ASSIGNMENT

```
CREATE TABLE EMPLOYEE ( EMPLOYEE_ID NUMBER(6,0), NAME
VARCHAR2(20), SALARY NUMBER(8,2) );

INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(100,'Rishitha',4400);

 INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(100,'Rishitha',4400);

INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(101,'Sai',13000);

INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(101,'Sai',13000);

INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(101,'Sai',13000);

INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(102,'Kiran',6000);

INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(102,'Lakshman',6000);

 INSERT INTO EMPLOYEE(EMPLOYEE_ID,NAME,SALARY)
VALUES(103,'Priyanka',11000); SELECT * FROM EMPLOYEE;
```

1. Query to select UNIQUE records from the given table, write 2 queries with 2 approaches
2. Query to delete DUPLICATE records from the given table, write 2 queries with 2 approaches

3.Query to select TOP 5 records from the given table

4.Query to select LAST 5 records from the given table

5. Query to select second MAX Salary write 2 queries with 2 approaches (1 approach includes MAX function, the other approach shouldn't include MAX function)

6. Query to find third MAX Salaried employee without using Analytic Functions i.e., not using MAX function

7.Update Salary of 'Rishitha' employee by 10% (using sub query and direct UPDATE statement)

ANSWERS

```
CREATE TABLE EMPLOYEE ( EMPLOYEE_ID INT, NAME VARCHAR(20), SALARY DECIMAL(8, 2) );

 INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (100, 'Rishitha', 4400);

INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (100, 'Rishitha', 4400);

 INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);

INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);

INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);

INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (102, 'Kiran', 6000);

INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (102, 'Lakshman', 6000);

 INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (103, 'Priyanka', 11000);

SELECT * FROM EMPLOYEE;
```

1.Query to select UNIQUE records from the given table

```
-- Approach 1: Using DISTINCT
SELECT DISTINCT EMPLOYEE_ID, NAME, SALARY FROM EMPLOYEE;
-- Approach 2: Using GROUP BY
SELECT EMPLOYEE_ID, NAME, SALARY FROM EMPLOYEE GROUP BY EMPLOYEE_ID, NAME,
SALARY;
```

2. Query to delete DUPLICATE records from the given table.

```
-- Approach 1: Using a self-join with a subquery

DELETE e1

FROM EMPLOYEE e1

JOIN (SELECT MIN(EMPLOYEE_ID) AS min_id, NAME, SALARY

 FROM EMPLOYEE

 GROUP BY NAME, SALARY) e2
```

```
ON e1.EMPLOYEE_ID > e2.min_id AND e1.NAME = e2.NAME AND e1.SALARY = e2.SALARY;
 -- Approach 2: Using CTE and ROW_NUMBER()
WITH CTE AS (
SELECT EMPLOYEE_ID, NAME, SALARY,
ROW_NUMBER() OVER(PARTITION BY NAME, SALARY ORDER BY EMPLOYEE_ID) AS rn
FROM EMPLOYEE
 )
 DELETE FROM EMPLOYEE
WHERE EMPLOYEE_ID IN (
SELECT EMPLOYEE_ID
 FROM CTE
 WHERE rn > 1
);
```

3. Query to select TOP 5 records from the given table

```
SELECT * FROM EMPLOYEE
ORDER BY EMPLOYEE_ID
LIMIT 5;
-- 4. Query to select LAST 5 records from the given table
SELECT * FROM EMPLOYEE
ORDER BY EMPLOYEE_ID DESC
LIMIT 5;

-- 5. Query to select second MAX Salary
-- Approach 1: Using MAX function
SELECT MAX(SALARY) AS SECOND_MAX_SALARY
```

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


-- Approach 2: Without using MAX function
SELECT SALARY AS SECOND_MAX_SALARY
FROM (
    SELECT SALARY
    FROM EMPLOYEE
    ORDER BY SALARY DESC
    LIMIT 1 OFFSET 1
) AS T;
-- 6. Query to find third MAX Salaried employee without using Analytic Functions
SELECT SALARY
FROM (
    SELECT SALARY
    FROM EMPLOYEE
    ORDER BY SALARY DESC
    LIMIT 1 OFFSET 2
) AS T;
-- 7. Update Salary of 'Rishitha' employee by 10%
-- Approach 1: Using a temporary table
CREATE TEMPORARY TABLE TempEmployee AS
SELECT EMPLOYEE_ID
```

```
FROM EMPLOYEE
WHERE NAME = 'Rishitha';
UPDATE EMPLOYEE
SET SALARY = SALARY * 1.1
WHERE EMPLOYEE_ID IN (SELECT EMPLOYEE_ID FROM TempEmployee);
DROP TEMPORARY TABLE TempEmployee;

-- Approach 2: Direct UPDATE statement
UPDATE EMPLOYEE
SET SALARY = SALARY * 1.1
WHERE NAME = 'Rishitha';
```

In cmd

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sai1               |
| sys                |
| test1              |
| utube              |
+--------------------+
7 rows in set (0.00 sec)

mysql> use utube;
Database changed
mysql> show tables;
+-----------------+
| Tables_in_utube |
+-----------------+
| employee        |
+-----------------+
1 row in set (0.00 sec)

mysql> drop employee;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'employee' at line 1
mysql> drop table employee;
```

```
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE EMPLOYEE ( EMPLOYEE_ID INT, NAME VARCHAR(20), SALARY DECIMAL(8, 2) );
Query OK, 0 rows affected (0.02 sec)

mysql>  INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (100, 'Rishitha', 4400);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (100, 'Rishitha', 4400);
Query OK, 1 row affected (0.00 sec)

mysql>  INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (102, 'Kiran', 6000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (102, 'Lakshman', 6000);
Query OK, 1 row affected (0.00 sec)

mysql>  INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (103, 'Priyanka', 11000);
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM EMPLOYEE;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
8 rows in set (0.00 sec)

mysql> SELECT DISTINCT EMPLOYEE_ID, NAME, SALARY FROM EMPLOYEE;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql> SELECT EMPLOYEE_ID, NAME, SALARY FROM EMPLOYEE GROUP BY EMPLOYEE_ID, NAME,
SALARY;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql> SELECT * FROM EMPLOYEE
    -> ORDER BY EMPLOYEE_ID
    -> LIMIT 5;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql>
mysql> -- 4. Query to select LAST 5 records from the given table
mysql> SELECT * FROM EMPLOYEE
    -> ORDER BY EMPLOYEE_ID DESC
    -> LIMIT 5;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         103 | Priyanka | 11000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql> -- 5. Query to select second MAX Salary
mysql> -- Approach 1: Using MAX function
mysql> SELECT MAX(SALARY) AS SECOND_MAX_SALARY
    -> FROM EMPLOYEE
    -> WHERE SALARY < (SELECT MAX(SALARY) FROM EMPLOYEE);
+-------------------+
| SECOND_MAX_SALARY |
+-------------------+
|          11000.00 |
+-------------------+
1 row in set (0.00 sec)

mysql> ^C
mysql> -- Approach 2: Without using MAX function
mysql> SELECT SALARY AS SECOND_MAX_SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 1
    -> ) AS T;
+-------------------+
| SECOND_MAX_SALARY |
+-------------------+
|          13000.00 |
+-------------------+
1 row in set (0.00 sec)

mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)
```

```
mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)

mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)

mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    -> ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)
```

