Dhruvil Dobariya

dhruvildobariya21@gmail.com

Abstract

[Draw your reader in with an engaging abstract. It is typically a short summary of the document.   
When you’re ready to add your content, just click here and start typing.]

DBMS Learn

Documentation



Index

[1 Introduction To Database 1](#_Toc129275331)

[1.1 Introduction 1](#_Toc129275332)

[1.2 Advantages 1](#_Toc129275333)

[1.3 Basic Terms 1](#_Toc129275334)

[2 MySQL 4](#_Toc129275335)

[2.1 Introduction 4](#_Toc129275336)

[3 Overview of MySQL Workbench 5](#_Toc129275337)

[3.1 Introduction 5](#_Toc129275338)

[3.2 Functionality 5](#_Toc129275339)

[3.3 Editions 6](#_Toc129275340)

[3.4 Overview 6](#_Toc129275341)

[4 Database Design 8](#_Toc129275342)

[4.1 Introduction 8](#_Toc129275343)

[4.2 Database development life cycle 8](#_Toc129275344)

[4.3 Database Design Technique 9](#_Toc129275345)

[5 Basic SQL 11](#_Toc129275346)

[5.1 Introduction 11](#_Toc129275347)

[5.2 Data Definition Language 11](#_Toc129275348)

[5.3 Data Query Language 13](#_Toc129275349)

[5.4 Data Manipulation Language 15](#_Toc129275350)

[5.5 Data Control Language 18](#_Toc129275351)

[5.6 Transaction Control Language 18](#_Toc129275352)

[6 Data Sorting 20](#_Toc129275353)

[6.1 Introduction 20](#_Toc129275354)

[7 Null Value & Keyword 21](#_Toc129275355)

[7.1 Introduction 21](#_Toc129275356)

[8 Keys and Auto Increment 22](#_Toc129275357)

[8.1 Primary Key 22](#_Toc129275358)

[8.2 Auto Increment 22](#_Toc129275359)

[8.3 Foreign Key 23](#_Toc129275360)

[8.4 Unique Key 23](#_Toc129275361)

[9 Aggregate Functions 25](#_Toc129275362)

[9.1 Introduction 25](#_Toc129275363)

[9.2 Aggregate Function 25](#_Toc129275364)

[9.3 Group By 26](#_Toc129275365)

[9.4 Having 26](#_Toc129275366)

[9.5 Sequence of statement 27](#_Toc129275367)

1. Introduction To Database

# Introduction

**Data:**

* Fact that ca be record or stored
* For ex: Person Name, Age, Gender and Weight…

**Database:**

* Collection of logically related data
* For ex: Books Database in Library, Student Database in University…

**Management:**

* Manipulation, Searching and Securing of data.
* Viewing result in GTU website, Searching exam papers in GTU website…

**System:**

* Program or tool that used to manage database
* MS SQL, MySQL, Postgres SQL, Oracle…

**Database Management System:**

* It is a software designed to define, manipulate, retrieve and manage data in a database.

# Advantages

* Reduce data duplication
* Remove inconsistency
* Data isolation
* Guaranty of atomicity(0% or 100%)
* Allow implementing integrity constraints
* Sharing among the multiple user
* Restricted unauthorized access
* Provides backup and recovery services

# Basic Terms

**Data:**

* Data is raw, unorganized facts that need to be processed.
* For ex: Marks of students…

**Information:**

* When data is processed, organized, structured or presented in a given context so as to make it useful, it is called information.
* For ex: Result of students (Pass or Fail)…

**Metadata:**

* Metadata is data about data.
* Data such as table name, column name, data type, authorized user and user access privileges for any table is called metadata for that table.

**Data Dictionary:**

* A data dictionary is an information repository which contains metadata.

**Data Warehouse:**

* A data warehouse is an information repository which stores data.

**Field:**

* A field is a character or group of characters that have a specific meaning.
* For ex: The value of Emp\_Name, Address, Mobile\_No etc are all fields of Faculty table.

