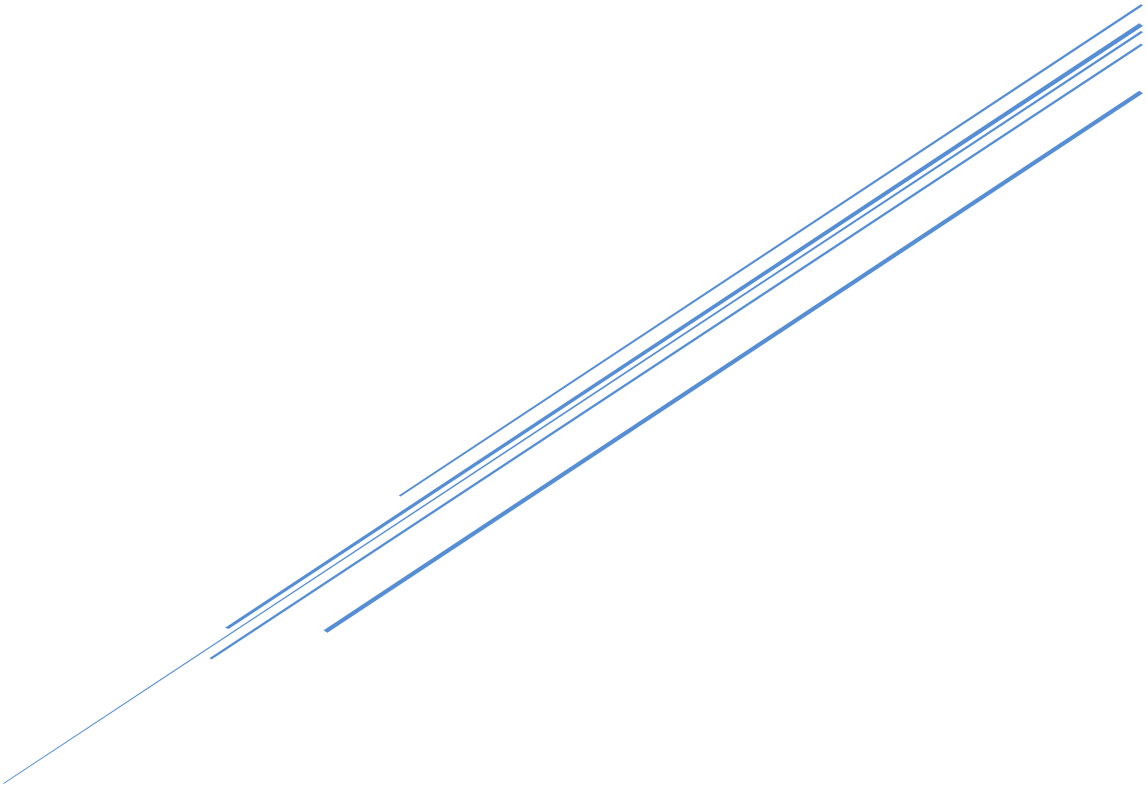
Database Management Systems
24AD2103

Lab Workbook



STUDENT ID: **ACADEMIC YEAR: 2025-26**

STUDENT NAME:

Table of Contents

1. ER Modelling & Mapping-1 1

2. ER Modelling & Mapping-2 15

3. Database Creation & Management 22

4. Basic SQL Queries 32

5.Data Manipulation and Views 46

6.Introduction to PL/SQL 56

7. Introduction to PL/PGSQL in a Bank Management System 70

8.Implementation of Transaction in SQL Server 78

9.Indexing Experimentation 84

10.Query Optimization 92

11.Query Optimization in a Library Management System 100

12.NoSQL Indexing 106

13.NO SQL Aggregation 115

Our Vision:

- To be a department of international repute through continuous research, innovation and industry led curriculum**.**

Our Mission:

- To Impart Quality Education with social consciousness and make them Globally Competent.

Organization of the STUDENT LAB WORKBOOK

The laboratory framework includes a creative element but shifts the time-intensive aspects outside of the Two-Hour closed laboratory period. Within this structure, each laboratory includes three parts: Prelab, In-lab and post-lab.

Pre-Lab

The Prelab exercise is a homework assignment that links the lecture with the laboratory period - typically takes 2 hours to complete. The goal is to synthesize the information they learn in lecture with material from their textbook to produce a working piece of software. Prelab Students attending a two-hour closed laboratory are expected to make a good-faith effort to complete the Prelab exercise before coming to the lab. Their work need not be perfect, but their effort must be real(roughly80percentcorrect).

In-Lab

The In-lab section takes place during the actual laboratory period. The First hour of the laboratory period can be used to resolve any problems the students might have experienced in completing the Prelab exercises. The intent is to give constructive feedback so that students leave the lab with working Prelab software - a significant accomplishment on their part. During the second hour, students complete the In-lab exercise to reinforce the concepts learned in the Prelab. Students leave the lab having received feedback on their Prelab and In-lab work.

Post-Lab

The last phase of each laboratory is a homework assignment that is done following the laboratory period. In the Post-lab, students analyse the efficiency or utility of a given system call. Each Post-lab exercise should take roughly 120minutes to complete.

LAB CONTINIOUS EVALUATION SHEET											
S. No	**Date**	**Experiment Name**	**Pre-Lab (5M)**	**In-Lab**				**Post Lab (5M)**	**Viva Voce (5M)**	**Total (50M)**	**Faculty Signature**
				LOGIC (10M)	**EXECUTION (10M)**	**RESULT (10M)**	**ANALYSIS (5M)**				
1		ER Modelling & Mapping-I									
2		ER Modelling & Mapping-II									
3		Database Creation & Management									
4		Basic SQL Queries									
5		Data Manipulation and Views									
6		Session 06: Introduction to PL/SQL									
7		PL/PGSQL in a Bank Management System									
8		Implementation of Transaction in SQL Server									
9		Indexing Experimentation									
10		Query Optimization									
11		Query Optimization in a Library Management System									
12		NoSQL Indexing									
13		NoSQL Aggregation									

Lab Session 01

1. ER Modelling & Mapping-1

Aim:

The aim of this experiment is to understand and apply the concepts of Entity Relationship (ER) Modelling to design a database schema that accurately represents the relationships between entities in a given domain.

