROUP BY column_name;	SELECT category, COUNT(*) FROM products GROUP BY category;
HAVING Clause	The HAVING clause filters grouped results based on a specified condition.	SELECT column_name, COUNT (*) FROM table_name GROUP BY column_name HAVING condition;	SELECT category, COUNT(*) FROM products GROUP BY category HAVING COUNT(*)>5;

Yogesh Tyagi
@ytyagi782

Joining Commands

Command	Description	Syntax	Example
INNER JOIN	The INNER JOIN command returns rows with matching values in both tables.	SELECT * FROM table1 INNER JOIN table2 ON table1. column table2. column ;	SELECT * FROM employees INNER JOIN departments ON employees.department_id = departments.id;
LEFT JOIN/LEFT OUTER JOIN	The LEFT JOIN command returns all rows from the left table (first table) and the matching rows from the right table (second table),	SELECT * FROM table1 LEFT JOIN table2 ON table1.column table2. cotumn;	SELECT * FROM employees LEFT JOIN departments ON employees.department_id = departments.id;
RIGHT JOIN/ RIGHT OUTER JOIN	The RIGHT JOIN command returns all rows from the right table (second table) and the matching rows from the left table (first table),	SELECT * FROM table1 RIGHT JOIN table2 ON table1.column table2.column ;	SELECT * FROM employees RIGHT JOIN departments ON employees.department_id = departments.department_id
FULL JOIN/ FULL OUTER JOIN	The FULL JOIN command returns all rows when there is a match in either the left table or the right table.	SELECT * FROM table1 FULL JOIN tabte2 ON table1. column= table2.column;	SELECT * FROM employees LEFT JOIN departments ON employees.employee _ id= departments.employee_id UNION SELECT * FROM employees RIGHT JOIN departments ON employees. employee_id= departments.employee-id;
CROSS JOIN	The CROSS JOIN command combines every row from the first table With every row from the second table, creating a Cartesian product.	SELECT FROM table1 CROSS JOIN table2;	SELECT * FROM employees CROSS JOIN departments;
SELF JOIN	The SELF JOIN command joins a table with itself.	SELECT FROM table1 t1, tablel t2 WHERE t1.column t2.column;	SELECT * FROM employees t1, employees t2 WHERE t1. employee_id t2.employee_id;
NATURAL JOIN	The NATURAL JOIN command matches columns with the same name in both tables.	SELECT * FROM table1 NATURAL JOIN table2;	SELECT * FROM employees NATURAL JOIN departments;


Yogesh Tyagi
@ytyagi782

Subqueries in SQL

Command	Description	Syntax	Example
IN	The IN command is used to determine whether a value matches any value in a subquery result. It is often used in the WHERE clause.	SELECT column(s) tatte WHERE value IN (subquery) ;	SELECT * FROM customers WHERE city IN (SELECT city FROM suppliers) ;
ANY	The ANY command is used to compare a value to any value returned by a subquery. It can be used with comparison operators like =, >, <, etc.	SELECT column(s) FROM table WHERE value < ANY (subquery) ;	SELECT * FROM products WHERE price < ANY (SELECT unit_price FROM supplier_products) ;
ALL	The ALL command iS used to compare a value to all values returned by a subquery. It can be used with comparison operators like =, >, <, etc.	SELECT column(s) FROM table WHERE value > ALL (subquery) ;	SELECT * FROM orders WHERE order_amount >ALL (SELECT total_amount FROM previous_orders) ;


Yogesh Tyagi
@ytyagi782

Aggregate Functions Commands

Command	Description	Syntax	Example
COUNT()	The COUNT command counts the number of rows or non-null values in a specified column.	SELECT COUNT(column_name) FROM table_name;	SELECT COUNT(age) FROM employees ;
SUM()	The SUM command is used to calculate the sum of all values in a specified column.	SELECT SUM(column_name) FROM table_nane;	SELECT sum(revenue) FROM sales;
AVG()	The AVG command is used to calculate the average (mean) of all values in a specified column,	SELECT AVG(column_name) FROM table_name;	SELECT AVG(price) FROM products ;
MIN()	The MIN command returns the minimum (lowest) value in a specified column.	SELECT MIN(column_name) FRON table_name;	SELECT MIN(price) FROM products;
MAX()	The MAX command retums the maximum (highest) value in a specified column.	SELECT MAX(column_name) FROM table_name;	SELECT MAX(price) FROM products;