**Record/ Tuple:**

* A record is a collection of logically related fields.
* For ex: The collection of fields (Emp\_Name, Address, Mobile\_No, Subject) forms a record for the Faculty.

**Primary Key:**

* A key which is unique as well as not null.

**Unique Key:**

* A key which is unique but it could be null.

**Foreign Key:**

* A key which liked two table.

**Compose Key:**

* A key that consists of multiple columns, because one column is not sufficiently identify record uniquely.

1. MySQL

# Introduction

* MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses.
* MySQL is developed, marketed and supported by MySQL AB, which is a Swedish company.
* MySQL is released under an open-source license.
* So you have nothing to pay to use it.
* MySQL is a very powerful program in its own right.
* It handles a large subset of the functionality of the most expensive and powerful database packages.
* MySQL uses a standard form of the well-known SQL data language.
* MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
* MySQL works very quickly and works well even with large data sets.
* MySQL is very friendly to PHP, the most appreciated language for web development.
* MySQL supports large databases, up to 50 million rows or more in a table.
* The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
* MySQL is customizable.
* The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

1. Overview of MySQL Workbench

# Introduction

* MySQL Workbench is graphical user interface tool that used for working with database architects, developers, and Database Administrators.
* It is developed and maintained by Oracle.
* It provides SQL development, data modelling, data migration, and comprehensive administration tools for server configuration, user administration, backup, and many more.
* We can use this Server Administration for creating new physical data models, E-R diagrams, and for SQL development (run queries, etc.).
* It is available for all major operating systems like Mac OS, Windows, and Linux.
* MySQL Workbench fully supports MySQL Server version v5.6 and higher.

# Functionality

**SQL Development:**

* This functionality provides the capability that enables you to execute SQL queries, create and manage connections to the database Servers with the help of built-in SQL editor.

**Data Modelling (Design):**

* This functionality provides the capability that enables you to create models of the database Schema graphically, performs reverse and forward engineering between a Schema and a live database, and edit all aspects of the database using the comprehensive Table editor.
* The Table editor gives the facilities for editing tables, columns, indexes, views, triggers, partitioning, etc.

**Server Administration:**

* This functionality enables you to administer MySQL Server instances by administering users, inspecting audit data, viewing database health, performing backup and recovery, and monitoring the performance of MySQL Server.

**Data Migration:**

* This functionality allows you to migrate from Microsoft SQL Server, SQLite, Microsoft Access, PostgreSQL, Sybase ASE, SQL Anywhere, and other RDBMS tables, objects, and data to MySQL.
* It also supports migrating from the previous versions of MySQL to the latest releases.

**MySQL Enterprise Supports:**

* This functionality gives the support for Enterprise products such as MySQL firewall, MySQL Enterprise Backup, and MySQL Audit.

# Editions

* MySQL Workbench is mainly available in three editions…
  + Community Edition (Open Source, GPL)
  + Standard Edition (Commercial)
  + Enterprise Edition (Commercial)

**Community Edition:**

* The Community Edition is an open-source and freely downloadable version of the most popular database management system.
* It came under the GPL license and is supported by a hug community of developers.

**Standard Edition:**

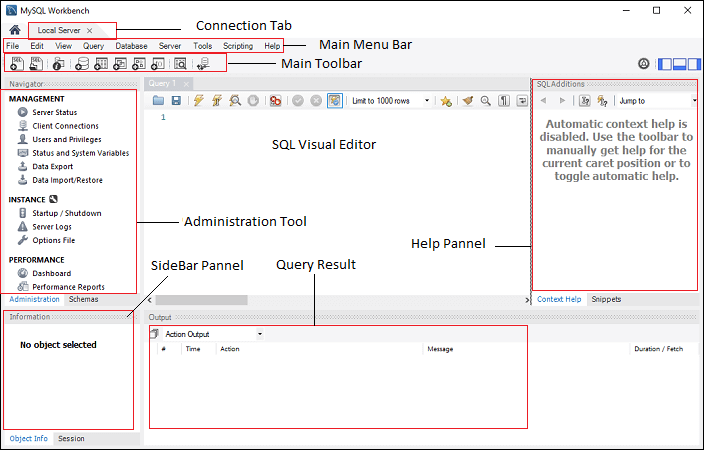
* It is the commercial edition that provides the capability to deliver high-performance and scalable Online Transaction Processing (OLTP) applications.
* It has made MySQL famous along with industrial-strength, performance, and reliability.