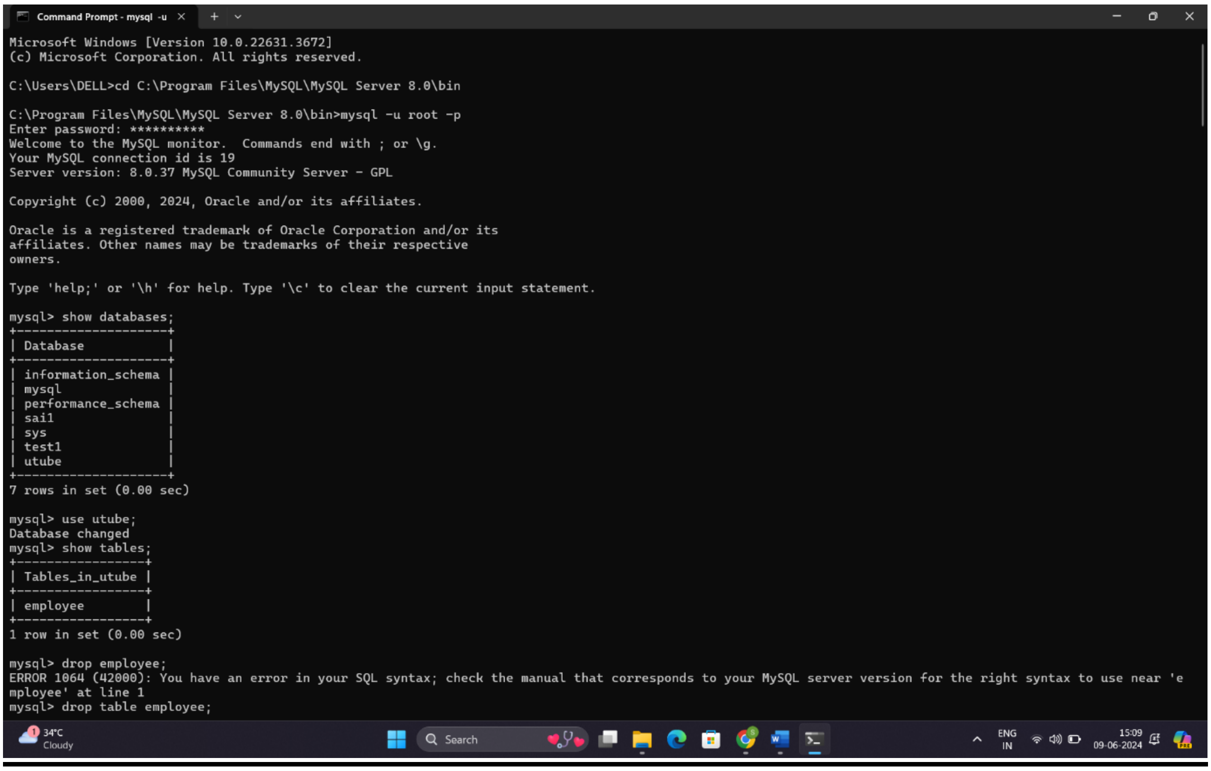
```
mysql> -- 7. Update Salary of 'Rishitha' employee by 10%
mysql> -- Approach 1: Using a temporary table
mysql> CREATE TEMPORARY TABLE TempEmployee AS
    -> SELECT EMPLOYEE_ID
    -> FROM EMPLOYEE
    -> WHERE NAME = 'Rishitha';
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from employee;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
8 rows in set (0.00 sec)

mysql> UPDATE EMPLOYEE
    -> SET SALARY = SALARY * 1.1
    -> WHERE EMPLOYEE_ID IN (SELECT EMPLOYEE_ID FROM TempEmployee);
Query OK, 2 rows affected (0.01 sec)
Rows matched: 2  Changed: 2  Warnings: 0
```

```
mysql> select * from tempemployee;

+-------------+

| EMPLOYEE_ID |

+-------------+

|         100 |

|         100 |

+-------------+

2 rows in set (0.00 sec)


mysql> DROP TEMPORARY TABLE TempEmployee;

Query OK, 0 rows affected (0.00 sec)


mysql> -- Approach 2: Direct UPDATE statement

mysql> UPDATE EMPLOYEE

    -> SET SALARY = SALARY * 1.1

    -> WHERE NAME = 'Rishitha';

Query OK, 2 rows affected (0.01 sec)

Rows matched: 2  Changed: 2  Warnings: 0


mysql> 2. Query to delete DUPLICATE records from the given table.

    -> -- Approach 1: Using a self-join with a subquery

    -> DELETE e1

    -> FROM EMPLOYEE e1

    -> JOIN (SELECT MIN(EMPLOYEE_ID) AS min_id, NAME, SALARY

    -> FROM EMPLOYEE

    ->  GROUP BY NAME, SALARY) e2

    -> ON e1.EMPLOYEE_ID > e2.min_id AND e1.NAME = e2.NAME AND e1.SALARY = e2.SALARY;

ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to
your MySQL server version for the right syntax to use near '2. Query to delete DUPLICATE records
from the given table.


DELETE e1
```

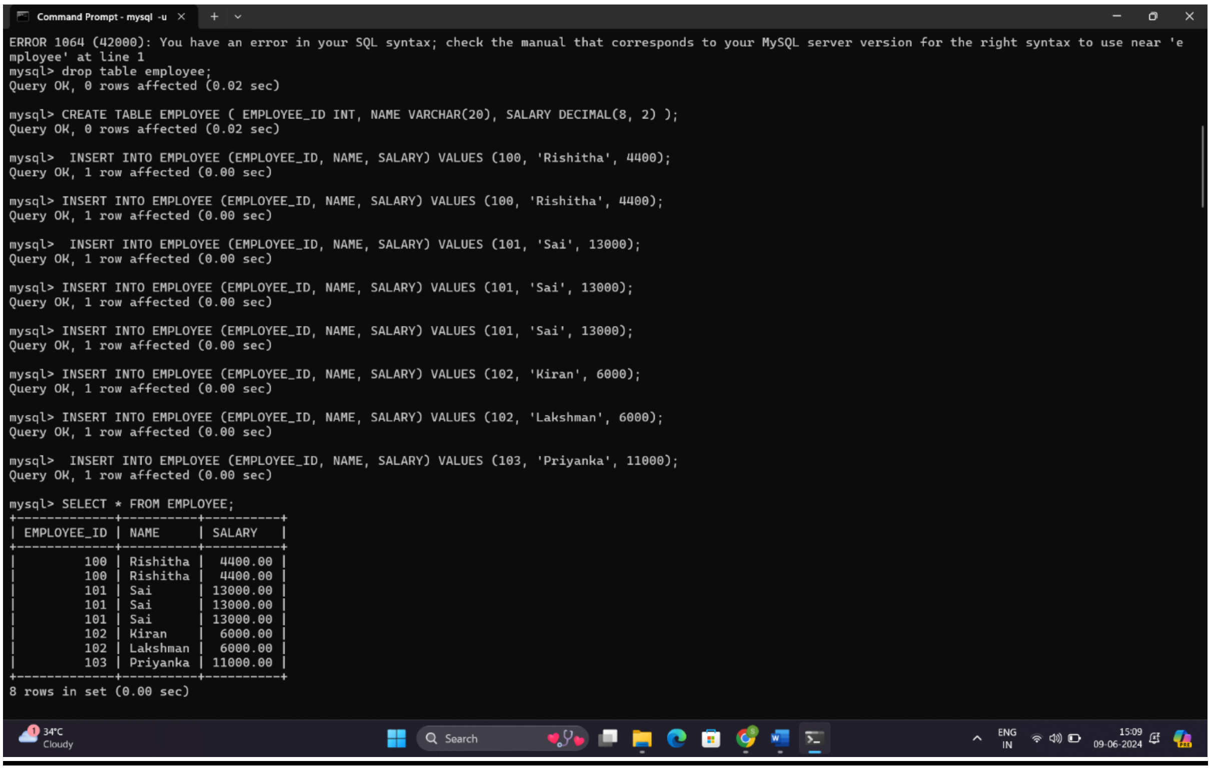
```
FROM EMPL' at line 1
mysql> DELETE e1
    -> FROM EMPLOYEE e1
    -> JOIN (SELECT MIN(EMPLOYEE_ID) AS min_id, NAME, SALARY
    ->  FROM EMPLOYEE
    ->  GROUP BY NAME, SALARY) e2
    -> ON e1.EMPLOYEE_ID > e2.min_id AND e1.NAME = e2.NAME AND e1.SALARY = e2.SALARY;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from employee;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  5324.00 |
|         100 | Rishitha |  5324.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
8 rows in set (0.00 sec)

mysql> -- Approach 2: Using CTE and ROW_NUMBER()
mysql> WITH CTE AS (
    -> SELECT EMPLOYEE_ID, NAME, SALARY,
    -> ROW_NUMBER() OVER(PARTITION BY NAME, SALARY ORDER BY EMPLOYEE_ID) AS rn
    -> FROM EMPLOYEE
    ->  )
    ->  DELETE FROM EMPLOYEE
```

```
   -> WHERE EMPLOYEE_ID IN (
   -> SELECT EMPLOYEE_ID
   ->  FROM CTE
   ->  WHERE rn > 1
   -> );
Query OK, 5 rows affected (0.00 sec)

mysql> select * from employee;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
3 rows in set (0.00 sec)

mysql>
```

