ASSESMENT -1 SQL

NAME – RIYA BANSAL

REGISTRATION NO. – 21BCE0024

SQL Script to Seed Sample Data.

```
CREATE DATABASE ORG1;
SHOW DATABASES;
USE ORG1;
CREATE TABLE Worker (
WORKER_ID INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
FIRST_NAME CHAR(25),
LAST_NAME CHAR(25),
SALARY INT(15),
JOINING_DATE DATETIME,
DEPARTMENT CHAR(25)
);
INSERT INTO Worker
(WORKER_ID, FIRST_NAME, LAST_NAME, SALARY, JOINING_DATE,
DEPARTMENT) VALUES
(001, 'Monika', 'Arora', 100000, '14-02-20
09.00.00', 'HR'),
(002, 'Niharika', 'Verma', 80000, '14-06-11
09.00.00', 'Admin'),
(003, 'Vishal', 'Singhal', 300000, '14-02-20
09.00.00', 'HR'),
(004, 'Amitabh', 'Singh', 500000, '14-02-20
09.00.00', 'Admin'),
(005, 'Vivek', 'Bhati', 500000, '14-06-11
09.00.00', 'Admin'),
(006, 'Vipul', 'Diwan', 200000, '14-06-11
09.00.00', 'Account'),
(007, 'Satish', 'Kumar', 75000, '14-01-20
09.00.00', 'Account'),
(008, 'Geetika', 'Chauhan', 90000, '14-04-11
09.00.00', 'Admin');
CREATE TABLE Bonus (
WORKER_REF_ID INT,
BONUS_AMOUNT INT(10),
BONUS_DATE DATETIME,
FOREIGN KEY (WORKER_REF_ID)
REFERENCES Worker(WORKER_ID)
ON DELETE CASCADE
);
INSERT INTO Bonus
(WORKER_REF_ID, BONUS_AMOUNT, BONUS_DATE) VALUES
(001, 5000, '16-02-20'),
(002, 3000, '16-06-11'),
(003, 4000, '16-02-20'),
(001, 4500, '16-02-20'),
(002, 3500, '16-06-11');
CREATE TABLE Title (
WORKER_REF_ID INT,
WORKER_TITLE CHAR(25),
AFFECTED_FROM DATETIME,
FOREIGN KEY (WORKER_REF_ID)
REFERENCES Worker(WORKER_ID)
ON DELETE CASCADE
);
INSERT INTO Title
(WORKER_REF_ID, WORKER_TITLE, AFFECTED_FROM) VALUES
(001, 'Manager', '2016-02-20 00:00:00'),
(002, 'Executive', '2016-06-11 00:00:00'),
(008, 'Executive', '2016-06-11 00:00:00'),
(005, 'Manager', '2016-06-11 00:00:00'),
(004, 'Asst. Manager', '2016-06-11 00:00:00'),
(007, 'Executive', '2016-06-11 00:00:00'),
(006, 'Lead', '2016-06-11 00:00:00'),
(003, 'Lead', '2016-06-11 00:00:00');
```

QUESTIONS:-

Q-1. Write an SQL query to fetch "FIRST_NAME" from Worker table using the alias name as <WORKER NAME>.

```
SELECT FIRST_NAME AS 'WORKER NAME'

FROM Worker;
```

Q-2. Write an SQL query to fetch "FIRST_NAME" from Worker table in upper case.

```
SELECT UPPER(FIRST_NAME)

FROM Worker;
```

Q-3. Write an SQL query to fetch unique values of DEPARTMENT from Worker table.

```
SELECT DISTINCT DEPARTMENT

FROM Worker;
```

Q-4. Write an SQL query to print the first three characters of FIRST NAME from Worker table.

```
SELECT LEFT(FIRST_NAME, 3)

FROM Worker;
```

Q-5. Write an SQL query to find the position of the alphabet ('a') in the first name column 'Amitabh' from Worker table.

```
SELECT INSTR(FIRST_NAME, 'a')

FROM Worker

WHERE FIRST_NAME = 'Amitabh';
```

Q-6. Write an SQL query to print the FIRST_NAME from Worker table after removing white spaces from the right side.

```
SELECT RTRIM(FIRST_NAME)

FROM Worker;
```

Q-7. Write an SQL query to print the DEPARTMENT from Worker table after removing white spaces from the left side.

```sql
SELECT LTRIM(DEPARTMENT)

FROM Worker;
```

