Lab Session 04

4. Basic SQL Queries

Aim:

The aim of this experiment is to familiarize students with the syntax and usage of these join operations, enabling them to effectively combine data from multiple tables based on specified conditions.

Description:

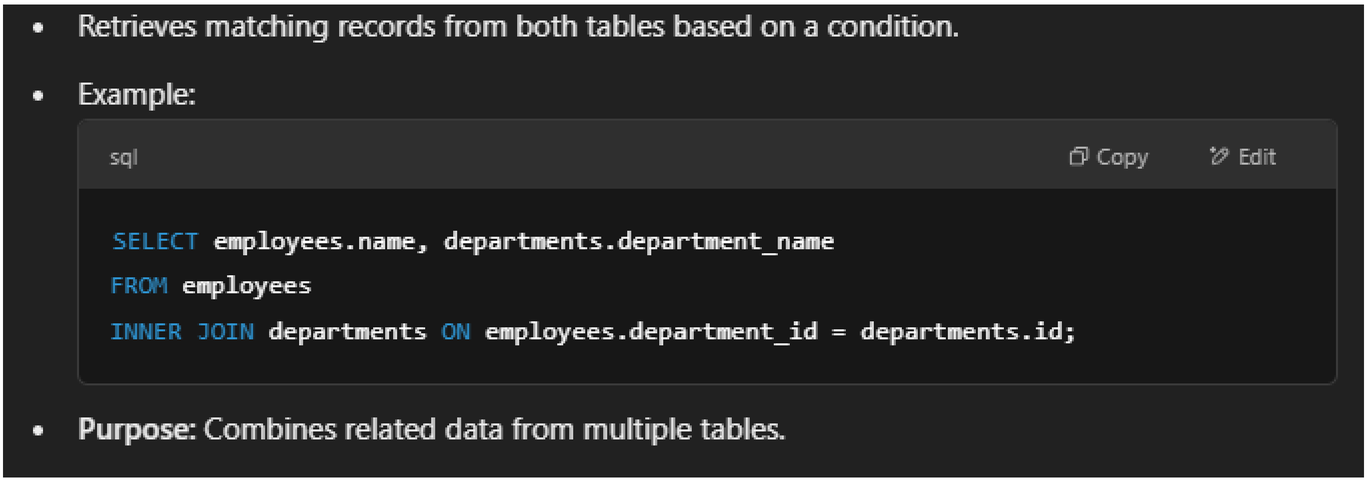
The "Implementation of Joins: Inner Join, Outer Join, Natural Join" lab experiment involves a procedural description where students gain hands-on experience in executing different types of joins in a database system. The lab begins with a brief introduction to the concepts of joins, emphasizing Inner Join, Outer Join, and Natural Join. Students then proceed to practice implementing these join operations using SQL queries. They learn how to write queries that combine data from multiple tables based on specified conditions, such as matching values in related columns. The lab provides students with sample datasets and real-world scenarios, enabling them to apply the appropriate join type to retrieve the desired results. Through this procedural description, students enhance their understanding of join operations and develop the skills needed to effectively query and analyze relational databases using Inner Join, Outer Join, and Natural Join.

Pre-Requisites:

PostgreSQL, TerraER Tool, Windows/ Ubuntu/CentOS/Debian, DBMS Concepts

Pre Lab-Task:

1) Explain the concept of Inner Join in PostgreSQL. How does it work, and what is its purpose?

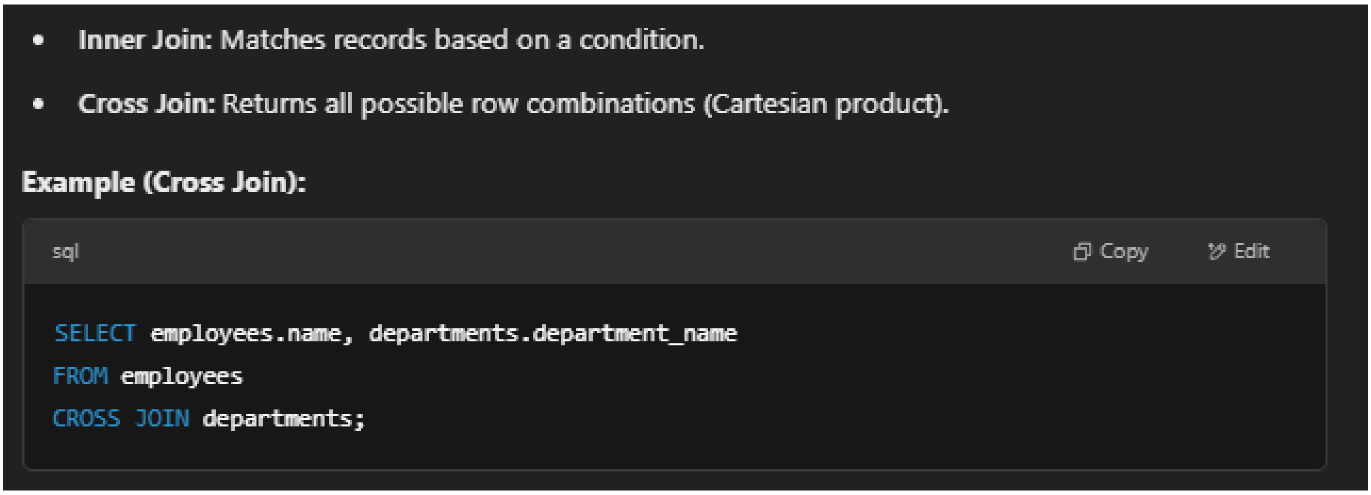


- Retrieves matching records from both tables based on a condition.
- Example:

```sql
SELECT employees.name, departments.department_name
FROM employees
INNER JOIN departments ON employees.department_id = departments.id;
```

- **Purpose:** Combines related data from multiple tables.

2) What are the key differences between an Inner Join and a Cross Join in PostgreSQL?

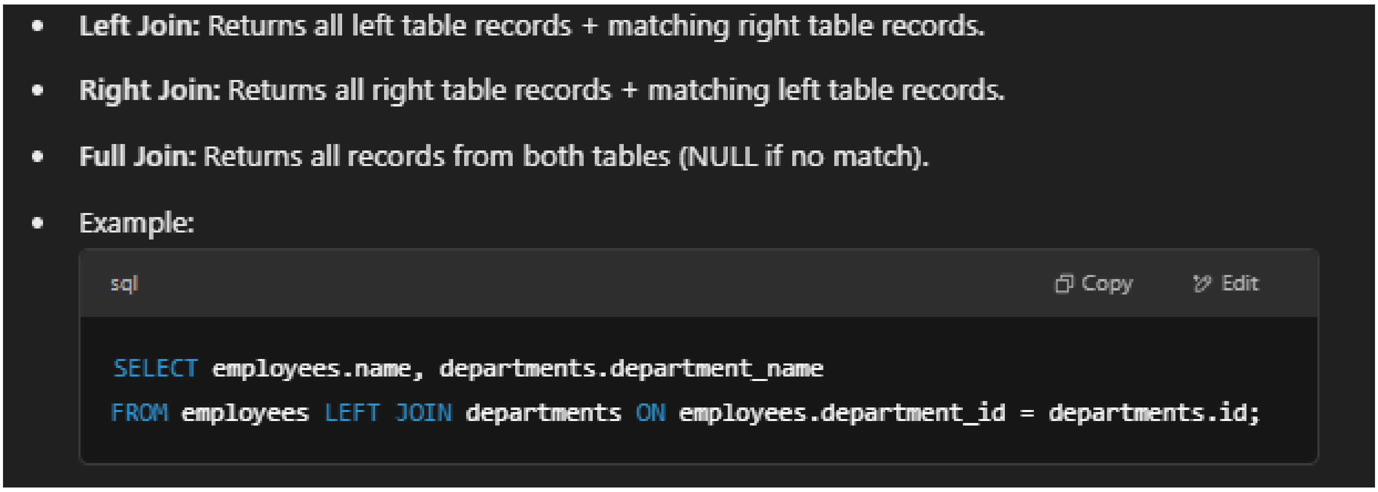


- **Inner Join:** Matches records based on a condition.
- **Cross Join:** Returns all possible row combinations (Cartesian product).

