Data Structures and Algorithms

Introduction to Databases and Database Management System

A Day in the Life of a MERN Stack Developer

You are hired as a MERN stack developer in an organization and have been assigned to an application development project. The application is a job description portal that collects data from the candidate along with their resumes and suggests an appropriate job opening for them.

Being a developer of the application, you need to store all the incoming data in the database and whenever the user searches for the details of the applied jobs, data is displayed to the user.

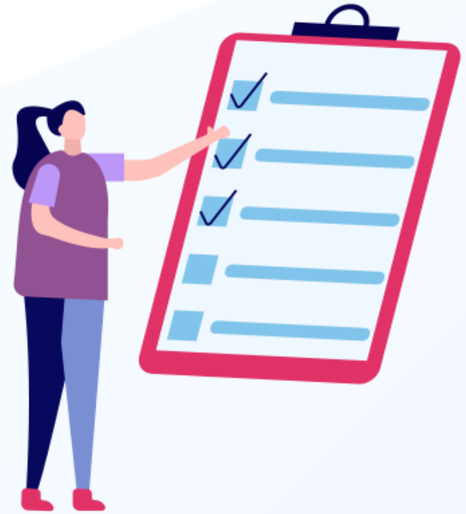
It is important for you to understand which database can be used here and how can you connect and fetch data from it.

For this, explore more about how data is stored, the need for Database Management System (DBMS), and its working.

Learning Objectives

By the end of this lesson, you will be able to:

- Analyze the fundamental differences between databases and database management systems (DBMS) to design and implement databases effectively
- Demonstrate the basics of SQL, including data types, operators, and command precedence, for efficient database querying and manipulation
- Execute SQL commands proficiently, including DDL, DML, DCL, TCL, and DQL, for accurate and secure database management and data handling
- Apply SQL subqueries and joins effectively in different scenarios, for complex data retrieval and analysis
- Utilize built-in SQL functions for data conversion, manipulation, and aggregation, for enhancing data processing and reporting capabilities



Basics of Database

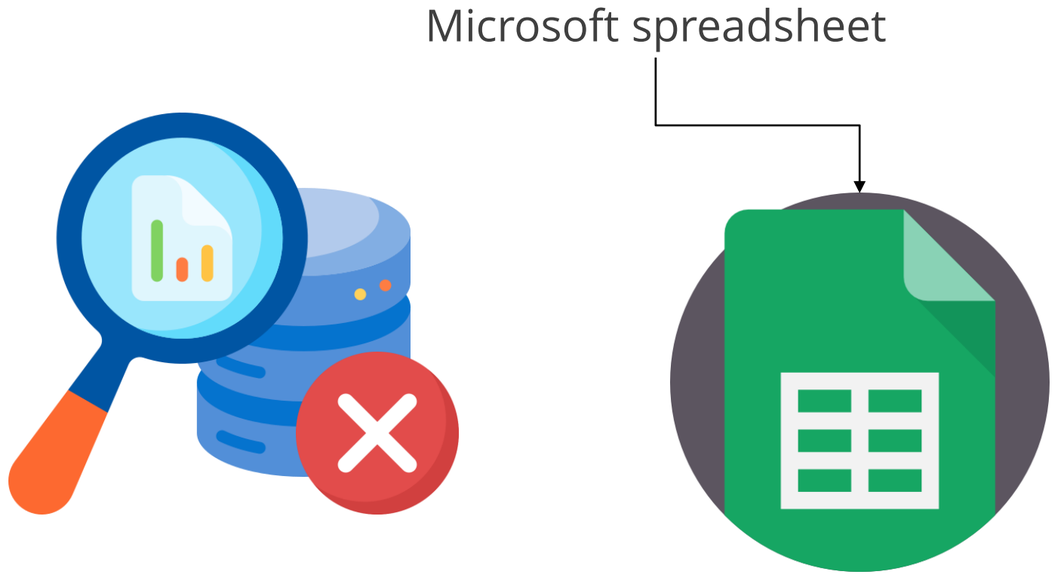
Importance of Data

Data, in simple words, is a raw form of an information. It provides insights that lead to effective business strategies and decisions.

Organizations need data and predictive analytics to develop better products.

Importance of Data

Consider that a digital marketer has been hired by an organization, and the organization does not have a database.

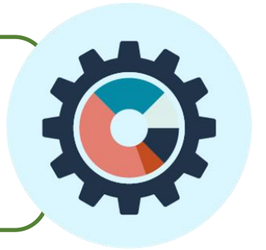


Hence, due to lack of database, the information on previous customers and a few prospective buyers is stored in a Microsoft spreadsheet.

Importance of Data

A strong database is needed to analyze prospects and digital touchpoints.

Data should be leveraged for audience analysis.

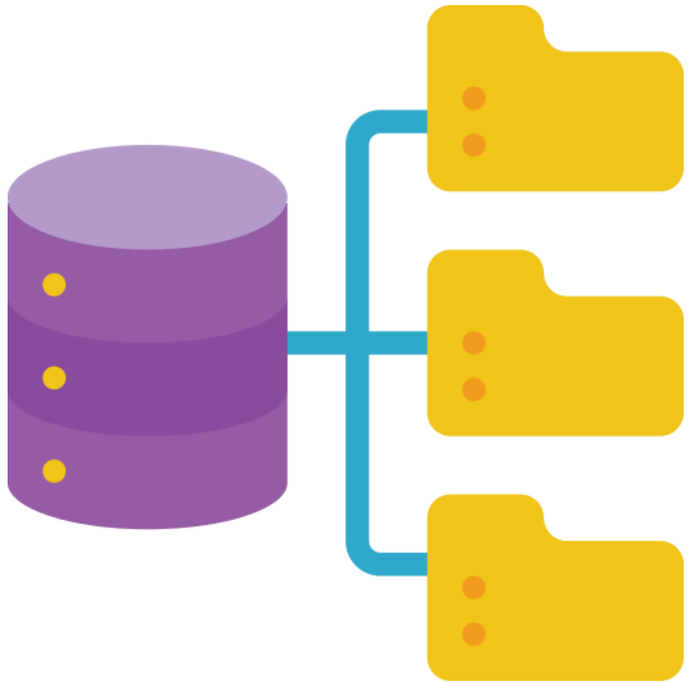


Data helps to create a personalized experience for every customer.

Database systems help in collecting huge data and storing it in the right format.

What Is Database System?

A database system is a computer-based record-keeping system.



Enables collection of structured data stored on a computer system

Allows to pull the data, edit the data

Stores data from any application

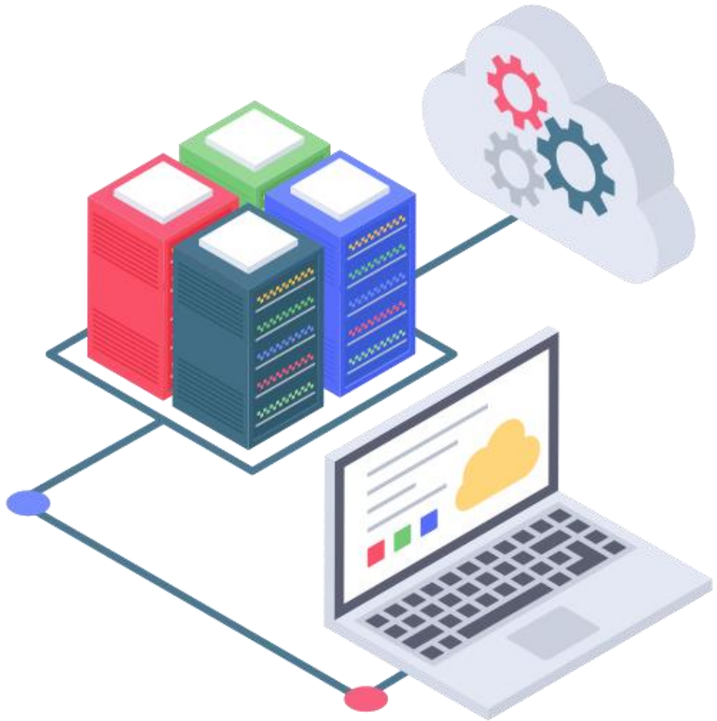
Record-Keeping

Record-keeping is a terminology used in banking, accounting, and finance.

It helps to retrieve records such as e-mails and agreements.

Database

A database is an organized collection of data stored in a structured format on a computer system that can easily be accessed.

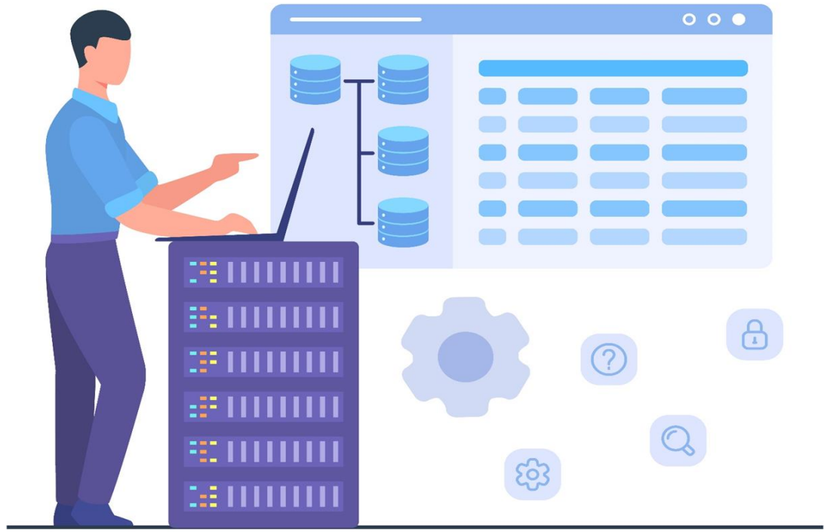


This software (DBMS) also allows users or programs to create and store, retrieve, update, and manage data with strong database security.

Database Management System (DBMS)

The DBMS executes instructions and sends the results back.

Connect with a DBMS to provide instructions for querying or modifying data.



Data + DBMS + Applications = Database system

DBMS

Any enterprise depends a lot on its proper functioning.

The data and information about different aspects of an enterprise are very crucial.

A DBMS provides enterprises with centralized control of their operational data, especially sensitive and crucial data.

Digital Marketing: Example

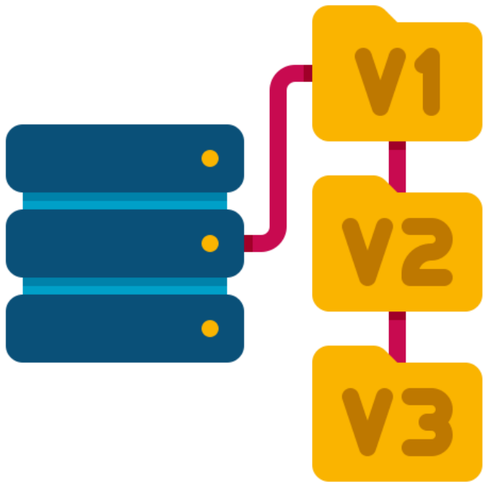
Suppose one wants to know which sector of cars has made a good sale during the last quarter.

One will access the database application and key in the information (query).

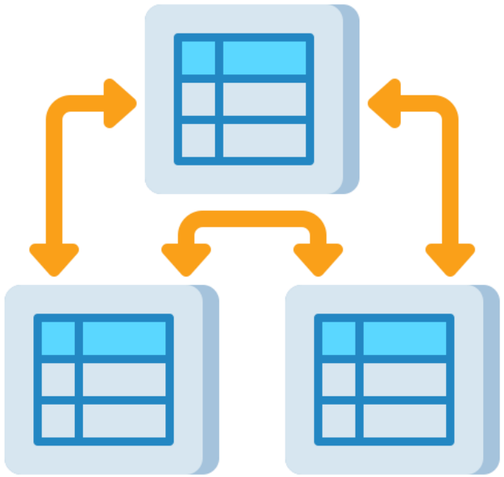
A report with numbers and a sales graph

Database

A database is also described as a set of interrelated data held together to serve various applications.



It may serve as a basis for future application development.



A database is frequently defined as the repository of information required to carry out specific duties in a firm or organization.

Database

The database enables the recovery of information and the change of data required for operation control.

A database can be searched for queries.

Database

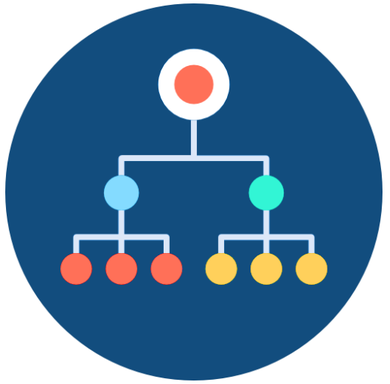
A database is a reservoir for the data required for an organization's information processing.

The information needs to be:

Accurate

Confidential

Secure



Structured

Database

The database holds all the pertinent information about the company.

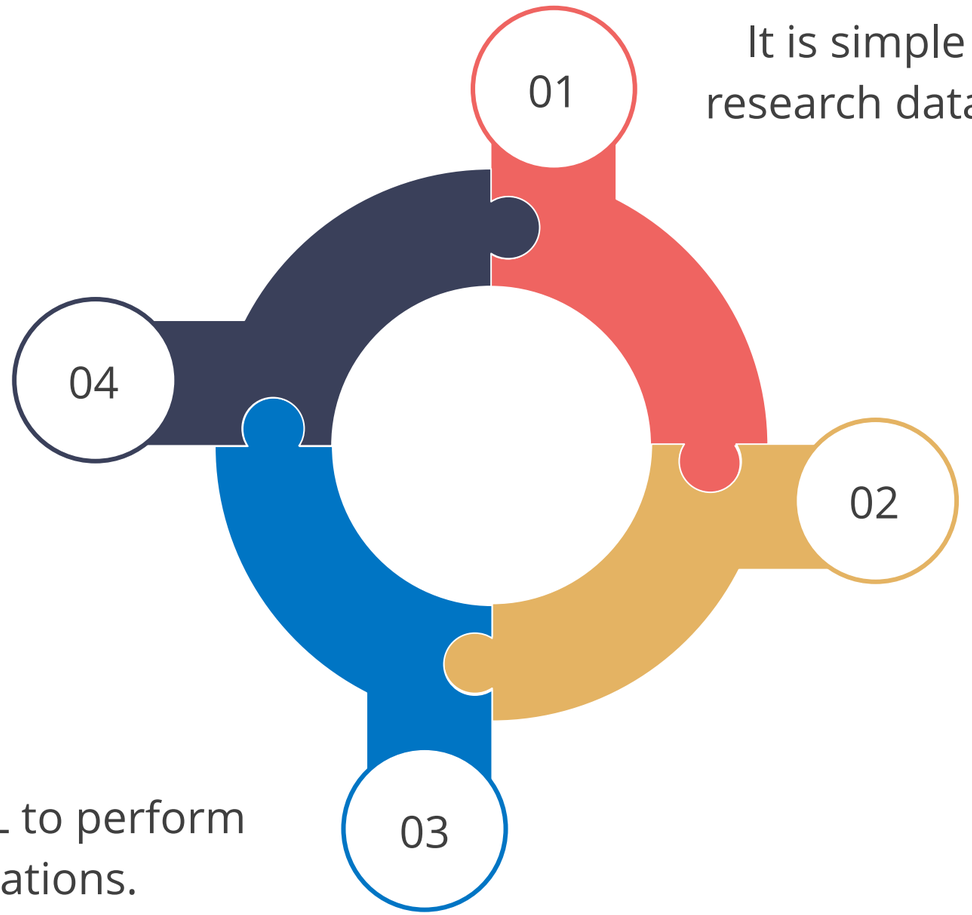
Worker records

Value-based records

Remuneration subtleties

Database

MySQL is one of the most popular databases.



