****

**Department of Computer Science and Engineering**

**LAB MANUAL**

**Branch:** B. Tech- CSE

**Year & Semester:** 3-year 5sem

**Database Management System**

**(PCS 503)**

Prepared by

Parul Madan (AP)

**Vision and Mission of the Department of Computer Sc. and Engineering**

**Vision**

To impart quality education for producing world class technocrats and entrepreneurs with sound ethics, latest knowledge and innovative ideas in Computer Science and Engineering to meet industrial needs and societal expectations.

**Mission**

1. To impart world class value based technical education in all aspects of Computer Science and Engineering through state of the art infrastructure and innovative approach.
2. To produce ethical, motivated and skilled engineers through theoretical knowledge and practical applications.
3. To inculcate ability for tackling simple to complex individually as well as in a team..
4. To develop globally competent engineers with strong foundations, capable of “out of the box” thinking so as to adapt to the rapidly changing scenarios requiring socially conscious green computing solutions.

Program Educational Objectives (PEOs)

1. To produce the students employable towards building a successful career based on sound understanding of theoretical and applied aspects and methodology to solve multidisciplinary real life problems.
2. To produce professional graduates ready to work with a sense of responsibility, ethics and enabling them to work efficiently individually and also in team.
3. To inculcate competent students so that they are able to pursue higher studies and research in areas of engineering and other professionally related fields.
4. To inculcate ability to adapt to the changing technology through continual learning.

**PROGRAM SPECIFIC OUTCOMES (PSOs)**

PSO1. Ability to analyze, design, implement, and test software systems based on requirement specifications and development methodologies of software systems.

PSO2. Apply computer science theory blended with engineering mathematics to solve computational tasks and model real world problems using appropriate programming language, data structure, and algorithms.

PSO3. Ability to explore technological advancements in various domains, evaluate its merits and identify research gaps to provide solution to new ideas and innovations.

**Course Outcomes**

1. Students get practical knowledge on designing and analysis of conceptual model and mapping of conceptual model to relational database systems.
2. Design and implement SQL queries using DDL and DML concepts
3. Design and implement advance SQL queries such as relational constraints, joins, set operations, aggregate functions, trigger, views, and embedded SQL.
4. Application of transaction control language(TCL), data control language(DCL) in SQL in order to evaluate practical implications of DBA such as transaction, recovery and security.

**Rational behind DBMS Lab**

Database management has evolved from a specialized computer application to a central component of a modern com putting environment and as a result knowledge about database system has become an essential part of computer science. The course serves as a visual guide to the material presented during our lectures. The aim of this course is to provide an introduction to Database management system, with an emphasis on foundational material the fundamental concepts and algorithms covered are based on those used in existing commercial or experimental database systems. Our aim is to present these concepts and algorithms in general setting.

Software and Hardware Requirements

**Software Required:**

1. VB, ORACLE and/or DB2

2. VB, MSACCESS

3. ORACLE, D2K

4. VB, MS SQL SERVER 2000

**Hardware Required**

Processor: Pentium III

RAM: 128 MB

Hard Disk: 40 GB

**Practical 1**

**Objective: Create tables and specify the Questionnaires in SQL.**

Introduction about SQL-SQL (Structured Query Language) is a nonprocedural language, you specify what you want, not how to get it. A block structured format of English key words is used in this Query language. It has the following components.

**DDL (Data Definition Language)-**

The SQL DDL provides command for defining relation schemas, deleting relations and modifying relation schema.

**DML (DATA Manipulation Language)-**

It includes commands to insert tuples into, delete tuples from and modify tuples in the database.

**View definition-**

The SQL DDL includes commands for defining views.

Transaction Control- SQL includes for specifying the beginning and ending of transactions.

**Embedded SQL and Dynamic SQL-**

Embedded and Dynamic SQL define how SQL statements can be embedded with in general purpose programming languages, such as C, C++, JAVA, COBOL, Pascal and Fortran.

**Integrity-**

The SQL DDL includes commands for specifying integrity constraints that the data stored in the database must specify. Updates that violate integrity

Constraints are allowed.

**Authorization-**

The SQL DDL includes commands for specifying access rights to relations and views.

**Data Definition Language-**

The SQL DDL allows specification of not only a set of relations but also information about each relation, including-

* Schema for each relation
* The domain of values associated with each attribute.
* The integrity constraints.
* The set of indices to be maintained for each relation.
* The SQL DDL includes commands for defining views.
* The physical storage structure of each relation on disk.

**Domain types in SQL-**

The SQL standard supports a variety of built in domain types, including-

* Char (n)- A fixed length character length string with user specified length.
* Varchar (n)- A variable character length string with user specified maximum length n.
* Int- An integer.
* Small integer- A small integer.
* Numeric (p, d)-A Fixed point number with user defined precision.
* Real, double precision- Floating point and double precision floating point numbers with machine dependent precision.
* Float (n)- A floating point number, with precision of at least n digits.
* Date- A calendar date containing a (four digit) year, month and day of the month.
* Time- The time of day, in hours, minutes and seconds Eg. Time ’09:30:00’.
* Number- Number is used to store numbers (fixed or floating point).

