Question - 02:

Firstly create the Database and add data using these SQL Queries and adding data into tables:

```
use master
GO
CREATE DATABASE [Workforce_Management_System];
GO
USE [Workforce_Management_System];

-- Create Tables with their Schema
CREATE TABLE [Jobs]
(
    [JOB_ID] VARCHAR(100) PRIMARY KEY NOT NULL,
    JOB_TITLE VARCHAR(100) NOT NULL,
    MIN_SALARY INT NOT NULL,
    MAX_SALARY INT NOT NULL
);

CREATE TABLE [Locations]
(
    [LOCATION_ID] INT PRIMARY KEY NOT NULL,
    STREET_ADDRESS VARCHAR(100) NOT NULL,
    POSTAL_CODE VARCHAR(100) DEFAULT NULL,
    CITY VARCHAR(100) NOT NULL,
    STATE_PROVINCE VARCHAR(100) DEFAULT NULL,
    COUNTRY_ID CHAR(2) NOT NULL
);

CREATE TABLE [Departments]
(
    [DEPARTMENT_ID] INT PRIMARY KEY NOT NULL,
    DEPARTMENT_NAME VARCHAR(100) NOT NULL,
    MANAGER_ID INT NOT NULL,
    LOCATION_ID INT NOT NULL
    FOREIGN KEY (LOCATION_ID) REFERENCES Locations(LOCATION_ID)
);

CREATE TABLE [Employee]
(
    [EMPLOYEE_ID] INT PRIMARY KEY NOT NULL,
    FIRST_NAME VARCHAR(100) NOT NULL,
    LAST_NAME VARCHAR(100) NOT NULL,
    EMAIL VARCHAR(100) NOT NULL,
    PHONE_NUMBER VARCHAR(100) NOT NULL,
    HIRE_DATE DATE NOT NULL,
    JOB_ID VARCHAR(100) NOT NULL,
    SALARY INT NOT NULL,
```

```
    COMMISION_PCT INT NOT NULL DEFAULT 0,
    MANAGER_ID INT NOT NULL,
    DEPARTMENT_ID INT NOT NULL
    FOREIGN KEY (JOB_ID) REFERENCES Jobs(JOB_ID),
    FOREIGN KEY (DEPARTMENT_ID) REFERENCES Departments(DEPARTMENT_ID)
);

-- Entering Values into the Tables
INSERT INTO Jobs(JOB_ID, JOB_TITLE, MIN_SALARY, MAX_SALARY) VALUES
('AD_PRES', 'President', 20000, 40000),
('AD_VP', 'Administration Vice President', 15000, 30000),
('AD_ASST', 'Administration Assistant', 3000, 6000),
('FI_MGR', 'Finance Manager', 8200, 16000),
('FI_ACCOUNT', 'Accountant', 4200, 9000),
('AC_MGR', 'Accounting Manager', 8200, 16000),
('AC_ACCOUNT', 'Public Accountant', 4200, 9000),
('SA_MAN', 'Sales Manager', 10000, 20000),
('SA_REP', 'Sales Representative', 6000, 12000),
('PU_MAN', 'Purchasing Manager', 8000, 15000),
('PU_CLERK', 'Purchasing Clerk', 2500, 5500),
('ST_MAN', 'Stock Manager', 5500, 8500),
('ST_CLERK', 'Stock Clerk', 2000, 5000),
('SH_CLERK', 'Shipping Clerk', 2500, 5500),
('IT_PROG', 'Programmer', 4000, 10000),
('MK_MAN', 'Marketing Manager', 9000, 15000),
('MK_REP', 'Marketing Representative', 4000, 9000),
('HR_REP', 'Human Resources Representative', 4000, 9000),
('PR_REP', 'Public Relations Representative', 4500, 10500);