**Enterprise Edition:**

* It is the commercial edition that includes a set of advanced features, management tools, and technical support to achieve the highest scalability, security, reliability, and uptime.
* This edition also reduces the risk, cost, complexity in the development, deployment, and managing MySQL applications.

# Overview

* When we open my sql workbench we have this type of window open.

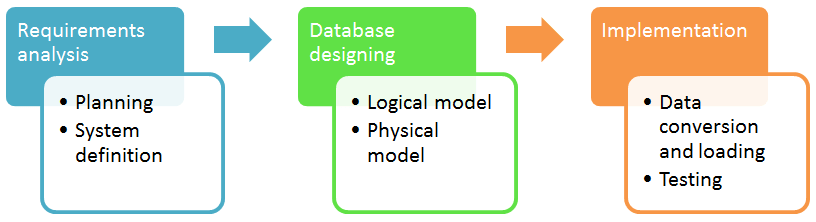


1. Database Design

# Introduction

* Database Design is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems.
* Properly designed database are easy to maintain, improves data consistency and are cost effective in terms of disk storage space.
* The database designer decides how the data elements correlate and what data must be stored.

# Database development life cycle

* The database development life cycle has a number of stages that are followed when developing database systems.
* But it is not necessary to follow every stapes.

## Requirement Analysis:

**Planning:**

* This stages of database design concepts are concerned with planning of entire Database Development Life Cycle.
* It takes into consideration the Information Systems strategy of the organization.

**System definition:**

* This stage defines the scope and boundaries of the proposed database system.

## Database designing:

**Logical model:**

* This stage is concerned with developing a database model based on requirements.
* The entire design is on paper without any physical implementations or specific DBMS considerations.

**Physical model:**

* This stage implements the logical model of the database taking into account the DBMS and physical implementation factors.

## Implementation:

**Data conversion and loading:**

* This stage of relational databases design is concerned with importing and converting data from the old system into the new database.

**Testing:**

* This stage is concerned with the identification of errors in the newly implemented system.
* It checks the database against requirement specifications.

# Database Design Technique

* We have two types of database design techniques.
  + Normalization
  + ER Modeling

## ER Modeling

* Entity Relationship Model (ER Modeling) is a graphical approach to database design.
* It is a high-level data model that defines data elements and their relationship for a specified software system.
* An ER model is used to represent real-world objects.

## Normalization

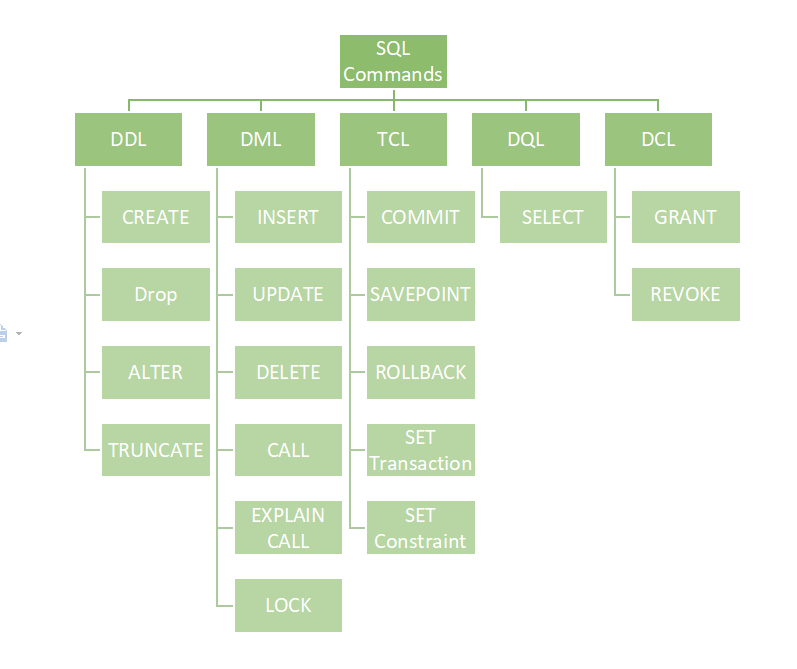
* Normalization is the process of removing redundant data from tables to improve data integrity(completeness, accuracy and consistency of data), scalability and storage efficiency.
* We have 6 type of normal forms
  + 1NF
  + 2NF
  + 3NF
  + BCNF
  + 4NF
  + 5NF

