Mihir Ughade

Project Overview:

Vision Heaven is an online eyewear retailer offering a wide range of eyeglasses, sunglasses, and contact lenses.

It provides services like home eye tests and a virtual try-on feature.

aims to deliver high-quality, affordable eyewear with an emphasis on convenience through both online and offline channels, including physical stores across India.

The company focuses on personalized customer service, with products available for both prescription and non-prescription needs.

EyeVision is known for innovative technology in eyewear shopping and a strong online presence.

SQL INTRODUCTION

Concept In Focus:

Data :

Any Sort of Information that is stored is called data .

Examples: 1) Messages and Multimedia on Whatsapp

2) Products and Order on Amazon

3) Contact details in telephone directory, etc.

Database :

An organized collection of data is called a database.

Database Management System (DBMS):

A Software that is used to easily store and access data from the database in a secure way:

Types of Databases: -

There are different types of databases bases on how we organize the data:

1) Relational Databases:
   In relational databases, the data is organized in the form of tables.

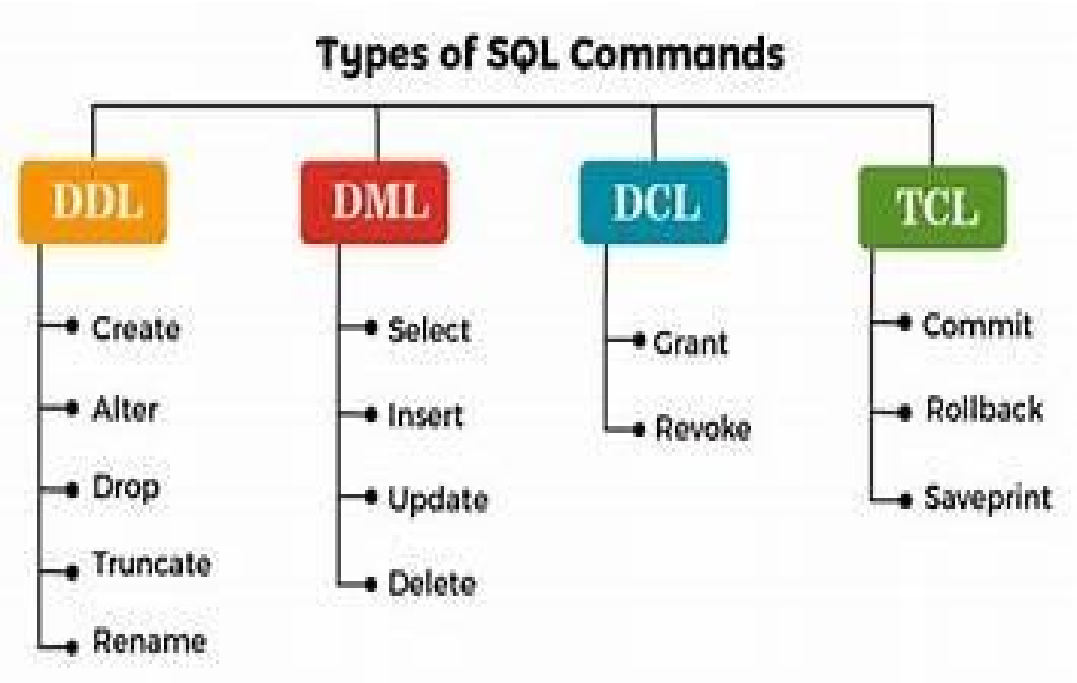
2) Non-Relational Databases:
   Graph, key value, column family, Document. These four types are commonly referred as non-relational databases

What is SQL?

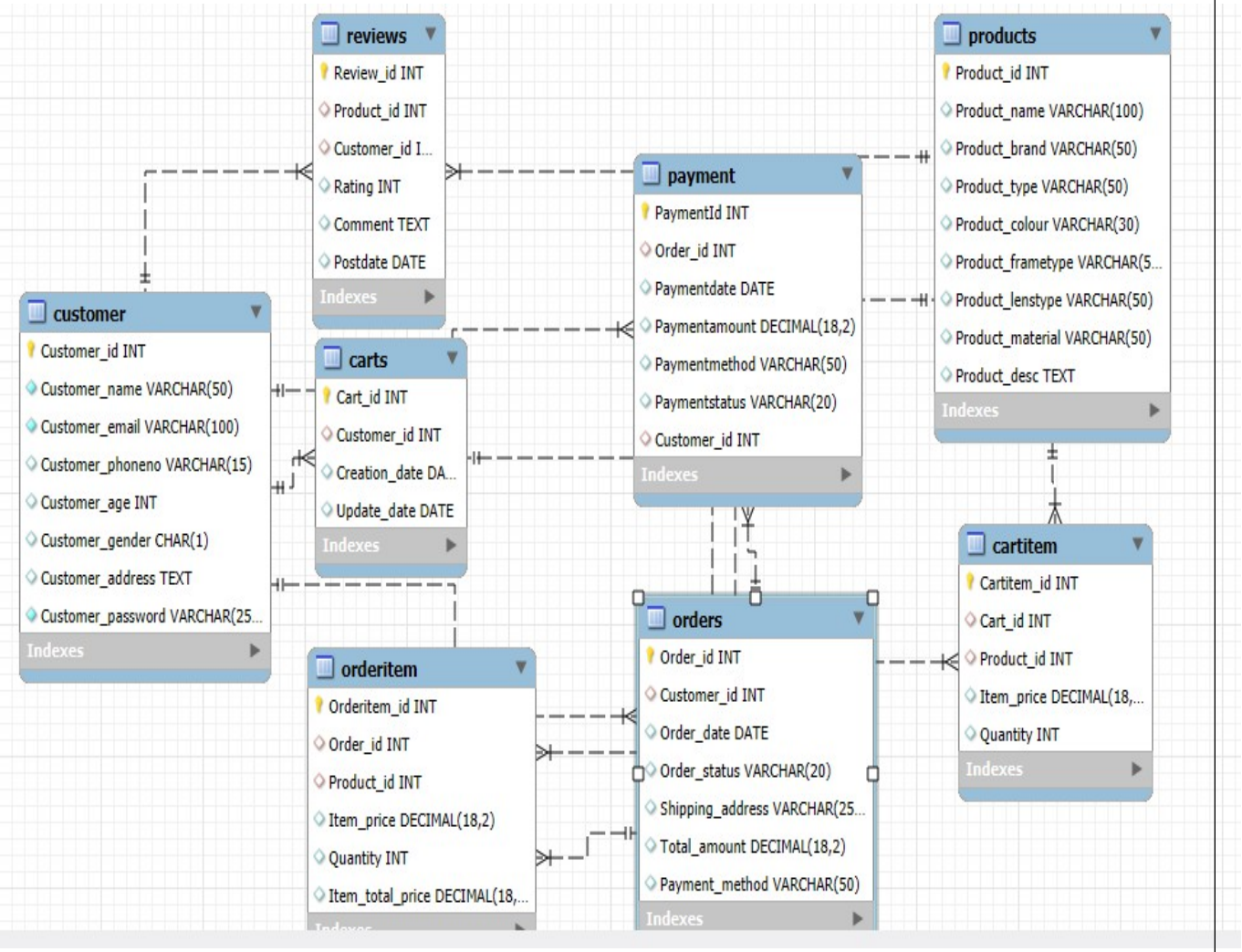
Structured Query Language is a database tool which is used to create and access database to support software application.

