Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah **SAP ID:** 60018230034
Branch: Artificial Intelligence(AI) and Data Science **Div:** S
Course: Database Technologies **Roll No.:** S009

Experiment no. 3

Code:

```
create database InventoryManagement;
useInventoryManagement;

create table products
(
      p_id int primary key,
      p_price int,
      p_expiry varchar(20)
);

create table provider
(
      pr_id int primary key,
      pr_type varchar(40),
      pr_address varchar(100)
);

create table warehouse
(
      w_no int primary key,
      w_capacity int,
      w_location varchar(100)
);

create table orders
(
      or_id int primary key,
      or_date varchar(16),
      or_time varchar(10)
);

create table sales
(
      s_number int primary key,
```

```
      s_type varchar(20),
      s_date varchar(16)
);

create table employee
(
      e_id int primary key,
      e_name varchar(50) not null,
      e_age int,
      e_experience int
);

create table customer
(
      c_id int primary key,
      c_name varchar(50) not null,
      c_contact int,
      c_age int
);

create table customer_care
(
      cc_id int primary key,
      cc_contact int,
      cc_location varchar(100)
);

create table offers
(
      o_no int primary key,
      o_name varchar(50),
      o_type varchar(20)
);

create table payment
(
      py_id int primary key,
      py_time varchar(10),
      py_date varchar(16),
      py_mode varchar(50) not null
);

create table online
(
      on_upi varchar(80),
```

```
      on_credit varchar(80),
      on_debit varchar(80)
);

create table offline
(
      off_cod varchar(80)
);


alter table products
      add pr_id int;
alter table products
      add constraint foreign key(pr_id) references provider(pr_id);

alter table products
      add c_id int;
alter table products
      add constraint foreign key(c_id) references customer(c_id);

alter table products
      add w_no int;
alter table products
      add constraint foreign key(w_no) references warehouse(w_no);


alter table orders
      add w_no int;
alter table orders
      add constraint foreign key(w_no) references warehouse(w_no);


alter table sales
      add e_id int;
alter table sales
      add constraint foreign key(e_id) references employee(e_id);


alter table customer
      add cc_id int;
alter table customer
      add constraint foreign key(cc_id) references
customer_care(cc_id);
```

```
alter table customer_care
      add e_id int;
alter table customer_care
      add constraint foreign key(e_id) references employee(e_id);


alter table offers
      add c_id int;
alter table offers
      add constraint foreign key(c_id) references customer(c_id);


alter table payment
      add c_id int;
alter table payment
      add constraint foreign key(c_id) references customer(c_id);


alter table online
      add py_id int;
alter table online
      add constraint foreign key(py_id) references payment(py_id);

alter table offline
      add py_id int;
alter table offline
      add constraint foreign key(py_id) references payment(py_id);




desc products;
desc provider;
desc warehouse;
desc orders;
desc sales;
desc employee;
desc customer;
desc customer_care;
desc offers;
desc payment;
desc online;
desc offline;

insert into provider values(987654,'retail','Mumbai');
```

```
insert into provider values(654321,'wholesale','Mumbai');
insert into provider values(020202,'wholesale','Mumbai');
select * from provider;
delete from provider
  where pr_id=987654;
select * from provider;
update provider
  set pr_address='Pune';
select * from provider;
update provider
  set pr_address='Mumbai'
  where pr_id=654321;
select * from provider;

insert into warehouse values(40105,2000,'Thane');
select * from warehouse;
delete from warehouse;
select * from warehouse;
```

Output:

```
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| p_id     | int         | NO   | PRI | NULL    |       |
| p_price  | int         | YES  |     | NULL    |       |
| p_expiry | varchar(20) | YES  |     | NULL    |       |
| pr_id    | int         | YES  | MUL | NULL    |       |
| c_id     | int         | YES  | MUL | NULL    |       |
| w_no     | int         | YES  | MUL | NULL    |       |
+----------+-------------+------+-----+---------+-------+

+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| pr_id      | int          | NO   | PRI | NULL    |       |
| pr_type    | varchar(40)  | YES  |     | NULL    |       |
| pr_address | varchar(100) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+

+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
```

```
| w_no       | int          | NO   | PRI | NULL    |       |
| w_capacity | int          | YES  |     | NULL    |       |
| w_location | varchar(100) | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+

+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| or_id   | int         | NO   | PRI | NULL    |       |
| or_date | varchar(16) | YES  |     | NULL    |       |
| or_time | varchar(10) | YES  |     | NULL    |       |
| w_no    | int         | YES  | MUL | NULL    |       |
+---------+-------------+------+-----+---------+-------+

+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| s_number | int         | NO   | PRI | NULL    |       |
| s_type   | varchar(20) | YES  |     | NULL    |       |
| s_date   | varchar(16) | YES  |     | NULL    |       |
| e_id     | int         | YES  | MUL | NULL    |       |
+----------+-------------+------+-----+---------+-------+

+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| e_id         | int         | NO   | PRI | NULL    |       |
| e_name       | varchar(50) | NO   |     | NULL    |       |
| e_age        | int         | YES  |     | NULL    |       |
| e_experience | int         | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+

+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| c_id      | int         | NO   | PRI | NULL    |       |
| c_name    | varchar(50) | NO   |     | NULL    |       |
| c_contact | int         | YES  |     | NULL    |       |
| c_age     | int         | YES  |     | NULL    |       |
| cc_id     | int         | YES  | MUL | NULL    |       |
+-----------+-------------+------+-----+---------+-------+

+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
```

```
| cc_id       | int          | NO   | PRI | NULL    |       |
| cc_contact  | int          | YES  |     | NULL    |       |
| cc_location | varchar(100) | YES  |     | NULL    |       |
| e_id        | int          | YES  | MUL | NULL    |       |
+-------------+--------------+------+-----+---------+-------+

+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| o_no   | int         | NO   | PRI | NULL    |       |
| o_name | varchar(50) | YES  |     | NULL    |       |
| o_type | varchar(20) | YES  |     | NULL    |       |
| c_id   | int         | YES  | MUL | NULL    |       |
+--------+-------------+------+-----+---------+-------+

+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| py_id   | int         | NO   | PRI | NULL    |       |
| py_time | varchar(10) | YES  |     | NULL    |       |
| py_date | varchar(16) | YES  |     | NULL    |       |
| py_mode | varchar(50) | NO   |     | NULL    |       |
| c_id    | int         | YES  | MUL | NULL    |       |
+---------+-------------+------+-----+---------+-------+

+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| on_upi    | varchar(80) | YES  |     | NULL    |       |
| on_credit | varchar(80) | YES  |     | NULL    |       |
| on_debit  | varchar(80) | YES  |     | NULL    |       |
| py_id     | int         | YES  | MUL | NULL    |       |
+-----------+-------------+------+-----+---------+-------+

+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| off_cod | varchar(80) | YES  |     | NULL    |       |
| py_id   | int         | YES  | MUL | NULL    |       |
+---------+-------------+------+-----+---------+-------+

+--------+-----------+------------+
| pr_id  | pr_type   | pr_address |
+--------+-----------+------------+
|  20202 | wholesale | Mumbai     |
```

| 654321 | wholesale | Mumbai |
| 987654 | retail | Mumbai |

pr_id	pr_type	pr_address
20202	wholesale	Mumbai
654321	wholesale	Mumbai

pr_id	pr_type	pr_address
20202	wholesale	Pune
654321	wholesale	Pune

pr_id	pr_type	pr_address
20202	wholesale	Pune
654321	wholesale	Mumbai

w_no	w_capacity	w_location
40105	2000	Thane

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah **SAP ID:** 60018230034
Branch: Artificial Intelligence(AI) and Data Science **Div:** S
Course: Database Technologies **Roll No.:** S009