Yogesh Tyagi
@ytyagi782

String Functions In SQL

Command	Description	Syntax	Example
CONCAT()	The CONCAT command concatenates two or more strings into a single string.	SELECT CONCAT(string1, string2,) AS concatenated_string FROM table_name;	SELECT CONCAT(first_name, ' ', last_name) AS full_name FROM employees;
SUBSTRING()/ LENGTH()	The SUBSTRING command extracts a substring from a string.	SELECT SUBSTRING(string FROM start_position [FOR Length]) AS substring FROM table_name;	SELECT SUBSTRING(product-name FROM 1 FOR 5) AS substring FROM products;
CHAR_LENGTH() / LENGTH()	The LENGTH command returns the length (number of characters) of a string	SELECT CHAR_LENGTH(String) AS length FROM table_name;	SELECT CHAR_LENGTH(product_name) AS length FROM products;
UPPER()	The UPPER command converts all characters in a string to uppercase.	SELECT UPPER(string) AS uppercase_string FROM table_name ;	SELECT UPPER(first_name) AS uppercase_first_name FROM employees;
LOWER()	The LOWER command converts all characters in a string to lowercase.	SELECT LOWER(string) AS lowercase_string FROM table_name;	SELECT LOWER(last_name) AS lowercase_last_name FROM emptoyees;
TRIM()	The TRIM command removes specified prefixes or suffixes (or whitespace by default) from a string.	SELECT TRIM([LEADING | TRAILING | BOTH] characters FROM string) AS trimmed_string FROM table_name;	SELECT TRIM(TRAILING FROM full_name) AS trimmed_full_name FROM customers ;
LEFT()	The LEFT command returns a specified number of characters from the left of a string.	SELECT LEFT(string, num_characters) AS left_string FROM table_name;	SELECT LEFT (product_name, 5) AS left_product_name FROM products;
RIGHT()	The RIGHT command returns a specified number of characters from the right of a string.	SELECT RIGHT(string, num_characters) AS right_string FROM table_name ;	SELECT RIGHT (order_number, 4) AS right_order_number FROM orders ;
REPLACE()	The REPLACE command replaces occurrences of a substring within a string,	SELECT REPLACE (string, Old_substring , new_substring) AS replaced_string FROM table-name;	SELECT REPLACE (description, 'old_string' , 'new_string') AS replaced_description FROM product_descriptions ;

Yogesh Tyagi
@ytyagi782

Date and Time SQL Commands

Command	Description	Syntax	Example
CURRENT_DATE()	The CURRENT_DATE command returns the current date.	SELECT CURRENT_DATE() AS current_date;	
CURRENT_TIME()	The CURRENT_TIME command returns the current time.	SELECT CURRENT_TIME() AS current_time;	
CURRENT_ TIMESTAMP()	The CURRENT_TIMESTAMP command returns the current date and time.	SELECT CURRENT_TIMESTAMP() AS current_timestamp;	
DATE_PART()	The DATE_PART command extracts a specific part (e.g., year, month, day) from a date or time.	SELECT DATE_PART('part', date_expression) AS extracted_part;	SELECT DATE_PART('year', '2024 -04-11') AS extracted_part ;
DATE_ADD()/ DATE_SUB()	The DATE_ADD command adds or subtracts a specified number of days, months, or years to/from a date.	SELECT DATE_ADD (date_expression , INTERVAL value unit) AS new_date ;	DATE_ADD Example SELECT DATE_ADD('2024-04-11' , INTERVAL 1 DAY) AS new_date; DATE_SUB Example SELECT DATE_SUB('2024-04-11' INTERVAL 1 DAY) AS new_date;
EXTRACT()	The EXTRACT command extracts a specific part (e.g., year, month, day) from a date or time.	SELECT EXTRACT (part FROM date_expression) AS extracted_part;	SELECT EXTRACT(YEAR FROM '2024-04-11') AS extracted_part ;
TO_CHAR()	The TO_CHAR command converts a date or time to a specified format.	SELECT TO_CHAR(date_expression,'format') AS formatted_date ;	SELECT TO_CHAR('2024-04-11', 'YYYY-HH-DD') AS formatted_date;
TIMESTAMPDIFF()	The TIMESTAMPDIFF command calculates the difference between two timestamps in a specified unit (e.g., days, hours, minutes).	SELECT TIMESTAMPDIFF (unit, timestamp1, timestamp2) AS difference;	SELECT TIMESTAMPDIFF(DAY, '2024-04-10' , '2024-04-11') AS difference ;
DATEDIFF()	The DATEDIFF command calculates the difference in days between two dates.	SELECT DATEDIFF(date1, date2) AS difference _ in _ days;	SELECT DATEDIFF(' 2024-04-11' , '2024-04-10') AS difference_in_days;