Q-8. Write an SQL query that fetches the unique values of DEPARTMENT from Worker table and prints its length.

```sql
SELECT DISTINCT DEPARTMENT, LENGTH(DEPARTMENT) AS DeptLength

FROM Worker;
```

Q-9. Write an SQL query to print the FIRST_NAME from Worker table after replacing 'a' with 'A'.

```sql
SELECT REPLACE(FIRST_NAME, 'a', 'A')

FROM Worker;
```

Q-10. Write an SQL query to print the FIRST_NAME and LAST_NAME from Worker table into a single column COMPLETE_NAME. A space char should separate them.

```sql
SELECT CONCAT(FIRST_NAME, ' ', LAST_NAME) AS COMPLETE_NAME

FROM Worker;
```

Q-11. Write an SQL query to print all Worker details from the Worker table order by FIRST_NAME Ascending.

```sql
SELECT *

FROM Worker

ORDER BY FIRST_NAME ASC;
```

Q-12. Write an SQL query to print all Worker details from the Worker table order by FIRST_NAME Ascending and DEPARTMENT Descending.

```sql
SELECT *

FROM Worker

ORDER BY FIRST_NAME ASC, DEPARTMENT DESC;
```

Q-13. Write an SQL query to print details for Workers with the first name as "Vipul" and "Satish" from Worker table.

```
SELECT *

FROM Worker

WHERE FIRST_NAME IN ('Vipul', 'Satish');
```

Q-14. Write an SQL query to print details of workers excluding first names, "Vipul" and "Satish" from Worker table.

```
SELECT *

FROM Worker

WHERE FIRST_NAME NOT IN ('Vipul', 'Satish');
```

Q-15. Write an SQL query to print details of Workers with DEPARTMENT name as "Admin".

```
SELECT *

FROM Worker

WHERE DEPARTMENT = 'Admin';
```

Q-16. Write an SQL query to print details of the Workers whose FIRST_NAME contains 'a'.

```
SELECT *

FROM Worker

WHERE FIRST_NAME LIKE '%a%';
```

Q-17. Write an SQL query to print details of the Workers whose FIRST_NAME ends with 'a'.

```
SELECT *
```

```
FROM Worker
WHERE FIRST_NAME LIKE '%a';
```

Q-18. Write an SQL query to print details of the Workers whose FIRST_NAME ends with 'h' and contains six alphabets.

```
SELECT *
FROM Worker
WHERE FIRST_NAME LIKE '_____h';
```

Q-19. Write an SQL query to print details of the Workers whose SALARY lies between 100000 and 500000.

```
SELECT *
FROM Worker
WHERE SALARY BETWEEN 100000 AND 500000;
```

Q-20. Write an SQL query to print details of the Workers who have joined in Feb'2014.

```
SELECT *
FROM Worker
WHERE JOINING_DATE BETWEEN '2014-02-01' AND '2014-02-28 23:59:59';
```

Q-21. Write an SQL query to fetch the count of employees working in the department 'Admin'.

```
SELECT COUNT(*) AS AdminCount
FROM Worker
WHERE DEPARTMENT = 'Admin';
```

Q-22. Write an SQL query to fetch worker names with salaries >= 50000 and <= 100000.

```
SELECT FIRST_NAME, LAST_NAME

FROM Worker

WHERE SALARY BETWEEN 50000 AND 100000;
```

Q-23. Write an SQL query to fetch the no. of workers for each department in the descending order.

```
SELECT DEPARTMENT, COUNT(*) AS NumberOfWorkers

FROM Worker

GROUP BY DEPARTMENT

ORDER BY NumberOfWorkers DESC;
```

Q-24. Write an SQL query to print details of the Workers who are also Managers.

```
SELECT w.*

FROM Worker w

JOIN Title t ON w.WORKER_ID = t.WORKER_REF_ID

WHERE t.WORKER_TITLE = 'Manager';
```

Q-25. Write an SQL query to fetch duplicate records having matching data in some fields of a table.

```
SELECT FIRST_NAME, LAST_NAME, COUNT(*)

FROM Worker

GROUP BY FIRST_NAME, LAST_NAME

HAVING COUNT(*) > 1;
```

Q-26. Write an SQL query to show only odd rows from a table.

```
SELECT *
FROM Worker
WHERE MOD(WORKER_ID, 2) = 1;
```

Q-27. Show only even rows from a table:

```
SET @rownum := 0;
SELECT * FROM (SELECT *, @rownum := @rownum + 1 AS
row_num FROM Worker) AS temp WHERE MOD(row_num, 2)
= 0;
```

Q-28. Clone a new table from another table:

```
CREATE TABLE NewWorker AS SELECT * FROM Worker;
```

Q-29. Fetch intersecting records of two tables:

```
SELECT * FROM Worker INTERSECT SELECT * FROM
AnotherTable;
-- Note: INTERSECT is supported by some SQL databases
like PostgreSQL, SQL Server, but not directly in MySQL.
-- For MySQL:
SELECT * FROM Worker w JOIN AnotherTable a ON
w.WORKER_ID = a.WORKER_ID;
```

Q-30. Show records from one table that another table does not have:

```
SELECT * FROM Worker w WHERE NOT EXISTS (SELECT 1
FROM AnotherTable a WHERE w.WORKER_ID =
a.WORKER_ID);
```

Q-31. Show the current date and time:

```
SELECT NOW();
```

Q-32. Show the top 10 records of a table:

```
SELECT * FROM Worker ORDER BY WORKER_ID LIMIT 10;
```

Q-33. Determine the 5th highest salary from a table:

```
SELECT DISTINCT SALARY FROM Worker ORDER BY SALARY
DESC LIMIT 1 OFFSET 4;
```

Q-34. Determine the 5th highest salary without using TOP or LIMIT method:

```
SELECT SALARY FROM Worker w1 WHERE 4 = (SELECT
COUNT(DISTINCT SALARY) FROM Worker w2 WHERE
w2.SALARY > w1.SALARY);
```

Q-35. Fetch the list of employees with the same salary:

```
SELECT SALARY, GROUP_CONCAT(FIRST_NAME, ' ',
LAST_NAME) AS EMPLOYEES FROM Worker GROUP BY
SALARY HAVING COUNT(*) > 1;
```

Q-36. Show the second highest salary from a table:

```
SELECT MAX(SALARY) FROM Worker WHERE SALARY <
(SELECT MAX(SALARY) FROM Worker);
```

Q-37. Show one row twice in results from a table:

```
SELECT * FROM Worker UNION ALL SELECT * FROM
Worker WHERE WORKER_ID = 1 LIMIT 1;
```

Q-38. Fetch intersecting records of two tables:

```
SELECT * FROM Worker w JOIN AnotherTable a ON
w.WORKER_ID = a.WORKER_ID;
```

Q-39. Fetch the first 50% records from a table:

```
SELECT * FROM Worker ORDER BY WORKER_ID LIMIT
(SELECT COUNT(*)/2 FROM Worker);
```

Q-40. Fetch the departments that have less than five people in it:

```
SELECT DEPARTMENT FROM Worker GROUP BY
DEPARTMENT HAVING COUNT(*) < 5;
```

Q-41. Show all departments along with the number of people in them:

```
SELECT DEPARTMENT, COUNT(*) AS NumberOfPeople
FROM Worker GROUP BY DEPARTMENT;
```

Q-42. Show the last record from a table:

```
SELECT * FROM Worker ORDER BY WORKER_ID DESC LIMIT
1;
```

Q-43. Fetch the first row of a table:

```
SELECT * FROM Worker ORDER BY WORKER_ID LIMIT 1;
```

Q-44. Fetch the last five records from a table:

```
SELECT * FROM Worker ORDER BY WORKER_ID DESC LIMIT
5;
```

Q-45. Print the name of employees having the highest salary in each department:

```
SELECT DEPARTMENT, FIRST_NAME, LAST_NAME, SALARY
FROM Worker w WHERE SALARY = (SELECT MAX(SALARY)
FROM Worker WHERE DEPARTMENT = w.DEPARTMENT);
```

Q-46. Fetch three max salaries from a table:

```
SELECT DISTINCT SALARY FROM Worker ORDER BY SALARY
DESC LIMIT 3;
```

Q-47. Fetch three min salaries from a table:

```
SELECT DISTINCT SALARY FROM Worker ORDER BY SALARY
LIMIT 3;
```

Q-48. Fetch nth max salaries from a table:

```
SELECT DISTINCT SALARY FROM Worker ORDER BY SALARY
DESC LIMIT 1 OFFSET (n-1); -- Replace n with the desired
rank, for example, OFFSET 2 for the 3rd highest salary
```

Q-49. Fetch departments along with the total salaries paid for each of them:

```
SELECT DEPARTMENT, SUM(SALARY) AS TotalSalaries
FROM Worker GROUP BY DEPARTMENT;
```

Q-50. Fetch the names of workers who earn the highest salary:

```
SELECT FIRST_NAME, LAST_NAME FROM Worker WHERE
SALARY = (SELECT MAX(SALARY) FRO
```