**Task-1: create a table with name products and entries are**

ProductID, Product name, supplier ID, categoryID, Quantity per unit, unit price, units in stock, units on order, reorder level, discontinued.

**solution**

Indexes: primary key product name, foreign key is products categories & products suppliers.

Queries:

1. Write a mysql query to get Product name and quantity/unit.

select product\_name,quality from product;

1. Write a MySQL query to get current Product list (Product ID and name).

select product\_id,product\_name from product;

1. Write a MySQL query to get discontinued Product list (Product ID and name).

select product.product\_id,product.product\_name,entries.supplier\_name from product join entries on (entries.ent\_id=product.product\_id);

1. Write a MySQL query to get most expense and least expensive Product list (name and unit price).

select cost\_product,product\_name from product order by cost\_product desc limit 1;

1. Write a MySQL query to get Product list (id, name, unit price) where current products cost less than 20 rupees.

select product\_id, product\_name,cost\_product from product where cost\_product<20;

**Task-2:**

1. Create a table name departments with primary key column(department\_id).

**solution**

create table departments(department\_id int primary key);

1. Columns in the table should be department\_id, department\_name & location\_id.

Hint:

CREATE TABLE departments

( department\_id INTEGER PRIMARY KEY

, department\_name VARCHAR(30)

, location\_id INTEGER

) ;

1. Create another table with name employees with a foreign key.

Hint:

CREATE TABLE employees

( employee\_id INTEGER

, first\_name VARCHAR(20)

, last\_name VARCHAR(25)

, email VARCHAR(25)

, phone\_number VARCHAR(20)

, hire\_date DATE

, job\_id VARCHAR(10)

, salary INTEGER

, commission\_pct INTEGER

, manager\_id INTEGER

s

, constraint pk\_emp primary key (employee\_id)

, constraint fk\_deptno foreign key (department\_id) references departments(department\_id)

) ;

1. **Insert 16 Records into departments Table.**

**Solution**

insert into department values(101,'ÇSE','5001'),

(102,'ECE','4001'),

(103,'EEE','3001'),

(104,'CSM','4001'),

(105,'CSE','5001'),

(106,'ECE','4001'),

(107,'EEE','3001'),

(108,'CSM','4001'),

(109,'CSE','5001'),

(110,'EEE','3001'),

(111,'CSM','4001'),

(112,'CSE','5001'),

(113,'ECE','4001'),

(114,'EEE','3001'),

(115,'CSM','4001'),

(116,'CSE','5001');

1. **Insert 20 Records into employees Table.**

**Solution**

INSERT INTO employees

VALUES

(1, 'koya', 'madhuri', 'koyamadhuri@gmail.com', '9948246214', '2024-02-01', 'J001', 40000, 10000, 201, 101),

(2, 'koya', 'malathi', 'koyamalathi@gmail.com', '9948246211', '2021-02-01', 'J002', 40000, 10000, 202, 102),

(3, 'koya', 'mounika', 'koyamounika@gmail.com', '9948246212', '2024-06-01', 'J003', 45000, 15000, 203, 103),

(4, 'koya', 'srilaxmi', 'koyasrilaxmi@gmail.com', '9948246213', '2025-02-01', 'J004', 30000, 12000, 204, 104),

(5, 'vinjam', 'akhila', 'vinjamakhila@gmail.com', '9948246215', '2023-03-01', 'J005', 34000, 11000, 205, 105),

(6, 'kurry', 'bhargavi', 'kurrybhargavi@gmail.com', '9948246216', '2021-02-01', 'J006', 30000, 10000, 206, 106),

(7, 'pasupulati', 'alekhya', 'koyamadhuri@gmail.com', '9948246217', '2021-02-01', 'J007', 40000, 10000, 207, 107),

(8, 'saminani', 'sruthi', 'saminanisruthi@gmail.com', '9948246218', '2021-06-22', 'J008', 30000, 10000, 208, 108),

(9, 'kolli', 'chandana', 'kollichandana@gmail.com', '9948246219', '2021-09-09', 'J009', 45000, 12000, 209, 109),

(10, 'eputi', 'priyanka', 'epuripriyanka@gmail.com', '9848246214', '2021-11-04', 'J010', 30000, 10000, 210, 110),

