Module 4.4 -Practical Project Assignment

1. Create Database command.

Ans:

```
create database InsuranceDB;
```

1.1 setting up the Insurance DB

```
 use InsuranceDB;
```

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

Ans:

Customers Table:

```
CREATE TABLE customers (
   customerid INT PRIMARY KEY,
   firstname VARCHAR(50) NOT NULL,
   lastname VARCHAR(50),
   DOB DATE,
   phone VARCHAR(15),
   email VARCHAR(50) UNIQUE
);
```

Agents Table:

```
CREATE TABLE agents (
   agentid INT PRIMARY KEY,
   agentname VARCHAR(50),
   phone VARCHAR(20),
   city VARCHAR(50)
);
```

Policies Table:

```
CREATE TABLE policies (
        policyid INT PRIMARY KEY,
        policyname VARCHAR(50),
        policytype VARCHAR(50),
        premiumamount MONEY,
        durationyears INT
      );
```

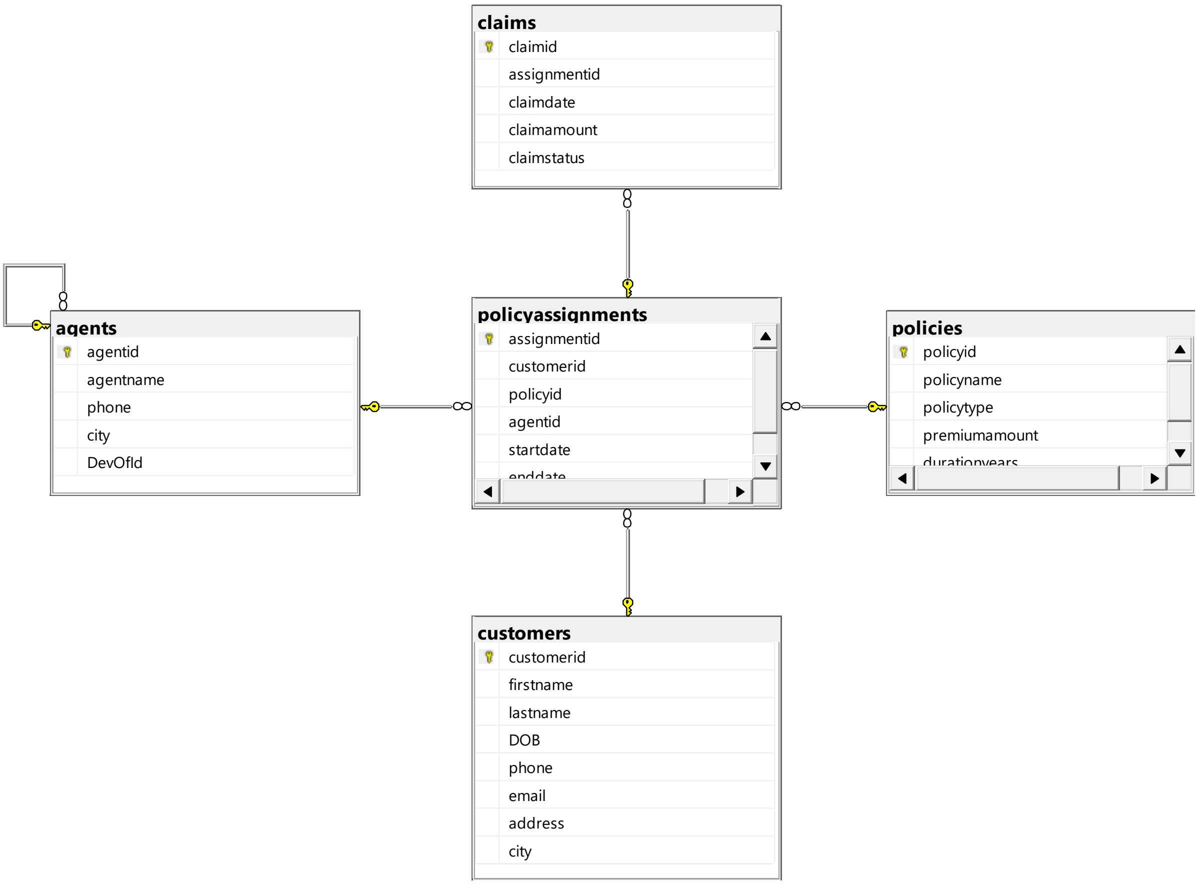
policyassignments Table:

```
CREATE TABLE policyassignments (
        assignmentid INT PRIMARY KEY,
        customerid INT,
        policyid INT,
        agentid INT,
        startdate DATE,
        enddate DATE,
      CONSTRAINT FK_policyassignments_customer FOREIGN KEY
      (customerid) REFERENCES customers(customerid),
      CONSTRAINT FK_policyassignments_policy FOREIGN KEY (policyid)
          REFERENCES policies(policyid),
      CONSTRAINT FK_policyassignments_agent FOREIGN KEY (agentid)
          REFERENCES agents(agentid),
      );
```

