Name : U Ganesh Chowdari

Email : ganeshchowdary077@gmail.com

Date: 29-12-2025 Day 2 – Querying & Modifying Data

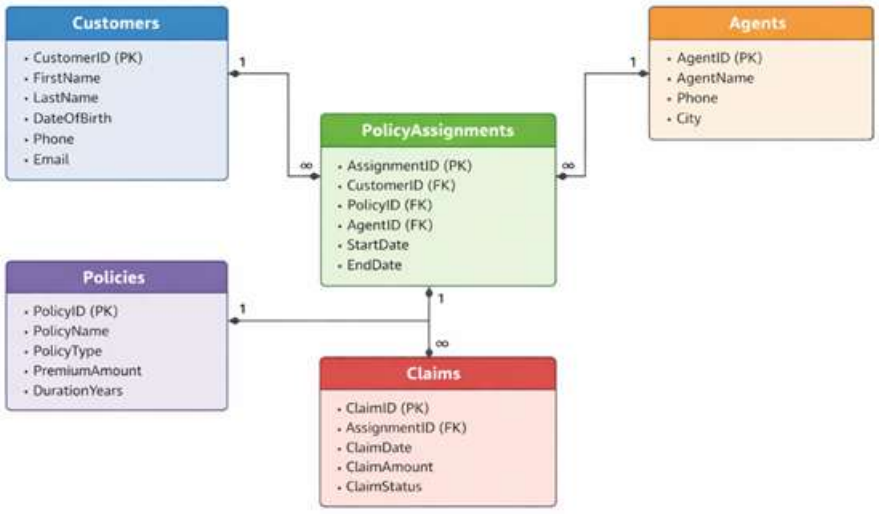
Create Sample Insurance database named **InsuranceDB** with tables, constraints, and initial sample data. Based on following er diagram and descriptions.

Entities we'll model:

- **Customers** – people who buy insurance
- **Policies** – insurance products (Health, Motor, Life)
- **Agents** – insurance agents
- **PolicyAssignments** – which customer bought which policy
- **Claims** – claims raised against policies

Relationship Explanation

- **One Customer → Many PolicyAssignments**
- **One Policy → Many PolicyAssignments**
- **One Agent → Many PolicyAssignments**
- **One PolicyAssignment → Many Claims**



1. Create Database command.

Query:

```
CREATE DATABASE InsuranceDB;
```

Output:

2. Create table commands for all the tables with constraints, relationships etc.

Query:

```
-- Customers Table
CREATE TABLE Customers
(
	CustomerID INT IDENTITY NOT NULL,
	FirstName VARCHAR(30) NOT NULL,
	LastName VARCHAR(30),
	DateOfBirth DATE,
	PhoneNumber VARCHAR(15) NOT NULL,
	Email VARCHAR(50) NOT NULL UNIQUE,
	CONSTRAINT custID_pk PRIMARY KEY (CustomerID)

);
```

Output:

dbo.Customers Design View:

	Column Name	Data Type	Allow Nulls
🔑	CustomerID	int	☐
	FirstName	varchar(30)	☐
	LastName	varchar(30)	☑
	DateOfBirth	date	☑
	PhoneNumber	varchar(15)	☐
	Email	varchar(50)	☐

Query:

```
-- Policies Table
CREATE TABLE Policies
(
	PolicyID INT IDENTITY NOT NULL,
	PolicyName VARCHAR(50) NOT NULL,
	PolicyType VARCHAR(50) NOT NULL,
	PremiumAmount DECIMAL(10,2),
	DurationYears INT
	CONSTRAINT PK_policyID PRIMARY KEY (PolicyID)

);
```

Output:

dbo.Policies Design View:

	Column Name	Data Type	Allow Nulls
🔑	PolicyID	int	☐
	PolicyName	varchar(50)	☐
	PolicyType	varchar(50)	☐
	PremiumAmount	decimal(10, 2)	☑
	DurationYears	int	☑

Query:

```
-- Customers Table
CREATE TABLE Agents
(
	AgentID INT IDENTITY NOT NULL,
	AgentName VARCHAR(50),
	Phone VARCHAR(15),
	City VARCHAR(50)
	CONSTRAINT PK_agentID PRIMARY KEY (AgentID)

);
```