Description:

The lab experiment on Entity Relationship (ER) Modelling involves studying and applying the fundamental principles of ER modelling to design a database schema. Students will learn how to identify and define entities, relationships, and attributes within a given domain. They will practice creating an ER diagram that accurately represents the relationships between entities and their attributes. The experiment will also cover the use of cardinality and participation constraints to establish the nature and degree of relationships. By the end of the lab, students will gain hands-on experience in translating realworld scenarios into an ER model, providing a solid foundation for database design and development. Pre-Requisites: PostgreSQL, TerraER Tool, Windows/ Ubuntu/CentOS/Debian, DBMS Concepts.

Pre-Requisites:

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

Pre Lab-Task:

1.What is an associative entity in ER modelling, and when is it used?

2.Discuss the concept of specialization and generalization in ER modelling.

3.What is the difference between a weak entity and a strong entity in ER modelling? Provide examples of each.

4.What is the purpose of a composite attribute, and how is it represented in an ER diagram?

5.How do you handle complex relationships, such as ternary relationships, in an ER diagram?

6.Explain the concept of recursive relationships in ER modelling and provide an example.

In Lab Task:

Airline Reservation System

Problem Description: There are 6 different airlines in 6 different countries: Canada – Air Can, USA - USAir, UK – Brit Air, France – Air France, Germany – Luft Air, Italy – Ital Air. Their flights involve the following 12 cities: Toronto and Montreal in Canada, New York and Chicago in US, London and Edinburgh in UK, Paris and Nice in France, Bonn and Berlin in Germany, Rome and Naples in Italy. In each of the 12 cities, there is a (single) booking office. You are going to design a central air- reservation database to be used by all booking offices.

The flight has a unique flight number, airline code, business class indicator, smoking allowed indicator. Flight availability has flight number, date + time of departure, number of total seats available in business class, number of booked seats in business class, number of total seats available in economy class, and number of booked seats in economy class.

The customers may come from any country, not just the 6 above, and from any province/state, and from any city. The customer has first & last name, mailing address, zero or more phone numbers, zero or more fax numbers, and zero or more email addresses. Mailing address has street, city, province or state, postal code and country. Phone/fax number has country code, area code and local number. Email address has only one string, and no structure is assumed. A customer can book one or more flights. Two or more customers may have same mailing address and/or same phone number(s) and/or same fax number(s). But the email address is unique for each customer. First and last names do not have to be unique.

Booking has an unique booking number, booking city, booking date, flight number, date + time of departure (in local time, and time is always in hours and minutes), date

+ time of arrival (in local time), class indicator, total price (airport tax in origin + airport tax in destination + flight price – in local currency. The flight price for business class is

1.5 times of the listed flight price), status indicator (three types: booked. Canceled – the customer canceled the booking, scratched – the customer had not paid in full 30 days prior to the departure), customer who is responsible for payment, amount-paid- so far (in local currency), outstanding balance (in local currency), the first & last names to be printed on the ticket. The airport taxes must be stored in local currencies (i.e. Canadian 19 dollars, US dollars, British Pounds, French francs, German marks, and Italian Liras). Since the exchange rates change daily, they also must be stored for calculations of all prices involved. Though France, Germany, and Italy have had a common currency for a while, we used the names of their original currencies to involve in this exercise currency exchange rates and their changes.

1.Design an ER diagram that represents the entities, attributes, relationships, and cardinalities for the Airline Reservation System. Consider any necessary assumptions and justify them if required

2.Write the Relational schema that represents the entities, attributes, relationships, and cardinalities for the Airline Reservation System. Consider any necessary assumptions and justify them if required.

Viva-Voce Questions (In-Lab):

1.Can you explain the limitations or drawbacks of using an ER diagram for complex database systems?

2.How would you handle a scenario where an entity has multiple relationships with the same entity type in an ER diagram?

3.What are some common challenges or pitfalls that can arise when identifying entities and relationships in a real-world domain for ER modelling?

4.How do you handle evolving or changing requirements in ER modelling, especially when the relationships between entities need to be modified or updated?

5.Can you discuss the trade-offs between using a single ER diagram versus multiple interconnected ER diagrams for a complex database system?

Post Lab:

Fashion Hub Retail Company Database:

Problem Description: A retail company named "Fashion Hub" wants to design a database to manage their product inventory and sales. The company sells various fashion items such as clothing, accessories, and footwear. They have multiple physical stores located in different cities. The following information is provided to design the database:

Products: Fashion Hub sells a wide range of products, including clothing, accessories, and footwear. Each product has a unique product code, name, brand, price, and quantity in stock. Additionally, each product belongs to a specific category (e.g., shirts, dresses, bags, shoes).

Stores: The company operates multiple physical stores in different cities. Each store has a unique store ID, name, address, and contact information.

Customers: Customers can create accounts with Fashion Hub to make purchases. Each customer has a unique customer ID, name, address, phone number, and email. The company also stores information about customer preferences and purchase history.

Sales: Fashion Hub keeps track of sales transactions made by customers. Each sale has a unique sale ID, date and time, customer ID, and store ID. It also includes the total amount paid, any discounts applied, and the payment method used (e.g., cash, credit card).

Employees: The company employs staff members at each store. Each employee has a unique employee ID, name, address, phone number, and position (e.g., store manager, sales associate).

Design an ER diagram that represents the entities, attributes, relationships, and cardinalities for the Fashion Hub database. Consider any necessary assumptions and justify them if required.

Viva-Voce Questions (In-Lab):

1.Can you explain the limitations or drawbacks of using an ER diagram for complex database systems?

2.How would you handle a scenario where an entity has multiple relationships with the same entity type in an ER diagram?

3.What are some common challenges or pitfalls that can arise when identifying entities and relationships in a real-world domain for ER modelling?

4.How do you handle evolving or changing requirements in ER modelling, especially when the relationships between entities need to be modified or updated?

5.Can you discuss the trade-offs between using a single ER diagram versus multiple interconnected ER diagrams for a complex database system?

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:

Lab Session 02

2. ER Modelling & Mapping-2

Aim:

The aim of this experiment is to understand and apply the conceptsof Entity Relationship (ER) Modelling to design a database schema that accurately represents the relationships between entities in each domain.