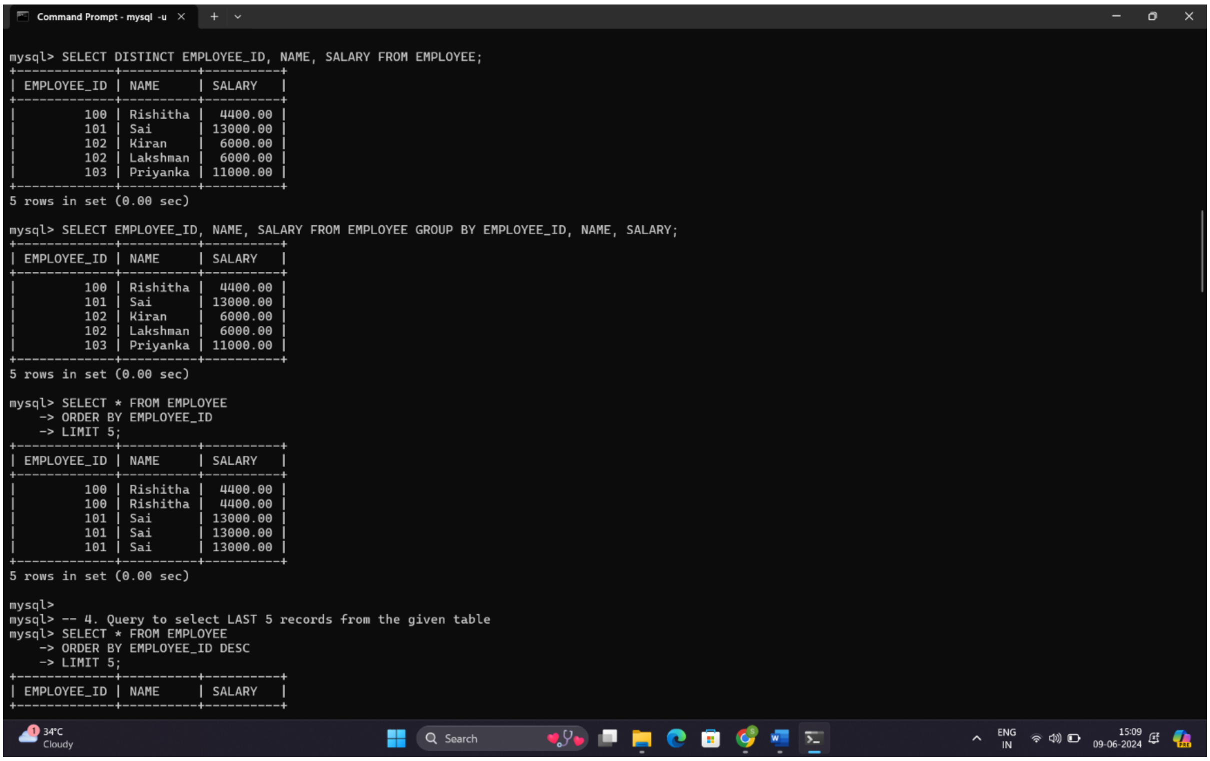
output Screens



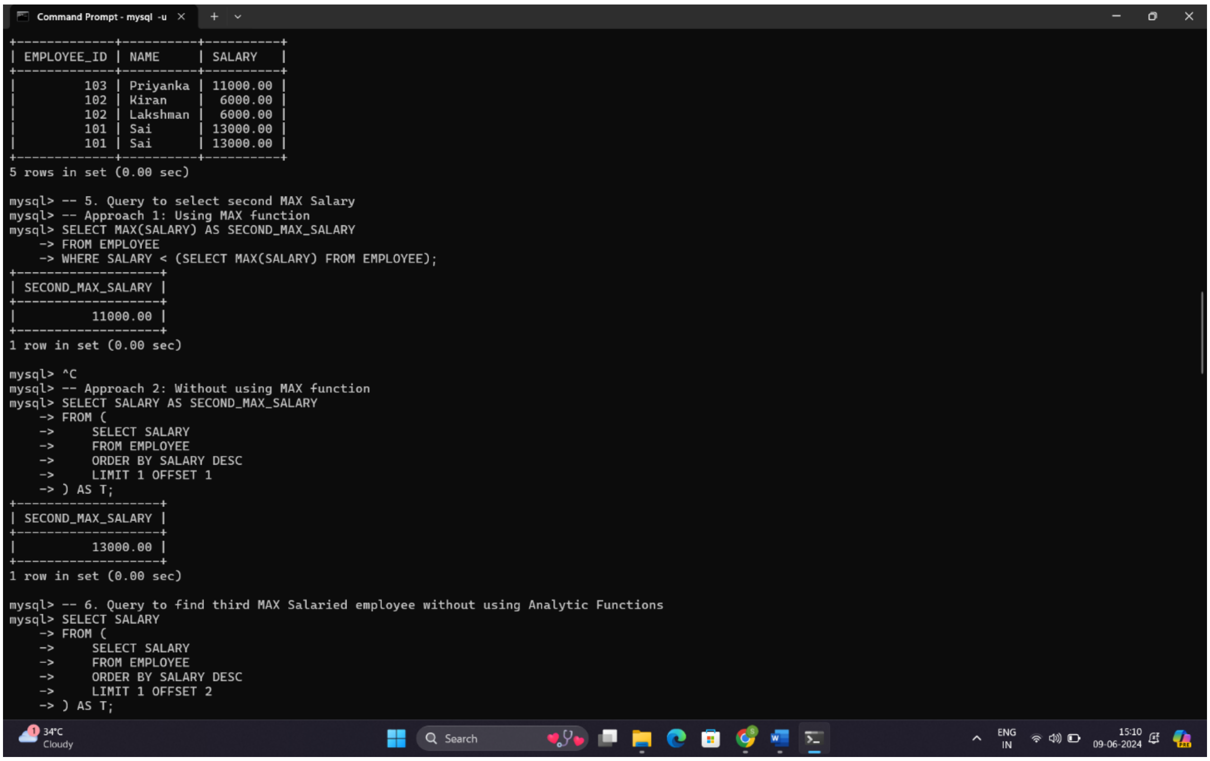
```
Microsoft Windows [Version 10.0.22631.3672]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DELL>cd C:\Program Files\MySQL\MySQL Server 8.0\bin

C:\Program Files\MySQL\MySQL Server 8.0\bin>mysql -u root -p
Enter password: **********
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 19
Server version: 8.0.37 MySQL Community Server - GPL

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sai1               |
| sys                |
| test1              |
| utube              |
+--------------------+
7 rows in set (0.00 sec)

mysql> use utube;
Database changed
mysql> show tables;
+-----------------+
| Tables_in_utube |
+-----------------+
| employee        |
+-----------------+
1 row in set (0.00 sec)

mysql> drop employee;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'e
mployee' at line 1
mysql> drop table employee;
```



