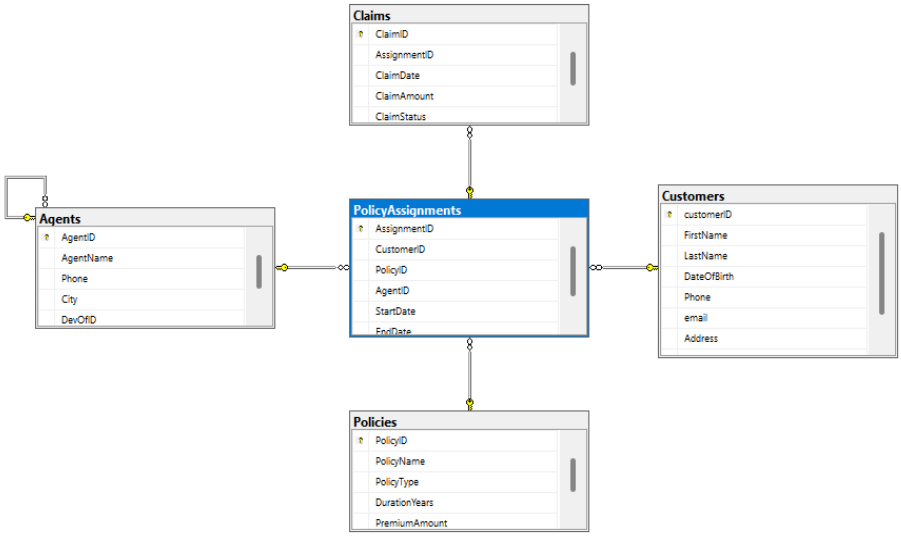
MODULE 4.4 PRACTICAL PROJECT ASSIGNMENT.

1.DB CREATION

create database insuranceDB;

use insuranceDB;

2.SCHEMA DIAGRAM



3.TABLE CREATIONS

CUSTOMERS TABLE:

```
create table Customers(
customerID int primary key,
FirstName varchar(50) not null,
LastName varchar(50),
DateOfBirth date,
Phone varchar(20),
email varchar(20) unique
);
```

POLICIES TABLE:

```
create table Policies(
PolicyID int primary key,
PolicyName varchar(50),
PolicyType varchar(50),
PremiumAccount varchar(10),
DurationYears date
);
```

AGENTS:

```
create table Agents(
AgentID int primary key,
AgentName varchar(20),
Phone varchar(20) not null,
City Varchar(20)
);
```

POLICY ASSIGNMENTS:

```
create table PolicyAssignments(
AssignmentID int primary key,
CustomerID int,
PolicyID int,
AgentID int,
StartDate date,
EndDate date,
constraint fk_policyass_custid_customers foreign key(CustomerID) references Customers(CustomerID),
constraint fk_policyass_policyid_policies foreign key(PolicyID) references Policies(PolicyID),
constraint fk_policyass_agentid_agents foreign key(AgentID) references Agents(AgentID)
);
```

CLAIMS:

```
create table Claims(
ClaimID int primary key,
AssignmentID int,
ClaimDate date,
ClaimAmount decimal,
ClaimStatus varchar(20),
constraint fk_claims_Assignmentid_assignments foreign key(AssignmentID) references PolicyAssignments(AssignmentID)
);
```

Modifying datatype from date to int(incompatible types where conversion is not possible so dropping column and adding again with new datatype):

```
alter table Policies drop column DurationYears;
alter table Policies add DurationYears int;
```

Insertions:

1.Customers

```
insert into Customers values(1,'poojitha','bejgum','2004-07-13',9347758510,'poojitha@gmail.com'),
(2,'ruchitha','bejgum','2001-12-11',9390423662,'ruchitha@gmail.com');
```

```
INSERT INTO Customers VALUES
(3, 'Aditi', 'Sharma', '1998-03-21', '8123456789', 'aditi@gmail.com'),
(4, 'Meera', 'Rao', '1985-11-30', '9988776655', 'meera@gmail.com'),
(5, 'Kabir', 'Khan', '2000-06-15', '7766554433', 'kabir@gmail.com');
```

2.Policies:

```
insert into Policies values(101,'HealthPlus','Health','PRE01',50),
(102,'Care Plus','Health','PRE02',50),
(103,'Drive Safe Plus','Vehicle','PRE03',40),
(104,'Safe Nest','Property','PRE04',100),
(105,'LifeSecure','Life','PRE05',40);
```