Description:

The lab experiment on Entity Relationship (ER) Modelling involves studying and applying the fundamental principles of ER modelling to design a database schema. Students will learn how to identify and define entities, relationships, and attributes within a given domain. They will practice creating an ER diagram that accurately represents the relationships between entities and their attributes. The experiment will also cover the use of cardinality and participation constraints to establish the nature and degree of relationships. By the end of the lab, students will gain hands-on experience in translating realworld scenarios into an ER model, providing a solid foundation for database design and development. Pre-Requisites: PostgreSQL, TerraER Tool, Windows/ Ubuntu/CentOS/Debian, DBMS Concepts.

Pre-Requisites:

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

Pre-Lab Task:

1.Explain about derived attributes

2.Explain about simple attribute

3.Explain about composite attribute

4.Explain about key attribute

5.Explain about Multi-valued attribute

In Lab Task:

Draw an ER Diagram for the case study 3, Railway reservation system. The railway reservation system facilitates the passengers to enquire about the trains available based on source and destination, booking and cancellation of tickets, enquire about the status of the booked ticket. Our aim here is to design and develop a database maintaining records of different trains, train status, and passengers. the record of train includes its number, name, source, destination, and days on when its available. Passengers can book their tickets for the train in which seats are available. Before booking a ticket, the validity of train number and booking date is checked. Once they are validated, seat availability is checked, and ticket is booked. Ticket once booked can be cancelled at any time. For this we need to provide ticket id. Also consider that initially passenger needs authentication as well. Design a database for the above scenario. Include the entities whatever are required according to your requirements.

Draw an ER Diagram for the following scenario:

In a university, there are several departments. Each department has a Head of Department (HoD), who is a member of the faculty. A department has a name, phone extension, specific mailing address, and students associated with it. A student belongs to only one department at a time, while a department can have many or no students. Students and faculty members have names and unique identification numbers, along with other details such as address, age, gender, etc.

Students enroll in various courses offered by the university. These courses are taught by faculty members. In each semester, a student can enroll in multiple courses, and a faculty member can teach multiple courses. Faculty members may teach courses across different departments. Each course can be taught by multiple faculty members or none. Faculty members also work on multiple research projects. These projects are funded by the government or the university. A single project can involve multiple faculty members, and a faculty member can work on multiple projects.

Viva-Voce Questions (In-Lab):

1.What is an ER Diagram?

2.Why are composite keys important when modelling databases with entity-relationship diagrams?

3.What does it mean to generalize/specialize an object in an ER diagram?

4.What is an identifying relationship?

5.What is the primary key? How is it represented in an ER diagram?

Post Lab Task:

Case Study: University Online Exam System – ER Diagram Design

We are all aware of the current pandemic situation. To ensure timely completion of the semester, our university is conducting online exams with integrated invigilation and proctoring features. This system is highly effective in maintaining social distancing, which is crucial in the present scenario.

Students benefit by continuing their academic progress without losing a year, and the exams are conducted fairly and efficiently through this system.

To design this system, the database should contain information about:

Courses: This includes details of the courses for which exams are conducted, along with the number of students enrolled for each course.

Sections: Each course may have multiple sections.

Exams: The Online Exam database should store exam-specific information such as: Exam ID, Exam date, Associated squad (monitoring staff)

Faculty: A faculty database should store details of staff who are assigned exam duties, such as: Proctors, Squad members, Rooms: Based on the number of students enrolled for a course exam, rooms are allocated. The room database should: Limit each room to a maximum of 50 students, include details of the proctors assigned to each room. This system ensures proper planning and execution of online exams by managing the allocation of students, rooms, courses, faculty, and exam logistics.

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:

Lab Session 03

3. Database Creation & Management

Aim

The aim of this experiment is to familiarize students with the To download, install the PostgreSQL and practical implementation of creating database objects, defining their structure, and apply constraints using DDL operations.

Description:

PostgreSQL is a popular relational database management system (RDBMS). PostgreSQL is not controlled by any corporation or other private entity and the source code is available free of charge. PostgreSQL runs on all major operating systems, including Linux, UNIX and Windows. It supports text, images, sounds, and video, and includes programming interfaces for C / C++, Java, Perl, Python, Ruby, Tcl and Open Database Connectivity (ODBC). The "Implementation of DDL Commands" lab experiment focuses on providing students with practical exposure to executing Data Definition Language (DDL). The experiment involves creating database objects, such as tables, views, and indexes, using DDL commands. Through this hands-on experience, students will gain proficiency in effectively utilizing DDL. The lab aims to enhance students' understanding of database management and reinforce their skills in executing DDL.

Pre-Requisites:

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

In Lab Task:

Implementation of basic PostgreSQL Queries with DDL commands.

Schema:

SAILORS (SID: INTEGER, SNAME: STRING, RATING: INTEGER, AGE: REAL);
BOATS (BID: INTEGER, BNAME: STRING, COLOR: STRING);

RESERVES (SID: INTEGER, BID: INTEGER, DAY: DATE).

SAILORS			
SID	SNAME	RATING	AGE
22	Dustin	7	45
29	Brutus	1	33
31	Lubber	8	56
32	Andy	8	26
58	Rusty	10	35
64	Horatio	7	35
71	Zor	10	16
74	Horatio	9	40
85	Art	3	26
95	Bob	3	64

BOATS		
ID	BNAME	COLOR
101	Interlake	Blue
102	Interlake	Red
103	Clipper	Green
104	Marine	Red

RESERVES		
SID	BID	DAY
22	101	1998-10-10
22	102	1998-10-10
22	103	1998-10-8
22	104	1998-10-7
31	102	1998-11-10
31	103	1998-11-6
31	104	1998-11-12
64	101	1998-9-5
64	102	1998-9-8
74	103	1998-9-8

In the reserves Questions:

i) Create tables for Sailors, Reserves, and Boats with appropriate key constraints.

ii) Write PostgreSQL queries to insert values into the above database.

iii) Write a query to retrieve distinct sailor names and sailor ages from the Sailors table.

iv) Write a query to retrieve the boat name and boat ID for boats that are either red or green in color.

v) Find the names of sailors who have reserved a red boat, and list them in the order of their age.

vi) Find the IDs of sailors who have reserved either a red boat or a green boat.

vii) Find the names of sailors who have reserved at least one boat.

viii) Write a query to insert a value in the age column.

ix) Write a query to insert the following values into the Reserves table:

sid: 100, bid: 205, day: '12-09-17'.

x) Write a query to insert the following values into the Sailors table:

sid: 95, sname: 'Anil', rating: 3, age: 64.

xi) Write a query to update the rating to 10.5 where sid = 95.

xii) Format the date in the format dd/mm/yyyy.

