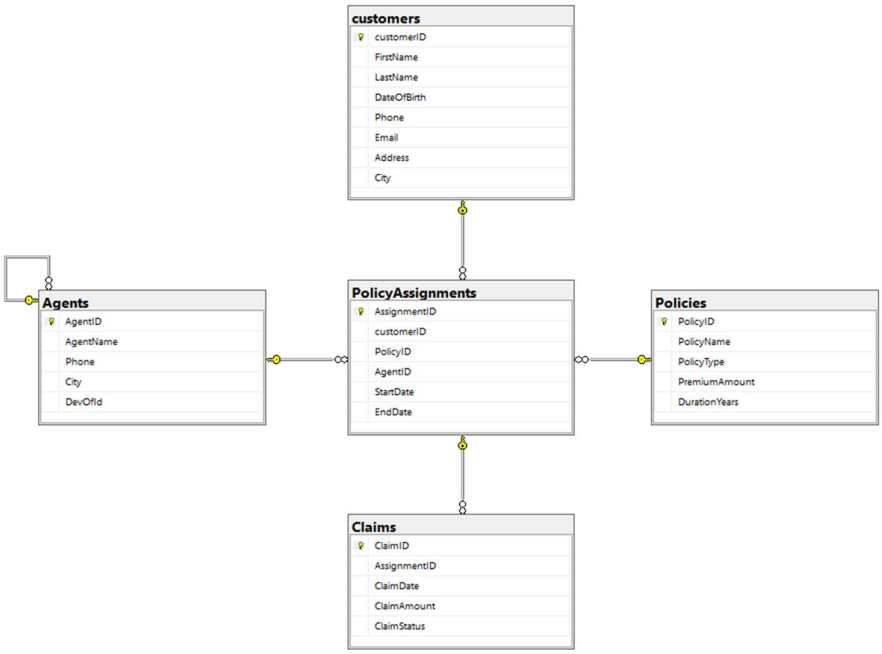
Module 4.4 Practical Project Assignment

1.Creating Database command:

Create database Insurance;

Database schema



Creating Tables:

customers

create table customers (customerID int not null PRIMARY KEY identity(1,1), FirstName varchar(50),LastName varchar(50) , DateOfBirth date, Phone varchar(50), Email varchar(50));

Policies

create table Policies (PolicyID int not null PRIMARY KEY, PolicyName varchar(50), PolicyType varchar(50) , PremiumAmount int, DurationYears int);

Agents

create table Agents(AgentID int not null PRIMARY KEY, AgentName varchar(50),Phone int, City varchar(50));

PolicyAssignments

create table PolicyAssignments (AssignmentID int not null Primary key, customerID int not null Foreign key References customers(customerID),PolicyID int not null Foreign key References

Policies(PolicyID), AgentID int not null Foreign key References Agents(AgentID), StartDate date, EndDate date);

Claims

create table Claims(ClaimID int not null PRIMARY KEY, AssignmentID int not null Foreign key References PolicyAssignments(AssignmentID),ClaimDate date, ClaimAmount int, ClaimStatus varchar(50));

Inserting Data:

customers

insert into customers values('Anand','chedapangu','2003-08-23','9573634476','anandch1119@gamil.com');

insert into customers values('Abhi','Banda','2003-12-26','7396939296','abhi7@gmail.com');

insert into customers values('Abhi','chanda','2005-12-26','7396908296','abhichanda@gmail.com');

Policies

insert into Policies values(1,'Health','General',50000,2);

insert into Policies values(2,'Term','Life Insurane',100000,5);

insert into Policies values(3,'General','Health',500000,2);

insert into Policies values(4,'Life', 'Health',100000,9);

insert into Policies values(5,'Term','Life',100000,1);

insert into Policies values(6,'Motor','Motor',500000,1);

insert into Policies values(7,'Family', 'Health',100000,1);

Agents

insert into Agents values(1,'Ram','848483210','Hyderabad');

insert into Agents values(2,'Ramesh','848903210','Hyderabad');

insert into Agents values(3,'Poojitha','9191918323','kamareddy');

insert into Agents values(4,'Srinitha','848909890','karimnagar');

PolicyAssignments

insert into PolicyAssignments values(1,4,1,1,'2023-08-23','2025-08-24');

insert into PolicyAssignments values(2,5,1,2,'2023-10-23','2026-08-24');

