



CHRIST
(DEEMED TO BE UNIVERSITY)
B A N G A L O R E • I N D I A

DSC201-3 DATABASE MANAGEMENT SYSTEMS

CIA 3

DBMS project– Indian Police Services

2341322

Submitted on: 16 Oct 2024

Submitted to: Dr Ummesalma M

Project model on Indian Police Service Database
Using MYSQL

Tables creation

```
CREATE TABLE Police_Station (  
    Station_ID INT PRIMARY KEY AUTO_INCREMENT,  
    Name VARCHAR(50) NOT NULL,  
    Location VARCHAR(100) NOT NULL,  
    Contact_No VARCHAR(15)  
);  
  
CREATE TABLE Officer (  
    Officer_ID INT PRIMARY KEY AUTO_INCREMENT,  
    Name VARCHAR(50) NOT NULL,  
    `Rank` ENUM('Inspector', 'Sub-Inspector', 'Constable') NOT NULL,  
    Age INT CHECK (Age > 20),  
    Joining_Date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
    Station_ID INT,  
    FOREIGN KEY (Station_ID) REFERENCES Police_Station(Station_ID)  
);  
  
CREATE TABLE Victim (  
    Victim_ID INT PRIMARY KEY AUTO_INCREMENT,  
    Name VARCHAR(50) NOT NULL,  
    Age INT,  
    Contact_No VARCHAR(15)  
);  
  
CREATE TABLE Criminal (  
    Criminal_ID INT PRIMARY KEY AUTO_INCREMENT,  
    Name VARCHAR(50) NOT NULL,  
    Crime_Type VARCHAR(50) NOT NULL  
);  
  
CREATE TABLE Case_Record (  
    Case_ID INT PRIMARY KEY AUTO_INCREMENT,  
    Case_Type VARCHAR(50) NOT NULL,
```

```

Date_Reported DATE NOT NULL,
Status VARCHAR(20) CHECK (Status IN ('Open', 'Closed', 'Pending')) DEFAULT 'Open',
Officer_ID INT,
Victim_ID INT,
Criminal_ID INT,
FOREIGN KEY (Officer_ID) REFERENCES Officer(Officer_ID),
FOREIGN KEY (Victim_ID) REFERENCES Victim(Victim_ID),
FOREIGN KEY (Criminal_ID) REFERENCES Criminal(Criminal_ID),
UNIQUE (Case_Type, Date_Reported)
);

```

Inserting data into Police_Station

```

INSERT INTO Police_Station (Name, Location, Contact_No)
VALUES
('Station A', 'New Delhi', '1234567890'),
('Station B', 'Mumbai', '9876543210'),
('Station C', 'Chennai', '5551234567'),
('Station D', 'Kolkata', '4449876543'),
('Station E', 'Hyderabad', '3337654321');

```

Inserting data into Officer

```

INSERT INTO Officer (Name, `Rank`, Age, Joining_Date, Station_ID)
VALUES
('Ravi Kumar', 'Inspector', 45, '2020-08-01', 1),
('Amit Sharma', 'Sub-Inspector', 38, '2018-04-15', 2),
('Suresh Nair', 'Constable', 28, '2022-02-10', 3),
('Preeti Singh', 'Inspector', 40, '2017-12-05', 1),
('Rajesh Gupta', 'Constable', 30, '2021-01-20', 4);

```

Inserting data into Victim

```

INSERT INTO Victim (Name, Age, Contact_No)
VALUES
('Anil Mehta', 32, '1234512345'),
('Sunita Rao', 28, '2345623456'),
('Manoj Tiwari', 40, '3456734567'),
('Geeta Verma', 25, '4567845678'),
('Ramesh Patel', 50, '5678956789');

```

Inserting data into Criminal

```

INSERT INTO Criminal (Name, Crime_Type)
VALUES
('Ajay Sharma', 'Theft'),

```

```
('Rahul Yadav', 'Robbery'),
('Sanjay Mishra', 'Fraud'),
('Mohit Gupta', 'Murder'),
('Vikas Saxena', 'Burglary');
```