What are different types of statements supported by sql ?

Types of SQL Commands



Er Diagram:


reviews
Review_id INT
Product_id INT
Customer_id I...
Rating INT
Comment TEXT
Postdate DATE
Indexes
products
Product_id INT
Product_name VARCHAR(100)
Product_brand VARCHAR(50)
Product_type VARCHAR(50)
Product_colour VARCHAR(30)
Product_frametype VARCHAR(5...
Product_lenstype VARCHAR(50)
Product_material VARCHAR(50)
Product_desc TEXT
Indexes
payment
PaymentId INT
Order_id INT
Paymentdate DATE
Paymentamount DECIMAL(18,2)
Paymentmethod VARCHAR(50)
Paymentstatus VARCHAR(20)
Customer_id INT
Indexes
customer
Customer_id INT
Customer_name VARCHAR(50)
Customer_email VARCHAR(100)
Customer_phoneno VARCHAR(15)
Customer_age INT
Customer_gender CHAR(1)
Customer_address TEXT
Customer_password VARCHAR(25...
Indexes
carts
Cart_id INT
Customer_id INT
Creation_date DA...
Update_date DATE
Indexes
cartitem
Cartitem_id INT
Cart_id INT
Product_id INT
Item_price DECIMAL(18,...
Quantity INT
Indexes
orders
Order_id INT
Customer_id INT
Order_date DATE
Order_status VARCHAR(20)
Shipping_address VARCHAR(25...
Total_amount DECIMAL(18,2)
Payment_method VARCHAR(50)
Indexes
orderitem
Orderitem_id INT
Order_id INT
Product_id INT
Item_price DECIMAL(18,2)
Quantity INT
Item_total_price DECIMAL(18,...

1. NOT NULL Constraint

The **NOT NULL** constraint ensures that specific columns must always have a value in each row of the table. If you attempt to insert a record without values in these columns, the database will return an error.

Columns with NOT NULL Constraint

In this table:

1. **Customer_name**: Must have a value, ensuring every customer has a name.
2. **Customer_email**: Must have a value and be unique, ensuring every customer has an email address that isn't empty and doesn't duplicate another entry.
3. **Customer_gender**: Must have a value and can only be 'M' or 'F', so every customer has a specified gender.
4. **Customer_password**: Must have a value, ensuring a password is stored for each customer.

If any of these columns are left empty in an INSERT or UPDATE operation, the database will prevent the operation from completing.

```sql
CREATE TABLE Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name VARCHAR(50) NOT NULL,                    -- Customer name must be filled in
    Customer_email VARCHAR(100) UNIQUE NOT NULL,           -- Email must be unique and not empty
    Customer_phoneno VARCHAR(15),
    Customer_age INT CHECK (Customer_age >= 18),           -- Age should be 18 or older
    Customer_gender CHAR(1) NOT NULL CHECK (Customer_gender IN ('M', 'F')),  -- Gender must be 'M' or 'F' and cannot be empty
    Customer_address TEXT,
    Customer_password VARCHAR(255) NOT NULL                -- Password must be filled in
);
```

Field	Type	Null	Key	Default	Extra
Customer_id	int	NO	PRI	NULL	
Customer_name	varchar(50)	NO		NULL	
Customer_email	varchar(100)	NO	UNI	NULL	
Customer_phoneno	varchar(15)	YES		NULL	
Customer_age	int	YES		NULL	
Customer_gender	char(1)	NO		NULL	
Customer_address	text	YES		NULL	
Customer_password	varchar(255)	NO		NULL	

2. UNIQUE Constraint

- Ensures all values in a column are unique across rows, which prevents duplicate entries in that column.
- Example: In the Customer table, Customer_email has a UNIQUE constraint to ensure each customer's email address is unique.

```sql
CREATE TABLE Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name VARCHAR(50) NOT NULL,
    Customer_email VARCHAR(100) UNIQUE NOT NULL,  -- Email must be unique and not null
    Customer_phoneno VARCHAR(15),
    Customer_age INT CHECK (Customer_age >= 18),
    Customer_gender CHAR(1) NOT NULL CHECK (Customer_gender IN ('M', 'F')),
    Customer_address TEXT,
    Customer_password VARCHAR(255) NOT NULL
);
```

```sql
INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno, Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES (1, 'Ajit Pandey', 'ajitpandy@gmail.com', '9819978432', 28, 'M', 'Kalyan west', 'passwordAjit123');
INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno, Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES (2, 'Chirag Wamanacharya', 'ajitpandy@gmail.com', '9819978433', 30, 'M', 'Kalyan East', 'passwordChirag456');
```

Error Code: 1062. Duplicate entry 'ajitpandy@gmail.com' for key 'customer.Customer_email'

3. PRIMARY KEY Constraint

- A primary key uniquely identifies each row in a table. It is a unique, non-nullable column or a set of columns.
- Example: In the Products table, Product_id is the primary key.

The primary key constraint ensures:

1. **Uniqueness**: Each value in the primary key column(s) is unique across all rows in the table.
2. **Not Null**: Primary key columns cannot have NULL values.

```sql
CREATE TABLE Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name VARCHAR(50) NOT NULL,
    Customer_email VARCHAR(100) UNIQUE NOT NULL,  -- Email must be unique and not null
    Customer_phoneno VARCHAR(15),
    Customer_age INT CHECK (Customer_age >= 18),
    Customer_gender CHAR(1) NOT NULL CHECK (Customer_gender IN ('M', 'F')),
    Customer_address TEXT,
    Customer_password VARCHAR(255) NOT NULL
);
```

```sql
INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno, Customer_age,
Customer_gender, Customer_address, Customer_password)
VALUES (1, 'Ajit Pandey', 'ajitpandy@gmail.com', '9819978432', 28, 'M', 'Kalyan west', 'passwordAjit123');
```

```sql

INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno, Customer_age,
Customer_gender, Customer_address, Customer_password)
VALUES (1, 'Chirag Wamanacharya', 'chiragwamanacharya@gmail.com', '9819978433', 30, 'M', 'Kalyan East', 'passwordChirag456');

```

Since the primary key column (Customer_id) must be unique, any attempt to insert a duplicate Customer_id will result in an error.

4. FOREIGN KEY Constraint

- Ensures referential integrity between two tables by linking a column (or set of columns) in one table to a primary key in another table.
- Example: In the Orders table, Customer_id is a foreign key that references Customer(Customer_id). This ensures that each order is associated with a valid customer.

