|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Database Management System Laboratory** | **Semester:** | **IV** |
| **Date of Performance:** | **15 / 02 / 2023** | **Batch No:** | **A – 2** |
| **Faculty Name:** | **Prof. Shila Dhande** | **Roll No.:** | **16014022050** |
| **Faculty Sign & Date:** |  | **Grade / Marks:** | **\_\_\_ / 25** |

**Experiment No.: 4**

**Title: Implementation of Database in SQL – DML I**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| **Aim: DML – select, insert, update, delete**   1. Nested queries: And, Or, Not, In, Not In, Exists, Not, Exists, Between, Like, Alias, Distinct 2. Update 3. Delete   **Objective:** To perform various DML Operations and executing nested queries. |

|  |
| --- |
| **COs to be achieved:** |
| **CO3:** Use SQL for Relational database creation, maintenance and query processing. |

|  |
| --- |
| **Books / Journals / Websites Referred:** |
| 1. Dr. P.S. Deshpande, SQL and PL/SQL for Oracle 10g.Black book, Dreamtech Press www.db-book.com 2. Korth, Slberchatz, Sudarshan: “Database Systems Concept”, 6th Edition, McGraw Hill 3. Elmasri and Navathe, “Fundamentals of Database Systems”, 5th Edition, PEARSON Education.   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  **PreLab / Prior Concepts:** |

|  |
| --- |
| **Tools Required:** |
| * Postgresql Software |

|  |
| --- |
| **Theory:** |
| **Select:**  The SQL **SELECT** statement is used to fetch the data from a database table which returns this data in the form of a result table. These result tables are called result-sets. Syntax **–** The basic syntax of the SELECT statement is as follows:  SELECT column1, column2, columnN FROM table\_name;  Here, column1, column2... are the fields of a table whose values you want to fetch. If you want to fetch all the fields available in the field, then you can use the following syntax.  SELECT \* FROM table\_name;  The following code is an example, which would fetch the ID, Name and Salary fields of the customers available in CUSTOMERS table.  SQL> SELECT ID, NAME, SALARY FROM CUSTOMERS;  **Insert:**  The SQL INSERT INTO Statement is used to add new rows of data to a table in the database. Syntax **–** There are two basic syntaxes of the INSERT INTO statement which are shown below:  INSERT INTO TABLE\_NAME (column1, column2, column3,...columnN)  VALUES (value1, value2, value3,...valueN); Example **–** The following statements would create record in the CUSTOMERS table.  INSERT INTO CUSTOMERS (ID, NAME, AGE, ADDRESS, SALARY)  VALUES (1, 'Ramesh', 32, 'Ahmedabad', 2000.00);  **Update:**  The SQL UPDATE Query is used to modify the existing records in a table. You can use the WHERE clause with the UPDATE query to update the selected rows, otherwise all the rows would be affected. Syntax **–** The basic syntax of the UPDATE query with a WHERE clause is as follows:  UPDATE table\_name  SET column1 = value1, column2 = value2...., columnN = valueN  **WHERE [condition]:**  You can combine N number of conditions using the AND or the OR operators.  The following query will update the ADDRESS for a customer whose ID number is 6 in the table.  SQL> UPDATE CUSTOMERS  SET ADDRESS = 'Pune'  WHERE ID = 6;  **Delete:**  The SQL DELETE Query is used to delete the existing records from a table.  You can use the WHERE clause with a DELETE query to delete the selected rows, otherwise all the records would be deleted. Syntax **–** The basic syntax of the DELETE query with the WHERE clause is as follows:  DELETE FROM table\_name  WHERE [condition];  The following code has a query, which will DELETE a customer, whose ID is 6.  SQL> DELETE FROM CUSTOMERS  WHERE ID = 6;  **Operators used in WHERE:**   * = : Equal * > : Greater than * < : Less than * >= : Greater than or equal * <= : Less than or equal * <> : Not equal   The WHERE clause can be combined with AND, OR, and NOT operators.  The AND and OR operators are used to filter records based on more than one  condition:   * The AND operator displays a record if all the conditions separated by AND are * TRUE. * The OR operator displays a record if any of the conditions separated by OR is TRUE. * The NOT operator displays a record if the condition(s) is NOT TRUE.  Syntax **–**  SELECT column1, column2, ... FROM table\_name WHERE condition1 AND condition2 AND  condition3 ...;  SELECT column1, column2, ... FROM table\_name WHERE condition1 OR condition2 OR  condition3 ...;  SELECT column1, column2, ... FROM table\_name WHERE NOT condition;  Example:  SELECT \* FROM Customers WHERE Country=’India’ AND City=’Japan’;  SELECT \* FROM Customers WHERE Country=’America’ AND (City=’India’ OR  City=’Korea’);  **Distinct:**  The SELECT DISTINCT statement is used to return only distinct (different) values.  Syntax –  SELECT DISTINCT *column1*,*column2, ...* FROM *table\_name*;  Example –  SELECT DISTINCT Country FROM Customers;  **LIKE:**  The LIKE operator is used in a WHERE clause to search for a specified pattern in a column.  There are two wildcards used in conjunction with the LIKE operator:   1. % - The percent sign represents zero, one, or multiple characters 2. \_ - The underscore represents a single character  Syntax **–**  SELECT *column1, column2, ...* FROM *table\_name* WHERE *columnN* LIKE *pattern* Examples **–** selects all customers with a CustomerName starting with "a": SELECT \* FROM Customers WHERE CustomerName LIKE 'a%'; selects all customers with a CustomerName that have "r" in the second position: SELECT \* FROM Customers WHERE CustomerName LIKE '\_r%';  **IN and Not In:**  SQL allows testing tuples for membership in a relation. The “in‟ connective tests for set membership where the set is a collection of values produced by select clause. The “not in‟ connective tests for the absence of set membership. The in and not in connectives can also be used on enumerated sets. Example **–** Select fname, mname, lname from employee where designation In (“ceo‟,‟manager‟,‟hod‟,‟assistant‟)  Select fullname from department where relationship not in(“brother”);  **Between:**  The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates. The BETWEEN operator is inclusive. Begin and end values are included. Syntax **–** SELECT column\_name(s)  FROM table\_name  WHERE column\_name BETWEEN value1 AND value2;  Example –  SELECT \* FROM Products WHERE Price BETWEEN 10 AND 20;  **Alias:**  The use of table aliases is to rename a table in a specific SQL statement. The renaming is a temporary change and the actual table name does not change in the database. The column aliases are used to rename a table's columns for the purpose of a particular SQL query.  Syntax –  SELECT column1, column2....  FROM table\_name AS alias\_name  WHERE [condition];  The basic syntax of a column alias is as follows.  SELECT column\_name AS alias\_name  FROM table\_name  WHERE [condition];  Example –  SELECT C.ID, C.NAME, C.AGE, O.AMOUNT  FROM CUSTOMERS AS C, ORDERS AS O  WHERE C.ID = O.CUSTOMER\_ID; |

|  |
| --- |
| **Implementation Details (Problem Statement, Query and Screenshots of Results):** |
|  |

|  |
| --- |
| **Post Lab Subjective / Objective Type Questions:** |
| **In SQL, which of the following is not a data Manipulation Language Commands?****Delete****Truncate****Update****Create****Write SQL query for following statements:** Data -   **Retrieve all student who his grade has not been awarded**  **Find the names of all instructors in the Computer Science department.**  **Find the names of all student whose name starts with ‘S’.**  **Find the names of instructors with salary amounts between $90,000 and $100,000.** |

|  |
| --- |
| **Conclusion:** |
| In this experiment, we practiced basic SQL operations like SELECT, INSERT, UPDATE, and DELETE, as well as nested queries using various operators. These exercises help in understanding database manipulation techniques using SQL. |

**Signature of faculty in-charge with Date:**