**DDL statement for creating a table-**

**Syntax-**

Create table tablename (columnname datatype(size), columnname datatype(size));

**Syntax-**

CREATE TABLE TABLENAME

[(columnname, columnname, .........)]

AS SELECT columnname, columnname........FROM tablename;

**Insertion of data into tables-**

**Syntax-**

INSERT INTO tablename

[(columnname, columnname, .........)]

Values(expression, expression);

**Inserting data into a table from another table:**

**Syntax-**

INSERT INTO tablename

SELECT columnname, columnname, .......

FROM tablename;

**Insertion of selected data into a table from another table:**

**Syntax-**

INSERT INTO tablename

SELECT columnname, columnname........

FROM tablename

WHERE columnname= expression;

**Retrieving of data from the tables-**

**Syntax-**

SELECT \* FROM tablename;

**The retrieving of specific columns from a table-**

**Syntax-**

SELECT columnname, columnname, ....

FROM tablename;

**Elimination of duplicates from the select statement-**

**Syntax-**

SELECT DISTINCT columnname, columnname

FROM tablename;

**Selecting a data set from table data-**

**Syntax-**

SELECT columnname, columnname

FROM tablename

WHERE searchcondition;

**Assignment No:1**

Q. 1 Define the working of SQL commands of Data Definition Language, Data Manipulation

Language, Data Control Language and Transaction Control Language.

Q. 2 Write about the different data types used in SQL.

**Practical #2**

**Objective:- To Manipulate the Operations on the table.**

* DML ( Data Manipulation Language) Data manipulation is
* The retrieval of information stored in the database.
* The insertion of new information into the database.
* The deletion of information from the database.

The modification of information stored by the appropriate data model. There are basically two types.

**(i)Procedural DML:-** require a user to specify what data are needed and how to get those data.

**(ii)Non Procedural DML:** require a user to specify what data are needed without specifying how to get those data.

**Updating the content of a table:** In creation situation we may wish to change a value in table without changing all values in the tuple. For this purpose the update statement can be used.

**Update table name**

Set columnname = expression, columnname =expression......

Where columnname = expression.

**Deletion Operation: -**

A delete query is expressed in much the same way as Query. We can delete whole tuple (rows) we can delete values on only particulars attributes.

**Deletion of all rows**

**Syntax:**

Delete from tablename :

**Deletion of specified number of rows**

**Syntax:**

Delete from table name

Where search condition;

**Computation in expression lists used to select data**

+ Addition - Subtraction

\* multiplication \*\* exponentiation

/ Division () Enclosed opera

**Renaming columns used with Expression Lists:** - The default output column names can be renamed by the user if required

Syntax:

Select column name result\_columnname,

Columnname result\_columnname,

From table name.

**Logical Operators:**

The logical operators that can be used in SQL sentenced are

AND all must be included

OR any of may be included

NOT none of could be included

**Range Searching:**

Between operations is used for range searching.

**Pattern Searching:**

The most used operation on string is pattern matching using the operation ‘like’ we describe patterns by using two special characters.

* Percent (%) ; the % character matches any substring we consider the following examples.
* ‘Perry %’ matches any string beginning with perry
* ‘% idge % matches any string containing’ idge as substring.
* ‘ - - - ‘matches any string exactly three characters.
* ‘ - - - % matches any string of at least of three characters.

**Oracle functions:**

Functions are used to manipulate data items and return result. function follow the format of function \_name (argument1, argument2 ..) . An arrangement is user defined variable or constant. The structure of function is such that it accepts zero or more arguments.

**Examples:**

Avg return average value of n

**Syntax:**

Avg ([distinct/all]n)

Min return minimum value of expr.

**Syntax:**

MIN((distinct/all )expr)

Count Returns the no of rows where expr is not null

**Syntax:**

Count ([distinct/all)expr]

Count (\*) Returns the no rows in the table, including duplicates and those with nulls.

Max Return max value of expr

**Syntax:**

Max ([distinct/all]expr)

Sum Returns sum of values of n

**Syntax:**

Sum ([distinct/all]n)

**Sorting of data in table**

**Syntax:**

Select columnname, columnname From table Order by columnname;

**Assignment No: 2**

Q.1 Create a table employee having following columns:

firstname, lastname, title, age and salary

Q.2 Enter the data of eight employee into employee table.

Q.3 On the basis of employee table answer the following questions:

i) Select all columns for everyone in your emp table.

ii) Select first and last names for everybody that’s under 30 year old.

iii) Select all columns for everyone with a salary over 30000.

iv) Select firstname, lastname and salary for anyone with ‘Programmer’ in their title.

v) Select the firstname for everyone whose lastname contains ’a’.

vi) select all columns for everyone whose firstname starts with ’D’.