|  |  |
| --- | --- |
| Normal Form | Description |
| [1NF](https://www.javatpoint.com/dbms-first-normal-form) | A relation is in 1NF if it contains an atomic value. |
| [2NF](https://www.javatpoint.com/dbms-second-normal-form) | A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key. |
| [3NF](https://www.javatpoint.com/dbms-third-normal-form) | A relation will be in 3NF if it is in 2NF and no transition dependency exists. |
| BCNF | A stronger definition of 3NF is known as Boyce Codd's normal form. |
| [4NF](https://www.javatpoint.com/dbms-forth-normal-form) | A relation will be in 4NF if it is in Boyce Codd's normal form and has no multi-valued dependency. |
| [5NF](https://www.javatpoint.com/dbms-fifth-normal-form) | A relation is in 5NF. If it is in 4NF and does not contain any join dependency, joining should be lossless. |

1. Basic SQL

# Introduction

* SQL have basic five components,
  + Data Definition Language
  + DQL – Data Query Language
  + DML – Data Manipulation Language
  + DCL – Data Control Language
  + TCL – Transaction Control Language



# Data Definition Language

* It contains SQL command that used for define schema.
* DDL is a set of SQL commands used to create, modify, and delete database structures but not data.
* DDL contains following commands,
  + Create
  + Drop
  + Alter
  + Truncate
  + Comment
  + Rename

**Create:**

* This command is used to create the database or its objects (like table, index, function, views, store procedure, and triggers).

**Example:**

Create Database CollageDB;

Create Table Student(

    Id int Not Null Auto\_Increment,

    Name Varchar(250) Not Null,

    DateOfBirth Date Not Null,

    ContactNo Varchar(25),

    Gender Varchar(1),

    Primary Key(Id)

);

**Drop:**

* This command is used to delete objects from the database.

**Example:**

Drop Database CollageDB;

Drop Table Faculty;

**Alter:**

* This is used to update the structure of the database.

**Example:**

*-- For Single Column*

*-- Add new column in table*

Alter Table Faculty

Add Email Varchar(50);

*-- Edit column in table*

Alter Table Faculty

Modify Column Email Varchar(250);

*-- Delete column in table*

Alter Table Faculty

Drop Column Email;

*-- For Multiple Column*

*-- Add new columns in table*

Alter Table Faculty

Add Email Varchar(50),

Add Subject Varchar(50);

*-- Edit columns in table*

Alter Table Faculty

Modify Column Email Varchar(250),

Modify Column Subject Varchar (25);

*-- Delete colomuns in table*

Alter Table Faculty

Drop Column Email,

Drop Column Subject;

**Truncate:**

* This is used to remove all records from a table, including all spaces allocated for the records are removed.

**Example:**

Truncate Table student;

**Comment:**

* This is used to add comments to the data dictionary.

**Example:**

**Rename:**

* This is used to rename an object existing in the database.

**Example:**

*-- Raname table name*

Alter Table Faculty

Rename To FacultyNew

# Data Query Language

* DQL is used to perform a query on schema.
* It is used to retrieve data from schema.
* It have only one command which is select.
* When we fired select command on table that time data stored in temporary table and this table we should see in output window.

