1) ADD a column in the table

**ALTER** **TABLE** cus\_tbl

**ADD** cus\_age **varchar**(40) NOT NULL;

Similary we can add multiple coomn separated by colon

## MODIFY column definition in the table

**ALTER** **TABLE** cus\_tbl

**MODIFY** cus\_surname **varchar**(50) NULL;

## DROP column in table

**ALTER** **TABLE** cus\_tbl

**DROP** **COLUMN** cus\_address;

## RENAME column in table

**ALTER** **TABLE**  cus\_tbl

CHANGE **COLUMN** cus\_surname cus\_title  **varchar**(20) NOT NULL;

## RENAME table

**ALTER** **TABLE** cus\_tbl

RENAME **TO** cus\_table;

Sequence of wriring Query=

SELECT->FROM->WHERE->GROUP BY->HAVING and ORDER BY

**1-a: Write a SQL query to fetch top n records(MySQL)?**

SELECT \*

FROM EmployeeSalary

ORDER BY Salary DESC LIMIT N;

**1-b: Write a SQL query to fetch top n records (SQL Server)?**

 SELECT DISTINCT TOP N salary FROM #Employee ORDER BY salary DESC;

**2-a: Write SQL query to find the nth highest salary from table (MySql).**

SELECT Salary

FROM Employee

ORDER BY Salary DESC LIMIT N-1, 1;

**3rd highest salary in MySQL using LIMIT clause:**

SELECT salary FROM Employee ORDER BY salary DESC LIMIT 2,1;

**2-b: Write SQL query to find the nth highest salary from table(SQL Server).**  
  
SELECT TOP 1 salary FROM

(SELECT DISTINCT TOP N salary FROM #Employee ORDER BY salary DESC )

AS temp ORDER BY salary [ASC];

**Write SQL query to find the 3rd highest salary from table without using TOP/limit keyword.**

SELECT Salary

FROM EmployeeSalary Emp1

WHERE N-1 = (

                SELECT COUNT( DISTINCT ( Emp2.Salary ) )

                FROM EmployeeSalary Emp2

                WHERE Emp2.Salary > Emp1.Salary

            )

**Find 3rd highest salary**

SELECT Salary

FROM EmployeeSalary Emp1

WHERE 2 = (

                SELECT COUNT( DISTINCT ( Emp2.Salary ) )

                FROM EmployeeSalary Emp2

                WHERE Emp2.Salary > Emp1.Salary

            )

**Write a SQL query to create an empty table with the same structure as some other table.**

CREATE TABLE ‘newTable’

LIKE ‘EmployeeDetails’;

**Write a SQL query to create a new table with data and structure copied from another table.**

SELECT \* INTO newTable

FROM EmployeeDetails;

**Write a SQL query to fetch common records between two tables.**  
Ans. **Using INTERSECT-**

SELECT \* FROM EmployeeSalary

**INTERSECT**

SELECT \* FROM ManagerSalary

**Write a SQL query to fetch records that are present in one table but not in another table.**

SELECT \* FROM EmployeeSalary

MINUS

SELECT \* FROM ManagerSalary

**Write a SQL query to fetch all the Employees details from EmployeeDetails table who joined in the Year 2016.**

**we can extract year part from the joining date (using YEAR in mySQL)**

SELECT \* FROM EmployeeDetails

WHERE YEAR(DateOfJoining) = '2016';

**Write a SQL query to fetch top n records?**

SELECT \*

FROM EmployeeSalary

ORDER BY Salary DESC LIMIT N

#### **Write an SQL query to print the first three characters of  FIRST\_NAME from Worker table.**

Select substring(FIRST\_NAME,1(start),3(no. of char.)) from Worker;

**Write a SQL query to fetch employee names having a salary greater than or equal to 5000 and less than or equal 10000.**

SELECT FullName

FROM **EmployeeDetails(Table 1)**

WHERE EmpId IN

(SELECT EmpId FROM **EmpolyeeSalary (Table 2)**

WHERE Salary BETWEEN 5000 AND 10000);

**Write a query to fetch employee names and salary records. Return employee details even if the salary record is not present for the employee.**

SELECT E.FullName, S.Salary

FROM EmployeeDetails E LEFT JOIN EmployeeSalary S

ON E.EmpId = S.EmpId;

**Write a SQL query to fetch all the Employees who are also managers from EmployeeDetails table.(IMP bz here same table use)**

**Here, we have to use Self-Join**

SELECT **DISTINCT** E.FullName

FROM EmpDetails E

INNER JOIN EmpDetails M **//Manager also Employee**

ON E.EmpID = M.ManagerID;

**Write a SQL query to fetch duplicate records from a table.**

SELECT EmpId, Project, Salary, COUNT(\*)

FROM EmployeeSalary

GROUP BY EmpId, Project, Salary

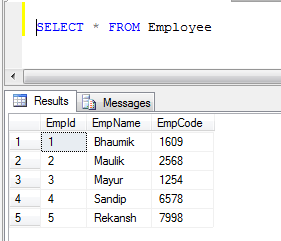
HAVING COUNT(\*) > 1;

UNION and UNION ALL

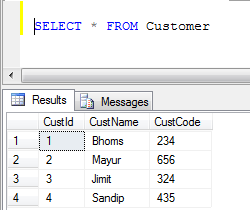
* UNION performs a DISTINCT on the result set, eliminating any duplicate rows.
* UNION ALL does not remove duplicates, and it therefore faster than UNION.
* **Note:** While using this commands all **selected columns** need to be of the same data type.
* **Number of column is same but name can be different.**
* **So UNION ALL is faster**

Example: If we have two tables

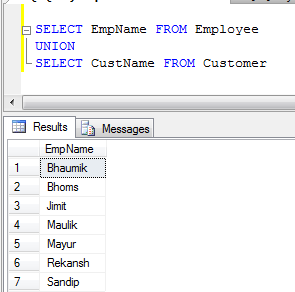
1. Employee table data:



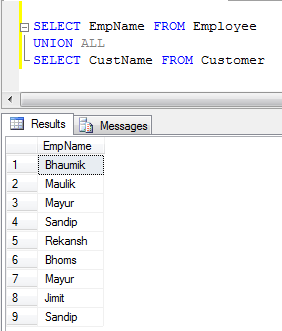
1. Customer table data:



1. UNION Example (It removes all duplicate records):



1. UNION ALL Example (It just concatenate records, not eliminate duplicates, so it is faster than UNION):



***Insert Into ... Select*** **statement.**

INSERT INTO "table1" ("column1", "column2", ...)

SELECT "column3", "column4", ...

FROM "table2"

## What is a Datawarehouse?

**Datawarehouse** refers to a central repository of data where the data is assembled from multiple sources of information. Those data are consolidated, transformed and made available for the mining as well as online processing. Warehouse data also have a subset of data called **Data Marts.**

## How can you fetch alternate records from a table?

You can fetch alternate records i.e both odd and even row numbers. For example- To display even numbers, use the following command:

Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=0

Now, to display odd numbers:

Select studentId from (Select rowno, studentId from student) where mod(rowno,2)=1

**In MySQL:**

SELECT \* FROM employee WHERE MOD(id,2)=0;//even

SELECT \* FROM employee WHERE MOD(id,2)=1;//odd