Create db: create database nextDB;

use nextDB;

create table Customers;

create table Customers(

customerID int unique not null primary key auto_increment,

CustomerName varchar(20),

City varchar(20),

PostalCode int

);

Insert:

insert into Customers(CustomerName,City,PostalCode) values ('Bhargov','Guwahati',5653);

insert into Customers(CustomerName,City,PostalCode) values ('Sandip','Silchar',7657);

insert into Customers(CustomerName,City,PostalCode) values ('Bhargov','Guwahati',5653);

insert into Customers(Customername,City) values ('Himanka','Rangia');

Select: select * from Customers;

Select Distinct: Select only different values.

select distinct CustomerName from Customers;

Where: Used to filter records

select CustomerName from Customers where City = 'Guwahati';

select distinct(CustomerName) from Customers where City = 'Guwahati';

AND OR NOT:AND, OR Used to filter records based on more than one conditions. Not:Displays result if condition is not true.

select * from Customers where CustomerName ='Sandip' and City='Silchar';

select CustomerName from Customers where CustomerName ='Sandip' or City='Guwahati';

select * from Customers where not CustomerName ='Sandip';

Order by: sorts results in asc or desc order.

select * from Customers order by PostalCode;

select * from Customers order by PostalCode desc;

select * from Customers order by City, PostalCode;

Null: Field with no value.

```
select * from Customers where PostalCode is null;
select * from Customers where PostalCode is not null;
```

Update: used to modify existing records in a table.

```
update Customers set city = 'Tezpur' where City = 'Rangia';
update Customers set city = 'Guwahati' where customerID = 2;
update Customers set PostalCode = '-' ;
```

Delete: Delete existing records in a table.

```
delete from Customers where customerID = 2;
```

Delete all records: delete from Customers;

Limit: Specify number of records to return.

```
select * from Customers limit 2;
```

Min, Max function: Min function returns the smallest value.

```
select MIN(PostalCode) as MinPostal from Customers;
select Max(PostalCode) as MaxPostal from Customers;
```

Count(),Avg(),Sum():

```
select count(City) from Customers;
select avg(PostalCode) from Customers;
select sum(customerID) from Customers;
```

Like: Used in a where clause to search for a specified pattern in column.

```
select * from Customers where CustomerName like 'b%';
select * from Customers where CustomerName like '%v';
```

IN: Used in where clause to find multiple values.

```
select * from Customers where City in('Guwahati','Rangia');
```

Between:

```
select * from Customers where customerID between 5 and 7;
```

Alias: Temporary name to a table or a column.

```
select customerID as ID, CustomerName as Name from Customers;
```

Group by:

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country;
```

Having: The HAVING clause was added to SQL because the WHERE keyword cannot be used with aggregate functions.

```
SELECT COUNT(CustomerID), Country
FROM Customers
GROUP BY Country
HAVING COUNT(CustomerID) > 5;
```

Exists: The EXISTS operator is used to test for the existence of any record in a subquery.

```
SELECT SupplierName
FROM Suppliers
WHERE EXISTS (Price < 20);
```

Case:

```
SELECT OrderID, Quantity,

CASE WHEN Quantity > 30 THEN 'The quantity is greater than 30'

WHEN Quantity = 30 THEN 'The quantity is 30'

ELSE 'The quantity is under 30'

END AS QuantityText

FROM OrderDetails;
```