Viva-Voce Questions (In-Lab):

1.What is PostgreSQL?

2.What are the advantages of PostgreSQL?

3.Which data types are used in PostgreSQL?

4.What are some common challenges or considerations when designing and implementing complex database schemas using DDL commands?

5.Explain the concept of data integrity constraints in DDL commands and provide examples of commonly used constraints.

Post Lab Task:

Use the tables below and implement the below queries

Customer			
CID	PID	Renatl_Date	Rental_ Period
104	300	10-06-2020	6
105	302	05-07-2020	10
108	304	15-07-2020	3
109	305	25-06-2020	6

Artist			
Aid	Name	Address	Phone
200	John	Delhi	7786549803
201	Samuel	Mumbai	7123458790
202	Samson	Lucknow	9460367777
203	David	Hyderabad	9797276764
204	Raghu	Hyderabad	8134185751
205	Ravi	Mumbai	7471094738
206	Kiran	Delhi	9808003725

Painting				
Cid	Cname	Address	Phone	Category
100	Raju	Hyderabad	9876045789	Bronze
101	Hari	Vijayawada	8877678956	Gold
102	Devi	Guntur	7879312123	Silver
103	Rani	Delhi	8780945290	Platinum
104	Jaya	Mumbai	9612578457	Gold
105	Haritha	Kolkata	9611665513	Silver
106	Kalyan	Vijayawada	9610752569	Bronze
107	Roja	Hyderabad	9609839625	Platinum
108	Amar	Vijayawada	9608926681	Gold
109	Padma	Vijayawada	9608013737	Bronze

Owner				
Pid	Oid	Name	Address	Phone
300	500	Raju	Hyderabad	9460367777
301	500	Hari	Vijayawada	8134185751
302	501	Giri	Hyderabad	7808003725
303	501	Gopi	Delhi	9481821699
304	503	Krishna	Mumbai	7155639673
305	502	Verma	Delhi	8829457647
306	502	Guna	Delhi	7503275621

Rent			
Pid	Aid	Rental_Cost	Type
300	201	4500	Hired
301	202	3500	Not Hired
302	203	7500	Hired
303	205	2500	Not Hired
304	202	10000	Not Hired
305	201	8000	Not Hired
306	203	6500	Not Hired

1.Create the tables identifying the constraints and relationships.

2.Insert at least 10 records into these tables

3.Display the customer details who got the category as

4.Display the list of artists who belong to

5.Create a PostgreSQL query display the painting details which are not hired

6.Create a PostgreSQL query to update the rental cost of the painting with 1000 for the paintings which are not hired.

7.Display the details of the artist whose paintings are hired

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:

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?

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

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

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

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

In Lab Task:

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

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

Questions:

1. Create the database in PostgreSQL and create the necessary tables for the given case study using appropriate keys and relationships between the tables
2. Display the vehicles that were registered by the dealer name 'Raghu'
3. Display the list of customers who have applied for new license
4. Display the vehicles who have been given 30 days of contract permission.
5. Create a query to display all the records who applied for renewal of license.
6. Display the count of vehicles of different types.
7. Create a query to display customer details who have 2-wheeler vehicle.
8. Create a query that displays details of the customer whose license expires in 5 days.
9. Dislay the list of educational institutions who applied for permits
10. Display the total number of vehicles license allotted by each branch
11. Display the number of customers present under each dealer.

POST-LAB

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

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

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

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

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

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

6.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.

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

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?

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

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

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

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

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:

Lab Session 05

5.Data Manipulation and Views

Aim:

The aim of this experiment is to familiarize students with the concept of DML commands, subqueries, their syntax, and usage for retrieving specific data subsets within queries. Additionally, the lab focuses on DML commands and creating views, which are virtual tables derived from queries, to simplify complex queries and enhance data accessibility.

Description:

The lab experiment involves a procedural description where students gain hands-on experience in implementing DML commands, subqueries and creating views in a database system. The lab begins with an introduction to the DML commands like INSERT, UPDATE, DELETE, subqueries, explaining their purpose and syntax.

Students then proceed to practice writing queries with subqueries, embedding them within SELECT, FROM, and WHERE clauses to retrieve specific subsets of data based on conditions. They learn how to use subqueries to perform calculations, filtering, and data manipulation. The lab also covers the creation of views, allowing students to create virtual tables derived from queries to simplify complex queries and improve data accessibility. Students learn to create, modify, and utilize views in their queries.

Pre Lab-Task:

1.How can you delete rows from a table.

2.How to Update Rows in a Predefined Table

3.What is a View in PostgreSQL, and What Purpose Does it Serve?

4.How to Create a View in PostgreSQL using the CREATE VIEW statement?

5.Can You Modify Data Through a View in PostgreSQL? Why or Why Not?

6.How to Update the Definition of a View in PostgreSQL Using the ALTER VIEW Statement

In Lab Task:

Write PostgreSQL queries using DML commands and nested subqueries to analyze the following data.

Patient:

Pid	Fname	Lname	Email	Phno	Address	Date_Admn
200	David	Samson	aaa@gmail.com	7654328976	Mumbai	13-10-2019
201	Bharath	Kumar	bbb@gmail.com	8765438907	Hyderabad	22-01-2020
202	Eswar	Prasad	ccc@gmail.com	9876548838	Vijayawada	15-03-2020
203	Jaya	Laxmi	ddd@gmail.com	8765489765	Delhi	06-07-2020
204	Laxmi	Devi	eee@gmail.com	7654430692	Mumbai	12-03-2020
205	Pramod	Reddy	fff@gmail.com	6543371619	Hyderabad	05-02-2020
206	Charan	Kumar	ggg@gmail.com	5432312546	Vijayawada	22-04-2020
207	Kalyan	Reddy	hhh@gmail.com	4321253473	Delhi	10-07-2020
208	Rajesh	Yadav	iklmn@gmail.com	3210194400	Chennai	03-03-2020
209	Naveen	Kumar	jghfr@gmail.com	2099135327	Hyderabad	22-04-2020

Pcase	
Bednum	Pid
20	200
21	201
22	202
23	203
24	204
25	205
26	206
27	207
28	208
29	209