Inserting data into Case_Record

```
INSERT INTO Case_Record (Case_Type, Date_Reported, Status, Officer_ID, Victim_ID,
Criminal_ID)
```

```
VALUES
```

```
('Theft', '2024-10-10', 'Open', 1, 1, 1),
('Robbery', '2024-09-20', 'Closed', 2, 2, 2),
('Fraud', '2024-08-15', 'Pending', 1, 3, 3),
('Murder', '2024-07-05', 'Open', 4, 4, 4),
('Burglary', '2024-06-12', 'Closed', 5, 5, 5);
```

```
SELECT * FROM Officer;
```

```
SELECT * FROM Victim;
```

```
SELECT * FROM Criminal;
```

```
SELECT * FROM Case_Record;
```

Adding Email column to Officer table (Alter)

```
ALTER TABLE Officer ADD Email VARCHAR(100);
desc table Officer;
```

Drop the Email column from Officer table

```
ALTER TABLE Officer DROP COLUMN Email;
desc table Officer;
```

Truncate Case_Record table

```
TRUNCATE TABLE Case_Record;
desc table Case_Record;
```

Rename Victim table to Complainant

```
RENAME TABLE Victim TO Complainant;
show tables;
```

Output:

```
+-----+-----+-----+-----+-----+-----+
| Officer_ID | Name      | Rank      | Age | Joining_Date      | Station_ID |
+-----+-----+-----+-----+-----+-----+
| 1 | Ravi Kumar | Inspector | 45 | 2020-08-01 00:00:00 | 1 |
| 2 | Amit Sharma | Sub-Inspector | 38 | 2018-04-15 00:00:00 | 2 |
| 3 | Suresh Nair | Constable | 28 | 2022-02-10 00:00:00 | 3 |
| 4 | Preeti Singh | Inspector | 40 | 2017-12-05 00:00:00 | 1 |
```

5	Rajesh Gupta	Constable	30	2021-01-20 00:00:00	4
---	--------------	-----------	----	---------------------	---

Victim_ID	Name	Age	Contact_No
1	Anil Mehta	32	1234512345
2	Sunita Rao	28	2345623456
3	Manoj Tiwari	40	3456734567
4	Geeta Verma	25	4567845678
5	Ramesh Patel	50	5678956789

Criminal_ID	Name	Crime_Type
1	Ajay Sharma	Theft
2	Rahul Yadav	Robbery
3	Sanjay Mishra	Fraud
4	Mohit Gupta	Murder
5	Vikas Saxena	Burglary

Case_ID	Case_Type	Date_Reported	Status	Officer_ID	Victim_ID	Criminal_ID
1	Theft	2024-10-10	Open	1	1	1
2	Robbery	2024-09-20	Closed	2	2	2
3	Fraud	2024-08-15	Pending	1	3	3
4	Murder	2024-07-05	Open	4	4	4
5	Burglary	2024-06-12	Closed	5	5	5

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Officer	NULL	ALL	NULL	NULL	NULL	NULL	5	100.00	NULL

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Officer	NULL	ALL	NULL	NULL	NULL	NULL	5	100.00	NULL

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Case_Record	NULL	ALL	NULL	NULL	NULL	NULL	1	100.00	NULL

Tables_in_sandbox_db											
case_record											
complainant											
criminal											
officer											
police_station											

1) Example of WHERE clause

SELECT * FROM Officer WHERE Age > 30;

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Inspector	45	2020-08-01 00:00:00	1
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1

2) Example of GROUP BY clause

SELECT `Rank`, COUNT(*) AS NumberOfOfficers
FROM Officer
GROUP BY `Rank`;

Rank	NumberOfOfficers
Inspector	2
Sub-Inspector	1
Constable	2

3) Example of HAVING clause

SELECT `Rank`, COUNT(*) AS NumberOfOfficers
FROM Officer
GROUP BY `Rank`
HAVING COUNT(*) > 1;