INSERT INTO Locations(LOCATION_ID, STREET_ADDRESS, POSTAL_CODE, CITY,
STATE_PROVINCE, COUNTRY_ID) VALUES
(1000, '1297 Via Cola di Rie', '00989', 'Roma', NULL, 'IT'),
(1100, '93091 Calle della Testa', '10934', 'Venice', NULL, 'IT'),
(1200, '2017 Shinjuku-ku', '1689', 'Tokyo', 'Tokyo Perfecture', 'JP'),
(1300, '9450 Kamiya-cho', '6823', 'Hiroshima', NULL, 'JP'),
(1400, '2014 Jabberwocky Rd', '26192', 'Southlake', 'Texas', 'US'),
(1500, '2011 Interiors Blvd', '99236', 'South San Francisco', 'California',
'US'),
(1600, '2007 Zagora St', '50090', 'South Brunswick', 'New Jersey', 'US'),
(1700, '2004 Charade Rd', '98199', 'Seattle', 'Washington', 'US'),
(1800, '147 Spadina Ave', 'M5V 2L7', 'Toronto', 'Ontario', 'CA'),
(1900, '6092 Boxwood St', 'YSW 9T2', 'Whitehorse', 'Yukon', 'CA'),
(2000, '40-5-12 Laogianggen', '190518', 'Beijing', NULL, 'CN'),
(2100, '1298 Vileparle (E)', '490231', 'Mumbai', 'Maharashtra', 'IN'),
(2200, '12-98 Victoria Street', '2901', 'Sydney', 'New South Wales', 'AU'),
(2300, '198 Clementi North', '540198', 'Singapore', NULL, 'SG'),
(2400, '8204 Arthur St', NULL, 'London', NULL, 'UK'),
(2500, 'Magdalen Centre, The Oxford Science Park', 'OX9 9ZB', 'Oxford',
```

```
'Oxford', 'UK'),
(2600, '9702 Chester Road', '9629850293', 'Stretford', 'Manchester', 'UK'),
(2700, 'Schwanthalerstr. 7031', '80925', 'Munich', 'Bavaria', 'DE'),
(2800, 'Rua Frei Caneca 1360', '01307-002', 'Sao Paulo', 'Sao Paulo', 'BR'),
(2900, '20 Rue des Corps-Saints', '1730', 'Geneva', 'Geneve', 'CH'),
(3000, 'Murtenstrasse 921', '3095', 'Bern', 'BE', 'CH'),
(3100, 'Pieter Breughelstraat 837', '3029SK', 'Utrecht', 'Utrecht', 'NL'),
(3200, 'Mariano Escobedo 9991', '11932', 'Mexico City', 'Distrito Federal
Mexico', 'MX');

INSERT INTO Departments(DEPARTMENT_ID, DEPARTMENT_NAME, MANAGER_ID,
LOCATION_ID) VALUES
(10, 'Administration', 200, 1700),
(20, 'Marketing', 201, 1800),
(30, 'Purchasing', 114, 1700),
(40, 'Human Resources', 203, 2400),
(50, 'Shipping', 121, 1500),
(60, 'IT', 103, 1400),
(70, 'Public Relations', 204, 2700),
(80, 'Sales', 145, 2500),
(90, 'Executive', 100, 1700),
(100, 'Finance', 108, 1700),
(110, 'Accounting', 205, 1700),
(120, 'Treasury', 0, 1700),
(130, 'Corporate Tax', 0, 1700),
(140, 'Control And Credit', 0, 1700),
(150, 'Shareholder Services', 0, 1700),
(160, 'Benefits', 0, 1700);

INSERT INTO Employee(EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL,
PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISION_PCT, MANAGER_ID,
DEPARTMENT_ID) VALUES