**Example:**

*-- Select all fields*

Select \* From Student;

*-- select specific fields*

Select Id, Name, Email From Student;

*-- Where condition*

Select \* from Student

Where Id = 1;

Select \* from Student

Where Id != 1;

*-- OR*

Where Id <> 1;

SELECT \* from Student

Where RollNo > 5 and RollNo <= 10

SELECT \* from Student

Where RollNo = 5 or RollNo = 10

SELECT \* from Student

Where RollNo In(5, 10, 15, 20)

SELECT \* from Student

Where RollNo BETWEEN 5 and 10

*-- Between include uper bound and lower bound*

SELECT \* from Student

Where RollNo not BETWEEN 5 and 10

Select \* from Student

Where Email is not Null;

*-- Lilke*

*-- (\_) represent one character*

*-- (%) represent more the one character*

Select \* from Student

Where RollNo like "1\_";

Select \* from Student

Where Name like "a%";

*-- starting name from a*

Select \* from Student

Where Name like "%e";

*-- starting name from e*

Select \* from Student

Where Name like "a%e";

*-- starting name from a and ending from b*

*-- orderby*

Select \* from Student

ORDER BY Name, Email, RollNo, Id;

Select \* from Student

ORDER BY Name Desc;

Select DISTINCT RollNo from Student

Select Name from Student

Limit 5;

Select Name as Username from Student;

# Data Manipulation Language

* These commands is used for data manipulation in existing schema.
* It is the component of the SQL statement that controls access to data and to the database.
* Basically, DCL statements are grouped with DML statements.
* It contains following commands,
  + Insert
  + Update
  + Delete
  + Lock
  + Call
  + Explain Plan

**Insert:**

* It is used to insert data into a table.

**Example:**

*-- Insert one record*

Insert into Student (Name, DateOfBirth, ContactNo, Gender) values ("Dhruvil Dobariya", "2002-04-04", "9487587380", "M");

*-- Insert multiple record*

Insert into Student (Name, DateOfBirth, ContactNo, Gender) values

    ("Dhaval Dobariya", "2001-04-12","","M"),

    ("Bhargav Vachhani", "2002-01-04", "9408574858", ""),

    ("Jenil Vasoya", "2002-04-11", "", ""),

    ("Dhruv Rathod", "2002-07-11", "8594003858", "M");

**Update:**

* It is used to update existing data within a table.

**Example:**

Update Student

Set Name = "Dhruvi Savaliya", Gender = "F"

Where Id = 5

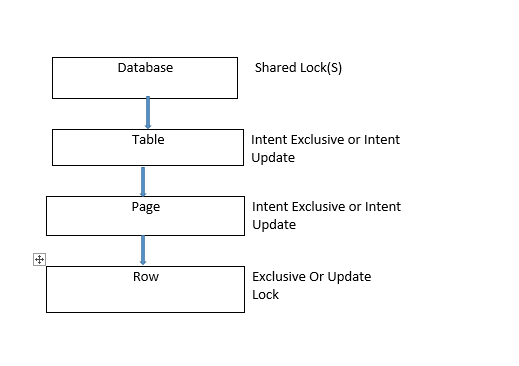
**Delete:**

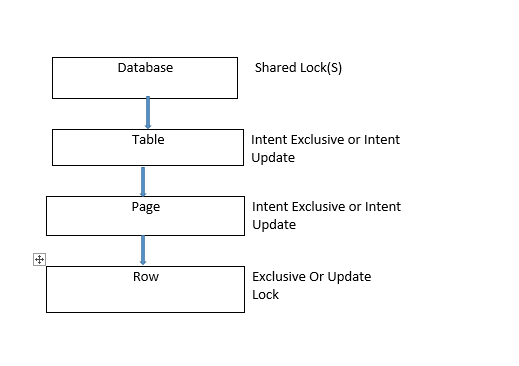
* It is used to delete records from a database table.