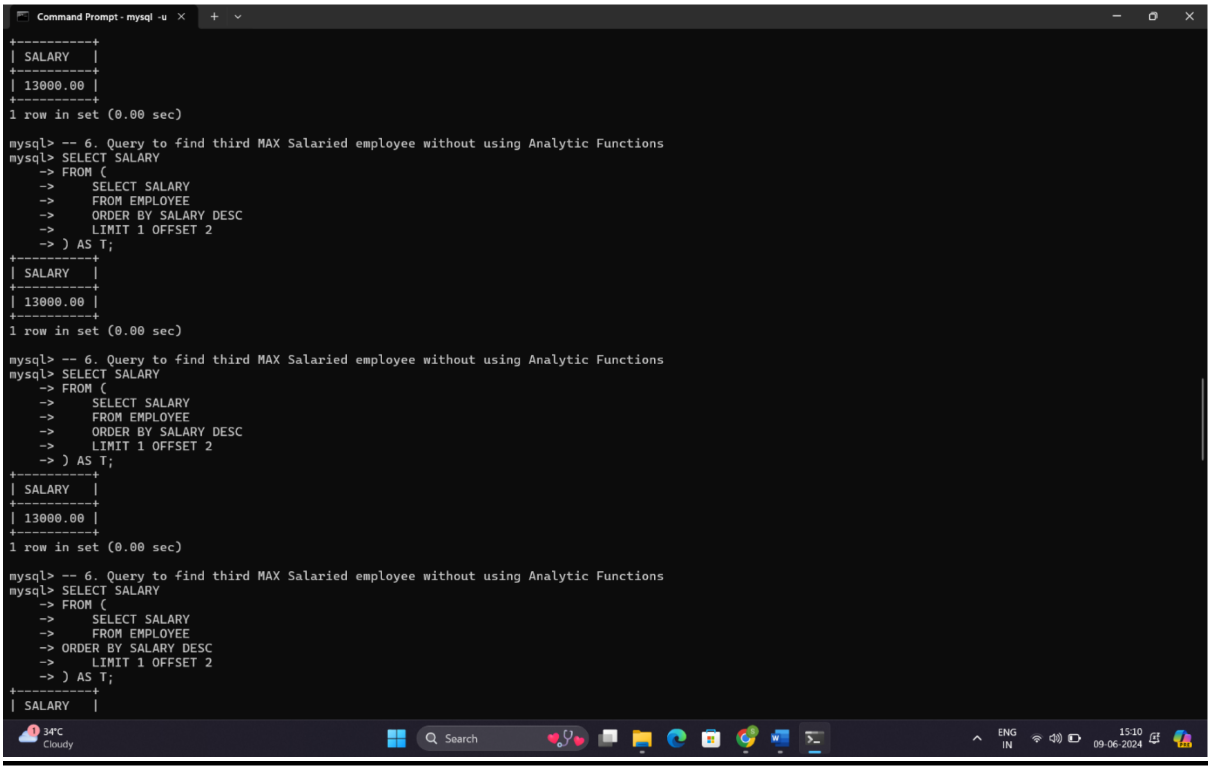
```
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'e
mployee' at line 1
mysql> drop table employee;
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE EMPLOYEE ( EMPLOYEE_ID INT, NAME VARCHAR(20), SALARY DECIMAL(8, 2) );
Query OK, 0 rows affected (0.02 sec)

mysql>  INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (100, 'Rishitha', 4400);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (100, 'Rishitha', 4400);
Query OK, 1 row affected (0.00 sec)

mysql>  INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (101, 'Sai', 13000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (102, 'Kiran', 6000);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (102, 'Lakshman', 6000);
Query OK, 1 row affected (0.00 sec)

mysql>  INSERT INTO EMPLOYEE (EMPLOYEE_ID, NAME, SALARY) VALUES (103, 'Priyanka', 11000);
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM EMPLOYEE;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
8 rows in set (0.00 sec)
```



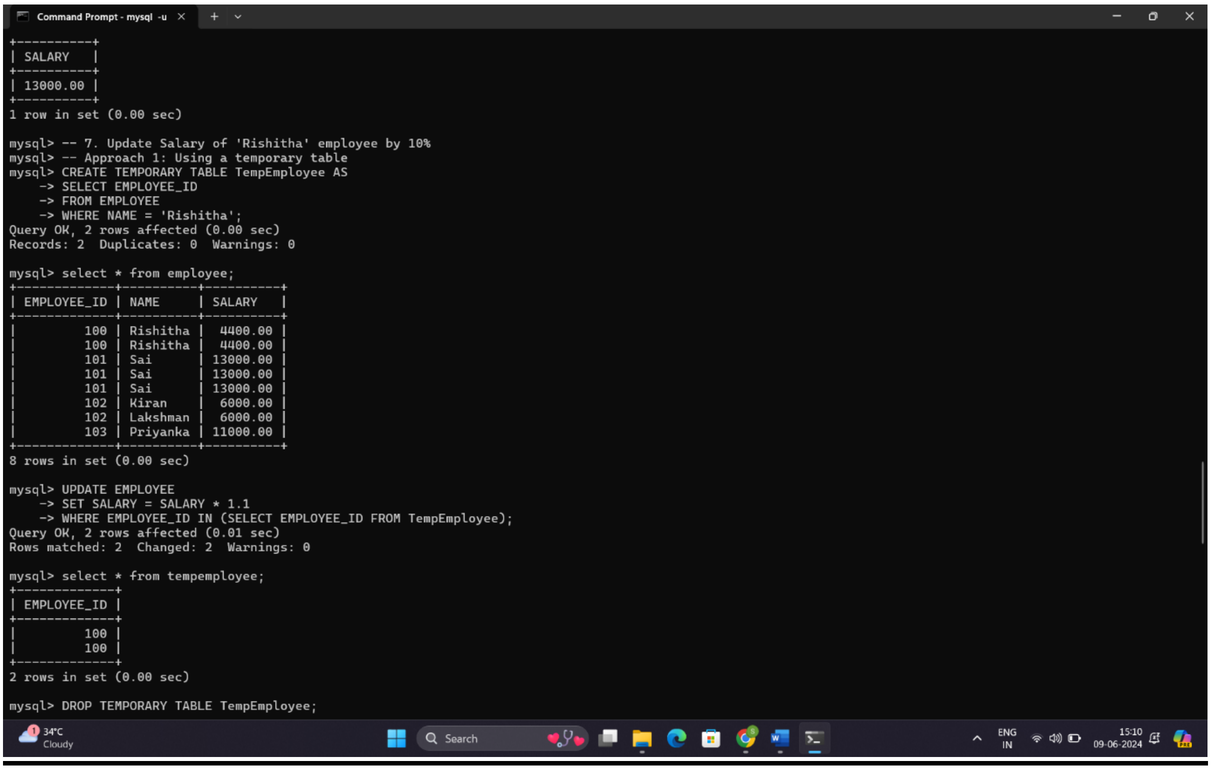
```
mysql> SELECT DISTINCT EMPLOYEE_ID, NAME, SALARY FROM EMPLOYEE;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql> SELECT EMPLOYEE_ID, NAME, SALARY FROM EMPLOYEE GROUP BY EMPLOYEE_ID, NAME, SALARY;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql> SELECT * FROM EMPLOYEE
    -> ORDER BY EMPLOYEE_ID
    -> LIMIT 5;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql>
mysql> -- 4. Query to select LAST 5 records from the given table
mysql> SELECT * FROM EMPLOYEE
    -> ORDER BY EMPLOYEE_ID DESC
    -> LIMIT 5;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
```