Claims Table:

```
CREATE TABLE claims (
    claimid INT PRIMARY KEY,
    assignmentid INT,
    claimdate DATE,
    claimamount MONEY,
    claimstatus VARCHAR(50),
    CONSTRAINT FK_claims_assignment FOREIGN KEY (assignmentid)
        REFERENCES policyassignments(assignmentid),
);
```

Schema for InsuranceDB tables



3. Insert commands for all tables.

Ans:

Customers Table:

```
INSERT INTO customers (customerid, firstname, lastname, DOB, phone, email)
        VALUES
        (1, 'Jaya', 'Prakash', '2004-08-11', '7981655855', 'jaya@gmail.com'),
        (2, 'Virat', 'Kohli', '2005-06-21', '9876543210', 'virat@gmail.com'),
        (3, 'Masoom', 'Baba', '2001-12-15', '9123456789', 'masoom@gmail.com'),
        (4, 'Rosi', 'Reddy', '2010-09-03', '9012345678', 'rosi@gmail.com'),
        (5, 'Radhika', 'Reddy', '1998-01-28', '8899776655', 'radhika@gmail.com'),
        (6, 'Raja', 'Vardhan', '2008-04-19', '9876112233', 'raja@gmail.com'),
        (7, 'Anurag', 'Vepur', '1995-11-25', '9876223344', 'anurag@gmail.com');
```

Agents Table:

```
INSERT INTO agents (agentid, agentname, phone, city)
        VALUES
        (1, 'Ranjan Kumar', '9876500001', 'Nagpur'),
        (2, 'Greeshmanth', '9876500002', 'Jaipur'),
        (3, 'Siddartha Reddy', '9876500003', 'Delhi'),
        (4, 'Pranouti', '9876500004', 'Mumbai'),
        (5, 'Sarath Sai', '9876500005', 'Patna');
```

Policies Table:

```
INSERT INTO policies (policyid, policyname, policytype, premiumamount, durationyears)
        VALUES
        (1, 'Life Secure', 'Life Insurance', 25000, 20),
        (2, 'Health Plus', 'Health Insurance', 15000, 1),
        (3, 'Car Protect', 'Motor Insurance', 12000, 1),
        (4, 'Health Gold', 'Health Insurance', 20000, 2),
        (5, 'Life Diamond', 'Life Insurance', 30000, 25);
```

Policyassignments Table:

```
INSERT INTO policyassignments (assignmentid, customerid, policyid, agentid, startdate, enddate)
        VALUES
        (1, 1, 1, 1, '2023-01-01', '2043-01-01'),
        (2, 2, 2, 2, '2022-06-15', '2023-06-15'),
        (3, 3, 3, 3, '2021-03-10', '2022-03-10'),
        (4, 4, 4, 4, '2024-01-01', '2026-01-01'),
        (5, 6, 5, 5, '2020-01-01', '2045-01-01');
```

Claims Table

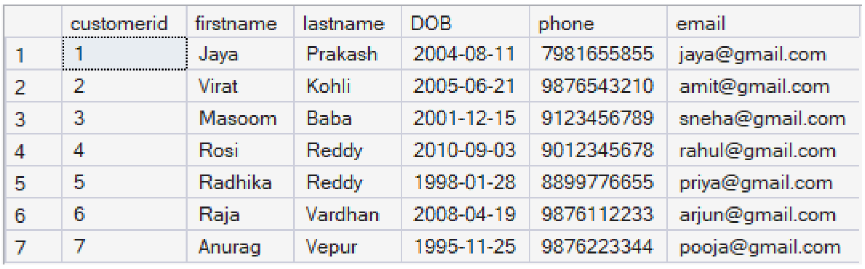
```
INSERT INTO claims (claimid, assignmentid, claimdate, claimamount, claimstatus)
        VALUES
        (1, 1, '2024-02-10', 50000, 'Approved'),
        (2, 2, '2023-11-05', 30000, 'Rejected'),
        (3, 3, '2024-07-18', 20000, 'Rejected'),
        (4, 4, '2024-09-01', 45000, 'Pending'),
        (5, 5, '2024-11-15', 60000, 'Rejected');
```

