SQL for Data Analysis

PostgreSQL

MySQL

Oracle DB

SQL Server

SQLite Database

Big Query

DSA INCUBATOR
DIGITAL SKILLUP AFRICA

The Facilitator

FEMI AYODELE: Business Intelligence Developer || Analytics Engineer || Data Analyst || Microsoft Trainer

WORK EXPERIENCE

GTCO · Lotus Beta Analytics NIGERIA LIMITED · Snapnet

COUNTRY EXPERIENCE

EDUCATION

INTERESTS & OBSESSIONS

PROFESSIONAL QUALIFICATIONS

- Microsoft certified Power BI Data Analyst
- Microsoft Certified Azure Relational Database
- Microsoft certified Azur AI
- Oracle Certified Database Management

SQL for Data Analysis Course breakdown

Week 1:SQL for Data Analysis

- Introduction to SQL
- What is the Databases
- What are the advantage of Databases
- How to Database Store Data
- Why SQL (Why Data Analyst use SQL)
- Types of Databases (Microsoft Access, SQL Server, MySQL, PostgreSQL, Oracle, etc)

SQL Commands

- Download and Install SQL Server 2022
- Introduction to SQL Command
- Types of SQL Commands : DDL, DML, DCL, DQL, and TCL

Writing SQL Queries

- Introduction to SQL Queries (Create first Database)
- SQL - Data Types
- SQL - Create table
- SQL - Keys: Unique keys, Primary Keys, Foreign Keys.
- SQL - Insert into Table, select Table
- SQL - Drop, Delete, Truncate, Rename
- SQL – Alter tables, Update tables, Drop tables, Delete tables, Truncate tables

SQL for Data Analysis Course Breakdown

Week 2: SQL Operators and Clauses

SQL Aggregation Part 1

- SQL – Clause: where, order by, Group by, Having, Top.
- SQL – Operators: Like, AND, OR, Between, Not Between, IN, NOT IN.
- SQL - UNION Vs UNION ALL, INTERSECT.
- SQL- Wild Card

SQL Joins

- Introduction to Join
- Type of Join

3rd Week of SQL for Data Analysis

SQL Aggregation Part 2

- Introduction to SQL Aggregation
- Introduction to Null
- Aggregation: COUNT, SUM, AVERAGE, MIN, MAX etc

SQL Views

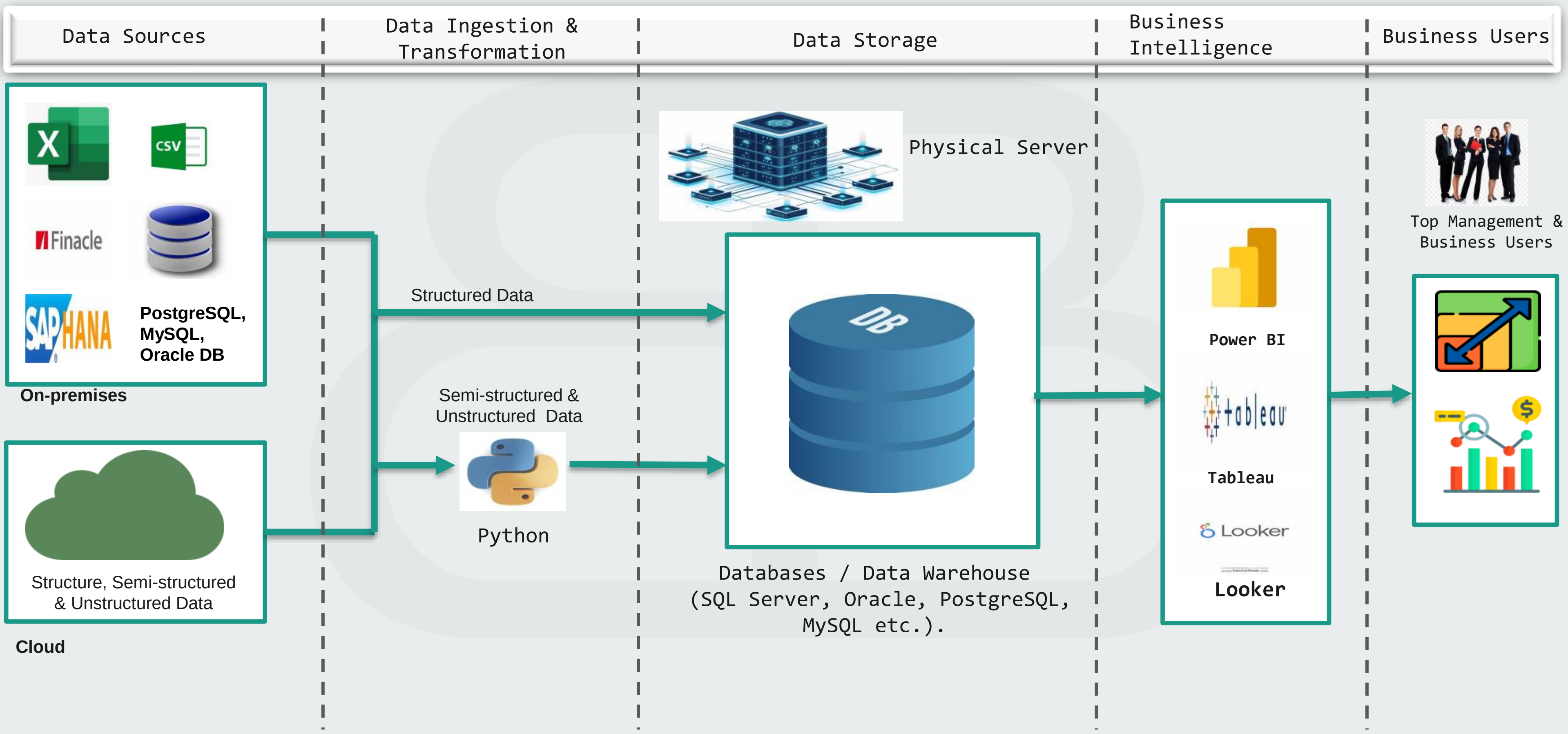
- Introduction to SQL Views
- SQL – Create views
- SQL - update views
- SQL – Drops
- SQL – Rename views

SQL CASE WHEN STATEMENT

- Introduction to CASE Statement
- SQL – Create CASE WHEN Statement.