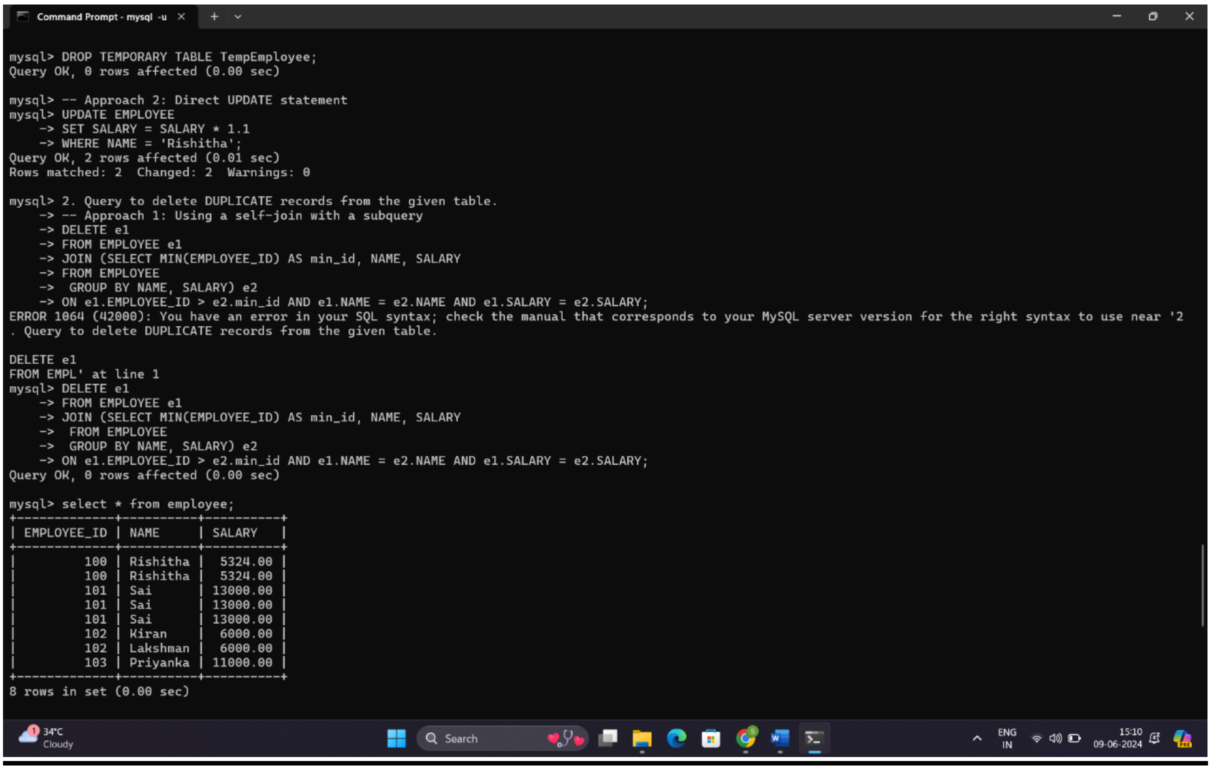
```
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         103 | Priyanka | 11000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
+-------------+----------+----------+
5 rows in set (0.00 sec)

mysql> -- 5. Query to select second MAX Salary
mysql> -- Approach 1: Using MAX function
mysql> SELECT MAX(SALARY) AS SECOND_MAX_SALARY
    -> FROM EMPLOYEE
    -> WHERE SALARY < (SELECT MAX(SALARY) FROM EMPLOYEE);
+-------------------+
| SECOND_MAX_SALARY |
+-------------------+
|          11000.00 |
+-------------------+
1 row in set (0.00 sec)

mysql> ^C
mysql> -- Approach 2: Without using MAX function
mysql> SELECT SALARY AS SECOND_MAX_SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 1
    -> ) AS T;
+-------------------+
| SECOND_MAX_SALARY |
+-------------------+
|          13000.00 |
+-------------------+
1 row in set (0.00 sec)

mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
```



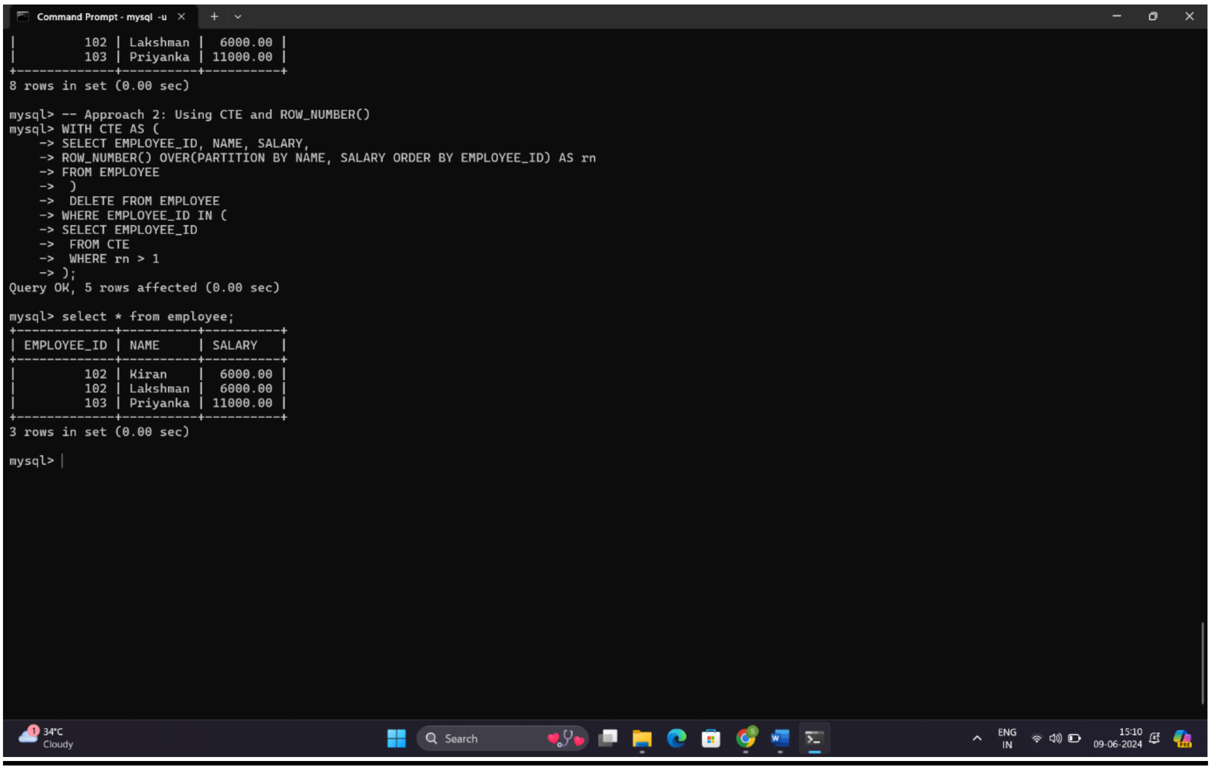
```
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)

mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)

mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    ->     ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)

mysql> -- 6. Query to find third MAX Salaried employee without using Analytic Functions
mysql> SELECT SALARY
    -> FROM (
    ->     SELECT SALARY
    ->     FROM EMPLOYEE
    -> ORDER BY SALARY DESC
    ->     LIMIT 1 OFFSET 2
    -> ) AS T;
+----------+
| SALARY   |
```