Yogesh Tyagi
@ytyagi782

Conditional Expressions

Command	Description	Syntax	Example
CASE Statment	The CASE statement allows you to perform conditional logic within a query.	SELECT column1, column2, CASE WHEN condition1 THEN result1 WHEN condition2 THEN result2 ELSE default_result END AS alias FROM table_name;	SELECT order_id, total_amount, CASE WHEN total_amount > 1000 THEN 'High Value Order' WHEN total_amount > 500 THEN 'Medium value Order' Order' ELSE 'Low Value END AS order_Status FROM orders;
IF Statement	The IF function evaluates a condition and returns a value based on the evaluation.	SELECT IF (condition, true_value, false_value) AS alias FROM table_name;	SELECT name, age , IF(age > 58, 'Senior' 'Junior') AS employee-category FROM employees;
COALESCE() Function	The COALESCE function returns the first non-null value from a list of values.	SELECT COALESCE(value1, value2 ,) AS alias FROM table_name;	SELECT COALESCE(first_name, middle_name) AS preferred_name FROM employees;
NULLIF() Function	The NULLIF function returns null if two specified expressions are equal	SELECT NULL IF (expression1, expression2) AS alias FROM table_name;	SELECT NULLIF (total-amount, discounted_amount) AS diff_amount FROM orders;


Yogesh Tyagi
@ytyagi782

Set Operations

Command	Description	Syntax	Example
UNION	The UNION operator combines the result sets of two or more SELECT statements into a single result set.	SELECT column1,column 2 FROM table1 UNION SELECT column1, column2 FROM table2;	SELECT first_name, last-name FROM customers UNION SELECT first_name, last _ name FROM employees;
INTERSECT	The INTERSECT operator returns the common rows that appear in both result sets.	SELECT column1, cotumn2 FROM table1 INTERSECT SELECT column1, column2 FROM table2 ;	SELECT first_name, last_name FROM customers INTERSECT SELECT first_name, last-name FROM employees;
EXCEPT	The EXCEPT operator returns the distinct rows from the left result set that are not present in the right result	SELECT column1, cotumn2 FROM table1 EXCEPT SELECT column1, column2 FROM table2 ;	SELECT first _ name, last_name FROM customers EXCEPT SELECT first_name, last_name FROM employees;

Yogesh Tyagi
@ytyagi782

Transaction Control Commands

Command	Description	Syntax	Example
COMMIT	The COMMIT command is used to save all the changes made during the current transaction and make them permanent,	COMMIT;	BEGIN TRANSACTION; SQL statements and changes within the transaction INSERT INTO employees (name, age) VALUES ('Alice' , 30); UPDATE products SET price = 25.00 WHERE category =' Electronics ; COMMIT;
ROLLBACK	The ROLLBACK command is used to undo all the changes made during the current transaction and discard them.	ROLLBACK;	BEGIN TRANSACTION; SQL statements and changes within the transaction INSERT INTO employees (name, age) VALUES ('Bob', 35) UPDATE products SET price = 30.00 WHERE category ='Electronics ' ; ROLLBACK;
SAVEPOINT	The SAVEPOINT command is used to set a point within a transaction to Which you can later roll back.	SAVEPOINT savepoint_name;	BEGIN TRANSACTION; INSERT INTO employees (name,age) VALUES('carol', 28); SAVEPOINT before_update; UPDATE products SET price = 40.00 WHERE category ='Electronics' ; SAVEPOINT after_update; DELETE FROM customers WHERE age > 60; ROLLBACK TO before_update; At this point, the DELETE is rolled back, UPDATE remains. COMMIT;
ROLLBACK TO SAVE-POINT	The ROLLBACK TO SAVEPOINT command is used to roll back to a specific savepoint Within a transaction.	ROLLBACK TO SAVEPOINT savepoint_name ;	BEGIN TRANSACTION; INSERT INTO employees (name, age) VALUES ('David', 42); SAVEPOINT before_update; UPDATE products SET price = 50.00 WHERE category='Electronics' ; SAVEPOINT after_update; DELETE FROM customers WHERE age > 60; Rollback to the savepoint before the update ROLLBACK TO SAVEPOINT before_update; At this point, the UPDATE is rolled back, but the INSERT remains. COMMIT;
SET TRANSACTION	The SET TRANSACTION command is used to configure properties for the current transaction, such as isolation level and transaction mode.	SET TRANSACTION [ISOLATION LEVEL { READ COMMITTED $ERIALIZABLE}]	BEGIN TRANSACTION; Set the isolation levet to READ COMMITTED SET TRANSACTION ISOLATION LEVEL READ COMMITTED; SQL statements and Changes within the transaction INSERT INTO employees (name, age) VALUES ('Emily', 35); UPDATE products SET price = 60.00 WHERE category ='Electronics ; COMMIT;

Yogesh Tyagi

@ytyagi782

Follow for More