DSA DIGITAL SKILLUP AFRICA INCUBATOR

Conceptual Architecture of Data Movement


Data Sources
Data Ingestion & Transformation
Data Storage
Business Intelligence
Business Users
CSV
Finacle
SAP HANA
PostgreSQL, MySQL, Oracle DB
On-premises
Structure, Semi-structured & Unstructured Data
Cloud
Structured Data
Semi-structured & Unstructured Data
Python
Physical Server
DB
Databases / Data Warehouse (SQL Server, Oracle, PostgreSQL, MySQL etc.).
Power BI
Tableau
Looker
Top Management & Business Users

DSA INCUBATOR
DIGITAL SKILLUP AFRICA

Introduction to SQL

- SQL stands for Structured Query Language.
- It is used for storing and managing data in Relational Database Management System (RDBMS).
- It is a standard language for Relational Database System. It enables a user to relate databases and tables.
- All the RDBMS like **MySQL**, **PostgreSQL**, **Oracle**, **MS Access**, **and** **SQL Server** use SQL as their standard database language.
- SQL allows users to query the database in several ways, using English-like statements

What are the SQL?

SQL follows the following rules:

- Structure query language is not case sensitive. Generally, keywords of SQL are written in uppercase.
- Statements of SQL are dependent on text lines. We can use a single SQL statement on one or multiple text line.
- Using the SQL statements, you can perform most of the actions in a database.
- SQL depends on tuple relational calculus and relational algebra

What is the Databases

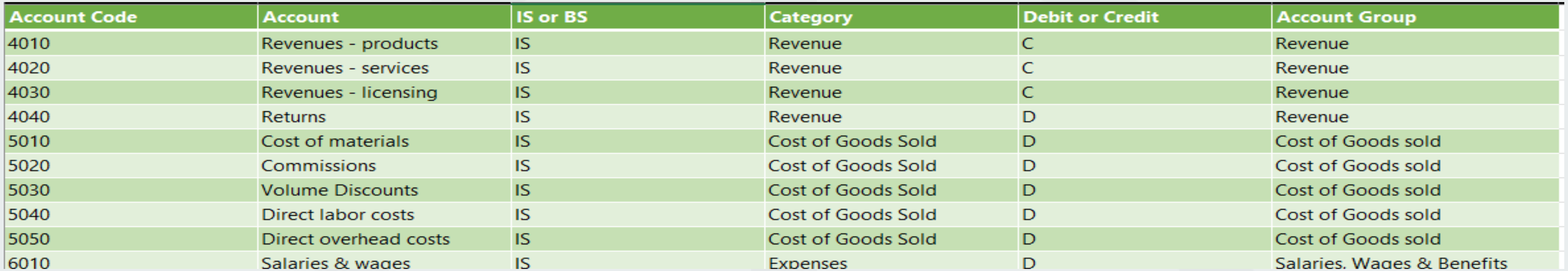
A database is an organized collection of data that is stored and managed in a structured way to allow for easy access, retrieval, and manipulation.

Advantage of Databases:

- **Data Integrity:** Ensures accuracy and consistency of data.
- **Security:** Protects data from unauthorized access and breaches.
- **Backup and Recovery:** Allows data to be recovered in case of loss or corruption.
- **Concurrency:** Supports multiple users accessing the database simultaneously.
- **Scalability:** Ability to handle increasing amounts of data and users without performance degradation.
- **Efficient Data Management:** Databases are optimized for quick access to large volumes of data, enabling fast retrieval of information through queries.

How Databases Store Data

Databases store data in a structured format using tables, which are composed of rows and columns. Each table represents a specific type of data, and each row (or record) in the table represents a single entry, such as a customer or transaction. The columns (or fields) define the attributes of the data, such as a customer's name, age, or account number.



Account Code	Account	IS or BS	Category	Debit or Credit	Account Group
4010	Revenues - products	IS	Revenue	C	Revenue
4020	Revenues - services	IS	Revenue	C	Revenue
4030	Revenues - licensing	IS	Revenue	C	Revenue
4040	Returns	IS	Revenue	D	Revenue
5010	Cost of materials	IS	Cost of Goods Sold	D	Cost of Goods sold
5020	Commissions	IS	Cost of Goods Sold	D	Cost of Goods sold
5030	Volume Discounts	IS	Cost of Goods Sold	D	Cost of Goods sold
5040	Direct labor costs	IS	Cost of Goods Sold	D	Cost of Goods sold
5050	Direct overhead costs	IS	Cost of Goods Sold	D	Cost of Goods sold
6010	Salaries & wages	IS	Expenses	D	Salaries, Wages & Benefits

- If you have used Excel, you should already be familiar with tables
- Tables have rows and columns just like Excel.
- Database tables for instance are organized by column
- Each column must have a unique name,
- You will notice, that some columns contain numbers, while other contain texts. In a spreadsheet, each cell can have its own data types.
- But in databases tables, all the data in a column must be of the same type.
- This makes performing analysis on database tables pretty simples, while the data type must be consistent.

How Databases Store Data

A few Key points about stored in SQL Databases:

- ➢ Data in Databases is stored in tables that can be thought of just like Excel spreadsheets.
- ➢ All the data in the same column must match in terms of data type.
- ➢ Consistent columns types are one of the main reasons working with databases is fast.: Often databases hold a LOT of data. So, knowing that the columns are all the same type of data means that obtaining data from a database can still be fast

Why SQL

There are some major advantages to using **traditional relational databases,** which we interact with using SQL.
The five most apparent are:

- ✓ SQL is easy to understand.
- ✓ Traditional databases allow us to access data directly.
- ✓ Traditional databases allow us to audit and replicate our data.
- ✓ SQL is a great tool for analyzing multiple tables at once.
- ✓ SQL allows you to analyze more complex questions than dashboard tools.

Basic SQL Command

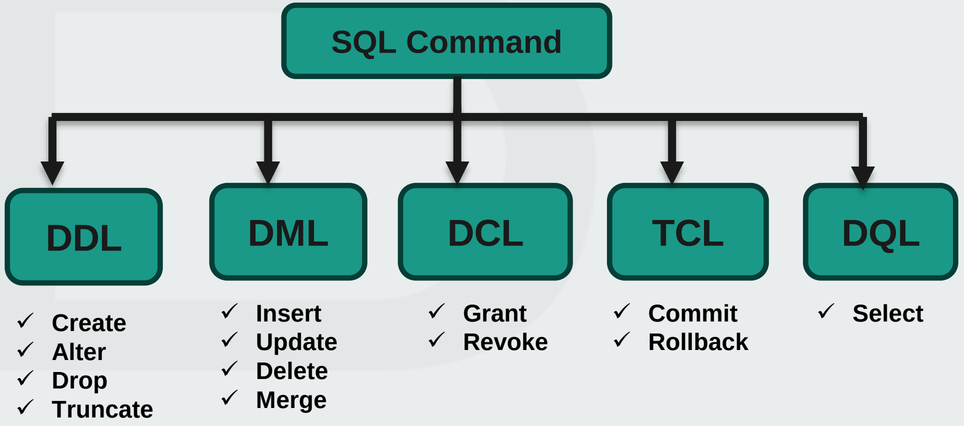
SQL Commands

- ✓ SQL commands are instructions. It is used to communicate with the database.
- ✓ It is also used to **perform specific tasks, functions, and queries of data.**
- ✓ SQL can perform various tasks like **create a table, add data to tables, drop the table, modify the table, set permission for users.**

Types of SQL Commands

There are five types of SQL commands:

- ✓ DDL: Data Definition Language
- ✓ DML: Data Manipulation Language
- ✓ DCL: Data Control Language
- ✓ TCL: Transaction Control Language
- ✓ DQL: Data Query Language



Data Definition Language (DDL)

DDL changes the structure of the table like **creating a table, deleting a table, altering a table**, etc.