(100, 'Steven', 'King', 'SKING', '515.123.4567', '1987-06-17', 'AD_PRES',
24000, 0, 0, 90),
(101, 'Neena', 'Kochhar', 'NKOCHHAR', '515.123.4568', '1987-06-18', 'AD_VP',
17000, 0, 100, 90),
(102, 'Lex', 'De Haan', 'LDEHAAN', '515.123.4569', '1987-06-19', 'AD_VP',
17000, 0, 100, 90),
(103, 'Alexander', 'Hunold', 'AHUNOLD', '590.423.4567', '1987-06-20',
'IT_PROG', 9000, 0, 102, 60),
(104, 'Bruce', 'Ernst', 'BERNST', '590.423.4568', '1987-06-21', 'IT_PROG',
6000, 0, 103, 60),
(105, 'David', 'Austin', 'DAUSTIN', '590.423.4569', '1987-06-22', 'IT_PROG',
4800, 0, 103, 60),
(106, 'Valli', 'Pataballa', 'VPATABAL', '590.423.4560', '1987-06-23',
'IT_PROG', 4800, 0, 103, 60),
(107, 'Diana', 'Lorentz', 'DLORENTZ', '590.423.5567', '1987-06-24',
'IT_PROG', 4200, 0, 103, 60),
```

```
(108, 'Nancy', 'Greenberg', 'NGREENBE', '515.124.4569', '1987-06-25',
'FI_MGR', 12000, 0, 101, 100),
(109, 'Daniel', 'Faviet', 'DFAVIET', '515.124.4169', '1987-06-26',
'FI_ACCOUNT', 9000, 0, 108, 100),
(110, 'John', 'Chen', 'JCHEN', '515.124.4269', '1987-06-27', 'FI_ACCOUNT',
8200, 0, 108, 100),
(111, 'Ismael', 'Sciarra', 'ISCIARRA', '515.124.4369', '1987-06-28',
'FI_ACCOUNT', 7700, 0, 108, 100),
(112, 'Jose Manuel', 'Urman', 'JMURMAN', '515.124.4469', '1987-06-29',
'FI_ACCOUNT', 7800, 0, 108, 100),
(113, 'Luis', 'Popp', 'LPOPP', '515.124.4567', '1987-06-30', 'FI_ACCOUNT',
6900, 0, 108, 100),
(114, 'Den', 'Raphaely', 'DRAPHEAL', '515.127.4561', '1987-07-01', 'PU_MAN',
11000, 0, 100, 30),
(115, 'Alexander', 'Khoo', 'AKHOO', '515.127.4562', '1987-07-02',
'PU_CLERK', 3100, 0, 114, 30),
(116, 'Shelli', 'Baida', 'SBAIDA', '515.127.4563', '1987-07-03', 'PU_CLERK',
2900, 0, 114, 30),
(117, 'Sigal', 'Tobias', 'STOBIAS', '515.127.4564', '1987-07-04',
'PU_CLERK', 2800, 0, 114, 30),
(118, 'Guy', 'Himuro', 'GHIMURO', '515.127.4565', '1987-07-05', 'PU_CLERK',
2600, 0, 114, 30),
(119, 'Karen', 'Colmenares', 'KCOLMENA', '515.127.4566', '1987-07-06',
'PU_CLERK', 2500, 0, 114, 30),
(120, 'Matthew', 'Weiss', 'MWEISS', '650.123.1234', '1987-07-07', 'ST_MAN',
8000, 0, 100, 50),
(121, 'Adam', 'Fripp', 'AFRIPP', '650.123.2234', '1987-07-08', 'ST_MAN',
8200, 0, 100, 50),
(122, 'Payam', 'Kaufling', 'PKAUFLIN', '650.123.3234', '1987-07-09',
'ST_MAN', 7900, 0, 100, 50),
(123, 'Shanta', 'Vollman', 'SVOLLMAN', '650.123.4234', '1987-07-10',
'ST_MAN', 6500, 0, 100, 50),
(124, 'Kevin', 'Mourgos', 'KMOURGOS', '650.123.5234', '1987-07-11',
'ST_MAN', 5800, 0, 100, 50),
(125, 'Julia', 'Nayer', 'JNAYER', '650.124.1214', '1987-07-12', 'ST_CLERK',
3200, 0, 120, 50);
```