01 It is simple to access and research data in a database.

02 It uses Data Query Languages (DQL) to look for data in the database.

03 It uses DQL to perform calculations.

04 It uses Data Manipulation Languages (DML) to update data.

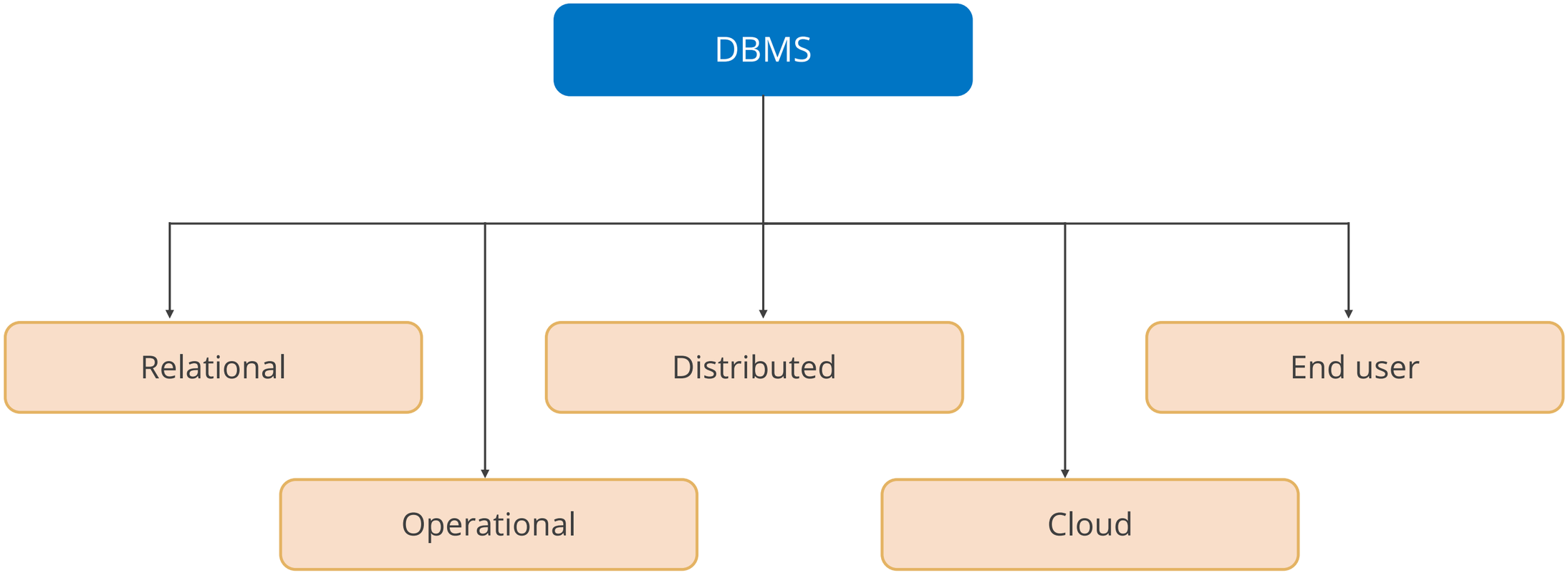
Database vs. Spreadsheets

The difference between databases and spreadsheets are as follows:

Spreadsheets	Vs	Databases
• Allow individual user		• Allow multiple users on various devices
• Store limited data storage		• Store massive amount of data
• Offer less chance for manipulation		• Allow multiple users access at once
• Are unscalable for big data		• Handle large amount of data with ease

Types of Databases

The types of databases are as follows:



Relational Database

A relational database stores data in rows and columns that form a table. It uses SQL for:

Storing data

Changing data

Maintaining data

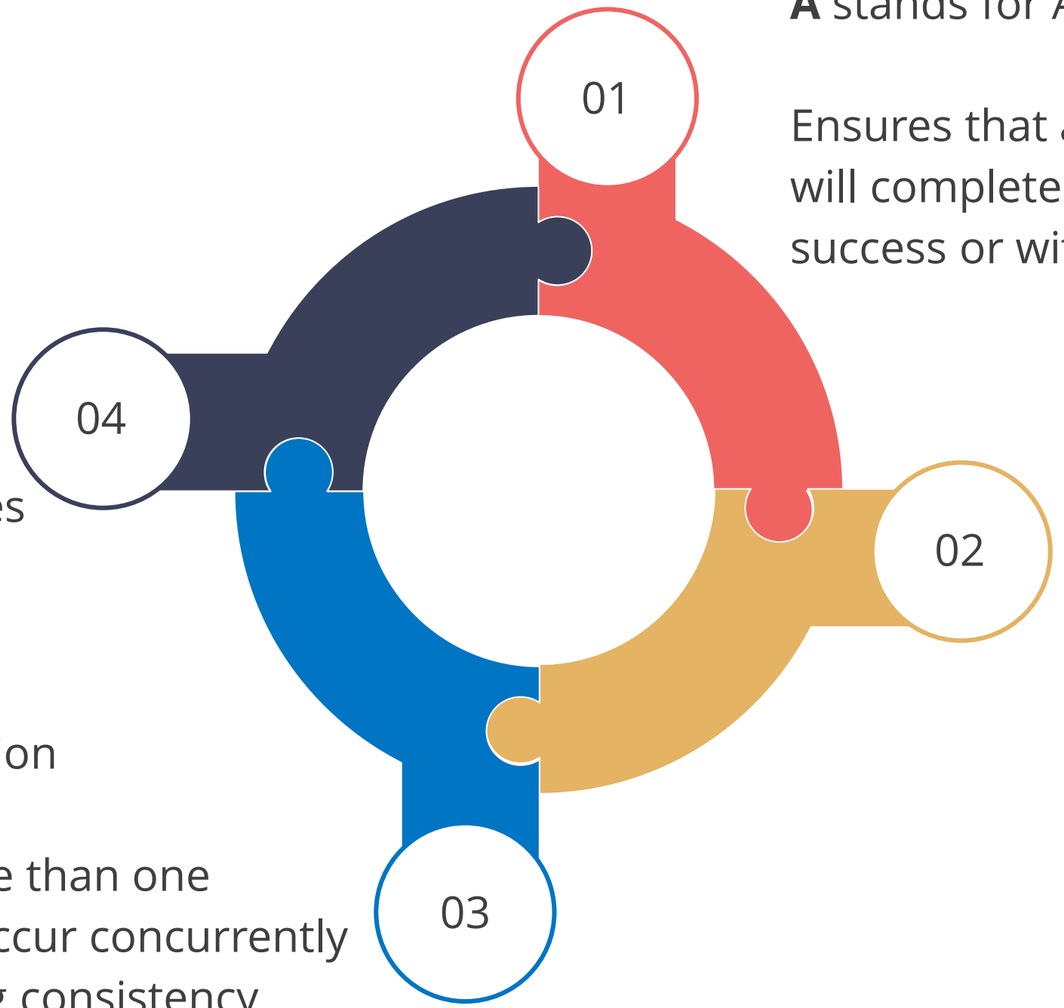
Every table in the database contains a key that differentiates data from other sources.

Relational Database: Properties

There are four properties of relational databases which are known as **ACID** properties.

A stands for Atomicity

Ensures that a data operation will complete either with success or with failure



C stands for Consistency

Ensures that when the data is modified, its value before and after the modification must be preserved

I stands for Isolation

Ensures that more than one transaction can occur concurrently without impacting consistency

D stands for Durability

Ensures that once the operation is finished and data finalized, data changes must be permanent

Operational Database

Operational databases help to create and edit the database in real time.

Companies use operational databases for handling daily transactions.

Distributed Database

A distributed database is spread over different sites, computers, or networks of computers.

Delivers files quickly and easily

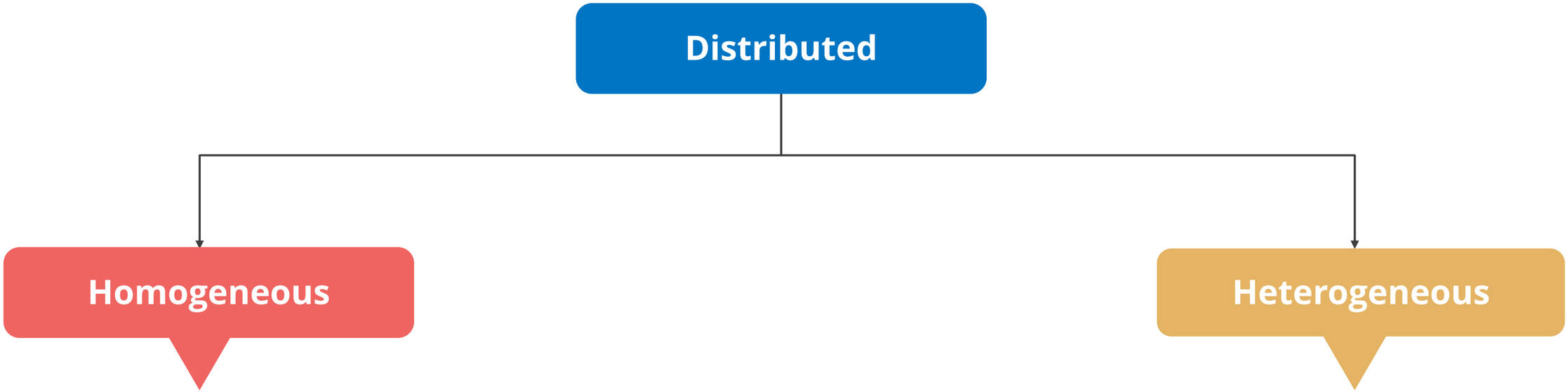
Allows to upsize the database system

Note

Failure of one server does not impact the overall database.

Distributed Database: Types

A distributed database is divided into two types.



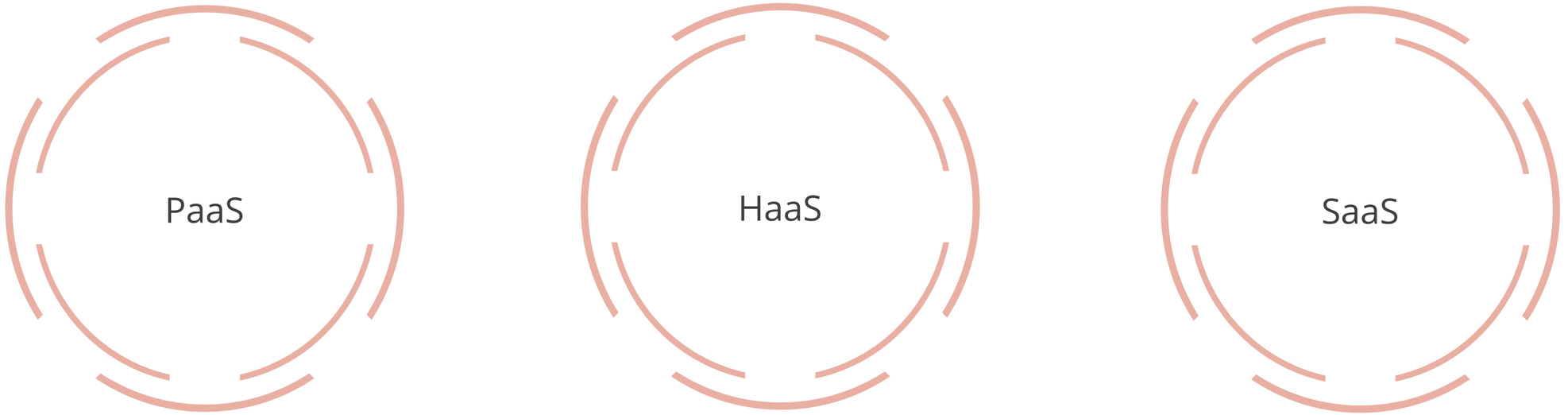
- Runs on the same operating system
- Uses the same process and hardware
- Is quite easy to manage

- Runs on different operating systems with different processes
- Leads to problems and failures as all sites are unaware of others

Cloud Database

In a cloud database, the data is stored in a virtual environment and runs on the cloud computing area.

It offers various computing services like:



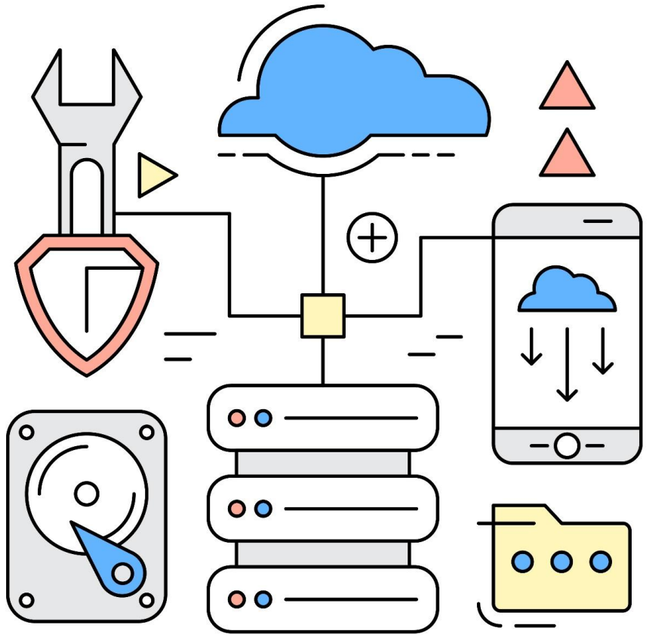
Cloud Database

There are many cloud platforms, but the most popular ones are:

End-User Database

The end-user is mindful of the product.

It is a common database that is intended for the end-user.



The end-user database is known as a **shared database**.

MySQL

MySQL is a Relational Database Management System (RDBMS) that uses SQL to query from databases.

- It is a widely used database as it is free, fast, reliable, and scalable.
- It is written in C++ and C programming language.
- It allows for keeping records of an important database.
- It contains many tables and stores thousands of individual records.

MySQL

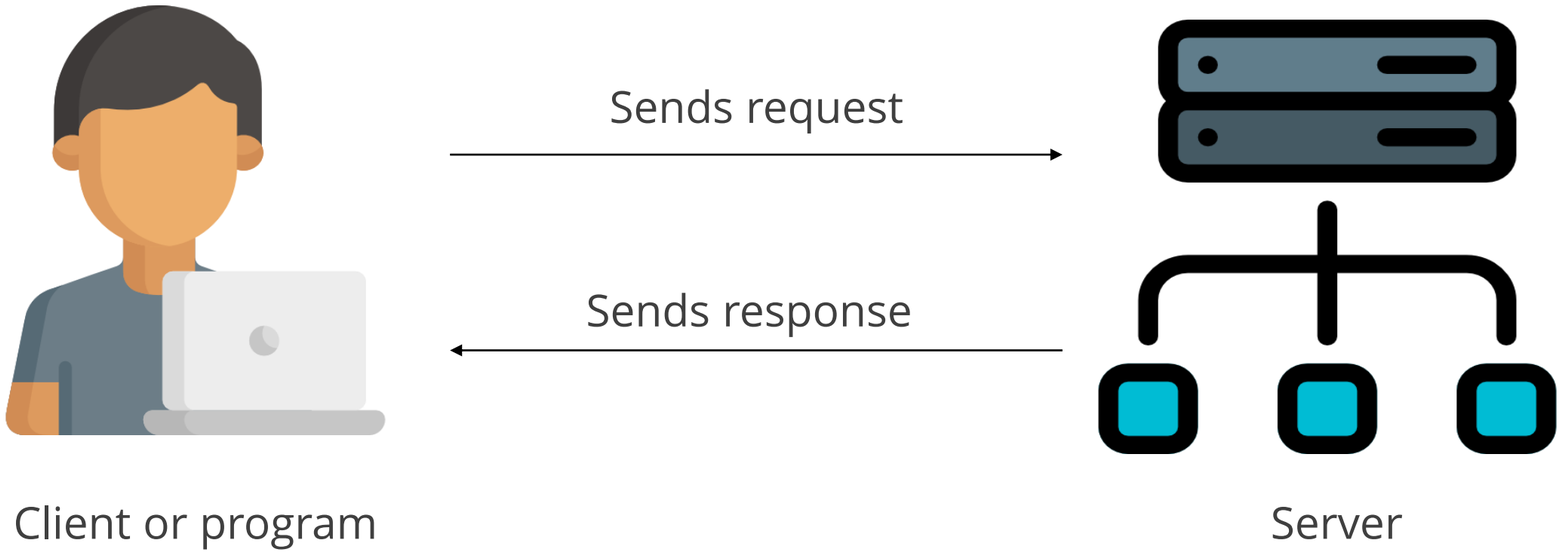
MySQL is a Relational Database Management System (RDBMS) that uses SQL to query from databases.

- It provides in-built features that support a secure environment.
- It manages information.
- It operates using client/server architecture.
- It is a multi-user database system.

MySQL= MySQL server instance + MySQL database

How Does MySQL Function?

The server takes the client requests that are received on the network through Graphic User Interface (GUI) and accesses database contents according to those requests.



Clients refer to the programs that connect to the database server and issue queries in a pre-specified format.

MySQL: Features

The features of MySQL are as follows:

Speed

- Runs very fast
- Supports clustered servers demanding applications
- Provides multiple functionalities

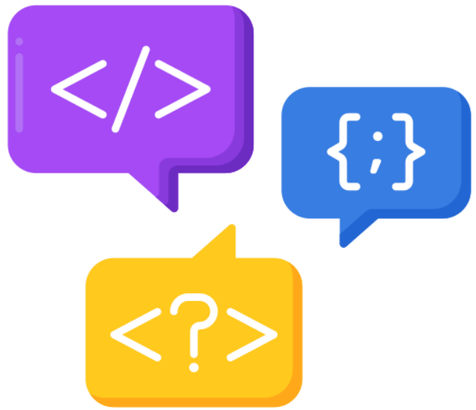
MySQL: Features

The features of MySQL are as follows:

Ease of Use

- Is a high-performance simple database system
- Supports multiple OS with different programming languages like:

1. PHP
2. PERL
3. C, C++
4. JAVA



MySQL: Features

The features of MySQL are as follows:

Data Types

- Supports fixed-length and variable-length records
- Offers a standard limit of 4 GB per table

MySQL: Features

The features of MySQL are as follows:

Connectivity

Clients connect to MySQL server using various protocols.