All the command of DDL are auto-committed that means it permanently save all the changes in the database

- **Create:** It is used to create a new table in the database.
- **Drop:** It is used to delete both the structure and record stored in the table.
- **Alter:** It is used to alter the structure of the database. This change could be either to modify the characteristics of an existing attribute or probably to add a new attribute.
- **Truncate:** It is used to delete all the rows from the table and free the space containing the table.

Data Manipulation Language (DML)

- ✓ DML commands are used to modify the database. It is responsible for all form of **CHANGES** in the database.
- ✓ The command **of DML is not auto-committed** that means it can't permanently save all the changes in the database. They can be rollback.

Here are some commands that come under DML:

- ➢ INSERT
- ➢ UPDATE
- ➢ DELETE

- ➢ **Insert:** The **INSERT** statement is a SQL query. It is used to insert data into the row of a table.

- ➢ **Update:** This command is used to update or modify the value of a column in the table.

- ➢ **Delete:** The delete statement is used to delete existing records in a table

Data Control Language (DCL)

DCL commands are used to **GRANT** and **TAKE BACK** authority from any database user.

Here are some commands that come under DCL:

- ➢Grant
- ➢Revoke

Grant: It is used to give user access privileges to a database.
Revoke: It is used to take back permissions from the user.

Data Query Language (DQL)

DQL is used to fetch the data from the database. It uses only one command.

Select: This is the same as the projection operation of relational algebra. It is used to select the attribute based on the condition described by WHERE clause

Syntax:

```
SELECT expressions FROM TABLES WHERE conditions;
```

Example:

```
SELECT emp_name FROM employee WHERE age > 20
```

Transaction Control Language (TCL)

TCL commands are used to manage transactions in the database. These are used to manage the changes made DML Statement (**INSERT**, **DELETE** and **UPDATE** only). It also allows statements to be grouped into logical transactions

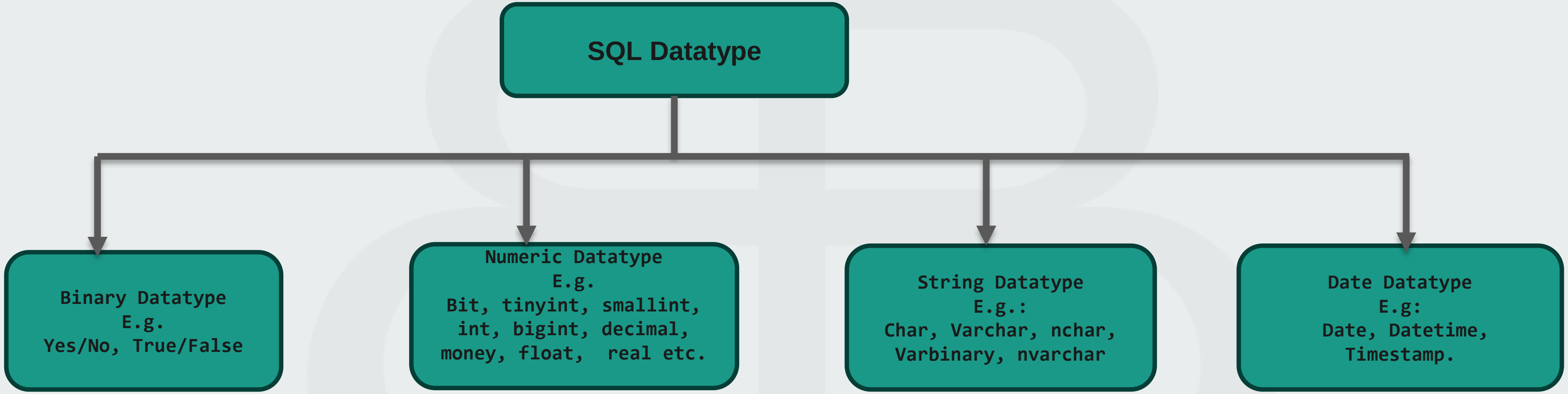
COMMIT
Commit command is used to permanently save any transaction.
Commit command is used to save all the transactions to the database.
Example:
DELETE FROM CUSTOMERS WHERE AGE = 25
COMMIT;

Rollback
This command restores the database to last committed state
Rollback command is used to undo transactions that have not already been saved to the database.
Example:
DELETE FROM CUSTOMERS WHERE AGE = 25;
ROLLBACK;

Savepoint:
It is used to roll the transaction back to a certain point without rolling back the entire transaction. Savepoint command is used to temporarily save a transaction so that you can rollback to that point whenever necessary.

SQL - Data Types

- ✓ SQL Datatype is used to define the values that a column can contain.
- ✓ Every column is required to have a name and data type in the database table.