**Example:**

Delete From Student Where Id = 6

**Lock:**

* Data consistency is an important mechanism, and it can be done by means of SQL Locks.
* A lock is established in SQL Server when a transaction starts, and it will released when it is ended.
* We have different types of locks available in relational database,
  + **Shared (S) Locks:** 
    - When the object needs to be read, this type of lock will occur.
    - But this is not harmful.
  + **Exclusive (X) Locks:** 
    - It prevents other transactions like inserting/updating/deleting.
    - So no modifications can be done when we apply this type of lock on object.
  + **Update (U) Locks:** 
    - It’s like Exclusive lock but here the operation can be viewed as “read phase” and “write phase”.
    - During the read phase, other transactions are prevented.
  + **Intent Locks:**
    - Intent lock happens on a table, when the shared (S) lock or exclusive (X) lock or Update (U) lock happens on the row.
  + **Regular intent locks:**
    - Intent exclusive (IX)
    - Intent shared (IS)
    - Intent update (IU).
  + **Conversion locks:**
    - Shared with intent exclusive (SIX)
    - Shared with intent update (SIU)
    - Update with intent exclusive (UIX)
* We have hierarchy for lock.

(Select)

(Update/Insert/Delete)

**Call:**

* Call a PL/SQL

**Explain Plan:**

* It describes the access path to data.

# Data Control Language

* DCL includes commands which mainly use for user rights, permissions and other controls on database.
* It contains two command,
  + Grant
  + Revoke

**Grant:**

* This command is used to give user access privileges of database to user.

**Example:**

GRANT insert,

select on studentdb to root

*-- We give permision of insrt into studentdb to root*

**Revoke:**

* This command revoke the user privileges of database from the user.

**Example:**

REVOKE insert,

select on studentdb from root

*-- We revoke permision of insrt into studentdb from root*

# Transaction Control Language

* We have group of some transection which used for execute single query.
* Transection done when this group of transections id done,
* If any one is failed then whole transection is failed.
* So transection have only two result, success and failure.
* Transection contains some commands,
  + Begin
  + Commit
  + Rollback
  + Savepoint
  + Set Transection

**Begin:**

* Opens a Transaction.

**Commit:**

* Commits a Transaction.

**Rollback:**

* Rollback transection if any error occur during transaction.

**Savepoint:**

* Set a save point within the transection.

**Set Transection:**

* Specify characteristics for transection.

1. Data Sorting

# Introduction

* We have “Order By” key word to sort our result set.
* By default it’s sort in ascending order, But we can specify if we want to sort in descending using “Desc” Key word.

**Syntax:**

SELECT column1, column2, ...

FROM table\_name

ORDER BY column1, column2, ... ASC|DESC;

**Example:**

*-- ascending order*

SELECT \* FROM Customers

ORDER BY Country;

*-- descending order*

SELECT \* FROM Customers

ORDER BY Country; DESC

1. Null Value & Keyword

# Introduction

* If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field.
* Then, that field will be saved with a NULL value.
* Null value different from zero or empty.
* Null means nothing.
* We have two key work to check null value, “Is Null” and “Is Not Null”.

**Syntax:**

SELECT column\_names

FROM table\_name

WHERE column\_name IS NULL | IS NOT NULL;

**Example:**

*-- Get rows which have address is null*

SELECT CustomerName, ContactName, Address

FROM Customers

WHERE Address IS NULL;

*-- Get rows which have address is not null*

SELECT CustomerName, ContactName, Address

FROM Customers

WHERE Address IS NOT NULL;

1. Keys and Auto Increment

# Primary Key

* Primary key is key that used to uniquely identify record in table.
* Primary key must be unique and not null.
* One table contains one, primary key, but this primary key may combination one or more column.
* We are use “Primary Key” key word to define primary key.