- **SQL Queries:**

Q1: find the name (first_name, last_name) and the salary of the employees who have a higher salary than the employee whose last_name = 'Bull'.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM Employee
WHERE SALARY > (SELECT SALARY FROM Employee WHERE LAST_NAME = 'Bull');
```

Q2: find the name (first_name, last_name) of all employees who work in the IT department.

```
SELECT FIRST_NAME, LAST_NAME
FROM Employee
WHERE DEPARTMENT_ID = (SELECT DEPARTMENT_ID FROM Departments WHERE
DEPARTMENT_NAME = 'IT');
```

Q3: find the name (first_name, last_name) of the employees who have a manager and worked in a USA based department.

```
SELECT DISTINCT FIRST_NAME, LAST_NAME
FROM Employee
WHERE MANAGER_ID IS NOT NULL AND DEPARTMENT_ID IN
(SELECT DEPARTMENT_ID FROM Departments WHERE LOCATION_ID IN
(SELECT LOCATION_ID FROM Locations WHERE COUNTRY_ID = 'US'));
```

Q4: find those employees who earn more than the average salary. Return employee ID, first name, last name.

```
SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME
FROM Employee
WHERE SALARY > (SELECT AVG(SALARY) FROM Employee);
```

Q5: find those employees whose department is located at 'Toronto'. Return first name, last name, employee ID, job ID.

```
SELECT e.FIRST_NAME, e.LAST_NAME, e.EMPLOYEE_ID, e.JOB_ID
FROM Employee e
JOIN Departments d ON e.DEPARTMENT_ID = d.DEPARTMENT_ID
JOIN Locations l ON d.LOCATION_ID = l.LOCATION_ID
WHERE l.CITY = 'Toronto';
```

Q6: find those employees who report to that manager whose first name is 'Payam'. Return first name, last name, employee ID and salary.

```
SELECT FIRST_NAME, LAST_NAME, EMPLOYEE_ID, SALARY
FROM Employee
WHERE MANAGER_ID IN
(SELECT EMPLOYEE_ID FROM Employee WHERE FIRST_NAME = 'Payam');
```

Q7: find all those departments where at least one employee is employed. Return department name.

```
SELECT DEPARTMENT_NAME
FROM Departments
```

```
WHERE DEPARTMENT_ID IN
(SELECT DEPARTMENT_ID FROM Employee);
```

Q8: find those employees who do not work in the departments where managers' IDs are between 100 and 200 (Begin and end values are included.). Return all the fields of the employees.

```
SELECT *
FROM Employee
WHERE DEPARTMENT_ID NOT IN (
    SELECT DEPARTMENT_ID
    FROM Departments
    WHERE MANAGER_ID BETWEEN 100 AND 200
);
```

Q9: From the following table, find those employees whose salary matches the lowest salary of any of the departments. Return first name, last name and department ID.

```
SELECT FIRST_NAME, LAST_NAME, DEPARTMENT_ID
FROM Employee
WHERE SALARY = (SELECT MIN(SALARY) FROM Employee);
```

Q10: find the name (first_name, last_name) of the employees who are managers.

```
SELECT FIRST_NAME, LAST_NAME
FROM Employee
WHERE EMPLOYEE_ID IN (SELECT MANAGER_ID FROM Departments);
```

Q11: find those employees whose salary is lower than that of employees whose job title is 'MK_MAN'. Exclude employees of Job title 'MK_MAN'. Return employee ID, first name, last name, job ID.