Output:

dbo.Agents Design View:

	Column Name	Data Type	Allow Nulls
🔑	AgentID	int	☐
	AgentName	varchar(50)	☑
	Phone	varchar(15)	☑
	City	varchar(50)	☑

Query:

```
-- PolicyAssignments Table
CREATE TABLE PolicyAssignments
(
	AssignmentID INT NOT NULL IDENTITY,
	CustomerID INT NOT NULL,
	PolicyID INT NOT NULL,
	AgentID INT NOT NULL,
	StartDate DATETIME DEFAULT GetDate(),
	EndDate DATETIME,
	CONSTRAINT PK_asgnmtID PRIMARY KEY (AssignmentID),
	CONSTRAINT FK_custID FOREIGN KEY (CustomerID) REFERENCES
	Customers(CustomerID),
	CONSTRAINT FK_policyID FOREIGN KEY (PolicyID) REFERENCES
	Policies(PolicyID),
	CONSTRAINT FK_AgentID FOREIGN KEY (AgentID) REFERENCES Agents(AgentID)
);
```

Output:

dbo.PolicyAssignments Design View:

	Column Name	Data Type	Allow Nulls
🔑	AssignmentID	int	☐
	CustomerID	int	☐
	PolicyID	int	☐
	AgentID	int	☐
	StartDate	datetime	☑
	EndDate	datetime	☑

Query:

```
-- Claims Table
CREATE TABLE Claims
(
	ClaimID INT IDENTITY NOT NULL,
	AssignmentID INT NOT NULL,
	ClaimDate DATETIME NOT NULL DEFAULT GetDate(),
	ClaimAmount DECIMAL(10,2),
	ClaimStatus VARCHAR(30)
	CONSTRAINT PK_claimID PRIMARY KEY (ClaimID),
	CONSTRAINT FK_asgnmtID FOREIGN KEY (AssignmentID) REFERENCES
	PolicyAssignments(AssignmentID)

);
```

Output:

dbo.Claims Design View:

	Column Name	Data Type	Allow Nulls
🔑	ClaimID	int	☐
	AssignmentID	int	☐
	ClaimDate	datetime	☐
	ClaimAmount	decimal(10, 2)	☑
	ClaimStatus	varchar(30)	☑

3. Insert commands for all tables.

Query:

```
-- Customers Data Insertion
INSERT INTO Customers (FirstName, LastName, DateOfBirth, PhoneNumber, Email)
VALUES
('Amit', 'Sharma', '1998-05-12', '9876543210', 'amit@gmail.com'),
('Neha', 'Verma', '2003-07-20', '9876543211', 'neha@gmail.com'),
('Ravi', 'Kumar', '1995-03-15', '9876543212', 'ravi@gmail.com'),
('Sneha', 'Patel', '2001-01-10', '9876543213', 'sneha@gmail.com'),
('Arjun', 'Mehta', '2005-11-25', '9876543214', 'arjun@gmail.com'),
('Kiran', 'Rao', '1988-06-18', '9876543215', 'kiran@gmail.com');
```

Output:

Results | Messages

	CustomerID	FirstName	LastName	DateOfBirth	PhoneNumber	Email
1	1	Amit	Sharma	1998-05-12	9876543210	amit@gmail.com
2	2	Neha	Verma	2003-07-20	9876543211	neha@gmail.com
3	3	Ravi	Kumar	1995-03-15	9876543212	ravi@gmail.com
4	4	Sneha	Patel	2001-01-10	9876543213	sneha@gmail.com
5	5	Arjun	Mehta	2005-11-25	9876543214	arjun@gmail.com
6	6	Kiran	Rao	1988-06-18	9876543215	kiran@gmail.com

Query:

```
-- Policies Data Insertion
INSERT INTO Policies (PolicyName, PolicyType, PremiumAmount, DurationYears)
VALUES
('Life Shield', 'Life', 15000, 10),
('Health Plus', 'Health', 12000, 1),
('Motor Secure', 'Motor', 8000, 1),
('Health Gold', 'Health', 20000, 2),
('Life Premium', 'Life', 25000, 15);
```