Rank	NumberOfOfficers
Inspector	2
Constable	2

4) Example of ORDER BY clause

```
SELECT * FROM Case_Record ORDER BY Date_Reported DESC;
```

```
| Crime_Type |
```

```
+-----+
```

```
| Theft      |
```

```
| Robbery    |
```

```
| Fraud      |
```

```
| Murder     |
```

```
| Burglary   |
```

```
+-----+
```

5) Example of aggregate functions

```
SELECT AVG(Age) AS AverageAge FROM Complainant;
```

```
+-----+
```

```
| AverageAge |
```

```
+-----+
```

```
| 35.0000    |
```

```
+-----+
```

6) Example of DISTINCT keyword

```
SELECT DISTINCT Crime_Type FROM Criminal;
```

7) Example of LIMIT keyword

```
SELECT * FROM Case_Record LIMIT 3;
```

8) Example of pattern matching using LIKE

```
SELECT * FROM Complainant WHERE Name LIKE 'A%'; -- Names starting with 'A'
```

```
+-----+-----+-----+-----+
```

```
| Victim_ID | Name      | Age | Contact_No |
```

```
+-----+-----+-----+-----+
```

```
| 1         | Anil Mehta | 32  | 1234512345 |
```

```
+-----+-----+-----+-----+
```

9) Update the rank and station for Ravi Kumar

```
UPDATE Officer
```

```
SET 'Rank' = 'Sub-Inspector', Station_ID = 2
```

```
WHERE Name = 'Ravi Kumar';
```

```
+-----+
```

```
| Officer_Details |
```

```
+-----+
```

```
| Ravi Kumar (Sub-Inspector) |
```

```
| Amit Sharma (Sub-Inspector) |
```

```
| Suresh Nair (Constable)    |
```

```
| Preeti Singh (Inspector)   |
```

```
| Rajesh Gupta (Constable)   |
```

```
+-----+
```

10) RIGHT JOIN among Officer, Case_Record, and Complainant

```
SELECT Officer.Name, Case_Record.Case_Type, Complainant.Name AS
```

```
Complainant_Name
```

```
FROM Officer
```

```
RIGHT JOIN Case_Record ON Officer.Officer_ID = Case_Record.Officer_ID
```

```
RIGHT JOIN Complainant ON Case_Record.Victim_ID = Complainant.Victim_ID;
```

```
+-----+-----+-----+
```

```
| Name | Case_Type | Complainant_Name |
```

```
+-----+-----+-----+
```


NULL NULL	Anil Mehta	
NULL NULL	Sunita Rao	
NULL NULL	Manoj Tiwari	
NULL NULL	Geeta Verma	
NULL NULL	Ramesh Patel	
+-----+-----+-----+		

11) LEFT JOIN among Officer, Police_Station, and Case_Record

```
SELECT Officer.Name AS Officer_Name, Police_Station.Name AS Station_Name,
Case_Record.Case_Type
FROM Officer
LEFT JOIN Police_Station ON Officer.Station_ID = Police_Station.Station_ID
LEFT JOIN Case_Record ON Officer.Officer_ID = Case_Record.Officer_ID;
```

+-----+-----+-----+		
Officer_Name	Station_Name	Case_Type
+-----+-----+-----+		
Ravi Kumar	Station B	NULL
Amit Sharma	Station B	NULL
Suresh Nair	Station C	NULL
Preeti Singh	Station A	NULL
Rajesh Gupta	Station D	NULL
+-----+-----+-----+		

12) FULL OUTER JOIN between Officer and Police_Station

```
SELECT Officer.Name AS Officer_Name, Police_Station.Name AS Station_Name
FROM Officer
LEFT JOIN Police_Station ON Officer.Station_ID = Police_Station.Station_ID
UNION
SELECT Officer.Name AS Officer_Name, Police_Station.Name AS Station_Name
FROM Officer
RIGHT JOIN Police_Station ON Officer.Station_ID = Police_Station.Station_ID;
```

+-----+-----+	
Officer_Name	Station_Name
+-----+-----+	
Ravi Kumar	Station B
Amit Sharma	Station B
Suresh Nair	Station C
Preeti Singh	Station A
Rajesh Gupta	Station D
NULL	Station E
NULL	Station F
NULL	Station G
+-----+-----+	