Example (Cross Join):

```sql
SELECT employees.name, departments.department_name
FROM employees
CROSS JOIN departments;
```

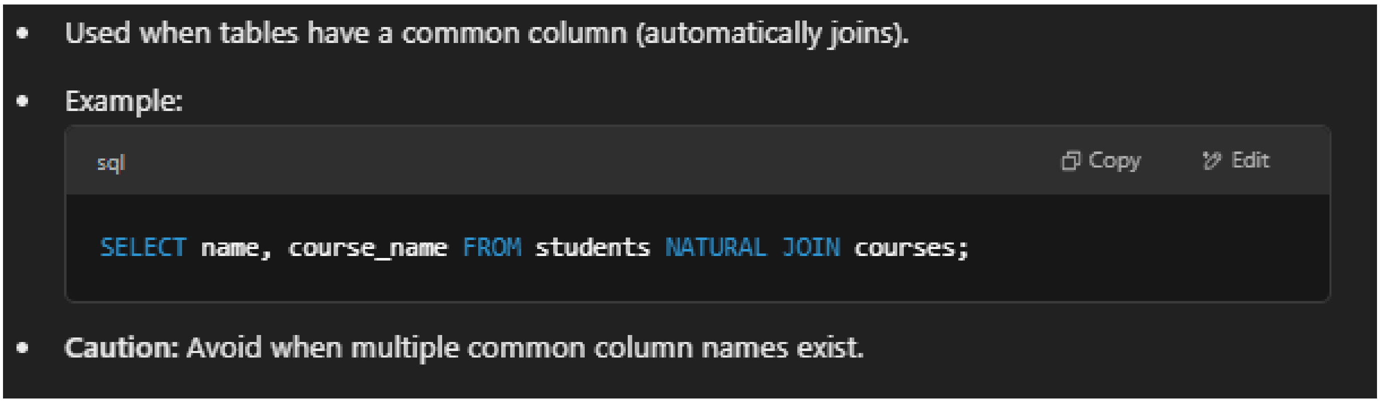
3) Explain the types of Outer Joins supported by PostgreSQL: Left Outer Join, Right Outer Join, and Full Outer Join.



- **Left Join:** Returns all left table records + matching right table records.
- **Right Join:** Returns all right table records + matching left table records.
- **Full Join:** Returns all records from both tables (NULL if no match).
- Example:

```sql
SELECT employees.name, departments.department_name
FROM employees LEFT JOIN departments ON employees.department_id = departments.id;
```

4) When would you use a Natural Join in PostgreSQL? Give an example scenario where a Natural Join is appropriate.

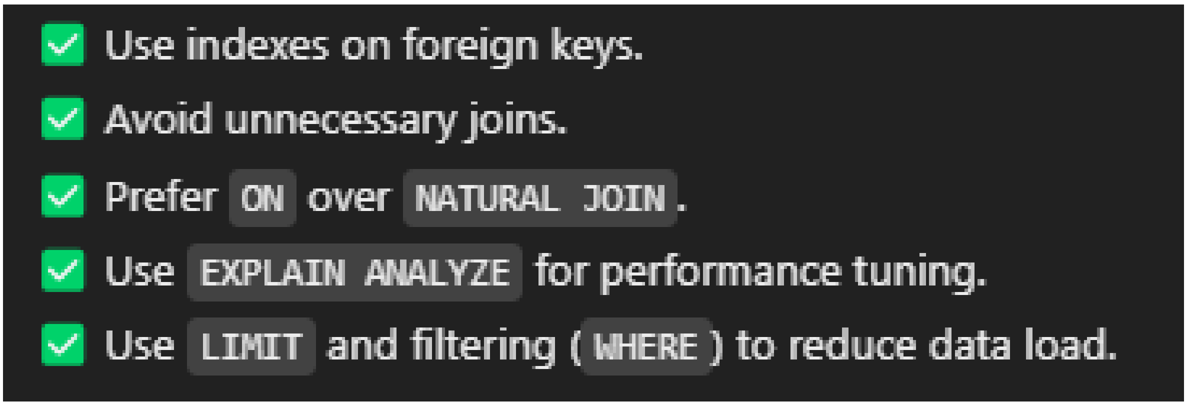


- Used when tables have a common column (automatically joins).
- Example:

```sql
SELECT name, course_name FROM students NATURAL JOIN courses;
```

- **Caution:** Avoid when multiple common column names exist.

5) What are some best practices when working with Joins in PostgreSQL to optimize performance and avoid common pitfalls?



✅ Use indexes on foreign keys.

✅ Avoid unnecessary joins.

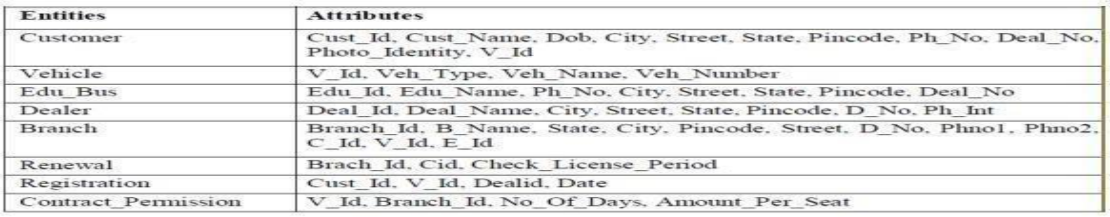
✅ Prefer `ON` over `NATURAL JOIN`.

✅ Use `EXPLAIN ANALYZE` for performance tuning.

✅ Use `LIMIT` and filtering (`WHERE`) to reduce data load.

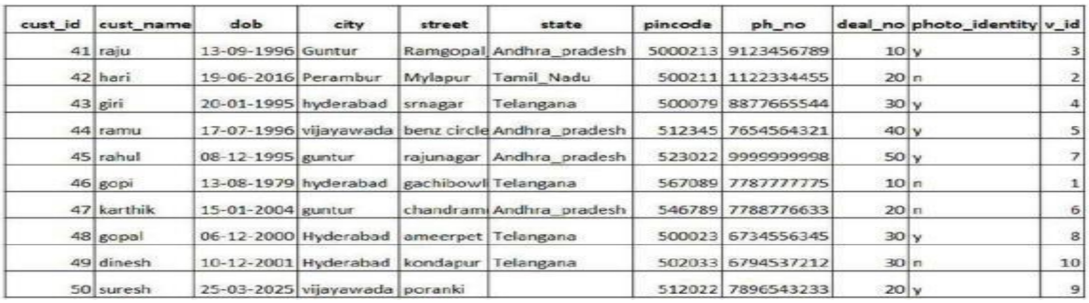
In Lab Task:

1. Implementation of distinct types of Joins: Inner Join, Outer Join, Natural Join etc.



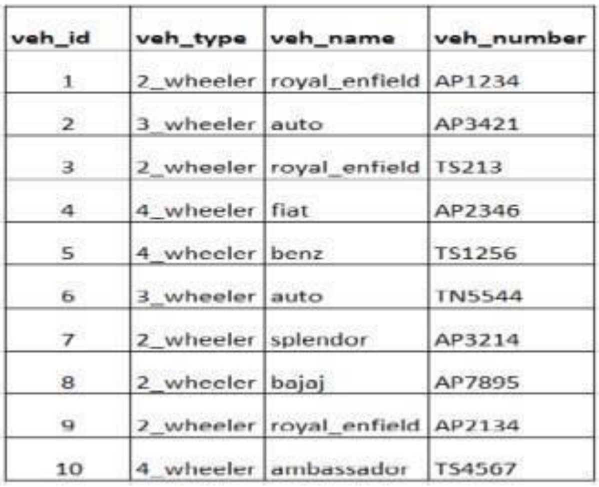
Entities	Attributes
Customer	Cust_Id, Cust_Name, Dob, City, Street, State, Pincode, Ph_No, Deal_No, Photo_Identity, V_Id
Vehicle	V_Id, Veh_Type, Veh_Name, Veh_Number
Edu_Bus	Edu_Id, Edu_Name, Ph_No, City, Street, State, Pincode, Deal_No
Dealer	Deal_Id, Deal_Name, City, Street, State, Pincode, D_No, Ph_Int
Branch	Branch_Id, B_Name, State, City, Pincode, Street, D_No, Phno1 , Phno2, C_Id, V_Id, E_Id
Renewal	Brach_Id, Cid, Check_License_Period
Registration	Cust_Id, V_Id, Dealid, Date
Contract_Permission	V_Id, Branch_Id, No_Of_Days, Amount_Per_Seat