Shift			
Doctor Id	Day	Starttime	Endtime
100	12-04-2020	9	5
101	20-06-2020	4	8
102	01-08-2020	6	12
105	05-08-2020	13	18
106	13-10-2019	4	8
107	22-01-2020	6	12
108	15-03-2020	13	18
109	06-07-2020	9	5
110	12-03-2020	4	8
111	05-02-2020	6	12
112	22-04-2020	13	18
113	10-07-2020	4	8
114	03-03-2020	6	12
115	22-04-2020	13	18

Appointment	
Pid	Amt
200	200
201	200
202	300
203	600
204	200
205	300
206	200
207	200
208	200
209	600

Questions:

i) Write a query to insert multiple rows into the Appointment table as shown above.

ii) Write a query to update all rows of personal email addresses to academic email addresses and display the Patient table.

iii) Write a query to create a view from the Patient table.

iv) Write a PostgreSQL query to update the Patient view where PatientID > 3, using a REPLACE VIEW statement.

v) Display the patient's date of joining in the format: Month (e.g., January, February).

vi) Write a PostgreSQL query to display the count of patients who come from the same city.

vii) Write a PostgreSQL query to retrieve the first name of patients who are admitted and allotted a bed number.

viii) Write a query to retrieve the first name and last name of patients who have an appointment, and whose appointment amount is greater than the average appointment amount.

ix) Create a view that shows the number of appointments made by each doctor, along with their total earnings from those appointments.

x) Write a query to find the total number of patients admitted on each date.

Viva-Voce Questions (In-Lab):

1.List various DML commands.

2.Differentiate the commands ALTER and UPDATE.

3.How can you optimize performance when working with subqueries in PostgreSQL?

4.What are the benefits of using views in a database system?

5.Can you provide an example of a situation where creating a view would be beneficial?

6.Discuss the potential limitations or considerations when using subqueries or views in complex queries.

Post Lab Task:

Write PostgreSQL queries using nested subqueries to analyse the following data.

Questions:

1.Write a query to find the top 3 doctors who have earned the highest total amount from appointments.

2.Create a function that takes a Doctor ID as input and returns the total number of appointments made by that doctor.

3.Write a query to find the patients who have spent the most on appointments, including their total amount spent and the doctor they visited the most.

4.Create a view that displays the average appointment amount for each doctor, categorized by the year of the appointment.

5.Write a query to find the doctor who has treated the highest number of unique patients.

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:

Lab Session 06

6.Introduction to PL/SQL

Aim:

The aim of this experiment is to familiarize students with the PL/SQL, programs, creating reusable procedures & functions and its capabilities for developing database-centric applications.

Description:

The lab experiment involves a procedural description where students gain hands-on experience in writing and executing PL/SQL programs, procedures, and functions. The lab begins with a brief introduction to the PL/SQL language, highlighting its role in developing database-centric applications. Students then proceed to practice implementing PL/SQL programs by writing code blocks that include variable declarations, control structures, and SQL statements. They learn how to create reusable procedures and functions to encapsulate specific logic or calculations. The lab provides students with sample scenarios and real-world problems, enabling them to apply PL/SQL programming concepts to solve database-related tasks. Through this procedural description, students enhance their understanding of PL/SQL programming, develop skills in writing efficient and maintainable code, and gain insights into the benefits of using PL/SQL for database development.

Pre Lab-Task:

1.What Compare SQL & PL/SQL

2.How do you declare and initialize variables in a PL/SQL program or procedure in PostgreSQL?

3.Can a PL/SQL function return multiple values in PostgreSQL? If so, how can it be achieved?

4.What are the benefits of using packages in PL/SQL programming in PostgreSQL? Provide an example of a situation where packages can be useful.

5.How can you handle transactions within PL/SQL programs or procedures in PostgreSQL? Explain the use of the COMMIT and ROLLBACK statements.

6.What is the difference between an anonymous PL/SQL block and a stored procedure in PostgreSQL? When would you choose to use one over the other?

7.How can you debug PL/SQL code in PostgreSQL? Describe some debugging techniques or tools available.

8.Can you call a PL/SQL function from a SQL statement in PostgreSQL? If so, provide an example and explain the process

In Lab Task:

Write 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

Execute the following queries:

1.Write a PL/SQL program that retrieves all the records from the "Student" table and displays the student details.

2.Create a PL/SQL procedure that takes a student's registration number as input and updates their age to 20. Test the procedure by updating the age of a specific student.

3.Develop a PL/SQL function that calculates the average age of all students in the "Student" table and returns the result.

4.Create a PL/SQL procedure that inserts a new row into the "Course" table, enrolling a student with a given registration number into a specified course. Test the procedure by enrolling a student in a course.

5.Write a PL/SQL function that retrieves the count of students in the "Student" table and returns the result.

6.Develop a PL/SQL procedure that deletes a student's record from the "Student" table based on their registration number. Test the procedure by deleting a specific student's record.

7.Write a PL/SQL program that retrieves all the records from the "Course" table and displays the course details.

8.Create a PL/SQL function that counts the number of students enrolled in each course. and returns the result as a cursor.

9.Develop a PL/SQL procedure that updates the address of a student based on their registration number. Test the procedure by updating a specific student's address.

10.Write a PL/SQL program that accepts an age as input and retrieves all the student details from the "Student" table with that age

Viva-Voce Questions (In-Lab):

1.How can you pass parameters to PL/SQL procedures and functions, and what are the benefits of doing so?

2.What are the benefits of encapsulating logic within procedures and functions in PL/SQL?

3.Discuss the usage of cursors in PL/SQL and their significance in handling result sets.

4.Explain the concept of packages in PL/SQL and how they aid in organizing and modularizing code.

5.How can you use PL/SQL programs, procedures, and functions to enhance the performance and efficiency of database operations?

Post Lab Task:

Consider the following Relational Table as shown in below.