Experiment no. 4

Employee (e_no, e_name, jobtitle, joindate, sal, d_no);
Department (d_no, d_name)

1. Display the names of the employees whose department no. is 4
2. Find the names of the employees whose department name is 'AI&DS'
3. Display the names of employees whose job title is manager
4. List all names and join date of employees whose name starts with letter 'P'
5. List names of employees who earn more than 50000
6. List name of all employees who joined between 2020 and 2024
7. List all names and join date of employees whose name has letter e as the second letter
8. Display name of employees excluding job title CEO
9. Display employees in descending order of join date
10. Calculate average sal of department no. 5
11. Find the average salary of employees whose average salary is greater than 10000 and department no. is 2

Code:

```
create database exp_4;
use exp_4;

create table department(
  d_no int PRIMARY KEY,
  d_name varchar(100) NOT NULL
);

create table Employees(
  e_no int PRIMARY KEY,
  e_name varchar(100) NOT NULL,
  jobtitle varchar(100),
  joindate int,
  sal int,
  d_no int references department(d_no)
```

```
);



insert into department values(1,'Computer');
insert into department values(2,'IT');
insert into department values(3,'AI&DS');
insert into department values(4,'DS');
insert into department values(5,'AI&ML');
select * from department;

insert into Employees values(1010,'Clark','CEO',2012,4000000,2);
insert into Employees values(1011,'Pearson','CFO',2012,4000000,2);
insert into Employees values(1050,'Harvey','COO',2012,3000000,1);
insert into Employees values(1100,'Louis','HR',2016,2000000,3);
insert into Employees values(1005,'Mike','Department
Head',2011,1000000,3);
insert into Employees values(1080,'Donna','Director',2015,500000,4);
insert into Employees values(2050,'Rachel','Manager',2022,50000,4);
insert into Employees values(2100,'Harold','Manager',2024,40000,5);
insert into Employees values(2001,'Peter','Manager',2020,60000,1);
insert into Employees values(2023,'Steve','Manager',2021,64000,3);
insert into Employees values(2000,'Mabel','Manager',2020,50000,4);
insert into Employees values(2120,'Oliver','Employee',2024,25000,5);
insert into Employees values(2110,'Marissa','Employee',2024,25000,5);
select * from Employees;

select e_name
from Employees
where d_no = 4;

select e_name
from Employees e,department d
where e.d_no = d.d_no and d_name= 'AI&DS';

select e_name
from Employees
where jobtitle = 'Manager';

select e_name, joindate
from Employees
where e_name like 'P%';

select e_name, sal
from Employees
```

```
where sal>50000
order by sal desc;

select e_name
from Employees
where joindate between 2020 and 2024;

select e_name, joindate
from Employees
where e_name like '_e%';

select e_name
from Employees
where jobtitle != 'CEO';

select e_name, joindate
from Employees
order by joindate desc;

select avg(sal) as Average_Salary
from Employees
where d_no=5;

select avg(sal) as Average_Salary
from Employees
where d_no=2
group by d_no
having avg(sal)>10000;
```

Output:

```
+------+----------+
| d_no | d_name   |
+------+----------+
|    1 | Computer |
|    2 | IT       |
|    3 | AI&DS    |
|    4 | DS       |
|    5 | AI&ML    |
+------+----------+

+------+---------+-----------------+----------+---------+------+
| e_no | e_name  | jobtitle        | joindate | sal     | d_no |
```

```
+------+---------+-----------------+----------+---------+------+
| 1005 | Mike    | Department Head |     2011 | 1000000 |    3 |
| 1010 | Clark   | CEO             |     2012 | 4000000 |    2 |
| 1011 | Pearson | CFO             |     2012 | 4000000 |    2 |
| 1050 | Harvey  | COO             |     2012 | 3000000 |    1 |
| 1080 | Donna   | Director        |     2015 |  500000 |    4 |
| 1100 | Louis   | HR              |     2016 | 2000000 |    3 |
| 2000 | Mabel   | Manager         |     2020 |   50000 |    4 |
| 2001 | Peter   | Manager         |     2020 |   60000 |    1 |
| 2023 | Steve   | Manager         |     2021 |   64000 |    3 |
| 2050 | Rachel  | Manager         |     2022 |   50000 |    4 |
| 2100 | Harold  | Manager         |     2024 |   40000 |    5 |
| 2110 | Marissa | Employee        |     2024 |   25000 |    5 |
| 2120 | Oliver  | Employee        |     2024 |   25000 |    5 |
+------+---------+-----------------+----------+---------+------+

+--------+
| e_name |
+--------+
| Donna  |
| Mabel  |
| Rachel |
+--------+

+--------+
| e_name |
+--------+
| Mike   |
| Louis  |
| Steve  |
+--------+

+--------+
| e_name |
+--------+
| Mabel  |
| Peter  |
| Steve  |
| Rachel |
| Harold |
+--------+

+---------+----------+
| e_name  | joindate |
+---------+----------+
```

```
| Pearson |     2012 |
| Peter   |     2020 |
+---------+----------+

+---------+---------+
| e_name  | sal     |
+---------+---------+
| Clark   | 4000000 |
| Pearson | 4000000 |
| Harvey  | 3000000 |
| Louis   | 2000000 |
| Mike    | 1000000 |
| Donna   |  500000 |
| Steve   |   64000 |
| Peter   |   60000 |
+---------+---------+

+---------+
| e_name  |
+---------+
| Mabel   |
| Peter   |
| Steve   |
| Rachel  |
| Harold  |
| Marissa |
| Oliver  |
+---------+

+---------+----------+
| e_name  | joindate |
+---------+----------+
| Pearson |     2012 |
| Peter   |     2020 |
+---------+----------+

+---------+
| e_name  |
+---------+
| Mike    |
| Pearson |
| Harvey  |
| Donna   |
| Louis   |
| Mabel   |
```

```
| Peter   |
| Steve   |
| Rachel  |
| Harold  |
| Marissa |
| Oliver  |
+---------+

+---------+----------+
| e_name  | joindate |
+---------+----------+
| Harold  |     2024 |
| Marissa |     2024 |
| Oliver  |     2024 |
| Rachel  |     2022 |
| Steve   |     2021 |
| Mabel   |     2020 |
| Peter   |     2020 |
| Louis   |     2016 |
| Donna   |     2015 |
| Clark   |     2012 |
| Pearson |     2012 |
| Harvey  |     2012 |
| Mike    |     2011 |
+---------+----------+

+----------------+
| Average_Salary |
+----------------+
|     30000.0000 |
+----------------+

+----------------+
| Average_Salary |
+----------------+
|   4000000.0000 |
+----------------+
```

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah **SAP ID:** 60018230034
Branch: Artificial Intelligence(AI) and Data Science **Div:** S
Course: Database Technologies **Roll No.:** S009

Experiment no. 5

Employee (ename, street, city)
Company (cname, city)
Works (ename, cname, sal)
Manages (ename, mgrname)

1. Find name and cities of all employees who work for Infosys
2. Find all employees who lives in the same city as their manager
3. Find all employees who earn more than every employee of TCS
4. Find the company that has smallest payroll

Code:

```
create database Experiment5;
use Experiment5;

create table employee (
      enamevarchar(50) PRIMARY KEY,
    street varchar(20),
    city varchar(20)
);

create table company (
      cnamevarchar(50) PRIMARY KEY,
    city varchar(20)
);

create table works (
      salint,
      enamevarchar(50) references employee(ename),
cnamevarchar(50) references company(cname)
);

create table manages (
      mgrnamevarchar(50),
```

```
enamevarchar(50) references employee(ename)
);

insert into employee values("ABC","S.V.Road","Mumbai");
insert into employee values("XYZ","ShaniwarWada","Pune");
insert into employee values("KLM","S.V.P.Road","Mumbai");
insert into employee values("PQR","M.G.Road","Pune");
insert into employee values("DEF","M.G.Road","Bangalore");
insert into employee values("GHI","M.G.Road","Mumbai");
select * from employee;

insert into company values("Infosys","Mumbai");
insert into company values("TCS","Pune");
insert into company values("Google","Bangalore");
select * from company;

insert into works values(50000,"ABC","Google");
insert into works values(80000,"XYZ","Infosys");
insert into works values(40000,"KLM","TCS");
insert into works values(40000,"PQR","TCS");
insert into works values(500000,"DEF","Infosys");
insert into works values(500000,"GHI","Google");
select * from works;

insert into manages values("DEF","XYZ");
insert into manages values("GHI","ABC");
select * from manages;


-- Query1
SELECT e.ename, e.city
FROM employee e
JOIN works w ON e.ename = w.ename
WHERE w.cname = 'Infosys';

--  Query2
SELECT e.ename
FROM employee e
JOIN manages m ON e.ename = m.ename
JOIN employee mgr ON m.mgrname = mgr.ename
WHERE e.city = mgr.city;

--  Query3
SELECT e.ename
FROM works w
```

```
JOIN employee e ON w.ename = e.ename
WHERE w.sal> ALL (
    SELECT w2.sal
    FROM works w2
    WHERE w2.cname = 'TCS'
);

--  Query4
SELECT cname
FROM works
GROUP BY cname
HAVING sum(sal) <= ALL(
     SELECT sum(sal)
    FROM works
    WHERE cname="Infosys"
);
```

Output:

```
+-------+--------------+-----------+
| ename | street       | city      |
+-------+--------------+-----------+
| ABC   | S.V.Road     | Mumbai    |
| DEF   | M.G.Road     | Bangalore |
| GHI   | M.G.Road     | Mumbai    |
| KLM   | S.V.P.Road   | Mumbai    |
| PQR   | M.G.Road     | Pune      |
| XYZ   | Shaniwar Wada| Pune      |
+-------+--------------+-----------+

+---------+-----------+
| cname   | city      |
+---------+-----------+
| Google  | Bangalore |
| Infosys | Mumbai    |
| TCS     | Pune      |
+---------+-----------+

+--------+-------+---------+
| sal    | ename | cname   |
+--------+-------+---------+
|  50000 | ABC   | Google  |
|  80000 | XYZ   | Infosys |
```

```
|  40000 | KLM   | TCS     |
|  40000 | PQR   | TCS     |
| 500000 | DEF   | Infosys |
| 500000 | GHI   | Google  |
+--------+-------+---------+
```

```
+---------+-------+
| mgrname | ename |
+---------+-------+
| DEF     | XYZ   |
| GHI     | ABC   |
+---------+-------+
```

```
+-------+-----------+
| ename | city      |
+-------+-----------+
| XYZ   | Pune      |
| DEF   | Bangalore |
+-------+-----------+
```

```
+-------+
| ename |
+-------+
| ABC   |
+-------+
```

```
+-------+
| ename |
+-------+
| ABC   |
| XYZ   |
| DEF   |
| GHI   |
+-------+
```

```
+---------+
| cname   |
+---------+
| TCS     |
+---------+
```

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah **SAP ID:** 60018230034
Branch: Artificial Intelligence(AI) and Data Science **Div:** S
Course: Database Technologies **Roll No.:** S009

Experiment no. 6

customer(c_no, c_name, c_add, c_age)
loan(c_no, l_no, loan_amt, l_date)

1. Create a view which includes the customer name whose loan amount is between 50000 and 100000.
2. Find the customer name from view whose loan amount is less than 80000.
3. Find the customer name whose name ends with letter 'm' and taken a loan amount as 90000.

Code:

```
create database Experiment6;
use Experiment6;

create table customer (
      c_no int PRIMARY KEY,
      c_name varchar(50) NOT NULL,
      c_add varchar(150) NOT NULL,
      c_age int NOT NULL
);

create table loan (
      c_no int references customer(c_no),
      l_no int PRIMARY KEY,
      loan_amt int NOT NULL,
      l_date varchar(10)
);

insert into customer values(1, 'Alice Johnson', '123 Maple St,
Springfield', 28);
insert into customer values(2, 'Bob Smith', '456 Oak Ave,
Shelbyville', 34);
insert into customer values(3, 'Charlie Brown', '789 Pine Rd, Capital
City', 22);
```

```
insert into customer values(4, 'Diana Prince', '101 Elm St, Gotham',
30);
insert into customer values(5, 'Edward Elric', '202 Birch Ln, Central
City', 26);
insert into customer values(6, 'Adam', '200 Birch Ln, Central City',
28);
select * from customer;

insert into loan values(1, 10001, 40000, '10/10/2023');
insert into loan values(2, 10002, 75000, '12/11/2023');
insert into loan values(3, 10003, 50000, '01/12/2023');
insert into loan values(4, 10004, 90000, '04/12/2023');
insert into loan values(5, 10005, 200000, '29/12/2023');
insert into loan values(6, 10006, 90000, '04/01/2024');
select * from loan;

-- Query 1
create view customer_loan_data
as (
      select c.c_no, c.c_name, l.loan_amt
    from customer c, loan l
    where (c.c_no=l.c_no) AND (l.loan_amt between 50000 and 100000)
);
select * from customer_loan_data;

-- Query 2
select c_name
from customer_loan_data
where loan_amt<80000;

-- Query 3
select c_name
from customer_loan_data
where (loan_amt=90000) AND (c_name like '%m');
```

Output:

Customer table:

	c_no	c_name	c_add	c_age
▶	1	Alice Johnson	123 Maple St, Springfield	28
	2	Bob Smith	456 Oak Ave, Shelbyville	34
	3	Charlie Brown	789 Pine Rd, Capital City	22
	4	Diana Prince	101 Elm St, Gotham	30
	5	Edward Elric	202 Birch Ln, Central City	26
	6	Adam	200 Birch Ln, Central City	28
*	NULL	NULL	NULL	NULL

Loan table:

	c_no	l_no	loan_amt	l_date
▶	1	10001	40000	10/10/2023
	2	10002	75000	12/11/2023
	3	10003	50000	01/12/2023
	4	10004	90000	04/12/2023
	5	10005	200000	29/12/2023
	6	10006	90000	04/01/2024
*	NULL	NULL	NULL	NULL

Query 1:

	c_no	c_name	loan_amt
▶	2	Bob Smith	75000
	3	Charlie Brown	50000
	4	Diana Prince	90000
	6	Adam	90000

Query 2:

	c_name
▶	Bob Smith
	Charlie Brown

Query 3:

	c_name
▶	Adam

Triggers

Code:

```
create database Expt6_Triggers;
use Expt6_Triggers;

create table Employee (
      e_id int PRIMARY KEY,
      e_name varchar(50) NOT NULL,
      e_sal int
);

create table audit (
      emp_id int,
      emp_name varchar(50),
      emp_sal int,
      audit_action varchar(100),
      audit_date date
);

insert into Employee values(1, 'Alice Johnson', 75000);
insert into Employee values(2, 'Bob Smith', 62000);
insert into Employee values(3, 'Charlie Brown', 85000);
insert into Employee values(4, 'Diana Prince', 78000);
insert into Employee values(5, 'Mark McGann', 96000);
select * from Employee;


DELIMITER //
create trigger audit_trigger after insert on Employee
for each row
BEGIN
      DECLARE emp_id int;
    DECLARE emp_name varchar(50);
    DECLARE emp_sal int;
    DECLARE audit_action varchar(100);

    SET emp_id = NEW.e_id;
    SET emp_name = NEW.e_name;
    SET emp_sal = NEW.e_sal;
    SET audit_action = 'New Record Inserted';
```

```
      insert into audit
values(emp_id,emp_name,emp_sal,audit_action,now());
END;
//

insert into Employee values(6,"Chaitanya",5000);
select * from Employee;
select * from audit;


DELIMITER //
create trigger audit_trigger_delete after delete on Employee
for each row
BEGIN
      DECLARE emp_id int;
    DECLARE emp_name varchar(50);
    DECLARE emp_sal int;
    DECLARE audit_action varchar(100);

    SET emp_id = OLD.e_id;
    SET emp_name = OLD.e_name;
    SET emp_sal = OLD.e_sal;
    SET audit_action = 'Record Deleted';

      insert into audit
values(emp_id,emp_name,emp_sal,audit_action,now());
END;
//

delete from Employee where e_id=2;
select * from Employee;
select * from audit;

DELIMITER //
create trigger audit_trigger_update after update on Employee
for each row
BEGIN
      DECLARE emp_id int;
    DECLARE emp_name varchar(50);
    DECLARE emp_sal int;
    DECLARE audit_action varchar(100);

    SET emp_id = NEW.e_id;
    SET emp_name = NEW.e_name;
    SET emp_sal = NEW.e_sal;
```

```
    SET audit_action = 'Record Updated';

     insert into audit
values(emp_id,emp_name,emp_sal,audit_action,now());
END;
//

update Employee set e_sal=80000 where e_id=4;
select * from Employee;
select * from audit;
```

Output:

e_id	e_name	e_sal
1	Alice Johnson	75000
3	Charlie Brown	85000
4	Diana Prince	78000
5	Mark McGann	96000
6	Chaitanya	5000
NULL	NULL	NULL

e_id	e_name	e_sal
1	Alice Johnson	75000
3	Charlie Brown	85000
4	Diana Prince	78000
5	Mark McGann	96000
6	Chaitanya	5000
NULL	NULL	NULL

emp_id	emp_name	emp_sal	audit_action	audit_date
6	Chaitanya	5000	New Record Inserted	2024-10-01
2	Bob Smith	62000	Record Deleted	2024-10-01

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah **SAP ID:** 60018230034
Branch: Artificial Intelligence(AI) and Data Science **Div:** S
Course: Database Technologies **Roll No.:** S009

Experiment no. 7

Procedures & Functions

Code:

```
create database Expt_7;
use Expt_7;

create table Student (
      s_rollnoint PRIMARY KEY,
s_namevarchar(50) NOT NULL,
s_ageint,
s_addvarchar(100)
);
desc Student;

insert into Student values(1, 'John Doe', 20, '123 Maple Street, New
York, NY 10001');
insert into Student values(2, 'Jane Smith', 21, '456 Oak Avenue, Los
Angeles, CA 90001');
insert into Student values(3, 'Michael Johnson', 22, '789 Pine Road,
Chicago, IL 60007');
insert into Student values(4, 'Emily Davis', 19, '321 Birch Lane,
Houston, TX 77001');
select * from Student;

-- Procedure 1: Get Student Name using Roll No.
delimiter //
create procedure get_student_name(in student_rollnoint)
begin
      select s_name
    from Student
    where s_rollno=student_rollno;
end
//

call get_student_name(2);
```

```
-- Procedure 2: Insert Data in Student Table using Procedure
delimiter //
create procedure insert_student_data(in std_rollnoint, std_name
varchar(50), std_ageint, std_add varchar(100))
begin
     insert into Student values(std_rollno, std_name, std_age,
std_add);
end
//

call insert_student_data(5, 'Max Payne', 20, '123 Main St,
Springfield, IL');
call insert_student_data(6, 'Keri Smith', 22, '456 Oak St, Rivertown,
TX');
select * from Student;


-- Functions
-- Factorial Function
delimiter //
create function factofnum(numint)
returns int
deterministic
begin
     declare iint default 1;
    declare fact int default 1;
    while(i<=num) do
           set fact=fact*i;
        set i=i+1;
     end while;
    return fact;
end;
//

select factofnum(5);
```

Output:

Student Table Description:

	Field	Type	Null	Key	Default	Extra
▶	s_rollno	int	NO	PRI	NULL	
	s_name	varchar(50)	NO		NULL	
	s_age	int	YES		NULL	
	s_add	varchar(100)	YES		NULL	

Student Table:

	s_rollno	s_name	s_age	s_add
▶	1	John Doe	20	123 Maple Street, New York, NY 10001
	2	Jane Smith	21	456 Oak Avenue, Los Angeles, CA 90001
	3	Michael Johnson	22	789 Pine Road, Chicago, IL 60007
	4	Emily Davis	19	321 Birch Lane, Houston, TX 77001
*	NULL	NULL	NULL	NULL

Procedure 1:

	s_name
▶	Jane Smith

Procedure 2:

Student Table:

	s_rollno	s_name	s_age	s_add
▶	1	John Doe	20	123 Maple Street, New York, NY 10001
	2	Jane Smith	21	456 Oak Avenue, Los Angeles, CA 90001
	3	Michael Johnson	22	789 Pine Road, Chicago, IL 60007
	4	Emily Davis	19	321 Birch Lane, Houston, TX 77001
	5	Max Payne	20	123 Main St, Springfield, IL
	6	Keri Smith	22	456 Oak St, Rivertown, TX
*	NULL	NULL	NULL	NULL

Factorial Function:

	factofnum(5)
▶	120

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah **SAP ID:** 60018230034
Branch: Artificial Intelligence(AI) and Data Science **Div:** S
Course: Database Technologies **Roll No.:** S009

Experiment no. 8

Transactions

Code:

```
create database Experiment8;
use Experiment8;

create table Employee (
      e_idint PRIMARY KEY,
e_namevarchar(50),
e_salint
);
desc Employee;

-- Create Transaction & Commit (Value is permanently added to table):
start transaction;
      insert into Employee values (1001, 'Max Payne', 6000000), (1002,
'Clark ', 6000000);
commit;
select * from Employee;

-- Rollback (Value is not updated in the table):
start transaction;
      update Employee
    set e_sal=8000000
    where e_id=1002;
rollback;
select * from Employee;

-- Commit (Value is updated in the table):
start transaction;
      update Employee
    set e_sal=8000000
    where e_id=1002;
commit;
select * from Employee;
```

```
-- Creating Checkpoints :
start transaction;
      update Employee
    set e_sal=5000000
    where e_id=10001;
      savepoint S1;

    delete from Employee where e_id=1001;
savepoint S2;
rollback to S1;
```

MySQL Command Line:

```
mysql> use Experiment8;
Database changed
mysql> set transaction isolation level read uncommitted;
Query OK, 0 rows affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> set session transaction isolation level read uncommitted;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from Employee;
+------+-----------+---------+
| e_id | e_name    | e_sal   |
+------+-----------+---------+
| 1001 | Max Payne | 6000000 |
| 1002 | Clark     | 4000000 |
+------+-----------+---------+
2 rows in set (0.00 sec)

mysql> update Employee set e_sal=6000000 where e_id=1002;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Employee;
+------+-----------+---------+
| e_id | e_name    | e_sal   |
+------+-----------+---------+
| 1001 | Max Payne | 6000000 |
| 1002 | Clark     | 6000000 |
+------+-----------+---------+
2 rows in set (0.00 sec)
```

```
mysql> commit;
Query OK, 0 rows affected (0.02 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> set session transaction isolation level read committed;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from Employee;
+------+-----------+---------+
| e_id | e_name    | e_sal   |
+------+-----------+---------+
| 1001 | Max Payne | 6000000 |
| 1002 | Clark     | 6000000 |
+------+-----------+---------+
2 rows in set (0.00 sec)

mysql> delete from Employee where e_id=1001;
Query OK, 1 row affected (0.01 sec)

mysql> select * from Employee;
+------+--------+---------+
| e_id | e_name | e_sal   |
+------+--------+---------+
| 1002 | Clark  | 6000000 |
+------+--------+---------+
1 row in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.03 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> set session transaction isolation level serializable;
Query OK, 0 rows affected (0.00 sec)

mysql> update Employee set e_sal=40000 where e_id=1002;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from Employee;
+------+--------+-------+
| e_id | e_name | e_sal |
+------+--------+-------+
| 1002 | Clark  | 40000 |
+------+--------+-------+
```

```
1 row in set (0.00 sec)
```

```
mysql> use Experiment8;
Database changed
mysql> set transaction isolation level read uncommitted;
Query OK, 0 rows affected (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> set session transaction isolation level read uncommitted;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from Employee;
+------+-----------+---------+
| e_id | e_name    | e_sal   |
+------+-----------+---------+
| 1001 | Max Payne | 6000000 |
| 1002 | Clark     | 4000000 |
+------+-----------+---------+
2 rows in set (0.00 sec)

mysql> select * from Employee;
+------+-----------+---------+
| e_id | e_name    | e_sal   |
+------+-----------+---------+
| 1001 | Max Payne | 6000000 |
| 1002 | Clark     | 4000000 |
+------+-----------+---------+
2 rows in set (0.01 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> set session transaction isolation level read committed;
Query OK, 0 rows affected (0.00 sec)

mysql> select * from Employee;
+------+-----------+---------+
| e_id | e_name    | e_sal   |
+------+-----------+---------+
```

```
| 1001 | Max Payne | 6000000 |
| 1002 | Clark     | 6000000 |
+------+-----------+---------+
2 rows in set (0.00 sec)

mysql> select * from Employee;
+------+--------+---------+
| e_id | e_name | e_sal   |
+------+--------+---------+
| 1002 | Clark  | 6000000 |
+------+--------+---------+
1 row in set (0.00 sec)

mysql> commit;
Query OK, 0 rows affected (0.00 sec)

mysql> start transaction;
Query OK, 0 rows affected (0.00 sec)

mysql> set session transaction isolation level serializable;
Query OK, 0 rows affected (0.00 sec)
```

Output:

Employee Table Description:

	Field	Type	Null	Key	Default	Extra
▶	e_id	int	NO	PRI	NULL	
	e_name	varchar(50)	YES		NULL	
	e_sal	int	YES		NULL	

Transaction &Commit:

	e_id	e_name	e_sal
▶	1001	Max Payne	6000000
	1002	Clark	6000000
*	NULL	NULL	NULL

After Rollback:

	e_id	e_name	e_sal
▶	1001	Max Payne	6000000
	1002	Clark	6000000
*	NULL	NULL	NULL

After Commit:

e_id	e_name	e_sal
1001	Max Payne	6000000
1002	Clark	8000000
NULL	NULL	NULL

Rollback to Checkpoint 1:

e_id	e_name	e_sal
1001	Max Payne	6000000
1002	Clark	8000000
NULL	NULL	NULL

Rollback to Checkpoint 2:

e_id	e_name	e_sal
1002	Clark	8000000
NULL	NULL	NULL

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah
SAP ID: 60018230034
Branch: Artificial Intelligence(AI) and Data Science
Div: S
Course: Database Technologies
Roll No.: S009

Experiment no. 9

ETL Pipeline (Python & MySQL)

Code:

```
MySQL Code:
create database Experiment9;
use Experiment9;

create table customercleaned (
      customer_idint,
first_namevarchar(50),
last_namevarchar(50),
    country varchar(50),
    email varchar(100)
);

select * from customercleaned;

Python Code:
!pip install mysql-connector-python

import mysql.connector
import pandas as pd

conn =
mysql.connector.connect(user='root',password='pass@123',host='localhos
t',database='Experiment9')
cursor=conn.cursor()

data = pd.read_csv('customer1.csv')
data.head()

data.fillna('',inplace=True)
data.drop_duplicates(inplace=True)

data = data[data['email'].str.contains('@',na=False)]
```

```
print("Data transformed successfully.")

cols = ",".join([str(i) for i in data.columns.tolist()])

for i,row in data.iterrows():
    sql = f""" INSERT INTO customercleaned({cols})
        VALUES (%s, %s, %s, %s, %s)
        ON DUPLICATE KEY UPDATE
        first_name = VALUES(first_name),
        last_name = VALUES(last_name),
        country = VALUES(country),
        email = VALUES(email)"""
    cursor.execute(sql,(row['customer_Id'],row['first_name'],row['last
_name'],row['country'],row['email']))


conn.commit()

cursor.close()

conn.close()

print("Data successfully stored in the 'customercleaned' table.")

output_file = 'transformed_customer.csv'
data.to_csv(output_file, index=False)

print(f"Data loaded into CSV file : {output_file}")
transformed_data = pd.read_csv(output_file)

# Display the first few rows (by default, it shows 5 rows)
print(transformed_data.head())

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10,6))
sns.countplot(x='country', data=data, palette='viridis')
plt.title("Number of Customers per Country")
plt.xlabel('Country')
plt.ylabel('Count of Customers')
plt.xticks(rotation=45)
plt.show()
```

ipynb file:

DBT Experiment 9

ETL (Extract Transform Load)

In [1]:
```
!pip install mysql-connector-python
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: mysql-connector-python in c:\users\djsce.student\appdata\roaming\python\python311\site-packages
(9.1.0)
```

In [2]:
```
import mysql.connector
import pandas as pd
```

Extract :

In [3]:
```
conn = mysql.connector.connect(user='root',password='pass@123',host='localhost',database='Experiment9')
cursor=conn.cursor()
```

In [4]:
```
data = pd.read_csv('customer1.csv')
data.head()
```

Out[4]:

	customer_id	first name	last_name	country	email
0	101	Sheryl	Baxter	Chile	zunigavanessa@smith.info
1	102	Preston	Lozano	Djibouti	vmata@colon.com
2	103	Roy	Berry	Antigua and Barbuda	beckycarr@hogan.com
3	104	Linda	Olsen	Dominican Republic	stanleyblackwell@benson.org
4	105	Joanna	Bender	Slovakia (Slovak Republic)	colinalvarado@miles.net

Transform :

In [5]:
```
data.fillna('',inplace=True)
data.drop_duplicates(inplace=True)
```

```
data = data[data['email'].str.contains('@',na=False)]

print("Data transformed successfully.")
```

```
Data transformed successfully.
```

In [6]:
```
cols = ",".join([str(i) for i in data.columns.tolist()])
```

Load :

In [7]:
```
for i,row in data.iterrows():
    sql = f""" INSERT INTO customercleaned({cols})
        VALUES (%s, %s, %s, %s, %s)
        ON DUPLICATE KEY UPDATE
        first_name = VALUES(first_name),
        last_name = VALUES(last_name),
        country = VALUES(country),
        email = VALUES(email)"""
    cursor.execute(sql,(row['customer_Id'],row['first_name'],row['last_name'],row['country'],row['email']))
```

In [8]:
```
conn.commit()
```

In [9]:
```
cursor.close()
```

Out[9]:
```
True
```

In [10]:
```
conn.close()
```

In [11]:
```
print("Data successfully stored in the 'customercleaned' table.")
```

```
Data successfully stored in the 'customercleaned' table.
```

Output File & Visualisation :

In [12]:
```
output_file = 'transformed_customer.csv'
data.to_csv(output_file, index=False)
```

In [13]:
```
print(f"Data loaded into CSV file : {output_file}")
transformed_data = pd.read_csv(output_file)
```

```
Data loaded into CSV file : transformed_customer.csv
```

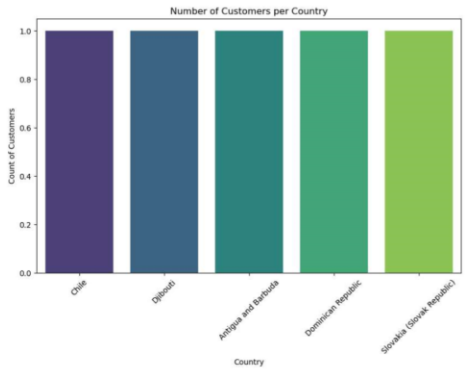
In [14]:
```
# Display the first few rows (by default, it shows 5 rows)
print(transformed_data.head())
```

```
   customer_Id first_name last_name                    country  \
0          101     Sheryl    Baxter                      Chile
1          102    Preston    Lozano                   Djibouti
2          103        Roy     Berry        Antigua and Barbuda
3          104      Linda     Olsen         Dominican Republic
4          105     Joanna    Bender  Slovakia (Slovak Republic)

                        email
0     zunigavanessa@smith.info
1              vmata@colon.com
2          beckycarr@hogan.com
3  stanleyblackwell@benson.org
4      colinalvarado@miles.net
```

In [15]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [16]:
```
plt.figure(figsize=(10,6))
sns.countplot(x='country', data=data, palette='viridis')
plt.title("Number of Customers per Country")
plt.xlabel('Country')
plt.ylabel('Count of Customers')
plt.xticks(rotation=45)
plt.show()
```



Output:

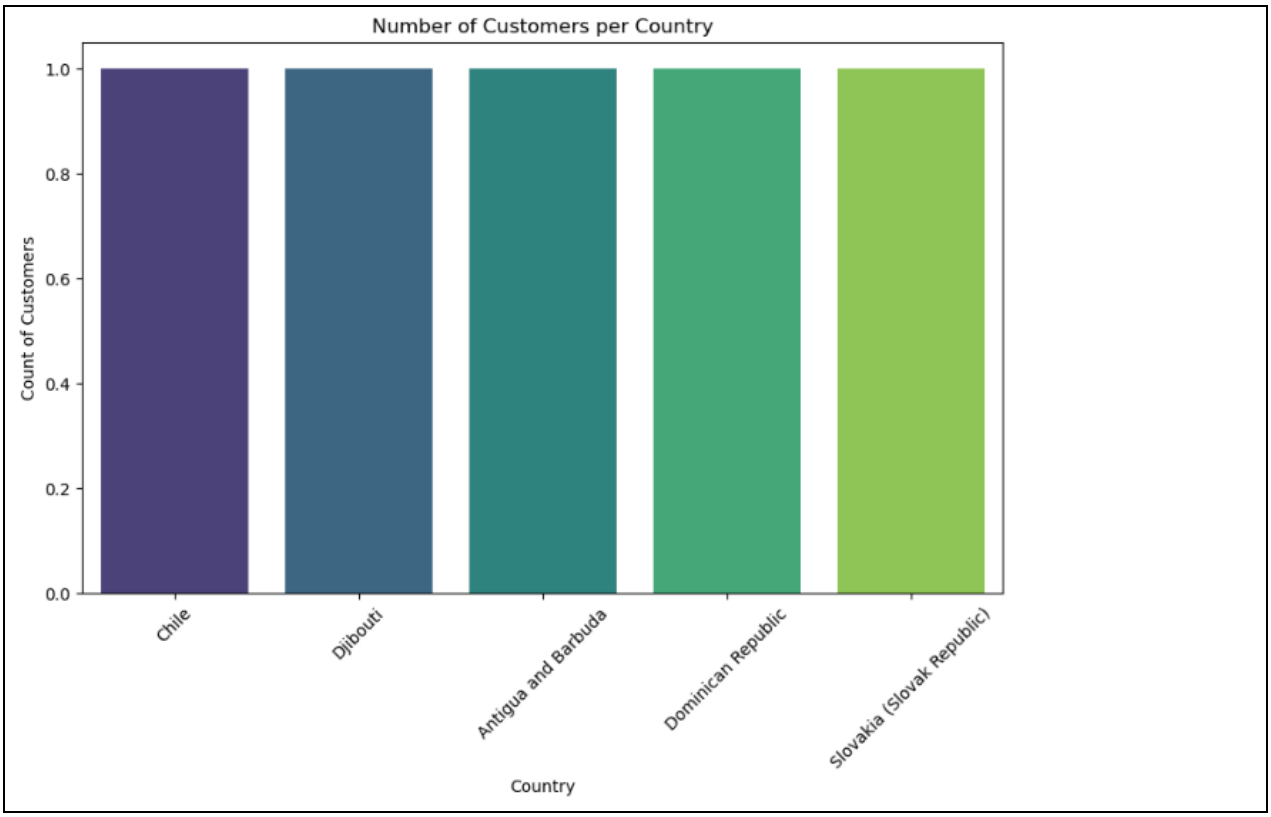
customercleaned table:

	customer_id	first_name	last_name	country	email
▶	101	Sheryl	Baxter	Chile	zunigavanessa@smith.info
	102	Preston	Lozano	Djibouti	vmata@colon.com
	103	Roy	Berry	Antigua and Barbuda	beckycarr@hogan.com
	104	Linda	Olsen	Dominican Republic	stanleyblackwell@benson.org
	105	Joanna	Bender	Slovakia (Slovak Republic)	colinalvarado@miles.net

transformed_customer.csv:

customer_Id	first_name	last_name	country	email
101	Sheryl	Baxter	Chile	zunigavanessa@smith.info
102	Preston	Lozano	Djibouti	vmata@colon.com
103	Roy	Berry	Antigua and Barbuda	beckycarr@hogan.com
104	Linda	Olsen	Dominican Republic	stanleyblackwell@benson.org
105	Joanna	Bender	Slovakia (Slovak Republic)	colinalvarado@miles.net

visualisation:


Number of Customers per Country
Count of Customers
1.0
0.8
0.6
0.4
0.2
0.0
Chile
Djibouti
Antigua and Barbuda
Dominican Republic
Slovakia (Slovak Republic)
Country

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah
SAP ID: 60018230034
Branch: Artificial Intelligence(AI) and Data Science
Div: S
Course: Database Technologies
Roll No.: S009

Experiment no. 10

Data Warehousing& OLAP

Code:

```
import pandas aspd
```

```
data = {'Year':[2021,2021,2022,2023],
'Region':['North','South','North','South'],
'Product':['A','B','A','B'],
'Sales':[100,200,300,400]}
```

```
df = pd.DataFrame(data)
pivot_table = pd.pivot_table(df,
values='Sales',
index=['Year','Region'],
columns=['Product'],
aggfunc='sum')
print(pivot_table)
```

Output:

```
Product          A      B
Year Region
2021 North   100.0    NaN
     South     NaN  200.0
2022 North   300.0    NaN
2023 South     NaN  400.0
```

```
pivot_table
```

Output:

	Product	A	B
Year	Region		
2021	North	100.0	NaN
	South	NaN	200.0
2022	North	300.0	NaN
2023	South	NaN	400.0

```
importnumpyas np
```

```
data_cube = np.array([[[100,200],[150,250]],[[300,400],[350,450]]])
```

```
# Labels for reference
years = [2021,2022]
regions = ['North','South']
products = ['A','B']
print("Data Cube:\n",data_cube)
```

Output:

```
Data Cube:
 [[[100 200]
  [150 250]]

 [[300 400]
  [350 450]]]
```

OLAP:

```
year_2022 = data_cube[1,:,:]      # Select 2022 data
print(f"Drill-Down: Sales for Year 2022 (Product by Region):\n{year_2022}")

year_2021_total_by_region = np.sum(data_cube[0,:,:], axis=1)
print(f"\nRoll-Up: Total Sales by Region for Year
2021:\n{year_2021_total_by_region}")
```

```
product_a_sales = data_cube[1,:,0]
print(f"\nSlice: Sales for Product A across years and
regions:\n{product_a_sales}")

north_sales = data_cube[:,0,:]
print(f"\nDice: Sales for north Region across both years and
products:\n{north_sales}")
```

Output:

```
Drill-Down: Sales for Year 2022 (Product by Region):
[[300 400]
 [350 450]]

Roll-Up: Total Sales by Region for Year 2021:
[300 400]

Slice: Sales for Product A across years and regions:
[300 350]

Dice: Sales for north Region across both years and products:
[[100 200]
 [300 400]]
```

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah
SAP ID: 60018230034
Branch: Artificial Intelligence(AI) and Data Science
Div: S
Course: Database Technologies
Roll No.: S009

Experiment no. 11

MongoDB (NoSQL)

Code:

MongoDB Shell:

Create & Use Database:

```
use Experiment11
```

```
switched to db Experiment11
```

Create Collection:

```
db.createCollection('Emp')
```

```
{ ok: 1 }
```

Insert Data in collection:

```
db.Emp.insertMany(
[
  {
    "eid": 1,
    "ename": "Alice Johnson",
    "eage": 30,
    "esal": 60000
  },
  {
    "eid": 2,
    "ename": "Bob Smith",
    "eage": 25,
    "esal": 50000
  },
  {
    "eid": 3,
    "ename": "Charlie Brown",
    "eage": 35,
    "esal": 75000
  },
  {
    "eid": 4,
```

```
  "ename": "Diana Prince",
  "eage": 28,
  "esal": 55000
 }
]
)
```

```
< {
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('67176d995ddb001d75ad89b5'),
    '1': ObjectId('67176d995ddb001d75ad89b6'),
    '2': ObjectId('67176d995ddb001d75ad89b7'),
    '3': ObjectId('67176d995ddb001d75ad89b8')
  }
}
```

Display data from collection:

```
db.Emp.find()
```

```
< {
    _id: ObjectId('67176d995ddb001d75ad89b5'),
    eid: 1,
    ename: 'Alice Johnson',
    eage: 30,
    esal: 60000
  }
  {
    _id: ObjectId('67176d995ddb001d75ad89b6'),
    eid: 2,
    ename: 'Bob Smith',
    eage: 25,
    esal: 50000
  }
  {
    _id: ObjectId('67176d995ddb001d75ad89b7'),
    eid: 3,
    ename: 'Charlie Brown',
    eage: 35,
    esal: 75000
  }
  {
    _id: ObjectId('67176d995ddb001d75ad89b8'),
    eid: 4,
    ename: 'Diana Prince',
    eage: 28,
    esal: 55000
  }
```

Find specific data:

```
db.Emp.find({"ename":"Charlie Brown"})
```

```
< {
    _id: ObjectId('67176d995ddb001d75ad89b7'),
    eid: 3,
    ename: 'Charlie Brown',
    eage: 35,
    esal: 75000
  }
```

Update data:

```
db.Emp.update({"eid":1},{$set:{"esal":75000}})
```

```
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
```

```
db.Emp.find({"eid":1})
```

```
< {
    _id: ObjectId('67176d995ddb001d75ad89b5'),
    eid: 1,
    ename: 'Alice Johnson',
    eage: 30,
    esal: 75000
  }
```

Delete single data:

```
db.Emp.deleteOne({"eid":3})
```

```
< {
    acknowledged: true,
    deletedCount: 1
  }
```

```
db.Emp.find().pretty()
```

```
< {
    _id: ObjectId('67176d995ddb001d75ad89b5'),
    eid: 1,
    ename: 'Alice Johnson',
    eage: 30,
    esal: 75000
  }
  {
    _id: ObjectId('67176d995ddb001d75ad89b6'),
    eid: 2,
    ename: 'Bob Smith',
    eage: 25,
    esal: 50000
  }
  {
    _id: ObjectId('67176d995ddb001d75ad89b8'),
    eid: 4,
    ename: 'Diana Prince',
    eage: 28,
    esal: 55000
  }
```

Aggregation:

```
db.Emp.aggregate([{$group:{_id:"$eage", same_age:{$sum:1}}}])
```

```
< {
    _id: 25,
    same_age: 1
  }
  {
    _id: 28,
    same_age: 1
  }
  {
    _id: 30,
    same_age: 1
  }
```

Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Name: Chaitanya Shah
SAP ID: 60018230034
Branch: Artificial Intelligence(AI) and Data Science
Div: S
Course: Database Technologies
Roll No.: S009

Experiment no. 12

Graph Database (Neo4j)

Question:

Create two Person nodes for "Alice" (Employee ID: 101, Position: "Engineer") and "Bob" (Employee ID: 102, Position: "Manager"), and one Project node called "Alpha" (Project ID: 201). Then, perform the following in a single query:

1. Create a WORKS_ON relationship between both Alice and Bob with the Project "Alpha."
2. Change Alice's position to "Senior Engineer" and set hours_per_week to 40 on her WORKS_ON relationship with Alpha.
3. Retrieve all Person nodes along with their positions and associated projects.
4. Delete Bob and any relationships connected to him.
5. Retrieve all employees who are working on a project for more than 30 hours per week.

Code:

Create :-

(Q.1)

```
create(alice: person{name:'Alice',emp_id:101,position:'Engineer'}),(bob: person{name:'Bob',emp_id:102,position:'Manager'}),(alpha:project{pname:'Alpha',p_id:201}),(alice)-[r:works_on{hours_per_week:0}]->(alpha),(bob)-[r1:works_on{hours_per_week:0}]->(alpha);
```

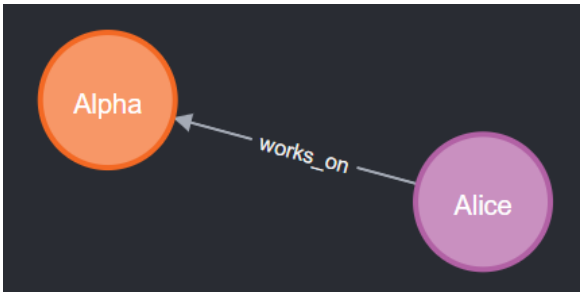
Output:

```
Added 3 labels, created 3 nodes, set 10 properties, created 2 relationships, completed after 3 ms.
```

Retrieve Data :-

```
match(p:person)-[rel:works_on]->(proj:project)
return p,rel,proj;
```

Output:



project

<elementId>	4:e7d18714-4d86-4888-90a3-6d9dde3129fa:23
<id>	23
p_id	201
pname	Alpha

works_on

<elementId>	5:e7d18714-4d86-4888-90a3-6d9dde3129fa:1152921504606846997
<id>	1152921504606846997
hours_per_week	40

person

<elementId>	4:e7d18714-4d86-4888-90a3-6d9dde3129fa:21
<id>	21
emp_id	101
name	Alice
position	Senior Engineer

Q.2 :-

```
match(a:person{name:'Alice'})-[r:works_on]->(alpha:project{pname:'Alpha'})
set a.position="Senior Engineer",r.hours_per_week=40;
```

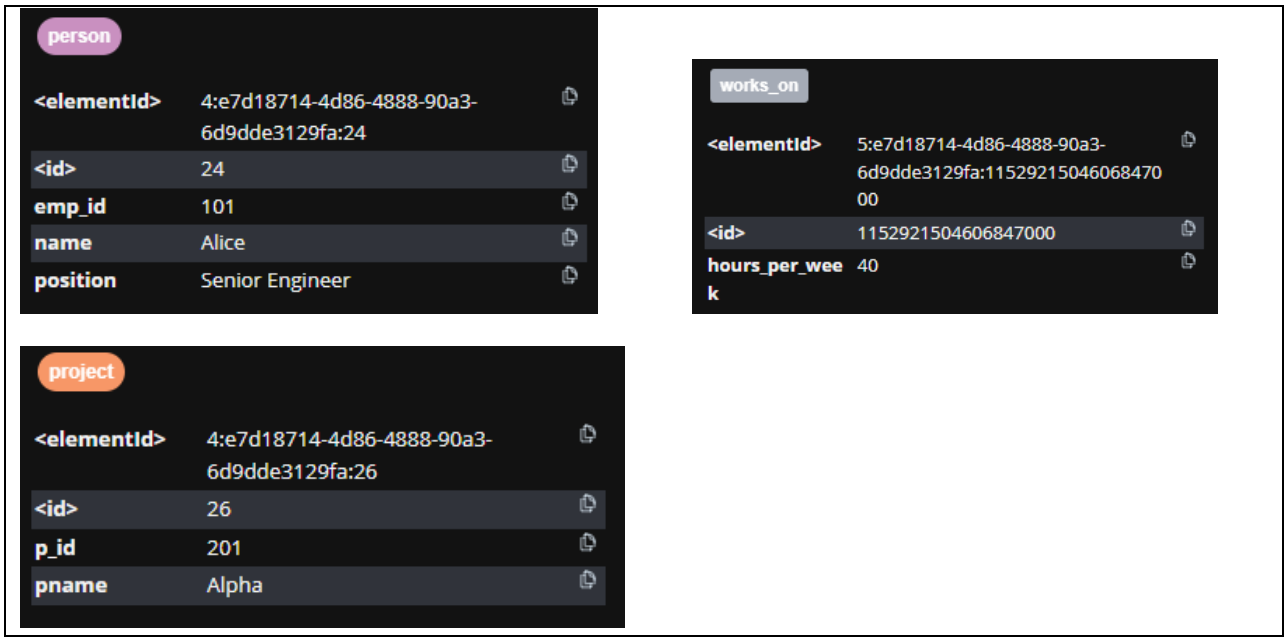
Output:

Set 22 properties, completed after 12 ms.

Q.3 :-

```
match(a:person{name:'Alice'})-[r:works_on]->(alpha:project{pname:'Alpha'})
return a,r,alpha;
```

Output:



Q.4 :-

```
match(b:person{name:'Bob'}) detach delete b;
```

Output:

Deleted 1 node, deleted 1 relationship, completed after 3 ms.

Q.5 :-

```
match(p:person)-[r:works_on]->(proj:project)
where r.hours_per_week>30
return p.pname as employee,proj.pname as project;
```

Output:

employee	project
null	"Alpha"

Full Code:

```
create(alice: person{name:'Alice',emp_id:101,position:'Engineer'}),(bob: person{name:'Bob',emp_id:102,position:'Manager'}),(alpha:project{pname:'Alpha',p_id:201}),(alice)-[r:works_on{hours_per_week:0}]->(alpha),(bob)-[r1:works_on{hours_per_week:0}]->(alpha);

match(p:person)-[rel:works_on]->(proj:project)
return p,rel,proj;

match(a:person{name:'Alice'})-[r:works_on]->(alpha:project{pname:'Alpha'})
set a.position="Senior Engineer",r.hours_per_week=40;

match(a:person{name:'Alice'})-[r:works_on]->(alpha:project{pname:'Alpha'})
return a,r,alpha;

match(b:person{name:'Bob'}) detach delete b;

match(p:person)-[r:works_on]->(proj:project)
where r.hours_per_week>30
return p.pname as employee,proj.pname as project;
```