3.PolicyAssignments:

```
insert into PolicyAssignments values(1,1,101,1,'2025-12-28','2075-12-28'),
(2,2,102,2,'2025-12-28','2075-12-28');
```

```
insert into PolicyAssignments values
(3, 3, 105, 3, '2025-01-10', '2030-01-10'),
(4, 4, 104, 4, '2024-05-20', '2049-05-20'),
(5, 4, 101, 1, '2023-08-15', '2043-08-15');
```

4.Agents:

```
insert into Agents values(1,'Rohith','6309874810','Hyderabad');
insert into Agents values(2,'Swetha','9489928999','Nizamabad');
```

```
INSERT INTO Agents VALUES
(3, 'Arjun', '8899001122', 'Hyderabad'),
(4, 'Farah', '9900112233', 'Secunderabad');
```

5.Claims:

```
insert into Claims values(1,1,'2026-02-02',50000,'Active'),
(2,2,'2026-02-12',100000,'Active');

insert into  Claims values
(3, 3, '2025-03-01', 12000, 'PENDING'),
(4, 4, '2025-06-18', 55000, 'APPROVED'),
(5, 5, '2025-09-30', 25000, 'REJECTED');
```

```
alter table Policies drop column PremiumAccount;
alter table Policies add PremiumAmount int;

update Policies set PremiumAmount=50000 where PolicyID=101;
update Policies set PremiumAmount=20000 where PolicyID=102;
update Policies set PremiumAmount=40000 where PolicyID=103;
update Policies set PremiumAmount=80000 where PolicyID=104;
update Policies set PremiumAmount=25000 where PolicyID=105;
```

3.Queries:

I.Select:

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

Query:

select CustomerId,PolicyId,StartDate,EndDate from PolicyAssignments;

2.Display all policies of Health type.

Query:

Select * from Policies where PolicyType like 'Health%';

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

Query:

select * from Policies where Policytype like 'Life%' or PolicyType like 'Health%' or PolicyType like 'Motor%';

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

Query: select distinct city from Agents;

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

Query: select * from Customers where DateOfBirth>='2001-01-01' and DateOfBirth<='2020-12-31';

II.DATE FUNCTIONS:

1. display current system date and time

select getdate() as currentdatetime;

2.display claim date in dd-mm-yyyy format

select ClaimId, format(ClaimDate, 'dd-mm-yyyy') as claimdateformatted from Claims;

3. get policies started in the current year

select * from Policies where datepart(year, StartDate) = datepart(year, getdate());

4.find policy duration in days

select PolicyId, datediff(day, StartDate, EndDate) as policydurationdays from Policies;

5. List customers born in 1990

select * from Customers where year(dob) = 1990;

III.STRING FUNCTIONS:

1.display lastname in uppercase

select Upper(LastName) as uppcase_lastname from Customers;

2. Find customers whose LastName length is more than 6 characters.

select * from Customers where len(LastName)>6;

3.Display the first 3 characters of FirstName

select SUBSTRING(FirstName,1,3) from Customers;

4. Replace the word 'Life' with 'Term Life' in PolicyType.

select replace(PolicyType,'Life','Term Life') from Policies;

5.Remove leading and trailing spaces from customer FirstName.

select trim(FirstName) from Customers;

IV.NUMERIC FUNCTION QUESTIONS:

1.Display PremiumAmount rounded to nearest integer

```
select round(PremiumAmount,0) from Policies;
```

2.Find the square root of PremiumAmount.

```
Select sqrt(PremiumAmount) from Policies;
```

3. Display the ceiling value of ClaimAmount.

```
select ceiling(ClaimAmount) as ceil_claim from Claims;
```

4. Display the floor value of PremiumAmount.

```
select floor(PremiumAmount) as floor_premium from Policies;
```

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

```
update Policies set PremiumAmount=PremiumAmount*1.1 where PolicyType='Health';
select * from Policies where PolicyType='Health';
```

V.AGGREGATE FUNCTIONS:

1.Find the total number of customers.