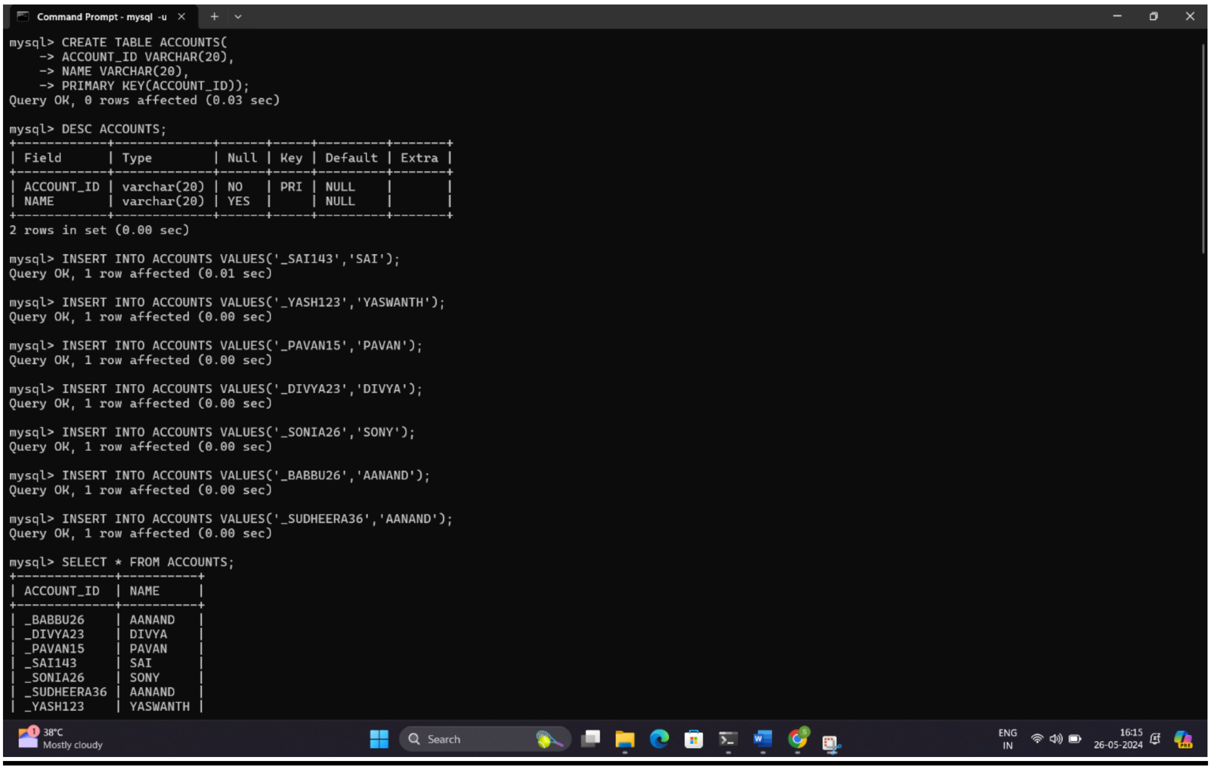
```
+----------+
| SALARY   |
+----------+
| 13000.00 |
+----------+
1 row in set (0.00 sec)

mysql> -- 7. Update Salary of 'Rishitha' employee by 10%
mysql> -- Approach 1: Using a temporary table
mysql> CREATE TEMPORARY TABLE TempEmployee AS
    -> SELECT EMPLOYEE_ID
    -> FROM EMPLOYEE
    -> WHERE NAME = 'Rishitha';
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from employee;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  4400.00 |
|         100 | Rishitha |  4400.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
8 rows in set (0.00 sec)

mysql> UPDATE EMPLOYEE
    -> SET SALARY = SALARY * 1.1
    -> WHERE EMPLOYEE_ID IN (SELECT EMPLOYEE_ID FROM TempEmployee);
Query OK, 2 rows affected (0.01 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> select * from tempemployee;
+-------------+
| EMPLOYEE_ID |
+-------------+
|         100 |
|         100 |
+-------------+
2 rows in set (0.00 sec)

mysql> DROP TEMPORARY TABLE TempEmployee;
```



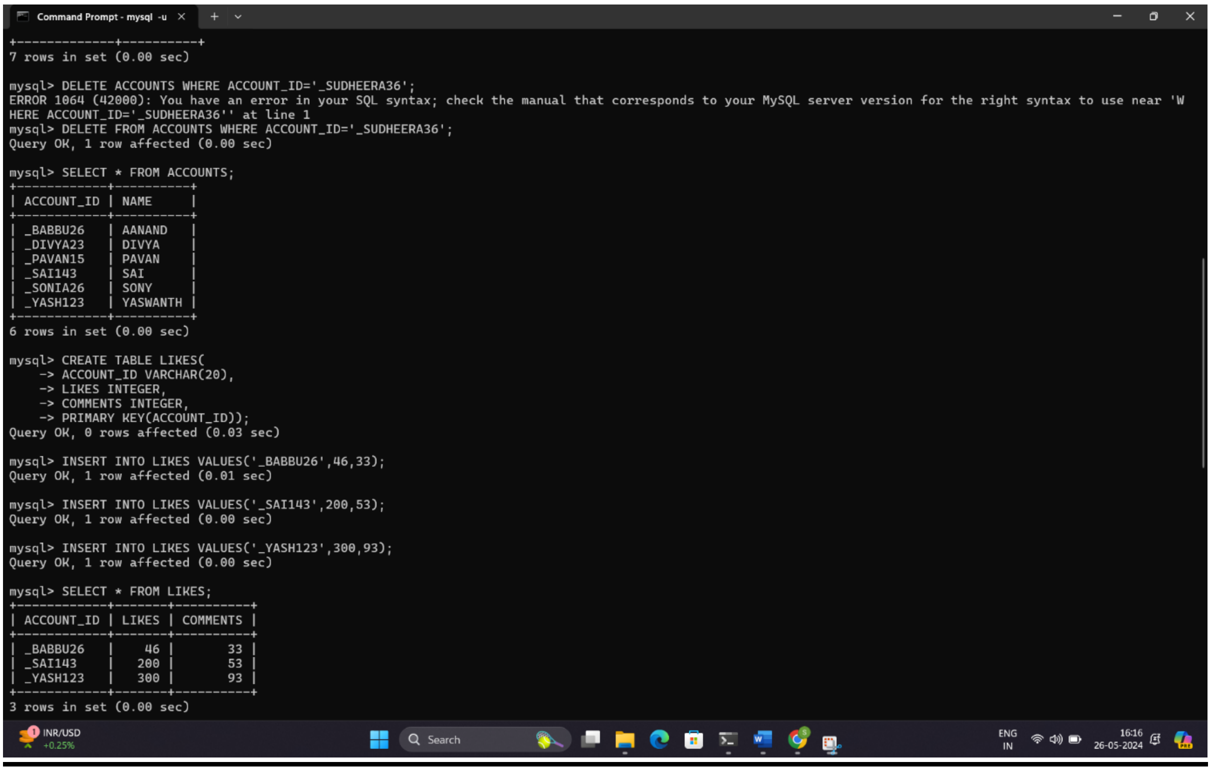
```
mysql> DROP TEMPORARY TABLE TempEmployee;
Query OK, 0 rows affected (0.00 sec)

mysql> -- Approach 2: Direct UPDATE statement
mysql> UPDATE EMPLOYEE
    -> SET SALARY = SALARY * 1.1
    -> WHERE NAME = 'Rishitha';
Query OK, 2 rows affected (0.01 sec)
Rows matched: 2  Changed: 2  Warnings: 0

mysql> 2. Query to delete DUPLICATE records from the given table.
    -> -- Approach 1: Using a self-join with a subquery
    -> DELETE e1
    -> FROM EMPLOYEE e1
    -> JOIN (SELECT MIN(EMPLOYEE_ID) AS min_id, NAME, SALARY
    -> FROM EMPLOYEE
    ->  GROUP BY NAME, SALARY) e2
    -> ON e1.EMPLOYEE_ID > e2.min_id AND e1.NAME = e2.NAME AND e1.SALARY = e2.SALARY;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '2
. Query to delete DUPLICATE records from the given table.

DELETE e1
FROM EMPL' at line 1
mysql> DELETE e1
    -> FROM EMPLOYEE e1
    -> JOIN (SELECT MIN(EMPLOYEE_ID) AS min_id, NAME, SALARY
    ->  FROM EMPLOYEE
    ->  GROUP BY NAME, SALARY) e2
    -> ON e1.EMPLOYEE_ID > e2.min_id AND e1.NAME = e2.NAME AND e1.SALARY = e2.SALARY;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from employee;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         100 | Rishitha |  5324.00 |
|         100 | Rishitha |  5324.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         101 | Sai      | 13000.00 |
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
8 rows in set (0.00 sec)
```