13) CONCAT to combine officer's name and rank

```
SELECT CONCAT(Name, ' (', 'Rank', ')') AS Officer_Details FROM Officer;
```

+-----+	
Officer_Details	
+-----+	
Ravi Kumar (Sub-Inspector)	
Amit Sharma (Sub-Inspector)	
Suresh Nair (Constable)	

Preeti Singh (Inspector)
Rajesh Gupta (Constable)

14) SUBSTRING to get the first 5 characters of the criminal name

```
SELECT SUBSTRING(Name, 1, 5) AS Shortened_Name FROM Criminal;
```

Shortened_Name

Ajay

Ajay

Rahul

Sanja

Mohit

Vikas

15) UPPER to convert names to uppercase

```
SELECT UPPER(Name) AS Upper_Name FROM Complainant;
```

Upper_Name

ANIL MEHTA

SUNITA RAO

MANOJ TIWARI

GEETA VERMA

RAMESH PATEL

16) average age of officers

```
SELECT AVG(Age) AS Avg_Age FROM Officer;
```

Avg_Age

36.2000

17) minimum and maximum age of complainants

```
SELECT MIN(Age) AS Min_Age, MAX(Age) AS Max_Age FROM Complainant;
```

Min_Age	Max_Age
---------	---------

25	50
----	----

18) current date

```
SELECT CURRENT_DATE AS Today;
```

Today

```
+-----+
| 2024-10-16 |
+-----+
```

19) Extract year from the case report date

```
SELECT Case_Type, YEAR(Date_Reported) AS Year_Reported FROM Case_Record;
```

20) Calculate the number of days since the case was reported

```
SELECT Case_Type, DATEDIFF(CURRENT_DATE, Date_Reported) AS
Days_Since_Reported FROM Case_Record;
```

21) Self-join on Officer table to find pairs of officers who work in the same police station

```
SELECT A.Name AS Officer1, B.Name AS Officer2, A.Station_ID
FROM Officer A, Officer B
WHERE A.Station_ID = B.Station_ID AND A.Officer_ID <> B.Officer_ID;
```

```
+-----+-----+-----+
| Officer1 | Officer2 | Station_ID |
+-----+-----+-----+
| Ravi Kumar | Amit Sharma | 2 |
| Amit Sharma | Ravi Kumar | 2 |
+-----+-----+-----+
```

22) officers who have dealt with a case involving a criminal whose name starts with 'A'

```
SELECT Officer.Name
FROM Officer
WHERE Officer.Officer_ID IN (
    SELECT Case_Record.Officer_ID
    FROM Case_Record
    JOIN Criminal ON Case_Record.Criminal_ID = Criminal.Criminal_ID
    WHERE Criminal.Name LIKE 'A%'
);
```

```
+-----+
| Name |
+-----+
| Ravi Kumar |
| Amit Sharma |
| Suresh Nair |
| Preeti Singh |
| Rajesh Gupta |
+-----+
```

23) Find the oldest criminal associated with a closed case

```
SELECT Name, Crime_Type
FROM Criminal
WHERE Criminal_ID = (
    SELECT Criminal_ID
    FROM Case_Record
    WHERE Status = 'Closed'
    ORDER BY Date_Reported DESC
    LIMIT 1
);
```

Name
ANIL MEHTA

24) Find officers who are older than the average age of officers

```
SELECT Name, Age
FROM Officer
WHERE Age > (SELECT AVG(Age) FROM Officer);
```

Name	Age
Ravi Kumar	45
Amit Sharma	38
Preeti Singh	40

25) Find officers who are younger than any officer from Station B

```
SELECT Name, Age
FROM Officer
WHERE Age < ANY (SELECT Age FROM Officer WHERE Station_ID = 2);
```