**Assignment No: 3**

Q. Following relations are given to you.

**EMPLOYEE**

|  |  |  |
| --- | --- | --- |
| MPLOYEE\_ID | NOT NULL | NUMBER(6) |
| FIRST\_NAME |  | VARCHAR2(20) |
| LAST\_NAME | NOT NULL | VARCHAR2(25) |
| EMAIL | NOT NULL | VARCHAR2(25) |
| PHONE\_NUMBER |  | VARCHAR2(20) |
| HIRE\_DATE | NOT NULL | DATE |
| JOB\_ID | NOT NULL | VARCHAR2(10) |
| SALARY |  | NUMBER(8,2) |
| COMMISSION\_PCT |  | NUMBER(2,2) |
| MANAGER\_ID |  | NUMBER(6) |
| DEPARTMENT\_ID |  | NUMBER(4) |

**EMP\_DETAILS\_VIEW:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Null?** | **Type** |
| EMPLOYEE\_ID | NOT NULL | NUMBER(6) |
| JOB\_ID | NOT NULL | VARCHAR2(10) |
| MANAGER\_ID |  | NUMBER(6) |
| DEPARTMENT\_ID |  | NUMBER(4) |
| LOCATION\_ID |  | NUMBER(4) |
| COUNTRY\_ID |  | CHAR(2) |
| FIRST\_NAME |  | VARCHAR2(20) |
| LAST\_NAME | NOT NULL | VARCHAR2(25) |
| SALARY |  | NUMBER(8,2) |
| COMMISSION\_PCT |  | NUMBER(2,2) |
| DEPARTMENT\_NAME | NOT NULL | VARCHAR2(30) |
| JOB\_TITLE | NOT NULL | VARCHAR2(35) |
| CITY | NOT NULL | VARCHAR2(30) |
| STATE\_PROVINCE |  | VARCHAR2(25) |
| COUNTRY\_NAME |  | VARCHAR2(40) |
| REGION\_NAME |  | VARCHAR2(25) |

DEPARTMENTS:

|  |  |  |
| --- | --- | --- |
| **Name** | **Null?** | **Type** |
| DEPARTMENT\_ID | NOT NULL | NUMBER(4) |
| DEPARTMENT\_NAME | NOT NULL | VARCHAR2(30) |
| MANAGER\_ID |  | NUMBER(6) |
| LOCATION\_ID |  | NUMBER(4) |

1>select all record of employees.

2> select all records of employees where salary is >10000

3>select first\_name, last\_name and title of employees having salary between 25000 >=salary >=10000

4>select first\_name, emp\_id , phone no of records of employees having department id=50

5>select all records of employees in the increasing order of salary.

6>select all records of employees in the decreasing order of salary.

7>select first\_name of employees in the increasing order of salary and if salary matches then decreasing order of the department\_id.

8>Retrieve first\_name and depart\_id and the salary of all the puchase managers and display them with modified salary(5% increase in salaryrename the coloumn by new\_salary) for there good work.

9>Retrieve first\_name of all programmers of IT department whose first name starts with a and ends with b.

10> Retrieve first\_name of all programmers of IT department whose first name contains a substring ‘ish’

11>Retrieve all the records except David, Peter,Oliver,Allan.

12>Display all the records of employees in CSV (comma separated file) form under the heading Employee\_ Details.

13> There are four coding errors in this statement. Can you identify them?SELECT    employee\_id, last\_name sal x 12  ANNUAL SALARY

14) The following SELECT statement executes successfully:           (True / False)

     SELECT last\_name, job\_id, salary AS Sal FROM   employees;

15) The following SELECT statement executes successfully:           (True / False)

SELECT \* FROM   job\_grades;

16) There are four coding errors in this statement. Can you identify them?

SELECT    employee\_id, last\_name sal x 12  ANNUAL SALARY FROM   employees;

17) Show the structure of the DEPARTMENTS table. Select all data from the table.

18) Show the structure of the EMPLOYEES table.

Create a query to display the last name, job code, hire date, and employee number for each employee, with employee number appearing first.

Provide an alias STARTDATE for the HIRE\_DATE column. Save your SQL statement to a file named Assignment11.sql.

19) Create a query to display unique job codes from the EMPLOYEES table.

19) Display the last name concatenated with the job ID, separated by a comma and space, and name the column Employee and Title.

20) Create a query to display the last name and salary of employees earning more than $12,000.

21) Create a query to display the employee last name and department number for employee number 176.

22) Display the employee last name, job ID, and start date of employees hired between February 20, 1998, and May 1, 1998. Order the query in ascending order by start date.

23) Display the last name and department number of all employees in departments 20 and 50 in alphabetical order by name.

24) Modify Assignment13.sql to list the last name and salary of employees who earn between $5,000 and $12,000, and are in department 20 or 50.

     Label the columns Employee and Monthly Salary, respectively.