*-- Define Primary Key*

CREATE TABLE Persons (

    ID int NOT NULL,

    LastName varchar(255) NOT NULL,

    FirstName varchar(255),

    Age int,

    PRIMARY KEY (ID)

);

*-- OR*

CREATE TABLE Persons (

    ID int NOT NULL,

    LastName varchar(255) NOT NULL,

    FirstName varchar(255),

    Age int,

    CONSTRAINT PK\_Person PRIMARY KEY (ID,LastName)

);

*-- Alter Primary Key*

ALTER TABLE Persons

ADD PRIMARY KEY (ID);

*-- OR*

ALTER TABLE Persons

ADD CONSTRAINT PK\_Person PRIMARY KEY (ID,LastName);

*-- Drop Primary Key*

ALTER TABLE Persons

DROP PRIMARY KEY;

# Auto Increment

* Auto Increment generate automatic unique and incremental number in particular field.

**Example:**

CREATE TABLE Persons (

    Personid int NOT NULL AUTO\_INCREMENT,

    LastName varchar(255) NOT NULL,

    FirstName varchar(255),

    Age int,

    PRIMARY KEY (Personid)

);

# Foreign Key

* Foreign key is the key that used to linked two table.
* Parent table primary key is used as a foreign key in child table.
* We have “Foreign Key” keyword to define foreign key.

CREATE TABLE Orders (

    OrderID int NOT NULL,

    OrderNumber int NOT NULL,

    PersonID int,

    PRIMARY KEY (OrderID),

    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)

);

*-- OR*

CREATE TABLE Orders (

    OrderID int NOT NULL,

    OrderNumber int NOT NULL,

    PersonID int,

    PRIMARY KEY (OrderID),

    CONSTRAINT FK\_PersonOrder FOREIGN KEY (PersonID)

    REFERENCES Persons(PersonID)

);

# Unique Key

* Unique key is the key that used to set unique behavior of particular field.
* Unique key may null, primary key must not.
* Unique key may one or more in table.
* We have “Unique Key” keyword to define unique key.

CREATE TABLE Persons (

    ID int NOT NULL,

    LastName varchar(255) NOT NULL,

    FirstName varchar(255),

    Age int,

    UNIQUE (ID)

);

*-- OR*

CREATE TABLE Persons (

    ID int NOT NULL,

    LastName varchar(255) NOT NULL,

    FirstName varchar(255),

    Age int,

    CONSTRAINT UC\_Person UNIQUE (ID,LastName)

);

1. Aggregate Functions

# Introduction

* Aggregate function is used to perform calculation on row of single column.
* It return only single value.
* It is also used to summarize the data.

# Aggregate Function

* We have five types of aggregate function,
  + Count
  + Sum
  + Avg
  + Min
  + Max

**Count:**

* Count number is used to count number of rows in table.

**Example:**

select Count(\*) from Product;

select count(distinct Company) from Product;

**Sum:**

* Sum is used to calculate sum of all selected column.
* It works on only numeric fields.

**Example:**

SELECT Sum(Quantity) As TotalQuantity from Product;

**Avg:**

* Avg function is used to calculate average of selected column.
* It works on only numeric fields.

**Example:**

SELECT AVG(Cost) from Product;

**Min:**

* Min is used to find minimum value of particular column.
* It works on only numeric fields.

**Example:**

SELECT Min(Quantity) from Product;

**Max:**

* Max is used to find maximum value of particular column.
* It works on only numeric fields.

**Example:**

SELECT Max(Quantity) from Product;

# Group By

* Group By is used to make collection of same value so we can summarize data.
* Group By statement is used with aggregate functions.

**Example:**

SELECT Company, Sum(Quantity) from Product

Group By Company

# Having

* Having is used to specify condition after group by with aggregate function.
* We must use “Having” with aggregate function we can’t use “Where”.

**Example:**

SELECT Company, Count(Company) From Product

Group By Company

Having Count(Company) >= 5;

SELECT Company, Sum(Quantity) As TotalQuantity from Product

Group By Company

HAVING TotalQuantity > 50;

SELECT Company, Sum(Quantity) As TotalQuantity from Product

Where Quantity >= 4

Group By Company

HAVING TotalQuantity > 50;

SELECT Company, Sum(Quantity) As TotalQuantity from Product

Where Quantity >= 4

Group By Company

HAVING TotalQuantity > 40

ORDER BY Company

LIMIT 2;

# Sequence of statement

* We have particular sequence that we must follow in SQL queries.