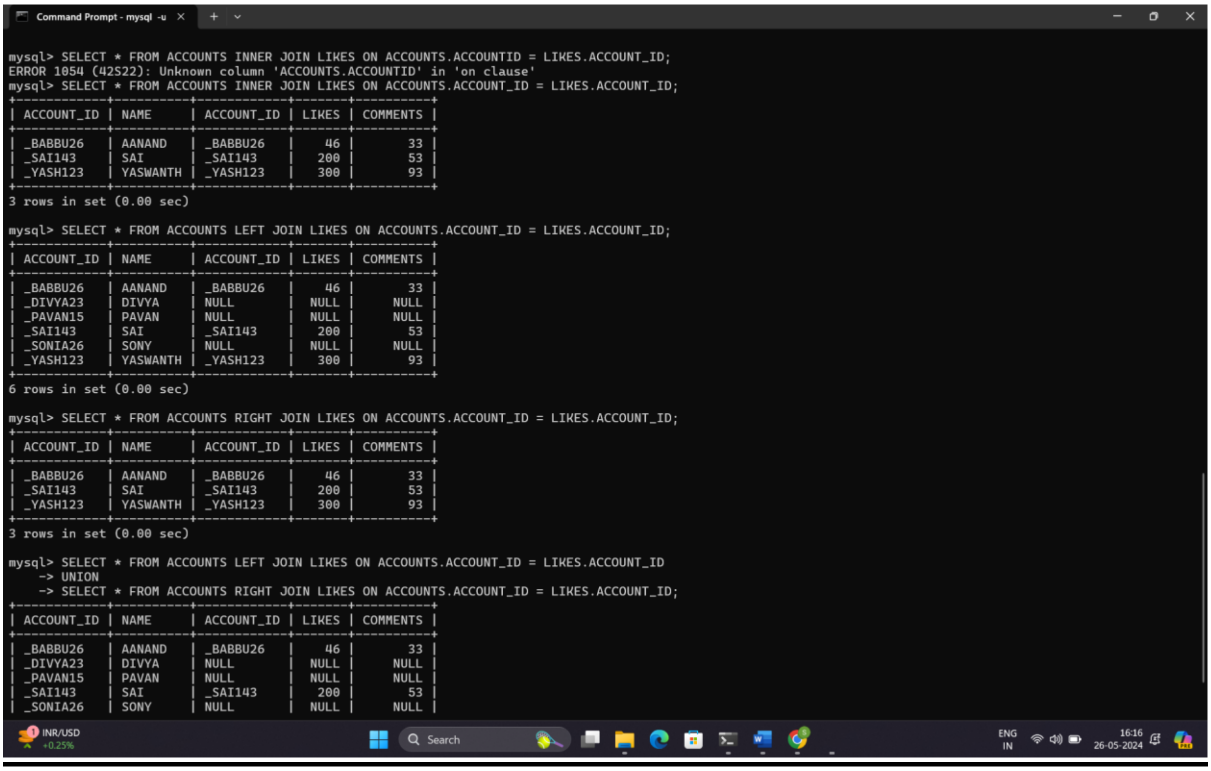
```
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
8 rows in set (0.00 sec)

mysql> -- Approach 2: Using CTE and ROW_NUMBER()
mysql> WITH CTE AS (
    -> SELECT EMPLOYEE_ID, NAME, SALARY,
    -> ROW_NUMBER() OVER(PARTITION BY NAME, SALARY ORDER BY EMPLOYEE_ID) AS rn
    -> FROM EMPLOYEE
    ->  )
    ->  DELETE FROM EMPLOYEE
    -> WHERE EMPLOYEE_ID IN (
    -> SELECT EMPLOYEE_ID
    ->  FROM CTE
    ->  WHERE rn > 1
    -> );
Query OK, 5 rows affected (0.00 sec)

mysql> select * from employee;
+-------------+----------+----------+
| EMPLOYEE_ID | NAME     | SALARY   |
+-------------+----------+----------+
|         102 | Kiran    |  6000.00 |
|         102 | Lakshman |  6000.00 |
|         103 | Priyanka | 11000.00 |
+-------------+----------+----------+
3 rows in set (0.00 sec)

mysql> |
```