```
SELECT EMPLOYEE_ID, FIRST_NAME, LAST_NAME, JOB_ID
FROM Employee
WHERE SALARY < (
    SELECT SALARY
    FROM Employee
    WHERE JOB_ID = 'MK_MAN'
)
AND JOB_ID != 'MK_MAN';
```

Q12: Find the name (first_name, last_name), and salary of the employees whose salary is greater than the average salary.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM Employee
```

```
WHERE SALARY > (
    SELECT AVG(SALARY)
    FROM Employee
);
```

Q13: Find the name (first_name, last_name), and salary of the employees whose salary is equal to the minimum salary for their job grade.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM Employee
WHERE SALARY = (
    SELECT MIN_SALARY
    FROM Jobs
    WHERE JOB_ID = Employee.JOB_ID
);
```

Q14: Find the name (first_name, last_name), and salary of the employees who earns more than the average salary and works in any of the IT departments.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM Employee
WHERE SALARY > (
    SELECT AVG(SALARY)
    FROM Employee
)
AND DEPARTMENT_ID IN (
    SELECT DEPARTMENT_ID
    FROM Departments
    WHERE DEPARTMENT_NAME = 'IT'
);
```

Q15: Find the name (first_name, last_name), and salary of the employees who earns more than the earning of Mr. Bell.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM Employee
WHERE SALARY > (
    SELECT SALARY
    FROM Employee
    WHERE LAST_NAME = 'Bell'
);
```

Q16: .Find the name (first_name, last_name), and salary of the employees who earn the same salary as the minimum salary for all departments.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
```

```
FROM Employee
WHERE SALARY = (
    SELECT MIN(SALARY)
    FROM Employee
);
```

Q17: Find the name (first_name, last_name), and salary of the employees whose salary is greater than the average salary of all departments.

```
SELECT FIRST_NAME, LAST_NAME, SALARY
FROM Employee
WHERE SALARY > (
    SELECT AVG(SALARY)
    FROM Employee
);
```

Q18: Find the 3rd maximum salary in the employees table.

```
SELECT DISTINCT SALARY
FROM Employee
ORDER BY SALARY DESC
OFFSET 2 ROWS
FETCH NEXT 1 ROWS ONLY;
```

- ## Using Relational Algebra:

Q1: find the name (first_name, last_name) and the salary of the employees who have a higher salary than the employee whose last_name = 'Bull'.

$$\pi_{first\_name,last\_name,salary}(Employee) \bowtie_{Employee.salary>Bull.salary} (Employee \bowtie_{Employee.last\_name='Bull'} \sigma_{salary}(Employee))$$

Q2: find the name (first_name, last_name) of all employees who work in the IT department.

$$\pi_{first\_name,last\_name}(Employee \bowtie_{Employee.department\_id=IT.department\_id} \sigma_{department\_name='IT'}(Departments \bowtie Locations))$$

Q3: find the name (first_name, last_name) of the employees who have a manager and worked in a USA based department.

$$\pi_{first\_name,last\_name}(Employee \bowtie_{Employee.manager\_id=manager.employee\_id \cap location.country\_id='US'} (\sigma_{country\_id='US'}(Locations) \bowtie Departments))$$

Q4: find those employees who earn more than the average salary. Return employee ID, first name, last name.

$$\pi_{employee\_id,first\_name,last\_name}(\sigma_{salary>avg\_salary}(Employee \bowtie_{Employee.salary>avg\_salary} (\rho_{avg\_salary}(avg(salary)))))$$

Q5: find those employees whose department is located at 'Toronto'. Return first name, last name, employee ID, job ID.

$$\pi_{first\_name,last\_name,employee\_id,job\_id}(Employee \bowtie_{Employee.department\_id=Departments.department\_id \cap Departments.location\_id=Toronto.location\_id} (\sigma_{city='Toronto'}(Locations \bowtie Toronto)))$$

Q6: find those employees who report to that manager whose first name is 'Payam'. Return first name, last name, employee ID and salary.

$$\pi_{first\_name,last\_name,employee\_id,salary}(Employee \bowtie_{Employee.manager\_id=Payam.employee\_id} (\sigma_{first\_name='Payam'}(Employee)))$$

Q7: find all those departments where at least one employee is employed. Return department name.

$$\pi_{department\_name}(Departments \bowtie_{Departments.department\_id=Employee.department\_id} Employee)$$

Q8: find those employees who do not work in the departments where managers' IDs are between 100 and 200 (Begin and end values are included.). Return all the fields of the employees.