Name	Age
Amit Sharma	38
Suresh Nair	28
Preeti Singh	40
Rajesh Gupta	30

26) Officers who are not associated with any case (Set Difference using LEFT JOIN and IS NULL)

```
SELECT Officer.Name
FROM Officer
LEFT JOIN Case_Record ON Officer.Officer_ID = Case_Record.Officer_ID
WHERE Case_Record.Officer_ID IS NULL;
```

Name
Ravi Kumar

Amit Sharma
Suresh Nair
Preeti Singh
Rajesh Gupta
+-----+

27) cartesian product

```
SELECT Officer.Name AS Officer_Name, Police_Station.Name AS Station_Name
FROM Officer
CROSS JOIN Police_Station;
```

+-----+
Officer_Name Station_Name
+-----+
Rajesh Gupta Station A
Preeti Singh Station A
Suresh Nair Station A
Amit Sharma Station A
Ravi Kumar Station A
Rajesh Gupta Station B
Preeti Singh Station B
Suresh Nair Station B
Amit Sharma Station B
Ravi Kumar Station B
Rajesh Gupta Station C
Preeti Singh Station C
Suresh Nair Station C
Amit Sharma Station C
Ravi Kumar Station C
Rajesh Gupta Station D
Preeti Singh Station D
Suresh Nair Station D
Amit Sharma Station D
Ravi Kumar Station D
Rajesh Gupta Station E
Preeti Singh Station E
Suresh Nair Station E
Amit Sharma Station E
Ravi Kumar Station E
Rajesh Gupta Station F
Preeti Singh Station F
Suresh Nair Station F
Amit Sharma Station F
Ravi Kumar Station F
Rajesh Gupta Station G
Preeti Singh Station G
Suresh Nair Station G
Amit Sharma Station G
Ravi Kumar Station G
+-----+

28) Find officers who have worked in every police station

```

SELECT Officer.Name
FROM Officer
WHERE NOT EXISTS (
    SELECT Station_ID
    FROM Police_Station
    WHERE NOT EXISTS (
        SELECT Officer.Station_ID
        FROM Officer
        WHERE Officer.Station_ID = Police_Station.Station_ID
    )
);

```

Officer_Name	Officer_Age	Officer_Rank
Ravi Kumar	45	Sub-Inspector
Amit Sharma	38	Sub-Inspector
Suresh Nair	28	Constable
Preeti Singh	40	Inspector
Rajesh Gupta	30	Constable

29) Rename Operation

```

SELECT Officer.Name AS Officer_Name, Officer.Age AS Officer_Age, Officer.Rank AS
Officer_Rank
FROM Officer;

```

30) Turn off autocommit

```

SET AUTOCOMMIT = 0;

```

31) Start a transaction

```

START TRANSACTION;

```

32) Insert a new officer

```

INSERT INTO Officer (Name, `Rank`, Age, Station_ID)
VALUES ('Deepak Jain', 'Inspector', 35, 1);

```

33) Create a savepoint

```

SAVEPOINT sp1;
select * from officer;

```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Sub-Inspector	45	2020-08-01 00:00:00	2
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
3	Suresh Nair	Constable	28	2022-02-10 00:00:00	3
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1
5	Rajesh Gupta	Constable	30	2021-01-20 00:00:00	4
6	Deepak Jain	Inspector	35	2024-10-16 17:03:56	1

34) Insert another officer

```
INSERT INTO Officer (Name, `Rank`, Age, Station_ID)
```

```
VALUES ('Rahul Mehra', 'Sub-Inspector', 30, 2);
```

```
select * from officer;
```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Sub-Inspector	45	2020-08-01 00:00:00	2
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
3	Suresh Nair	Constable	28	2022-02-10 00:00:00	3
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1
5	Rajesh Gupta	Constable	30	2021-01-20 00:00:00	4
6	Deepak Jain	Inspector	35	2024-10-16 17:03:56	1
7	Rahul Mehra	Sub-Inspector	30	2024-10-16 17:03:56	2