Output:

Results | Messages

	PolicyID	PolicyName	PolicyType	PremiumAmount	DurationYears
1	1	Life Shield	Life	15000.00	10
2	2	Health Plus	Health	12000.00	1
3	3	Motor Secure	Motor	8000.00	1
4	4	Health Gold	Health	20000.00	2
5	5	Life Premium	Life	25000.00	15

Query:

```
-- Agents Data Insertion
INSERT INTO Agents (AgentName, Phone, City)
VALUES
('Rajesh', '9000011111', 'Delhi'),
('Anita', '9000011112', 'Mumbai'),
('Suresh', '9000011113', 'Bangalore'),
('Kavya', '9000011114', 'Chennai'),
('Mahesh', '9000011115', 'Jaipur');
```

Output:

Results Messages

	AgentID	AgentName	Phone	City
1	1	Rajesh	9000011111	Delhi
2	2	Anita	9000011112	Mumbai
3	3	Suresh	9000011113	Bangalore
4	4	Kavya	9000011114	Chennai
5	5	Mahesh	9000011115	Jaipur

Query:

```
-- PolicyAssignments Data Insertion
INSERT INTO PolicyAssignments (CustomerID, PolicyID, AgentID, StartDate,
EndDate)
VALUES
(1, 1, 1, '2020-01-01', '2030-01-01'),
(2, 2, 2, '2022-06-01', '2023-06-01'),
(3, 3, 3, '2021-03-01', '2022-03-01'),
(4, 4, 1, '2023-01-01', '2025-01-01'),
(5, 2, 4, '2024-02-01', NULL),
(6, 5, 5, '2019-05-01', '2034-05-01');
```

Output:

Results Messages

	AssignmentID	CustomerID	PolicyID	AgentID	StartDate	EndDate
1	1	1	1	1	2020-01-01 00:00:00.000	2030-01-01 00:00:00.000
2	2	2	2	2	2022-06-01 00:00:00.000	2023-06-01 00:00:00.000
3	3	3	3	3	2021-03-01 00:00:00.000	2022-03-01 00:00:00.000
4	4	4	4	1	2023-01-01 00:00:00.000	2025-01-01 00:00:00.000
5	5	5	2	4	2024-02-01 00:00:00.000	NULL
6	6	6	5	5	2019-05-01 00:00:00.000	2034-05-01 00:00:00.000

Query:

```
-- Claims Data Insertion
INSERT INTO Claims (AssignmentID, ClaimDate, ClaimAmount, ClaimStatus)
VALUES
(1, '2022-05-10', 30000, 'Approved'),
(1, '2023-07-15', 25000, 'Rejected'),
(2, '2023-08-20', 15000, 'Approved'),
(3, '2022-01-10', 10000, 'Rejected'),
(4, '2024-06-01', 60000, 'Approved'),
(4, '2024-07-10', 45000, 'Rejected');
```

Output:

Results | Messages

	ClaimID	AssignmentID	ClaimDate	ClaimAmount	ClaimStatus
1	1	1	2022-05-10 00:00:00.000	30000.00	Approved
2	2	1	2023-07-15 00:00:00.000	25000.00	Rejected
3	3	2	2023-08-20 00:00:00.000	15000.00	Approved
4	4	3	2022-01-10 00:00:00.000	10000.00	Rejected
5	5	4	2024-06-01 00:00:00.000	60000.00	Approved
6	6	4	2024-07-10 00:00:00.000	45000.00	Rejected

4. Select commands

1. View all records Customers table.

Query:

```
SELECT CustomerID, FirstName, LastName, DateOfBirth, PhoneNumber, Email
FROM Customers;
```

Output:

Results | Messages

	CustomerID	FirstName	LastName	DateOfBirth	PhoneNumber	Email
1	1	Amit	Sharma	1998-05-12	9876543210	amit@gmail.com
2	2	Neha	Verma	2003-07-20	9876543211	neha@gmail.com
3	3	Ravi	Kumar	1995-03-15	9876543212	ravi@gmail.com
4	4	Sneha	Patel	2001-01-10	9876543213	sneha@gmail.com
5	5	Arjun	Mehta	2005-11-25	9876543214	arjun@gmail.com
6	6	Kiran	Rao	1988-06-18	9876543215	kiran@gmail.com