$$\pi_*(Employee) - \pi_*(Employee \bowtie_{Employee.department\_id=Departments.department\_id \cap manager\_id BETWEEN 100 AND 200} Departments)$$

It can also be written as:

$$\text{Employee} - \text{DEPARTMENT\_ID NOT IN}(\pi_{\text{DEPARTMENT\_ID}}(\sigma_{\text{MANAGER\_ID} \geq 100 \wedge \text{MANAGER\_ID} \leq 200}(\text{Departments})))$$

Q9: From the following table, find those employees whose salary matches the lowest salary of any of the departments. Return first name, last name and department ID.

$$\pi_{\text{FIRST\_NAME, LAST\_NAME, DEPARTMENT\_ID}}(\sigma_{\text{SALARY=MIN(SALARY)}}(\text{Employee}))$$

Q10: find the name (first_name, last_name) of the employees who are managers.

$$\pi_{first\_name,last\_name}(Employee \bowtie_{Employee.employee\_id=Departments.manager\_id} Departments)$$

Q11: find those employees whose salary is lower than that of employees whose job title is 'MK_MAN'. Exclude employees of Job title 'MK_MAN'. Return employee ID, first name, last name, job ID.

$$\pi_{employee\_id,first\_name,last\_name,job\_id}(Employee \bowtie_{Employee.salary<MK\_MAN.salary} (\rho_{MK\_MAN}(\sigma_{job\_id='MK\_MAN'}(Employee))))$$

Q12: Find the name (first_name, last_name), and salary of the employees whose salary is greater than the average salary.

$$\pi_{\text{FIRST\_NAME, LAST\_NAME, SALARY}}(\sigma_{\text{SALARY>AVG(SALARY)}}(\text{Employee}))$$

Q13: Find the name (first_name, last_name), and salary of the employees whose salary is equal to the minimum salary for their job grade.

$$\pi_{first\_name,last\_name,salary}(Employee \bowtie_{Employee.salary=min\_salary} (Jobs))$$

Q14: Find the name (first_name, last_name), and salary of the employees who earns more than the average salary and works in any of the IT departments.

$$\pi_{\text{FIRST\_NAME,LAST\_NAME,SALARY}} (\sigma_{\text{SALARY}>\text{AVG\_SALARY}}(Employee) \bowtie$$

$$\text{Employee.DEPARTMENT\_ID} = \text{Departments.DEPARTMENT\_ID AND Departments.DEPARTMENT\_NAME}='IT'$$

$$Departments)$$

Q15: Find the name (first_name, last_name), and salary of the employees who earns more than the earning of Mr. Bell.

$$\pi_{\text{FIRST\_NAME, LAST\_NAME, SALARY}}(\sigma_{\text{SALARY}>(\pi_{\text{SALARY}}(\sigma_{\text{LAST\_NAME='Bell'}}(\text{Employee})))}(\text{Employee}))$$

Q16: .Find the name (first_name, last_name), and salary of the employees who earn the same salary as the minimum salary for all departments.

$$\pi_{\text{FIRST\_NAME, LAST\_NAME, SALARY}}(\sigma_{\text{SALARY}=\text{MIN(SALARY)}}(\text{Employee}))$$

Q17: Find the name (first_name, last_name), and salary of the employees whose salary is greater than the average salary of all departments.

$$\pi_{\text{FIRST\_NAME, LAST\_NAME, SALARY}}(\sigma_{\text{SALARY}>\text{AVG(SALARY)}}(\text{Employee}))$$

Q18: Find the 3rd maximum salary in the employees table.

$$\pi_{Salary} (\sigma_{3=COUNT(DISTINCTSalary)} (\rho_{e1} (Employee)))$$

Note: The Relational Algebra equations written in more than 1 different lines are the same query. But, it was a long query. So, I divided it into multiple lines. Moreover, please check the `Workforce_Management_System.sql` file which is submitted on Google Classroom and contains all SQL Queries for Question 02.