**Syntax:**

SELECT column\_name(s)

FROM table\_name

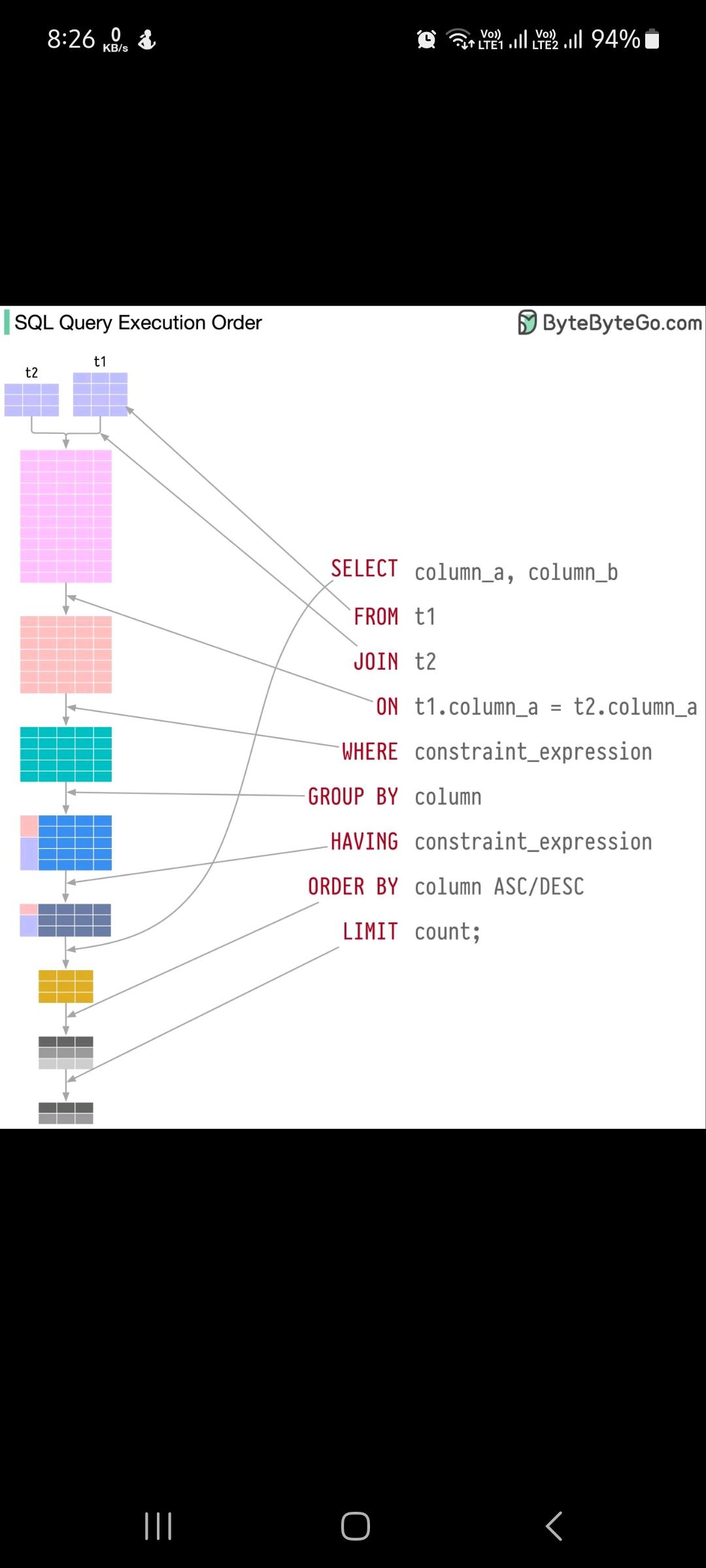
WHERE condition

GROUP BY column\_name(s)

HAVING condition

ORDER BY column\_name(s)

LIMIT number;

****