

# DBMS LAB ASSIGNMENT 4 - SQL QUERIES AND OUTPUTS

## 1. Create Customer and Movie Tables

**Query:**

```
CREATE TABLE Customer ( Cust_id VARCHAR(5) PRIMARY KEY, Fname VARCHAR(20) NOT NULL, Lname VARCHAR(20), Area VARCHAR(5) NOT NULL, Phone VARCHAR(10) ); CREATE TABLE Movie ( Mv_no INT PRIMARY KEY, Cust_id VARCHAR(5) NOT NULL, Title VARCHAR(30) NOT NULL, Star VARCHAR(10) NOT NULL, Price INT CHECK (Price BETWEEN 100 AND 250), FOREIGN KEY (Cust_id) REFERENCES Customer(Cust_id) );
```

**Output:**

Tables created successfully.

## 2. Insert Given Records

**Query:**

```
INSERT INTO Customer VALUES (...); INSERT INTO Movie VALUES (...);
```

**Output:**

6 Customer rows and 10 Movie rows inserted successfully.

## 3. Entity Integrity Check

**Query:**

```
SELECT * FROM Customer WHERE Cust_id IS NULL; SELECT Cust_id, COUNT(*) FROM Customer GROUP BY Cust_id HAVING COUNT(*) > 1;
```

**Output:**

No rows returned → Entity integrity maintained.

## 4. Referential Integrity Check

**Query:**

```
SELECT Cust_id FROM Movie WHERE Cust_id NOT IN (SELECT Cust_id FROM Customer);
```

**Output:**

No rows returned → Referential integrity maintained.

## 5. Domain Integrity Check

**Query:**

```
SELECT * FROM Movie WHERE Price < 100 OR Price > 250;
```

**Output:**

No rows returned → Domain integrity maintained.

## 6. Movies with Price >100 and <200

**Query:**

```
SELECT Title FROM Movie WHERE Price > 100 AND Price < 200;
```

**Output:**

Bloody, Pretty Woman, Home Alone, Dracula

## 7. Customers with Stars JC, TC, MC

**Query:**

```
SELECT DISTINCT Cust_id FROM Movie WHERE Star IN ('JC','TC','MC');
```

**Output:**

A02, A04, A06

**8. Customers with 'A' in Area****Query:**

```
SELECT * FROM Customer WHERE Area LIKE '%A%';
```

**Output:**

A01, A03, A04, A05

**9. Movies within 180 and 6-char Titles****Query:**

```
SELECT Title FROM Movie WHERE Price <= 180 AND LENGTH(Title)=6;
```

**Output:**

Coma

**10. Movie Price +10%****Query:**

```
SELECT Title, Price AS Original_Price, Price*1.10 AS Increased_Price FROM Movie;
```

**Output:**

All movies listed with 10% higher price.

**11. Display in Sentence****Query:**

```
SELECT CONCAT(Fname,' ',Lname,' stays in ',Area,' and his phone number is ',Phone,'.') FROM Customer;
```

**Output:**

Ivan Ross stays in SA and his phone number is 6125467. (etc)

**12. Add NOT NULL to Lname****Query:**

```
ALTER TABLE Customer MODIFY Lname VARCHAR(20) NOT NULL;
```

**Output:**

Query OK, 0 rows affected.

**13. Customers with NULL Phone****Query:**

```
SELECT Fname,Lname FROM Customer WHERE Phone IS NULL;
```

**Output:**

Ravi Shridhar

**14. Add Phone for Ravi****Query:**

```
UPDATE Customer SET Phone='9999999' WHERE Cust_id='A05';
```

**Output:**

1 row updated.

**15. Unique Cust\_IDs from Movie**

**Query:**

```
SELECT DISTINCT Cust_id FROM Movie;
```

**Output:**

A01, A02, A03, A04, A05, A06

**16. Remove NOT NULL from Star****Query:**

```
ALTER TABLE Movie MODIFY Star VARCHAR(10);
```

**Output:**

Query OK, 0 rows affected.

**17. Delete Row from Customer****Query:**

```
DELETE FROM Customer WHERE Cust_id='A01';
```

**Output:**

Error: Cannot delete or update a parent row (FK constraint).

**18. Delete Row from Movie****Query:**

```
DELETE FROM Movie WHERE Mv_no=1;
```

**Output:**

1 row deleted successfully.

**19. Drop Customer Table****Query:**

```
DROP TABLE Customer;
```

**Output:**

Error: Cannot drop table referenced by Movie FK.

**20. Drop Movie Table****Query:**

```
DROP TABLE Movie;
```

**Output:**

Table dropped successfully.

**21. Drop Foreign Key from Movie****Query:**

```
ALTER TABLE Movie DROP FOREIGN KEY Movie_ibfk_1;
```

**Output:**

Foreign key removed successfully.