```
create database test
use database test

create schema exam
use schema exam

create or replace table customers(
customer_id int primary key,
first_name varchar(50)not null,
last_name varchar(50)not null,
date_of_birth date,
email varchar(100)unique not null,
country  varchar(50),
created_at TIMESTAMP_NTZ DEFAULT CURRENT_TIMESTAMP
);

INSERT INTO customers VALUES
 (1, 'Asha', 'Patel', '1988-05-12', 'asha.patel@example.com', 'India', '2023-11-01 09:10:00'),
 (2, 'Ravi', 'Kumar', '1992-01-22', 'ravi.kumar@example.com', 'India', '2024-02-20 14:30:00'),
 (3, 'Maya', 'Singh', '1979-07-15', 'maya.singh@example.com', 'USA', '2022-08-05 10:00:00'),
 (4, 'Liam', 'Brown', '1985-10-02', 'liam.brown@example.com', 'UK', '2024-01-12 16:45:00'),
 (5, 'Priya', 'Sharma', '1990-03-18', 'priya.sharma@example.com', 'India', '2023-06-15
11:20:00'),
 (6, 'John', 'Davis', '1987-09-25', 'john.davis@example.com', 'USA', '2023-12-03 08:45:00'),
 (7, 'Emma', 'Wilson', '1993-11-08', 'emma.wilson@example.com', 'UK', '2024-03-10
13:30:00'),
 (8, 'Raj', 'Mehta', '1985-07-20', 'raj.mehta@example.com', 'India', '2022-09-18 15:15:00'),
 (9, 'Sophie', 'Martin', '1991-04-14', 'sophie.martin@example.com', 'France', '2023-08-22
09:55:00'),
 (10, 'David', 'Lee', '1989-12-30', 'david.lee@example.com', 'Singapore', '2024-01-05
12:10:00'),
 (11, 'Anita', 'Reddy', '1986-08-07', 'anita.reddy@example.com', 'India', '2023-04-12
10:25:00'),
 (12, 'Michael', 'Johnson', '1988-02-28', 'michael.johnson@example.com', 'Australia',
'2023-10-15
14:40:00'),
 (13, 'Kavya', 'Nair', '1994-06-11', 'kavya.nair@example.com', 'India', '2024-02-08 16:20:00'),
 (14, 'Oliver', 'Smith', '1982-05-03', 'oliver.smith@example.com', 'UK', '2022-11-25 11:05:00'),
 (15, 'Aisha', 'Khan', '1990-10-19', 'aisha.khan@example.com', 'UAE', '2023-07-30 13:50:00'),
 (16, 'Carlos', 'Rodriguez', '1987-01-16', 'carlos.rodriguez@example.com', 'Spain',
'2023-09-12
08:30:00'),
 (17, 'Sneha', 'Gupta', '1992-09-05', 'sneha.gupta@example.com', 'India', '2024-04-18
15:45:00'),
 (18, 'James', 'Taylor', '1984-03-22', 'james.taylor@example.com', 'Canada', '2023-05-07
12:35:00'),
 (19, 'Lisa', 'Anderson', '1991-12-14', 'lisa.anderson@example.com', 'USA', '2023-11-28
09:15:00'),
```

```
 (20, 'Arjun', 'Iyer', '1989-07-09', 'arjun.iyer@example.com', 'India', '2024-01-22 17:00:00'),
 (21, 'Hannah', 'White', '1993-04-26', 'hannah.white@example.com', 'New Zealand',
'2023-08-14
10:40:00'),
 (22, 'Vikram', 'Chopra', '1986-11-12', 'vikram.chopra@example.com', 'India', '2022-12-09
14:25:00'),
 (23, 'Grace', 'Thompson', '1988-08-31', 'grace.thompson@example.com', 'Ireland',
'2023-06-03
11:55:00'),
 (24, 'Rohan', 'Joshi', '1990-02-17', 'rohan.joshi@example.com', 'India', '2024-03-25
13:10:00'),
 (25, 'Isabella', 'Garcia', '1987-10-04', 'isabella.garcia@example.com', 'Mexico', '2023-09-08
16:30:00');

select*from customers;

CREATE OR REPLACE TABLE branches (
branch_code VARCHAR(10) PRIMARY KEY,
branch_name VARCHAR(100),
city VARCHAR(50),
manager VARCHAR(50));

INSERT INTO branches VALUES
 ('BR001', 'Downtown Branch', 'Chennai', 'S. Laxmi'),
 ('BR002', 'Main Street Branch', 'Bengaluru', 'D. Subhashini'),
 ('BR003', 'City Center Branch', 'New York', 'J. Williams'),
 ('BR004', 'Park Avenue Branch', 'Mumbai', 'R. Sharma'),
 ('BR005', 'Liberty Square Branch', 'London', 'M. Johnson'),
 ('BR006', 'Marina Bay Branch', 'Singapore', 'L. Tan'),
 ('BR007', 'Central Plaza Branch', 'Delhi', 'A. Gupta'),
 ('BR008', 'Harbor View Branch', 'Sydney', 'K. Smith'),
 ('BR009', 'Tech Hub Branch', 'Hyderabad', 'V. Reddy'),
 ('BR010', 'Financial District Branch', 'Toronto', 'C. Brown'),
 ('BR011', 'Riverside Branch', 'Pune', 'N. Patil'),
 ('BR012', 'Business Center Branch', 'Dubai', 'H. Al-Rashid'),
 ('BR013', 'Innovation Square Branch', 'Kolkata', 'S. Chatterjee'),
 ('BR014', 'Metro Plaza Branch', 'Bangkok', 'P. Thanakit'),
 ('BR015', 'Corporate Tower Branch', 'Frankfurt', 'A. Müller'),
 ('BR016', 'Green Valley Branch', 'Ahmedabad', 'M. Patel'),
 ('BR017', 'Sunset Boulevard Branch', 'Los Angeles', 'R. Garcia'),
 ('BR018', 'Trade Center Branch', 'Hong Kong', 'W. Chan'),
 ('BR019', 'Silicon Valley Branch', 'San Francisco', 'J. Davis'),
 ('BR020', 'Financial Hub Branch', 'Zurich', 'L. Weber'),
 ('BR021', 'Business Bay Branch', 'Abu Dhabi', 'K. Al-Mahmoud'),
 ('BR022', 'Tech Park Branch', 'Gurgaon', 'R. Singh'),
 ('BR023', 'Capital Square Branch', 'Washington DC', 'M. Wilson'),
 ('BR024', 'Innovation Center Branch', 'Tokyo', 'T. Yamamoto'),
 ('BR025', 'Global Trade Branch', 'Amsterdam', 'P. Van Der Berg');
```

```
 select * from branches;


create or replace table accounts(
account_id int primary key,
customer_id int not null,
account_type varchar(20) not null,
balance decimal(18,2)default 0.00,
opened_date date not null,
branch_code varchar(10) not null
);

INSERT INTO accounts VALUES
(1001, 1, 'SAVINGS', 12500.50, '2023-11-01', 'BR001'),
(1002, 1, 'CURRENT', 550000.00, '2024-03-05', 'BR002'),
(1003, 2, 'SAVINGS', 7800.00, '2024-02-21', 'BR001'),
(1004, 3, 'LOAN', -250000.00, '2022-08-10', 'BR003'),
(1005, 4, 'SAVINGS', 15000.00, '2024-01-15', 'BR002'),
(1006, 5, 'CURRENT', 85000.00, '2023-06-16', 'BR004'),
(1007, 6, 'SAVINGS', 23500.75, '2023-12-04', 'BR003'),
(1008, 7, 'LOAN', -180000.00, '2024-03-11', 'BR005'),
(1009, 8, 'SAVINGS', 45000.00, '2022-09-19', 'BR001'),
(1010, 9, 'CURRENT', 125000.00, '2023-08-23', 'BR006'),
(1011, 10, 'SAVINGS', 18750.25, '2024-01-06', 'BR007'),
(1012, 11, 'LOAN', -75000.00, '2023-04-13', 'BR004'),
(1013, 12, 'CURRENT', 95000.00, '2023-10-16', 'BR008'),
(1014, 13, 'SAVINGS', 32000.00, '2024-02-09', 'BR001'),
(1015, 14, 'CURRENT', 215000.00, '2022-11-26', 'BR005'),
(1016, 15, 'SAVINGS', 41500.50, '2023-07-31', 'BR009'),
(1017, 16, 'LOAN', -120000.00, '2023-09-13', 'BR010'),
(1018, 17, 'CURRENT', 78000.00, '2024-04-19', 'BR001'),
(1019, 18, 'SAVINGS', 27500.00, '2023-05-08', 'BR011'),
(1020, 19, 'CURRENT', 145000.00, '2023-11-29', 'BR003'),
(1021, 20, 'SAVINGS', 19200.75, '2024-01-23', 'BR007'),
(1022, 21, 'LOAN', -95000.00, '2023-08-15', 'BR012'),
(1023, 22, 'CURRENT', 167000.00, '2022-12-10', 'BR004'),
(1024, 23, 'SAVINGS', 38500.00, '2023-06-04', 'BR013'),
(1025, 24, 'CURRENT', 92000.00, '2024-03-26', 'BR001');

select*from accounts;


create or replace table transactions(
transaction_id int primary key,
account_id int not null,
transaction_date TIMESTAMP_NTZ DEFAULT CURRENT_TIMESTAMP(),
amount_number decimal(18,2)not null,
currency varchar(10)default'INR',
```

```
transaction_type varchar(10),
description varchar(200)
);

INSERT INTO transactions VALUES
(5001, 1001, '2024-04-01 10:15:00', -1500.00, 'INR', 'DEBIT', 'ATM Withdrawal'),
(5002, 1002, '2024-05-03 12:00:00', 20000.00, 'INR', 'CREDIT', 'Salary Deposit'),
(5003, 1003, '2024-05-05 09:30:00', -800.00, 'INR', 'DEBIT', 'POS Purchase'),
(5004, 1004, '2024-03-01 11:00:00', -50000.00, 'USD', 'DEBIT', 'Loan Repayment'),
(5005, 1005, '2024-06-10 15:20:00', 2500.00, 'GBP', 'CREDIT', 'Transfer In'),
(5006, 1006, '2024-04-15 08:45:00', 15000.00, 'INR', 'CREDIT', 'Business Deposit'),
(5007, 1007, '2024-03-22 14:30:00', -2200.00, 'USD', 'DEBIT', 'Online Purchase'),
(5008, 1008, '2024-05-18 16:10:00', -25000.00, 'GBP', 'DEBIT', 'Loan EMI'),
(5009, 1009, '2024-02-28 11:25:00', 8500.00, 'INR', 'CREDIT', 'Fixed Deposit Maturity'),
(5010, 1010, '2024-06-02 13:40:00', -12000.00, 'EUR', 'DEBIT', 'International Transfer'),
(5011, 1011, '2024-04-08 09:55:00', 3500.00, 'SGD', 'CREDIT', 'Forex Exchange'),
(5012, 1012, '2024-03-12 15:20:00', -18000.00, 'INR', 'DEBIT', 'Loan Payment'),
(5013, 1013, '2024-05-25 10:30:00', 22000.00, 'AUD', 'CREDIT', 'Export Payment'),
(5014, 1014, '2024-04-30 12:15:00', -950.00, 'INR', 'DEBIT', 'Utility Bill'),
(5015, 1015, '2024-06-12 14:50:00', 45000.00, 'GBP', 'CREDIT', 'Investment Return'),
(5016, 1016, '2024-03-18 11:35:00', -3200.00, 'AED', 'DEBIT', 'Shopping'),
(5017, 1017, '2024-05-08 16:45:00', -35000.00, 'EUR', 'DEBIT', 'Loan Installment'),
(5018, 1018, '2024-06-05 08:20:00', 12500.00, 'INR', 'CREDIT', 'Freelance Payment'),
(5019, 1019, '2024-04-20 13:10:00', -1800.00, 'CAD', 'DEBIT', 'Restaurant Bill'),
(5020, 1020, '2024-05-15 15:30:00', 28000.00, 'USD', 'CREDIT', 'Consulting Fee'),
(5021, 1021, '2024-03-28 10:40:00', -750.00, 'SGD', 'DEBIT', 'Mobile Recharge'),
(5022, 1022, '2024-06-08 12:55:00', -40000.00, 'NZD', 'DEBIT', 'Property Loan EMI'),
(5023, 1023, '2024-04-25 14:25:00', 18500.00, 'INR', 'CREDIT', 'Stock Dividend'),
(5024, 1024, '2024-05-12 11:50:00', -2100.00, 'EUR', 'DEBIT', 'Insurance Premium'),
(5025, 1025, '2024-06-01 16:15:00', 9800.00, 'MXN', 'CREDIT', 'Remittance Received');

select*from transactions;
```

Question 1:
Find all branches that have more than 1 account and show the branch name, city, total number of accounts, average
balance, and total balance. Only include branches where the average balance is greater than 50,000. Also show the
manager name in uppercase.

```
select b.branch_name,
b.city,
upper(b.manager)as manager_case,
avg(a.balance) as account_balance,
count(a.account_id)as account_id
from branches as b
join accounts as a
```

```
on b.branch_code = a.branch_code
group by b.branch_name,b.city,b.manager
having count(a.account_id) >1
and avg(a.balance) > 50000;
```

2.Find all customers who have account balances higher than the average balance of all positive-balance accounts, AND
who have made at least one transaction since January 1, 2024. Show customer ID, full name, email, account type, and
balance.

```
select c.customer_id,
c.first_name || c.last_name as cust_name,
c.email,
a.balance,
a.account_type,
t.transaction_id
from customers as c
join accounts as a
on c.customer_id = a.customer_id
join transactions t
on a.account_id = t.account_id
where balance > 1
and transaction_date > '2024-01-1';
```

3.Create a stored procedure called 'get_customer_transactions' that accepts a customer ID and number of days as
parameters, and returns all transactions for that customer within the specified number of days from current date. Then call
this procedure for customer ID 1 with last 30 days

3.Create a stored procedure called 'get_customer_transactions' that accepts a customer ID and number of days as
parameters, and returns all transactions for that customer within the specified number of days from current date. Then call
this procedure for customer ID 1 with last 30 days.

```
--4.
--Find all customers who have made at least one transaction in the last 6 months, but DO NOT have any loan accounts with balance less than-100,000. Show customer ID, full name, and email
```

```
select c.customer_id,
c.first_name || c.last_name as customer,
c.email from customers c
```

```
where exists(
select 1
from accounts as a
where c.customer_id = a.customer_id
and a.balance<6
)
and not exists(
select 1
from transactions as t
where c.customer_id = t.customer_id
and t.balance<100000
)
```

```
   select c.customer_id,concat(c.first_name,c.last_name) as full_name,c.email from
customers as c
       where exists (select * from transactions as t
    join accounts a on t.account_id = a.account_id
       where a.customer_id = c.customer_id
       and t.tx_date >= dateadd(month, -6, current_date))
       and not exists (select * from accounts as a
    where a.customer_id = c.customer_id
       and a.account_type = 'loan'
       and a.balance < - 100000
   );
```

1.Write a query using CTE to find employees who earn above the average salary.

```
with salaries as (
select e.employee_id,s.salary_amount
from employees
)
select e.first_name || e.last_name,s.salary_amount
from employees as e
join salaries as s
on e.employee_id = s.employee_id;
```

```
create database projects
```

```
use database projects

create schema pro_manner;
use schema pro_manner;

create or replace table departments(
 department_id int primary key autoincrement,
 department_name varchar(100)unique not null
);

INSERT INTO departments (department_name) VALUES
('Human Resources'),
('Finance'),
('Information Technology'),
('Marketing'),
('Sales'),
('Operations'),
('Customer Support'),
('Research and Development'),
('Legal'),
('Administration');

select *from departments;

create or replace table employees(
 employee_id int  autoincrement primary key,
 first_name varchar(50) not null,
 last_name varchar(50) not null,
 department_id int foreign key references departments(department_id),
 hire_date date not null
 );

 insert into employees (first_name, last_name, department_id, hire_date) values('Alice',
'Johnson', 1, '2022-11-15'),
('Bob', 'Smith', 2, '2023-02-10'),
('Charlie', 'Brown', 3, '2023-06-01'),
('Diana', 'Miller', 4, '2021-09-20'),
('Ethan', 'Wilson', 2, '2023-03-05'),
('Fiona', 'Clark', 3, '2023-07-12'),
('George', 'Lopez', 1, '2022-12-30'),
('Hannah', 'White', 4, '2023-04-22'),
('Ian', 'Taylor', 3, '2023-08-14'),
('Julia', 'Davis', 2, '2020-05-18');

select*from employees;


create table projects(
```

```
 project_id int primary key autoincrement,
 project_name varchar(100) not null,
 start_date date,
 end_date date,
 department_id int foreign key references departments(department_id)
 );

INSERT INTO projects (project_name, start_date, end_date, department_id) VALUES
('Employee Onboarding Program', '2023-01-10', '2023-03-30', 1), -- HR
('Annual Budget Planning', '2023-02-01', '2023-04-15', 2),      -- Finance
('IT Infrastructure Upgrade', '2023-03-01', '2023-07-31', 3),   -- IT
('Social Media Marketing Campaign', '2023-04-01', '2023-06-30', 4), -- Marketing
('Global Sales Strategy', '2023-05-01', '2023-08-15', 5),       -- Sales
('Logistics Optimization', '2023-06-01', '2023-09-30', 6),      -- Operations
('Customer Service Chatbot', '2023-07-10', '2023-10-20', 7),    -- Customer Support
('AI Product Research', '2023-08-01', '2024-01-15', 8),         -- R&D
('Regulatory Compliance Audit', '2023-09-01', '2023-11-30', 9), -- Legal
('Head Office Renovation', '2023-10-01', '2023-12-31', 10);     -- Administration

select * from projects;


create table employeeprojects(
 employee_id int foreign key references employees (employee_id),
 project_id int foreign key references projects(project_id),
 assigned_date date not null
);

insert into employeeprojects(employee_id,project_id,assigned_date)values (1, 101,
'2024-01-15'),
(2, 101, '2024-01-20'),
(3, 102, '2024-02-05'),
(1, 103, '2024-02-10'),
(4, 103, '2024-02-15'),
(5, 104, '2024-03-01'),
(2, 104, '2024-03-05'),
(3, 105, '2024-03-10'),
(4, 106, '2024-04-01'),
(5, 106, '2024-04-05');

select*from employeeprojects;


Assessment questions;
1
select first_name,last_name,hire_date
from employees e
where hire_date>'2022-01-01'
```

```
order by hire_date desc;
```

2

```
select end_date
from projects e
where end_date is not null
limit 5;
```

3

```
select d.department_name,count(employee_id) as e
from departments as d
join employees as e
on d.department_id = e.department_id
group by d.department_name
having count (employee_id) > 2
order by department_name;
```

4

```
select avg(datediff(day,start_date,end_date))from projects;
```

5

```
select max(e.hire_date),min(e.hire_date),d.department_name
from employees as e
join departments as d
on d.department_id =
e.department_id
group by d.department_name
```

6.write a query to display emp_id,first_name,and department_name for all employees.

```
select e.employee_id,e.first_name,d.department_name
from employees as e
join departments as d
on d.department_id = e.department_id;
```

7.Display all projects along with their department_name and the names of employees assigned to them.if a project has no employees,it should still be shown.

```
select p.project_name,d.department_name,
e.first_name || e.last_name as employee_name,ep.assigned_date
from projects as p
left join departments as d
on d.department_id = p.department_id
left join employees as e
on d.department_id = e.department_id
left join employeeprojects as ep
on e.employee_id = ep.employee_id
order by p.project_name;
```

8
write a query to list all employees and the projects they are assigned to,along with their department name and project start date.If a project exists without employees,it should still be shown.

```
select e.first_name || e.last_name,p.project_id,d.department_name,
p.start_date
from employees as e
left join projects as p
on e.department_id = p.department_id
left join departments as d
on d.department_id = e.department_id
```

9

```
create or replace view active_employees as
select e.employee_id,e.first_name || e.last_name as empl_name,
d.department_name
from employees as e
join departments as d
on d.department_id = e.department_id
where hire_date>=(dateadd(year,-2,current_date));

select *from active_employees view;
```

10

```
select employee_id,first_name || last_name as emp_name,department_id
from employees as e
where employee_id =(select(department_id)from employees as e where employee_id=5);
```

11.write a stored procedure sp_assign_employee_to_project (empID int,projectid int,assigndate,date)which record into employee_projects.

```
create or replace procedure sp_assign_employee_to_project
(
  employee_id int,
  project_id int,
  assigned_date date
)
Begin

 insert into employee_projects(employee_id,project_id,
assigned_date)values(1, 101, '2024-01-15');

end;


CALL sp_assign_employee_to_project(1, 101, '2024-01-15');
```

13

```
select hire_date,department_id
from employees as e
where hire_date = (select min(e.hire_date)
from employees as e)
```

14

```
select ep.employee_id,ep.project_id
from employeeprojects as ep
where exists(
  select 1
  from employees e
  where e.employee_id = ep.employee_id
);
```

15

```
select ep.employee_id,ep.project_id
from employeeprojects as ep
where not exists(
select *
from employees as e
where e.employee_id = ep.employee_id
);
```