Customer_id INT,

FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id)

```sql
CREATE TABLE Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name VARCHAR(50) NOT NULL,
    Customer_email VARCHAR(100) UNIQUE NOT NULL,  -- Email must be unique and not null
    Customer_phoneno VARCHAR(15),
    Customer_age INT CHECK (Customer_age >= 18),
    Customer_gender CHAR(1) NOT NULL CHECK (Customer_gender IN ('M', 'F')),
    Customer_address TEXT,
    Customer_password VARCHAR(255) NOT NULL
);
```

```sql
CREATE TABLE `Order` (
    Order_id INT PRIMARY KEY,                    -- Primary Key for Order table
    Order_date DATE NOT NULL,
    Shipping_address TEXT,
    Total_amount DECIMAL(10, 2),
    Customer_id INT,                             -- Foreign Key column referencing Customer_id
    FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id)   -- Establishes the foreign key relationshi
);
```

Field	Type	Null	Key	Default	Extra
Order_id	int	NO	PRI	NULL	
Order_date	date	NO		NULL	
Shipping_address	text	YES		NULL	
Total_amount	decimal(10,2)	YES		NULL	
Customer_id	int	YES	MUL	NULL	

5. CHECK Constraint

- Enforces a condition on the values in a column. Only values that satisfy the condition are allowed.
- Example: In the Customer table, Customer_age has a CHECK constraint to ensure that customers are at least 18 years old.

```sql
CREATE TABLE Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name VARCHAR(50) NOT NULL,
    Customer_email VARCHAR(100) UNIQUE NOT NULL,  -- Email must be unique and not null
    Customer_phoneno VARCHAR(15),
    Customer_age INT CHECK (Customer_age >= 18),
    Customer_gender CHAR(1) NOT NULL CHECK (Customer_gender IN ('M', 'F')),
    Customer_address TEXT,
    Customer_password VARCHAR(255) NOT NULL
);
```

```sql
INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno,
Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES (1, 'Ajit Pandey', 'ajitpandy@gmail.com', '9819978432', 28, 'M', 'Kalyan West', 'passwordAjit123');

INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno,
Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES (2, 'Chirag Wamanacharya', 'chiragwamanacharya@gmail.com', '9819978433', 30, 'M', 'Kalyan East', 'passwordChirag456');
```

```sql
INSERT INTO Customer (Customer_id, Customer_name, Customer_email,
Customer_phoneno, Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES (3, 'Pooja Shah', 'poojashah@gmail.com', '9819978434', 16, 'F', 'Kalyan West', 'passwordPooja789');
```

Error Code: 3819. Check constraint 'customer_chk_1' is violated.

6. DEFAULT Constraint

- Assigns a default value to a column if no value is specified when a new row is inserted.
- Example: In the Orders table, Order_status has a default value of 'Pending'.

  Order_status VARCHAR(20) DEFAULT 'Pending'

```sql
CREATE TABLE Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name VARCHAR(50) NOT NULL,
    Customer_email VARCHAR(100) UNIQUE NOT NULL,
    Customer_phoneno VARCHAR(15),
    Customer_age INT DEFAULT 18,  -- Default age is 18
    Customer_gender CHAR(1) NOT NULL CHECK (Customer_gender IN ('M', 'F')),
    Customer_address TEXT,
    Customer_password VARCHAR(255) NOT NULL
);
```

```sql
INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno,
Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES (1, 'Ajit Pandey', 'ajitpandy@gmail.com', '9819978432', 28, 'M', 'Kalyan West', 'passwordAjit123');
INSERT INTO Customer (Customer_id, Customer_name, Customer_email, Customer_phoneno,
 Customer_gender, Customer_address, Customer_password)
VALUES (2, 'Chirag Wamanacharya', 'chiragwamanacharya@gmail.com', '9819978433', 'M', 'Kalyan East', 'passwordChirag456');
select * from Customer;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	Customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandy@gmail.com	9819978432	28	M	Kalyan West	passwordAjit123
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com	9819978433	18	M	Kalyan East	passwordChirag456
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Creation of Table in SQL

Creating a table in SQL involves defining the structure of the data it will store. This includes specifying the **table name**, **column definitions**, **constraints**, and any **table options**. Below is a breakdown of each component involved in table creation.

Definition of Table Creation

Table creation in SQL is the process of defining a new table with a specified structure. This structure dictates how data will be stored and accessed. SQL tables are created using the CREATE TABLE statement, which includes specifications for each column, data types, constraints, and options for indexing, storage, and performance.

Components of Table Creation

1. **Table Name**
   - The name of the table is unique within the database. It represents the entity that the table stores data about, such as Customer, Product, Order, etc.
   - Naming conventions usually follow capitalized words or underscore-separated words for better readability.
2. **Column Definition**

- Each column in a table must have a defined name and data type.
- Data types define the kind of data that can be stored in the column (e.g., INT for integers, VARCHAR for variable-length strings, DATE for date values).
- Columns can also be assigned default values.

```sql
CREATE TABLE Customer (
    Customer_id INT PRIMARY KEY,
    Customer_name VARCHAR(50) NOT NULL,
    Customer_email VARCHAR(100) UNIQUE NOT NULL,
    Customer_phoneno VARCHAR(15),
    Customer_age INT CHECK (Customer_age >= 18),
    Customer_gender CHAR(1) CHECK (Customer_gender IN ('M', 'F')),
    Customer_address TEXT,
    Customer_password VARCHAR(255) NOT NULL
);
```

	Field	Type	Null	Key	Default	Extra
▶	Customer_id	int	NO	PRI	NULL	
	Customer_name	varchar(50)	NO		NULL	
	Customer_email	varchar(100)	NO	UNI	NULL	
	Customer_phoneno	varchar(15)	YES		NULL	
	Customer_age	int	YES		18	
	Customer_gender	char(1)	NO		NULL	
	Customer_address	text	YES	YES	NULL	
	Customer_password	varchar(255)	NO		NULL	

```sql
CREATE TABLE Products (
    Product_id INT PRIMARY KEY,
    Product_name VARCHAR(100),
    Product_brand VARCHAR(50),
    Product_type VARCHAR(50),
    Product_colour VARCHAR(30),
    Product_frametype VARCHAR(50),
    Product_lenstype VARCHAR(50),
    Product_material VARCHAR(50),
    Product_desc TEXT
);
```

	Field	Type	Null	Key	Default	Extra
▶	Product_id	int	NO	PRI	NULL	
	Product_name	varchar(100)	YES		NULL	
	Product_brand	varchar(50)	YES		NULL	
	Product_type	varchar(50)	YES		NULL	
	Product_colour	varchar(30)	YES		NULL	
	Product_frametype	varchar(50)	YES		NULL	
	Product_lenstype	varchar(50)	YES		NULL	
	Product_material	varchar(50)	YES		NULL	
	Product_desc	text	YES		NULL	

Mihir Ughade

```sql
CREATE TABLE Orders (
    Order_id INT PRIMARY KEY,
    Customer_id INT,
    Order_date DATE,
    Order_status VARCHAR(20),
    Shipping_address VARCHAR(255),
    Total_amount DECIMAL(10, 2),
    Payment_method VARCHAR(50),
    FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id)
);
```

Field	Type	Null	Key	Default	Extra
Order_id	int	NO	PRI	NULL	
Customer_id	int	YES	MUL	NULL	
Order_date	date	YES		NULL	
Order_status	varchar(20)	YES		NULL	
Shipping_address	varchar(255)	YES		NULL	
Total_amount	decimal(10,2)	YES		NULL	
Payment_method	varchar(50)	YES		NULL	

```sql
CREATE TABLE OrderItem (
    Orderitem_id INT PRIMARY KEY,
    Order_id INT,
    Product_id INT,
    Item_price DECIMAL(10, 2),
    Quantity INT,
    Item_total_price DECIMAL(10, 2) GENERATED ALWAYS AS (Item_price * Quantity) STORED,
    FOREIGN KEY (Order_id) REFERENCES Orders(Order_id),
    FOREIGN KEY (Product_id) REFERENCES Products(Product_id)
);
desc orderitem;
```

Field	Type	Null	Key	Default	Extra
Orderitem_id	int	NO	PRI	NULL	
Order_id	int	YES	MUL	NULL	
Product_id	int	YES	MUL	NULL	
Item_price	decimal(10,2)	YES		NULL	
Quantity	int	YES		NULL	
Item_total_price	decimal(10,2)	YES		NULL	STORED GENERATED

```
CREATE TABLE Carts (
    Cart_id INT PRIMARY KEY,
    Customer_id INT,
    Creation_date DATE,
    Update_date DATE,
    FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id)
);
desc carts;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content

Field	Type	Null	Key	Default	Extra
Cart_id	int	NO	PRI	NULL	
Customer_id	int	YES	MUL	NULL	
Creation_date	date	YES		NULL	
Update_date	date	YES		NULL	

```
CREATE TABLE CartItem (
    Cartitem_id INT PRIMARY KEY,
    Cart_id INT,
    Product_id INT,
    Item_price DECIMAL(10, 2),
    Quantity INT,
    FOREIGN KEY (Cart_id) REFERENCES Carts(Cart_id),
    FOREIGN KEY (Product_id) REFERENCES Products(Product_id)
);
desc cartitem;
```

Field	Type	Null	Key	Default	Extra
Cartitem_id	int	NO	PRI	NULL	
Cart_id	int	YES	MUL	NULL	
Product_id	int	YES	MUL	NULL	
Item_price	decimal(10,2)	YES		NULL	
Quantity	int	YES		NULL	

```sql
CREATE TABLE Payment (
        PaymentId INT PRIMARY KEY,
        Order_id INT,
        Paymentdate DATE,
        Paymentamount DECIMAL(10, 2),
        Paymentmethod VARCHAR(50),
        Paymentstatus VARCHAR(20),
        Customer_id INT,
        FOREIGN KEY (Order_id) REFERENCES Orders(Order_id),
        FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id)
);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
PaymentId	int	NO	PRI	NULL	
Order_id	int	YES	MUL	NULL	
Paymentdate	date	YES		NULL	
Paymentamount	decimal(10,2)	YES		NULL	
Paymentmethod	varchar(50)	YES		NULL	
Paymentstatus	varchar(20)	YES		NULL	
Customer_id	int	YES	MUL	NULL	

```sql
CREATE TABLE Reviews (
Review_id INT PRIMARY KEY,
Product_id INT,
Customer_id INT,
Rating INT CHECK (Rating BETWEEN 1 AND 5),
Comment TEXT,
Postdate DATE,
FOREIGN KEY (Product_id) REFERENCES Products(Product_id),
FOREIGN KEY (Customer_id) REFERENCES Customer(Customer_id)
);
desc reviews;
```

Field	Type	Null	Key	Default	Extra
Review_id	int	NO	PRI	NULL	
Product_id	int	YES	MUL	NULL	
Customer_id	int	YES	MUL	NULL	
Rating	int	YES		NULL	
Comment	text	YES		NULL	
Postdate	date	YES		NULL	

Modifying the structure of a table :

Tables Structure Modification :

Types of Modification

1) Adding Columns

```
ALTER TABLE Customer
ADD COLUMN Customer_date_of_birth DATE;
```

Field	Type	Null	Key	Default	Extra
Customer_name	varchar(50)	NO		NULL	
Customer_email	varchar(100)	NO	PRI	NULL	
Customer_password	varchar(255)	NO		NULL	
Customer_date_of_birth	date	YES		NULL	

2) Modifying Columns

```
ALTER TABLE Customer
MODIFY COLUMN Customer_name VARCHAR(100);
desc customer;
```

Field	Type	Null	Key	Default	Extra
Customer_name	varchar(100)	YES		NULL	
Customer_email	varchar(100)	NO	PRI	NULL	
Customer_password	varchar(255)	NO		NULL	
Customer_date_of_birth	date	YES		NULL	

3) Dropping Columns

```
ALTER TABLE Customer
DROP COLUMN Customer_date_of_birth;
```

Field	Type	Null	Key	Default	Extra
Customer_name	varchar(100)	YES		NULL	
Customer_email	varchar(100)	NO	PRI	NULL	
Customer_password	varchar(255)	NO		NULL	

4) Adding Constraints :

```
ALTER TABLE Customer
ADD CONSTRAINT unique_phoneno UNIQUE (Customer_phoneno);
```

	Field	Type	Null	Key	Default	Extra
	Customer_name	varchar(50)	NO		NULL	
	Customer_email	varchar(100)	NO	UNI	NULL	
	Customer_phoneno	varchar(15)	YES	UNI	NULL	
	Customer_age	int	YES		NULL	
	Customer_gender	char(1)	YES		NULL	
	Customer_address	text	YES		NULL	
	Customer_password	varchar(255)	NO		NULL	

DQL (Data Query Language) and the SELECT Statement

DQL refers to the subset of SQL used to query the database and retrieve data. The SELECT statement is the primary DQL command used to retrieve data from a database. It is designed to fetch records from one or more tables based on specified criteria.

Syntax of the SELECT Statement

The basic syntax of the SELECT statement consists of the following components:

1. **SELECT**: Specifies the columns to be retrieved from the table.
2. **FROM**: Specifies the table from which the data will be fetched.
3. **WHERE** (optional): Filters the records based on a specified condition.
4. **ORDER BY** (optional): Sorts the result set based on one or more columns.

SELECT **Clause**

Example: Select specific columns:

```
771 •   SELECT Customer_name, Customer_email
772     FROM Customer;
773
774
```

Result Grid | Filter Rows: | Export: | Wrap Ce

Customer_name	Customer_email
Ajit Pandey	ajitpandey@gmail.com
Chirag Wamanacharya	chiragwamanacharya@gmail.com
Priya Mehta	priyamehta@gmail.com
Rajesh Iyer	rajeshiyer@gmail.com
Anjali Desai	anjalidesai@gmail.com
Vikram Kapoor	vikramkapoor@gmail.com
Sneha Rao	sneharao@gmail.com
Nilam Shah	nilam.shah@email.com
Amit Desai	amit.desai@email.com

Example: Select all columns:

```
select * from Customer;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	Customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandey@gmail.com	9819978432	28.00	M	Kalyan West	passwordAjit123
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com	9876543210	35.00	M	Kalyan East	passwordChirag456
3	Priya Mehta	priyamehta@gmail.com	9819978434	24.00	F	Andheri East	passwordPriya789
4	Rajesh Iyer	rajeshiyer@gmail.com	9819978435	35.00	M	Thane	passwordRajesh321
5	Anjali Desai	anjalidesai@gmail.com	9819978436	30.00	F	Dadar	passwordAnjali654
6	Vikram Kapoor	vikramkapoor@gmail.com	9819978437	27.00	M	Borivali	passwordVikram987
7	Sneha Rao	sneharao@gmail.com	9819978438	22.00	F	Bandra	passwordSneha123
8	Nilam Shah	nilam.shah@email.com	1122334455	30.00	F	101 East Ave, City, Country	password111
9	Amit Desai	amit.desai@email.com	5566778899	40.00	M	505 Central Ave, City, Country	password555
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

FROM Clause

```
SELECT Order_id, Total_amount
FROM Orders;
```

Result Grid | Filter Rows: | Edit:

Order_id	Total_amount
1	835.55
2	550.00
3	1000.05
4	1100.05
5	500.28
6	700.25
7	700.75
NULL	NULL

WHERE Clause

```
SELECT Customer_name, Customer_age
FROM Customer
WHERE Customer_age > 25;
```

Customer_name	Customer_age
Ajit Pandey	28
Chirag Wamanacharya	35
Rajesh Iyer	35
Anjali Desai	30
Vikram Kapoor	27
Nilam Shah	30
Amit Desai	40

ORDER BY Clause

```
SELECT Customer_name, Customer_age
FROM Customer
ORDER BY Customer_age ASC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell

Customer_name	Customer_age
Sneha Rao	22
Priya Mehta	24
Vikram Kapoor	27
Ajit Pandey	28
Anjali Desai	30
Nilam Shah	30
Chirag Wamanacharya	35
Rajesh Iyer	35
Amit Desai	40

```
SELECT Order_id, Total_amount
FROM Orders
ORDER BY Total_amount DESC;
```

Result Grid | Filter Rows: | Edit:

Order_id	Total_amount
4	1100.05
3	1000.05
1	835.55
7	700.75
6	700.25
2	550.00
5	500.28
NULL	NULL

Components of the WHERE **Clause**

1) Condition

```
SELECT Customer_name, Customer_age
FROM Customer
WHERE Customer_age > 25;
```

Result Grid | Filter Rows: | Export:

Customer_name	Customer_age
Ajit Pandey	28
Chirag Wamanacharya	35
Rajesh Iyer	35
Anjali Desai	30
Vikram Kapoor	27
Nilam Shah	30
Amit Desai	40

Logical Operators

Using AND:

```
SELECT Customer_name, Customer_age, Customer_address
FROM Customer
WHERE Customer_age > 25 AND Customer_address = 'Kalyan west';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_name	Customer_age	Customer_address
Ajit Pandey	28	Kalyan West

Using Or :

```
820 •   SELECT Customer_name, Customer_age, Customer_address
821     FROM Customer
822     WHERE Customer_age > 25 OR Customer_address = 'Dadar';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_name	Customer_age	Customer_address
Ajit Pandey	28	Kalyan West
Chirag Wamanacharya	35	Kalyan East
Rajesh Iyer	35	Thane
Anjali Desai	30	Dadar
Vikram Kapoor	27	Borivali
Nilam Shah	30	101 East Ave, City, Country
Amit Desai	40	505 Central Ave, City, Country

Using NOT:

```
823
824 •   SELECT Customer_name, Customer_address
825     FROM Customer
826     WHERE NOT Customer_address = 'Borivali';
827
828
```

Result Grid | Filter Rows: | Export: | Wrap Cell

Customer_name	Customer_address
Ajit Pandey	Kalyan West
Chirag Wamanacharya	Kalyan East
Priya Mehta	Andheri East
Rajesh Iyer	Thane
Anjali Desai	Dadar
Sneha Rao	Bandra
Nilam Shah	101 East Ave, City, Country
Amit Desai	505 Central Ave, City, Country

Using BETWEEN:

```
SELECT Customer_name, Customer_age
FROM Customer
WHERE Customer_age BETWEEN 20 AND 30;
```

Result Grid | Filter Rows: | Export:

Customer_name	Customer_age
Ajit Pandey	28
Priya Mehta	24
Anjali Desai	30
Vikram Kapoor	27
Sneha Rao	22
Nilam Shah	30

Using **LIKE**

```
SELECT Customer_name
FROM Customer
WHERE Customer_name LIKE 'A%';
```

Result Grid | Filter Rows: | Export:

Customer_name
Ajit Pandey
Anjali Desai
Amit Desai

IN for Checking Multiple Values

```
SELECT Order_id, Payment_method
FROM Orders
WHERE Payment_method IN ('Credit Card', 'PayPal', 'Bank Transfer');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | W

Order_id	Payment_method
1	Credit Card
3	Credit Card
5	PayPal
6	Credit Card
7	Bank Transfer
NULL	NULL

Arithmetic Operations

Addition:

```
select Item_price + Quantity as Total_Cost
from Cartitem;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Total_Cost
3000.99
1600.50
1102.00
1900.75
3500.00
2701.80
1201.99
3151.45

Subtraction:

```
select Item_price - (Item_price * Quantity) as Remaining_Amount
from cartitem;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Remaining_Amount
0.00
0.00
-1100.00
0.00
0.00
0.00
0.00
0.00

Multiplication:

```
select item_price * Quantity as Total_price
from cartitem;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conte

Total_price
2999.99
1599.50
2200.00
1899.75
3499.00
2700.80
1200.99
3150.45

Division:

```
select Item_price / Quantity as avg_price_per_item
from cartitem;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

avg_price_per_item
2999.990000
1599.500000
550.000000
1899.750000
3499.000000
2700.800000
1200.990000
3150.450000

Modulo (%)

```
select total_amount % 10 as Remainder
from orders;
```

Result Grid | Filter Rows: | Export:

Remainder
9.99
9.50
0.00
9.75
9.00
0.80
0.99
0.45

Comparison Operators:

1) Equal to (=):

```
select * from orders
where customer_id =1;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Order_id	Customer_id	Order_date	Order_status	Shipping_address	Total_amount	Payment_method
1	1	2024-03-01	Shipped	Mumbai, India	2999.99	Credit Card
NULL	NULL	NULL	NULL	NULL	NULL	NULL

2) Not equal to (! = or <>):

```
select * from products
where product_name!='Round glasses';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Product_id	Product_name	Product_brand	Product_type	Product_colour	Product_frametype	Product_lenstype	Product_material	Product_desc
1	Aviator Sunglasses	Ray-Ban	Eyewear	Black	Metal	Polarized	Aluminium	Classic aviator sunglasses
2	Wayfarer Sunglasses	Oakley	Eyewear	Brown	Plastic	Non-Polarized	Plastic	Trendy wayfarer style
4	Sports Sunglasses	Nike	Eyewear	Red	Plastic	UV Protection	Polycarbonate	Perfect for outdoor sports
5	Blue Light Glasses	Titan	Eyewear	Blue	Metal	Blue Light Filter	Aluminium	Reduces screen glare
6	Cat Eye Glasses	Prada	Eyewear	Pink	Plastic	Gradient	Acetate	Fashionable cat eye design
7	Oversized Glasses	Versace	Eyewear	Silver	Metal	Photochromic	Stainless Steel	Oversized modern look
8	Retro Glasses	Fastrack	Eyewear	Green	Plastic	Tinted	Resin	Retro-inspired stylish glasses
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3) Greater than (>):

```
select * from orders
where total_amount >2000;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Order_id	Customer_id	Order_date	Order_status	Shipping_address	Total_amount	Payment_method
1	1	2024-03-01	Shipped	Mumbai, India	2999.99	Credit Card
3	3	2024-03-03	Processing	Bangalore, India	2200.00	Credit card
5	5	2024-03-05	Cancelled	Pune, India	3499.00	Paypal
6	6	2024-03-06	Delivered	Hyderabad, India	2700.80	Credit card
8	8	2024-03-08	Shipped	Ahmedabad, India	3150.45	Debit Card
NULL	NULL	NULL	NULL	NULL	NULL	NULL

4) Less than (<):

```
select * from orders
where total_amount < 2000;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Order_id	Customer_id	Order_date	Order_status	Shipping_address	Total_amount	Payment_method
2	2	2024-03-02	Delivered	Delhi, India	1599.50	Debit Card
4	4	2024-03-04	Shipped	Chennai, India	1899.75	Net Banking
7	7	2024-03-07	Processing	Kolkata, India	1200.99	Bank Transfer
NULL	NULL	NULL	NULL	NULL	NULL	NULL

5) Greater than or equal to (>=):

```
select * from orders
where total_amount>=3499;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Order_id	Customer_id	Order_date	Order_status	Shipping_address	Total_amount	Payment_method
5	5	2024-03-05	Cancelled	Pune, India	3499.00	Paypal
NULL	NULL	NULL	NULL	NULL	NULL	NULL

6) Less than or equal to (<=):

```
select * from orders
where total_amount<=1300;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Order_id	Customer_id	Order_date	Order_status	Shipping_address	Total_amount	Payment_method
7	7	2024-03-07	Processing	Kolkata, India	1200.99	Bank Transfer
NULL	NULL	NULL	NULL	NULL	NULL	NULL

1) ORDER BY Clause

The ORDER BY clause is used to sort the result set based on one or more columns. Sorting can be in ascending (ASC) or descending (DESC) order. By default, ORDER BY sorts in ascending order if ASC or DESC isn't specified.

```
select order_id,total_amount
from orders
order by total_amount desc;
```

Result Grid | Filter Rows: | Export: | Wrap

order_id	total_amount
5	3499.00
8	3150.45
1	2999.99
6	2700.80
3	2200.00
4	1899.75
2	1599.50
7	1200.99

2) DISTINCT Keyword

The DISTINCT keyword is used to remove duplicate rows from the result set, ensuring that only unique values are returned for the specified columns.

```
select distinct order_status
from orders;
```

Result Grid | Filter Rows: | Export:

order_status
Shipped
Delivered
Processing
Cancelled

3) Limit Clause

The LIMIT clause limits the number of rows returned in the result set.

```
select customer_id,customer_name,customer_email
from customer
limit 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	customer_name	customer_email
1	Ajit Pandey	ajitpandey@gmail.com
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com
3	Priya Mehta	priyamehta@gmail.com
4	Rajesh Iyer	rajeshiyer@gmail.com
5	Anjali Desai	anjalidesai@gmail.com

DML

Single Row

INSERT statement in SQL is used to add new data rows into a table. For inserting a single row, we specify the target table, followed by the column names in which we want to add values, and then provide the values in the same order. This is a basic operation in Data Manipulation Language (DML) that helps populate tables with data.

```
INSERT INTO Customer ( Customer_name, Customer_email, Customer_phoneno,
Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES
( 'Amit Desai', 'amit.desai@email.com', '5566778899', 40, 'M', '505 Central Ave, City, Country', 'password55');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandey@gmail.com	9819978432	28	M	kalyan west	passwordAjit123
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com	9876543210	35	M	kalyan east	passwordChirag456
3	Priya Mehta	priyamehta@gmail.com	9819978434	24	F	Bangalore, India	passwordPriya789
4	Rajesh Iyer	rajeshiyer@gmail.com	9819978435	35	M	thane	passwordRajesh321
5	Anjali Desai	anjalidesai@gmail.com	9819978436	30	F	Dadar	passwordAnjali654
6	Vikram Kapoor	vikramkapoor@gmail.com	9819978437	27	M	Borivali	passwordVikram987
7	Sneha Rao	sneharao@gmail.com	9819978438	22	F	Kolkata, India	passwordSneha123
8	Nilam Shah	nilam.shah@email.com	1122334455	30	F	101,East Ave,City, Country	passwordNilam111
9	Amit Desai	amit.desai@email.com	5566778899	40	M	505 Central Ave, City, Country	password55

Multiple Row

```
INSERT INTO Customer (Customer_name, Customer_email, Customer_phoneno, Customer_age, Customer_gender, Customer_address, Customer_password)
VALUES('Boby Roy', 'boby.roy@email.com', '5546677899', 45, 'M', '515 Central Ave, City, Country', 'password585'),
 ('Sneha Reddy', 'sneha.reddy@email.com', '6677889900', 26, 'F', '606 Maple St, City, Country', 'password666');
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandey@gmail.com	9819978432	28	M	kalyan west	passwordAjit123
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com	9876543210	35	M	kalyan east	passwordChirag456
3	Priya Mehta	priyamehta@gmail.com	9819978434	24	F	Bangalore, India	passwordPriya789
4	Rajesh Iyer	rajeshiyer@gmail.com	9819978435	35	M	thane	passwordRajesh321
5	Anjali Desai	anjalidesai@gmail.com	9819978436	30	F	Dadar	passwordAnjali654
6	Vikram Kapoor	vikramkapoor@gmail.com	9819978437	27	M	Borivali	passwordVikram987
7	Sneha Rao	sneharao@gmail.com	9819978438	22	F	Kolkata, India	passwordSneha123
8	Nilam Shah	nilam.shah@email.com	1122334455	30	F	101,East Ave,City, Country	passwordNilam111
9	Amit Desai	amit.desai@email.com	5566778899	40	M	505 Central Ave, City, Country	password55
11	Boby Roy	boby.roy@email.com	5546677899	45	M	515 Central Ave, City, Country	password585
12	Sneha Reddy	sneha.reddy@email.com	6677889900	26	F	606 Maple St, City, Country	password666
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

1. UPDATE

The UPDATE keyword indicates the table where the records are to be modified.

2. SET

The SET keyword specifies the column(s) to be updated and assigns them new values. You can update multiple columns by separating them with commas.

3. WHERE

The WHERE clause is optional but highly recommended. It defines the conditions that the rows must meet to be updated. Without it, all rows in the table will be updated, which might not be intended.

Update a Single Column

```
update customer
set customer_email='ajitpandey123@gmail.com'
where customer_id=1;
select * from customer;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandey123@gmail.com	9819978432	28	M	kalyan west	passwordAjit123

Update Multiple Columns for a Specific Customer

```
update customer
set customer_phoneno='7798805101' , customer_age=40
where customer_id=2;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandey123@gmail.com	9819978432	28	M	kalyan west	passwordAjit123
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com	7798805101	40	M	kalyan east	passwordChirag456
3	Priya Mehta	priyamehta@gmail.com	9819978434	24	F	Bangalore, India	passwordPriya789
4	Rajesh Iyer	rajeshiyer@gmail.com	9819978435	35	M	thane	passwordRajesh321
5	Anjali Desai	anjalidesai@gmail.com	9819978436	30	F	Dadar	passwordAnjali654

Update Multiple Rows Based on a Condition

```
update customer
set customer_age=45
where customer_gender='f';
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandey123@gmail.com	9819978432	28	M	kalyan west	passwordAjit123
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com	7798805101	40	M	kalyan east	passwordChirag456
3	Priya Mehta	priyamehta@gmail.com	9819978434	45	F	Bangalore, India	passwordPriya789
4	Rajesh Iyer	rajeshiyer@gmail.com	9819978435	35	M	thane	passwordRajesh321
5	Anjali Desai	anjalidesai@gmail.com	9819978436	45	F	Dadar	passwordAnjali654
6	Vikram Kapoor	vikramkapoor@gmail.com	9819978437	27	M	Borivali	passwordVikram987
7	Sneha Rao	sneharao@gmail.com	9819978438	45	F	Kolkata, India	passwordSneha123
8	Nilam Shah	nilam.shah@email.com	1122334455	45	F	101,East Ave,City, Country	passwordNilam111
9	Amit Desai	amit.desai@email.com	5566778899	40	M	505 Central Ave, City, Country	password55
11	Boby Roy	boby.roy@email.com	5546677899	45	M	515 Central Ave, City, Country	password585
12	Sneha Reddy	sneha.reddy@email.com	6677889900	45	F	606 Maple St, City, Country	password666
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

The **DELETE** statement in SQL is used to remove rows from a table. There are two primary ways to use it:

1. **Deleting Specific Rows**: Use the DELETE statement with a WHERE clause to specify which rows to delete. Only rows that match the WHERE condition will be deleted.
2. **Deleting All Rows**: Use the DELETE statement without a WHERE clause to delete all rows in the table.

1. Deleting Specific Rows

```
delete from customer
where customer_id=12;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Customer_id	customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password
1	Ajit Pandey	ajitpandey123@gmail.com	9819978432	28	M	kalyan west	passwordAjit123
2	Chirag Wamanacharya	chiragwamanacharya@gmail.com	7798805101	40	M	kalyan east	passwordChirag456
3	Priya Mehta	priyamehta@gmail.com	9819978434	45	F	Bangalore, India	passwordPriya789
4	Rajesh Iyer	rajeshiyer@gmail.com	9819978435	35	M	thane	passwordRajesh321
5	Anjali Desai	anjalidesai@gmail.com	9819978436	45	F	Dadar	passwordAnjali654
6	Vikram Kapoor	vikramkapoor@gmail.com	9819978437	27	M	Borivali	passwordVikram987
7	Sneha Rao	sneharao@gmail.com	9819978438	45	F	Kolkata, India	passwordSneha123
8	Nilam Shah	nilam.shah@email.com	1122334455	45	F	101,East Ave,City, Country	passwordNilam111
9	Amit Desai	amit.desai@email.com	5566778899	40	M	505 Central Ave, City, Country	password55
11	Boby Roy	boby.roy@email.com	5546677899	45	M	515 Central Ave, City, Country	password585
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

2. Deleting All Rows

```
delete from customer;
```

Concatenate Two or More Strings

```
select concat(customer_name, '-' ,customer_email) as customer_details
from customer;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_details
Ajit Pandey-ajitpandey123@gmail.com
Chirag Wamanacharya-chiragwamanacharya@...
Priya Mehta-priyamehta@gmail.com
Rajesh Iyer-rajeshiyer@gmail.com
Anjali Desai-anjalidesai@gmail.com
Vikram Kapoor-vikramkapoor@gmail.com
Sneha Rao-sneharao@gmail.com
Nilam Shah-nilam.shah@email.com
Amit Desai-amit.desai@email.com
Bobby Roy-bobby.roy@email.com

Upper

```
select upper(customer_name) as upper_case_cust_name
from customer;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

upper_case_cust_name
AJIT PANDEY
CHIRAG WAMANACHARYA
PRIYA MEHTA
RAJESH IYER
ANJALI DESAI
VIKRAM KAPOOR
SNEHA RAO
NILAM SHAH
AMIT DESAI

Lower

```
311 •  select lower(customer_name) as lower_case_cust_name
312    from customer;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

lower_case_cust_name
ajit pandey
chirag wamanacharya
priya mehta
rajesh iyer
anjali desai
vikram kapoor
sneha rao
nilam shah
amit desai

Find the Length of a String

```
select customer_name,length(customer_name) as customer_length
from customer;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	customer_length
Ajit Pandey	11
Chirag Wamanacharya	19
Priya Mehta	11
Rajesh Iyer	11
Anjali Desai	12
Vikram Kapoor	13
Sneha Rao	9
Nilam Shah	10
Amit Desai	10

Extract a Substring from a String

```
select customer_name,substring(customer_name,1,5) as Substring_name
from customer;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	Substring_name
Ajit Pandey	Ajit
Chirag Wamanacharya	Chira
Priya Mehta	Priya
Rajesh Iyer	Rajes
Anjali Desai	Anjal
Vikram Kapoor	Vikra
Sneha Rao	Sneha
Nilam Shah	Nilam
Amit Desai	Amit
Bobv Roy	Bobv

COUNT( ): Counts the Number of Rows

```
select count(*) as total_rows_in_customer_table
from customer;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_rows_in_customer_table
10

SUM (): Calculates the Sum of Values

```
select sum(total_amount) as total_sum
from orders;
```

Result Grid | Filter Rows: | Export: | Wrap

total_sum
19250.48

AVG (): Calculates the Average Value