2. View all records of PolicyAssignment table with CustomerId, PolicyId, StartDate and EndDate columns only.

Query:

```
SELECT CustomerID, PolicyID, StartDate, EndDate
FROM PolicyAssignments;
```

Output:

Results | Messages

	CustomerID	PolicyID	StartDate	EndDate
1	1	1	2020-01-01 00:00:00.000	2030-01-01 00:00:00.000
2	2	2	2022-06-01 00:00:00.000	2023-06-01 00:00:00.000
3	3	3	2021-03-01 00:00:00.000	2022-03-01 00:00:00.000
4	4	4	2023-01-01 00:00:00.000	2025-01-01 00:00:00.000
5	5	2	2024-02-01 00:00:00.000	NULL
6	6	5	2019-05-01 00:00:00.000	2034-05-01 00:00:00.000

3. Display all policies of Health type.
   Query:

```
SELECT PolicyID, PolicyName, PolicyType, PremiumAmount, DurationYears
FROM Policies
WHERE PolicyType = 'Health';
```

   Output:

Results | Messages

	PolicyID	PolicyName	PolicyType	PremiumAmount	DurationYears
1	2	Health Plus	Health	12000.00	1
2	4	Health Gold	Health	20000.00	2

4. Display policies having premium amount more than 10000 and DurationYears is 1.
   Query:

```
SELECT PolicyID, PolicyName, PolicyType, PremiumAmount, DurationYears
FROM Policies
WHERE PremiumAmount > 10000 AND DurationYears = 1;
```

   Output:

Results | Messages

	PolicyID	PolicyName	PolicyType	PremiumAmount	DurationYears
1	2	Health Plus	Health	12000.00	1

5. Display unique city names from where agents belong to.
   Query:

```
SELECT DISTINCT(City)
FROM Agents;
```

   Output:

Results | Mes

	City
1	Bangalore
2	Chennai
3	Delhi
4	Jaipur
5	Mumbai

6. List policies of type Life, Health, Motor use OR clause.
   Query:

```
SELECT PolicyID, PolicyName, PolicyType, PremiumAmount, DurationYears
FROM Policies
WHERE PolicyType = 'Life' OR PolicyType = 'Motor' OR PolicyType = 'Health';
```

Output:

Results | Messages

	PolicyID	PolicyName	PolicyType	PremiumAmount	DurationYears
1	1	Life Shield	Life	15000.00	10
2	2	Health Plus	Health	12000.00	1
3	3	Motor Secure	Motor	8000.00	1
4	4	Health Gold	Health	20000.00	2
5	5	Life Premium	Life	25000.00	15

7. List policies of type Life, Health, Motor use IN operator.

Query:

```
SELECT PolicyID, PolicyName, PolicyType, PremiumAmount, DurationYears
FROM Policies
WHERE PolicyType IN ('Life', 'Motor', 'Health');
```

Output:

Results | Messages

	PolicyID	PolicyName	PolicyType	PremiumAmount	DurationYears
1	1	Life Shield	Life	15000.00	10
2	2	Health Plus	Health	12000.00	1
3	3	Motor Secure	Motor	8000.00	1
4	4	Health Gold	Health	20000.00	2
5	5	Life Premium	Life	25000.00	15

8. Display list of customers born after January 1st, 2001 and before December 31st, 2020 using >= and <= operators.

Query:

```
SELECT CustomerID, FirstName, LastName, DateOfBirth, PhoneNumber, Email
FROM Customers
WHERE DateOfBirth >= '2001-01-02' AND DateOfBirth <= '2020-12-30';
```

Output:

Results | Messages

	CustomerID	FirstName	LastName	DateOfBirth	PhoneNumber	Email
1	2	Neha	Verma	2003-07-20	9876543211	neha@gmail.com
2	4	Sneha	Patel	2001-01-10	9876543213	sneha@gmail.com
3	5	Arjun	Mehta	2005-11-25	9876543214	arjun@gmail.com