Claims

insert into Claims values(101,1,'2023-08-24',50000,'success');

insert into Claims values(102,2,'2023-12-24',100000,'success');

```
insert into Claims values(103,2,'2024-08-24',100000,'Rejected');

insert into Claims values(104,2,'2023-12-24',100000,'Approved');

insert into Claims values(105,1,'2024-06-24',100000,'Rejected');

insert into Claims values(106,1,'2025-02-24',100000,'Pending');
```

Queries:

1. Pattern Matching (LIKE Operator)

i. **Find customers whose FirstName starts with 'Ab'.**
```
select * from customers where FirstName like 'Ab%';
```

ii. **Find agents whose names end with "itha".**
```
select AgentName from Agents where AgentName like '%itha';
```

iii. **Get all customers whose first name starts with the letter A.**
```
select * from customers where FirstName like 'A%';
```

iv. **Display agents whose city contains the word "hyd".**
```
select * from Agents where City like '%hyd%';
```

v. **Display customers whose last name has only 5 letters.**
```
Select * from customers where LastName like '_____';
```

2. String Functions

i. **Convert all customer last names to uppercase.**
```
select upper(LastName)  from customers;
```

ii. **Find customers whose last name length is greater than 6.**
```
select * from customers where len(LastName)>6;
```

iii. **Display the first 3 characters of customer first names.**
```
select left(FirstName,3) from customers;
```

iv. **Display the last 4 characters of policy types.**
```
select Right(PolicyType,4) from Policies;
```

v. **Find the length of each policy type.**
```
select len(PolicyType) as len_of_policy_types from Policies;
```

vi. **Replace "Life Insurance" with "Term Life" in policy types.**
```
select REPLACE(PolicyType,'Life Insurane','Term Life') from Policies;
```

3.Filtering using WHERE Clause and using (IN and BETWEEN Operators)

i. **Find all agents who live in Hyderabad.**

```
select * from Agents where City='Hyderabad';
```

ii. **Display all Health policies where the premium is greater than 20000.**

```
select * from Policies where PolicyName='Health' and PremiumAmount>20000;
```

iii. **Retrieve all claims that are not approved.**

```
select * from Claims where ClaimStatus='Not Approved';
```

iv. **Display policy assignments that started after 10-Sep-2023.**

```
select * from PolicyAssignments where StartDate>'2023-09-10';
```

v. **Display policies whose type is Life Insurance or General.**

```
select * from Policies where  PolicyType IN ('Life Insurane', 'General');
```

vi. **Find all claims filed between 24-Dec-2023 and 31-Dec-2024.**

```
select * from Claims where ClaimDate between '2023-12-24' and '2024-12-31';
```

vii. **Retrieve all claims whose status is either Rejected or Pending.**

```
select * from Claims where ClaimStatus IN ('Rejected', 'Pending');
```

4. Date Functions

i. **Find all policy assignments that started more than 1 year ago.**

```
select * from PolicyAssignments where DATEDIFF(Year,StartDate,getDate() )>1;
```

ii. **Display Policy ID, start year, and end year of each policy assignment.**

```
select PolicyID,DATEPART(yyyy,StartDate) as startyear, DATEPART(yyyy,EndDate)  as endyear
from PolicyAssignments;
```

iii. **Calculate customer age from DOB.**

```
select concat(FirstName,' ',LastName) as name,DATEDIFF(year,DateofBirth,getdate()) as age
from customers;
```

iv. **Calculate policy duration in days.**

```
select PolicyName,DurationYears*365 as duration_in_days from Policies;
```

v. **Find claims filed in the last 6 months.**

```
select * from Claims where datediff(month,ClaimDate,'2024-12-20')<=6;
```

5. Aggregate Function

i. **Find the total number of customers.**

```
select count(*) from customers;
```

ii. **Find average premium amount.**
select avg(PremiumAmount) from Policies;

iii. **Find the maximum claim amount.**
select max(ClaimAmount) from Claims;

iv. **Find the minimum premium amount.**
select min(PremiumAmount) from Policies;

v. **Count number of policies per PolicyType.**
select PolicyType,count(PolicyName) from Policies group by PolicyType;

6.Joins

i. **List all Policies for a CustomerId 4.**
select Policies.PolicyID,Policies.PolicyName from Policies join PolicyAssignments on Policies.PolicyID=PolicyAssignments.PolicyID where customers.customerID=4;

ii. **View all customers with their policies.**
Select customers.customerID, customers.FirstName, customers.LastName,Policies.PolicyID, Policies.PolicyName from Policies join PolicyAssignments on Policies.PolicyID=PolicyAssignments.PolicyID join customers on PolicyAssignments.customerID=customers.customerID;

iii. **Display FirstName, PolicyName, AgentName, StartDate and EndDate from their respective tables.**
select customers.FirstName,Policies.PolicyName,Agents.AgentName, PolicyAssignments.StartDate,PolicyAssignments.EndDate
from Policies join PolicyAssignments on Policies.PolicyID=PolicyAssignments.PolicyID
join customers on PolicyAssignments.customerID=customers.customerID
join Agents on PolicyAssignments.AgentID=Agents.AgentID;

iv. **Display records of Customers with or without Policies.**
select customers.FirstName,customers.LastName,customers.DateOfBirth,customers.Email, Policies.PolicyName from customers full join PolicyAssignments on PolicyAssignments.customerID=customers.customerID
left join Policies on Policies.PolicyID=PolicyAssignments.PolicyID;

v. **Show CustomerName with Total Claim Amount per Customer.**
select customers.FirstName,customers.LastName, sum(Claims.ClaimAmount) as Total_claim_amount from customers full join PolicyAssignments on PolicyAssignments.customerID=customers.customerID left join Claims on Claims.AssignmentID=PolicyAssignments.AssignmentID group by customers.FirstName,customers.LastName;

7.SubQuery

i. **Find customers who have at least one policy.**
select * from Customers where CustomerID in (select CustomerID from PolicyAssignments);

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

```
select * from Policies where PremiumAmount> any(select PremiumAmount from Policies
join PolicyAssignments on Policies.PolicyID=PolicyAssignments.PolicyID join customers on
customers.customerID=PolicyAssignments.customerID where customers.customerID=4);
```

iii. Display customers who do not have any claims.

```
select * from customers where customerID not in (select customerID from PolicyAssignments
join Claims on PolicyAssignments.AssignmentID=Claims.AssignmentID);
```

iv. Find claims whose amount is less than ANY claim rejected.

```
select * from Claims where ClaimAmount< any(select ClaimAmount from Claims where
ClaimStatus='Rejected');
```

v. Find customers who have a policy with premium > 50,000.

```
select * from customers where customerID in (select customers.customerID from customers
join PolicyAssignments on customers.customerID=PolicyAssignments.customerID join
Policies on Policies.PolicyID=PolicyAssignments.PolicyID where
Policies.PremiumAmount>50000);
```

8. Set Operations

i. list all customer ids and agent ids with duplicates .

```
select customerID from Customers
union
select AgentID from Agents;
```

ii. list all policy ids from policies and policyassignments (unique values)

```
select PolicyID from Policies union select PolicyID from PolicyAssignments;
```

iii. find customers who are also agents

```
select customerID from Customers
intersect
select AgentID from Agents;
```

iv. find customers who are handled by any agent in policyassignments

```
select customerID from Customers
intersect select customerID from PolicyAssignments;
```

v. list all unique cities where either customers or agents operate

```
select city from Customers union select city from Agents;
```

9.Case

classify claims based on claim amount

```
select ClaimID,ClaimAmount,
    case
      when ClaimAmount >= 50000 then 'high'
      when ClaimAmount >= 20000 then 'medium'
      else 'low'
    end as ClaimCategory
from claims;
```

10. Merge

update or insert customer email details from a temporary source table

merge customers as target using tempcustomers as source on target.CustomerID = source.CustomerID when matched then update set target.Email = source.Email when not matched then insert (FirstName, LastName, DateOfBirth, Phone, Email) values (source.FirstName, source.LastName, source.DateOfBirth, source.Phone, source.Email);

11. Rollup

get total number of policies per agent including grand total

select AgentID,count(AssignmentID) as PolicyCount from PolicyAssignments group by rollup (AgentID);

12. Cube

get policy count by agent and policy including all subtotals

select AgentID, PolicyID, count(AssignmentID) as PolicyCount from PolicyAssignments group by cube (AgentID, PolicyID);