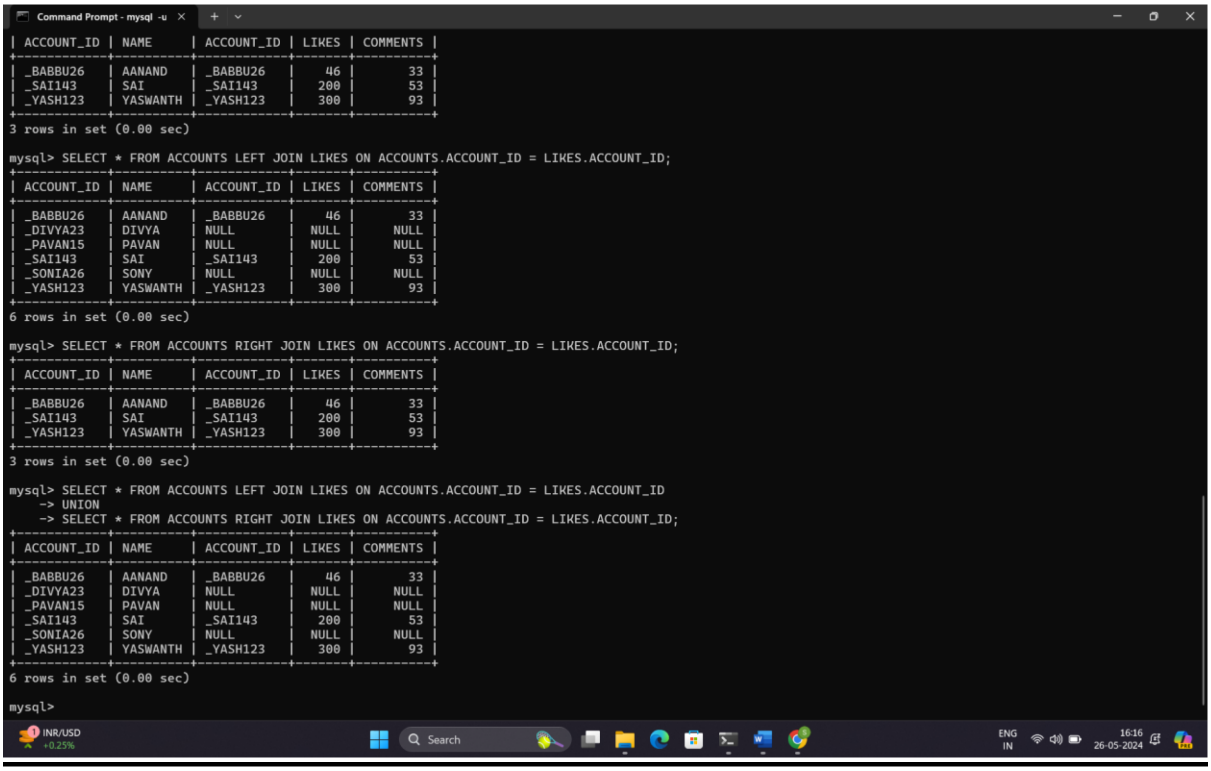
```
mysql> CREATE TABLE ACCOUNTS(
    -> ACCOUNT_ID VARCHAR(20),
    -> NAME VARCHAR(20),
    -> PRIMARY KEY(ACCOUNT_ID));
Query OK, 0 rows affected (0.03 sec)

mysql> DESC ACCOUNTS;
+------------+-------------+------+-----+---------+-------+
| Field      | Type        | Null | Key | Default | Extra |
+------------+-------------+------+-----+---------+-------+
| ACCOUNT_ID | varchar(20) | NO   | PRI | NULL    |       |
| NAME       | varchar(20) | YES  |     | NULL    |       |
+------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)

mysql> INSERT INTO ACCOUNTS VALUES('_SAI143','SAI');
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO ACCOUNTS VALUES('_YASH123','YASWANTH');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO ACCOUNTS VALUES('_PAVAN15','PAVAN');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO ACCOUNTS VALUES('_DIVYA23','DIVYA');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO ACCOUNTS VALUES('_SONIA26','SONY');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO ACCOUNTS VALUES('_BABBU26','AANAND');
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO ACCOUNTS VALUES('_SUDHEERA36','AANAND');
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM ACCOUNTS;
+-------------+----------+
| ACCOUNT_ID  | NAME     |
+-------------+----------+
| _BABBU26    | AANAND   |
| _DIVYA23    | DIVYA    |
| _PAVAN15    | PAVAN    |
| _SAI143     | SAI      |
| _SONIA26    | SONY     |
| _SUDHEERA36 | AANAND   |
| _YASH123    | YASWANTH |
```



```
+-------------+----------+
7 rows in set (0.00 sec)

mysql> DELETE ACCOUNTS WHERE ACCOUNT_ID='_SUDHEERA36';
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'W
HERE ACCOUNT_ID='_SUDHEERA36'' at line 1
mysql> DELETE FROM ACCOUNTS WHERE ACCOUNT_ID='_SUDHEERA36';
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM ACCOUNTS;
+------------+----------+
| ACCOUNT_ID | NAME     |
+------------+----------+
| _BABBU26   | AANAND   |
| _DIVYA23   | DIVYA    |
| _PAVAN15   | PAVAN    |
| _SAI143    | SAI      |
| _SONIA26   | SONY     |
| _YASH123   | YASWANTH |
+------------+----------+
6 rows in set (0.00 sec)

mysql> CREATE TABLE LIKES(
    -> ACCOUNT_ID VARCHAR(20),
    -> LIKES INTEGER,
    -> COMMENTS INTEGER,
    -> PRIMARY KEY(ACCOUNT_ID));
Query OK, 0 rows affected (0.03 sec)

mysql> INSERT INTO LIKES VALUES('_BABBU26',46,33);
Query OK, 1 row affected (0.01 sec)

mysql> INSERT INTO LIKES VALUES('_SAI143',200,53);
Query OK, 1 row affected (0.00 sec)

mysql> INSERT INTO LIKES VALUES('_YASH123',300,93);
Query OK, 1 row affected (0.00 sec)

mysql> SELECT * FROM LIKES;
+------------+-------+----------+
| ACCOUNT_ID | LIKES | COMMENTS |
+------------+-------+----------+
| _BABBU26   |    46 |       33 |
| _SAI143    |   200 |       53 |
| _YASH123   |   300 |       93 |
+------------+-------+----------+
3 rows in set (0.00 sec)
```



```
mysql> SELECT * FROM ACCOUNTS INNER JOIN LIKES ON ACCOUNTS.ACCOUNTID = LIKES.ACCOUNT_ID;
ERROR 1054 (42S22): Unknown column 'ACCOUNTS.ACCOUNTID' in 'on clause'
mysql> SELECT * FROM ACCOUNTS INNER JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID;
+------------+----------+------------+-------+----------+
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _YASH123   | YASWANTH | _YASH123   |   300 |       93 |
+------------+----------+------------+-------+----------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM ACCOUNTS LEFT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID;
+------------+----------+------------+-------+----------+
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _DIVYA23   | DIVYA    | NULL       |  NULL |     NULL |
| _PAVAN15   | PAVAN    | NULL       |  NULL |     NULL |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _SONIA26   | SONY     | NULL       |  NULL |     NULL |
| _YASH123   | YASWANTH | _YASH123   |   300 |       93 |
+------------+----------+------------+-------+----------+
6 rows in set (0.00 sec)

mysql> SELECT * FROM ACCOUNTS RIGHT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID;
+------------+----------+------------+-------+----------+
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _YASH123   | YASWANTH | _YASH123   |   300 |       93 |
+------------+----------+------------+-------+----------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM ACCOUNTS LEFT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID
    -> UNION
    -> SELECT * FROM ACCOUNTS RIGHT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID;
+------------+----------+------------+-------+----------+
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _DIVYA23   | DIVYA    | NULL       |  NULL |     NULL |
| _PAVAN15   | PAVAN    | NULL       |  NULL |     NULL |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _SONIA26   | SONY     | NULL       |  NULL |     NULL |
```



```
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _YASH123   | YASWANTH | _YASH123   |   300 |       93 |
+------------+----------+------------+-------+----------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM ACCOUNTS LEFT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID;
+------------+----------+------------+-------+----------+
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _DIVYA23   | DIVYA    | NULL       |  NULL |     NULL |
| _PAVAN15   | PAVAN    | NULL       |  NULL |     NULL |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _SONIA26   | SONY     | NULL       |  NULL |     NULL |
| _YASH123   | YASWANTH | _YASH123   |   300 |       93 |
+------------+----------+------------+-------+----------+
6 rows in set (0.00 sec)

mysql> SELECT * FROM ACCOUNTS RIGHT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID;
+------------+----------+------------+-------+----------+
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _YASH123   | YASWANTH | _YASH123   |   300 |       93 |
+------------+----------+------------+-------+----------+
3 rows in set (0.00 sec)

mysql> SELECT * FROM ACCOUNTS LEFT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID
    -> UNION
    -> SELECT * FROM ACCOUNTS RIGHT JOIN LIKES ON ACCOUNTS.ACCOUNT_ID = LIKES.ACCOUNT_ID;
+------------+----------+------------+-------+----------+
| ACCOUNT_ID | NAME     | ACCOUNT_ID | LIKES | COMMENTS |
+------------+----------+------------+-------+----------+
| _BABBU26   | AANAND   | _BABBU26   |    46 |       33 |
| _DIVYA23   | DIVYA    | NULL       |  NULL |     NULL |
| _PAVAN15   | PAVAN    | NULL       |  NULL |     NULL |
| _SAI143    | SAI      | _SAI143    |   200 |       53 |
| _SONIA26   | SONY     | NULL       |  NULL |     NULL |
| _YASH123   | YASWANTH | _YASH123   |   300 |       93 |
+------------+----------+------------+-------+----------+
6 rows in set (0.00 sec)

mysql>
```