Patient:

Pid	Fname	Lname	Email	Phno	Address	Date_Admn
200	David	Samson	aaa@gmail.com	7654328976	Mumbai	13-10-2019
201	Bharath	Kumar	bbb@gmail.com	8765438907	Hyderabad	22-01-2020
202	Eswar	Prasad	ccc@gmail.com	9876548838	Vijayawada	15-03-2020
203	Jaya	Laxmi	ddd@gmail.com	8765489765	Delhi	06-07-2020
204	Laxmi	Devi	eee@gmail.com	7654430692	Mumbai	12-03-2020
205	Pramod	Reddy	fff@gmail.com	6543371619	Hyderabad	05-02-2020
206	Charan	Kumar	ggg@gmail.com	5432312546	Vijayawada	22-04-2020
207	Kalyan	Reddy	hhh@gmail.com	4321253473	Delhi	10-07-2020
208	Rajesh	Yadav	iklmn@gmail.com	3210194400	Chennai	03-03-2020
209	Naveen	Kumar	jghfr@gmail.com	2099135327	Hyderabad	22-04-2020

Bednum	Pid
20	200
21	201
22	202
23	203
24	204
25	205
26	206
27	207
28	208
29	209

Shift			
Doctor Id	Day	Starttime	Endtime
100	12-04-2020	9	5
101	20-06-2020	4	8
102	01-08-2020	6	12
105	05-08-2020	13	18
106	13-10-2019	4	8
107	22-01-2020	6	12
108	15-03-2020	13	18
109	06-07-2020	9	5
110	12-03-2020	4	8
111	05-02-2020	6	12
112	22-04-2020	13	18
113	10-07-2020	4	8
114	03-03-2020	6	12
115	22-04-2020	13	18

Appointment	
Pid	Amt
200	200
201	200
202	300
203	600
204	200
205	300
206	200
207	200
208	200
209	600

Questions:

1.Write a PL/SQL program that retrieves all records from the "PATIENT" table and displays the patient details.

2.Create a PL/SQL procedure that takes a patient's ID as input and updates their phone number in the "PATIENT" table. Test the procedure by updating the phone number of a specific patient.

3.Develop a PL/SQL function that calculates the average appointment amount from the "APPOINTMENT" table and returns the result.

4.Create a PL/SQL procedure that inserts a new patient record into the "PATIENT" table using the provided details. Test the procedure by adding a new patient record.

5.Write a PL/SQL function that retrieves the count of patients from the "PATIENT" table and returns the result.

6.Develop a PL/SQL procedure that deletes a patient's record from the "PATIENT" table based on their ID. Test the procedure by deleting a specific patient's record.

7.Write a PL/SQL program that retrieves all the records from the "SHIFT" table and displays the doctor shift details.

8.Create a PL/SQL function that counts the number of patients per bed from the "PCASE" table and returns the result as a cursor.

9.Develop a PL/SQL procedure that updates the end time of a doctor’s shift based on their ID and the date. Test the procedure by updating the end time for a specific doctor’s shift.

10.Write a PL/SQL program that accepts an address as input and retrieves all patient details from the "PATIENT" table for that address.

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:

Lab Session 07

7. Introduction to PL/PGSQL in a Bank Management System

Aim:

The aim of this experiment is to familiarize students with PL/PGSQL by developing procedures, functions, and programs for managing customer accounts and transactions in a bank management system.

Description:

This lab introduces students to the PL/PGSQL language and its role in developing database-centric applications. Through hands-on practice, students will learn to declare variables, use control structures, handle SQL statements, and create reusable procedures and functions for tasks like managing customer accounts, handling transactions, and retrieving account details. By working on real-world banking scenarios, students will gain experience in writing modular, efficient, and secure code for robust database operations.

Pre-Lab Tasks:

1.Compare SQL & PL/pgSQL.

2.How do you declare and initialize variables in a PL/pgSQL program in PostgreSQL?

3.Can a PL/pgSQL function return multiple values in PostgreSQL? How?

4.What are the benefits of encapsulating logic in PL/pgSQL procedures/functions in a banking application?

5.How do COMMIT and ROLLBACK work in PostgreSQL PL/pgSQL transaction blocks?

6.Distinguish between an anonymous PL/pgSQL block and a stored procedure.

In-Lab Tasks:

Use the following sample tables:

CUSTOMER(cust_id, name, address, phone)

ACCOUNT(account_id, cust_id, balance)

TRANSACTION(txn_id, account_id, amount, txn_type, txn_date)

1.Write a PL/pgSQL program that retrieves all records from the CUSTOMER table and displays customer details.

2.Create a procedure that updates a customer's phone number based on their customer ID. Test it.

3.Create a function that returns the average balance from the ACCOUNT table.

4.Write a procedure to insert a new account for an existing customer. Test the procedure.

5.Write a function that returns the total number of customers in the bank.

6.Write a procedure that deletes a customer record based on cust_id. Test the deletion.

7.Write a PL/pgSQL program to retrieve and display all account details from the ACCOUNT table.

8.Create a function that returns the total number of transactions per account as a cursor.

9.Develop a procedure to update the balance of an account after a deposit. Test it.

10.Write a program that accepts an account balance threshold and retrieves all accounts below that balance.

Viva-Voce Questions (In-Lab):

1.How do procedures and functions differ in PL/pgSQL? When do you use each?

2.What are the benefits of using stored procedures in transaction-heavy applications like banking?

3.Explain the significance of cursors in PL/pgSQL.

4.What is exception handling in PL/pgSQL, and how is it used in financial applications?

Post-Lab Task:

Based on the following relational tables:

LOAN(loan_id, cust_id, amount, status)

PAYMENT(payment_id, loan_id, payment_date, amount_paid)

PL/pgSQL Post-Lab Programs:

i) Write a PL/pgSQL program that retrieves all loans with status = 'Pending'.

ii) Create a procedure to update a loan status based on loan_id.

iii) Write a function to calculate the total amount paid for a given loan.

iv) Create a procedure to insert a new payment record.

v) Write a function that returns the count of all loans for a given customer.

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:

Lab Session 08

8.Implementation of Transaction in SQL Server

Aim:

The aim of implementing transactions in SQL Server is to ensure the atomicity, consistency, isolation, and durability (ACID) properties of database operations.

Description:

Transactions in SQL Server provide a mechanism to group multiple database operations into a single logical unit, ensuring that they are executed as a whole or not at all. Transactions ensure the atomicity, consistency, isolation, and durability (ACID) properties of database operations. In SQL Server, transactions are managed using the BEGIN TRANSACTION, COMMIT, and ROLLBACK statements.

Pre-Requisites:

This Section contains the list of Software/Tools or required knowledge (Glossary) to complete the task under the Laboratory Session.

Pre Lab-Task:

1.What are the Transactions?

2.Explain ACID properties?

3.List the transaction states.

4.Difference between commit and roll back operations.

In Lab Task:

1.Create user and implement the following commands on relation (Emp and Dept)

2.Delete those records from the table which have age = 20 and then ROLLBACK the changes in the database.

3.Update those records from the table which have age = 30 and then ROLLBACK the changes in the database.

