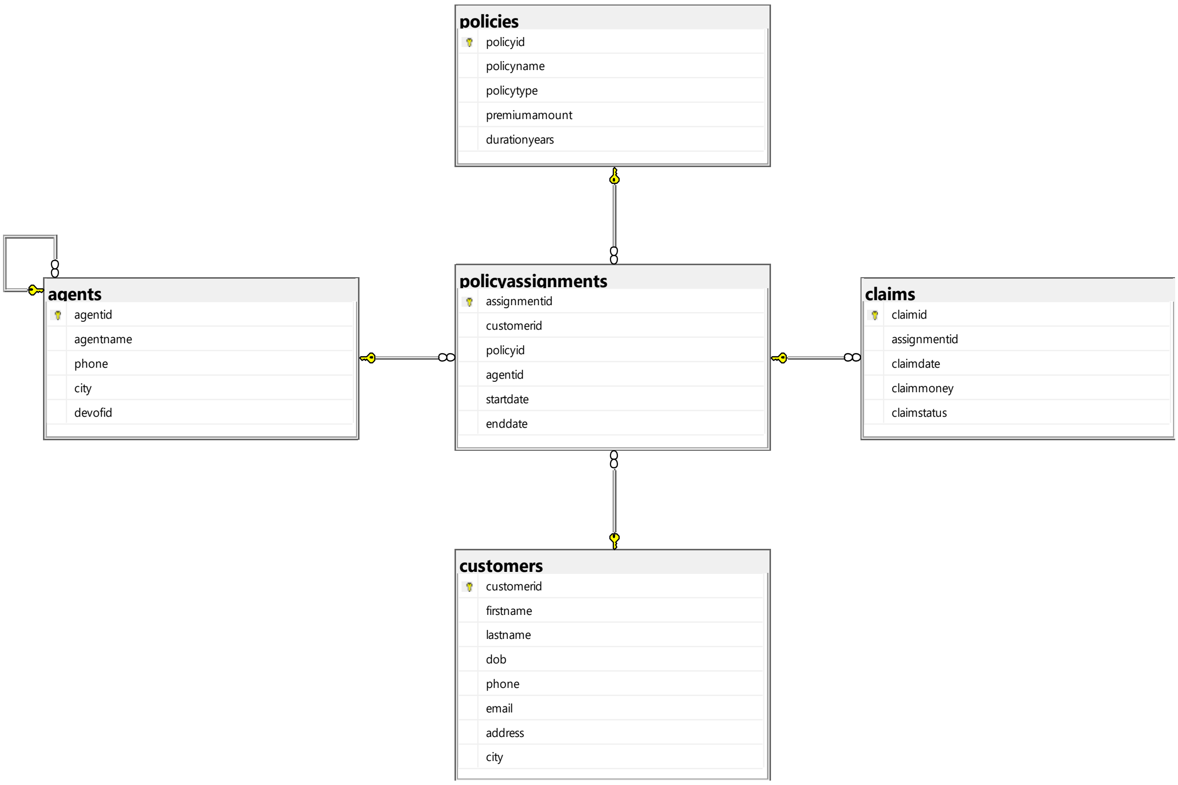
MODULE 4.4: PRACTICAL PROJECT ASSIGNMENT

```
--database creation   create database

InsuranceDB; use InsuranceDB;

--database diagram
```



```
--Customer Table Creation

create table customers( customerid int
identity primary key, firstname
varchar(50), lastname varchar(50), dob
date, phone varchar(10), email
varchar(30) unique
);

--Policies table creation

create table policies( policyid int  identity(200,1)
primary key, policyname varchar(50), policytype
varchar(50), premiumamount decimal(10,2),
durationyears int); -agents table creation   create
table agents( agentid int  identity(300,1) primary
key, agentname  varchar(50), phone varchar(10),
city varchar(50));

--policy assignments table creation

create table policyassignments( assignmentid int identity(1,1) primary key, customerid  int,
policyid int, agentid int, startdate date, enddate date, constraint fk_customer_id foreign
key (customerid) references customers(customerid), constraint fk_policy_id foreign
key(policyid) references policies(policyid), constraint fk_agent_id foreign key(agentid)
references agents(agentid)
);

--claims table creation

create table claims( claimid int identity(400,1) primary key, assignmentid
int, claimdate date, claimmoney decimal(10,2), claimstatus varchar(50),
constraint fk_assignment_id foreign key(assignmentid) references
policyassignments(assignmentid)
);

--inserting data into customers table

INSERT INTO customers (firstname, lastname, dob, phone, email) VALUES
('Rahul', 'Sharma', '1995-04-12', '9876543210', 'rahul@gmail.com'),
('Anita', 'Verma', '1998-08-20', '9123456789', 'anita@gmail.com'),
('Suresh', 'Kumar', '1990-01-15', '9988776655', 'suresh@gmail.com'),
('Priya', 'Singh', '1997-06-10', '9090909090', 'priya@gmail.com'),
('Amit', 'Patel', '1993-11-25', '9555666777', 'amit@gmail.com');

-- inserting data into policies

INSERT INTO policies (policyname, policytype, premiumamount, durationyears)
VALUES
('Health Secure', 'Health', 12000.00, 5),
('Life Shield', 'Life', 15000.00, 10),
('Car Protect', 'Vehicle', 8000.00, 3),
('Home Safe', 'Property', 10000.00, 7),
('Travel Guard', 'Travel', 5000.00, 1);

-- inserting data into agents

INSERT INTO agents (agentname, phone, city) VALUES
('Ramesh Rao', '8888888888', 'Hyderabad'),
('Sunita Das', '7777777777', 'Bangalore'),
('Kiran Mehta', '6666666666', 'Mumbai'),
('Neha Jain', '9999999999', 'Delhi'),
('Arjun Nair', '5555555555', 'Chennai');
```

```
-- inserting data into policy assignments

INSERT INTO policyassignments (customerid, policyid, agentid, startdate, enddate)
VALUES
(1, 200, 300, '2023-01-01', '2028-01-01'),
(2, 201, 301, '2022-05-15', '2032-05-15'),
(3, 202, 302, '2024-03-10', '2027-03-10'),
(4, 203, 303, '2021-09-20', '2028-09-20'),
(5, 204, 304, '2024-01-05', '2025-01-05');

-- inserting data into claims

INSERT INTO claims (assignmentid, claimdate, claimmoney, claimstatus) VALUES
(1, '2024-02-10', 50000.00, 'Approved'),
(2, '2023-11-05', 100000.00, 'Pending'),
(3, '2024-06-18', 20000.00, 'Rejected'),
(4, '2022-12-25', 75000.00, 'Approved'),
(5, '2024-08-01', 15000.00, 'Pending');
```

BASIC SELECT QUESTIONS

```
--Display all details of all customers
select  * from customers;

--Show only firstname, lastname, and email of customers?
select firstname,lastname ,email from customers;

--List all insurance policies with their premium amount and duration?
select policyname,premiumamount,durationyears from policies;

--Display all agents working in Hyderabad?
select agentname from agents
where city='Hyderabad';

--Show all claims where the claim status is Approved?
select * from claims where claimstatus='Approved';
```

WHERE & FILTERING

```
--Find customers born after 1995.
* select * from customers
  where  year(dob)>1995;

--Display policies with a premium amount greater than ₹10,000?
```

```sql
select * from policies where premiumamount>10000;

--List agents whose city is Mumbai or Delhi?
select agentname from agents
where  city in ('Mumbai','Delhi');

--Show claims where claim money is more than ₹50,000?
select * from claims
where  claimmoney>50000;

--Find policies with duration more than 5 years?
select * from policies where  durationyears>5;

--Find customers whose email ends with @gmail.com?
select * from customers where email like '%@gmail.com';

--Display policies whose premium amount is between ₹8,000 and ₹15,000?
* select  * from policies
   where premiumamount between 8000
   and 150000;

--list claims made in the year 2024 only
select * from claims where year(claimdate)=2024;

--concatenating fields

--concat fname and lname in customers table
select concat(firstname,' ',lastname) as full_name
from customers;
--(or) select firstname+lastname as full_name from customers;

--concat fname with mail id of customers

 select firstname+'->'+email from customers;
```

AGGREGATION QUERIES

```sql
--total number of customers
select count(*) as total_customers from
customers;
```

```
 --total number of policies available
select count(*) as total_policies from
policies;

--minimum, maximum, and average premium amount
select
min(premiumamount) as min_premium, max(premiumamount) as
max_premium, avg(premiumamount) as avg_premium from
policies;
--total number of agents in each city
select city, count(agentid) as
total_agents from agents group by city;

--number of policies taken by each customer
select  c.customerid,
c.firstname,
c.lastname, count(pa.policyid) as
policy_count from customers c join
policyassignments pa on
c.customerid = pa.customerid group
by
c.customerid,
c.firstname,
c.lastname;

--total premium amount per policy type
select policytype, sum(premiumamount)
as total_premium from policies group
by policytype;  --average policy
duration by policy type select
policytype, avg(durationyears) as
avg_duration from policies group by
policytype;
```

DATE & TIME QUERIES

```
--get today's date
select getdate() as current_date_time;

--get only current date (without time)
select cast(getdate() as date) as 'current_date';

--get only current time
select cast(getdate() as time) as 'current_time';
```

```
--find customers born after 2000
select * from customers where dob
> '2000-01-01';

--find policies started in the year 2024
select * from policyassignments where
year(startdate) = 2024;

--policies started in the month of april
select * from policyassignments where
month(startdate) = 4;
--policies ending in the next 30 days
select * from policyassignments where enddate between
getdate() and dateadd(day, 30, getdate());

--calculate policy duration in days
select assignmentid, datediff(day, startdate, enddate) as
policy_duration_days from policyassignments;

--number of policies started each year
select year(startdate) as policy_year,
count(*) as total_policies from
policyassignments group by year(startdate);

--total claim amount per year

select year(claimdate) as claim_year,
sum(claimmoney) as total_claim_amount from
claims group by year(claimdate);
```

STRING FUNCTION QUESTIONS

```
--display customer full name in uppercase;
 select upper(firstname + ' ' + lastname) as
full_name from customers;

--display only the first 3 characters of agent names
select agentname, left(agentname, 3) as short_name
from agents;

--display customer email domain name
select email, substring(email, charindex('@', email) + 1, len(email)) as domain from
customers;
```

```
--display policy names with length of name
select policyname, len(policyname) as
name_length from policies;

--replace word 'life' with 'health' in policy type
select policytype,  replace(policytype, 'life', 'health') as
updated_policytype from  policies;

--display customer names without leading/trailing spaces
select ltrim(rtrim(firstname)) as clean_firstname from
customers;
```

NUMERICAL FUNCTION QUESTIONS

```
--round premium amount to nearest integer
select  premiumamount, round(premiumamount,
0) as rounded_premium from policies;

--display premium amount with 2 decimal places
select  premiumamount, format(premiumamount, 'n2') as
formatted_premium from policies;

--display maximum claim amount
select  max(claimmoney) as max_claim
from claims;

--calculate 10% tax on premium amount
select  premiumamount, premiumamount *
0.10 as tax_amount from policies;

--display absolute value of claim amount difference
select  abs(claimmoney - 50000) as
amount_difference from claims;

--display power of duration years (square)
 select durationyears, power(durationyears,
2) as duration_square from policies;

--generate policy code (policyid + policy name)
 select concat(policyid, '_', policyname) as policy_code
from policies;
```

```
--mask customer phone number (show last 4 digits)

select  'xxxxxx' + right(phone, 4) as masked_phone from
customers;


--display customer name and policy name they have taken
select
c.customerid,
c.firstname,
c.lastname,
p.policyname from customers c
inner join policyassignments pa
on c.customerid = pa.customerid
inner join policies p   on
pa.policyid = p.policyid;

--display all customers and their policies (include customers with no policy)
 select  c.customerid,
c.firstname,
p.policyname from customers c
left join policyassignments pa on
c.customerid = pa.customerid
left join policies p   on
pa.policyid = p.policyid;

--display all policies and customers who have taken them
select
p.policyid,
p.policyname,
c.firstname from customers c
right join policyassignments pa on
c.customerid = pa.customerid right
join policies p    on pa.policyid =
p.policyid; -- display all customers
and all policies (matched and
unmatched) select   c.customerid,
c.firstname,
p.policyname from customers c full
outer join policyassignments pa on
c.customerid = pa.customerid full outer
join policies p   on pa.policyid =
p.policyid;
```

```
--display agents working in the same city
select a1.agentname as agent1,
a2.agentname as agent2, a1.city from
agents a1 join agents a2 on a1.city = a2.city
and a1.agentid < a2.agentid;  --display all
combinations of agents and policies select
a.agentname,
p.policyname from agents a
cross join policies p;

--number of policies handled by each agent
select  a.agentid,
a.agentname, count(pa.policyid) as
total_policies from agents a left join
policyassignments pa on
a.agentid = pa.agentid group by
a.agentid,
a.agentname;

--join with date condition
select  c.firstname,
p.policyname,  pa.startdate
from    customers    c    join
policyassignments pa on
c.customerid = pa.customerid
join policies p   on pa.policyid =
p.policyid where
year(pa.startdate) = 2024;

--agents handling more than 2 policies
select  a.agentid,   a.agentname, count(pa.policyid) as
policy_count from agents a join policyassignments pa
on a.agentid
= pa.agentid group by  a.agentid,
a.agentname having count(pa.policyid)
> 2;
```

SUBQUERY QUESTIONS

```
--find customers who have taken at least one policy
select * from customers where customerid in ( select
customerid from policyassignments   );
```

```
--find policies with premium greater than average premium
select * from policies where premiumamount > ( select
avg(premiumamount) from policies
);

--find agents who have assigned policies
select * from agents where agentid in (
select agentid from policyassignments
);

--find customers who have not taken any policy
 select * from customers where customerid not in
( select customerid from policyassignments
);

--find policies that have at least one claim
select * from policies where policyid in (
select policyid from policyassignments
where assignmentid in ( select assignmentid
from claims
)
);

--find customers who have made a claim
select * from customers where
customerid in ( select customerid from
policyassignments where assignmentid in
( select assignmentid from claims
)
);

--find customers who have taken more policies than average
select c.customerid, c.firstname from customers c where (
select count(*) from policyassignments pa where pa.customerid
= c.customerid
) > (
select avg(policy_count) from (
select count(*) as policy_count
from policyassignments group
by customerid
) t
);
```

```
--find policies with total claim amount > 50,000
select policyid from policyassignments where
assignmentid in ( select assignmentid from claims
group by assignmentid having sum(claimmoney)
> 50000
);

--find customers who have never made a claim
select * from customers c where not exists (
select 1 from policyassignments pa join claims cl
on pa.assignmentid = cl.assignmentid where
pa.customerid = c.customerid
);
```

CASE-ELSE QUERIES

```
--categorize policies based on premium amount
select policyname, premiumamount, case when premiumamount < 20000
then 'low premium' when premiumamount between 20000 and 50000 then
'medium premium' else 'high premium' end as premium_category from
policies;

--display claim status in user-friendly text
select claimid, claimstatus, case when claimstatus =
'approved' then '✔ approved' when claimstatus =
'rejected' then '❌ rejected' else ' pending' end as status_message from
claims;

--check whether a policy is active or expired
select assignmentid, startdate, enddate, case when getdate()
between startdate and enddate then 'active' else 'expired' end
as policy_status from policyassignments;

--apply discount based on policy duration
select policyname, durationyears, premiumamount, case when
durationyears >= 10 then premiumamount * 0.90 when durationyears
between 5 and 9 then premiumamount * 0.95 else premiumamount end
as discounted_premium from policies;
```

```
--categorize customers based on age
select firstname, dob, case when datediff(year, dob, getdate()) < 18
then 'minor' when datediff(year, dob, getdate()) between 18 and 60
then 'adult' else
'senior citizen' end as age_group from customers;

--show claim risk level based on claim amount
select claimmoney, case when claimmoney < 25000 then 'low
risk' when claimmoney between 25000 and 75000 then 'medium
risk' else
'high risk' end as risk_level from claims;

--count claims by status using case
select count(case when claimstatus =
'approved' then 1 end) as approved_claims, count(case when claimstatus =
'rejected' then 1 end) as rejected_claims, count(case when claimstatus =
'pending' then 1 end) as pending_claims from claims;

--calculate total claim amount based on status
 select claimstatus, sum( case when claimstatus
= 'approved' then claimmoney else 0
end   ) as total_claim_amount from
claims group by claimstatus;
```

GROUP BY ROLLUP QUERIES

```
--total premium amount by policy type + grand total
select policytype, sum(premiumamount) as
total_premium from policies group by rollup(policytype);

--count of policies by duration years + total
select durationyears, count(*) as policy_count from
policies group by rollup(durationyears);

--number of policies handled by each agent + agent total
select
a.agentname, count(pa.assignmentid) as
total_policies from agents a left join
policyassignments pa on a.agentid =
pa.agentid group by rollup(a.agentname);
```

```
--total claim amount by claim status + total
select claimstatus, sum(claimmoney) as
total_claim_amount from claims group by
rollup(claimstatus);
```

CUBE QUERIES

```
--total premium by policy type, duration, and all totals
select policytype, durationyears, sum(premiumamount)
as total_premium from policies group by
cube(policytype, durationyears);

--count of policies by policy type and total
select policytype, count(*) as policy_count
from policies group by cube(policytype);

--total claims by claim status and year
select claimstatus, year(claimdate) as claim_year,
sum(claimmoney) as total_claim_amount from
claims group by cube(claimstatus,
year(claimdate));

--number of policies handled by agents by city and agent
select
a.city,
a.agentname, count(pa.assignmentid) as
policies_handled from agents a join
policyassignments pa on a.agentid =
pa.agentid group by cube(a.city, a.agentname);
```

GROUPING SETS QUERIES

```
--total premium by policy type and grand total
select policytype, sum(premiumamount) as
total_premium from policies group by grouping sets
(
(policytype),
() );

--count of policies by duration and policy type
select durationyears, policytype, count(*) as
policy_count from policies group by grouping sets
(
(durationyears),
(policytype)
);

--total claim amount by claim status and year
select claimstatus, year(claimdate) as claim_year,
sum(claimmoney) as total_claim_amount from
claims group by grouping sets (
(claimstatus),
(year(claimdate))
);
```

SET OPERATIONS

```
--get all unique customer ids who either took a policy or made a claim
select customerid   from policyassignments
UNION
select pa.customerid from  policyassignments pa
join claims c on pa.assignmentid =
c.assignmentid;

--get unique cities of agents and customers (if customers had city column)

select city  from
agents
UNION
select 'unknown city';
--get all customer ids from policy assignments and claims (allow duplicates)
select customerid
from policyassignments
UNION ALL
select customerid  from
policyassignments;

--customers who have policies and have made claims
select customerid
from policyassignments
INTERSECT
select pa.customerid from
policyassignments pa  join claims
c on pa.assignmentid =
c.assignmentid;

--policies that are both assigned and claimed
select policyid  from policyassignments
```

```sql
INTERSECT
select pa.policyid from
policyassignments pa join
claims c on pa.assignmentid =
c.assignmentid;

--customers who have taken a policy but never made a claim
select customerid  from policyassignments
EXCEPT
select pa.customerid from policyassignments
pa join claims c on pa.assignmentid =
c.assignmentid;


--policies that were assigned but never claimed
select policyid  from policyassignments
EXCEPT
select pa.policyid from
policyassignments pa join claims c on
pa.assignmentid = c.assignmentid;

--gents who handled policies but never handled claims
select agentid  from policyassignments
EXCEPT
select pa.agentid from
policyassignments pa join claims c on
pa.assignmentid = c.assignmentid;

--all policy ids involved in assignments or claims
select policyid  from policyassignments
UNION
select pa.policyid from
policyassignments pa join claims c on
pa.assignmentid = c.assignmentid;
```