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

2. Calculate total premium collected.

```
select sum(PremiumAmount) from Policies;
```

3. Find average premium amount.

```
select avg(PremiumAmount) from Policies;
```

4. Find the maximum claim amount.

```
select max(ClaimAmount) from Claims;
```

5. Find the minimum premium amount

```
select min(PremiumAmount) from Policies;.
```

VI.OPERATORS:

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

Query: select * from Policies where Policytype like 'Life%' or PolicyType like 'Health%' or PolicyType like 'Motor%';

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

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

3.Select * from Policies where PremiumAmount>10000 and DurationYears=1;

Query: select * from Policies where PremiumAmount>10000 and DurationYears=1;

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

Query: select * from Customers where DateOfBirth>='2001-01-01' and DateOfBirth<='2020-12-31';

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

Query: select * from Customers where DateOfBirth>='2001-01-01' and DateOfBirth<='2020-12-31';

VII.JOINS,GROUP BY ,HAVING

1.List all Policies for a CustomerId 4.

select * from Policies p join PolicyAssignments pl on pl.PolicyID=p.PolicyID

join Customers c on pl.CustomerID=c.customerID where c.customerID=4;

2. View all customers with their policies.

select * from customers c join PolicyAssignments pl on c.customerID=pl.CustomerID

join Policies p on pl.PolicyID=p.PolicyID;

3. View claims with customer name.

Query:

select c.FirstName,cl.ClaimID from Claims cl join PolicyAssignments pl on cl.AssignmentID=pl.AssignmentID

join Customers c on pl.CustomerID=c.customerID;

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

```
select c.FirstName,p.PolicyName,a.AgentName,pl.StartDate,pl.EndDate
from Policies p join PolicyAssignments pl on p.PolicyID=pl.PolicyID
join Customers c on c.customerID=pl.CustomerID
join Agents a on a.AgentID=pl.AgentID;
```

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

Query:

```
select p.PolicyName,c.FirstName,a.AgentName,cm.ClaimAmount,
cm.ClaimStatus,cm.ClaimDate
from Policies p join PolicyAssignments pl on p.PolicyID=pl.PolicyID
join Customers c on c.customerID=pl.CustomerID
join Agents a on a.AgentID=pl.AgentID
join Claims cm on cm.AssignmentID=pl.AssignmentID;
```

VIII.SUBQUERIES:

1. Find customers who have at least one policy

```
select * from Customers  c where exists (select 1 from PolicyAssignments p where
c.customerID=p.CustomerID);
```

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

```
select * from customers where customerID in (
select CustomerID from PolicyAssignments where AssignmentID in (
select AssignmentID from Claims));
```

3. Display agents who handle at least one policy

```
select * from Agents where AgentID in (select AgentID from PolicyAssignments);
```

4. Find agents who serve customers from the same city.

select * from agents where agentid in (select agentid from policyassignments where customerid in (select customerid from customers where city = (select city from agents a where a.agentid = policyassignments.agentid)));

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

select * from Customers where customerID in (

select customerID from PolicyAssignments where PolicyID in (

select PolicyID from Policies where PremiumAmount < all (

select PremiumAmount from Policies where PolicyType = 'Life')));

IX.SETOPERATIONS:

1.list all customer ids and agent ids without duplicates .

select customerID from Customers

union

select AgentID from Agents;

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

select PolicyID from Policies union select PolicyID from PolicyAssignments;

3.find customers who are also agents

select customerID from Customers

intersect

select AgentID from Agents;

4.find customers who are not handled by any agent in policyassignments

select customerID from Customers

except select CustomerID from PolicyAssignments;

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

select city from Customers union select city from Agents;

X.CASEELSE

1.display policy types but rename them: if policy is life show term life, if vehicle show auto insurance, else show other

```
select PolicyID, PolicyName,
case
   when PolicyType = 'life' then 'term life'
   when PolicyType = 'vehicle' then 'auto insurance'
   else 'other'
end as updated_policytype
from Policies;
```

XI.ROLLUP

show city wise total policies with rollup.

```
select City,count(AssignmentID) as total
from Agents ag join PolicyAssignments pa on ag.agentid=pa.agentid
group by rollup(City);
```

XII.MERGE

update claimstatus using merge when claimid matches.

```
Merge Into Claims T
Using (Values (1,'Closed')) S(ClaimID,ClaimStatus)
On T.ClaimID=S.ClaimID
When Matched Then Update Set T.ClaimStatus=S.ClaimStatus;
```

XIII.CUBE :

get all combinations of policytype and claimstatus totals.

```
select PolicyType,ClaimStatus,sum(ClaimAmount) as total
from Policies p join PolicyAssignments a on p.PolicyID=a.PolicyID
join Claims c on a.AssignmentID=c.AssignmentID
group by cube(PolicyType,ClaimStatus);
```

XIV.GROUPING SETS:

get separate totals for policytype and agentid.

```
select PolicyType,AgentID,count(AssignmentID) as total
from Policies p join PolicyAssignments pa on p.policyid=pa.policyid
group by grouping sets((PolicyType),(AgentID));
```