25) Display the last name and hire date of every employee who was hired in 1994.

26) Display the last name and job title of all employees who do not have a manager.

27) Display the last name, salary, and commission for all employees who earn commissions. Sort data in descending order of salary and commissions.

28) Display the last names of all employees where the third letter of the name is an *a.*

29) Display the last name of all employees who have an *a* and an *e* in their last name.

30) Display the last name, job, and salary for all employees whose job is sales representative or stock clerk and whose salary is not equal to $2,500, $3,500, or $7,000.

31) Write a query to display the current date. Label the column Date.

32) For each employee, display the employee number, last\_name, salary, and salary increased by 15% and expressed as a whole number. Label the column New Salary. Place your SQL statement in a text file named lab3\_2.sql.

33) Write a query that displays the employee’s last names with the first letter capitalized and all other letters lowercase, and the length of the names, for all employees whose name starts with J, A, or M. Give each column an appropriate label. Sort the results by the employees’ last names.

34)  Create a query that displays the employees’ last names and commission amounts. If an employee does not earn commission, put “No Commission.

    ” Label the column COMM.

35)List the location id, hire date, job id of all the managers

**Assignment No: 4**

**Q1. Create the following tables:**

1. **client\_master**

columnname datatype size

client\_no varchar2 6

name varchar2 20

city varchar2 15

state varchar2 15

pincode number 6

bal\_due number 10,2

1. **Product\_master**

Columnname datatype size

Product\_no varchar2 10

Description varchar2 10

Profit\_percent number 10

Unit\_measure varchar2 5

Qty\_on\_hand number 5

Reoder\_lvl number 5

Sell\_price number 6, 2

Cost\_price number 6, 2

**Q2- Insert the following data into their respective tables:**

**Data for Clint Master:**

Clientno Name city pincode state bal.due

0001 Ivan Bombay 400054 Maharashtra 15000

0002 Vandana Madras 780001 Tamilnadu 0

0003 Pramada Bombay 400057 Maharashtra 5000

0004 Basu Bombay 400056 Maharashtra 0

0005 Ravi Delhi 100001 2000

0006 Amit Bombay 400050 Maharashtra 0

**Data for Product Master:**

Product No. Description Profit % Unit Qty Reorder Sell Cost Percent measured on hand lvl price price

P00001 1.44floppies 5 piece 100 20 525 500

P03453 Monitors 6 piece 10 3 12000 11200

P06734 Mouse 5 piece 20 5 1050 500

P07865 1.22 floppies 6 piece 100 20 525 500

P07868 Keyboards 2 piece 10 3 3150 3050

P07885 CD Drive 2.5 piece 10 3 5250 5100

P07965 540 HDD 4 piece 10 3 8400 8000

P07975 1.44 Drive 3 piece 10 3 1050 1000

P08865 1.22 Drive 5 piece 2 3 1050 1000

**Q3:- On the basis of above two tables answer the following Questionnaires:**

1. Find out the names of all the clients.
2. Retrieve the list of names and cities of all the clients.
3. Display the information for client no 0001 and 0002.
4. Find all the products whose sell price is greater then 5000.

**Practical 2**

**Objective:- To Implement the restrictions on the table.**

**Data constraints:**

Besides the cell name, cell length and cell data type there are other parameters i.e. other data constrains thatcan be passed to the DBA at check creation time. The constraints can either be placed at column level or at the table level.

**i.Column Level Constraints:** If the constraints are defined along with the column definition, it is called a column level constraint.

**ii.Table Level Constraints:** If the data constraint attached to a specify cell in a table reference the contents of another cell in the table then the user will have to use table level constraints.

**Null Value Concepts:-** while creating tables if a row locks a data value for particular column that value is said tobe null . Column of any data types may contain null values unless the column was defined as not null when the table was created

**Syntax:**

Create table tablename

(

columnname data type (size) not null ......)

**Primary Key:** primary key is one or more columns is a table used to uniquely identity each row in the table. Primary key values must not be null and must be unique across the column. A multicolumn primary key is called composite primary key.

**Syntax: primary key as a column constraint**

Create table tablename

(columnname datatype (size) primary key,....)

**Primary key as a table constraint**

Create table tablename

(columnname datatype (size), columnname datatype( size)...

Primary key (columnname,columnname));

**Default value concept**: At the line of cell creation a default value can be assigned to it. When the user is loading a record with values and leaves this cell empty, the DBA will automatically load this cell with the default value specified. The data type of the default value should match the data type of the column

**Syntax:**

Create table tablename

(columnname datatype (size) default value,....);

**Foreign Key Concept:** Foreign key represents relationship between tables. A foreign key is column whose values are derived from the primary key of the same of some other table. The existence of foreign key implies that the table with foreign key is related to the primary key table from which the foreign key is derived .A foreign key must have corresponding primary key value in the primary key table to have meaning.