4.Insert the records from the table Employees ('Tom Harris', 28, 'Sales'), ('Sara Parker', 22, 'HR').

5.Delete those records from the table which have age = 25 and then COMMIT the changes and then try to use ROLLBACK to the previous save point. What have you observed at this stage?

6.Develop a query to grant all privileges of employees table into departments table.

7.Develop a query to grant some privileges of employees table into departments table.

8.Develop a query to revoke all privileges of employees table from departments table.

9.Develop a query to revoke some privileges of employees table from departments table.

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:

Lab Session 09

9.Indexing Experimentation

Aim:

To evaluate the impact of various index types on query performance in a dataset, and to analyze the effectiveness of indexes in optimizing data retrieval operations.

Description:

Indexes are essential in database management systems to expedite data retrieval operations. Different index types (e.g., B-tree, Hash, Bitmap) have distinct characteristics that can significantly impact query performance. In this experiment, we aim to create multiple index types on a dataset, measure query performance, and analyze the influence of indexes on query execution time.

Prerequisites:

Understanding of database management systems and SQL.

Familiarity with index structures and their functionalities.

Basic knowledge of query optimization techniques.

Pre-Lab:

1.Define what an index is in the context of a database.

2.Why are indexes crucial for efficient data retrieval?

3.Explain the differences between B-tree, Hash, and Bitmap indexes. What are the strengths and weaknesses of each type?

4.How does the choice of index type affect query performance in a database?

5.Discuss the factors that influence the selection of an appropriate index type for a given dataset.

In Lab Task:

Select a suitable dataset for experimentation. Here we taken Orders dataset as shown below.

order_id	customer_id	product_id	order_date	quantity	total_price
1	101	201	2024-05-01	2	50.00
2	102	202	2024-05-02	1	25.00
3	103	201	2024-05-02	3	75.00
4	101	203	2024-05-03	2	60.00
5	104	204	2024-05-04	1	30.00
6	102	205	2024-05-05	4	100.00
7	101	201	2024-05-06	2	50.00
8	105	206	2024-05-06	3	90.00
9	103	207	2024-05-07	1	40.00
10	106	208	2024-05-08	2	55.00

1.Design a set of SQL queries representing various data retrieval operations, including simple lookups, range queries.

2.Create multiple index types (e.g., B-tree, Hash, Bitmap) on selected columns of the dataset.

3.Normal Query without INDEX to measure the execution time of queries

4.Query with BTREE INDEXING to measure the execution time of queries

5.Normal Query without INDEX to measure the execution time of queries

6.Query with BTREE INDEXING to measure the execution time of queries

Post-Lab:

1.What was the purpose of creating different index types on the dataset?

2.Describe the process you followed to create the different index types. Which indexing techniques did you use?

3.How did you measure the query performance before and after creating indexes?

4.What were the key metrics used to analyse the impact of indexes on query performance?

5.Did you observe any significant differences in query performance after implementing indexes? If so, elaborate on the changes.

6.Discuss any challenges or limitations encountered during the experiment.

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:

Lab Session 10

10.Query Optimization

Aim:

The aim of this experiment is to understand and apply various query optimization strategies to enhance the performance of database queries.

Description:

This involves analysing the execution plans of queries, identifying bottlenecks, and implementing optimization techniques such as indexing, query rewriting, and the use of efficient algorithms. By the end of the experiment, students should be able to: Understand the importance of query optimization in database management systems (DBMS) and how it impacts overall system performance. Analyse query execution plans to identify performance bottlenecks and inefficient operations. Implement indexing strategies to improve the speed of data retrieval operations. Apply query rewriting techniques to transform queries into more efficient forms without changing their semantics. Evaluate the performance improvements achieved through various optimization strategies by comparing query execution times before and after optimization.

Pre-Requisites: PostgreSQL, TerraER Tool, Windows/ Ubuntu/CentOS/Debian, DBMS Concepts.

Pre-Lab:

1.What is query optimization in the context of a DBMS?

2.Why is query optimization important for database performance?

3.Explain the role of the query optimizer in a DBMS.

4.What is an execution plan in the context of query optimization?

5.What is query rewriting, and how can it improve query performance?

6.What is the purpose of analysing query execution plans?

In-Lab:

Query optimization strategies for improving query performance.

a) Analysing Execution Plans

Objective: Understand how to generate and analyse execution plans to identify performance bottlenecks.

Steps:

1.Create and Populate Tables

2.Write Complex SQL Queries

3.Generate Execution Plans.

Questions:

1.Create tables for employee and department with appropriate key constraints?

2.Write Complex SQL Queries Involving Joins and Subqueries.

3.Use the EXPLAIN command to generate execution plans for each query.

4.Identifying Performance Bottlenecks

Objective: Identify performance bottlenecks in SQL queries and understand their impact on query performance.

Questions:

1.Create tables for customers and orders with appropriate key constraints?

2.To write SQL queries with potential performance issues, identify these issues using the EXPLAIN command, and understand their impact on query performance.

Query Optimization

Query Optimization in a Sales Database

A **Sales Database** typically contains tables like:

- **Customers** (CustomerID, Name, Contact, etc.)
- **Orders** (OrderID, CustomerID, OrderDate, TotalAmount, etc.)
- **Products** (ProductID, Name, Price, etc.)
- **OrderDetails** (OrderDetailID, OrderID, ProductID, Quantity, Price, etc.)

Objective: To optimize a query in a sales database by identifying and resolving performance bottlenecks.

Apply Optimization Techniques:

1.Add indexes on frequently queried columns.

2. Rewrite complex joins and subqueries.

3.Use EXISTS instead of IN for subqueries when appropriate

5.Write a query to retrieve sales data for products in a specific category

6. Create appropriate indexes to optimize the query.

5.Measure and compare the performance before and after optimization using execution plans and query execution times.

6.Optimize SQL queries based on identified performance bottlenecks to improve query efficiency.

Viva-Voce Questions (In-Lab):

1.What is the significance of using indexes in SQL queries, and how do they improve query performance?

2.Explain how the EXPLAIN statement can be used to identify performance bottlenecks in SQL queries.

3.What are composite indexes, and when should they be used in query optimization?

4.How can query performance be affected by the use of functions in the WHERE clause, and what are the alternatives to mitigate this impact?

5.Explain the importance of query execution time measurement and the methods to measure and compare query performance before and after optimization.