4.Select ,Update,Alter,Delete Commands

1. View all records Customers table.
Ans:

```
SELECT * FROM customers;
```

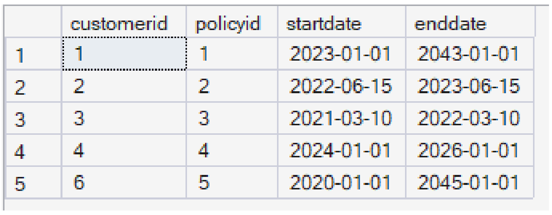
o/p:



	customerid	firstname	lastname	DOB	phone	email
1	1	Jaya	Prakash	2004-08-11	7981655855	jaya@gmail.com
2	2	Virat	Kohli	2005-06-21	9876543210	amit@gmail.com
3	3	Masoom	Baba	2001-12-15	9123456789	sneha@gmail.com
4	4	Rosi	Reddy	2010-09-03	9012345678	rahul@gmail.com
5	5	Radhika	Reddy	1998-01-28	8899776655	priya@gmail.com
6	6	Raja	Vardhan	2008-04-19	9876112233	arjun@gmail.com
7	7	Anurag	Vepur	1995-11-25	9876223344	pooja@gmail.com

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

Ans:select customerid,policyid,startdate,enddate from policyassignments;
o/p:

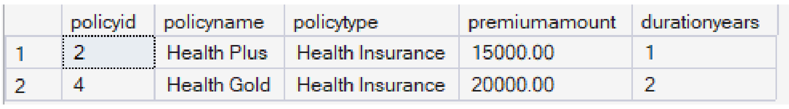


	customerid	policyid	startdate	enddate
1	1	1	2023-01-01	2043-01-01
2	2	2	2022-06-15	2023-06-15
3	3	3	2021-03-10	2022-03-10
4	4	4	2024-01-01	2026-01-01
5	6	5	2020-01-01	2045-01-01

3. Display all policies of Health type.

Ans:select *from policies where policytype='Health Insurance';

o/p:

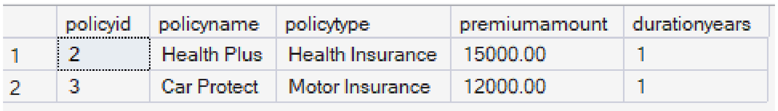


	policyid	policyname	policytype	premiumamount	durationyears
1	2	Health Plus	Health Insurance	15000.00	1
2	4	Health Gold	Health Insurance	20000.00	2

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

Ans: select *from policies where premiumamount>10000 and durationyears=1;

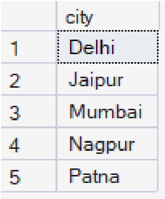
o/p:



	policyid	policyname	policytype	premiumamount	durationyears
1	2	Health Plus	Health Insurance	15000.00	1
2	3	Car Protect	Motor Insurance	12000.00	1

5. Display unique city names from where agents belong to.
Ans: select distinct city from agents;

o/p:



	city
1	Delhi
2	Jaipur
3	Mumbai
4	Nagpur
5	Patna

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

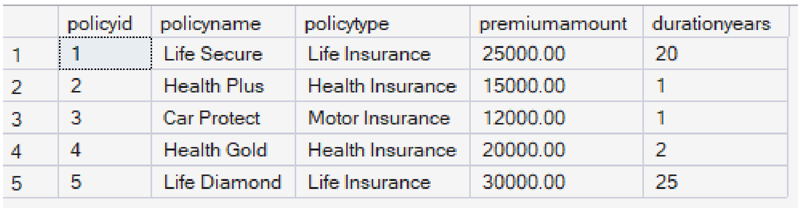
Ans: select * from Policies

where PolicyType = 'Life Insurance'

OR PolicyType = 'Health Insurance'

OR PolicyType = 'Motor Insurance';

o/p:

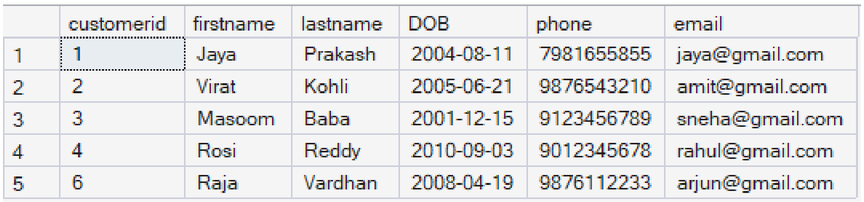


	policyid	policyname	policytype	premiumamount	durationyears
1	1	Life Secure	Life Insurance	25000.00	20
2	2	Health Plus	Health Insurance	15000.00	1
3	3	Car Protect	Motor Insurance	12000.00	1
4	4	Health Gold	Health Insurance	20000.00	2
5	5	Life Diamond	Life Insurance	30000.00	25

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

Ans: select * from customers where DOB between '2001-01-01' and '2020-12-31';

o/p:



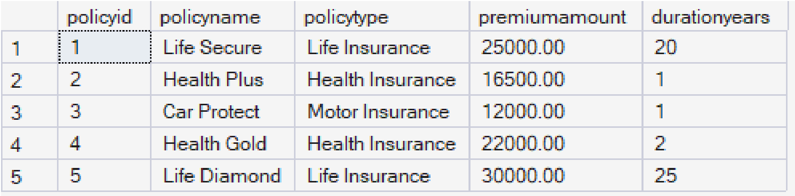
	customerid	firstname	lastname	DOB	phone	email
1	1	Jaya	Prakash	2004-08-11	7981655855	jaya@gmail.com
2	2	Virat	Kohli	2005-06-21	9876543210	amit@gmail.com
3	3	Masoom	Baba	2001-12-15	9123456789	sneha@gmail.com
4	4	Rosi	Reddy	2010-09-03	9012345678	rahul@gmail.com
5	6	Raja	Vardhan	2008-04-19	9876112233	arjun@gmail.com

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

Ans: update policies

```
set premiumamount=premiumamount*1.10
where policytype='Health Insurance';
```

o/p:



	policyid	policyname	policytype	premiumamount	durationyears
1	1	Life Secure	Life Insurance	25000.00	20
2	2	Health Plus	Health Insurance	16500.00	1
3	3	Car Protect	Motor Insurance	12000.00	1
4	4	Health Gold	Health Insurance	22000.00	2
5	5	Life Diamond	Life Insurance	30000.00	25

9. Delete the records from the PolicyAssignments table where the EndDate is earlier than today’s date.

Ans:DELETE FROM PolicyAssignments

```
WHERE EndDate < CAST(GETDATE() AS DATE);
```

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

Ans: ALTER TABLE customers

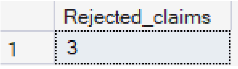
```
ADD address VARCHAR(50);

ALTER TABLE customers
ADD city VARCHAR(50);
```

5.Aggregate Functions:

11. Display no of claims rejected.

Ans: select count(*) as Rejected_claims from claims where claimstatus='Rejected';

o/p:



	Rejected_claims
1	3

12. Calculate the **total premium amount** of all policies

Ans:SELECT SUM(premiumamount) AS total_premium

FROM policies;

o/p:

13. Find the **average premium amount** of policies
Ans: SELECT AVG(premiumamount) AS avg_premium

FROM policies;
o/p:

14. Display the **highest claim amount** filed.
Ans: SELECT MAX(claimamount) AS highest_claim

FROM claims;
o/p:

15. Display the **lowest claim amount** filed.
Ans: SELECT MIN(claimamount) AS lowest_claim

FROM claims;
o/p:

6.Date and Time functions:

16. Display all policies that **expired before today**

Ans: SELECT * FROM policyassignments

```
WHERE enddate < CURRENT_DATE;
```

17. Find the **age of each customer**

Ans: SELECT firstname,

```
    EXTRACT(YEAR FROM CURRENT_DATE) - EXTRACT(YEAR FROM
DOB) AS age  FROM customers;
```

18.Display all claims filed in the year 2024

Ans: SELECT * FROM claims

```
WHERE EXTRACT(YEAR FROM claimdate) = 2024;
```

19. Display claims made in the **last 6 months**

Ans: SELECT * FROM claims

```
WHERE claimdate >= CURRENT_DATE - INTERVAL '6 months';
```

20. Display all claims filed **today**.

Ans: SELECT * FROM claims

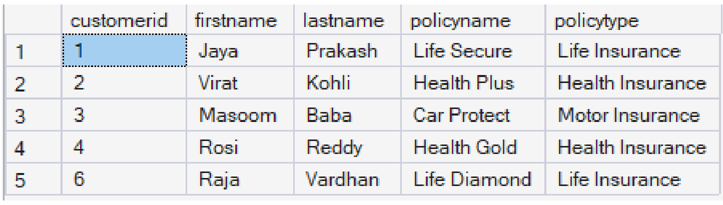
```
WHERE DATE(claimdate) = CURRENT_DATE;
```

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

1.View all customers along with their policy details..

```
Ans:   Select
             c.customerid,
             c.firstname,
             c.lastname,
             p.policyname,
             p.policytype
         From customers c
         INNER JOIN policyassignments pa
             ON c.customerid = pa.customerid
         INNER JOIN policies p
             ON pa.policyid = p.policyid;
```

o/p:

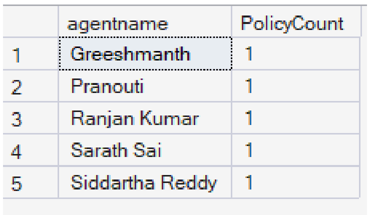


	customerid	firstname	lastname	policyname	policytype
1	1	Jaya	Prakash	Life Secure	Life Insurance
2	2	Virat	Kohli	Health Plus	Health Insurance
3	3	Masoom	Baba	Car Protect	Motor Insurance
4	4	Rosi	Reddy	Health Gold	Health Insurance
5	6	Raja	Vardhan	Life Diamond	Life Insurance

2. Display list with Agent Wise Policy Count.

```
Ans: SELECT
   a.agentname,
   COUNT(pa.policyid) AS PolicyCount
FROM agents a
LEFT JOIN policyassignments pa
   ON a.agentid = pa.agentid
GROUP BY a.agentname;
```

o/p:

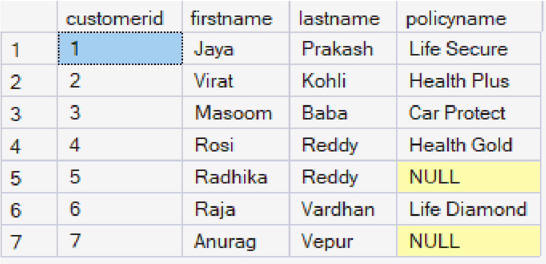


	agentname	PolicyCount
1	Greeshmanth	1
2	Pranouti	1
3	Ranjan Kumar	1
4	Sarath Sai	1
5	Siddartha Reddy	1

3. Display records of Customers with or without Policies.

```
Ans: SELECT
    c.customerid,
    c.firstname,
    c.lastname,
    p.policyname
FROM customers c
LEFT JOIN policyassignments pa
    ON c.customerid = pa.customerid
LEFT JOIN policies p
ON pa.policyid = p.policyid;
```

o/p:

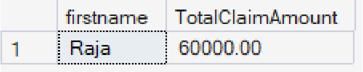


	customerid	firstname	lastname	policyname
1	1	Jaya	Prakash	Life Secure
2	2	Virat	Kohli	Health Plus
3	3	Masoom	Baba	Car Protect
4	4	Rosi	Reddy	Health Gold
5	5	Radhika	Reddy	NULL
6	6	Raja	Vardhan	Life Diamond
7	7	Anurag	Vepur	NULL

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

```
Ans: SELECT
    c.firstname,
    SUM(cl.claimamount) AS TotalClaimAmount
FROM customers c
INNER JOIN policyassignments pa
    ON c.customerid = pa.customerid
INNER JOIN claims cl
    ON pa.assignmentid = cl.assignmentid
GROUP BY c.firstname
HAVING SUM(cl.claimamount) > 50000;
```

o/p:



	firstname	TotalClaimAmount
1	Raja	60000.00

5. View all policies along with customers (include policies even if not assigned to any customer).

```
Ans: SELECT
    c.firstname,
    c.lastname,
    p.policyname,
    p.policytype
FROM customers c
RIGHT JOIN policyassignments pa
    ON c.customerid = pa.customerid
RIGHT JOIN policies p
    ON pa.policyid = p.policyid;
```

6. View all customers and all policies, including unmatched records from both tables.

```
Ans: SELECT
    c.firstname,
    c.lastname,
    p.policyname,
    p.policytype
FROM customers c
FULL OUTER JOIN policyassignments pa
    ON c.customerid = pa.customerid
FULL OUTER JOIN policies p
    ON pa.policyid = p.policyid;
```

8.subqueries

1. Customers who have taken **more than one policy**

```
Ans: SELECT firstname, lastname
FROM customers
WHERE customerid IN (
    SELECT customerid
    FROM policyassignments
    GROUP BY customerid
    HAVING COUNT(policyid) > 1 );
```

2. Policies whose **premium is higher than the average premium**

```
Ans: SELECT policyname, premiumamount
FROM policies
WHERE premiumamount > (
    SELECT AVG(premiumamount)
    FROM policies );
```

3. Customers who have **not made any claims**

```
Ans: SELECT firstname, lastname
FROM customers
WHERE customerid NOT IN (
    SELECT pa.customerid
    FROM policyassignments pa
    JOIN claims c
        ON pa.assignmentid = c.assignmentid
);
```

4. Agents who have handled **at least one rejected claim**

```
Ans: SELECT agentname
FROM agents
WHERE agentid IN (
    SELECT pa.agentid
    FROM policyassignments pa
    JOIN claims c
        ON pa.assignmentid = c.assignmentid
    WHERE c.claimstatus = 'Rejected'
);
```

5. Customers whose policy **has already expired**

```
Ans:
SELECT firstname, lastname
FROM customers
WHERE customerid IN (
    SELECT customerid
    FROM policyassignments
    WHERE enddate < GETDATE() );
```

9.Set Operations

1. List **all unique customer IDs** who either **have a policy** or **have made a claim**.
Ans:

```
SELECT customerid
FROM policyassignments
UNION
SELECT pa.customerid
FROM claims c
JOIN policyassignments pa
ON c.assignmentid = pa.assignmentid;
```

2. List **all customer IDs** who have policies and claims **including duplicates**.

Ans:

```
SELECT customerid
FROM policyassignments
UNION
SELECT pa.customerid
FROM claims c
JOIN policyassignments pa
ON c.assignmentid = pa.assignmentid;
```

3. Find customers who **have policies AND have filed claims**.
Ans: SELECT customerid

```
FROM policyassignments
INTERSECT
SELECT pa.customerid
FROM claims c
JOIN policyassignments pa
ON c.assignmentid = pa.assignmentid;
```

4. Find customers who **have policies but NO claims**.
Ans:

```
SELECT customerid
FROM policyassignments
EXCEPT
SELECT pa.customerid
FROM claims c
JOIN policyassignments pa
ON c.assignmentid = pa.assignmentid;
```

10.CASE Expression, ROLLUP, GROUPING(),GROUPING SETS,CUBE

1.Categorize policies based on premium amount
Ans:

```
SELECT
   policyname,
   premiumamount,
   CASE
      WHEN premiumamount < 15000 THEN 'Low Premium'
      WHEN premiumamount BETWEEN 15000 AND 25000 THEN 'Medium Premium'
      ELSE 'High Premium'
   END AS premium_category
FROM policies;
```

2. Total claim amount by status **with grand total**
Ans:

```
SELECT
   claimstatus,
   SUM(claimamount) AS total_claim_amount
FROM claims
GROUP BY ROLLUP (claimstatus);
```

3.Identify subtotal vs grand total

Ans:
```
SELECT

    policytype,

    SUM(premiumamount) AS total_premium,

    GROUPING(policytype) AS is_total

FROM policies

GROUP BY ROLLUP (policytype);
```

4. Claims total by status AND by year

Ans:
```
SELECT

    claimstatus,

    YEAR(claimdate) AS claim_year,

    SUM(claimamount) AS total_claim

FROM claims

GROUP BY GROUPING SETS (

    (claimstatus),

    (YEAR(claimdate))

);
```

5. Total claim amount by **status**, **year**, and **all combinations**

Ans:
```
SELECT

    claimstatus,

    YEAR(claimdate) AS claim_year,

    SUM(claimamount) AS total_claim_amount

FROM claims

GROUP BY CUBE (claimstatus, YEAR(claimdate));
```