```
select avg(total_amount) as avg_amount
from orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

avg_amount
2406.310000

MIN (): Finds the Minimum Value

```
select min(total_amount) as minimumn_total
from orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Cor

minimumn_total
1200.99

MAX (): Finds the Maximum Value

```
select max(total_amount) as maximum_total
from orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conten

maximum_total
3499.00

```
select count(*) as total_orders,
sum(total_amount) as total_revenue,
min(total_amount) as minimum_order,
max(total_amount) as maximum_order,
avg(total_amount) as avg_of_order
from orders;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

total_orders	total_revenue	minimum_order	maximum_order	avg_of_order
8	19250.48	1200.99	3499.00	2406.310000

Group By

```
select customer_id,sum(total_amount) as total_spent
from orders
group by customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	total_spent
1	2999.99
2	1599.50
3	2200.00
4	1899.75
5	3499.00
6	2700.80
7	1200.99
8	3150.45

Having:

```
select customer_id,sum(total_amount) as total_spent
from orders
group by customer_id
having sum(total_amount)>2000;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content

customer_id	total_spent
1	2999.99
3	2200.00
5	3499.00
6	2700.80
8	3150.45

Single Sub Query

```
select c.customer_id ,p.paymentmethod
from customer c join payment p on c.customer_id=p.customer_id
where p.paymentmethod=(select paymentmethod
                       from payment
                       where customer_id=4);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	paymentmethod
4	Net Banking

Multiple Sub Query

```
select c.customer_name
from customer c
where c.customer_id in (select o.customer_id
                        from orders o
                        where o.order_status in ('shipped','pending'));
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name
Ajit Pandey
Rajesh Iyer
Nilam Shah

Joins

a) Inner Join

```
select c.customer_name,o.order_id
from customer c inner join orders o on c.customer_id=o.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	order_id
Ajit Pandey	1
Chirag Wamanacharya	2
Priya Mehta	3
Rajesh Iyer	4
Anjali Desai	5
Vikram Kapoor	6
Sneha Rao	7
Nilam Shah	8

2) Left Join (Left Outer Join)

```
select c.customer_name,o.order_id
from customer c left join orders o on c.customer_id=o.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	order_id
Ajit Pandey	1
Chirag Wamanacharya	2
Priya Mehta	3
Rajesh Iyer	4
Anjali Desai	5
Vikram Kapoor	6
Sneha Rao	7
Nilam Shah	8

3) Right Join (Right Outer Join)

```
select c.customer_name,o.order_id
from customer c right join orders o on c.customer_id=o.customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	order_id
Ajit Pandey	1
Chirag Wamanacharya	2
Priya Mehta	3
Rajesh Iyer	4
Anjali Desai	5
Vikram Kapoor	6
Sneha Rao	7
Nilam Shah	8

4) Cross Join

```
select c.customer_name,p.product_id
from customer c cross join products p;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Conter

customer_name	product_id
Ajit Pandey	8
Ajit Pandey	7
Ajit Pandey	6
Ajit Pandey	5
Ajit Pandey	4
Ajit Pandey	3
Ajit Pandey	2
Ajit Pandey	1
Chirag Wamanacharya	8
Chirag Wamanacharya	7
Chirag Wamanacharya	6

5) Full Join (Full Outer Join)

```sql
SELECT c.Customer_name, o.Order_id
FROM Customer c
LEFT JOIN Orders o ON c.Customer_id = o.Customer_id

UNION

SELECT c.Customer_name, o.Order_id
FROM Customer c
RIGHT JOIN Orders o ON c.Customer_id = o.Customer_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_name	Order_id
Ajit Pandey	1
Ajit Pandey	3
Ajit Pandey	5
Chirag Wamanacharya	2
Chirag Wamanacharya	4
Priya Mehta	6
Rajesh Iyer	7
Anjali Desai	NULL
Vikram Kapoor	NULL
Sneha Rao	NULL
Nilam Shah	NULL

Simple Views

```
create view customer_view
as
select customer_name,customer_email,customer_phoneno
from customer;
select * from customer_view;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_name	customer_email	customer_phoneno
Ajit Pandey	ajitpandey123@gmail.com	9819978432
Chirag Wamanacharya	chiragwamanacharya@gmail.com	7798805101
Priya Mehta	priyamehta@gmail.com	9819978434
Rajesh Iyer	rajeshiyer@gmail.com	9819978435
Anjali Desai	anjalidesai@gmail.com	9819978436
Vikram Kapoor	vikramkapoor@gmail.com	9819978437
Sneha Rao	sneharao@gmail.com	9819978438
Nilam Shah	nilam.shah@email.com	1122334455
Amit Desai	amit.desai@email.com	5566778899

Complex Views

```
CREATE VIEW OrderSummary AS
SELECT c.Customer_id, c.Customer_name, SUM(o.Total_amount) AS Total_spent
FROM Customer c
JOIN Orders o ON c.Customer_id = o.Customer_id
GROUP BY c.Customer_id, c.Customer_name;

select * from ordersummary;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_id	Customer_name	Total_spent
1	Ajit Pandey	2335.88
2	Chirag Wamanacharya	1650.05
3	Priya Mehta	700.25
4	Rajesh Iyer	700.75

Stored Procedure

```
delimiter &&
create procedure Order_detail()
begin
select order_id,Customer_id,Order_id,Order_status,Shipping_address from Orders;
end &&


call Order_detail();
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_id	Customer_id	Order_id	Order_status	Shipping_address
1	1	1	Shipped	123 Maple Street
2	2	2	Pending	456 Oak Avenue
3	1	3	Completed	123 Maple Street
4	2	4	Cancelled	456 Oak Avenue
5	1	5	Pending	123 Maple Street
6	3	6	Shipped	789 Pine Lane
7	4	7	Completed	101 Maple Street

TRIGGERS

A trigger in SQL is a set of actions or SQL statements that are automatically executed (or "triggered") in response to specific events occurring in a database.

```
delimiter //
create trigger audit_customer
after insert on customer
for each row
begin
   insert into audit_customer(customer_id,Customer_name,Customer_email,Customer_phoneno,
   Customer_age,Customer_gender,Customer_address,Customer_password,action_time,user_name,action)
   values(new.customer_id,new.customer_name,new.customer_email,new.Customer_phoneno,new.Customer_age,
   new.Customer_gender,new.Customer_address,new.Customer_password,current_time(),current_user(),'Inserted');
end //
delimiter ;

select * from customer;
select * from audit_customer;
drop trigger audit_customer;

insert into customer values(101,'Vivek','vivek@gmail.com','7798805101',22,'M','Titwala','Vivek@123');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Customer_id	Customer_name	Customer_email	Customer_phoneno	Customer_age	Customer_gender	Customer_address	Customer_password	action_time	user_name	action
101	Vivek	vivek@gmail.com	7798805101	22	M	Titwala	Vivek@123	2025-03-04 12:18:56	root@localhost	Inserted

Thank You

Mihir Ughade