9. Display list of customers born after January 1st, 2001 and before December 31st, 2020 using between operator.

Query:

```
SELECT CustomerID, FirstName, LastName, DateOfBirth, PhoneNumber, Email
FROM Customers
WHERE DateOfBirth BETWEEN '2001-01-01' AND '2020-12-31';
```

Output:

Results | Messages

	CustomerID	FirstName	LastName	DateOfBirth	PhoneNumber	Email
1	2	Neha	Verma	2003-07-20	9876543211	neha@gmail.com
2	4	Sneha	Patel	2001-01-10	9876543213	sneha@gmail.com
3	5	Arjun	Mehta	2005-11-25	9876543214	arjun@gmail.com

10. Display claims data where claim status is Rejected.

Query:

```
SELECT ClaimID, ClaimDate, ClaimAmount, ClaimStatus
FROM Claims
WHERE ClaimStatus = 'Rejected';
```

Output:

Results | Messages

	ClaimID	ClaimDate	ClaimAmount	ClaimStatus
1	2	2023-07-15 00:00:00.000	25000.00	Rejected
2	4	2022-01-10 00:00:00.000	10000.00	Rejected
3	6	2024-07-10 00:00:00.000	45000.00	Rejected

11. Display records of Agents who stay in a city whose second letter is 'a'.

Query:

```
SELECT AgentID, AgentName, City, Phone
FROM Agents
WHERE LOWER(City) LIKE '_a%';
```

Output:

Results | Messages

	AgentID	AgentName	City	Phone
1	3	Suresh	Bangalore	9000011113
2	5	Mahesh	Jaipur	9000011115

12. Display highest and lowest claimAmount from Claims table.

Query:

```
SELECT MAX(claimAmount) AS [Highest ClaimAmount], MIN(claimAmount) AS [Lowest
ClaimAmount]
FROM Claims;
```

Output:

Results | Messages

	Highest ClaimAmount	Lowest ClaimAmount
1	60000.00	10000.00

13. Display latest claim record.

Query:

```
SELECT TOP 1 ClaimID, ClaimDate, ClaimAmount, ClaimStatus
FROM Claims
ORDER BY ClaimDate DESC;
```

Output:

Results | Messages

	ClaimID	ClaimDate	ClaimAmount	ClaimStatus
1	6	2024-07-10 00:00:00.000	45000.00	Rejected

14. Increase premium amount to 10% for all health insurance policies.

Query:

```
UPDATE Policies
SET PremiumAmount = (1.10 * PremiumAmount)
WHERE PolicyType = 'Health';
```

Output:

Results | Messages

	PolicyID	PolicyName	PolicyType	PremiumAmount	DurationYears
1	1	Life Shield	Life	15000.00	10
2	2	Health Plus	Health	13200.00	1
3	3	Motor Secure	Motor	8000.00	1
4	4	Health Gold	Health	22000.00	2
5	5	Life Premium	Life	25000.00	15

15. Delete the record of PolicyAssignments whose EndDate is before today's date.

Query:

```
DELETE FROM Claims
WHERE AssignmentID IN (
    SELECT AssignmentID
    FROM PolicyAssignments
    WHERE EndDate < CAST(GETDATE() AS DATE)
);

DELETE FROM PolicyAssignments
WHERE EndDate < CAST(GETDATE() AS DATE);
```

Output:

Results | Messages

	AssignmentID	CustomerID	PolicyID	AgentID	StartDate	EndDate
1	1	1	1	1	2020-01-01 00:00:00.000	2030-01-01 00:00:00.000
2	5	5	2	4	2024-02-01 00:00:00.000	NULL
3	6	6	5	5	2019-05-01 00:00:00.000	2034-05-01 00:00:00.000

16. Display no of claims rejected.

Query:

```
SELECT COUNT(*) AS [No Of Claims Rejected]
FROM Claims
WHERE ClaimStatus = 'Rejected';
```

Output:

Results | Messages

	No Of Claims Rejected
1	1

17. Display PolicyId, PolicyName, PremiumAmount along with computed fields not in table à 6% LocalTaxes, PremiumAmountWithTax and MonthlyPremiumAmount considering PremiumAmount is Annual.

Query:

```
SELECT PolicyID, PolicyName, PremiumAmount,
(PremiumAmount * 1.06) AS [PremiumAmountWithTax],((PremiumAmount * 1.06) / 12)
AS [MonthlyPremiumAmount]
FROM Policies;
```

Output:

Results | Messages

	PolicyID	PolicyName	PremiumAmount	PremiumAmountWithTax	MonthlyPremiumAmount
1	1	Life Shield	15000.00	15900.0000	1325.0000000
2	2	Health Plus	13200.00	13992.0000	1166.0000000
3	3	Motor Secure	8000.00	8480.0000	706.6666666
4	4	Health Gold	22000.00	23320.0000	1943.3333333
5	5	Life Premium	25000.00	26500.0000	2208.3333333

18. Write a command to add Address and City Columns in the Customers table.

Query:

```
ALTER TABLE Customers
ADD Address VARCHAR(50), City VARCHAR(50);
```

Output:

Column Name	Data Type	Allow Nulls
CustomerID	int	☐
FirstName	varchar(30)	☐
LastName	varchar(30)	☑
DateOfBirth	date	☑
PhoneNumber	varchar(15)	☐
Email	varchar(50)	☐
Address	varchar(50)	☑
City	varchar(50)	☑

19. Write a command to add a new column named DevOfId (DevelopmentOfficerId) in an existing Agents table.

Query:

```
ALTER TABLE Agents
ADD DevOfID INT;
```

Output:

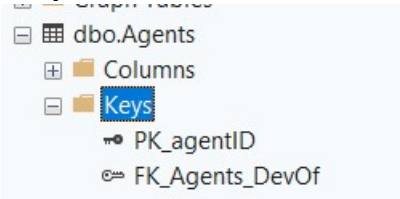
	Column Name	Data Type	Allow Nulls
🔑	AgentID	int	☐
	AgentName	varchar(50)	☑
	Phone	varchar(15)	☑
	City	varchar(50)	☑
	DevOfID	int	☑

20. Write command to make the above DevOfId as a recursive foreign key to AgentId as Parent.

Query:

```
ALTER TABLE Agents
ADD CONSTRAINT FK_Agents_DevOf
FOREIGN KEY (DevOfID) REFERENCES Agents(AgentID);
```

Output:



5. Queries using Joins, Group By, Having etc.

1. List all Policies for a CustomerId 5.

Query:

```
SELECT pa.CustomerID, p.PolicyID, p.PolicyName, p.PolicyType, p.PremiumAmount
FROM Policies p
JOIN PolicyAssignments pa
ON p.PolicyID = pa.PolicyID
WHERE pa.CustomerID = 5;
```

Output:

Results | Messages

	CustomerID	PolicyID	PolicyName	PolicyType	PremiumAmount
1	5	2	Health Plus	Health	13200.00

2. View all customers with their policies.

Query:

```
SELECT c.FirstName, c.LastName, c.PhoneNumber, c.Email,
p.PolicyID, p.PolicyName, p.PolicyType, p.PremiumAmount
FROM Customers c
LEFT JOIN PolicyAssignments pa
      ON c.CustomerID = pa.CustomerID
LEFT JOIN Policies p
      ON p.PolicyID = pa.PolicyID;
```

Output:

Results | Messages

	FirstName	LastName	PhoneNumber	Email	PolicyID	PolicyName	PolicyType	PremiumAmount
1	Amit	Sharma	9876543210	amit@gmail.com	1	Life Shield	Life	15000.00
2	Neha	Verma	9876543211	neha@gmail.com	NULL	NULL	NULL	NULL
3	Ravi	Kumar	9876543212	ravi@gmail.com	NULL	NULL	NULL	NULL
4	Sneha	Patel	9876543213	sneha@gmail.com	NULL	NULL	NULL	NULL
5	Arjun	Mehta	9876543214	arjun@gmail.com	2	Health Plus	Health	13200.00
6	Kiran	Rao	9876543215	kiran@gmail.com	5	Life Premium	Life	25000.00

3. View claims with customer name.

Query:

```
SELECT CONCAT(c.FirstName, ' ', c.LastName) AS [Customer Name],
cl.ClaimID, cl.ClaimDate, cl.ClaimAmount, cl.ClaimStatus
FROM Claims cl
LEFT JOIN PolicyAssignments pa
      ON cl.AssignmentID = pa.AssignmentID
LEFT JOIN Customers c
      ON c.CustomerID = pa.CustomerID;
```

Output:

Results | Messages

	Customer Name	ClaimID	ClaimDate	ClaimAmount	ClaimStatus
1	Amit Sharma	1	2022-05-10 00:00:00.000	30000.00	Approved
2	Amit Sharma	2	2023-07-15 00:00:00.000	25000.00	Rejected

4. Display FirstName, PolicyName, AgentName, StartDate and EndDate from their respective tables.

Query:

```
SELECT c.FirstName, p.PolicyName, a.AgentName, pa.StartDate, pa.EndDate
FROM Customers c
LEFT JOIN PolicyAssignments pa
      ON c.CustomerID = pa.CustomerID
LEFT JOIN Policies p
      ON p.PolicyID = pa.PolicyID
LEFT JOIN Agents a
      ON a.AgentID = pa.AgentID;
```

Output:

Results | Messages

	FirstName	PolicyName	AgentName	StartDate	EndDate
1	Amit	Life Shield	Rajesh	2020-01-01 00:00:00.000	2030-01-01 00:00:00.000
2	Neha	NULL	NULL	NULL	NULL
3	Ravi	NULL	NULL	NULL	NULL
4	Sneha	NULL	NULL	NULL	NULL
5	Arjun	Health Plus	Kavya	2024-02-01 00:00:00.000	NULL
6	Kiran	Life Premium	Mahesh	2019-05-01 00:00:00.000	2034-05-01 00:00:00.000

5. Display claims report with FirstName, PolicyName, ClaimAmount, ClaimStatus, and ClaimDate from their respective tables.

Query:

```
SELECT c.FirstName, p.PolicyName, cl.ClaimAmount,
cl.ClaimStatus, cl.ClaimDate
FROM Claims cl
LEFT JOIN PolicyAssignments pa
       ON cl.AssignmentID = pa.AssignmentID
LEFT JOIN Policies p
       ON p.PolicyID = pa.PolicyID
LEFT JOIN Customers c
       ON c.CustomerID = pa.CustomerID;
```

Output:

Results | Messages

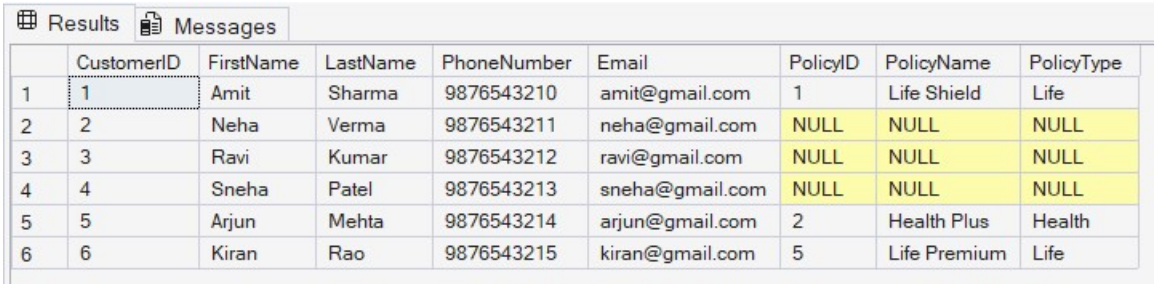
	FirstName	PolicyName	ClaimAmount	ClaimStatus	ClaimDate
1	Amit	Life Shield	30000.00	Approved	2022-05-10 00:00:00.000
2	Amit	Life Shield	25000.00	Rejected	2023-07-15 00:00:00.000

6. Display records of Customers with or without Policies.

Query:

```
SELECT c.CustomerID,
      c.FirstName,
      c.LastName,
      c.PhoneNumber,
      c.Email,
      p.PolicyID,
      p.PolicyName,
      p.PolicyType
FROM Customers c
LEFT JOIN PolicyAssignments pa
       ON c.CustomerID = pa.CustomerID
LEFT JOIN Policies p
       ON pa.PolicyID = p.PolicyID;
```

Output:



Results | Messages

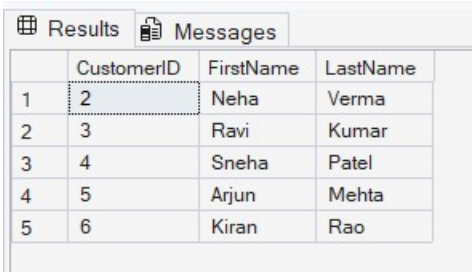
	CustomerID	FirstName	LastName	PhoneNumber	Email	PolicyID	PolicyName	PolicyType
1	1	Amit	Sharma	9876543210	amit@gmail.com	1	Life Shield	Life
2	2	Neha	Verma	9876543211	neha@gmail.com	NULL	NULL	NULL
3	3	Ravi	Kumar	9876543212	ravi@gmail.com	NULL	NULL	NULL
4	4	Sneha	Patel	9876543213	sneha@gmail.com	NULL	NULL	NULL
5	5	Arjun	Mehta	9876543214	arjun@gmail.com	2	Health Plus	Health
6	6	Kiran	Rao	9876543215	kiran@gmail.com	5	Life Premium	Life

7. Display all Customers with NO Claims.

Query:

```
SELECT c.CustomerID, c.FirstName, c.LastName
FROM Customers c
WHERE NOT EXISTS (
    SELECT 1
    FROM PolicyAssignments pa
    JOIN Claims cl
        ON pa.AssignmentID = cl.AssignmentID
    WHERE pa.CustomerID = c.CustomerID
        );
```

Output:



Results | Messages

	CustomerID	FirstName	LastName
1	2	Neha	Verma
2	3	Ravi	Kumar
3	4	Sneha	Patel
4	5	Arjun	Mehta
5	6	Kiran	Rao

8. Show CustomerName with Total Claim Amount per Customer.

Query:

```
SELECT CONCAT(c.FirstName, ' ', c.LastName) AS [Customer Name],
      SUM(cl.ClaimAmount) AS [Total Claim Amount]
FROM Customers c
LEFT JOIN PolicyAssignments pa
    ON c.CustomerID = pa.CustomerID
LEFT JOIN Claims cl
    ON cl.AssignmentID = pa.AssignmentID
GROUP BY c.CustomerID, c.FirstName, c.LastName;
```

Output:

Results | Messages

	Customer Name	Total Claim Amount
1	Amit Sharma	55000.00
2	Neha Verma	NULL
3	Ravi Kumar	NULL
4	Sneha Patel	NULL
5	Arjun Mehta	NULL
6	Kiran Rao	NULL

9. Show names and total claim amount of Customers With Claim Amount > 50000 (Use HAVING Clause).

Query:

```
SELECT CONCAT(c.FirstName, ' ', c.LastName) AS [Customer Name],
       SUM(cl.ClaimAmount) AS [Total Claim Amount]
FROM Customers c
LEFT JOIN PolicyAssignments pa
    ON c.CustomerID = pa.CustomerID
LEFT JOIN Claims cl
    ON cl.AssignmentID = pa.AssignmentID
GROUP BY c.CustomerID, c.FirstName, c.LastName
HAVING SUM(cl.ClaimAmount) > 50000;
```

Output:

Results | Messages

	Customer Name	Total Claim Amount
1	Amit Sharma	55000.00

10. Display list with Agent Wise Policy Count.

Query:

```
SELECT a.AgentID, a.AgentName, a.City, a.Phone,
COUNT(pa.AssignmentID) AS [Policy Count]
FROM Agents a
LEFT JOIN PolicyAssignments pa
    ON a.AgentID = pa.AgentID
GROUP BY a.AgentID, a.AgentName, a.City, a.Phone;
```

Output:

Results | Messages

	AgentID	AgentName	City	Phone	Policy Count
1	1	Rajesh	Delhi	9000011111	1
2	2	Anita	Mumbai	9000011112	0
3	3	Suresh	Bangalore	9000011113	0
4	4	Kavya	Chennai	9000011114	1
5	5	Mahesh	Jaipur	9000011115	1