CUSTOMER



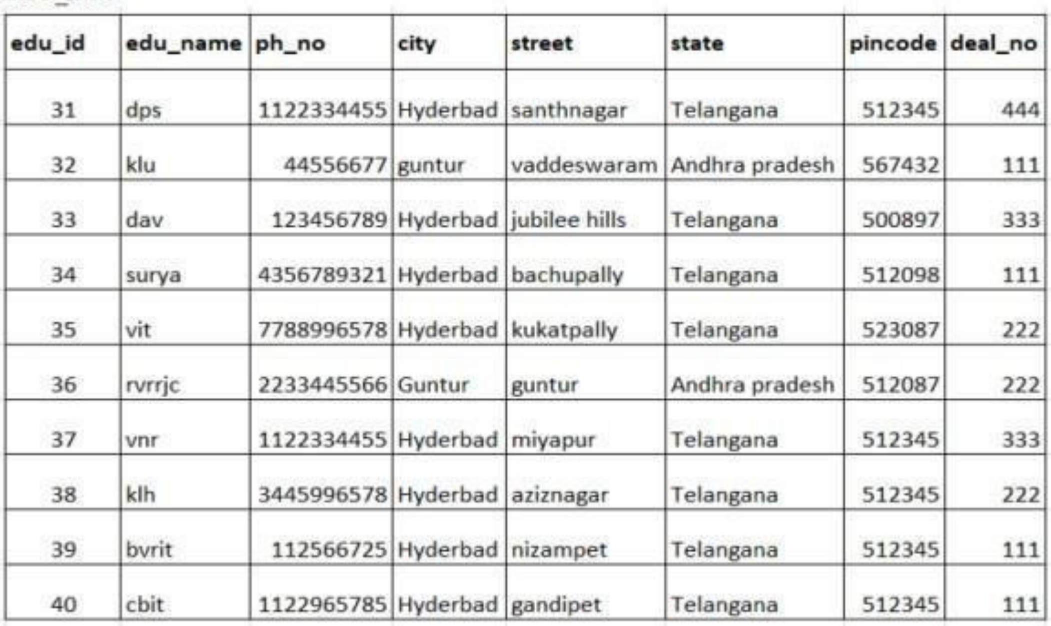
cust_id	cust_name	dob	city	street	state	pincode	ph_no	deal_no	photo_identity	v_id
41	raju	13-09-1996	Guntur	Ramgopal	Andhra_pradesh	5000213	9123456789	10	y	3
42	hari	19-06-2016	Perambur	Mylapur	Tamil_Nadu	500211	1122334455	20	n	2
43	giri	20-01-1995	hyderabad	srnagar	Telangana	500079	8877665544	30	y	4
44	ramu	17-07-1996	vijayawada	benz circle	Andhra_pradesh	512345	7654564321	40	y	5
45	rahul	08-12-1995	guntur	rajunagar	Andhra_pradesh	523022	9999999998	50	y	7
46	gopi	13-08-1979	hyderabad	gachibowl	Telangana	567089	7787777775	10	n	1
47	karthik	15-01-2004	guntur	chandram	Andhra_pradesh	546789	7788776633	20	n	6
48	gopal	06-12-2000	Hyderabad	ameerpet	Telangana	500023	6734556345	30	y	8
49	dinesh	10-12-2001	Hyderabad	kondapur	Telangana	502033	6794537212	30	n	10
50	suresh	25-03-2025	vijayawada	poranki		512022	7896543233	20	y	9

VEHICLE



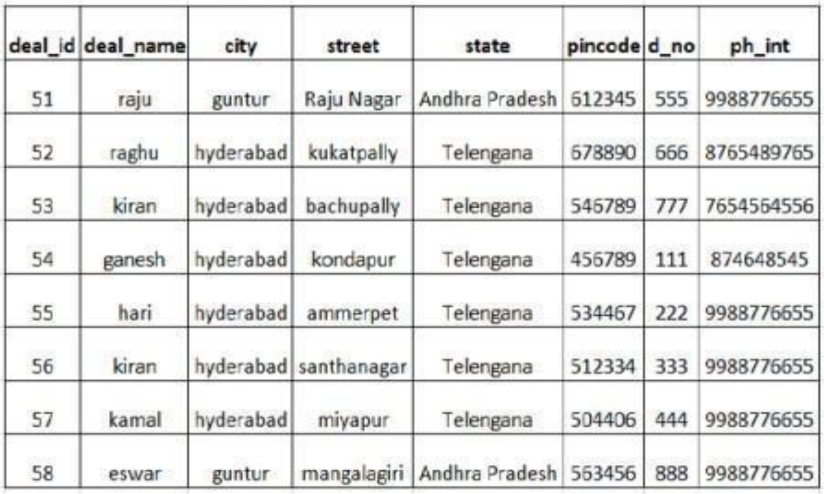
veh_id	veh_type	veh_name	veh_number
1	2_wheeler	royal_enfield	AP1234
2	3_wheeler	auto	AP3421
3	2_wheeler	royal_enfield	TS213
4	4_wheeler	fiat	AP2346
5	4_wheeler	benz	TS1256
6	3_wheeler	auto	TN5544
7	2_wheeler	splendor	AP3214
8	2_wheeler	bajaj	AP7895
9	2_wheeler	royal_enfield	AP2134
10	4_wheeler	ambassador	TS4567

EDU_BUS



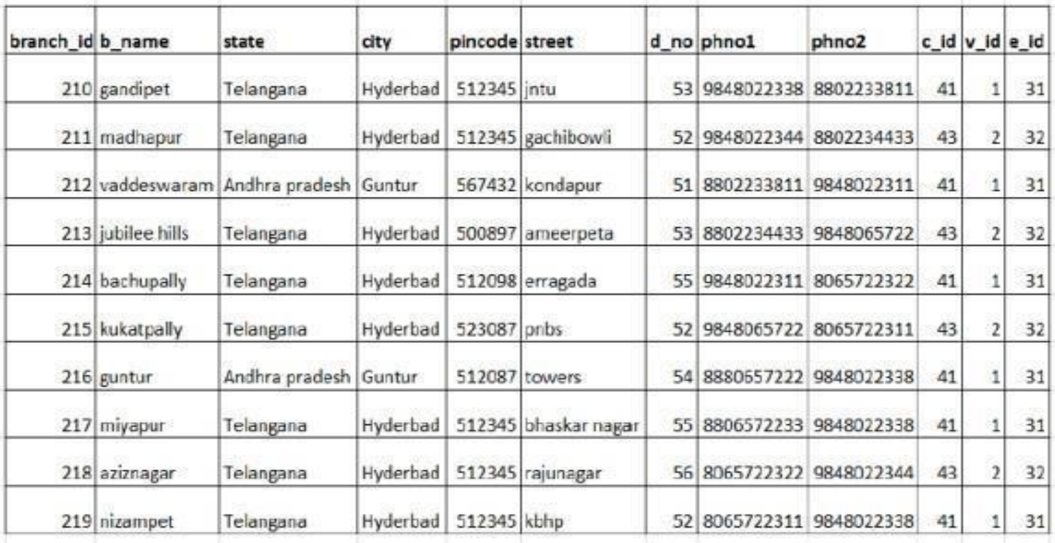
edu_id	edu_name	ph_no	city	street	state	pincode	deal_no
31	dps	1122334455	Hyderbad	santhnagar	Telangana	512345	444
32	klu	44556677	guntur	vaddeswaram	Andhra pradesh	567432	111
33	dav	123456789	Hyderbad	jubilee hills	Telangana	500897	333
34	surya	4356789321	Hyderbad	bachupally	Telangana	512098	111
35	vit	7788996578	Hyderbad	kukatpally	Telangana	523087	222
36	rvrrjc	2233445566	Guntur	guntur	Andhra pradesh	512087	222
37	vnr	1122334455	Hyderbad	miyapur	Telangana	512345	333
38	klh	3445996578	Hyderbad	aziznagar	Telangana	512345	222
39	bvrit	112566725	Hyderbad	nizampet	Telangana	512345	111
40	cbit	1122965785	Hyderbad	gandipet	Telangana	512345	111

DEALER



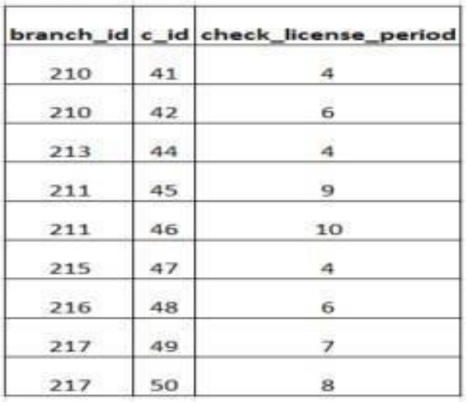
deal_id	deal_name	city	street	state	pincode	d_no	ph_int
51	raju	guntur	Raju Nagar	Andhra Pradesh	612345	555	9988776655
52	raghu	hyderabad	kukatpally	Telengana	678890	666	8765489765
53	kiran	hyderabad	bachupally	Telengana	546789	777	7654564556
54	ganesh	hyderabad	kondapur	Telengana	456789	111	874648545
55	hari	hyderabad	ammerpet	Telengana	534467	222	9988776655
56	kiran	hyderabad	santhanagar	Telengana	512334	333	9988776655
57	kamal	hyderabad	miyapur	Telengana	504406	444	9988776655
58	eswar	guntur	mangalagiri	Andhra Pradesh	563456	888	9988776655