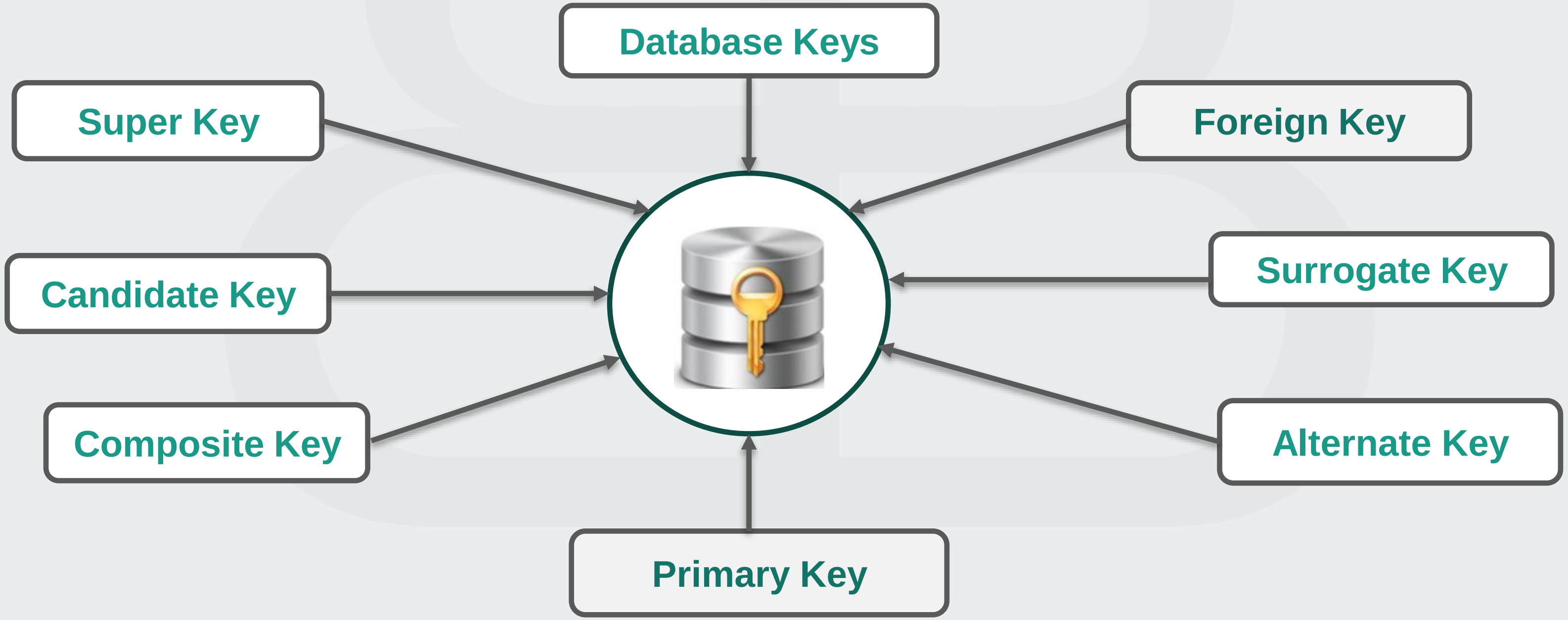
- ✓ **String Datatype** allow us to store a group of characters, enclosed in single quotes, in a record of a table column. These characters can be; numerals, letters, symbols etc.
- ✓ **Numeric Datatype:** are one of the most widely used data type in SQL. They are used to store numeric values only
- ✓ **Date Datatype** are used in SQL for values to be stored in date.

DSA DIGITAL SKILLUP AFRICA INCUBATOR

SQL - Keys

In SQL, Keys are special fields in a table that help:

- ✓ Create relationships between tables,
- ✓ Maintain uniqueness
- ✓ Ensure data is consistent and valid.



Primary Key:
A special type of key that **uniquely identifies each record** in a table. Each table can have only one primary key.
Example: **Employee_id** in the Employee table.

Foreign Key:
A field in one table that uniquely identifies a row of another table, creating a relationship between the two tables.
Example: **Employee_id** in the **Salary table** is a foreign key (FK) that references the Employee_id in the Employee table (PK)

Surrogate Key:
A surrogate key is a unique identifier for each record in a table, typically **created by the database itself (e.g. an auto-incrementing integer)**

Surrogate Key Vs Primary Key

Primary Key can have a real meaning, like Driving License, Matric No, while surrogate key is usually auto-incrementing integer with no real meaning.

Composite Key:
Composite key (also known as compound key concatenated key) is a group of two or more columns that identifies each row of a table uniquely.
Example: In salary tables, **Employee_id** and **Salary_month_year** are combined to identify each row uniquely in salary table.

Candidate Key:

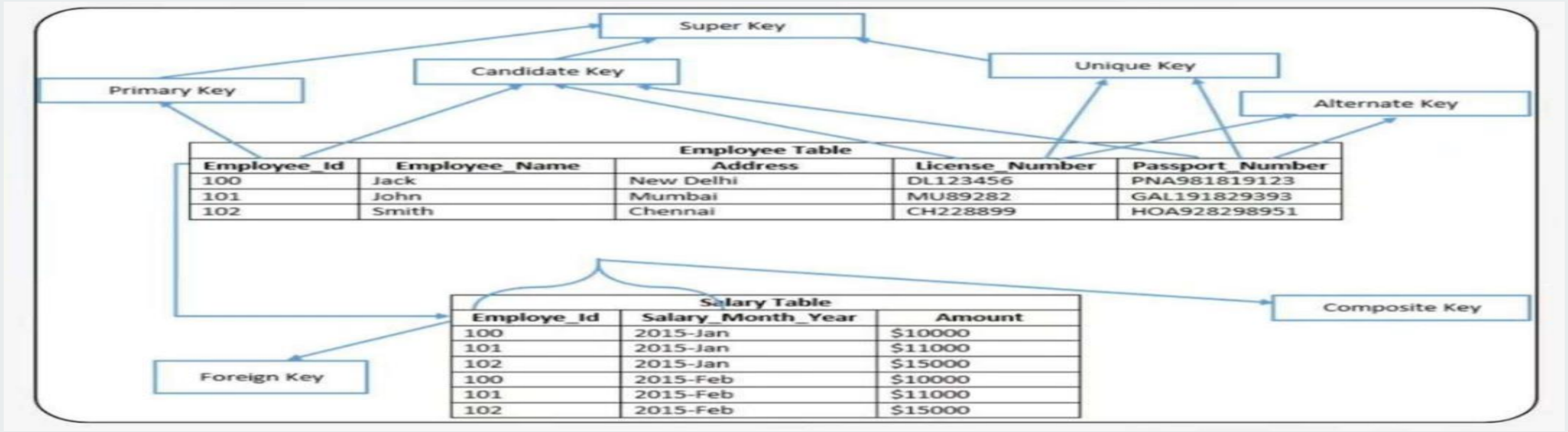
Candidate key is a key of a table which can be selected as primary key. A table can have multiple candidate keys, out of which one can be selected as a primary key.

Example: **Employee_id, License_Number**, and **Passport_Number**

Alternate key:

Alternate key is a candidate key, currently not selected as a primary key of the table

Example: **License_Number** and **Passport_Number**



Employee Table

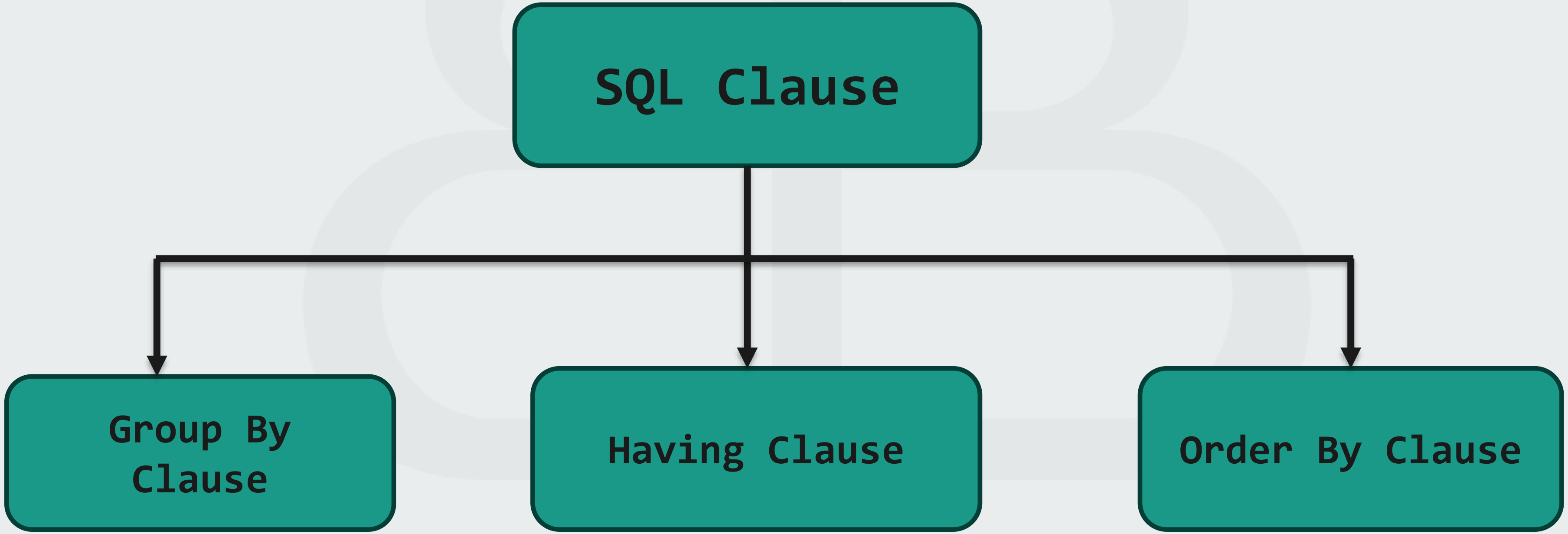
Employee_Id	Employee_Name	Address	License_Number	Passport_Number
100	Jack	New Delhi	DL123456	PNA981819123
101	John	Mumbai	MU89282	GAL191829393
102	Smith	Chennai	CH228899	HOA928298951

Salary Table

Employe_Id	Salary_Month_Year	Amount
100	2015-Jan	$10000
101	2015-Jan	$11000
102	2015-Jan	$15000
100	2015-Feb	$10000
101	2015-Feb	$11000
102	2015-Feb	$15000

SQL Clauses

SQL clauses are essential components of SQL (Structured Query Language) that define how queries interact with the database. They are used to specify conditions, modify data, and control how results are returned. Here’s an overview of some of the most common SQL clauses



Group BY Clause

SQL GROUP BY statement is used to arrange identical data into groups.

- The GROUP BY statement is used with the SQL SELECT statement.
- The GROUP BY statement follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.
- The GROUP BY statement is used with aggregation function

Having Clause

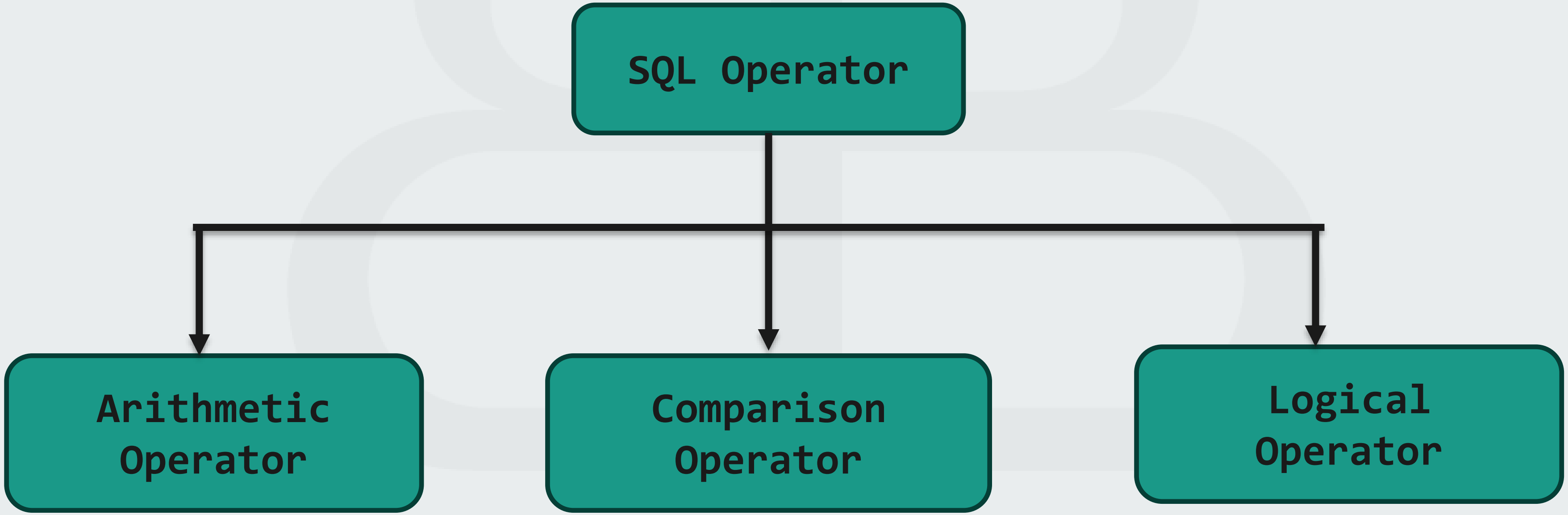
- HAVING clause is used to specify a search condition for a group or an aggregate.
- Having is used in a GROUP BY clause. If you are not using GROUP BY clause then you can use HAVING function like a WHERE clause

Order BY

- The ORDER BY clause sorts the result-set in ascending or descending order.
- It sorts the records in ascending order by default. DESC keyword is used to sort the records in descending order.

SQL Operators

An SQL operator is a reserved words or a character used primarily in an SQL statement's WHERE clause to perform operation(s), such as comparisons and arithmetic operations. These Operators are used to specify conditions in an SQL statement and to serve as conjunctions for multiple conditions in a statement



SQL Comparison Operators

S/N	Operator	Description
1	+	It adds the value of both operands.
2	-	It is used to subtract the right-hand operand from the left-hand operand.
3	*	It is used to multiply the value of both operands.
4	/	It is used to divide the left-hand operand by the right-hand operand.
5	%	It is used to divide the left-hand operand by the right-hand operand and returns reminder.

SQL Arithmetic Operators

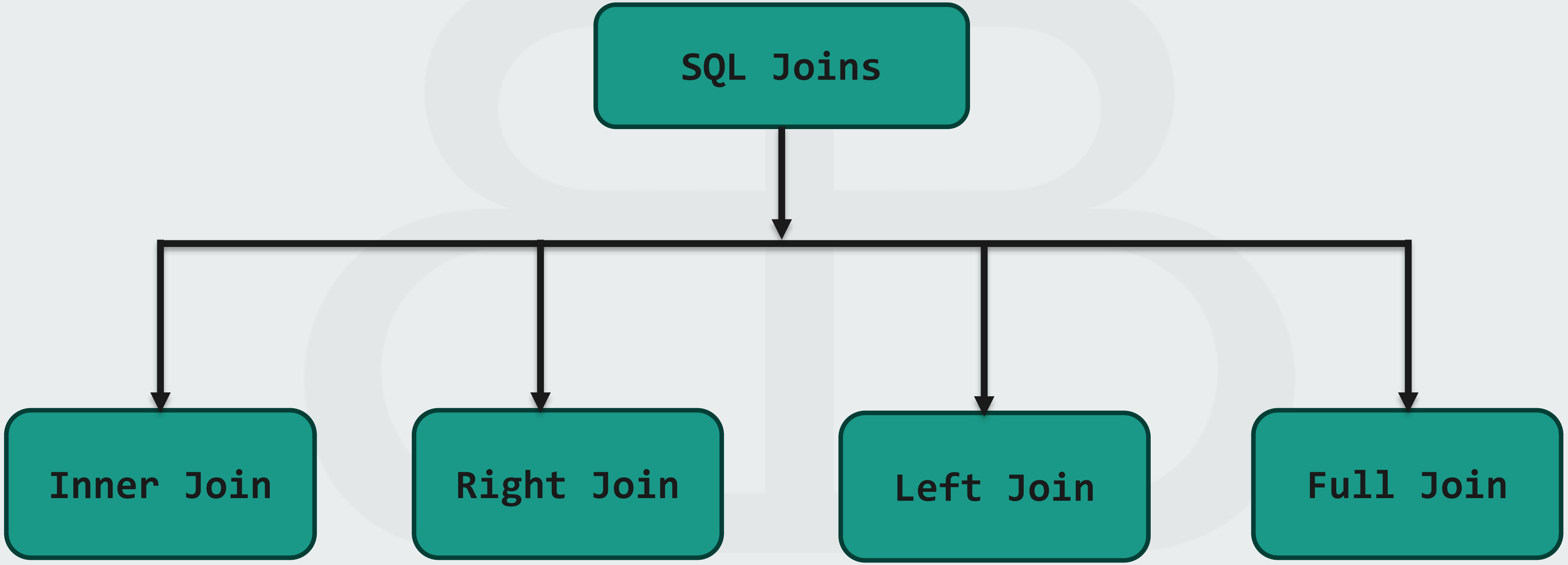
S/N	Operator	Description
1	=	It checks if two operands' values are equal or not, if the values are equal then condition becomes true.
2	!=	It checks if two operands' values are equal or not, if values are not equal, then condition becomes true.
3	<>	It checks if two operands' values are equal or not, if values are not equal then condition becomes true.
4	>	It checks if the left operand value is greater than right operand value, if yes then condition becomes true.
5	<	It checks if the left operand value is less than right operand value, if yes then condition becomes true.
6	>=	It checks if the left operand value is greater than or equal to the right operand value, if yes then condition becomes true.
7	<=	It checks if the left operand value is less than or equal to the right operand value, if yes then condition becomes true.
8	!<	It checks if the left operand value is not less than the right operand value, if yes then condition becomes true.
9	!>	It checks if the left operand value is not greater than the right operand value, if yes then condition becomes true.

SQL Logical Operators

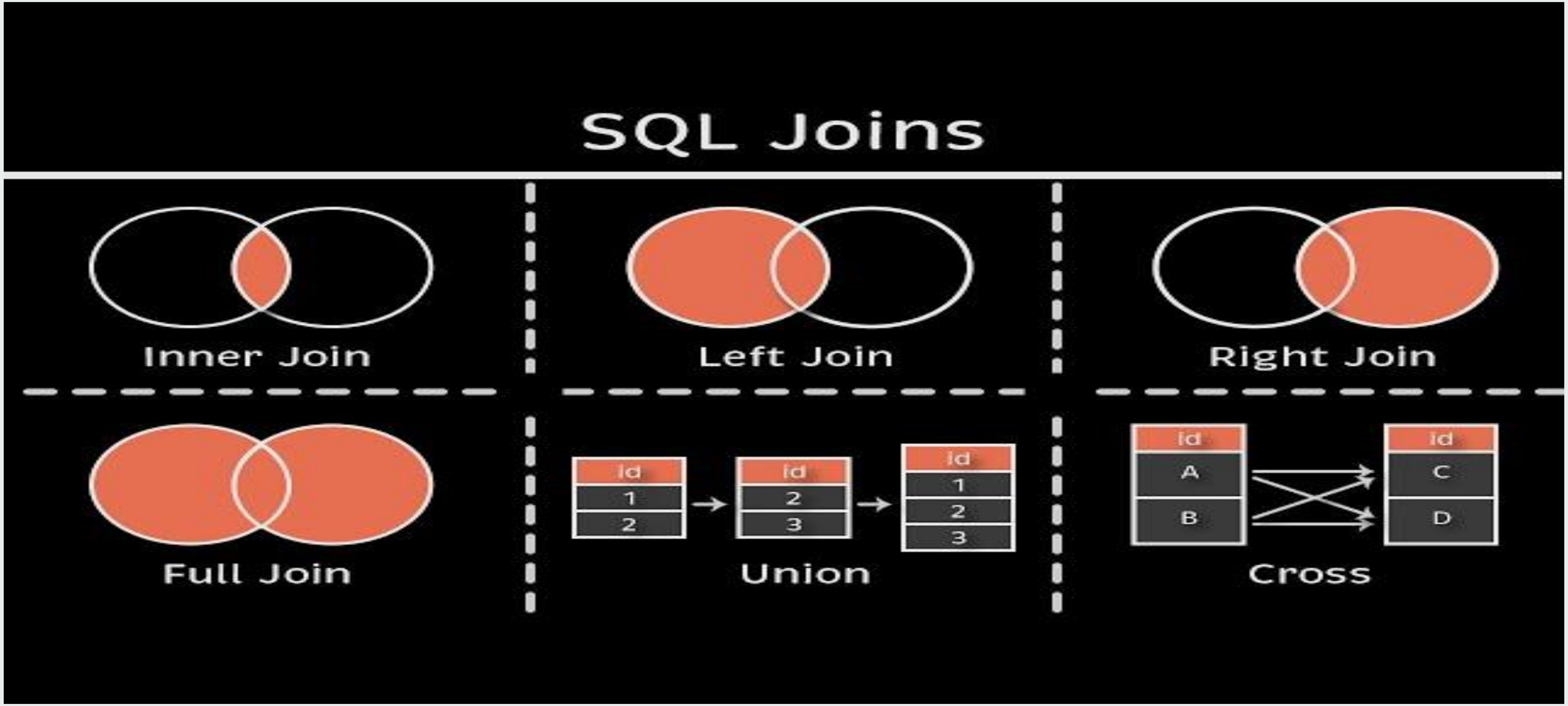
S/N	Operator	Description
1	ALL	It compares a value to all values in another value set.
2	AND	It allows the existence of multiple conditions in an SQL statement.
3	ANY	It compares the values in the list according to the condition.
4	BETWEEN	It is used to search for values that are within a set of values.
5	IN	It compares a value to that specified list value.
6	NOT	It reverses the meaning of any logical operator.
7	OR	It combines multiple conditions in SQL statements.
8	EXIST	It is used to search for the presence of a row in a specified table.
9	LIKE	It compares a value to similar values using wildcard operator.

SQL Joins

SQL JOIN means "to combine two or more tables". In SQL, JOIN clause is used to combine the records from two or more tables in a database.



DSA DIGITAL SKILLUP AFRICA INCUBATOR


SQL Joins
Inner Join
Left Join
Right Join
Full Join
id
1
2
id
2
3
id
1
2
3
Union
id
A
B
id
C
D
Cross

INNER JOIN

In SQL, INNER JOIN selects records that have matching values in both tables as long as the condition is satisfied. It returns the combination of all rows from both the tables where the condition satisfies.

```
SELECT EMPLOYEE.EMP_NAME, PROJECT.DEPARTMENT FROM EMPLOYEE
INNER JOIN
PROJECT ON PROJECT.EMP_ID = EMPLOYEE.EMP_ID
```

LEFT JOIN

The SQL left join returns all the values from left table and the matching values from the right table. If there is no matching join value, it will return NULL.

```
SELECT EMPLOYEE.EMP_NAME, PROJECT.DEPARTMENT FROM EMPLOYEE
LEFT JOIN
PROJECT ON PROJECT.EMP_ID = EMPLOYEE.EMP_ID
```

RIGHT JOIN

In SQL, RIGHT JOIN returns all the values from the values from the rows of right table and the matched values from the left table. If there is no matching in both tables, it will return NULL.

```
SELECT EMPLOYEE.EMP_NAME, PROJECT.DEPARTMENT FROM EMPLOYEE
 RIGHT JOIN
PROJECT ON PROJECT.EMP_ID = EMPLOYEE.EMP_ID
```

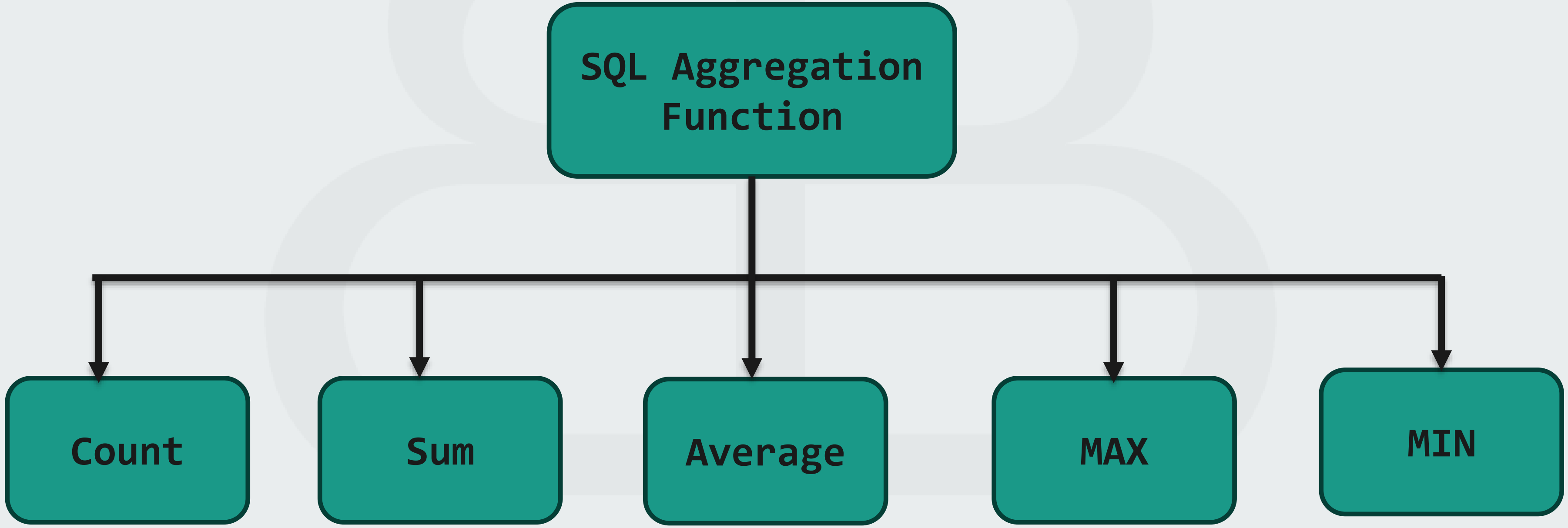
FULL JOIN

In SQL, FULL JOIN is the result of a combination of both left and right outer join. Join tables have all the records from both tables. It puts NULL on the place of matches not found.

```
SELECT EMPLOYEE.EMP_NAME, PROJECT.DEPARTMENT FROM EMPLOYEE
FULL JOIN
PROJECT ON PROJECT.EMP_ID = EMPLOYEE.EMP_ID
```

SQL Aggregate Functions

SQL aggregate functions are powerful tools used to perform calculations on a set of values, returning a single value that summarizes the data. They are commonly used in conjunction with the GROUP BY clause to group the data by one or more columns before applying the aggregate function.



1. COUNT()Purpose:

Counts the number of rows in a result set or the number of non-null values in a column.

Example: *SELECT COUNT(*) FROM employees;*
SELECT COUNT(department) FROM employees;

Use: Useful for finding the total number of records or counting non-null values in a column.

2. SUM()Purpose:

Calculates the total sum of a numeric column.

Example: *SELECT SUM(salary) FROM employees;*

Use: Adds up all the values in a numeric column.

3. AVG()Purpose:

Calculates the average value of a numeric column.

Example: *SELECT AVG(salary) FROM employees;*

Use: Returns the mean of the values in a numeric column.

4. MIN()Purpose:

Returns the smallest (minimum) value in a column.

Example: *SELECT MIN(salary) FROM employees;*

Use: Finds the lowest value in a column.

5. MAX()Purpose:

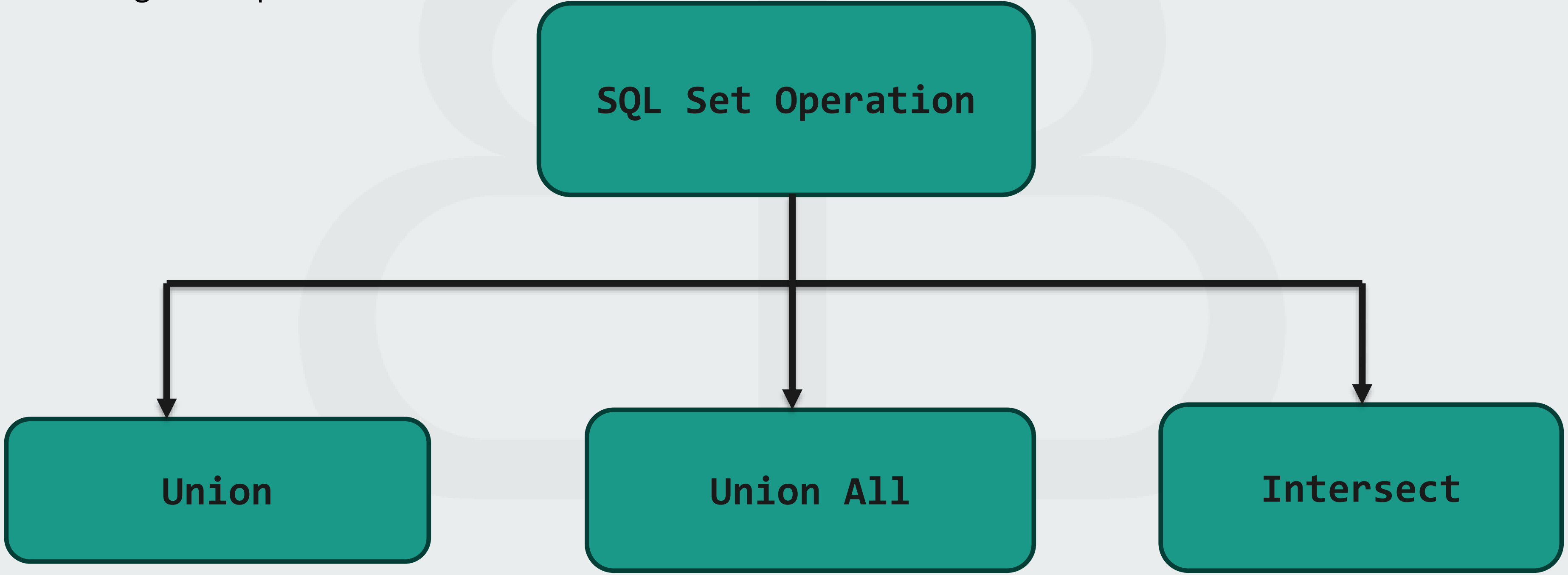
Returns the largest (maximum) value in a column.

Example: *SELECT MAX(salary) FROM employees;*

Use: Finds the highest value in a column.

SQL Set Operation

SQL set operations allow you to combine the results of two or more SELECT queries. These operations treat the result sets of each query as mathematical sets, enabling you to perform set operations like union, intersection, and difference on them. SQL provides the following set operations:



Union Operation

- The SQL Union operation is used to combine the result of two or more SQL SELECT queries.
- In the union operation, all the number of datatype and columns must be same in both the tables on which UNION operation is being applied.
- The union operation eliminates the duplicate rows from its result set.

```
SELECT column1, column2 FROM table1
UNION
SELECT column1, column2 FROM table2;
```

Union All

Combines the results of two or more SELECT queries but does not remove duplicates.

```
SELECT column1, column2 FROM table1
UNION ALL
SELECT column1, column2 FROM table2;
```

Intersect Operation

- It is used to combine two SELECT statements. The Intersect operation returns the common rows from the SELECT statements.
- In the Intersect operation, the number of datatype and columns must be the same.
- It has no duplicates, and it arranges the data in ascending order by default.

```
SELECT name FROM employees
INTERSECT
SELECT name FROM clients;
```

View in SQL

- An SQL View is a virtual table that is created based on the result set of a SQL query. Unlike a regular table, a view does not store data itself; instead, it dynamically retrieves data from one or more underlying tables whenever the view is queried.

- A view behaves like a table in SQL, allowing you to select, update, insert, and delete data (with some limitations).

- The data in a view is not stored physically; it is generated dynamically when the view is accessed.

- A view is defined using a SELECT statement that can join multiple tables, filter rows, and select specific columns.

Security

- Views can be used to restrict access to certain data in a table by exposing only specific columns or rows to the user.

- For example, you can create a view that only shows certain fields of a sensitive table, hiding the rest from the user.

Simplicity

- Views simplify complex queries. Instead of writing a complex query repeatedly, you can create a view and use it as a simple table.

- This is particularly useful in applications where complex logic needs to be reused.

SQL CASE WHEN STATEMENT

The CASE WHEN statement in SQL is a conditional expression that allows you to create different outputs based on certain conditions. It is similar to the IF-THEN-ELSE logic in programming languages.
This statement is often used to implement logic directly within SQL queries, making them more dynamic and adaptable to different scenarios.

```
SELECT
    product_name,
    CASE category_id
        WHEN 1 THEN 'Electronics'
        WHEN 2 THEN 'Furniture'
        WHEN 3 THEN 'Clothing'
        ELSE 'Other'
    END AS category_name
FROM products;
```

In this example, the CASE statement checks the category_id of each product and returns the corresponding category name.
If category_id is 1, it returns 'Electronics';
if 2, it returns 'Furniture';
and so on.

Key Points:

- ✓ **Flexibility:**
CASE can be used in SELECT, UPDATE, INSERT, and ORDER BY clauses, making it very versatile..

- ✓ **Readability:**
CASE statements can make your SQL queries more readable by replacing nested IF statements.

- ✓ **Efficiency:**
Even though CASE adds some complexity to your queries, it often improves efficiency by reducing the need for multiple queries or conditional logic in your application code.

Thank you

DSA DIGITAL SKILLUP AFRICA INCUBATOR