Localization

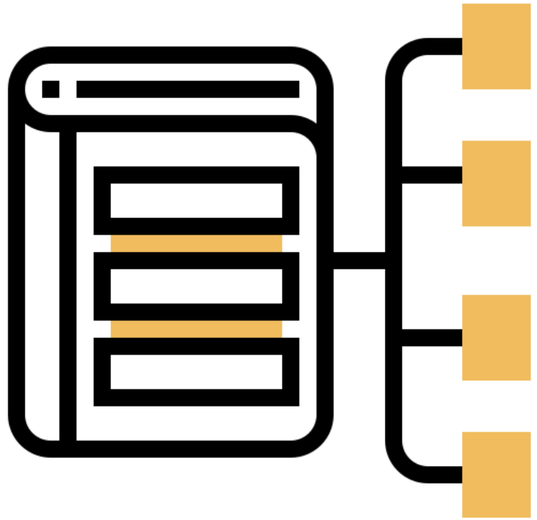
The server provides error messages in many languages.

Cost

MySQL is free.

Connecting and Disconnecting from the Server

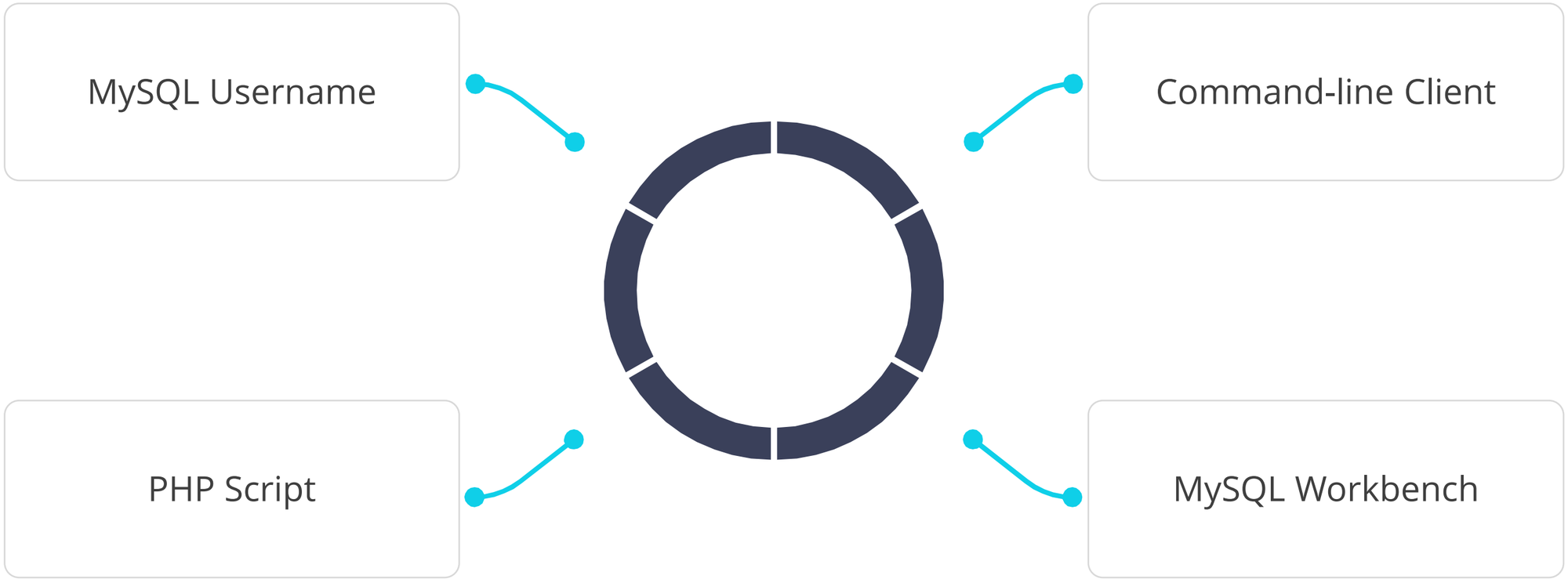
Connections are crucial in sending commands and receiving results from the other servers.



The IT facility helps to connect with the same machine servers.

Connecting to the Server

MySQL offers many ways to connect with database servers:



Connecting to the Server

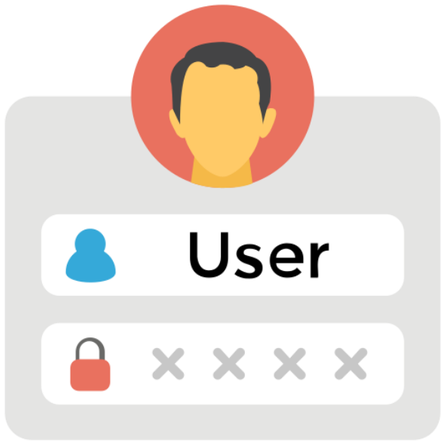
The following are the differences between Command-line Client and MySQL Workbench:

Command-line Client	vs.	MySQL Workbench
• Helps in interacting with a database server • Is available in the bin directory		• Gives the authority to: 1. Design 2. Develop 3. Create the database schemas 4. Insert queries as well as data to work with stored data

Connecting to the Server

MySQL provides the authority to its users for keeping their database secure by creating a username.

This helps to keep a record of the table that contains:



Login Information

Host Information

Account Privileges

Connecting to the Server

To make a connection with the server, enter the username and the password to authenticate the login.

If the server runs on a different machine, determine the hostname.

Connecting to the Server

Link to the host using **mysql –h host –u user -p**

MySQL database system will show the introductory information followed by a **mysql>** prompt:

```
mysql –h host –u user –p
   Enter password: *****

Welcome to the MySQL monitor. Commands end with: or Vg. Your
MySQL connection id is xxxx to the server version: x.x.x-
standard Type ‘help:’ or for help.
Type ‘\c’ to clear the buffer.mysql>
```

Connecting to the Server

When the users login into the system where SQL is running, the host can be omitted and the following code can be used:

```
Mysql –u user –p

Error message 00 Sign into the MySQL server


ERROR 2002 (HYOOO) : can't connect to local MySQL server
through socket ‘/tmp/mysql.sock’ (2) ,
```

Connect with that server by conjuring MySQL **>mysql**

Disconnecting to the Server

```
Enter QUIT or \q at the cli: mysql> QUIT

Or Press CTRL + D
```

Introduction to SQL

What Is SQL?

The Structured Query Language (SQL) allows users to create complex queries.

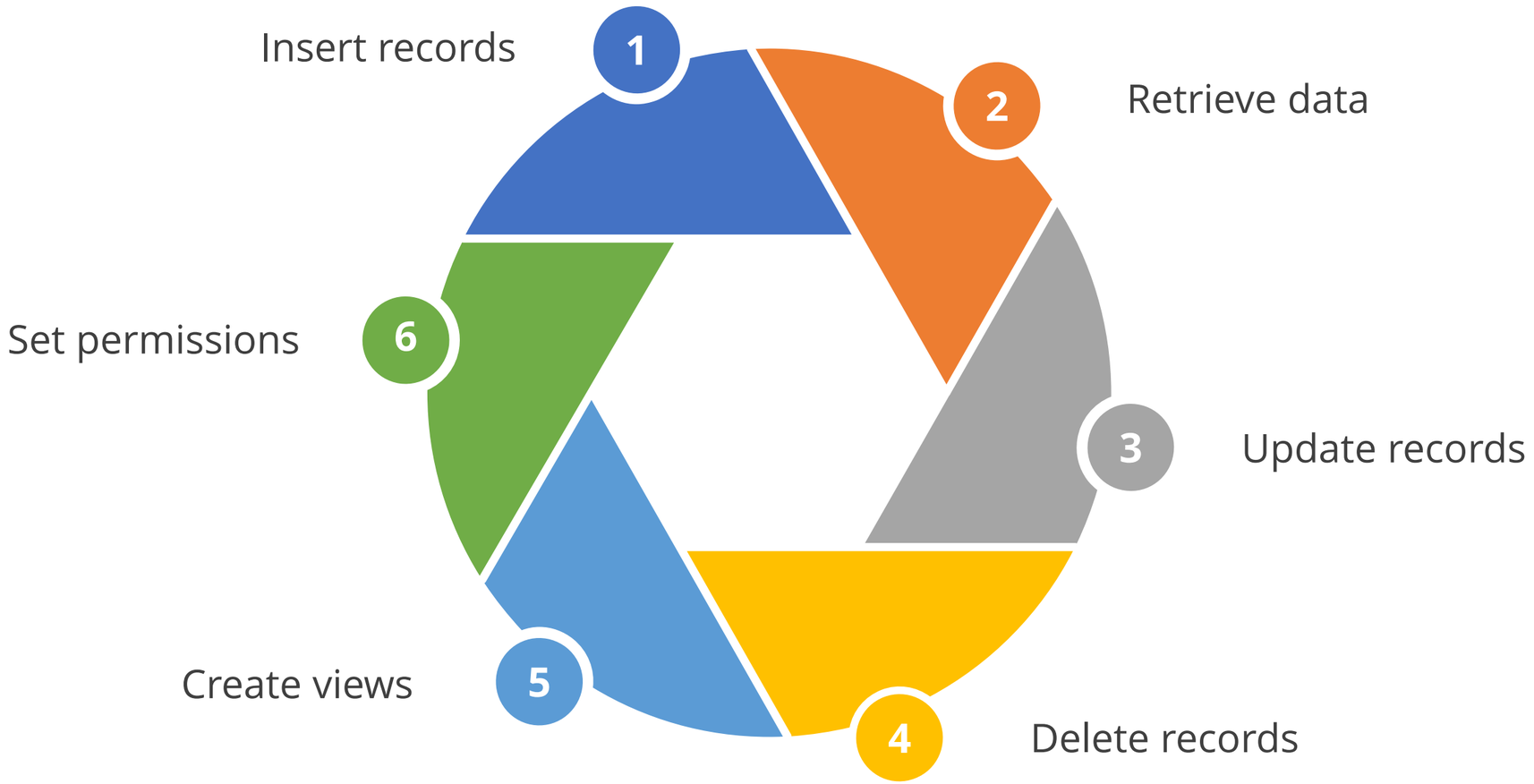
It is a database management system that is used to build, access, and edit databases.

What Is SQL?

It is compatible with SQL Server, Oracle, MySQL, and other relational database management systems.

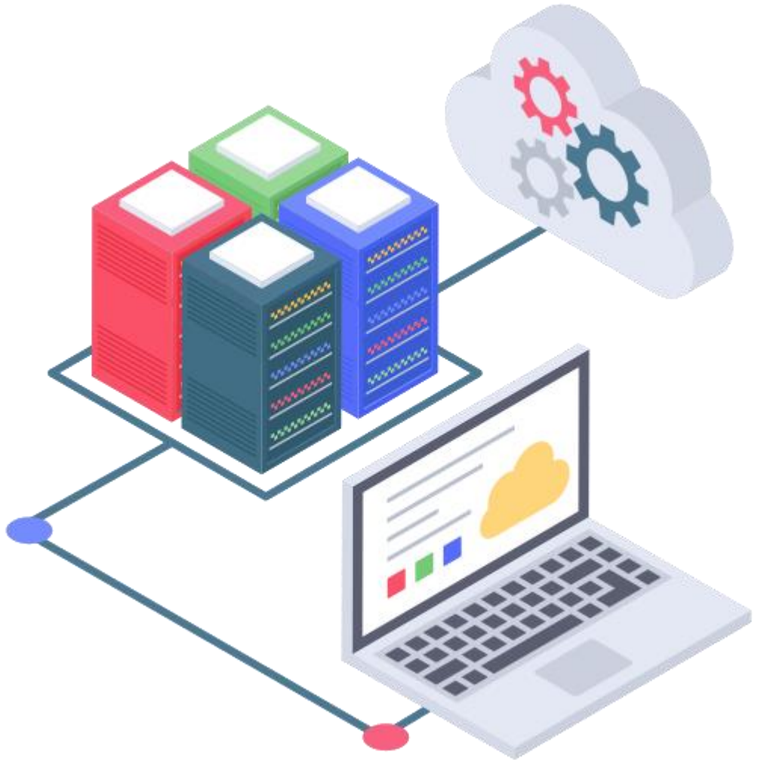
Applications of SQL

The following are the applications of SQL:



Relational Databases

A relational database is a collection of datasets that are related to each other.



These datasets are arranged in a table format with columns and rows.

Relational Databases

Each row in the table has a its unique ID called the key.

The attributes of the data are stored in the table's columns.

Benefits of Relational Databases

The following are the benefits of relational databases:

User-friendly

A relational database is more straightforward compared to a network or a hierarchical database.

Structure independent

A relational database is solely focused on data and not the structure.

Benefits of Relational Databases

The following are the benefits of relational databases:

Collaborative

Relational database allows multiple users to access the same database.

Less redundant

Data redundancy is reduced in relational databases. A single user's information is contained in a single entry in the user table.

Benefits of Relational Databases

The following are the benefits of relational databases:

Secure

The relational database does not allow unauthorized users to access the data.

Drawbacks of Relational Databases

The following are the drawbacks of relational databases:

Structural complexity

Relational databases requires a significant amount of planning because they can only store data in tabular form.

High maintenance

The upkeep of a relational database becomes more challenging as the amount of data increases.

Drawbacks of Relational Databases

The following are the drawbacks of relational databases:

Storage

A relational database takes a lot of physical memory as each action depends on separate physical storage.

Structured Query Language (SQL)

SQL functionalities include:

Connecting to a database and updating, retrieving, deleting, and inserting records

Creating new tables, stored procedures, and views in the database

Accessing to set permissions on tables, processes, and views

MySQL Security and Root Superuser

MySQL Security

Provides strong data security to protect data for:

- Secure connections
- Authentication services
- Authorization and controls
- Data encryption and security

Root Superuser

An admin who has the super privilege or GRANT statement that allows a user account to make changes and execute various operations in the database table

MySQL Security and Root Superuser

To create a superuser:

```
Login to MySQL server and type this command:

mysql -u root -p

mysql -h host_name_ip -u root -p

Create an admin user account

CREATE USER 'admin'@'localhost' IDENTIFIED BY
'the_secure_password';
```

Creating a Database

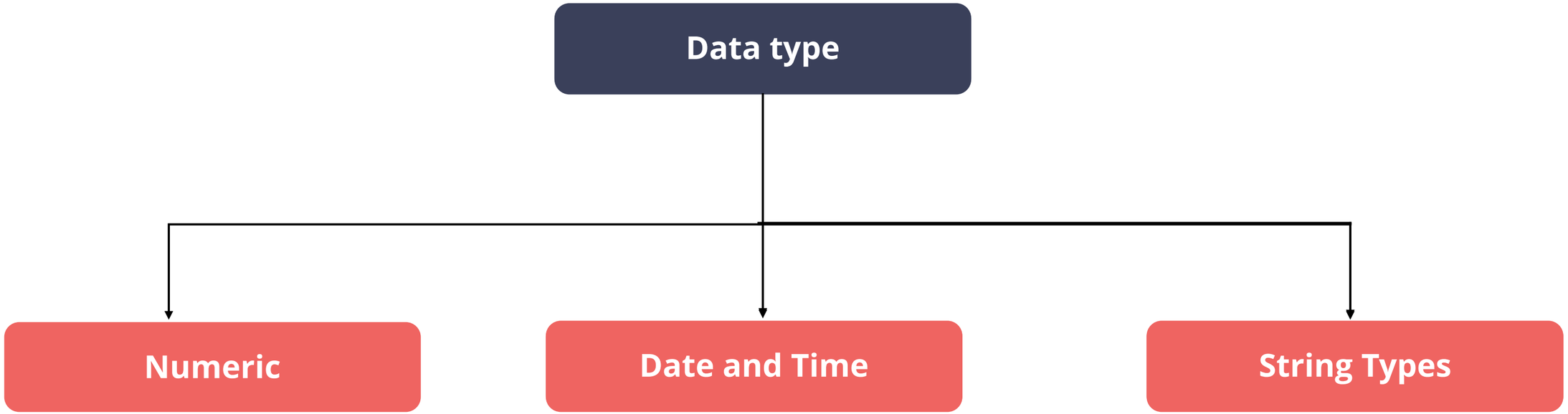
The syntax to create a database:

```
CREATE DATABASE databasename;
```

Creating a Table

Creating a table involves understanding data types.

A data type specifies a particular type of data.



Creating a Database

SQL supports all numeric data types, that include:

Data types	Description	Signed	Unsigned	Width
INT	integer	Permissible reach: 2147483648 - 2147483647	Permissible reach: 0 - 4294967295	11
TINYINT	Small integers	Permissible reach: -128 - 127	Permissible reach: 0 - 255	4
SMALLINT	Small integers	Permissible reach: -32768 - 32767	Permissible reach: 0 – 65535	5
MEDIUMINT	Medium-sized integers	Permissible reach: -8388608 - 8388607	Permissible reach: 0 – 16777215	9
BIGINT	Large integer	Permissible reach: 9223372036854775808 - 9223372036854775807	Permissible reach: 0 - 18446744073709551615	20

Numeric Data Type

SQL supports all numeric data types, that include:

FLOAT(M,D)

- Floating-point numbers (unsigned)
- Defines the visual length (M) and the number of decimals (D)
- Decimal accuracy: 24

DOUBLE(M,D)

- Double-precision floating-point numbers (unsigned)
- Visual length (M): 16
- Quantity of decimals (D): 4
- Decimal accuracy: 53

DECIMAL(M,D)

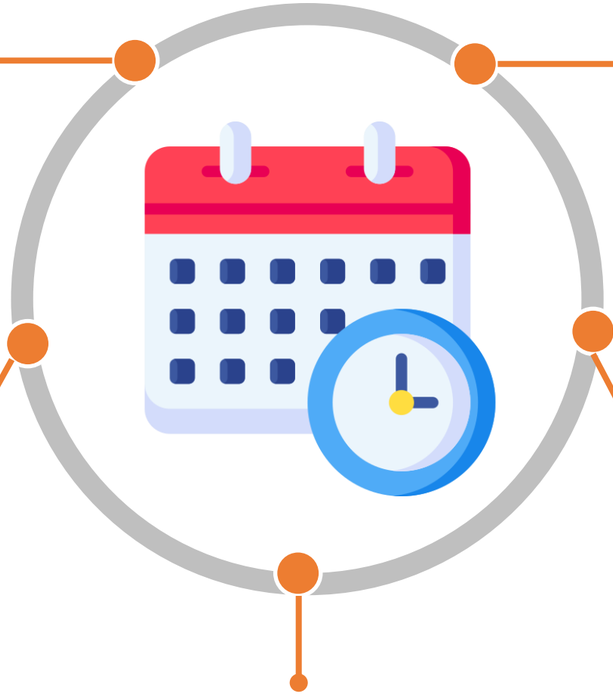
- Unpacked floating-point number (unsigned)
- Every decimal corresponds to 1 byte
- NUMERIC is equivalent to DECIMAL

Date and Time Data Types

The list of the date and time data types are as follows:

YEAR(M): Stored in a two-digit or a four-digit format

DATE: Written in YYYY-MM-DD format



TIME: Stored in HH:MM:SS format

DATETIME: Written in YYYY-MM-DD HH:MM:SS format

TIMESTAMP – Stored in YYYYMMDDHHMMSS format

String Data Types

The list of string data types are as follows:

CHAR(M)

- Fixed-length string
- **Length:** 1 and 255 characters

VARCHAR(M)

- Variable-length string
- **Length:** 1 and 255 characters

BLOB or TEXT

- **Length:** 65535 characters
- Stands for Binary Large Objects
- Helps to store large amounts of binary data

String Data Types

The list of string data types are as follows:

TINYBLOB or TINYTEXT

- BLOB or TEXT column
- **Length:** 255 characters
- Cannot define size

MEDIUMBLOB or MEDIUMTEXT

- BLOB or TEXT column
- **Length:** 16777215 characters
- Cannot define size

LONGBLOB or LONGTEXT

- BLOB or TEXT column
- **Length:** 4294967295 characters
- Cannot define size

ENUM or enumeration refers to an extravagant term for lists.

Creating a Table: Syntax

The syntax for creating a table:

```
CREATE TABLE name_of_table (
column1 datatype,
column2 datatype,
column3 datatype,
   ....
);
```

Creating a Table: Example

Code

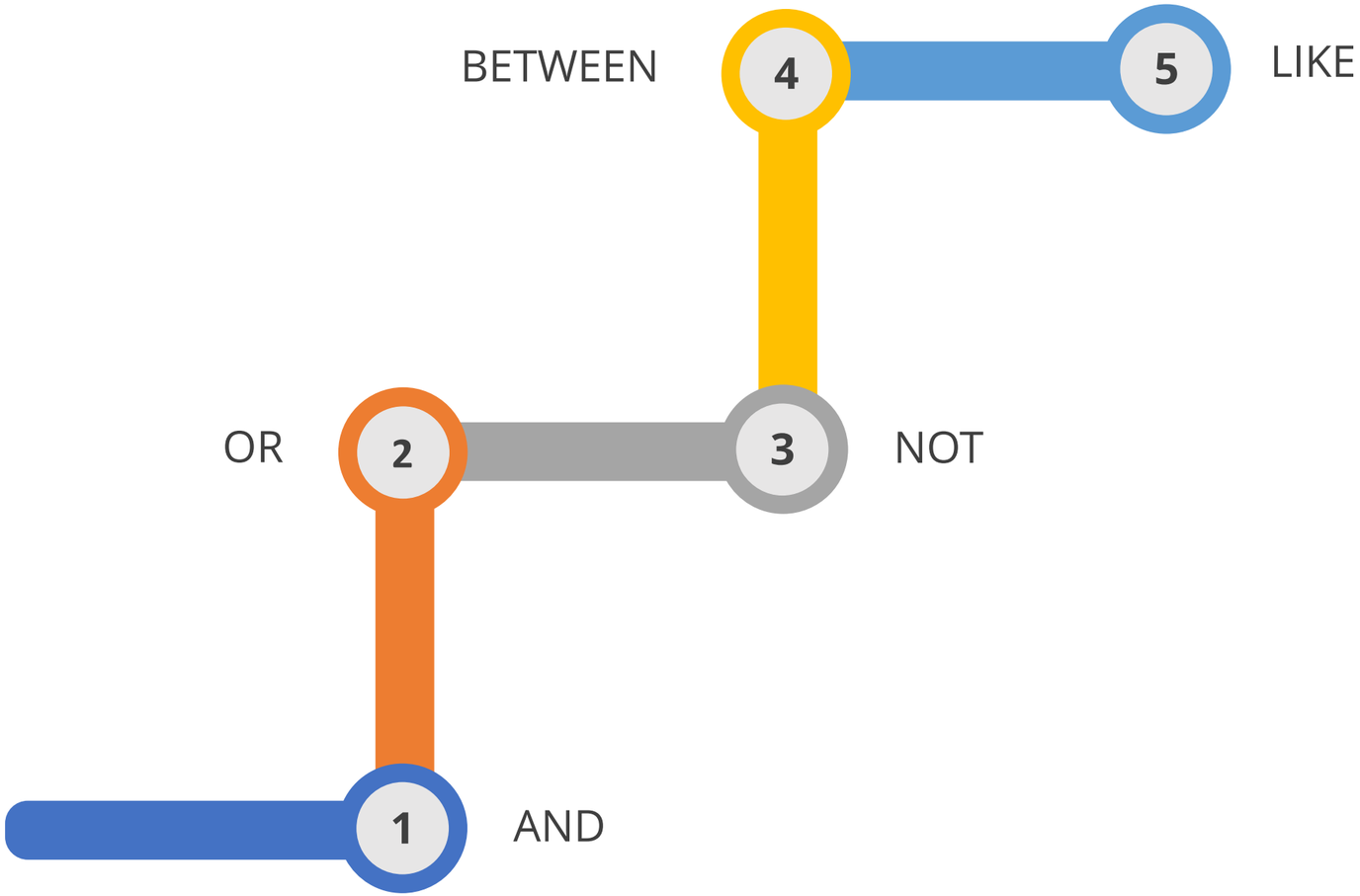
```
CREATE TABLE Persons (
PersonID int,
LastName varchar(255),
FirstName varchar(255),
Address varchar(255),
City varchar(255)
)
```

Output

PersonID	LastName	FirstName	Address	City

Logical Operators

The following are the different types Logical Operators:



Logical Operators: AND

The AND operator displays a record if all the conditions included within the AND operator are satisfied.

The following is the syntax for the AND operator:

SQL Query

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 AND condition2 AND condition3 ...;
```

Logical Operators: OR

The OR operator displays a record if any of the conditions included within the OR operator are satisfied.

The following is the syntax for the OR operator:

SQL Query

```
SELECT column1, column2, ...
FROM table_name
WHERE condition1 OR condition2 OR condition3 ...;
```

Logical Operators: NOT

The NOT operator displays a record if the condition is not satisfied.

The following is the syntax for the NOT operator:

SQL Query

```
SELECT column1, column2, ...
FROM table_name
WHERE NOT condition;
```

Logical Operators: BETWEEN

The BETWEEN operator is used to filter data within the specified range.

The following is the syntax for the BETWEEN operator:

SQL Query

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

Logical Operators: LIKE

The LIKE operator is used to search for a specified pattern in a column.

The following is the syntax for the LIKE operator:

SQL Query

```
SELECT column1, column2, ...
FROM table_name
WHERE column LIKE pattern;
```

Assisted Practice

Creating Databases and Tables **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate the process of creating, modifying, and managing databases and tables using MySQL commands in a terminal.

Assisted Practice: Guidelines

Steps to be followed:

1. Create database and tables

Assisted Practice

Implementing Logical Operators **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate the application of logical operators (AND, OR, NOT) in MySQL queries.

Assisted Practice: Guidelines

Steps to be followed:

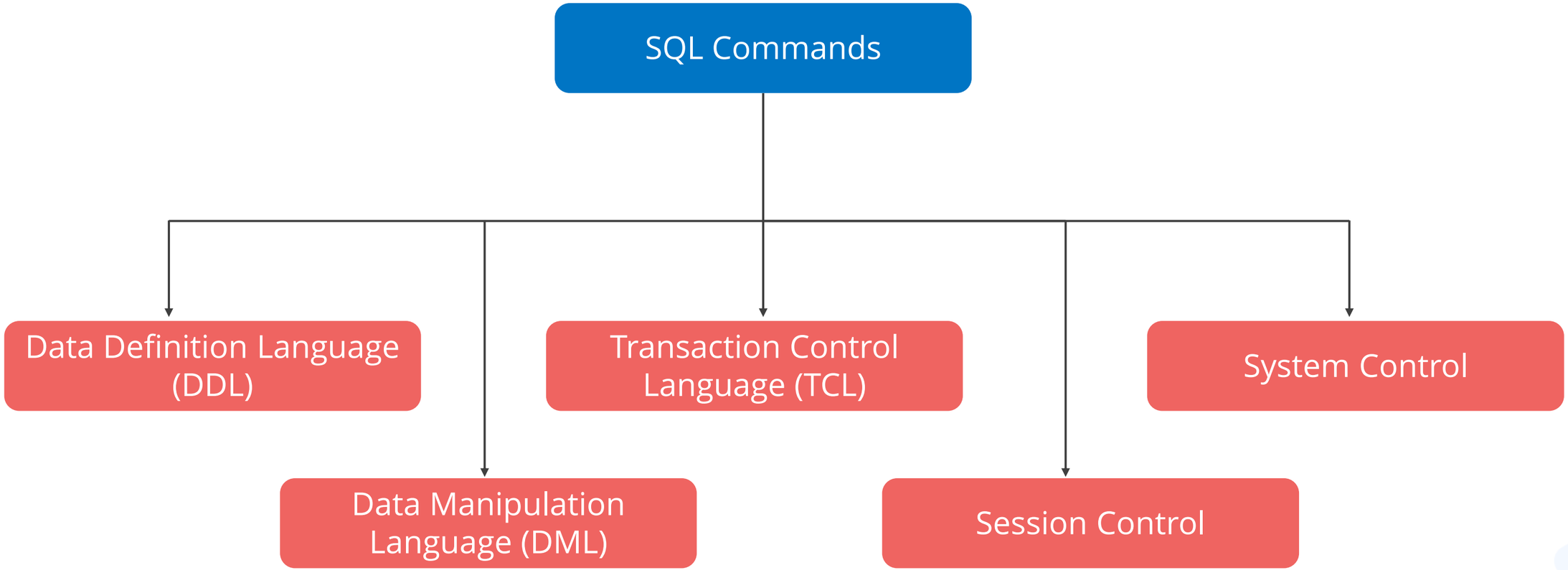
1. Setup database and table
2. Apply logical operators

ASSISTED PRACTICE

SQL Commands

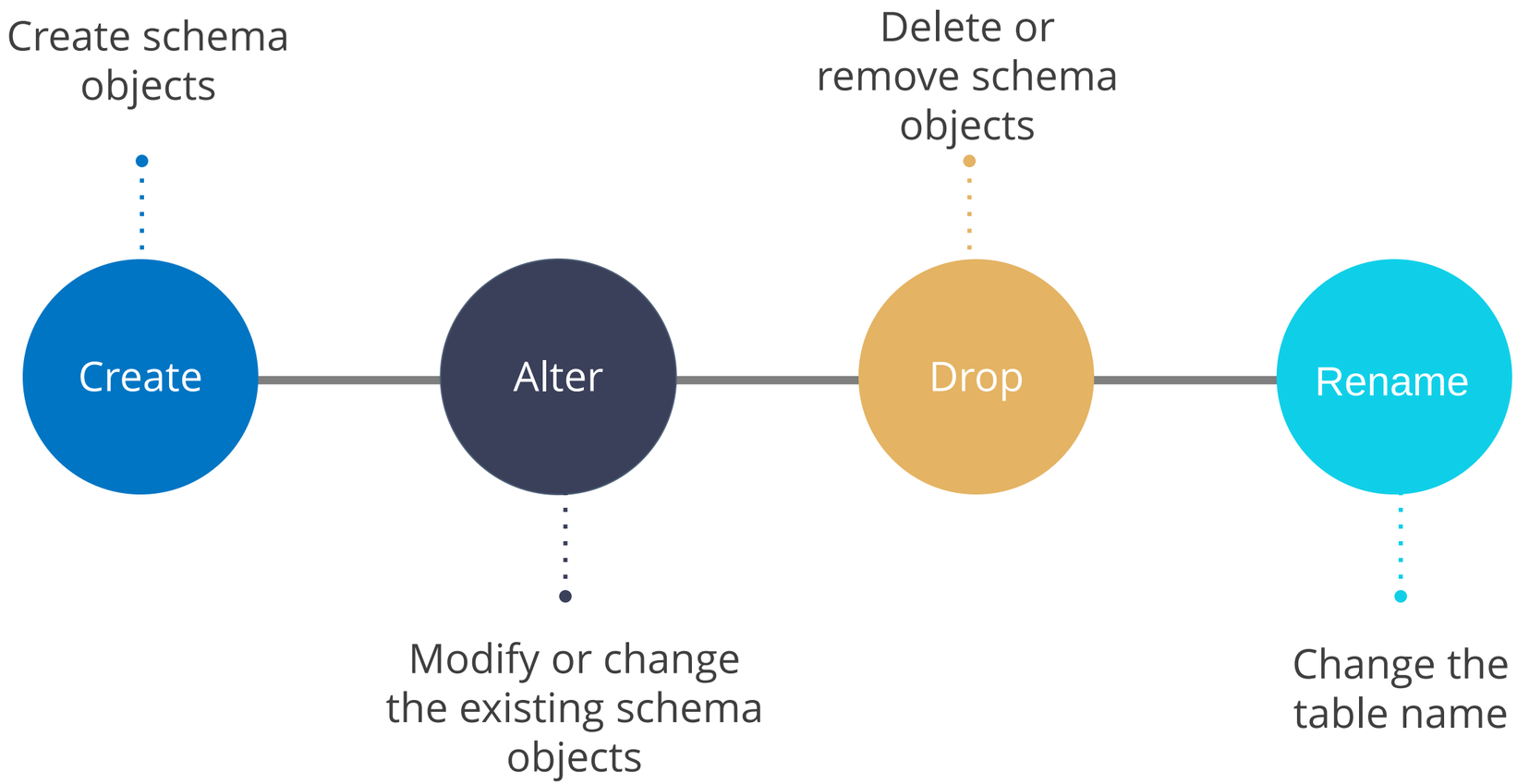
SQL Command Categories

SQL commands can be divided into five categories:



Data Definition Language (DDL) Commands

The DDL commands help the user perform data definitions tasks.



Data Definition Language (DDL) Commands

The DDL commands are also known as Data Control Language (DCL).

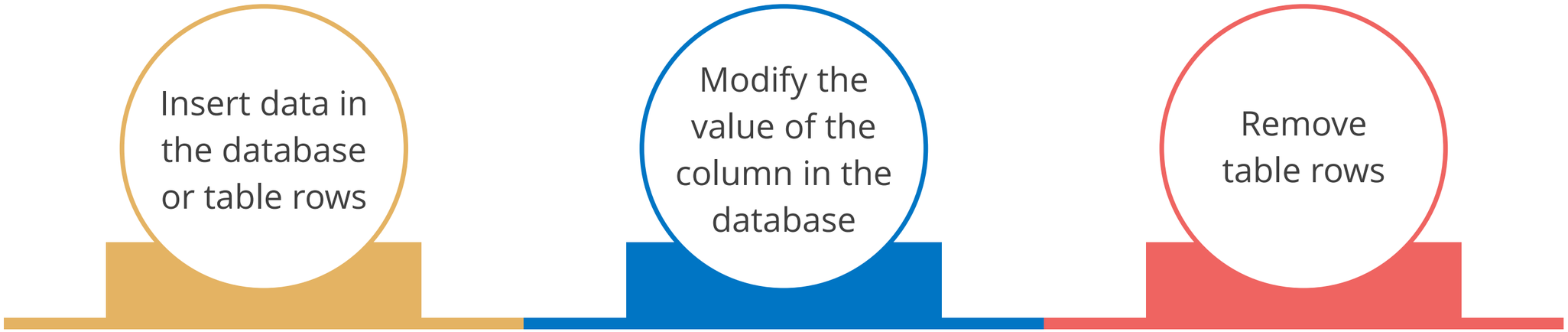
Grant or revoke permissions or privileges to work on schema objects

Analyze the table information

Data Manipulation Language (DML) Commands

The DML commands are employed to manipulate and modify data, for example, SELECT, and LOCK table.

This command is not permanently saved.



Transaction Control Language (TCL) Commands

Transaction control language is used to manage and manipulate the data generated by:

INSERT< UPDATE< DELETE commands

A transaction refers to one complete logical unit of work.

These commands are used to manage changes that are made by DML commands.

Transaction Control Language (TCL) Commands

- Makes all changes made by a statement issued
- Makes the transaction permanent

Undo the changes from the beginning or a save point

Saves the transaction temporarily

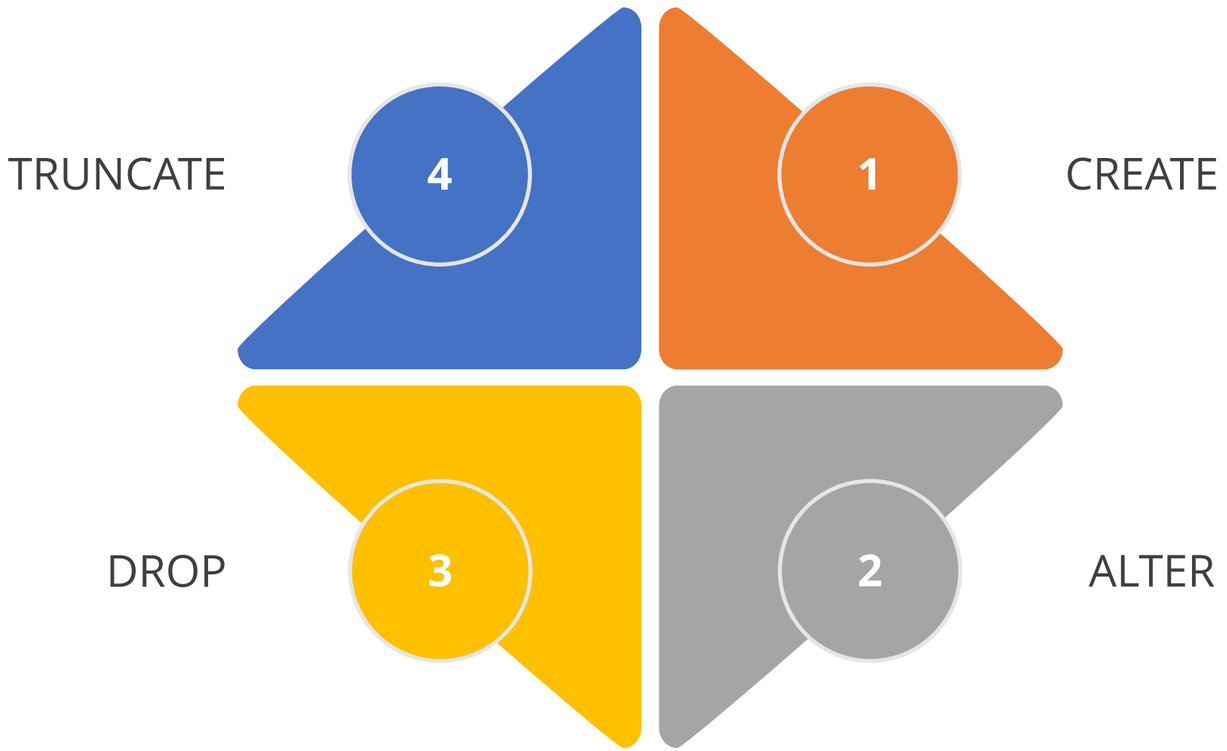
Implements properties for the current transaction

Data Definition Language (DDL) Commands

Data Definition Language (DDL) commands are a set of SQL commands that are used for creating, modifying, and deleting database structures.

Data Definition Language (DDL) Commands

The following is the list of DDL commands:



Data Definition Language (DDL) Commands: CREATE

The CREATE command is a data definition language command that is used for creating databases or its objects like database tables, database indexes, functions, and views.

Data Definition Language (DDL) Commands: CREATE

The following is the syntax for creating a database:

SQL Query

```
CREATE DATABASE database_name;
```

Data Definition Language (DDL) Commands: ALTER

The ALTER command is a data definition language command that is used to change the structure of a database by adding, renaming, or modifying columns in an existing database table, as well as dropping or deleting columns.

Data Definition Language (DDL) Commands: ALTER

The following is the syntax for altering a table in a database by adding a new column:

SQL Query

```
ALTER TABLE table_name
ADD (Columnname_1  datatype)
```

Data Definition Language (DDL) Commands: DROP

The DROP command is a data definition language command for deleting database objects. It can also be used to delete a database.

Note:

A rollback procedure cannot be performed on a dropped database or table command.

Data Definition Language (DDL) Commands: DROP

The following is the syntax for deleting a database using the DROP command:

SQL Query

```
DROP DATABASE database_name;
```

The following is the syntax for deleting a table using the DROP command:

SQL Query

```
DROP TABLE table_name;
```

Data Definition Language (DDL) Commands: TRUNCATE

The TRUNCATE command is a data definition language command that is used to remove all the records from a table. It deletes all the spaces allocated for the records permanently.

Note:

A rollback procedure cannot be performed on a truncated command.

Data Definition Language (DDL) Commands: TRUNCATE

The following is the syntax for using a TRUNCATE command:

SQL Query

```
TRUNCATE TABLE table_name;
```

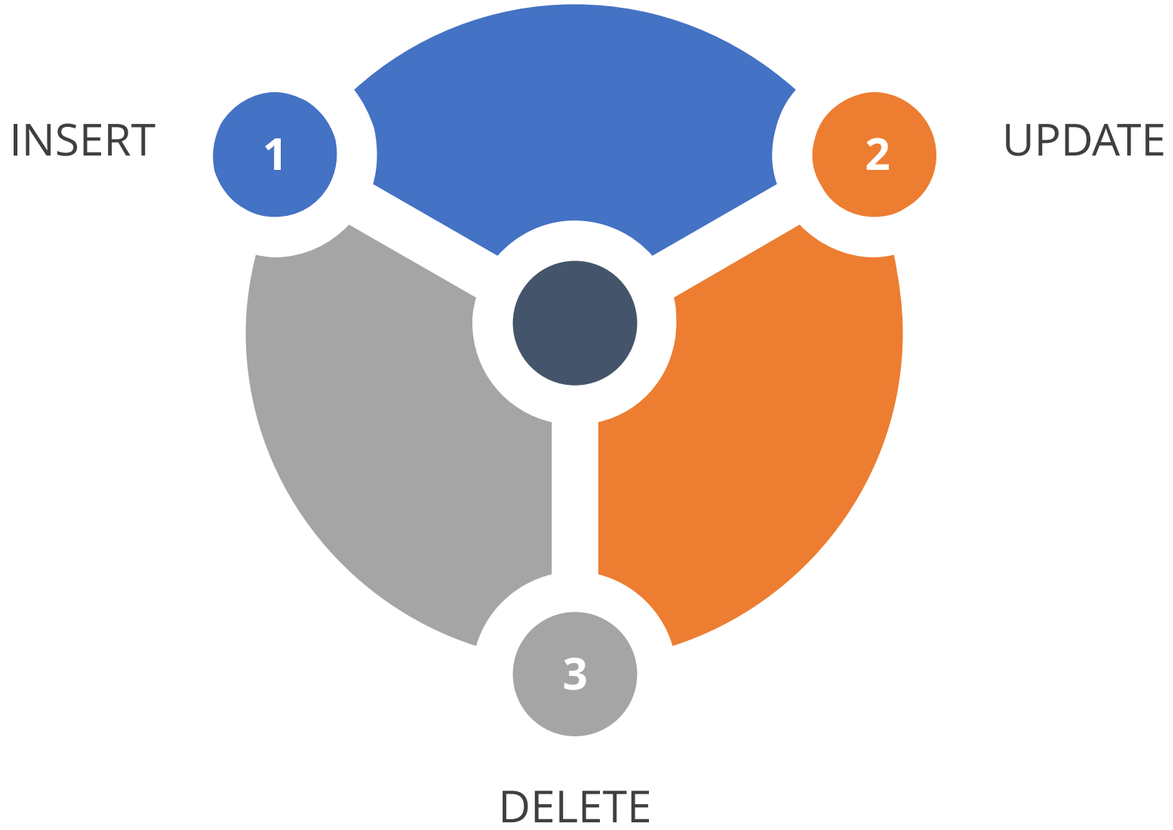
Data Manipulation Language (DML) Commands

Data Manipulation Language (DML) commands are a set of SQL commands that allows to insert, alter, and delete data from a database.

These commands are not automatically committed, so they can be rolled back.

Data Manipulation Language (DML) Commands

The list of DML commands includes:



Data Manipulation Language (DML) Commands: INSERT

The INSERT command is a data manipulation language command that is used to insert data into a database table.

Data Manipulation Language (DML) Commands: INSERT

The following is the syntax of the INSERT command:

SQL Query

```
INSERT INTO TABLE_NAME (column1, column2, column3,…columnN)
VALUES (value1, value2, value3,…valueN);
```

Data Manipulation Language (DML) Commands: UPDATE

The UPDATE command is a data manipulation language command that is used for modifying or updating existing data in a database table.

Data Manipulation Language (DML) Commands: UPDATE

The following is the syntax of the UPDATE command:

SQL Query

```
UPDATE Table_name
SET [column_name1= value_1, ….., column_nameN = value_N]
WHERE CONDITION;
```

Data Manipulation Language (DML) Commands: DELETE

The DELETE command is a data manipulation language command that is used for deleting one or multiple records from a database table.

Note:

The DELETE command does not remove the stored data from the database permanently.

Data Manipulation Language (DML) Commands: DELETE

The following is the syntax of the DELETE command:

SQL Query

```
DELETE FROM Table_Name WHERE condition;
```

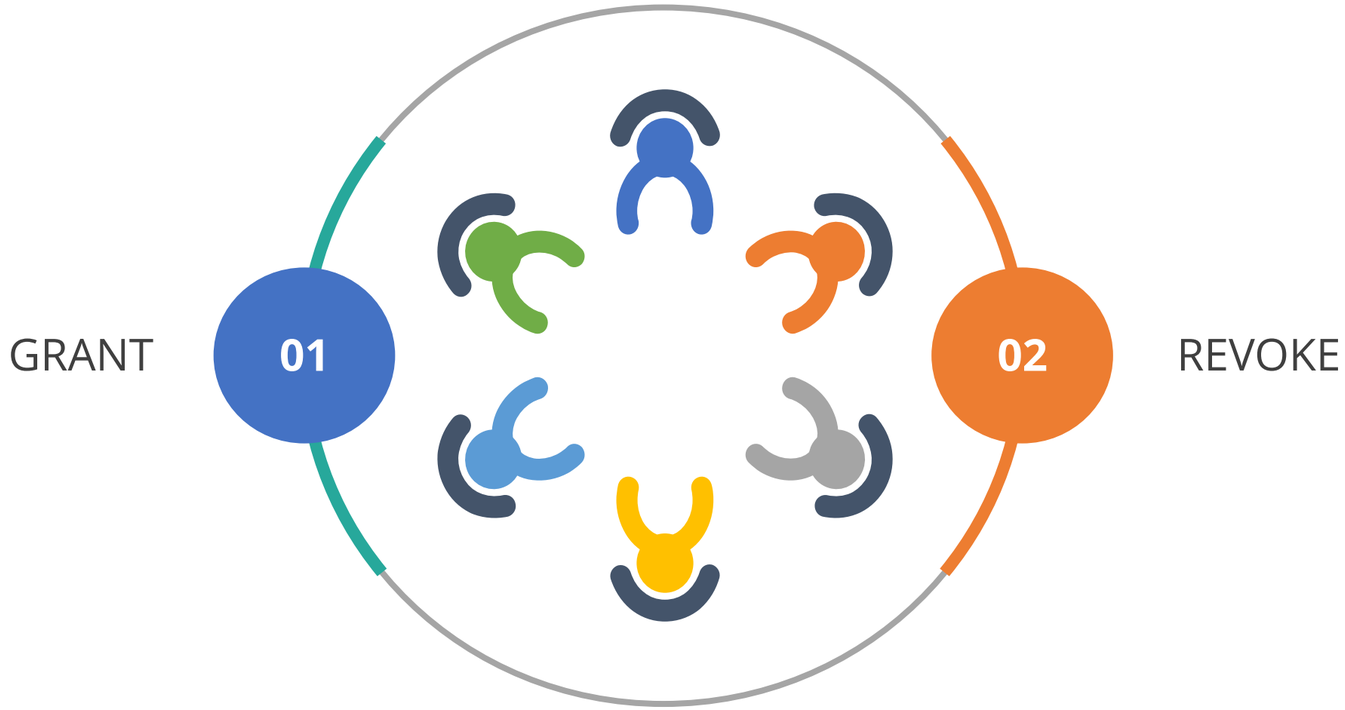
Data Control Language (DCL) Commands

Data Control Language (DCL) commands are a set of SQL commands that is used to grant or revoke database user authority.

These commands handle the rights, permissions, and other controls of the database system.

Data Control Language (DCL) Commands

The following is the list of DCL commands:



Data Control Language (DCL) Commands: GRANT

The GRANT command is a data control language command that grants database access to users.

Data Control Language (DCL) Commands: GRANT

The following is the syntax of the GRANT command:

SQL Query

```
GRANT privilege_name on objectname to user;
```

Data Control Language (DCL) Commands: REVOKE

The REVOKE command is a data control language command that allows one to withdraw any user's access.

Data Control Language (DCL) Commands: REVOKE

The following is the syntax of the REVOKE command:

SQL Query

```
REVOKE privilege_name on objectname from user;
```

Transaction Control Language (TCL) Commands

The Transaction Control Language (TCL) commands are a set of SQL commands that is used to manage database transactions.

These commands can only be used with DML commands such as INSERT, DELETE, and UPDATE.

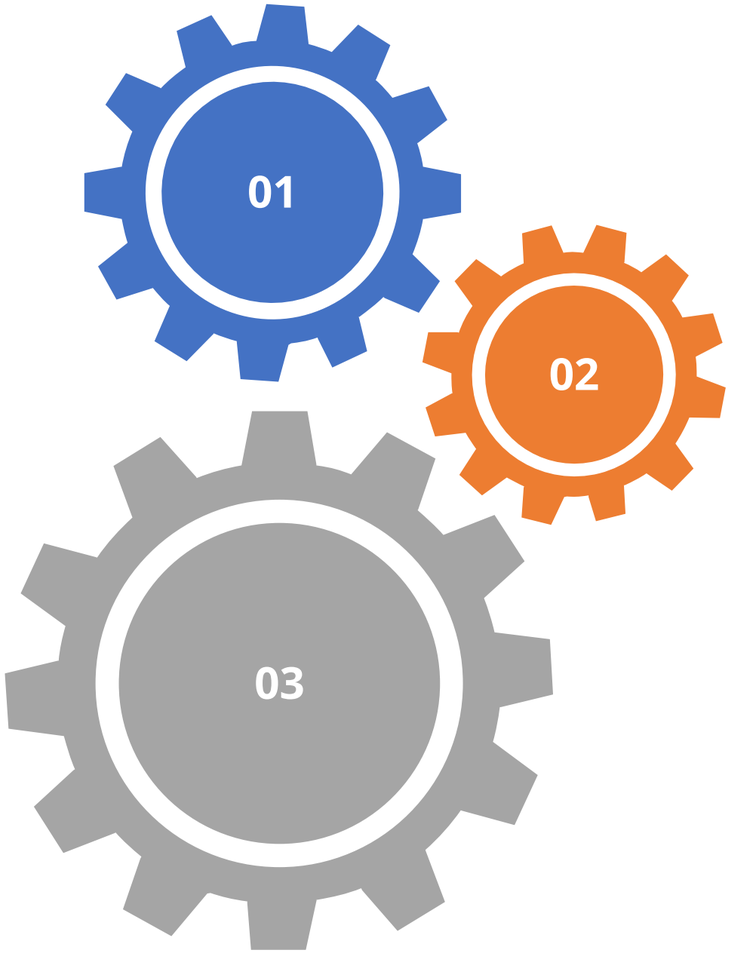
Transaction Control Language (TCL) Commands

The following is the list of TCL commands:

COMMIT

ROLLBACK

SAVEPOINT



Transaction Control Language (TCL) Commands: COMMIT

The COMMIT command is a transaction control language command that permanently saves all updates related to transactions in the database.

Transaction Control Language (TCL) Commands: COMMIT

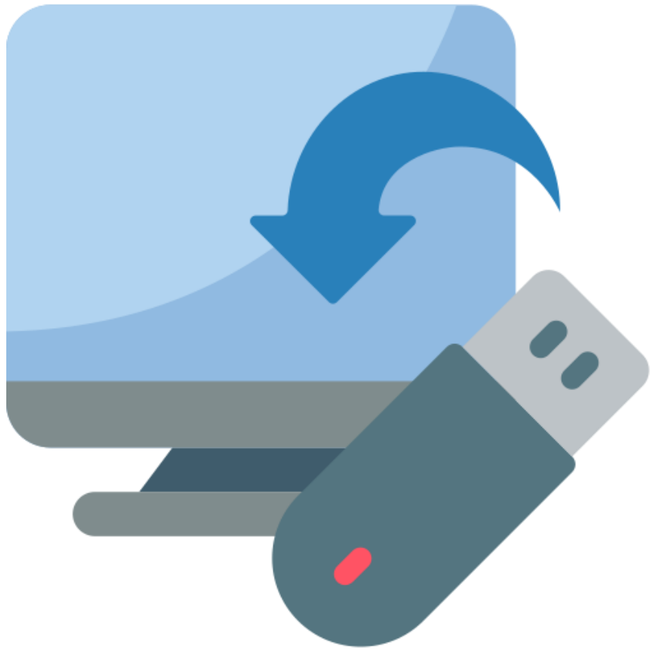
The following is the syntax of the COMMIT command:

SQL Query

```
COMMIT;
```

Transaction Control Language (TCL) Commands: ROLLBACK

The ROLLBACK command is a transaction control language command that restores data to the state it was last committed to.



Transaction Control Language (TCL) Commands: ROLLBACK

The following is the syntax of the ROLLBACK command:

SQL Query

```
ROLLBACK TO savepoint_name;
```

Transaction Control Language (TCL) Commands: SAVEPOINT

The SAVEPOINT command is a transaction control language command that is used to set a savepoint within a transaction. It may be rolled back to the starting point whenever required.

Transaction Control Language (TCL) Commands: SAVEPOINT

The following is the syntax of the SAVEPOINT command:

SQL Query

```
SAVEPOINT [savepoint_name;]
```

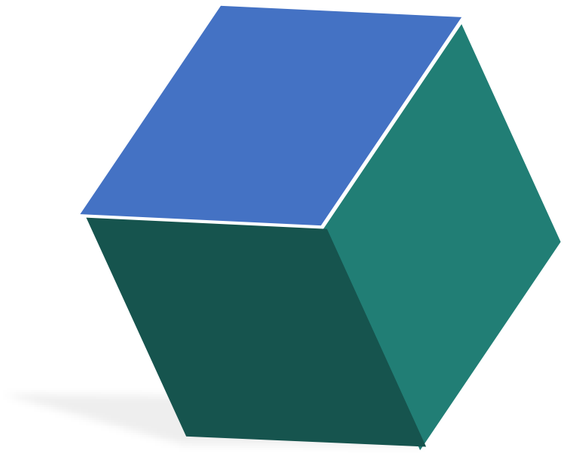
Data Query Language (DQL) Command

The Data Query Language (DQL) command is an SQL command that is used for retrieving data from a database.

These commands allow user to get data from the database and organize it.

Data Query Language (DQL) Command

It uses a single command:



SELECT

Data Query Language (DQL) Commands: SELECT

The SELECT command is a data query language command that retrieves data from the database depending on the WHERE clause condition.

Data Query Language (DQL) Commands: SELECT

The following is the syntax of the SELECT command:

SQL Query

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

Assisted Practice

Inserting, Updating, and Deleting Records **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate the basic operations of inserting, updating, and deleting records in a MySQL table.

Assisted Practice: Guidelines

Steps to be followed:

1. Setup database and table
2. Insert records
3. Update records
4. Delete records

ASSISTED PRACTICE

Assisted Practice

Using Select Statement with Various Clauses **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate the versatility of the SELECT statement in MySQL by utilizing various clauses for refined data retrieval.

Assisted Practice: Guidelines

Steps to be followed:

1. Setup database and table
2. Apply Select with various clauses

Assisted Practice

Demonstrating Transaction Control Language (TCL) Commands **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate the use of Transaction Control Language (TCL) commands in MySQL, focusing on managing and controlling transactions.

Assisted Practice: Guidelines

Steps to be followed:

1. Setup database and table
2. Demonstrate TCL commands

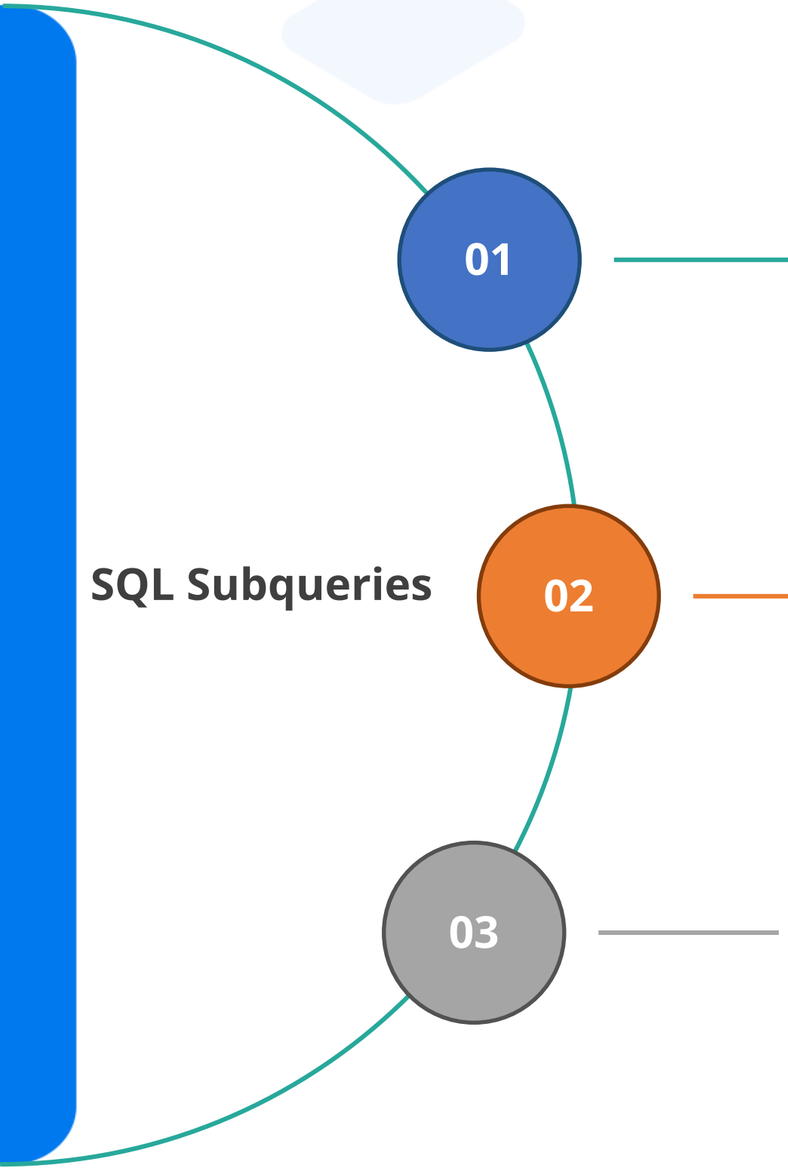
SQL Subqueries and Keys

Subqueries

A subquery is a query that is contained within a bigger query.

These SQL queries are contained inside the WHERE clause of other SQL queries.

Subqueries



It is also called **Inner Query** or **Inner Select**, while the statement that contains the subquery is called an **Outer query** or **Outer select**.

It can be used wherever an expression is used and must be closed in parentheses.

It can be used within a SELECT, INSERT, UPDATE, or DELETE statement.

Subqueries with the SELECT Statement

The SELECT statement is widely used with subqueries.

Subqueries with the SELECT Statement

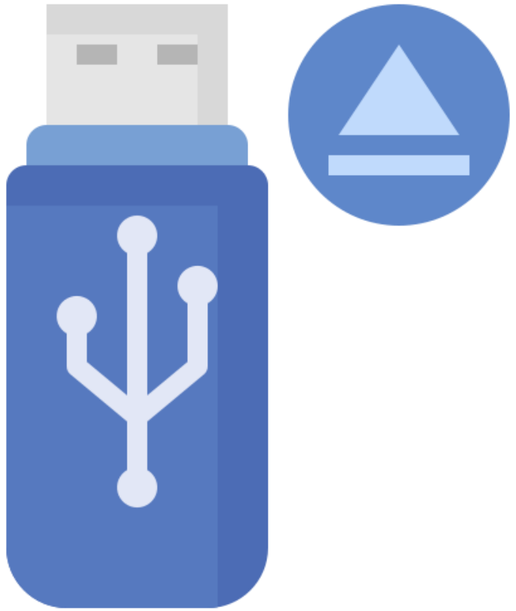
The following is the syntax of subqueries with the SELECT statement:

SQL Query

```
SELECT column_name [, column_name ]
FROM table1 [, table2 ]
WHERE column_name OPERATOR
(SELECT column_name [, column_name ]
FROM table1 [, table2 ]
[WHERE])
```

Subqueries with the INSERT Statement

INSERT statements can also utilize subqueries. The INSERT statement inserts data from a subquery into another table.



Subqueries with the INSERT Statement

The following is the syntax of subqueries with the INSERT statement:

SQL Query

```
INSERT INTO table_name [ (column1 [, column2 ]) ]
SELECT [ *|column1 [, column2 ]
FROM table1 [, table2 ]
[ WHERE VALUE OPERATOR ]
```

Subqueries with the UPDATE Statement

UPDATE statements can also utilize subqueries. Users can update a single or several columns in a table with the UPDATE statement.

Subqueries with the UPDATE Statement

The following is the syntax of subqueries with the UPDATE statement:

SQL Query

```
UPDATE table
SET column_name = new_value
[ WHERE OPERATOR [ VALUE ]
(SELECT COLUMN_NAME FROM TABLE_NAME)
[ WHERE) ]
```

Subqueries with the DELETE Statement

DELETE statements can also utilize subqueries.

Subqueries with the DELETE Statement

The following is the syntax of subqueries with the DELETE statement:

SQL Query

```
DELETE FROM TABLE_NAME
[ WHERE OPERATOR [ VALUE ]
(SELECT COLUMN_NAME
FROM TABLE_NAME)
[ WHERE) ]
```

MySQL Where Clause

This is used to filter out the results.

Syntax:

```
SELECT field1, field2,...fieldN name_of_table1,
name_of_table2...
[WHERE condition1 [AND [OR]] condition2.....
```

It allows specifying selection criteria to select the required records.

MySQL AND/OR Operator

Below is the syntax for MySQL AND operator:

Syntax:

```
A AND B
```

	TRUE	FALSE	NULL
TRUE	TRUE	FALSE	NULL
FALSE	FALSE	FALSE	FALSE
NULL	NULL	FALSE	NULL

MySQL AND/OR Operator

Below is the syntax for MySQL OR operator:

Syntax:

```
A OR B
```

	TRUE	FALSE	NULL
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	NULL
NULL	TRUE	NULL	NULL

MySQL OrderBy Keyword

It is used to sort the outcome in an ascending or a descending order.

Syntax:

```
SELECT column1, column2, ...
FROM name_of_table
ORDER BY column1, column2, ...
ASC|DESC;
```

MySQL GroupBy

It is used for collecting data from multiple records and grouping the output.

This is mostly used in SELECT statements along with SUM, COUNT, and MAX.

Syntax:

```
SELECT expression_a, expression_b, ...
expression_n,
aggregate_function (expression)
FROM tables
[WHERE conditions]
GROUP BY expression_a, expression_b, ...
expression_n;
```

MySQL Having

It is used by the GROUP clause. It returns the rows where the condition is TRUE.

Syntax:

```
SELECT expression_a, expression_b, ...
expression_n,
aggregate_function (expression)
FROM tables
[WHERE conditions]
GROUP BY expression_a, expression_b, ...
expression_n
HAVING condition;
```

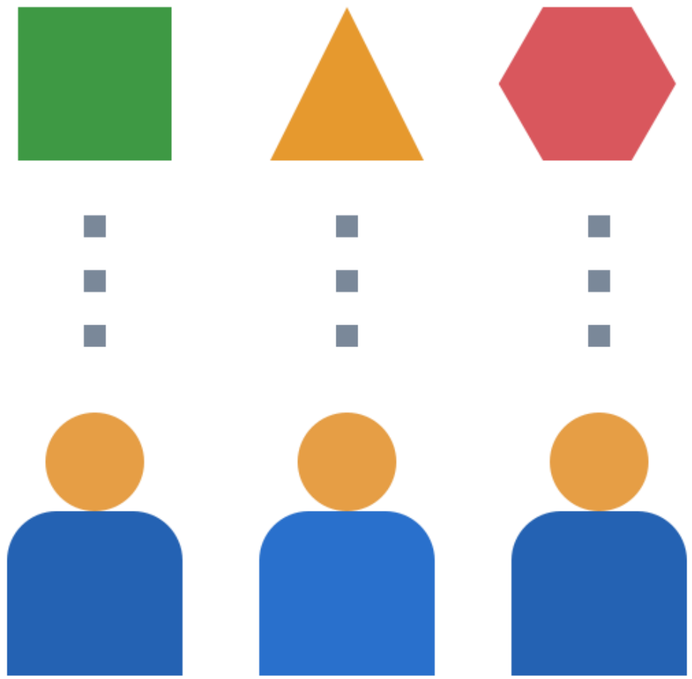
MySQL Keys

MySQL offers four keys:

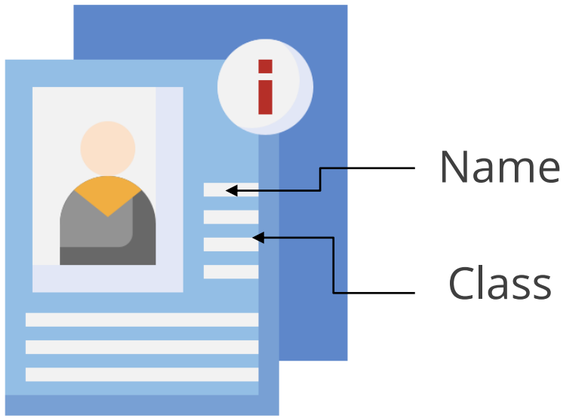
- Foreign Key
- Unique Key
- Primary Key
- Composite Key

Unique Key

In MySQL, there is a single field or combination of fields that makes sure all values that are going to be stored in the column are unique, which means the column cannot store the same values.



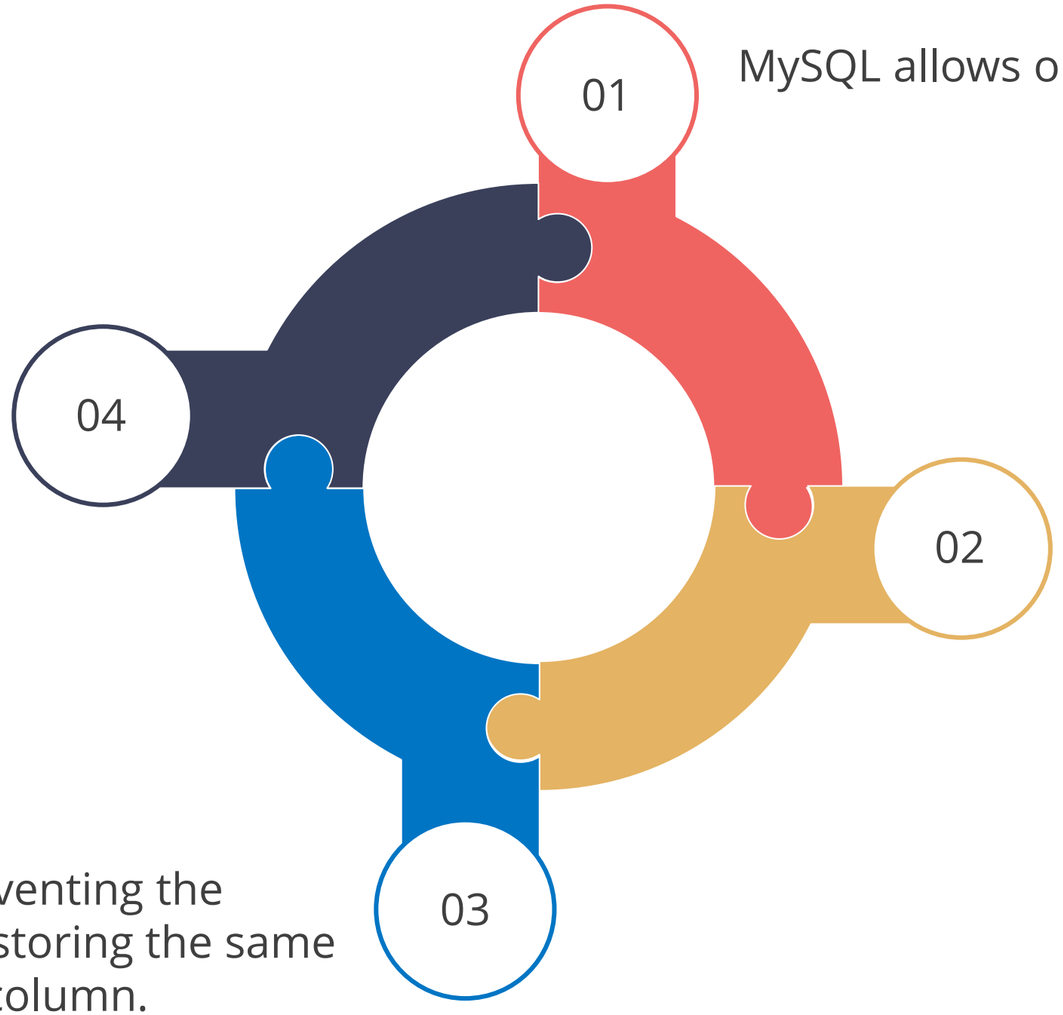
Example:



student-info

Unique Key

MySQL allows to use more than one column with UNIQUE values in the table.



01 MySQL allows only a null value.

02 It makes sure the columns store different values.

03 It helps in preventing the records from storing the same values in the column.

04 It stores only different values to maintain the integrity of the database.

Unique Key

The syntax to create a unique key in the table is:

```
CREATE TABLE table_name(
    col1 data_type,
    col2 data_type UNIQUE,
    ...
);
```

The syntax to insert multiple unique keys into a table is:

```
CREATE TABLE table_name(
  col1 column_definition,
  col2 column_definition,
  ...
  [CONSTRAINT constraint_name]
  UNIQUE(col_name(s))
);
```

If a unique constraint is not specified, SQL gives a name for that column.

Composite Keys

The composite key can be added in two different ways:

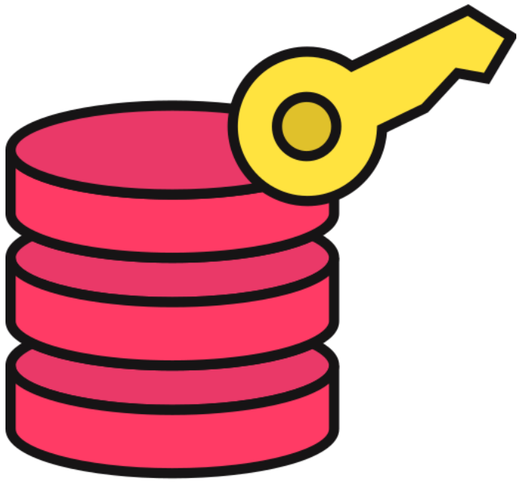
Create statement

Alter statement

Primary Keys

It helps to uniquely recognize each record in a table.

It contains unique values and not NULL values.



Syntax:

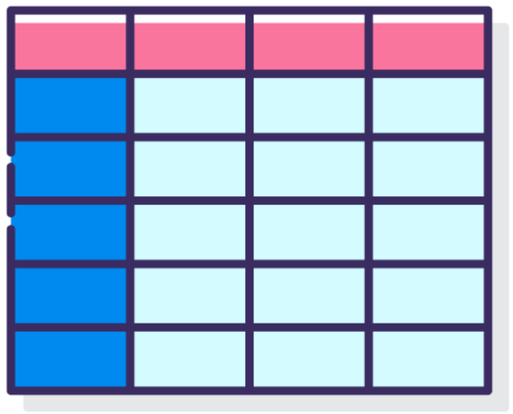
```
CREATE TABLE Persons (
ID int NOT NULL,
LastName varchar(255) NOT NULL,
FirstName varchar(255),
Age int,
    CONSTRAINT PK_Person PRIMARY KEY
(ID,LastName)
);
```

Foreign Keys

It is used to prevent actions that would impact connections between tables.

Foreign Key

Child Table



Parent Key

Referenced or
Parent Table

Foreign Keys

The syntax to create a Foreign Key on the ProductId column when the Orders table is created:

```
CREATE TABLE Orders (
OrderID int NOT NULL,
OrderNumber int NOT NULL,
ProductID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (ProductId) REFERENCES
Persons(ProductId)
);
```

MySQL Distinct Keyword

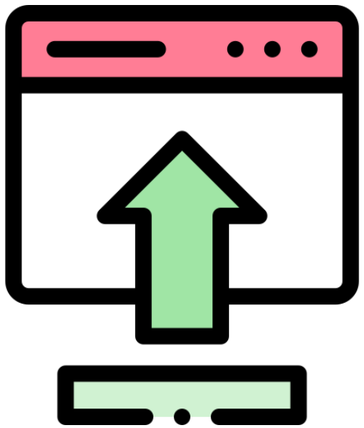
The SELECT DISTINCT statement is used to return different values.

Syntax:

```
SELECT DISTINCT column1, column2, ...
FROM name_of_table;
```

MySQL Insert Into Keyword

The MySQL Insert statement is used to insert single or multiple records into a table.



Syntax:

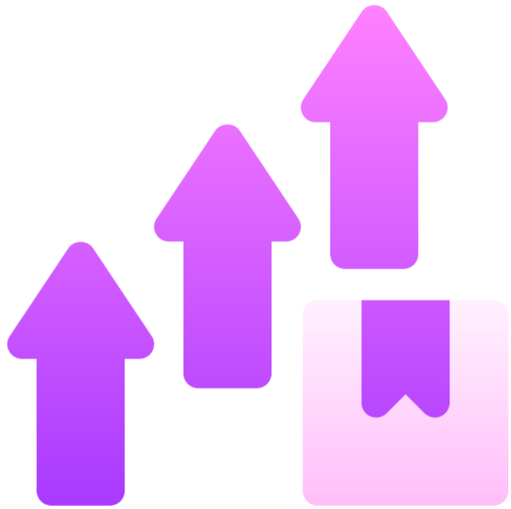
```
INSERT INTO table
(column1, column2, ... )
VALUES
(expression1, expression2, ... ),
(expression1, expression2, ... ),
...;
```

MySQL Get Last Inserted ID

The LAST_INSERT_ID() function returns the AUTO_INCREMENT ID of the last row added or updated in a table.

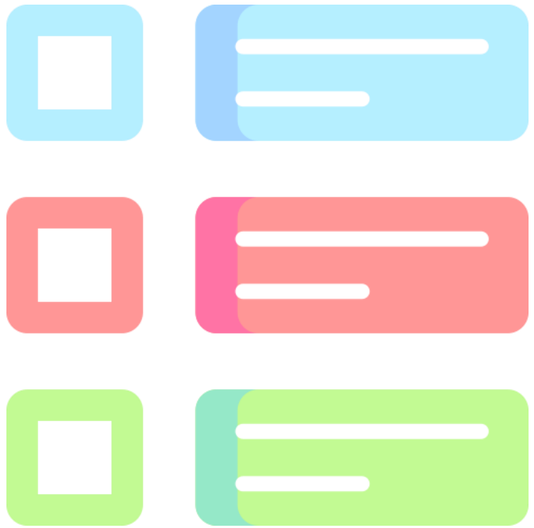
Syntax:

```
LAST_INSERT_ID(expression)
```



MySQL Insert Multiple Records Command

It is used to insert multiple rows into a table.



Syntax:

```
LAST_INSERT_ID(exINSERT INTO name_of_table
(column_list)
VALUES
(value_list_1),
(value_list_2),
...
(value_list_n);
pression)
```

Assisted Practice

Working with Subqueries and Keys **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate how to effectively utilize subqueries and work with different types of keys in MySQL for complex data operations.

Assisted Practice: Guidelines

Steps to be followed:

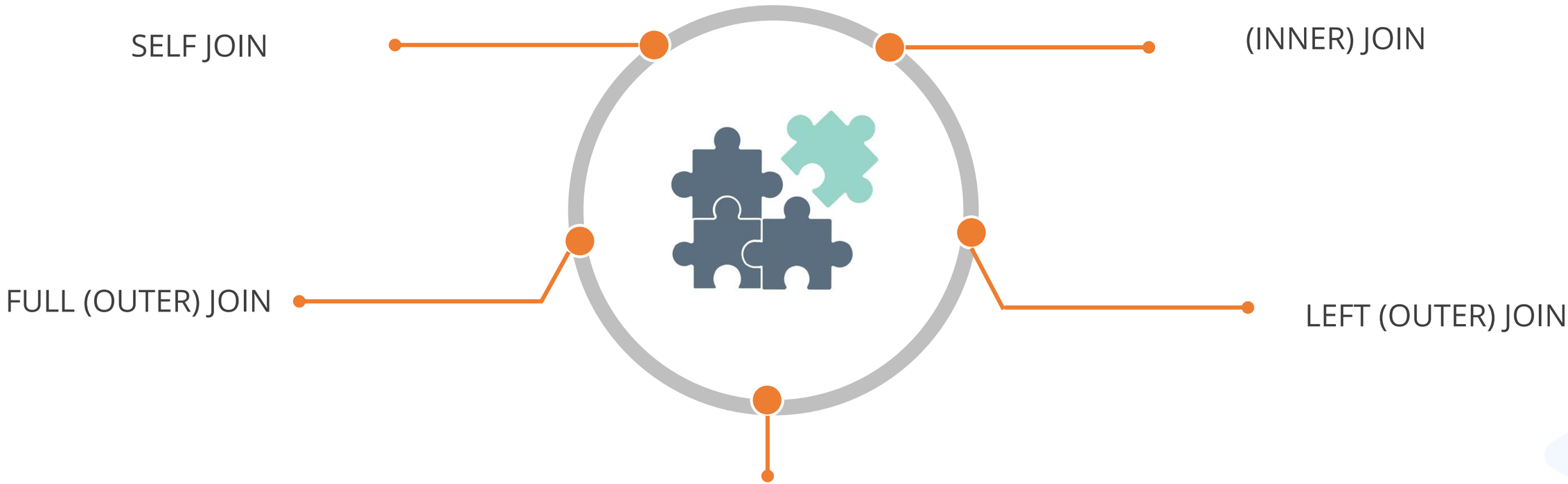
1. Setup database and table
2. Insert data with subquery
3. Use subqueries in SELECT
4. Work with keys

SQL Joins

MySQL Joins

A Join statement is used to merge the rows from two or more tables.

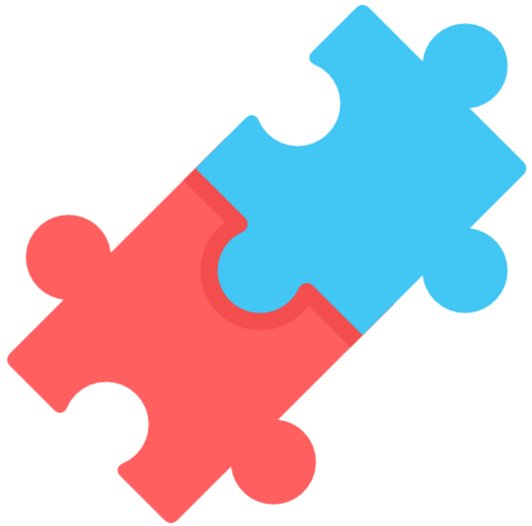
Different types of JOINS are:



Inner Join

It returns records that have the same values in both tables.

The keyword takes the records with matching values in both tables.



Syntax:

```
SELECT column_name_(s)
FROM table1
INNER JOIN table2
ON table1.column_name =
table2.column_name;
```

Left Join

It returns the same records from the right table and all records from the left table.

It shows zero records from the right side if no match is found.

Syntax:

```
SELECT column_name_(s)
FROM table1
LEFT JOIN table2
ON table1.column_name =
table2.column_name;
```

Right Join

It returns the same records from the left table and all records from the right table.

It shows zero records from the left side if no match is found.

Syntax:

```
SELECT column_name_(s)
FROM table1
RIGHT JOIN table2
ON table1.column_name =
table2.column_name;
```

Full Join

It returns all records when the value is the same, either on the right or left side.

Syntax:

```
SELECT column_name_(s)
FROM table1
FULL OUTER JOIN table2
ON table1.column_name = table2.column_name
WHERE condition;
```

Self Join

It is a regular join where the table is joined with itself.

Syntax:

```
SELECT column_name_(s)
FROM table1 T1, table1 T2
WHERE condition;
```

Assisted Practice

Performing Joins on Tables **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to showcase various types of join operations in MySQL, illustrating how to effectively combine data from multiple tables.

Assisted Practice: Guidelines

Steps to be followed:

1. Setup database and table
2. Perform different types of joins

ASSISTED PRACTICE

Assisted Practice

Working with Related Tables **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate the process of managing and querying related tables in MySQL, emphasizing the importance of relational database principles.

Assisted Practice: Guidelines

Steps to be followed:

1. Setup database and table
2. Query related tables

Built-in Functions

Model

Math functions Math functions can be used in SQL queries to perform calculations on numeric data. These functions can be combined with other SQL functions and operators to perform more complex calculations.

Aggregate functions An aggregate function performs a calculation on a set of values, and returns a single value

Built-in Functions: Overview

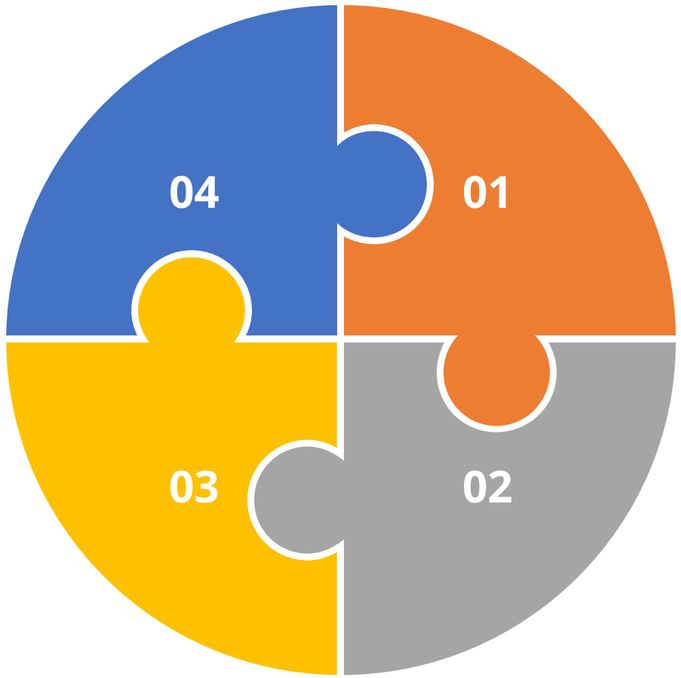
A built-in function is a function that accepts zero or more inputs and returns an output.

Built-in Functions: Overview

The following are the different types of built-in functions:

Date functions

Math functions

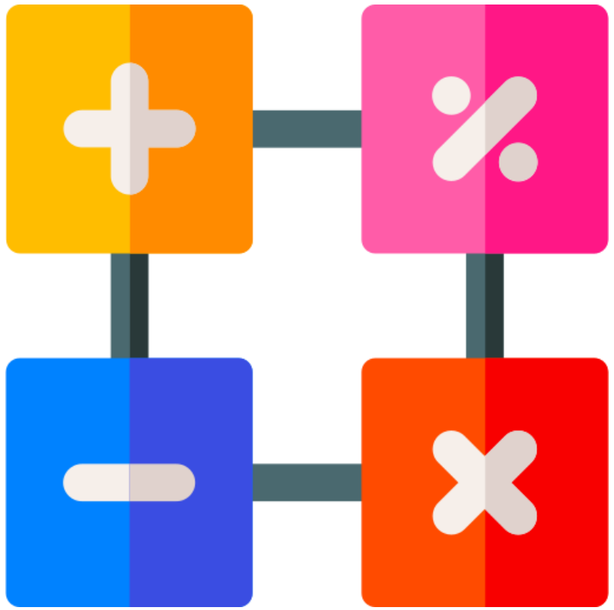


String functions

Aggregate functions

Math Functions

A Math function in SQL is used to execute arithmetic operations.



Math Functions

The following are the different types of Math functions:

Math Functions: MOD()

The MOD() function is used to return a remainder of a number that is divided by another number.

The following is the syntax for the MOD() function:

SQL Query

```
SELECT MOD(x, y);
```

Math Functions: POWER()

The POWER() function is used to find the value of a number that has been raised to the power of another number.

The following is the syntax for the POWER() function:

SQL Query

```
SELECT POWER(x, y);
```

Math Functions: ROUND()

The ROUND() function is used to round off a number to a certain decimal point.

The following is the syntax for the ROUND() function:

SQL Query