35) Rollback to the savepoint (this will undo the second insert, but keep the first)

```
ROLLBACK TO sp1;
```

```
select * from officer;
```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Sub-Inspector	45	2020-08-01 00:00:00	2
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
3	Suresh Nair	Constable	28	2022-02-10 00:00:00	3
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1
5	Rajesh Gupta	Constable	30	2021-01-20 00:00:00	4
6	Deepak Jain	Inspector	35	2024-10-16 17:03:56	1

```
-- Commit the changes (this will save the first insert)
```

```
COMMIT;
```

36) Turn autocommit back on

```
SET AUTOCOMMIT = 1;
```

37) Create a View Using an Original Table with All Fields

```
CREATE VIEW Officer_View AS
```

```
SELECT * FROM Officer;
```

38) Create a View from the Master Table with Selected Fields Only Satisfying a Condition

```
CREATE VIEW Senior_Officers_View AS
```

```
SELECT Name, `Rank`, Age
```

```
FROM Officer
```

```
WHERE Age > 30;
```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
5	Rajesh Gupta	Inspector	30	2021-01-20 00:00:00	4

39) Create a View Using Nested Queries (At Least 2 Tables)

```
CREATE VIEW Officer_Station_View AS
SELECT o.Name AS Officer_Name, p.Name AS Station_Name
FROM Officer o
INNER JOIN Police_Station p
ON o.Station_ID = p.Station_ID
WHERE o.Age > (SELECT AVG(Age) FROM Officer);
```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
5	Rajesh Gupta	Inspector	30	2021-01-20 00:00:00	4

40) Perform an Equi-Join or Full Outer Join on Any 2 Tables and Make the Result a View

```
CREATE VIEW Officer_Station_Join_View AS
SELECT o.Name AS Officer_Name, o.Rank, p.Name AS Station_Name
FROM Officer o
INNER JOIN Police_Station p
ON o.Station_ID = p.Station_ID;
```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Sub-Inspector	45	2020-08-01 00:00:00	2
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1
5	Rajesh Gupta	Inspector	30	2021-01-20 00:00:00	4
6	Deepak Jain	Inspector	35	2024-10-16 17:03:56	1

41) Update a View Where Only One Row is Updated and Verify in Both Master Table and View

```
UPDATE Officer_View
SET 'Rank' = 'Inspector'
WHERE Name = 'Rajesh Gupta';
SELECT * FROM Officer WHERE Name = 'Rajesh Gupta';
SELECT * FROM Officer_View WHERE Name = 'Rajesh Gupta';
```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Sub-Inspector	45	2020-08-01 00:00:00	2
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1
5	Rajesh Gupta	Inspector	30	2021-01-20 00:00:00	4
6	Deepak Jain	Inspector	35	2024-10-16 17:03:56	1

42) Delete Multiple Records from the Master Table and Verify the Content in Both Master Table and View

```
DELETE FROM Officer
WHERE Age < 30;
SELECT * FROM Officer;
SELECT * FROM Officer_View;
```


Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Sub-Inspector	45	2020-08-01 00:00:00	2
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1
5	Rajesh Gupta	Inspector	30	2021-01-20 00:00:00	4
6	Deepak Jain	Inspector	35	2024-10-16 17:03:56	1
8	Deepak Kumar	Inspector	36	2024-10-16 17:03:56	1
9	Alok Verma	Constable	28	2024-10-16 17:03:56	2
10	Simran Sharma	Sub-Inspector	33	2024-10-16 17:03:56	3

43) Insert at Least 3 Records in a View with Selected Columns and Verify in Both View and Master Table

```
INSERT INTO Officer_View (Name, 'Rank', Age, Station_ID)
VALUES
('Deepak Kumar', 'Inspector', 36, 1),
('Alok Verma', 'Constable', 28, 2),
('Simran Sharma', 'Sub-Inspector', 33, 3);
SELECT * FROM Officer;
SELECT * FROM Officer_View;
```

Officer_ID	Name	Rank	Age	Joining_Date	Station_ID
1	Ravi Kumar	Sub-Inspector	45	2020-08-01 00:00:00	2
2	Amit Sharma	Sub-Inspector	38	2018-04-15 00:00:00	2
4	Preeti Singh	Inspector	40	2017-12-05 00:00:00	1
5	Rajesh Gupta	Inspector	30	2021-01-20 00:00:00	4
6	Deepak Jain	Inspector	35	2024-10-16 17:03:56	1
8	Deepak Kumar	Inspector	36	2024-10-16 17:03:56	1
9	Alok Verma	Constable	28	2024-10-16 17:03:56	2
10	Simran Sharma	Sub-Inspector	33	2024-10-16 17:03:56	3

About the key words(Constraints and operations):

Table Creation

Defines structure and integrity of data within a database.

PRIMARY KEY

Uniquely identifies records, ensuring no duplicates in the column.

AUTO_INCREMENT

Automatically generates unique numeric values for primary key fields.

FOREIGN KEY

Links two tables, ensuring valid relationships between their records.

NOT NULL

Requires data entry, preventing null values in specified columns.

CHECK

Validates column values against conditions for data integrity.

UNIQUE

Ensures all values in a column are distinct and unique.

INSERT

Adds new records to a table with specified values.

SELECT

Retrieves specific records from one or more database tables.

UPDATE

Modifies existing records in a table based on criteria.

DELETE

Removes specified records from a table, managing data effectively.

ALTER TABLE

Modifies table structure by adding, changing, or removing columns.

TRUNCATE

Removes all records quickly without logging individual deletions.

RENAME

Changes the name of an existing table or column.

DESC (Describe)

Displays table structure, including columns, types, and constraints.

JOIN

Combines rows from multiple tables based on related columns.

RIGHT JOIN

Retrieves all records from the right table, matching left table.

LEFT JOIN

Retrieves all records from the left table, matching right table.

FULL OUTER JOIN

Combines results from both sides, including unmatched records.

GROUP BY

Aggregates records with identical values for statistical analysis.

HAVING

Filters grouped records based on specified conditions.

ORDER BY

Sorts records based on specified columns, ascending or descending.

LIMIT

Restricts number of returned records from a query.

DISTINCT

Eliminates duplicate records from query results, ensuring uniqueness.

CONCAT

Combines multiple strings into a single string output.

SUBSTRING

Extracts a portion of a string based on specified criteria.

UPPER

Converts all characters in a string to uppercase.

AVG

Calculates average value of a numeric column for analysis.

MIN

Retrieves the smallest value from a specified column.

MAX

Retrieves the largest value from a specified column.

CURRENT_DATE

Fetches current date from the database server for use.

DATEDIFF

Calculates difference in days between two specified dates.

SELF JOIN

Joins a table to itself for comparative analysis.

Select Officers by Criminal Name

Retrieve officer names linked to criminals starting with 'A'.

Oldest Criminal in Closed Case

Find oldest criminal associated with cases marked as 'Closed'.

Officers Older than Average Age

Select officers whose ages exceed the average officer age.

Officers Younger than Any from Station B

Retrieve officers younger than every officer from Station B.

Officers Without Associated Cases

Identify officers who are not linked to any case.

Cartesian Product

Generate all combinations of officers and police stations.

Officers Working in Every Police Station

Select officers present in all police stations listed.

Rename Operation

Rename columns for better clarity in the output results.

Turn Off Autocommit

Disable autocommit feature for transaction management control.

Start a Transaction

Begin a transaction to manage multiple SQL operations.

Insert New Officer

Add a new officer with specified details into the database.

Create a Savepoint

Establish a point to rollback transaction if needed.

Insert Another Officer

Add an additional officer record to the officer table.

Rollback to Savepoint

Revert to savepoint, undoing actions after the established point.

Commit Changes

Save all changes made during the transaction permanently.

Create a View with All Fields

Define a view displaying all fields from the officer table.

Create a View with Condition

Define a view for officers older than thirty years.

Create a Nested View

Define a view combining officer and station information with criteria.

Equi-Join View Creation

Create a view combining officer and police station data.

Update a View

Modify a view entry and reflect changes in the master table.

Delete Multiple Records

Remove officers below the age of thirty from the database.

Insert Records into View

Add new officer records through a defined view structure.