BRANCH



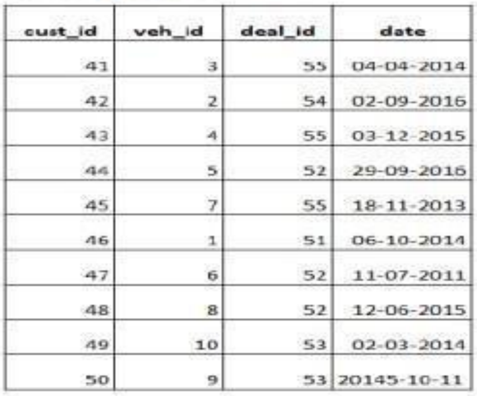
branch_id	b_name	state	city	pincode	street	d_no	phno1	phno2	c_id	v_id	e_id
210	gandipet	Telangana	Hyderbad	512345	jntu	53	9848022338	8802233811	41	1	31
211	madhapur	Telangana	Hyderbad	512345	gachibowli	52	9848022344	8802234433	43	2	32
212	vaddeswaram	Andhra pradesh	Guntur	567432	kondapur	51	8802233811	9848022311	41	1	31
213	jubilee hills	Telangana	Hyderbad	500897	ameerpeta	53	8802234433	9848065722	43	2	32
214	bachupally	Telangana	Hyderbad	512098	erragada	55	9848022311	8065722322	41	1	31
215	kukatpally	Telangana	Hyderbad	523087	pnbs	52	9848065722	8065722311	43	2	32
216	guntur	Andhra pradesh	Guntur	512087	towers	54	8880657222	9848022338	41	1	31
217	miyapur	Telangana	Hyderbad	512345	bhaskar nagar	55	8806572233	9848022338	41	1	31
218	aziznagar	Telangana	Hyderbad	512345	rajunagar	56	8065722322	9848022344	43	2	32
219	nizampet	Telangana	Hyderbad	512345	kbhp	52	8065722311	9848022338	41	1	31

RENEWAL



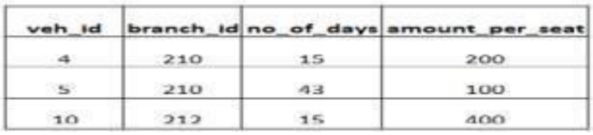
branch_id	c_id	check_license_period
210	41	4
210	42	6
213	44	4
211	45	9
211	46	10
215	47	4
216	48	6
217	49	7
217	50	8

REGISTRATION



cust_id	veh_id	deal_id	date
41	3	55	04-04-2014
42	2	54	02-09-2016
43	4	55	03-12-2015
44	5	52	29-09-2016
45	7	55	18-11-2013
46	1	51	06-10-2014
47	6	52	11-07-2011
48	8	52	12-06-2015
49	10	53	02-03-2014
50	9	53	20145-10-11

CONTRACT_PERMISSION



veh_id	branch_id	no_of_days	amount_per_seat
4	210	15	200
5	210	43	100
10	212	15	400

Questions:

i) Create the database in PostgreSQL and create the necessary tables for the given case study using appropriate keys and relationships between the tables

```
CREATE TABLE CUSTOMER (
   cust_id INT PRIMARY KEY,
   cust_name VARCHAR(50),
   dob DATE,
   city VARCHAR(50),
   street VARCHAR(50),
   state VARCHAR(50),
   pincode INT,
   ph_no VARCHAR(15),
   deal_no VARCHAR(10),
   photo_identity VARCHAR(5),
   v_id INT
);
```

```
INSERT INTO CUSTOMER (cust_id, cust_name, dob, city, street, state, pincode,
ph_no, deal_no, photo_identity, v_id)

VALUES

(41, 'raju', '1996-09-13', 'Guntur', 'Ramgopal', 'Andhra_pradesh', 5000213,
'9123456789', '10', 'y', 3),

(42, 'hari', '2016-06-19', 'Perambur', 'Mylapur', 'Tamil_Nadu', 500211,
'1122334455', '20', 'n', 2),

(43, 'giri', '1995-01-20', 'hyderabad', 'srnagar', 'Telangana', 500079, '8877665544',
'30', 'y', 4),

(44, 'ramu', '1996-07-17', 'vijayawada', 'benz circle', 'Andhra_pradesh', 512345,
'7654564321', '40', 'y', 5),

(45, 'rahul', '1995-12-08', 'guntur', 'rajunagar', 'Andhra_pradesh', 523022,
'9999999998', '50', 'y', 7),

(46, 'gopi', '1979-08-13', 'hyderabad', 'gachibowl', 'Telangana', 567089,
'7787777775', '10', 'n', 1),

(47, 'karthik', '2004-01-15', 'guntur', 'chandram', 'Andhra pradesh', 546789,
'7788776633', '20', 'n', 6),

(48, 'gopal', '2000-12-06', 'Hyderabad', 'ameerpet', 'Telangana', 500023,
'6734556345', '30', 'y', 8),
(49, 'dinesh', '2001-12-10', 'Hyderabad', 'kondapur', 'Telangana', 502033,
'5794537212', '30', 'n', 10),

(50, 'suresh', '2025-03-25', 'vijayawada', 'poranki', NULL, 512022, '7896543233',
'20','y', 9);
```

```
CREATE TABLE VEHICLE (
   veh_id INT PRIMARY KEY,
   veh_type VARCHAR(50),
   veh_name VARCHAR(50),
   veh_number VARCHAR(20)
);


INSERT INTO VEHICLE (veh_id, veh_type, veh_name, veh_number) VALUES

(1, '2_wheeler', 'royal_enfield', 'AP1234'),

(2, '3 wheeler', 'auto', 'AP3421'),

(3, '2_wheeler', 'royal_enfield', 'TS213'),

(4, '4_wheeler', 'fiat', 'AP2346'),

(5, '4_wheeler', 'benz', 'TS1256'),

(6, '3_wheeler', 'auto', 'TN5544'),

(7, '2_wheeler', 'splendor', 'AP3214'),

(8, '2_wheeler', 'bajaj', 'AP7895'),

(9, '2_wheeler', 'royal_enfield', 'AP2134'),

(10, '4_wheeler', 'ambassador', 'TS4567');
```

```
CREATE TABLE EDU_BUS (
   edu_id INT PRIMARY KEY,
   edu_name VARCHAR(50),
   ph_no VARCHAR(15),
   city VARCHAR(50),
   street VARCHAR(50),
   state VARCHAR(50),
   pincode INT,
   deal_no INT
);
```

```
INSERT INTO EDU_BUS (edu_id, edu_name, ph_no, city, street, state, pincode,
deal_no) VALUES

(31, 'dps', '1122334455', 'Hyderbad', 'santhnagar', 'Telangana', 512345, 444),

(32, 'klu', '44556677', 'guntur', 'vaddeswaram', 'Andhra pradesh', 567432, 111),

(33, 'dav', '123456789', 'Hyderbad', 'jubilee hills', 'Telangana', 500897, 333),

(34, 'surya', '4356789321', 'Hyderbad', 'bachupally', 'Telangana', 512098, 111),

(35, 'vit', '7788996578', 'Hyderbad', 'kukatpally', 'Telangana', 523087, 222),

(36, 'rvrrjc', '2233445566', 'Guntur', 'guntur', 'Andhra pradesh', 512087, 222),

(37, 'vnr', '1122334455', 'Hyderbad', 'miyapur', 'Telangana', 512345, 333),

(38, 'klh', '3445996578', 'Hyderbad', 'aziznagar', 'Telangana', 512345, 222),

(39, 'bvrit', '112566725', 'Hyderbad', 'nizampet', 'Telangana', 512345, 111),

(40, 'cbit', '1122965785', 'Hyderbad', 'gandipet', 'Telangana', 512345, 111);
```

```
CREATE TABLE DEALER (
   deal_id INT PRIMARY KEY,
   deal_name VARCHAR(50),
   city VARCHAR(50),
   street VARCHAR(50),
   state VARCHAR(50),
   pincode INT,
   deal_no INT,
   ph_no VARCHAR(15)
);
```

```
INSERT INTO DEALER (deal_id, deal_name, city, street, state, pincode, deal_no,
ph_no) VALUES

(51, 'raju', 'guntur', 'Raju Nagar', 'Andhra Pradesh', 612345, 555, '9988776655'),

(52, 'raghu', 'hyderabad', 'kukatpally', 'Telengana', 678890, 666, '8765489765'),

(53, 'kiran', 'hyderabad', 'bachupally', 'Telengana', 546789, 777, '7654554556'),

(54, 'ganesh', 'hyderabad', 'kondapur', 'Telengana', 456789, 111, '874648545'),

(55, 'hari', 'hyderabad', 'ammerpet', 'Telengana', 534467, 222, '9988776655'),

(56, 'kiran', 'hyderabad', 'santhanagar', 'Telengana', 512334, 333, '9988776655'),

(57, 'kamal', 'hyderabad', 'miyapur', 'Telengana', 504406, 444, '9988776655'),
(58, 'eswar', 'guntur', 'mangalagiri', 'Andhra Pradesh', 553456, 888, '9988776655');
```

```
CREATE TABLE BRANCH (
branch_id INT PRIMARY KEY,
b_name VARCHAR(50),
state VARCHAR(50),
city VARCHAR(50),
pincode INT,
street VARCHAR(50),
d_no INT,
phno1 VARCHAR(15),
phno2 VARCHAR(15),
c_id INT,
v_id INT,
e_id INT
);
```

```
INSERT INTO BRANCH (branch_id, b_name, state, city, pincode, street, d_no,
phno1, phno2, c_id, v_id, e_id) VALUES

(210, 'gandipet', 'Telangana', 'Hyderabad', 512345, 'intu', 53, 9848022338,
8802233811, 41, 1, 31),

(211, 'madhapur', 'Telangana', 'Hyderabad', 512345, 'gachibowli', 52, 9848022348,
8802234343, 43, 2, 32),

(212, 'vaddeswaram', 'Andhra Pradesh', 'Guntur', 567432, 'kondapur', 51,
9880233811, 9848023113, 41, 1, 31),

(213, 'jubilee hills', 'Telangana', 'Hyderabad', 500897, 'ameerpetta', 55,
8802234334, 9846057223, 43, 2, 32),

(214, 'bachupally', 'Telangana', 'Hyderabad', 512098, 'erragada', 53, 9848022311,
8065722322, 41, 1, 31),

(215, 'kukatpally', 'Telangana', 'Hyderabad', 523087, 'prbs', 52, 9846057222,
8065722311, 43, 2, 32),

(216, 'guntur', 'Andhra Pradesh', 'Guntur', 512087, 'towers', 54, 9880557222,
9848022338, 41, 1, 31),

(217, 'miyapur', 'Telangana', 'Hyderabad', 512345, 'bhaskar nagar', 55,
8065722233, 9848022338, 41, 1, 31),

(218, 'aziznagar', 'Telangana', 'Hyderabad', 512345, 'rajunagar', 56, 8065722232,
9848022344, 43, 2, 32),
(219, 'nizampet', 'Telangana', 'Hyderabad', 512345, 'kbhp', 52, 8065722311,
9848022338, 41, 1, 31);
```

```
CREATE TABLE RENEWAL (
   branch_id INT,
   c_id INT,
   check_license_period INT,
   PRIMARY KEY (branch_id, c_id)
);
```

```
INSERT INTO RENEWAL (branch_id, c_id, check_license_period) VALUES

(210, 41, 4),

(210, 42, 6),

(213, 44, 4),

(211, 45, 9),

(211, 46, 10),

(215, 47, 4),

(216, 48, 6),

(217, 49, 7),

(217, 50, 8);
```

```
CREATE TABLE REGISTRATION (
   cust_id INT,
   veh_id INT,
   deal_id INT,
   date DATE,
   PRIMARY KEY (cust_id, veh_id, deal_id)
);
```

```
INSERT INTO REGISTRATION (cust_id, veh_id, deal_id, date) VALUES
(41, 3, 55, '2014-04-04'),
(42, 2, 54, '2016-09-02'),
(43, 4, 55, '2015-12-03'),
(44, 5, 52, '2016-09-29'),
(45, 7, 55, '2013-11-18'),
(46, 1, 51, '2014-10-06'),
(47, 6, 52, '2011-07-11'),
(48, 8, 52, '2015-06-12'),
(49, 10, 53, '2014-03-02'),
(50, 9, 53, '2014-10-11');
```

```
CREATE TABLE CONTRACT_PERMISSION (

   veh_id INT,

   branch_id INT,

   no_of_days INT,

   amount_per_seat DECIMAL(10, 2),

   PRIMARY KEY (veh_id, branch_id)

);


INSERT INTO CONTRACT_PERMISSION (veh_id, branch_id, no_of_days,
amount_per_seat) VALUES
(4, 210, 15, 200.00),
(5, 210, 43, 100.00),
(10, 212, 15, 400.00);
```

ii) Display the vehicles that were registered by the dealer name 'Raghu'

```
SELECT v.*

FROM VEHICLE v

JOIN REGISTRATION r ON v.veh_id = r.veh_id

JOIN DEALER d ON r.deal_id = d.deal_id

WHERE d.deal_name = 'raghu';
```

iii)Display the list of customers who have applied for new license

```
SELECT c.*

FROM CUSTOMER c

LEFT JOIN RENEWAL r ON c.cust_id = r.c_id

WHERE r.c_id IS NULL;
```

iv) Display the vehicles who have been given 30 days of contract permission.

```
SELECT v.*
FROM VEHICLE v
JOIN CONTRACT_PERMISSION cp ON v.veh_id = cp.veh_id
WHERE cp.no_of_days = 30;
```

v) Create a query to display all the records who applied for renewal of license.

```
SELECT *
FROM RENEWAL;
```

vi) Display the count of vehicles of different types.

```
SELECT veh_type, COUNT(*) AS count
FROM VEHICLE
GROUP BY veh_type;
```

vii) Create a query to display customer details who have 2-wheeler vehicle.

```
SELECT c.*
FROM CUSTOMER c
JOIN VEHICLE v ON c.v_id = v.veh_id
WHERE v.veh_type = '2_wheeler';
```

viii) Create a query that displays details of the customer whose license expires in 5 days.

```
SELECT c.*
FROM CUSTOMER c
JOIN RENEWAL r ON c.cust_id = r.c_id
WHERE r.check_license_period = 5;
```

ix) Display the list of educational institutions who applied for permits

```
SELECT *
FROM EDU_BUS;
```

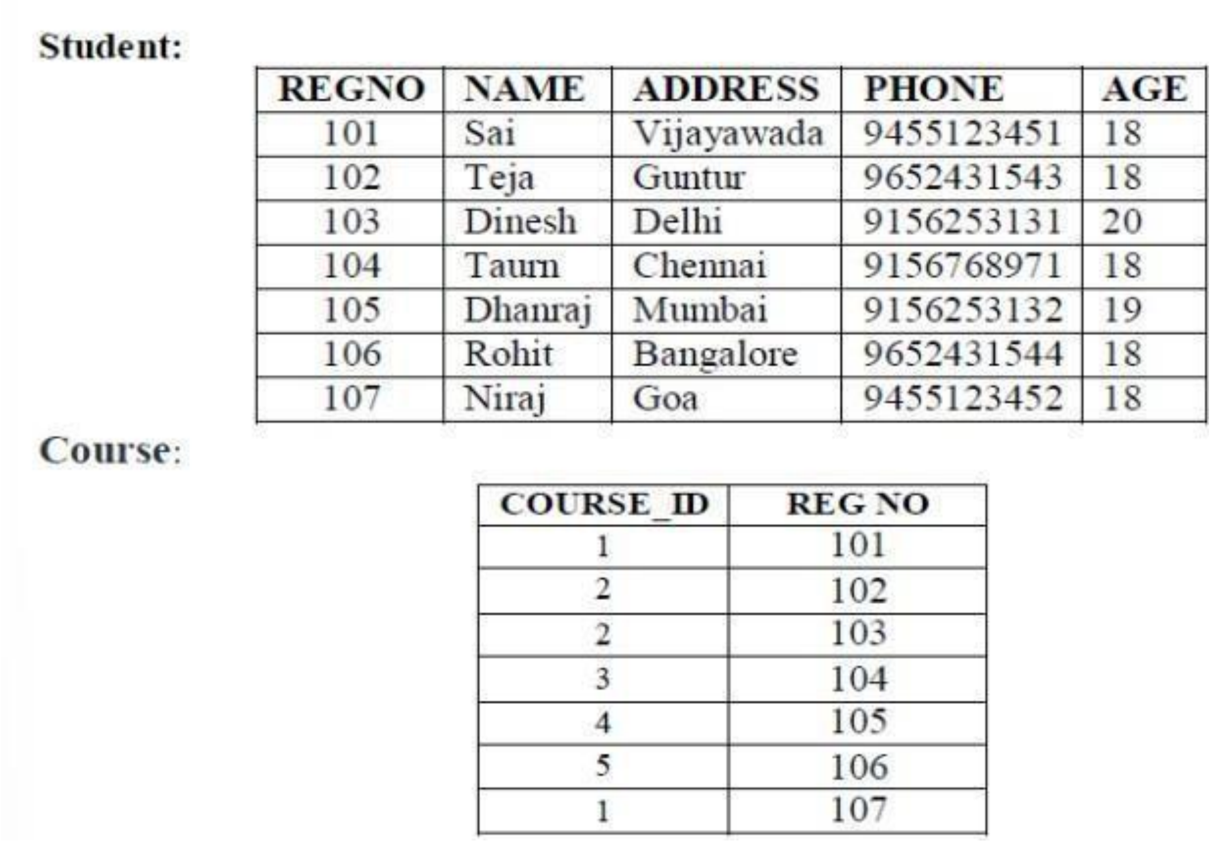
x) Display the total number of vehicles license allotted by each branch

```
SELECT b.b_name, COUNT(*) AS license_count
FROM BRANCH b
JOIN REGISTRATION r ON b.branch_id = b.branch_id
GROUP BY b.b_name;
```

xi) Display the number of customers present under each dealer.

```
SELECT d.deal_name, COUNT(*) AS customer_count
FROM DEALER d
JOIN REGISTRATION r ON d.deal_id = r.deal_id
JOIN CUSTOMER c ON r.cust_id = c.cust_id
GROUP BY d.deal_name;
```

2. Use the tables below and implement the below queries



Student:

REGNO	NAME	ADDRESS	PHONE	AGE
101	Sai	Vijayawada	9455123451	18
102	Teja	Guntur	9652431543	18
103	Dinesh	Delhi	9156253131	20
104	Taurn	Chennai	9156768971	18
105	Dhanraj	Mumbai	9156253132	19
106	Rohit	Bangalore	9652431544	18
107	Niraj	Goa	9455123452	18

Course:

COURSE_ID	REG NO
1	101
2	102
2	103
3	104
4	105
5	106
1	107

```
CREATE TABLE Student (
   REGNO INT PRIMARY KEY,
   NAME VARCHAR(50),
   ADDRESS VARCHAR(100),
   PHONE VARCHAR(15),
   AGE INT
);

INSERT INTO Student (REGNO, NAME, ADDRESS, PHONE, AGE) VALUES
(101, 'Sai', 'Vijayawada', '9455123451', 18),
(102, 'Teja', 'Guntur', '9652431543', 18),
(103, 'Dinesh', 'Delhi', '9156253131', 20),
(104, 'Taun', 'Chennai', '9156768971', 18),
(105, 'Dhanraj', 'Mumbai', '9156253132', 19),
(106, 'Rohit', 'Bangalore', '9652431544', 18),
(107, 'Niraj', 'Goa', '9455123452', 18);
```

```
CREATE TABLE Course (
   COURSE_ID INT,
   REGNO INT,
   FOREIGN KEY (REGNO) REFERENCES Student(REGNO)
);

INSERT INTO Course (COURSE_ID, REGNO) VALUES
(1, 101),
(2, 102),
(2, 103),
(3, 104),
(4, 105),
(5, 106),
(1, 107);
```

i) Write a PostgreSQL query to retrieve the details of students who have enrolled in Course 1

```
SELECT s.*
FROM Student s
JOIN Course c ON s.REGNO = c.REGNO
WHERE c.COURSE_ID = 1;
```

ii) How can you obtain the names of students who have enrolled in Course 2 using an Inner Join between the Student and Course tables?

```
SELECT s.NAME

FROM Student s

INNER JOIN Course c ON s.REGNO = c.REGNO

WHERE c.COURSE_ID = 2;
```

iii) Write a PostgreSQL query to fetch the names and phone numbers of students who have not enrolled in any course.

```
SELECT s.NAME, s.PHONE

FROM Student s

LEFT JOIN Course c ON s.REGNO = c.REGNO

WHERE c.COURSE_ID IS NULL;
```

iv) How can you retrieve the list of courses along with the corresponding student names using an Inner Join between the Student and Course tables?

```
SELECT c.COURSE_ID, s.NAME

FROM Student s

INNER JOIN Course c ON s.REGNO = c.REGNO;
```

v) Write a PostgreSQL query to find the number of students enrolled in each course. Include the course name and the count of students.

```
SELECT c.COURSE_ID, COUNT(c.REGNO) AS student_count

FROM Course c

GROUP BY c.COURSE_ID

ORDER BY student_count DESC;
```

vi) How can you retrieve the names of students along with their corresponding course IDs and course names, even if they have not enrolled in any course? Use an Outer Join for this query.

```
SELECT s.NAME, c.COURSE_ID

FROM Student s

LEFT JOIN Course c ON s.REGNO = c.REGNO;
```

vii) Write a PostgreSQL query to obtain the names of students who are enrolled in Course 1 and Course 2 simultaneously.

```
SELECT s.NAME
FROM Student s
JOIN Course c1 ON s.REGNO = c1.REGNO AND c1.COURSE_ID = 1
JOIN Course c2 ON s.REGNO = c2.REGNO AND c2.COURSE_ID = 2;
```

viii) How can you retrieve the names of students along with their course details, sorted by their age in descending order? Use appropriate join and sorting techniques.

```
SELECT s.NAME, c.COURSE_ID
FROM Student s
LEFT JOIN Course c ON s.REGNO = c.REGNO
ORDER BY s.AGE DESC;
```

ix) Write a PostgreSQL query to find the students who are enrolled in courses but are aged greater than 18.

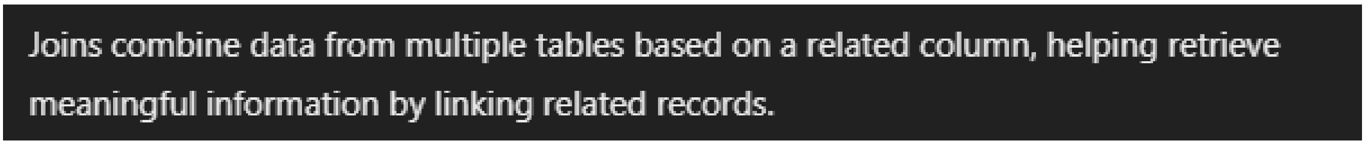
```
SELECT DISTINCT s.NAME

FROM Student s

JOIN Course c ON s.REGNO = c.REGNO

WHERE s.AGE > 18;
```

x) How can you retrieve the details of students who have the same course enrollment as the student with REGNO 101?

```
SELECT DISTINCT s.*
FROM Student s
JOIN Course c1 ON s.REGNO = c1.REGNO
WHERE c1.COURSE_ID IN (SELECT c2.COURSE_ID FROM Course c2 WHERE
c2.REGNO = 101)
AND s.REGNO <> 101;
```

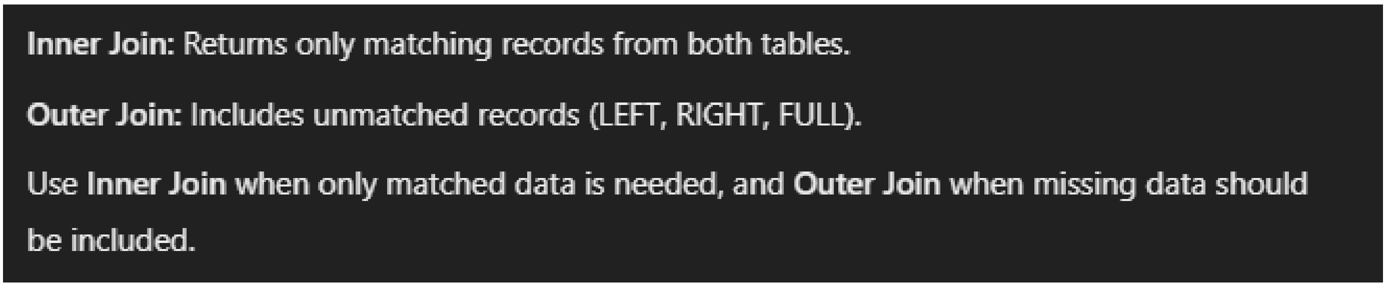
Viva-Voce Questions (In-Lab):

1) What is the purpose of using joins in a database query, and how do they help retrieve meaningful information from multiple tables?



Joins combine data from multiple tables based on a related column, helping retrieve meaningful information by linking related records.

2) When would you choose to use an Inner Join instead of an Outer Join, and vice versa?

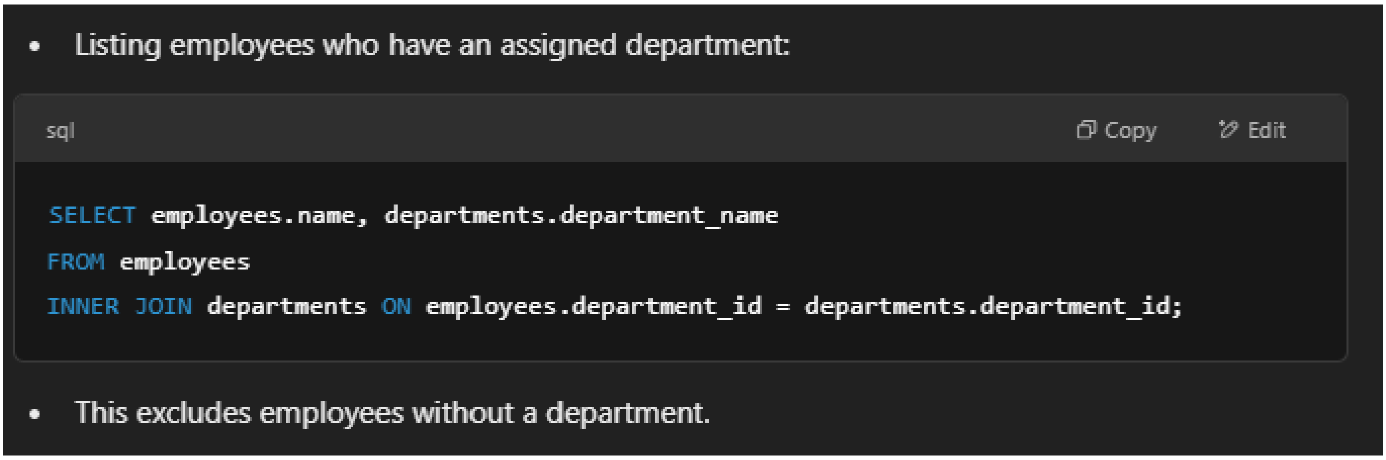


Inner Join: Returns only matching records from both tables.

Outer Join: Includes unmatched records (LEFT, RIGHT, FULL).

Use **Inner Join** when only matched data is needed, and **Outer Join** when missing data should be included.

3) Can you provide an example of a real-world scenario where an Inner Join is more appropriate than an Outer Join?

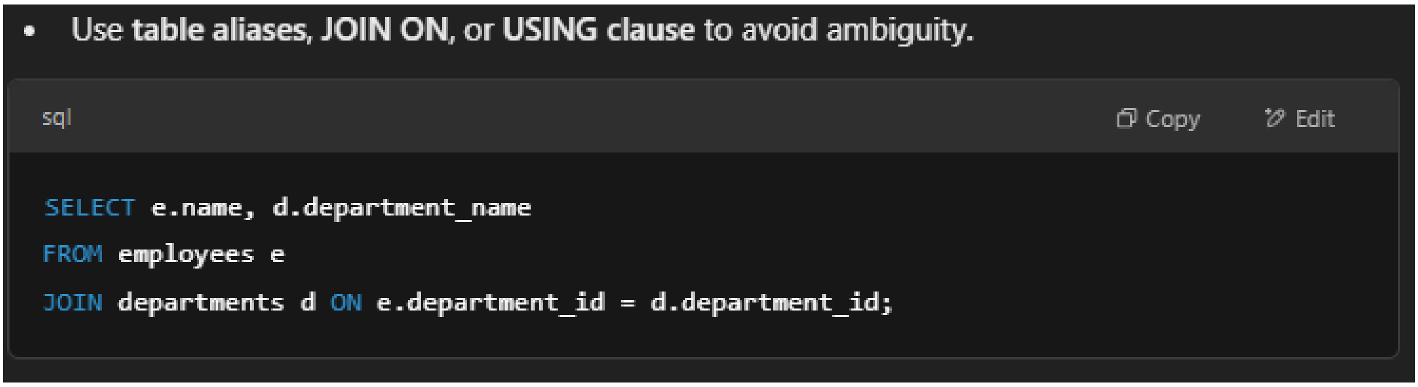


- Listing employees who have an assigned department:

```sql
SELECT employees.name, departments.department_name
FROM employees
INNER JOIN departments ON employees.department_id = departments.department_id;
```

- This excludes employees without a department.

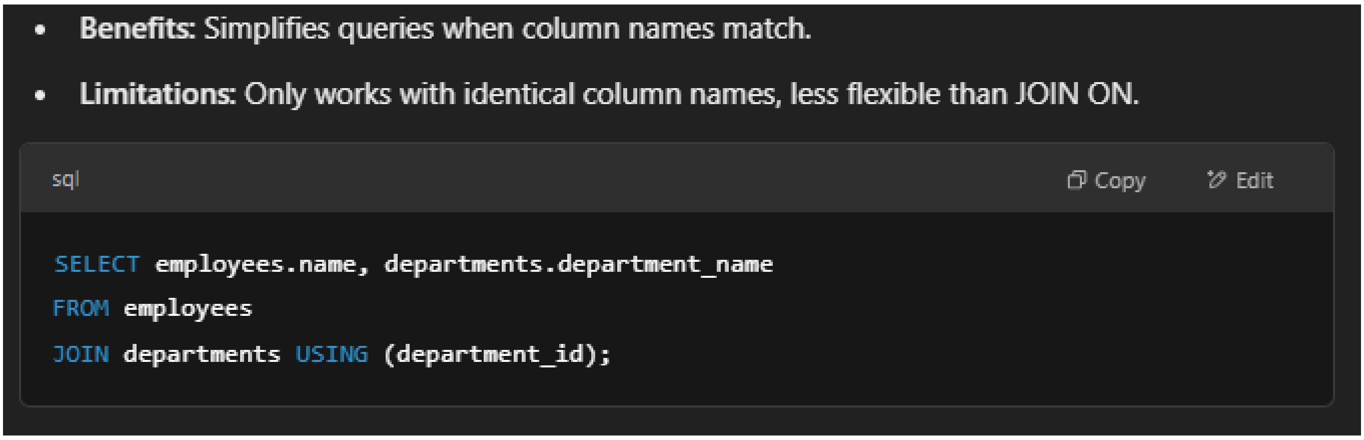
4) How can you handle situations where duplicate column names exist in the joined tables during a Natural Join?



- Use **table aliases, JOIN ON**, or **USING clause** to avoid ambiguity.

```sql
SELECT e.name, d.department_name
FROM employees e
JOIN departments d ON e.department_id = d.department_id;
```

5) What are the benefits and limitations of using the USING clause in a join operation?

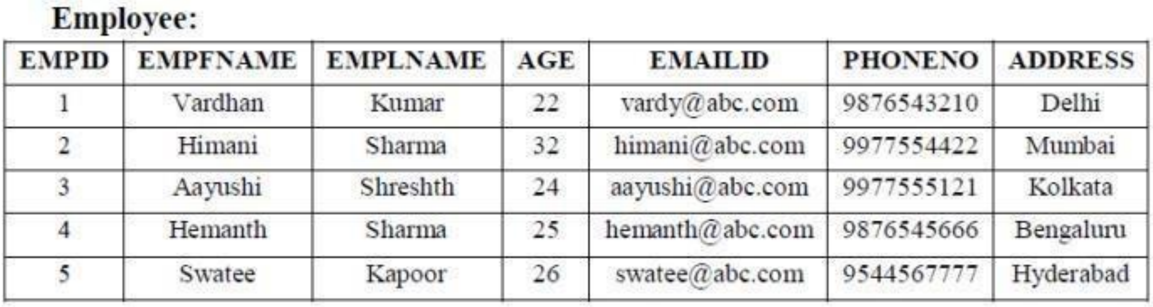


- **Benefits:** Simplifies queries when column names match.
- **Limitations:** Only works with identical column names, less flexible than JOIN ON.

```sql
SELECT employees.name, departments.department_name
FROM employees
JOIN departments USING (department_id);
```