```
SELECT ROUND(number, decimals, operation);
```

Math Functions: SQRT()

The SQRT() function is used to find the square root of a number.

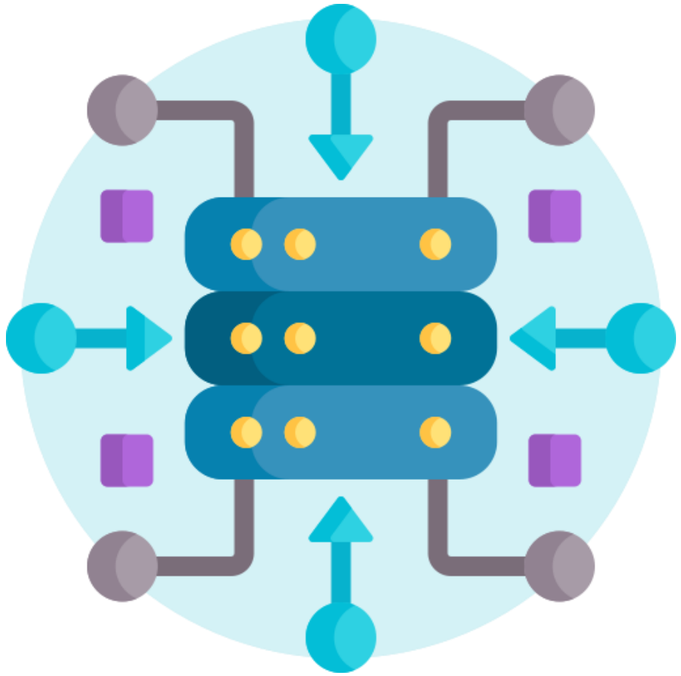
The following is the syntax for the SQRT() function:

SQL Query

```
SELECT SQRT(Number);
```

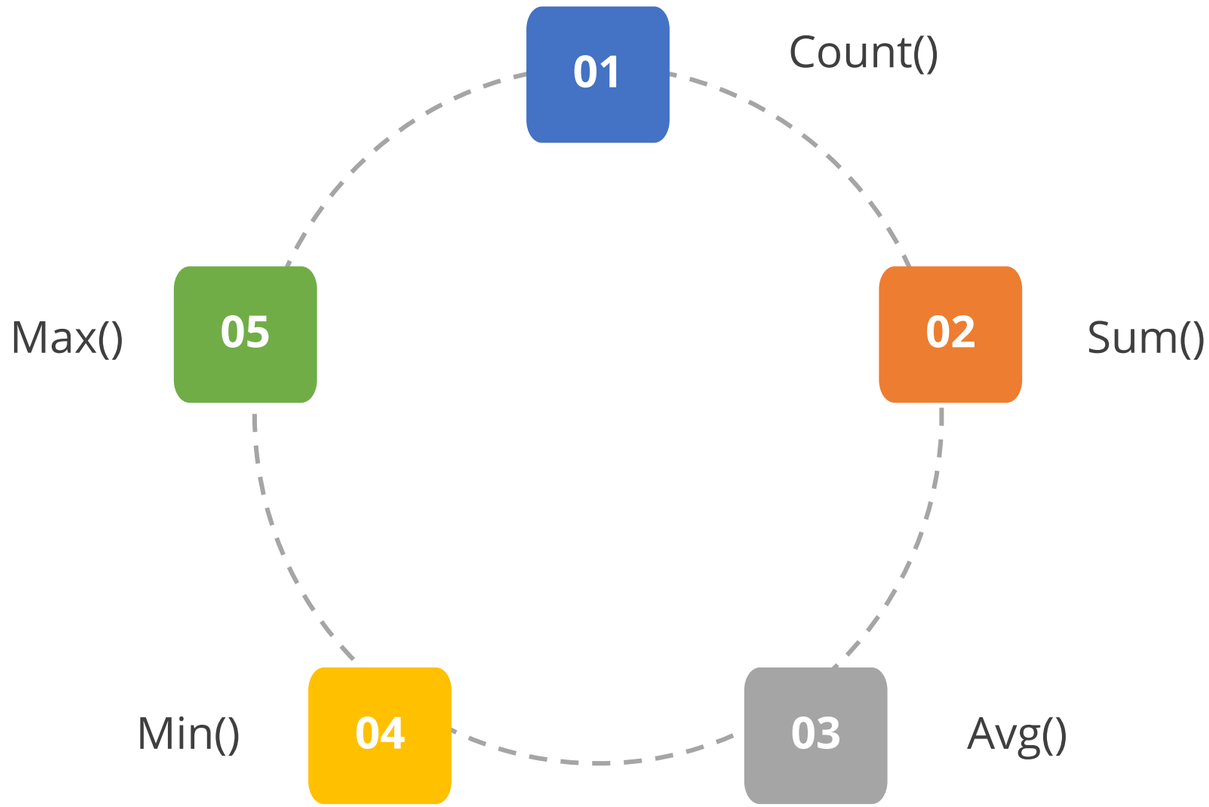
Aggregate Functions

An Aggregate function in SQL performs a calculation on multiple values and returns a single value.



Aggregate Functions

The following are the different types of Aggregate functions:



Aggregate Functions: COUNT()

The COUNT() function returns the number of rows that satisfy a given set of criteria.

The following is the syntax for the COUNT() function:

SQL Query

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

Aggregate Functions: SUM()

The SUM() function returns the total sum of a column.

The following is the syntax for the SUM() function:

SQL Query

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

Aggregate Functions: AVG()

The AVG() function returns the average value of a column.

The following is the syntax for the AVG() function:

SQL Query

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

Aggregate Functions: MIN()

The MIN() function returns the smallest value of the selected column.

The following is the syntax for the MIN() function:

SQL Query

```
SELECT MIN(column_name)
FROM table_name
WHERE condition;
```

Aggregate Functions: MAX()

The MAX() function returns the largest value of the selected column.

The following is the syntax for the MAX() function:

SQL Query

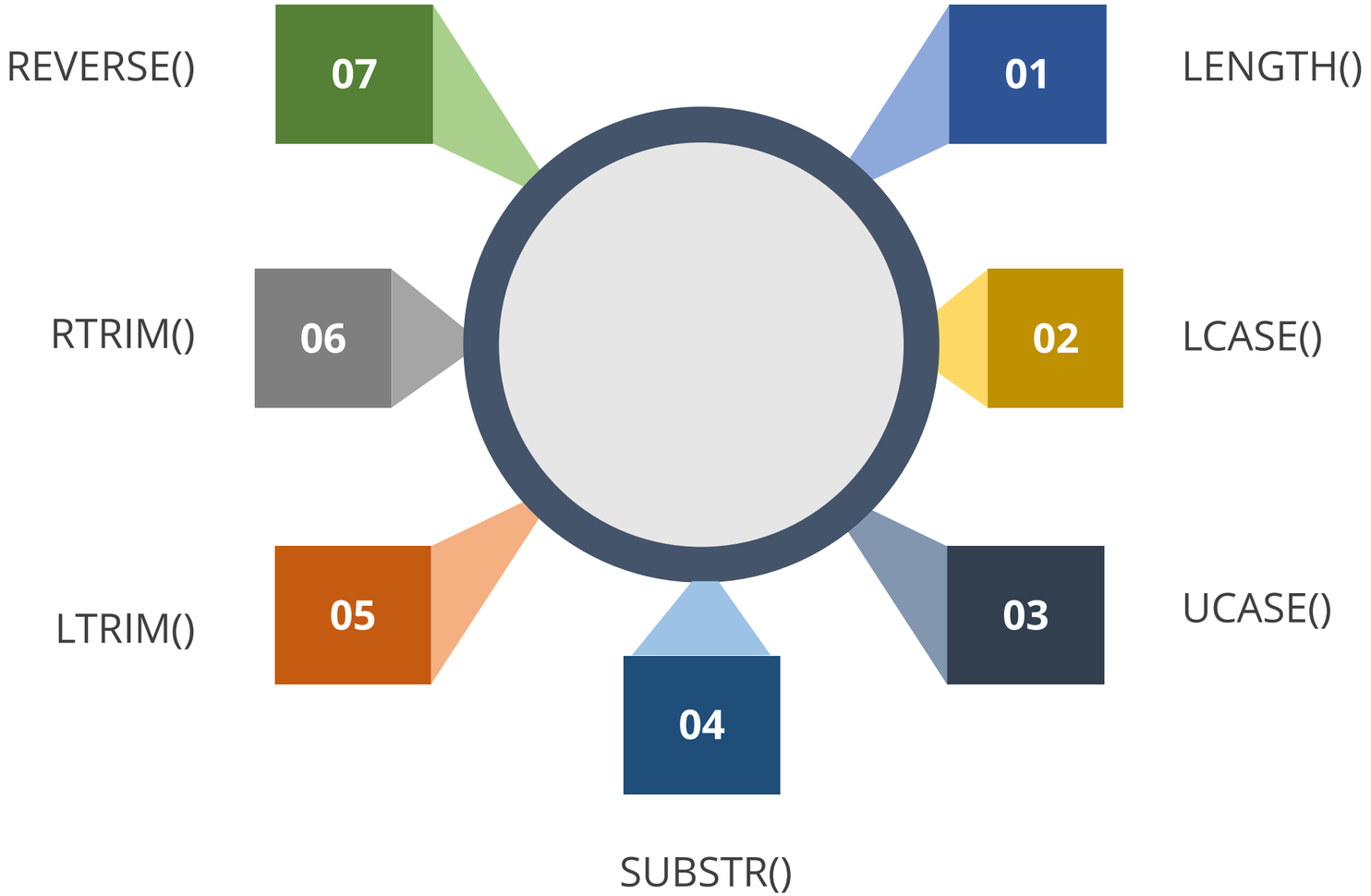
```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

String Functions

SQL string functions operate on input strings and return output strings.

String Functions

The following are the list of important String functions:



String Functions: LENGTH()

The LENGTH() function is used to determine the length of a word.

The following is the syntax for the LENGTH() function:

SQL Query

```
Syntax: LENGTH('Hello');

Output: 5
```

String Functions: LCASE()

The LCASE() function converts a string to lowercase.

The following is the syntax for the LCASE() function:

SQL Query

```
Syntax: LCASE ("Bootcamp and Certification Platform");

Output: bootcamp and certification platform
```

String Functions: UCASE()

The UCASE() function converts a string to uppercase.

The following is the syntax for the UCASE() function:

SQL Query

```
Syntax: UCASE ("Bootcamp and Certification Platform");

Output: BOOTCAMP AND CERTIFICATION PLATFORM
```

String Functions: SUBSTR()

The SUBSTR() function is used to extract a substring from a string at a specific position.

The following is the syntax for the SUBSTR() function:

SQL Query

```
Syntax: SUBSTR('Simplilearn', 1, 6);

Output: 'Simpli'
```

String Functions: LTRIM()

The LTRIM() function is used to remove a substring from a given string.

The following is the syntax for the LTRIM() function:

SQL Query

```
Syntax: LTRIM('123123Simplilearn', '123');

Output: Simplilearn
```

String Functions: RTRIM()

The RTRIM() function is used to remove a substring from a given string.

The following is the syntax for the RTRIM() function:

SQL Query

```
Syntax: RTRIM('Simplilearn123123', '123');

Output: Simplilearn
```

String Functions: REVERSE()

The REVERSE() function is used to reverse a string.

The following is the syntax for the REVERSE() function:

SQL Query

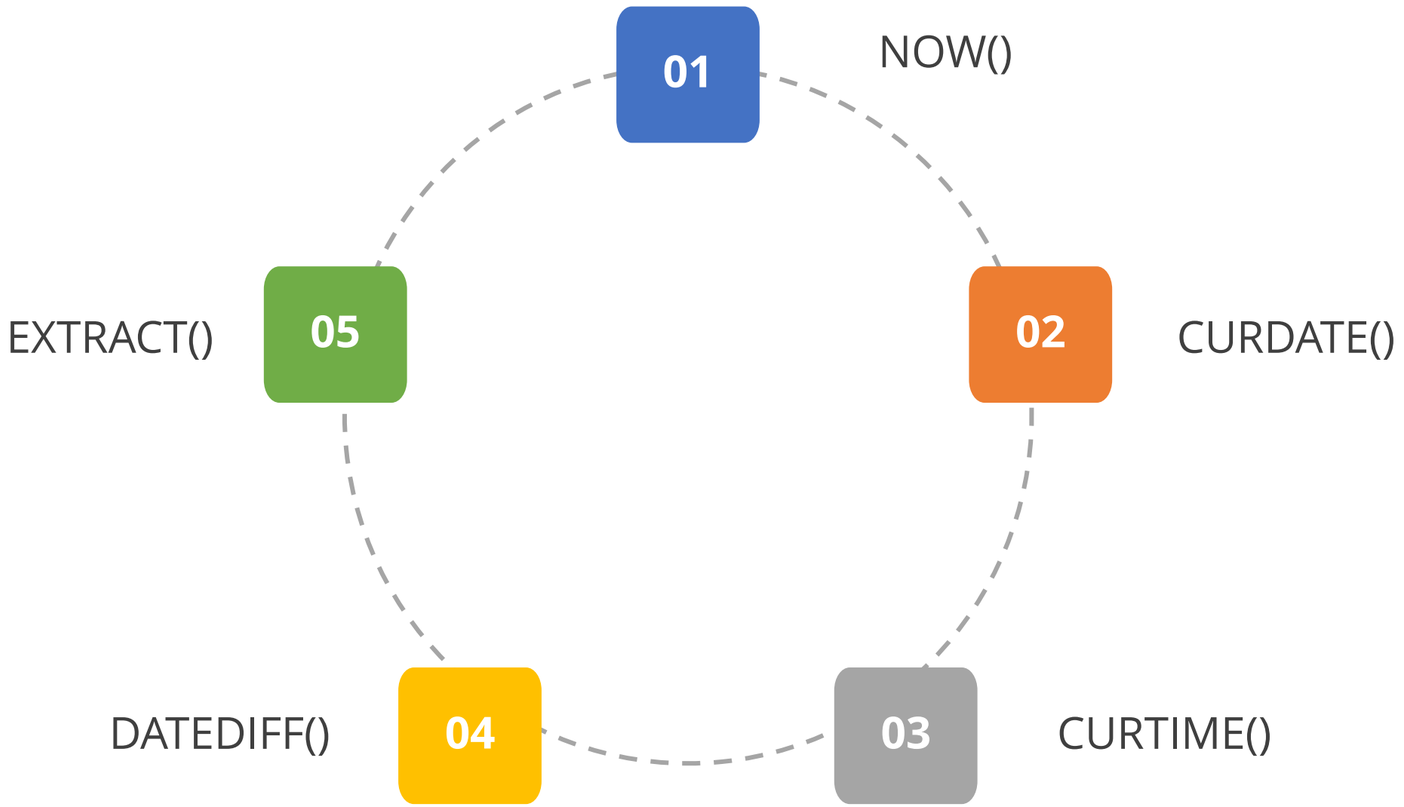
```
SELECT MAX(column_name)
FROM table_name
WHERE condition;
```

Date Functions

Date functions are used to format dates and perform date-related calculations.

Date Functions

The following are the different types of Date functions:



Date Functions: NOW()

The NOW() function returns the current system's date and time.

The following is the syntax for the NOW() function:

SQL Query

```
SELECT NOW ();
```

Date Functions: CURDATE()

The CURDATE() function returns the current system's date.

The following is the syntax for the CURDATE() function:

SQL Query

```
SELECT CURDATE ();
```

Date Functions: CURTIME()

The CURTIME() function returns the current system's date.

The following is the syntax for the CURTIME() function:

SQL Query

```
SELECT CURTIME ();
```

Date Functions: DATEDIFF()

The DATEDIFF() function returns the number of days from one date to another.

The following is the syntax for the DATEDIFF() function:

SQL Query

```
DATEDIFF(interval, date1, date2)
```

Assisted Practice

Working with Various Built-in SQL Functions **Duration: 15 Min.**

Problem Statement:

You have been assigned a task to demonstrate the use of various built-in SQL functions in MySQL for performing data manipulation and retrieval more efficiently.

Assisted Practice: Guidelines

Steps to be followed:

1. Setup database and table
2. Use built-in functions

Key Takeaways

- MySQL is a Relational Database Management System (RDBMS) that uses SQL to query from databases.
- Data Control Language (DCL) commands are a set of SQL commands that is used to grant or revoke database user authority.
- Data Definition Language (DDL) commands are a set of SQL commands that are used for creating, modifying, and deleting database structures.
- Data Manipulation Language (DML) commands are a set of SQL commands that allows to insert, alter, and delete data from a database.

Key Takeaways

- The Transaction Control Language (TCL) commands are a set of SQL commands that is used to manage database transactions.
- MySQL allows to use more than one column with UNIQUE values in the table.
- A Join statement is used to merge the rows from two or more tables.
- An Aggregate function in SQL performs a calculation on multiple values and returns a single value.

Thank You