**Foreign key as a column constraint**

**Syntax :**

Create table table name

(columnname datatype (size) re

ferences another table name);

**Foreign key as a table constraint:**

**Syntax :**

Create table name

(columnname datatype (size)....

primary key (columnname);

foreign key (columnname)references table name);

**Check Integrity Constraints:**

Use the check constraints when you need to enforce integrity rules that can be evaluated based on a logical expression following are a few examples of appropriate check constraints.

* A check constraints name column of the client\_master so that the name is entered in upper case.
* A check constraint on the client\_no column of the client \_master so that no client\_no value starts with ‘c’

Syntax:

Create table tablename

(columnname datatype (size) CONSTRAINT constraintname)

Check (expression));

**Practical 3**

**Objective: - To Implement the structure of the table**

Modifying the Structure of Tables- Alter table command is used to changing the structure of a table. Using the alter table clause you cannot perform the following tasks:

1. change the name of table
2. change the name of column
3. drop a column
4. decrease the size of a table if table data exists.

**The following tasks you can perform through alter table command**.

**(i)Adding new columns:**

**Syntax**

ALTER TABLE tablename

ADD (newcolumnname newdatatype (size));

**(ii)Modifying existing table**

**Syntax:**

ALTER TABLE tablename

MODIFY (newcolumnname newdatatype (size));

*NOTE:Oracle not allow constraints defined using the alter table, if the data in the table,*

*Violates such constraints*.

**Removing/Deleting Tables-** Following command is used for removing or deleting a table.

**Syntax:**

DROP TABLE tablename:

Defining Integrity constraints in the ALTER TABLE command- You can also define integrity constraints using the constraint clause in the ALTER TABLE command. The following examples show the definitions of several integrity constraints.

**(1)Add PRIMARY KEY-**

**Syntax:**

ALTER TABLE tablename

ADD PRIMARY KEY (columnname);

**(2)Add FOREIGN KEY-**

**Syntax:**

ALTER TABLE tablename

ADD CONSTRAINT constraintname

FOREIGN KEY(columnname) REFERENCES tablename;

**Dropping integrity constraints in the ALTER TABLE command:**

You can drop an integrity constraint if the rule that if enforces is no longer true or if the constraint is no longer needed. Drop the constraint using the ALTER TABLE command with the DROP clause. The following examples illustrate the dropping of integrity constraints.

**(1)DROP the PRIMARY KEY-**

**Syntax:**

ALTER TABLE tablename

DROP PRIMARY KEY

**(2)DROP FOREIGN KEY-**

**Syntax**:

ALTER TABLE tablename

DROP CONSTRAINT constraintname;

**Assignment No: 5**

**Q.1 Create the following tables:**

**Sales\_master**

**Columnname Datatype Size Attributes**

Salesman\_no varchar2 6 Primary key/first letter must start with ‘s’

Sal\_name varchar2 20 Not null

Address varchar2 Not null

City varchar2 20

State varchar2 20

Pincode Number 6

Sal\_amt Number 8,2 Not null, cannot be 0

Tgt\_to\_get Number 6,2 Not null, cannot be 0

Ytd\_sales Number 6,2 Not null, cannot be 0

Remarks Varchar2 30

**Sales\_order**

|  |  |  |  |
| --- | --- | --- | --- |
| **Columnname** | **Datatype** | **Size** | **Attributes** |
| S\_order\_no | varchar2 | 6 | Primary key |
| S\_order\_date | Date | 6 | Primary key reference clientno of client\_master table |
| Client\_no | Varchar2 | 25 |  |
| Dely\_add | Varchar2 | 6 |  |
| Salesman\_no | Varchar2 | 6 | Foreign key references salesman\_no of salesman\_master table |
| Dely\_type | Char | 1 | Delivery part(p)/full(f),default f |
| Billed\_yn | Char | 1 |  |
| Dely\_date | Date |  | Can not be lessthan s\_order\_date |
| Order\_status | Varchar2 | 10 | Values (‘in process’;’fulfilled’;back order’;’canceled |

**Sales\_order\_details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Datatype** | **Size** | **Attributes** |
| S\_order\_no | Varchar2 | 6 | Primary key/foreign key references s\_order\_no of sales\_order |
| Product\_no | Varchar2 | 6 | Primary key/foreign key references product\_no of product\_master |
| Qty\_order | Number | 8 |  |
| Qty\_disp | Number | 8 |  |
| Product\_rate | Number | 10,2 |  |

**Insert the following data into their respective tables using insert statement:**

**Data for sales\_man master table**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Salesman\_no** | **Salesman name** | **Address** | **City** | **Pin code** | **State** | **Salamt** | **Tgt\_to\_get** | **Ytd**  **Sales** | **Remark** |
| 500001 | Kiran | A/14 worli | Bombay | 400002 | Mah | 3000 | 100 | 50 | Good |
| 500002 | Manish | 65,nariman | Bombay | 400001 | Mah | 3000 | 200 | 100 | Good |
| 500003 | Ravi | P-7 Bandra | Bombay | 400032 | Mah | 3000 | 200 | 100 | Good |
| 500004 | Ashish | A/5 Juhu | Bombay | 400044 | Mah | 3500 | 200 | 150 | Good |

**Data for salesorder table:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| S\_orderno | S\_orderdate | Client no | Dely type | Bill yn | Salesman no | Delay date | Orderstatus |
| 019001 | 12-jan-96 | 0001 | F | N | 50001 | 20-jan-96 | Ip |
| 019002 | 25-jan-96 | 0002 | P | N | 50002 | 27-jan-96 | C |
| 016865 | 18-feb-96 | 0003 | F | Y | 500003 | 20-feb-96 | F |
| 019003 | 03-apr-96 | 0001 | F | Y | 500001 | 07-apr-96 | F |
| 046866 | 20-may-96 | 0004 | P | N | 500002 | 22-may-96 | C |
| 010008 | 24-may-96 | 0005 | F | N | 500004 | 26-may-96 | Ip |

**Data for sales\_order\_details table:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S\_order no | Product no | Qty ordered | Qty disp | Product\_rate |
| 019001 | P00001 | 4 | 4 | 525 |
| 019001 | P07965 | 2 | 1 | 8400 |
| 019001 | P07885 | 2 | 1 | 5250 |
| 019002 | P00001 | 10 | 0 | 525 |
| 046865 | P07868 | 3 | 3 | 3150 |
| 046865 | P07885 | 10 | 10 | 5250 |
| 019003 | P00001 | 4 | 4 | 1050 |
| 019003 | P03453 | 2 | 2 | 1050 |
| 046866 | P06734 | 1 | 1 | 12000 |
| 046866 | P07965 | 1 | 0 | 8400 |
| 010008 | P07975 | 1 | 0 | 1050 |
| 010008 | P00001 | 10 | 5 | 525 |

**Assignment 6**

**Q.1. Create the following tables:**

**Table Name :** **Challan\_Header**

**Column name data type size Attributes**

Challan\_no varchar2 6 Primary key

s\_order\_no varchar2 6 Foreign key references s\_order\_no of sales\_order table

challan\_date date not null

billed\_yn char 1 values (‘Y’,’N’). Default ‘N’

#### Table Name : Challan\_Details

**Column name data type size Attributes**

Challan\_no varchar2 6 Primary key/Foreign key references

Product\_no of product\_master

Qty\_disp number 4,2 not null

**Q2. Insert the following values into the challan header and challan\_details tables:**

(i) **Challan No S\_order No Challan Date Billed\_yn**

CH9001 019001 12-DEC-95 Y

CH865 046865 12-NOV-95 Y

CH3965 010008 12-OCT-95 Y

#### Data for challan\_details table

**Challan No Product No Qty Disp**

CH9001 P00001 4

CH9001 P07965 1

CH9001 P07885 1

CH6865 P07868 3

CH6865 P03453 4

CH6865 P00001 10

CH3965 P00001 5

CH3965 P07975 2

**Q.3. Using the table of assignment 4&5 answer the following Questionnaires.**

1. Select product\_no, total qty\_ordered for each product.
2. Select product\_no, product description and qty ordered for each product.
3. Find the order No, Client No and salesman No. where a client has been received by more

than one salesman.

**Practical 3**

**Objective: - To implement the concept of Joins**

Joint Multiple Table (Equi Join): Sometimes we require to treat more than one table as though manipulate data from all the tables as though the tables were not separate object but one single entity. To achieve this, we must join tables. Tables are joined on column that have dame data type and data within tables.

The tables that must be joined are specified in the FROM clause and the joining attributes in the WHERE clause.

**Algorithm for JOIN in SQL:**

1. Cartesian product of tables (specified in the FROM clause)
2. Selection of rows that match (predicate in the WHERE clause)
3. Project column specified in the SELECT clause.

**1. Cartesian product: -**

Consider two table student and course

Select B.\*,P.\*

FROM student B, course P;

**2. INNER JOIN:**

Cartesian product followed by selection

Select B.\*,P.\*

FROM student B, Course P

WHERE B.course # P.course # ;

**3. LEFT OUTER JOIN:**

LEFT OUTER JOIN = Cartesian product + selection but include rows from the left table which are unmatched pat nulls in the values of attributes belonging to the second table

Exam:

Select B.\*,P\*

FROM student B left join course p

ON B.course # P.course #;

1. **RIGHT OUTER JOIN: RIGHT OUTER JOIN** = Cartesian product + selection but include rows from right table which are unmatched

Exam:

Select B.\*,P.\*

From student B RIGHT JOIN course P

B.course# = P course # ;

**5. FULL OUTER JOIN**

Exam

Select B.\*,P.\*

From student B FULL JOIN course P

On B.course # = P course # ;

**ASSIGNMENT 7**

**OBJECTIVE:Answer the following Queries:**

1.Find out the product which has been sold to ‘Ivan Sayross.’

2.Find out the product and their quantities that will have do delivered.

3.Find the product\_no and description of moving products.

4.Find out the names of clients who have purchased ‘CD DRIVE’

5.List the product\_no and s\_order\_no of customers haaving qty ordered less than 5 from the order details table for the product “1.44 floppies”.

6.Find the products and their quantities for the orders placed by ‘Vandan Saitwal ’ and “Ivan Bayross”.

7.Find the products and their quantities for the orders placed by client\_no “

C00001” and “C00002”

8.Find the order No,, Client No and salesman No. where a client has been received by more than one salesman.

9.Display the s\_order\_date in the format “dd-mm-yy” e.g. “12- feb-96”

**Assignment 8**

**Consider the following relations:**

Student (snum: integer, sname: string, major: string, level: string,age: integer)

Class (name: string, meets at: string, room: string, d: integer)

Enrolled (snum: integer, cname: string)

Faculty (fid: integer, fname: string, deptid: integer)

**The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class. Level is a two character code with 4 different values (example: Junior: JR etc)**

Write the following queries in SQL. No duplicates should be printed in any of the answers.

i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by Prof. Harshith

ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

iii. Find the names of all students who are enrolled in two classes that meet at the same time.

iv. Find the names of faculty members who teach in every room in which some class is taught.

v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

Practical 4

**Objective:- To implement the concept of grouping of Data.**

**Grouping Data from Tables:**

There are circumstances where we would like to apply the aggregate function not only to a single set of tuples, but also to a group of sets of tuples, we specify this wish in SQL using the group by clause. The attribute or attributes given in the group by clause are used to form group. Tuples with the same value on all attributes in the group by clause are placed in one group.

**Syntax:**

SELECT columnname, columnname

FROM tablename

GROUP BY columnname;

At times it is useful to state a condition that applies to groups rather than to tuples. For example we might be interested in only those branches where the average account balance is more than 1200. This condition does not apply to a single tuple, rather it applies to each group constructed by the GROUP BY clause. To express such Questioner, we use the having clause of SQL.SQL applies predicates in the having may be used.

Syntax:

SELECT columnname, columnname

FROM tablename

GROUP BY columnname;

HAVING searchcondition;

**Assignment 9**

**Objective-Answer the following Queries:**

Q1.- Print the description and total quantity sold for each product.

Q2.- Find the value of each product sold.

Q3.- Calculate the average quantity sold for each client that has a maximum order value of 15000.

Q4.- Find out the products which has been sold to Ivan.

Q5.- Find the names of clients who have ‘CD Drive’.

Q6.- Find the products and their quantities for the orders placed by ‘Vandana’ and ‘Ivan’.

Q7.- Select product\_no, total qty\_ordered for each product.

Q8.- Select product\_no, product description and qty ordered for each product.

Q9.- Display the order number and day on which clients placed their order.

Q10.- Display the month and Date when the order must be delivered.

**Practical 5**

**Objective:- To implement the concept of Sub Queries.**

**Sub Queries:-** A sub Query is a form of an SQL statement that appears inside another SQL statement. It also termed as nested Query. The statement containing a subQuery called a parent statement. The rows returned by the sub Query are used by the following statement.

It can be used by the following commands:

1.To insert records in the target table.

2.To create tables and insert records in this table.

3.To update records in the target table.

4.To create view.

5.To provide values for the condition in the WHERE , HAVING IN ,

SELECT,UPDATE, and DELETE statements.

Exam:-

Creating clientmaster table from oldclient\_master, table

Create table client\_master

AS SELECT \* FROM oldclient\_master;

**Using the Union, Intersect and Minus Clause:**

**Union Clause:**

The user can put together multiple Queries and combine their output using the union clause . The union clause merges the output of two or more Queries into a single set of rows and column. The final output of union clause will be

Output: = Records only in Query one + records only in Query two + A single set of records with is common in the both Queries.

Syntax:

SELECT columnname, columname FROM tablename 1

UNION

SELECT columnname, columnname From tablename2;

**Intersect Clause:**

The use can put together multiple Queries and their output using the interest clause. The final output of the interest clause will be : Output =A single set of records which are common in both Queries

**Syntax:**

SELECT columnname, columnname FROM tablename 1

INTERSECT

SELECT columnname, columnname FROM tablename 2;

**MINUS CLAUSE:-** The user can put together multiple Queries and combine their output

= records only in Query one

**Syntax:**

SELECT columnname, columnname FROM tablename ;

MINUS

SELECT columnname, columnname FROM tablename ;

**Assignment 10**

**Objective: Answer the following Queries:**

Question.

1.Find the product\_no and description of non- moving products.

2.Find the customer name, address, city and pincode for the client who has placed order no “019001”

3.Find the client names who have placedorder before the month of may 96.

4.Find out if product “1.44 Drive” is ordered by only client and print the client\_no name to whom it was sold.

5.find the names of client who have placed orders worth Rs.10000 or more.

6.Select the orders placed by ‘Rahul Desai”

7.Select the names of persons who are in Mr. Pradeep’s department and who have also worked on an inventory control system.

8.Select all the clients and the salesman in the city of Bombay.

9.Select salesman name in “Bombay” who has atleast one client located at “Bombay”

10.Select the product\_no, description, qty\_on-hand,cost\_price of non\_moving items in the product\_master table.

**Practical 6**

**Objective:- To implement the concept of Indexes and views. Indexes-**

An index is an ordered list of content ofa column or group of columns in a table. An index created on the single column of the table is called simple index. When multiple table columns are included in the index it is called composite index.

**Creating an Index for a table:-**

Syntax(Simple)

CREATE INDEX index\_name

ON tablename(column name);

**Composite Index:-**

CREATE INDEX index\_name

ON tablename(columnname,columnname);

**Creating an UniQuestion Index: -**

CREATE UNIQUESTION INDEX indexfilename

ON tablename(columnname);

**Dropping Indexes: -**

An index can be dropped by using DROP INDEX

SYNTAX: -

DROP INDEX indexfilename;

**Views: -** Logical data is how we want to see the current data in our database. Physical data is how this data is actually placed in our database. Views are masks placed upon tables. This allows the programmer to develop a method via which we can display predetermined data to users according to our desire. Views may be created fore the following reasons:

1.The DBA stores the views as a definition only. Hence there is no duplication of data.

2.Simplifies Queries.

3.Can be Queried as a base table itself.

4.Provides data security.

5.Avoids data redundancy.

**Creation of Views:-**

**Syntax:-**

CREATE VIEW view name AS

SELECT columnname, columnname

FROM tablename

WHERE columnname=expression\_list;

**Renaming the columns of a view:-**

**Syntax:-**

CREATE VIEW viewname AS

SELECT newcolumnname....

FROM tablename

WHERE columnname=expression\_list;

**Selecting a data set from a view-**

**Syntax:-**

SELECT columnname, columnname

FROM viewname

WHERE search condition;

**Destroying a view-**

**Syntax:-**

DROP VIEW viewname;

**Assignment 10**

**Objective : Answer the following Questions**

Q1. Create an index on the table client\_master, field client\_no.

Q2. Create an index on the sales\_order, field s\_order\_no.

Q3. Create an composite index on the sales\_order\_details table for the columns s\_order\_no and product\_no.

Q4. Create an composite index ch\_index on challan\_header table for the columns challan no and s\_order\_no.

Q5. Create an uniQuestion index on the table salesman\_master, field salesman\_no.

Q6. Drop index ch\_index ontable challan\_header.

Q7. Create view on salesman\_master whose sal\_amt is less than 3500.

Q8. Create a view client\_view on client\_master and rename the columns as name, add1, add2, city, pcode,state respectively.

Q9. Select the client names from client\_view who lives in city ‘Bombay’.

Q10. Drop the view client\_view.

**Practical 7**

**Objective: To create triggers for various events such as insertion, updation, etc.**

TRIGGER: A Trigger is a stored procedure that defines an action that the database automatically take when some database-related event such as Insert, Update or Delete occur.

**TYPES OF TRIGGERS**

The various types of triggers are as follows,

•Before: It fires the trigger before executing the trigger statement.

•After: It fires the trigger after executing the trigger statement.

•For each row: It specifies that the trigger fires once per row.

•For each statement: This is the default trigger that is invoked. It specifies that the trigger fires once per statement.

**VARIABLES USED IN TRIGGERS**

•:new

•:old

These two variables retain the new and old values of the column updated in the database. The values in these variables can be used in the database triggers for data manipulation

**Row Level Trigger vs. Statement Level Trigger:**

Row Level Trigger

* These are fired for each row affected by the DML statement.
* These are used for generating/checking the values begin inserted or updated.

**Statement Level Trigger**

* These are fired once for the statement instead of the no of rows modified by it.
* These are used for generated the summary information.

**Before trigger vs. after trigger**

**Before Triggers**

Before triggers are fired before the DML statement is actually executed.

**After Triggers**

After triggers are fired after the DML statement has finished execution.

Sytax:

Create or replace trigger <trg\_name> Before /After Insert/ Update/ Delete

[of column\_name, column\_name....]

on <table\_name>

[for each row]

[when condition]

begin

---statement

end;

**Assignment 11**

Q1: Create a trigger that insert current user into a username column of an existing table

**Procedure for doing the experiment:**

1Createa table itstudent4 with name and username as arguments

2Create a trigger for each row that insert the current user as username into a table

3Execute the trigger by inserting value into the table

Q2: Create a Simple Trigger that does not allow Insert Update and Delete Operations on the Table

Program: Table used:

SQL> select \* from itempls.

ENAME EID SALARY

---------------------------------------------------

xxx 11 10000

yyy 12 10500

zzz 13 15500