(11,'apuri','surramma','suramma@gmail.com','9848246214', '2021-11-04', 'J010', 30000, 10000, 211, 111),

(12, 'vakayalapati', 'hrydayansh', 'hridayansh@gmail.com', '9648246214', '2021-10-01', 'J012', 40000, 10000, 212, 112),

(13, 'kotala', 'sravya', 'kotalasravya@gmail.com', '9548246214', '2021-09-01', 'J013', 30000, 10000, 213, 113),

(14, 'vadale', 'sarika', 'vadalesarika@gmail.com', '9448246214', '2021-08-01', 'J014', 30000, 12000, 214, 114),

(15, 'talure', 'kavya', 'talurekavya@gmail.com', '9348246214', '2022-02-01', 'J015', 45000, 15000, 215, 115),

(16, 'bajavada', 'srija', 'bajavadasrija@gmail.com', '9248246214', '2023-02-11', 'J016', 45000, 10000, 216, 116),

(17, 'garalapati', 'divya', 'garalapatidivya@gmail.com', '9148246214', '2021-02-01', 'J017', 30000, 10000, 217, 117),

(18, 'ravella', 'jaya', 'ravellajaya@gmail.com', '9948246214', '2022-02-01', 'J018', 20000, 10000, 218, 118),

(19, 'ravella', 'janvi', 'ravellajanvi@gmail.com', '9958246214', '2023-06-01', 'J019', 30000, 13000, 219, 119),

(20, 'ravella', 'geethu', 'ravallageethu@gmail.com', '9968246214', '2025-02-01', 'J020', 40000, 15000, 220, 120);

Queries:

1. **Select employees first name, last name, job\_id and salary whose first name starts with alphabet S.**

**Solution**

select first\_name,last\_name,job\_id,salary from employees where first\_name like 's%';

1. **Write a query to select employee with the highest salary**.

**Solution**

select max(salary) from employees;

1. **Select employee with the second highest salary**

**Solution**

select salary from employees order by salary desc limit 1 offset 1;

1. **Fetch employees with 2nd or 3rd highest salary**.

**Solution**

select salary from employees order by salary desc limit 2 offset 1;

1. **Write a query to select employees and their corresponding managers and their salaries.**

Now, this is a classic example of **SELF JOIN** in SQL exercises. Also, use the **CONCAT** function to concatenate the first name and last name of each employee and manager.

**Solution**

SELECT CONCAT(e.first\_name, ' ', e.last\_name) AS employee\_name,e.salary AS employee\_salary,

CONCAT(m.first\_name, ' ', m.last\_name) AS manager\_name,m.salary AS manager\_salary

FROM employees e

LEFT JOIN employees m ON e.manager\_id = m.employee\_id;

1. **Write a query to show count of employees under each manager in descending order.**

**Solution**

select count(employee\_id) from employees order by manager\_id desc;

1. **Find the count of employees in each department**.

**Solution**

select count(department\_id) from employees;

1. **Get the count of employees hired year wise.**
2. **Find the salary range of employees**.

**Solution**

select \* from employees where salary>40000;

1. **Write a query to divide people into three groups based on their salaries.**

Solution

SELECT \* FROM employees WHERE join\_date BETWEEN '1994-08-01' AND '1994-08-31';

1. **Select the employees whose first\_name contains “an”**.

**Solution**

select first\_name from employees where first\_name like '%an%';

1. **Select employee first name and the corresponding phone number in the format (\_ \_ \_)-(\_ \_ \_)-(\_ \_ \_ \_).**
2. **Find the employees who joined in August, 1994.**

**Solution**

SELECT \* FROM employees WHERE join\_date BETWEEN '1994-08-01' AND '1994-08-31';

1. **Write an SQL query to display employees who earn more than the average salary in that company**.

**Solution**

SELECT \* FROM employees WHERE salary > (SELECT AVG(salary) FROM employees);

1. **Find the maximum salary from each department.**

**Solution**

Select max(salary) from department;

1. **Write a SQL query to display the 5 least earning employees**.

**Solution**

SELECT \* FROM employees ORDER BY salary ASC LIMIT 5;

1. **Find the employees hired in the 80s.**
2. **Display the employees first name and the name in reverse order**.
3. **Find the employees who joined the company after 15th of the month.**
4. **Display the managers and the reporting employees who work in different departments**.