 ***DEPARTMENT OF INFORMATION TECHNOLOGY***

|  |  |
| --- | --- |
| Semester | S.E Semester III– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 01 | |
| Experiment Title | Problem Definition. | |
| Resources / Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  PostgreSQL |
| Objectives  (Skill Set / Knowledge Tested / Imparted) |  | |
| Theory of Operation | It becomes necessary for administration of libraries to keep a continuous check on the books issued and returned. This task if carried out manually will be tedious and includes chances of mistakes. These errors are avoided by ***Library Management System*** which helps in monitoring and controlling the transactions in a library .Library Management System mainly focuses on basic operations in a library like adding new books, and updating new information, searching books and members and return books.  ***LIBRARY MANAGEMENT*** gives us the complete information about the library. We can enter the record of new books and retrieve the details of books available in the library. We can issue the books to the members and maintain their records and can also check how many books are issued and stock available in the library.   * It will give a brief information about the members of the library and the employees working in library. * It will also provide information about the books and their respective publishers. * It will help in keeping a record of the number of books issued to a particular member. | |

** *DEPARTMENT OF INFORMATION TECHNOLOGY***

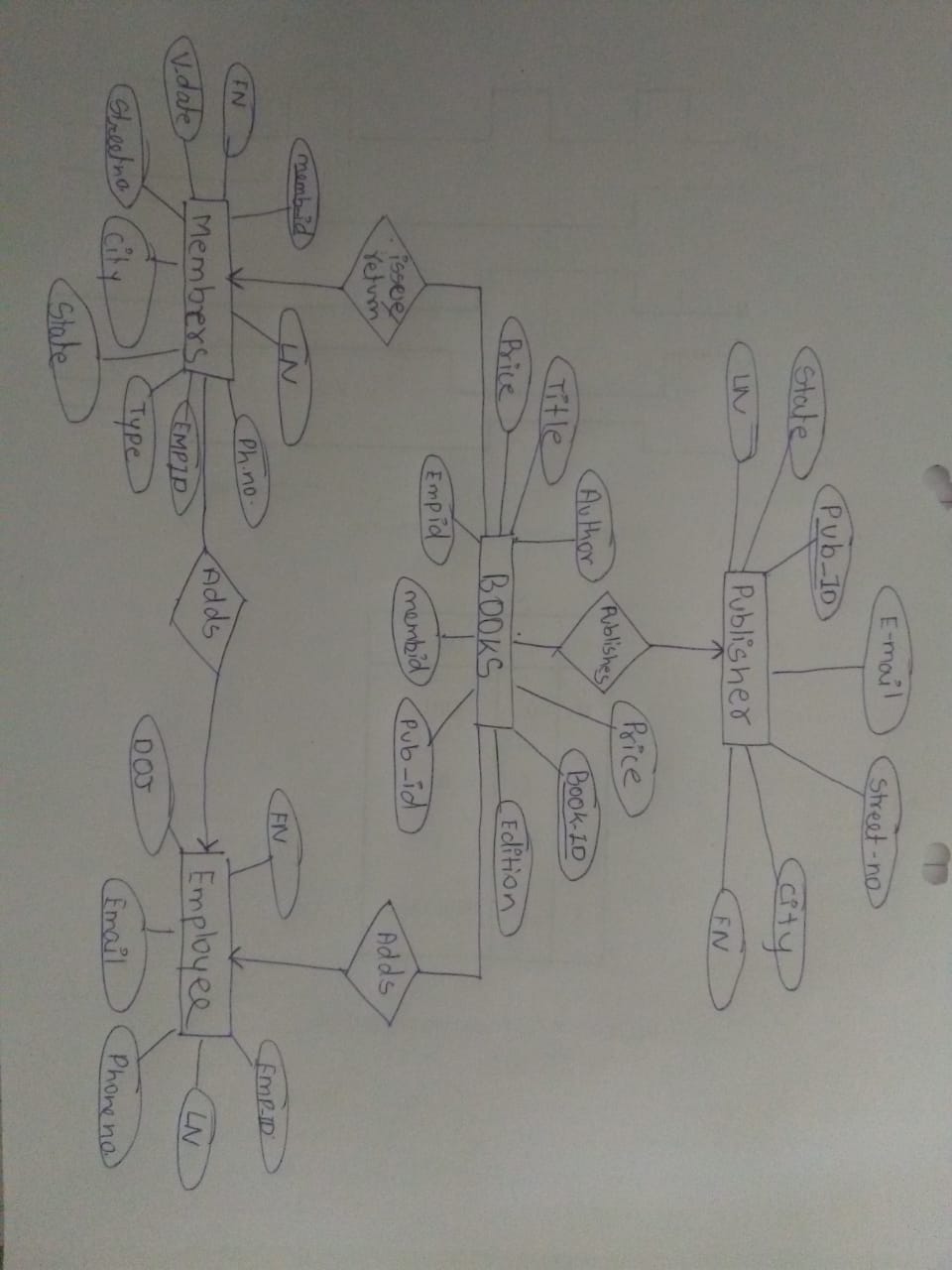
Q

|  |  |
| --- | --- |
| Semester | S.E Semester III– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Deepali Vohra |
| Assisting Teachers | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 02 | |
| Experiment Title | E-R Diagram. | |
| Resources / Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  PostgreSQL |
| Objectives  (Skill Set / Knowledge) | Entity set:   * **Publisher** * **Books** * **Employee** * **Member**   Relations:   * Publisher -**publishes**- Books(1 to many) * Employee -**adds**- Books(1 to many) * Books-**issued/return**- Member(many to 1) * Employee-**adds**-Member(1 to many) | |

**E.R Diagram**



*DEPARTMENT OF INFORMATION TECHNOLOGY*

|  |  |
| --- | --- |
| Semester | S.E Semester III– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 03 | |
| Experiment Title | ER to Relational | |
| Resources / Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  Oracle 10g and Isql plus  PostgreSQL |
| Objectives  (Skill Set / Knowledge Tested / Imparted) | To create relational model from ER model. | |
| Theory of Operation | **Explanation/ Stepwise-Procedure/ Algorithm:**  Mapping of ER/ERR model to Relational model:  Step 1.Strong Entity Set  It reduces to Relational schema with same attributes.  Step 2.Weak Entity Set  It is mapped as a separate schema with combination of primary key from identifying strong entity set and attributes of weak entity set.  There is a foreign key constraint that each row in weak entity set there is a corresponding strong entity set row.  Step 3.Relationships  1. One to One: Extra attribute is added to either of the two entity set schema corresponding to entity set participating in the relationship.  2.Many to One/One to Many : Addition of extra attribute to the many side of entity set schema with primary key of the one side entity set as a foreign key constraint.  3. Many to Many: The relationship is mapped as a separate schema with the primary keys of participating schema.  Step 4.Multivalued attribute  A multivalued attribute is mapped as a separate schema with attributes as primary key of Entity set and the multivalued attribute.  Step 5:Composite attribute  A composite attribute is flattened out by creating a separate attribute for each component attribute.  **Program code and Output:**  **Sample Schema:**   * Publisher (**pub\_id**, street\_no, city, state, FN, LN,email) * Books (**book\_id**, Author, title, price, edition, *pub\_id,emp\_id,mem\_id*) * Employee(**emp\_id**, FN, LN, DOJ) * Member(**mem\_id**, FN, LN, V\_date,street\_no, city, state,membertype,phone\_no,*emp\_id*) | |

 *DEPARTMENT OF INFORMATION TECHNOLOGY*

|  |  |
| --- | --- |
| Semester | S.E Semester III– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Deepali Vohra |
| Assisting Teachers | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 04 | |
| Experiment Title | Data Definition Language (DDL) Command. | |
| Resources / Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  PostgreSQL |
| Objectives  (Skill Set / Knowledge) | To create relational database in Oracle using SQL. | |
| Theory of Operation  Example & Output | **DDL Commands**  Stands for "Data Definition Language." A DDL is a language used to define data structures and modify [data](https://techterms.com/definition/data). For example, DDL commands can be used to add, remove, or modify [tables](https://techterms.com/definition/table) within in a [database](https://techterms.com/definition/database). DDLs used in database applications are considered a subset of [SQL](https://techterms.com/definition/sql), the Structured Query Language. However, a DDL may also define other types of data, such as [XML](https://techterms.com/definition/xml).   * **Create Command**: The create command is used to establish a new database, table, index, or stored procedure. The CREATE statement in SQL creates a component in a relational database management system (RDBMS).   Syntax: create table table\_name  (  Column\_name1 datatype(size),  Column\_name2 datatype(size),  Column\_name3 datatype(size),  Column\_name4 datatype(size),  Column\_name5 datatype(size)  );   * **Alter Command**: The SQL ALTER TABLE command is used to modify the definition (structure) of a table by modifying the definition of its columns. The ALTER command is used to perform the following functions Add, drop, modify table columns. * To add a new column in the table   Syntax: alter table table\_nameadd(column\_name datatype(size));   * To modify the datatype and width of a column.   Syntax: alter table table\_name modify (column\_name datatype(size));   * **Rename Command:** With RENAME statement you can rename a table. Some of the relational database management system (RDBMS) does not support this command, because this is not standardizing statement. For example renaming a table through MS SQL Server you must use storage procedure SP\_RENAME.   Syntax: rename old table\_name to new table\_name   * **Truncate Command:** Removes all rows from a table or specified partitions of a table, without logging the individual row deletions. TRUNCATE TABLE is similar to the DELETE statement with no WHERE clause; however, TRUNCATE TABLE is faster and uses fewer system and transaction log resources.   Syntax: truncate table emp;   * .**Drop Command**:The SQL DROP command is used to remove an object from the database. If you drop a table, all the rows in the table is deleted and the table structure is removed from the database. Once a table is dropped we cannot get it back, so be careful while using DROP command.   Syntax: drop table table\_name;  **Create**  CREATE TABLE BOOKS(  book\_id VARCHAR(10)PRIMARY KEY,  author VARCHAR(15),  title VARCHAR(15),  price INT,  edition INT,  pub\_id INT,  emp\_id INT,  mem\_id INT  );  CREATE TABLE EMPLOYEE(  emp\_id INT PRIMARY KEY,  FN VARCHAR(15),  LN VARCHAR(15),  DOJ date  );  CREATE TABLE MEMBER(  mem\_id INT,  FN VARCHAR(15),  LN VARCHAR(15),  V\_date DATE,  Street\_no INT,  city VARCHAR(10),  state VARCHAR(10),  membertype VARCHAR(15),  phone\_no BIGINT,  emp\_id INT  );  CREATE TABLE PUBLISHER(  pub\_id INT,  street\_no int,  city VARCHAR(100),  state VARCHAR(100),  FN VARCHAR(100),  LN VARCHAR(100),  email VARCHAR(100)  );    Insert values:  **INSERT INTO** PUBLISHER **VALUES** (100,172,'DHULE','MAHARSHTRA','KETAN','PATIL','K@GMAIL.COM');  **INSERT INTO** PUBLISHER **VALUES** (101,372,'NASHIK','MAHARSHTRA','CHETAN','WARKE','C@GMAIL.COM');  **INSERT INTO** PUBLISHER **VALUES** (102,272,'DHULE','MAHARSHTRA','NIKETAN','CHAUDHARI','N@GMAIL.COM');  **INSERT INTO** PUBLISHER **VALUES** (103,172,'HYDERABAD','TELANGANA','AKASH','RAJPUT','A@GMAIL.COM');  **INSERT INTO** EMPLOYEE **VALUES** (1001,'PRANJAL','NAIK', TO\_DATE('17/12/2015','DD/MM/YYYY'));  **INSERT INTO** EMPLOYEE **VALUES** (1002,'GANESH','SANKHE', TO\_DATE('11/12/2014','DD/MM/YYYY'));  **INSERT INTO** EMPLOYEE **VALUES** (1003,'JITENDRA','JOSHI', TO\_DATE('12/08/2017','DD/MM/YYYY'));  **INSERT INTO** EMPLOYEE **VALUES** (1004,'SANTOSH','WAGHMARE', TO\_DATE('15/02/2013','DD/MM/YYYY'));  **INSERT INTO** BOOKS **VALUES**(2001,'NAVATHE','DBMS',700,6,100,1001,201);  **INSERT INTO** BOOKS **VALUES**(2002,'KUMBOJHKAR','AM-II',450,10,101,1002,205);  **INSERT INTO** BOOKS **VALUES**(2003,'DAYAL','MECHANICS',550,6,103,1003,202);  **INSERT INTO** BOOKS **VALUES**(2004,'CHETAN\_B','2\_STATES',300,1,102,1004,203);  **INSERT INTO** BOOKS **VALUES**(2005,'JK\_ROWLING','HARRY\_POTTER',1000,1,102,1004,205);  **INSERT INTO** BOOKS **VALUES**(2006,'S\_CHAND','JEE\_PREP',650,22,101,1002,202);  **INSERT INTO** MEMBER **VALUES** (201,'RITESH','RAHATAL',TO\_DATE('17/12/2016', 'DD/MM/YYYY'),420,'NASHIK','MAHARASHTRA','GOLD',9321343467,1001);  **INSERT INTO** MEMBER **VALUES** (202,'GITENDRA','GADA', TO\_DATE('20/01/2015', 'DD/MM/YYYY'),421,'SURAT','GUJARAT','SILVER',7721343467,1001);  **INSERT INTO** MEMBER **VALUES** (203,'RAMESH','OAK',TO\_DATE('22/06/2004', 'DD/MM/YYYY'),422,'PUNE','MAHARASHTRA','SILVER',9861343467,1002);  **INSERT INTO** MEMBER **VALUES** (204,'AMAN','SINGH',TO\_DATE('13/07/2018', 'DD/MM/YYYY'),420,'AMRAVATI','MAHARASHTRA','GOLD',9283434670,1003);  **INSERT INTO** MEMBER **VALUES** (205,'RAJESH','NAGPURKAR',TO\_DATE('25/12/2013', 'DD/MM/YYYY'),428,'NAGPUR','MAHARASHTRA','BRONZE',9901343467,1002);    Alter Example:  **ALTER TABLE** PUBLISHER  **DROP** EMAIL;  **ALTER TABLE** PUBLISHER  **ADD** EMAIL VARCHAR(20);    Truncate Example :  **TRUNCATE TABLE** PUBLISHER;    DROP Example :  **DROP TABLE** BOOKS;  **DROP TABLE** EMPLOYEE;  **DROP TABLE** MEMBER;  **DROP TABLE** PUBLISHER**;** | |

*DEPARTMENT OF INFORMATION TECHNOLOGY*

|  |  |
| --- | --- |
| Semester | S.E SemesterIII– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Deepali Vohra |
| Assisting Teachers | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
|  |  | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 05 | |
| Experiment Title | Data Manipulation Language (DML) Command. | |
| Resources / Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  PostgreSQL |
| Objectives  (Skill Set / Knowledge Tested / Imparted) | To insert and manipulate data in the table | |
| Theory of Operation  Example & Output | **DML Commands**  A data manipulation language (DML) is a family of syntax elements similar to a computer programming language used for selecting, inserting, deleting and updating data in a database. Performing read-only queries of data is sometimes also considered a component of DML.   * **Select Command:** The SQL SELECT statement returns a result set of records from one or more tables. A SELECT statement retrieves zero or more rows from one or more database tables or database views. In most applications, SELECT is the most commonly used data query language (DQL) command. * To select all columns:   Syntax: select column-names  from table-name;   * Where clause with a select statement:   Syntax : Select Colum\_name  From table\_name  Where Condition;   * **Insert Command:** The SQL INSERT INTO Statement is used to add new rows of data to a table in the database.   Syntax: insert into table\_namevalues(value 1,value 2);   * **Update Command:** An SQL UPDATE statement changes the data of one or more records in a table. Either all the rows can be updated, or a subset may be chosen using a condition. The UPDATE statement has the following form: UPDATE table\_nameSETcolumn\_name = value [, column\_name = value ...]   Syntax:UPDATE table\_name  set column1 = value1, column2 = value2, ... WHERE condition;   * **Delete Command:** In the database structured query language (SQL),   the DELETE statement removes one or more records from a table. A subset may be defined for deletion using a condition, otherwise all records are removed.  Syntax :DELETE FROM table\_name  WHERE condition;   * Delete all records:   Syntax: DELETE FROM table\_name;  SELECT  **Example:**  Q1) DISPLAY ALL THE ATTRIBUTES OF BOOKS:  ANS) **SELECT** \*  **FROM** BOOKS;    Q2) DISPLAY ALL THE ATTRIBUTES OF PUBLISHER HAVING ID 102:  ANS) **SELECT** \*  **FROM** PUBLISHER  **WHERE** pub\_id=102;    Q3) DISPLAY ALL THE ATTRIBUTES OF BOOKS WHICH HAVE BEEN ADDED BY EMPLOYEE WHOSE FIRST NAME IS SANTOSH:  ANS) **SELECT** \*  **FROM** EMPLOYEE e , BOOKS b  **WHERE** e.emp\_id=b.emp\_id  **AND** FN=’SANTOSH’;    **Insert Example:**  **INSERT INTO** PUBLISHER **VALUES**(104,492,'BHUSAWAL','MAHARASHTRA','SHREE','LAD','S@GMAIL.COM');  **INSERT INTO** EMPLOYEE VALUES (1005,'PRAJWAL','WASNIK',TO\_DATE('26/08/2014', 'DD/MM/YYYY'));  **select** \* **from** PUBLISHER    **Delete Example:**  **Delete from** PUBLISHER; | |

 *DEPARTMENT OF INFORMATION TECHNOLOGY*

|  |  |
| --- | --- |
| Semester | S.E Semester III– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Deepali Vohra |
| Assisting Teachers | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 06 | |
| Experiment Title | GROUP BY clause in SQL | |
| Resources / Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  PostgreSQL |
| Objectives | To Manipulate Data from table using Clause. | |
| Theory of Operation  Example & Output | The SQL **GROUP BY** clause is used in collaboration with the SELECT statement to arrange identical data into groups. This GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.  Syntax:  SELECT column1, column2  FROM table\_name  WHERE [ conditions ]  GROUP BY column1, column2  Example :  Consider the PLAYER table is having the following records  number of player in the database from each country, then the GROUP BY query would be as follows.  Q1)COUNT NUMBER OF PUBLISHER WHO LIVE IN SAME STATE:  ANS)  **SELECT** COUNT(\*),STATE  **FROM** PUBLISHER  **GROUP BY** STATE;    Q2)COUNT NUMBER OF BOOKS WHOSE PRICE IS GREATER THAN 500:  ANS) **SELECT** COUNT(\*),PRICE  **FROM** BOOKS  **GROUP BY** PRICE  **HAVING** PRICE>500;    Q3)DISPLAY THE NUMBER OF MEMBERS HAVING DIFFERENT MEMBERSHIP:  ANS) **SELECT** COUNT(\*),MEMBERTYPE  **FROM** MEMBER  **GROUP BY** MEMBERTYPE;    Q4) DISPLAY THE AVERAGE PRICE OF A BOOK HAVING EDITION NUMBER 6 AND 1:  ANS) **SELECT** AVG(PRICE),EDITION  **FROM** BOOKS  **GROUP BY** EDITION  **HAVING** EDITION=6 OR EDITION=1; | |

 ***DEPARTMENT OF INFORMATION TECHNOLOGY***

|  |  |
| --- | --- |
| Semester | S.E Semester III– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 07 | |
| Experiment Title | Complex nested Queries for update and delete | |
| Resources / Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  PostgreSQL |
| Theory of Operation | Nested Queries:  In nested queries, a query is written inside a query. The result of inner query is used in execution of outer query  There are mainly two types of nested queries:   * **Independent Nested Queries:** In independent nested queries, query execution starts from innermost query to outermost queries. The execution of inner query is independent of outer query, but the result of inner query is used in execution of outer query. Various operators like IN, NOT IN, ANY, ALL etc. are used in writing independent nested queries. * **Co-related Nested Queries:** In co-related nested queries, the output of inner query depends on the row which is being currently executed in outer query.   Q1) Delete all the books whose employee ID is 1004:  Ans) **DELETE** **FROM** BOOKS b  **WHERE** b.emp\_id **IN**( **SELECT** emp\_id  **FROM** EMPLOYEE  **WHERE** emp\_id =1004);    Q2)Update Member type GOLD whose MEM\_ID is 202:  Ans)  **UPDATE** MEMBER  **SET** membertype='GOLD’  **WHERE** mem\_id **IN** (**SELECT** mem\_id  **FROM** MEMBER  **WHERE** mem\_id =202);    Q3)Delete Employee whose FN='PRANJAL' and LN=`NAIK`:  Ans)  **DELETE** **FROM** EMPLOYEE e  **WHERE** e.FN='PRANJAL' **AND** e.LN='NAIK' | |

*DEPARTMENT OF INFORMATION TECHNOLOGY*

|  |  |
| --- | --- |
| Semester | S.E Semester III– Information Technology |
| Subject | DATABASE MANAGEMENT SYSTEM |
| Subject  Professor In-charge | Prof. Deepali Vohra |
| Assisting Teachers | Prof. Vidya Chitre |
| Laboratory No. | L-05 |

|  |  |  |
| --- | --- | --- |
| Student Name | Pranjal Nandeshwar– 18101A0031  Prajwal Gawande – 18101A0032  Chinmay Chalke – 18101A0044  Paresh Devlekar – 18101A0047 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number | 08 | |
| Experiment Title | Views in SQL | |
| Resources/  Apparatus Required | Hardware:  Basic Computer with Windows or Linux OS installed. | Software:  PostgreSQL |
| Objectives | To perform operations on database like views. | |
| Theory of Operation  Output & Example | **Database Views**  A view is nothing more than a SQL statement that is stored in the database with an associated name. A view is actually a composition of a table in the form of a predefined SQL query.  A view can contain all rows of a table or select rows from a table. A view can be created from one or many tables which depends on the written SQL query to create a view.  Views, which are a type of virtual tables allow users to do the following −   * Structure data in a way that users or classes of users find natural or intuitive. * Restrict access to the data in such a way that a user can see and (sometimes) modify exactly what they need and no more. * Summarize data from various tables which can be used to generate reports.   **Creating Views**  Database views are created using the **CREATE VIEW** statement. Views can be created from a single table, multiple tables or another view.  To create a view, a user must have the appropriate system privilege according to the specific implementation.  The basic **CREATE VIEW** syntax is as follows −  **Syntax :**  CREATE VIEW view\_name AS  SELECT column1, column2.....  FROM table\_name  WHERE [condition];  **THE WITH CHECK OPTION**  The WITH CHECK OPTION is a CREATE VIEW statement option. The purpose of the WITH CHECK OPTION is to ensure that all UPDATE and INSERTs satisfy the condition(s) in the view definition.  If they do not satisfy the condition, the UPDATE or INSERT returns an error.  The following code block has an example of creating same view Player\_View with the WITH CHECK OPTION.  **Dropping Views**  Obviously, where you have a view, you need a way to drop the view if it is no longer needed. The syntax is very simple and is given below  **Syntax : Drop view**  **View\_name;**    Q1) CREATE A VIEW WHICH DISPLAY BOOK ID AND ITS PRICE:  ANS) **CREATE VIEW** BOOK  **AS**  **SELECT** BOOK\_ID,PRICE  **FROM** BOOKS;  **SELECT** \* **FROM** BOOK  Q2)CREATE A VIEW WHICH DISPLAYS AUTHOR,TITLE,EMPLOYEE ID OF BOOKS WHICH HAVE BEEN ADDED BY EMPLOYEE :  ANS) **CREATE VIEW** V1  **AS**  **SELECT** b.AUTHOR,b.TITLE,b.EMP\_ID  **FROM** EMPLOYEE e , BOOKS b  **WHERE** e.EMP\_ID=b.EMP\_ID;  **SELECT** \* **FROM** V1      Q3)CREATE A VIEW WHICH DISPLAY NAME OF THE PUBLISHER AND THE CITY WHERE HE/SHE LIVES:  ANS) **CREATE VIEW** V2  **AS**  **SELECT** FN,LN,CITY  **FROM** PUBLISHER  **SELECT** \* **FROM** V2 | |