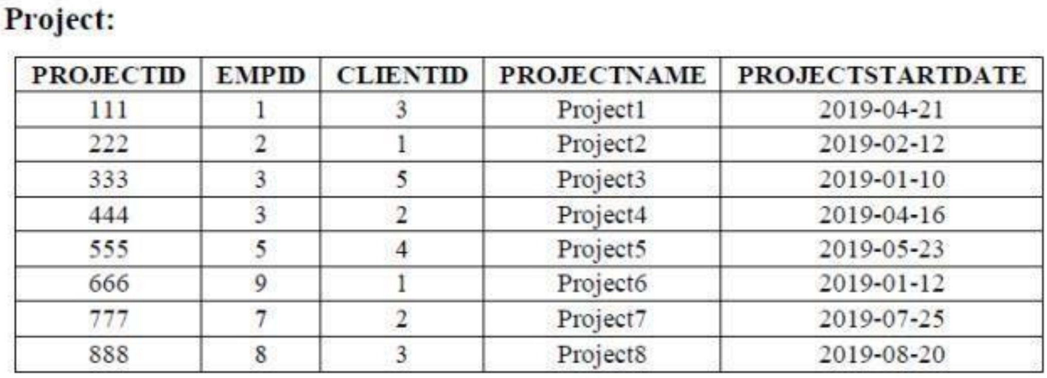
Post Lab Task:

1) Use the tables below and implement the below queries



Employee:

EMPID	EMPFNAME	EMPLNAME	AGE	EMAILID	PHONENO	ADDRESS
1	Vardhan	Kumar	22	vardy@abc.com	9876543210	Delhi
2	Himani	Sharma	32	himani@abc.com	9977554422	Mumbai
3	Aayushi	Shreshth	24	aayushi@abc.com	9977555121	Kolkata
4	Hemanth	Sharma	25	hemanth@abc.com	9876545666	Bengaluru
5	Swatee	Kapoor	26	swatee@abc.com	9544567777	Hyderabad



Project:

PROJECTID	EMPID	CLIENTID	PROJECTNAME	PROJECTSTARTDATE
111	1	3	Project1	2019-04-21
222	2	1	Project2	2019-02-12
333	3	5	Project3	2019-01-10
444	3	2	Project4	2019-04-16
555	5	4	Project5	2019-05-23
666	9	1	Project6	2019-01-12
777	7	2	Project7	2019-07-25
888	8	3	Project8	2019-08-20

```
CREATE TABLE Employee (
   EMPID SERIAL PRIMARY KEY,
   EMPFNAME VARCHAR(50),
   EMPLNAME VARCHAR(50),
   AGE INT,
   EMAILID VARCHAR(100),
   PHONENO VARCHAR(15),
   ADDRESS VARCHAR(100)
);
```

```
INSERT INTO Employee (EMPID, EMPFNAME, EMPLNAME, AGE, EMAILID,
PHONENO, ADDRESS) VALUES
(1, 'Vardhan', 'Kumar', 22, 'vardy@abc.com', '9876543210', 'Delhi'),
(2, 'Himani', 'Sharma', 32, 'himani@abc.com', '9977554422', 'Mumbai'),
(3, 'Aayushi', 'Shreshth', 24, 'aayushi@abc.com', '9977555121', 'Kolkata'),
(4, 'Hemanth', 'Sharma', 25, 'hemanth@abc.com', '9876545666', 'Bengaluru'),
(5, 'Swatee', 'Kapoor', 26, 'swatee@abc.com', '9544567777', 'Hyderabad');
```

```
CREATE TABLE Project (
   PROJECTID SERIAL PRIMARY KEY,
   EMPID INT,
   CLIENTID INT,
   PROJECTNAME VARCHAR(100),
   PROJECTSTARTDATE DATE,
   FOREIGN KEY (EMPID) REFERENCES Employee(EMPID)
);
```

```
INSERT INTO Project (PROJECTID, EMPID, CLIENTID, PROJECTNAME,
PROJECTSTARTDATE) VALUES
(111, 1, 3, 'Project1', '2019-04-21'),
(222, 2, 1, 'Project2', '2019-02-12'),
(333, 3, 5, 'Project3', '2019-01-10'),
(444, 3, 2, 'Project4', '2019-04-16'),
(555, 5, 4, 'Project5', '2019-05-23'),
(666, 9, 1, 'Project6', '2019-01-12'),
(777, 7, 2, 'Project7', '2019-07-25'),
(888, 8, 3, 'Project8', '2019-08-20');
```

Questions:

I) Write a PostgreSQL query to retrieve the details of employees who are assigned to projects

```
SELECT DISTINCT e.*

FROM Employee e

JOIN Project p ON e.EMPID = p.EMPID;
```

II) How can you obtain the names and project names of employees using an Inner Join between the employee and Project tables?

```
SELECT e.EMPFNAME, e.EMPLNAME, p.PROJECTNAME

FROM Employee e

INNER JOIN Project p ON e.EMPID = p.EMPID;
```

III) Write a PostgreSQL query to fetch the details of employees who have not been assigned to any project.

```
SELECT e.*

FROM Employee e

LEFT JOIN Project p ON e.EMPID = p.EMPID

WHERE p.PROJECTID IS NULL;
```

IV) How can you retrieve the names and email IDs of employees along with their corresponding project names, even if they have not been assigned to any project? Use an Outer Join for this query.

```
SELECT e.EMPFNAME, e.EMPLNAME, e.EMAILID, p.PROJECTNAME

FROM Employee e

LEFT JOIN Project p ON e.EMPID = p.EMPID;
```

V) Write a PostgreSQL query to find the number of projects assigned to each employee. Include the employee name and the count of projects.

```
SELECT e.EMPFNAME, e.EMPLNAME, COUNT(p.PROJECTID) AS ProjectCount

FROM Employee e

LEFT JOIN Project p ON e.EMPID = p.EMPID

GROUP BY e.EMPID, e.EMPFNAME, e.EMPLNAME;
```

VI) How can you retrieve the names of employees along with their corresponding project details, sorted by their age in ascending order? Use appropriate join and sorting techniques

```
SELECT e.EMPFNAME, e.EMPLNAME, e.AGE, p.PROJECTNAME,
p.PROJECTSTARTDATE

FROM Employee e

LEFT JOIN Project p ON e.EMPID = p.EMPID

ORDER BY e.AGE ASC;
```

VII) Write a PostgreSQL query to obtain the names of employees who are assigned to both Project 1 and Project 2 simultaneously

```
SELECT e.EMPFNAME, e.EMPLNAME

FROM Employee e

JOIN Project p1 ON e.EMPID = p1.EMPID AND p1.PROJECTNAME =
'Project1'

JOIN Project p2 ON e.EMPID = p2.EMPID AND p2.PROJECTNAME =
'Project2';
```

VIII) How can you retrieve the names of employees along with their project details, sorted by the project start date in descending order? Use appropriate join and sorting techniques.

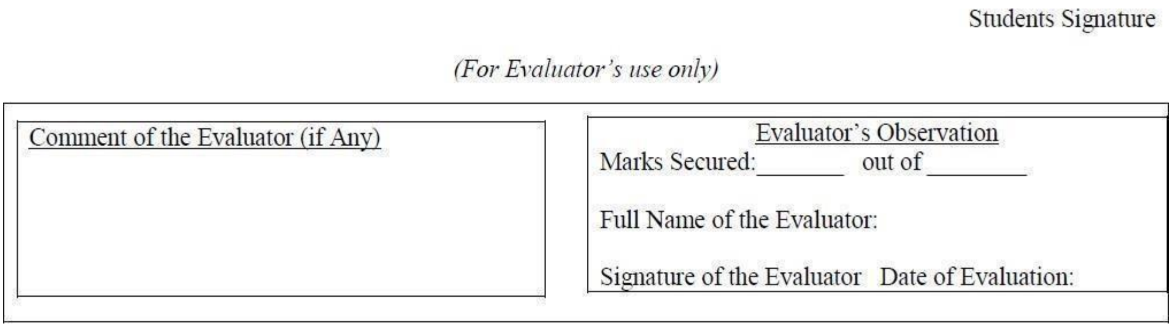
```
SELECT e.EMPFNAME, e.EMPLNAME, p.PROJECTNAME,
p.PROJECTSTARTDATE
FROM Employee e
LEFT JOIN Project p ON e.EMPID = p.EMPID
ORDER BY p.PROJECTSTARTDATE DESC;
```

IX) Write a PostgreSQL query to find the employees who are assigned to projects but have an age greater than 25.

```
SELECT DISTINCT e.*

FROM Employee e

JOIN Project p ON e.EMPID = p.EMPID

WHERE e.AGE > 25;
```

X) How can you retrieve the details of employees who have the same project assignment as the employee with EMPID 3?

```
SELECT DISTINCT e.*
FROM Employee e
JOIN Project p1 ON e.EMPID = p1.EMPID
WHERE p1.PROJECTID IN (
   SELECT PROJECTID FROM Project WHERE EMPID = 3
) AND e.EMPID <> 3;
```



Students Signature

(For Evaluator's use only)

Comment of the Evaluator (if Any)

Evaluator's Observation

Marks Secured:_______ out of ________

Full Name of the Evaluator:

Signature of the Evaluator Date of Evaluation: