Module 4.4 Practical Project Assignment

Database creation

create database insuranceDB;

Schema diagram

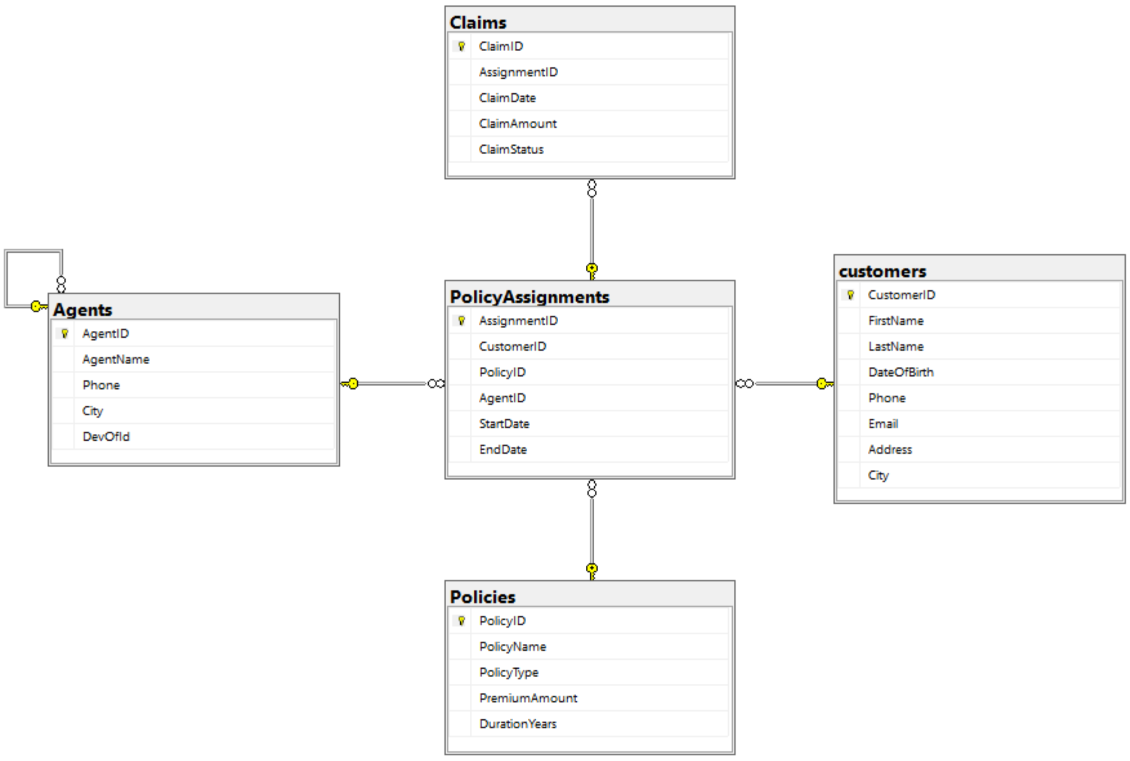

Table Creation

Customer Table Creation

```
create table customers(

CustomerID int PRIMARY KEY NOT NULL,

FirstName varchar(20),

LastName varchar(20),

DateOfBirth date,

Phone int,

Email varchar(20)

);
```

Policies Table Creation

```
create table Policies(

PolicyID int primary key NOT NULL,

PolicyName varchar(20),

PolicyType varchar(20),

PremiumAmount money,

DurationYears int

);
```

Agents Table Creation

```
create table Agents(

AgentID int PRIMARY KEY NOT NULL,

AgentName varchar(20),

Phone varchar(20) ,

City Varchar(20)

);
```

PolicyAssignments Table Creation

```
create table PolicyAssignments(

AssignmentID INT PRIMARY KEY NOT NULL,

CustomerID int FOREIGN KEY REFERENCES customers(CustomerID),

PolicyID int FOREIGN KEY REFERENCES Policies(PolicyID),

AgentID int FOREIGN KEY REFERENCES Agents(AgentID),

StartDate date,

EndDate date

);
```

Claims Table Creation

```
create table Claims(

ClaimID int PRIMARY KEY NOT NULL ,

AssignmentID int,

ClaimDate date,

ClaimAmount money,
ClaimStatus varchar(20));

alter table Claims add constraint FK_policyAssignment FOREIGN KEY (AssignmentID)  REFERENCES
PolicyAssignments(AssignmentID);
```

Values Insertion

Inserting Values into Customers

```
insert into customers values(1,'John','Smith','1998-09-1',9087654321,'john@gmail.com');

insert into customers values(2,'Alice','Josh','2000-2-10',9123456780,'alice@gmail.com');

insert into customers values(3,'Krish','Kumar','1999-3-12',8765432190,'krishkumar@gmail.com');

insert into customers values(4,'Sai','Kumar','1996-08-10',8008870875,'sai@gmail.com');

insert into customers values(5,'Sri','Priya','1995-09-25',9432156780,'priya@gmail.com');

insert into customers values(6,'Krishna',’sai’,'2004-10-12',7890654321,'krishna@gamil.com');

insert into customers values(7,'Ram','Kranthi','2004-09-21','9543216780','ram@gmail.com');
```

Inserting Values into Policies

```
insert into Policies values(101,'Health plus','Health',65000,15);

insert into Policies values(102,'Life Secure','Life',80000,10);

insert into Policies values(103,'Home Sheild','Property',45000,15);

insert into Policies values(104,'Car Protect','Vehicle',15000,5);

insert into Policies values(105,'Travel Safe','Travel',25000,4);

insert into Policies values(106,'Truck Protect','vechicle',15000,1);
```

Inserting values into Agents

```
insert into Agents values(1,'Ravi',9812345690,'Hyderabad');

insert into Agents values(2,'Sreshta',7890654321,'Chennai');

insert into Agents values(3,'Ritika',9123456780,'Pune');

insert into Agents values(4,'Bhanu',8790654321,'Mumbai');

insert into Agents values(5,'Anikha',8123456790,'Banglore');

insert into Agents values(6,'Ankith',6789054321,'Karimnagar');
```

Inserting values into PolicyAssignments

insert into PolicyAssignments values(1,1,101,1,'2024-01-01','2039-01-01');

insert into PolicyAssignments values(2,2,102,2,'2025-02-01','2030-12-31');

insert into PolicyAssignments values(3,3,103,3,'2023-01-01','2026-01-01');

insert into PolicyAssignments values(4,4,104,4,'2022-01-30','2030-12-31');

insert into PolicyAssignments values(5,5,105,5,'2020-02-01','2035-01-01');

insert into PolicyAssignments values(6,6,106,6,'2024-01-01','2025-01-01');

Inserting Values into Claims

insert into Claims values(1, 1, '2025-01-10', 15000, 'Pending');

insert into Claims values(2, 1, '2025-02-15', 8000, 'Approved');

insert into Claims values(3, 2, '2025-03-05', 12000, 'Rejected');

insert into Claims values(4, 2, '2025-03-05', 20000, 'Approved');

insert into Claims values(5, 3, '2025-06-05', 20000, 'Pending');

insert into Claims values(6, 3, '2025-06-29', 25000, 'Approved');

STRING FUNCTIONS

1. Display customer full name by concatenating FirstName and LastName.

select CONCAT(FirstName,' ' ,LastName) from customers;

2. Find customers whose FirstName starts with 'A'.

select FirstName,LastName from customers where FirstName like 'A%';

3. Display LastName in UPPERCASE.

select UPPER(LastName) as LastName from customers;

4. Extract last 4 characters from PolicyType.

select SUBSTRING(PolicyType,LEN(PolicyType)-3,LEN(PolicyType)) from Policies;

5. Remove leading and trailing spaces from customer FirstName.

select trim(FirstName) as FirstName from customers**;**

DATE FUNCTIONS

1. Display current system date.

select GETDATE() as current_system_date;

2. Find policies that started in the year 2023.

select StartDate from PolicyAssignments where year(StartDate)=2013;

3. Calculate policy duration in days

select DATEDIFF(day,StartDate,EndDate) as duration_in_days from PolicyAssignments;

4. Find customers whose birthday is in the current month.

select FirstName,LastName from customers where month(DateOfBirth)=month(GETDATE());

5. Display month name from Policy StartDate.

select datename(month,StartDate) from PolicyAssignments;

SELECT QUERIES

1. View all records Customers table.

select * from customers;

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

select CustomerID, PolicyId, StartDate, EndDate from PolicyAssignments;

3. Display all policies of Health type.

select * from Policies where PolicyType='Health';

4. Display claims data where claim status is Rejected.

select * from Claims where ClaimStatus='Rejected';

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

select distinct(city) from Agents ;

OPERATORS

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

select * from Policies where PolicyType='Life' or PolicyType='Health' or PolicyType='Motor';

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

Select * from Policies where PolicyType in ('Life','Health','Motor');

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

```
select * from customers where DateOfBirth>='2001-01-01' and DateOfBirth<='2020-12-31';
```

4. Display claims data where claim status is Rejected.

```
select * from Claims where ClaimStatus='Rejected';
```

5.List claims where ClaimAmount is less than 20000.

```
select ClaimID,ClaimAmount from Claims where ClaimAmount<20000;
```

AGGREGATE FUNCTIONS

1.Display highest and lowest claimAmount from Claims table.

```
select max(ClaimAmount) as highest_claimAmount,min(ClaimAmount) as lowest_claimAmount from Claims;
```

2. Display no of claims rejected.

```
select count(ClaimID) as claims_rejected from Claims where ClaimStatus='Rejected';
```

3. Find the total number of customers.

```
select count(CustomerID) as number_of_customers from customers;
```

4. Find total ClaimAmount per ClaimStatus.

```
select  ClaimStatus,sum(ClaimAmount) as total_claimAmount from Claims group by ClaimStatus;
```

5. Count number of policies per PolicyType.

```
select PolicyType,count(PolicyID) from Policies group by PolicyType;
```

JOINS

1. List all Policies for a CustomerId 5.

```
select p.PolicyName
from Policies p
join PolicyAssignments pa on
pa.PolicyID = p.PolicyID where pa.CustomerID = 5;
```

2. View all customers with their policies.

```
select c.CustomerID,c.FirstName,c.LastName,p.PolicyName
from customers c
join PolicyAssignments pa on pa.CustomerID=c.CustomerID
join Policies p on pa.PolicyID=p.PolicyID;
```

3. View claims with customer name.

select

c.FirstName+' '+c.LastName as customer_name,cl.ClaimID,cl.ClaimDate,cl.ClaimAmount,cl.ClaimStatus from Claims cl

join PolicyAssignments pa on pa.AssignmentID=cl.AssignmentID

join customers c on pa.CustomerID=c.CustomerID;

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

select

c.Firstname,p.PolicyName,a.AgentName,pa.StartDate,pa.EndDate

from customers c join PolicyAssignments pa

on pa.CustomerID=c.CustomerID

join Policies p on pa.PolicyID=p.PolicyID

join Agents a on pa.AgentID=a.AgentID ;

5. Display list with Agent Wise Policy Count.

select

a.AgentID,a.AgentName,count(pa.PolicyID) as policy_count

from Agents a join PolicyAssignments pa

on pa.AgentID=a.AgentID

group by a.AgentID,a.AgentName;

SUBQUERIES

1.Find customers who have at least one policy.

select * from customers c

where exists

(select * from policyassignments pa where pa.customerid = c.customerid);

2. List customers who have made at least one claim.

select * from customers c

where exists

(select * from PolicyAssignments pa join Claims cl on cl.AssignmentID=pa.AssignmentID

where pa.CustomerID=c.CustomerID);

3. find policies for which at least one claims exists.

```
select * from Policies p

where exists ( select 1 from PolicyAssignments pa join Claims cl on cl.AssignmentID=pa.AssignmentID
where pa.PolicyID=p.PolicyID);
```

4. Find policies with premium greater than ANY premium of policies held by CustomerID = 101.

```
select * from policies

where premiumamount > any

(select p.premiumamount from policies p join policyassignments pa on pa.policyid = p.policyid

where pa.customerid = 101);
```

5. Find claims where claim amount is greater than ANY claim made in 2024.

```
select * from claims

where ClaimAmount> any

(select ClaimAmount from Claims where year(ClaimDate)=2024);
```

SET OPERATORS

1.Find all distinct IDs of customers and agents

```
select CustomerID from customers

UNION

select AgentID from Agents;
```

2. Find AssignmentIDs that have claims

```
select AssignmentID from Claims

INTERSECT

select AssignmentID from PolicyAssignments;
```

3. Find policies that have not been assigned to any customer:

```
select PolicyID from Policies

EXCEPT

select PolicyID from PolicyAssignments;
```

4. List all start and end dates from policy assignments

```
select StartDate as DateValue from PolicyAssignments

UNION ALL

select EndDate as DateValue from PolicyAssignments;
```

5. Find all policies whether assigned or not.

```
select PolicyID from Policies

UNION

select PolicyID from PolicyAssignments;
```

CASE QUERY

Categorize claims based on ClaimAmount as 'Small', 'Medium', or 'High'.

```
select ClaimID,  AssignmentID, ClaimAmount,

   CASE

      WHEN ClaimAmount < 5000 THEN 'Small'

      WHEN ClaimAmount BETWEEN 5000 AND 20000 THEN 'Medium'

      ELSE 'High'

   END AS ClaimCategory

from Claims;
```

ROLL UP

Show total premium amount for each policy and a grand total.

```
select

    p.PolicyID,

    sum(PremiumAmount) AS TotalPremium

from Policies p

join PolicyAssignments pa  ON p.PolicyID = pa.PolicyID

group by ROLLUP(p.PolicyID);
```

CUBE

Show total premium amount grouped by PolicyID and AgentID, including all subtotals and grand total.

```
select

   pa.PolicyID,

   pa.AgentID,

   sum(p.PremiumAmount) AS TotalPremium
```

```
from Policies p

join PolicyAssignments pa ON p.PolicyID = pa.PolicyID

group by CUBE(pa.PolicyID, pa.AgentID);
```

MERGE

Using PolicyAssignments as target and Policies as source, write a MERGE query to maintain assignment data by updating EndDate, inserting new assignments, and deleting assignments for deleted policies.

```
MERGE PolicyAssignments as t

using Policies as s

on t.PolicyID = s.PolicyID

WHEN MATCHED THEN

   UPDATE SET t.EndDate = DATEADD(year, 1, GETDATE())

WHEN NOT MATCHED BY TARGET THEN

   INSERT (AssignmentID, CustomerID, PolicyID, AgentID, StartDate, EndDate)

   VALUES (

      (SELECT ISNULL(MAX(AssignmentID), 0) + 1 FROM PolicyAssignments),

      NULL,

      s.PolicyID,

      1,

      GETDATE(),

      DATEADD(year, 1, GETDATE())

   )

WHEN NOT MATCHED BY SOURCE THEN

   DELETE;
```