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:

Lab Session 11

11.Query Optimization in a Library Management System

Aim:

To apply query optimization techniques to a library management system database in order to improve the performance of search and reporting queries.

Description:

In this experiment, students will work with a library database containing books, authors, borrowers, and transactions. The focus is to identify inefficient SQL queries, analyze their execution plans, and apply optimization techniques such as indexing, query rewriting, and filtering to enhance performance. By the end of the experiment, students will:

Understand query execution flow in multi-table queries.

Analyze performance issues using EXPLAIN ANALYZE.

Optimize queries using indexes and appropriate query rewriting techniques.

Compare performance metrics before and after optimization.

Pre-Requisites:

PostgreSQL installed

DBMS Concepts: Joins, Subqueries, Indexing

Pre-Lab Questions:

1.What are the key components of a query execution plan?

2.How does indexing affect performance in large datasets?

3.When should you use JOIN instead of subqueries?

4.What are the advantages of filtering data early in a query?

5.How does PostgreSQL use cost-based optimization?

In-Lab Activities:

Scenario: Library Management System

Tables:

books(book_id, title, author_id, category)

authors(author_id, name)

borrowers(borrower_id, name)

transactions(trans_id, book_id, borrower_id, borrow_date, return_date)

1.Table Creation and Data Insertion

2.Write Queries with Potential Performance Issues

3.Analyze Execution Plan

4.Optimize the Query Using JOIN

5.Create Indexes to Improve Performance

6.Performance Comparison

Viva-Voce Questions (In-Lab):

1.Why is it better to use JOIN instead of IN in certain subquery cases?

2.How does indexing improve performance for search queries?

3.What is the difference between a sequential scan and an index scan in PostgreSQL?

4.How do you decide which columns to index?

5.What is the effect of using wildcard characters or functions on indexed columns?

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:

Lab Session 12

12.NoSQL Indexing

Aim:

The aim of this experiment is to improve query performance and data retrieval efficiency by creating data structures that allow for faster search and retrieval of information. Indexing enables MongoDB to locate and retrieve specific documents more efficiently, reducing the need to scan the entire collection.

Description:

Indexing in MongoDB is a mechanism that enhances query performance and data retrieval efficiency. It involves creating data structures, called indexes, that provide a way to access and retrieve specific information from a MongoDB collection quickly. When indexing is applied to a field or a combination of fields, MongoDB creates a separate data structure that organizes the indexed field(s) in a sorted manner. This structure, known as the index, contains references to the actual documents in the collection.

Pre-Requisites:

Choose a NoSQL database to work with (e.g., MongoDB, Cassandra, Redis)

Install the database software on your local machine or use a cloud-based service.

Set up a development environment with appropriate drivers or libraries for your chosen NoSQL database.

Access the database through a command-line interface or a graphical user interface.

Pre-Lab:

1.What is the role of the Wired Tiger storage engine in MongoDB?

2.What is the purpose of the $graphLookup operator in MongoDB aggregation?

3.What is the difference between a TTL index and a regular index in MongoDB?

4.What is the purpose of the $replaceRoot operator in MongoDB aggregation?

5.What is the role of the mongodump command in MongoDB?

IN LAB:

1.To demonstrate the optimization of queries using indexes, we have to use a fairly large database

2.Develop a query to demonstrate Text search using catalog data collection for a given word.

Viva-Voice Questions (In-Lab):

1.What is the purpose of indexing in MongoDB?

2.How does indexing improve query performance in MongoDB?

3.Explain the concept of an index in MongoDB.

4.What are the benefits of creating indexes in frequently queried fields?

5.Can you create indexes on multiple fields in MongoDB? If yes, how?

Post Lab:

MongoDB Indexing using a MongoDB Driver Objective: To create and use indexes in MongoDB for improved query performance using a MongoDB driver (e.g., pymongo for Python).

Materials: MongoDB installed and running (follow the installation steps mentioned earlier)

Procedure:

- Install the MongoDB driver for your chosen programming language (e.g., pymongo for Python).
- Open a text editor or IDE and create a new script file.
- Import the required MongoDB driver module in your script.
- Establish a connection to the MongoDB server using the driver's connection methods.
- Switch to the desired database using the driver's methods (e.g., client.get_database() in pymongo).
- Identify the collection on which you want to create an index. For example, consider a collection named mycollection.
- To create an index on a specific field, use the appropriate method provided by the driver (e.g., db.collection_name.create_index() in pymongo). For example, to create an index on the "name" field, use the following code: db.collection_name.create_index([("name", pymongo.ASCENDING)]) This code creates an ascending index on the "name" field. You can use pymongo.DESCENDING for a descending index.
- Once the index is created, perform query operations on the collection and observe the improved query performance.
- Optionally, you can use the appropriate method provided by the driver (e.g., db.collection_name.index_information() in pymongo) to view the list of indexes created on a collection.

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:

Lab Session 13

13.NO SQL Aggregation

Aim:

The aim of this experiment is to provide a powerful and flexible framework for performing complex data processing and analysis tasks on collections. Aggregation operations allow for the transformation, grouping, filtering, and computation of data, enabling developers to derive meaningful insights from their MongoDB data.

Description:

Aggregation operations in MongoDB provide a powerful framework for performing complex data processing and analysis tasks on collections. These operations allow developers to derive meaningful insights from their MongoDB data by transforming, grouping, filtering, and computing data in a flexible and efficient manner.

Pre-Requisites: This Section contains the list of Software/Tools or required knowledge to complete the task under the Laboratory Session.

Pre lab:

1.What is the purpose of the $redact operator in MongoDB aggregation?

2.What is the purpose of the $out operator in MongoDB aggregation?

3.What is the purpose of the $lookup pipeline in MongoDB?

In lab:

1.To demonstrate aggregation operations such as $avg, $min, $max, $push, and $addToSet in MongoDB, we will use a Sales collection. This collection will contain documents representing sales transactions.

2.Execute Aggregation Pipeline and its operations (pipeline must contain match, group, sort, project, $skip etc.)

Viva-Voice Questions (In-Lab):

1.How is data stored in collections in MongoDB?

2.What is the default port on which MongoDB listens?

3.How does MongoDB provide high availability and disaster recovery?

4.Describe how to read data from a MongoDB collection.

Post lab:

To display a summary of reviews in an e-commerce collection, we can assume the ecommerce database contains a products collection with documents structured to include reviews. Each product document could have a reviews array with review details such as rating, comment, and user.

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: