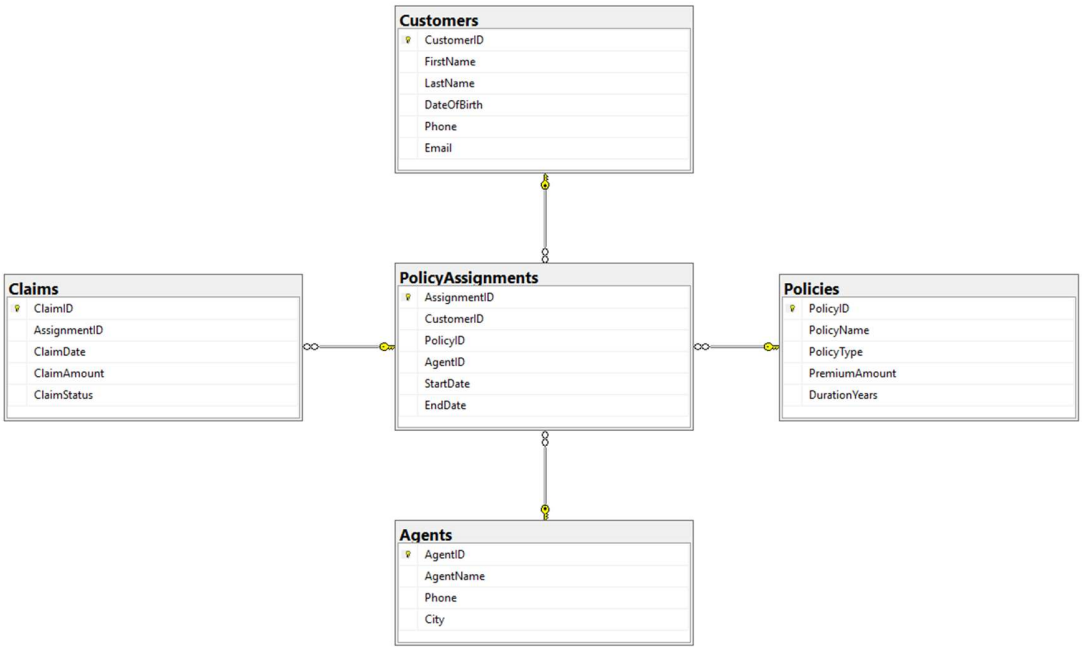
Module 4.4 Practical Project Assignment

Create Database command.

create database InsuranceDB;

use InsuranceDB;



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

```
CREATE TABLE Customers (
    CustomerID INT IDENTITY(1,1) PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DateOfBirth DATE,
    Phone VARCHAR(20),
    Email VARCHAR(50)
);
```

```
CREATE TABLE Policies (
    PolicyID INT IDENTITY(100,1) PRIMARY KEY,
    PolicyName VARCHAR(50),
    PolicyType VARCHAR(50),
    PremiumAmount MONEY,
    DurationYears INT
);

CREATE TABLE Agents (
    AgentID INT IDENTITY(200,1) PRIMARY KEY,
    AgentName VARCHAR(50),
    Phone VARCHAR(20),
    City VARCHAR(50)
);

CREATE TABLE PolicyAssignments (
    AssignmentID INT IDENTITY(308,1) PRIMARY KEY,
    CustomerID INT,
    PolicyID INT,
    AgentID INT,
    StartDate DATE,
    EndDate DATE
);
```

```
CREATE TABLE Claims (
   ClaimID INT IDENTITY(400,1) PRIMARY KEY,
   AssignmentID INT,
   ClaimDate DATE,
   ClaimAmount DECIMAL,
   ClaimStatus VARCHAR(20),
   CONSTRAINT FK_Claims_Assignment FOREIGN KEY (AssignmentID)
      REFERENCES PolicyAssignments(AssignmentID)
);

ALTER TABLE PolicyAssignments ADD CONSTRAINT FK_PA_Customer FOREIGN KEY (CustomerID)
REFERENCES Customers(CustomerID);
ALTER TABLE PolicyAssignments ADD CONSTRAINT FK_PA_Policy FOREIGN KEY (PolicyID)
REFERENCES Policies(PolicyID);
ALTER TABLE PolicyAssignments ADD CONSTRAINT FK_PA_Agent FOREIGN KEY (AgentID)
REFERENCES Agents(AgentID);
Alter table Customers add constraint UK_email Unique(Email);
```

Insert commands for all tables.

```
INSERT INTO Customers VALUES
('Vaibhav', 'Sharma', '1995-02-10', '9876543210', 'vaibhav@gmail.com'),
('Diya', 'Reddy', '1998-06-15', '9876543211', 'diya@gmail.com'),
('Karan', 'Mehta', '1992-09-21', '9876543212', 'karan@gmail.com'),
('Sneha', 'Iyer', '1997-12-05', '9876543213', 'sneha@gmail.com'),
('Rahul', 'Verma', '1990-03-18', '9876543214', 'rahul@gmail.com'),
('Ananya', 'Kapoor', '1999-08-30', '9876543215', 'ananya@gmail.com'),
```

('Vikram', 'Singh', '1988-11-25', '9876543216', 'vikram@gmail.com'),

('Pooja', 'Nair', '1996-01-14', '9876543217', 'pooja@gmail.com');

INSERT INTO Policies VALUES

('Health Secure', 'Health', 12000, 5),

('Life Protect', 'Life', 15000, 10),

('Car Shield', 'Vehicle', 8000, 3),

('Home Safe', 'Property', 10000, 7),

('Life Plus', 'Life', 5000, 1),

('Child Future', 'Education', 11000, 15),

('Senior Care', 'Health', 9000, 4),

('Business Guard', 'Business', 20000, 10);

INSERT INTO Agents VALUES

('Ramesh Kumar', '9000000001', 'Hyderabad'),

('Sita Devi', '9000000002', 'Bangalore'),

('Arjun Rao', '9000000003', 'Chennai'),

('Neha Jain', '9000000004', 'Mumbai'),

('Manoj Patel', '9000000005', 'Ahmedabad'),

('Kavya N', '9000000006', 'Kochi'),

('Suresh Yadav', '9000000007', 'Delhi'),

('Priya Malhotra', '9000000008', 'Pune');

INSERT INTO PolicyAssignments VALUES

(1, 100, 200, '2022-01-01', '2027-01-01'),

(1, 101, 200, '2023-03-01', '2033-03-01'),

(2, 102, 201, '2023-06-01', '2026-06-01'),

(3, 103, 202, '2020-09-15', '2027-09-15'),

(3, 104, 202, '2024-01-05', '2025-01-05'),

(4, 105, 203, '2019-07-20', '2034-07-20'),

(5, 106, 200, '2022-11-11', '2026-11-11'),

(6, 107, 201, '2021-05-25', '2031-05-25');

INSERT INTO Claims VALUES

(300, '2023-02-15', 3000, 'Approved'),

(300, '2024-01-10', 2500, 'Pending'),

(301, '2023-06-05', 1800, 'Approved'),

(302, '2024-04-01', 2000, 'Pending'),

(303, '2021-12-12', 7000, 'Approved'),

(304, '2022-08-20', 1500, 'Rejected'),

(305, '2024-03-18', 1200, 'Approved'),

(306, '2023-06-18', 3500, 'Rejected');

1.DML commands

1.Update claim status from pending to approved.

update Claims set claimstatus = 'approved' where claimstatus = 'pending';

2.Delete claims with claim amount less than 1000.

delete from Claims where claimamount < 1000;

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

Alter table Customers add Address varchar(50),City varchar(50);

4.Insert a new customer into customers table.

insert into customers (firstname, lastname, dateofbirth, phone, email) values ('rahul','patel','1996-04-12','9000002222','rahul@gmail.com');

SELECT COMMANDS

5.View all records Customers table.

select * from Customers;

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

select CustomerId, PolicyId, StartDate, EndDate from PolicyAssignments;

7.Display distinct policy types

select DISTINCT PolicyType FROM Policies;

8.Display latest claim record.

select top 1 * from Claims order by ClaimDate desc;

Another way:

select * from Claims order by ClaimDate desc offset 0 rows fetch next 1 rows only;

2.OPERATORS

9.Display all policies whose premium amount is greater than 10000.

select * from policies where premiumamount > 10000;

10.Display agents who are not from hyderabad.

select * from agents where city <> 'hyderabad';

11.Display policies whose duration is between 3 and 10 years.

select * from policies where durationyears between 3 and 10;

12.1List policies of type Life, Health, Motor use IN operator.

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

13.1Display list of customers born after January 1 st , 2001 and before December 31 st , 2020 using between operator.

select * from Customers where DateOfBirth between '2001-01-01' and '2020-12-31';

3.PATTERN MATCHING

14.Display records of Agents who stay in a city whose second letter is ‘a’.

select * from Agents where City like '_a%';

15.Display customers whose last name ends with letter a.

select * from customers where lastname like '%a';

16.Display agents whose name contains the string ar.

select * from agents where agentname like '%ar%';

17.Display agents whose city ends with the letter 'i'.

select * from agents where city like '%i';

4.AGGREGATE FUNCTIONS

18.display total number of customers.

select count(*) from customers;

19.Display highest and lowest claimAmount from Claims table.

select max(ClaimAmount) as HighestClaimAmount, min(ClaimAmount) as LowestClaimAmount from Claims;

20.display total premiumamount for each policytype and sort by total premium in descending order.

select policytype, sum(premiumamount) as totalpremium from policies group by policytype order by totalpremium desc;

21.Display average claimamount for each claimstatus

select claimstatus, avg(claimamount) as avgclaimamount from claims group by ClaimStatus;

22.Find policytypes where average premiumamount is greater than 6000.

select policytype from policies group by policytype having avg(premiumamount) > 6000;

23.Count number of claims per year.

select year(claimdate) as claimyear, count(*) as claimcount from claims group by year(claimdate);

5.CALCULATED FIELDS

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

Update Policies set PremiumAmount = PremiumAmount * 1.10 where PolicyType = 'Health Insurance';

25.Display claim id and add 500 as service charge.

select claimid, claimamount + 500 as adjustedamount from claims;

26.Display policy name and total premium for full duration.

select policyname, premiumamount * durationyears as totalpremium from policies;

27.Display policy name and duration in months.

select policyname, durationyears * 12 as durationmonths from policies;

28.Display PolicyId, PolicyName, PremiumAmount along with computed fields not in table à

6% LocalTaxes, PremiumAmountWithTax and MonthlyPremiumAmount considering

PremiumAmount is Annual.

select PolicyID, PolicyName, PremiumAmount, PremiumAmount * 0.06 as LocalTaxes, PremiumAmount * 1.06 as PremiumAmountWithTax, PremiumAmount / 12 as MonthlyPremiumAmount from Policies;

6.DATE FUNCTIONS

29.Calculate number of days between startdate and enddate.

select assignmentid, datediff(day, startdate, enddate) as policydays from policyassignments;

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

Delete from PolicyAssignments where EndDate < getdate();

31.Display policyassignments that started in the year 2022.

select * from policyassignments where year(startdate) = 2022;

32.calculate renewal date by adding one year to startdate.

select startdate, dateadd(year, 1, startdate) as renewaldate from policyassignments;

33.Display month from startdate.

select startdate, datename(month,startdate) as startmonth from policyassignments;

34.Find policies that expire within the next 30 days.

select * from policyassignments where enddate between getdate() and dateadd(day, 30, getdate());

7.STRING FUNCTIONS

35.Display firstname of customers in uppercase.

select upper(firstname) from customers;

36.Display agentname after removing leading and trailing spaces.

select trim(agentname) from agents;

37.Display length of agentname.

select agentname, len(agentname) from agents;

38.Display fullname of customers by combining firstname and lastname.

select concat(firstname,' ',lastname) as fullname from customers;

39.Extract domain name from email column.

select email, substring(email, charindex('@', email) + 1, len(email)) as domain from customers;

40.Replace the word 'life' with 'term life' in policytype column of policies table.

update policies set policytype = 'term life' where policytype = 'life';

8.MATH FUNCTIONS

41.Display square of durationyears.

select square(durationyears) as sq_durationyears from policies;

42.Display rounded premiumamount values.

select round(premiumamount, 2) as Rounded_amount from policies;

43.Display the floor value of premiumamount.

select policyname, floor(premiumamount) as floorpremium from policies;

44.Display absolute value of claimamount.

select claimid, abs(claimamount) as absoluteclaimamount from claims;

9.JOINS

Queries using Joins, Group By, Having etc.

45.List all Policies for a CustomerId 5.

select po.* from Policies po join PolicyAssignments pa on po.PolicyID = pa.PolicyID

where pa.CustomerID = 5;

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

select c.FirstName, po.PolicyName, a.AgentName, pa.StartDate, pa.EndDate

from PolicyAssignments pa

join Customers c on pa.CustomerID = c.CustomerID

join Policies po on pa.PolicyID = po.PolicyID

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

47.Display records of Customers with or without Policies.

select c.FirstName+' '+c.LastName as Name, p.PolicyName

from Customers c

left join PolicyAssignments pa

on c.CustomerID = pa.CustomerID

left join Policies p

on pa.PolicyID = p.PolicyID;

48.Display all Customers with NO Claims.

select distinct c.FirstName,c.LastName

from Customers c

left join PolicyAssignments pa

on c.CustomerID = pa.CustomerID

left join Claims cl

on pa.AssignmentID = cl.AssignmentID

where cl.ClaimID is null;

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

select c.FirstName,c.LastName,sum(cl.ClaimAmount) as TotalClaimAmount

from Customers c

join PolicyAssignments pa on c.CustomerID = pa.CustomerID

join Claims cl on pa.AssignmentID = cl.AssignmentID

group by c.FirstName,c.LastNameS

having sum(cl.ClaimAmount) > 50000;

10.SET OPERATORS

50.Display all customerid values from customers and policyassignments without duplicates.

select customerid from customers union select customerid from policyassignments;

51.Display all customerid values from customers and policyassignments including duplicates.

select customerid from customers union all select customerid from policyassignments;

52.Display agentid values present in agents but not assigned in policyassignments.

select agentid from agents except select agentid from policyassignments;

53.Display customerid values common in both customers and policyassignments.

select customerid from customers intersect select customerid from policyassignments;

54.Display customerid values present in customers but not in policyassignments.

select customerid from customers except select customerid from policyassignments;

11.SUB QUERIES

55.Find policies for which at least one claim exists

```
select * from policies where policyid in
(select pa.policyid
from policyassignments pa
join claims c
on pa.assignmentid = c.assignmentid);
```

56.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);
```

57.Find customers who have both policies and claims.

```
select * from customers where customerid in
(select pa.customerid
from policyassignments pa
join claims c
on pa.assignmentid = c.assignmentid);
```

58.List customers whose policy premium is less than ALL premiums of Life policies.

```
select * from customers where customerid in
(select pa.customerid from policyassignments pa
```

```
 join policies p on pa.policyid = p.policyid
where p.premiumamount < all
(select premiumamount from policies
where policytype = 'life'));
```

59.Find policies with start date later than ALL policies of CustomerID = 102

```
select * from policyassignments where startdate > all
(select startdate from policyassignments
where customerid = 102);
```

60.Find customers who have at least one policy assigned (using exists).

```
select * from customers c where exists
(select 1 from policyassignments pa
where pa.customerid = c.customerid);
```

CASE QUERY

61.Display policyname and show premium category using case statement.

```
select policyname,
case  when premiumamount > 15000 then 'high'
when premiumamount between 8000 and 15000 then 'medium'
else 'low'
end as premiumcategory from policies;
```

MERGE

62. Synchronize claims table with policyassignments using merge (update existing claims, insert missing claims, and mark unassociated claims).

```
merge claims as t using policyassignments as s
on t.assignmentid = s.assignmentid
when matched then update set t.claimstatus = 'reviewed'
```

when not matched then insert (assignmentid, claimdate, claimamount, claimstatus)

values (s.assignmentid, getdate(), 1000, 'pending')

when not matched by source then

update set t.claimstatus = 'inactive';

ROLL UP

63.Display total number of policyassignments handled by each agent along with grand total

select agentid, count(*) as totalassignments

from policyassignments

group by rollup(agentid);

CUBE

64.Display total premiumamount by policytype and durationyears with all subtotal combinations

select policytype, durationyears, sum(premiumamount) as totalpremium

from policies group by cube(policytype, durationyears);

GROUPING SETS

65.Display total premiumamount grouped by policytype and durationyears separately using grouping sets.

select policytype, durationyears, sum(premiumamount) as totalpremium

from policies

group by grouping sets ((policytype),(durationyears));