INFORMATION TECHNOLOGY DEPARTMENT

DBMS LAB MANUAL

1. Introduction to DBMS Laboratory

DBMS: A database management system (DBMS) is system software for creating and managing databases. DBMS provides users and programmers with a systematic way to make it possible for end users to create, read, update and delete data in a database. It essentially serves as an interface between the database and end users or application programs, ensuring that data is consistently organized and remains easily accessible.

RDBMS: A relational database management system (RDBMS) is a program that let create, update, and administer a relational database. Most commercial RDBMS's use the Structured Query Language (SQL) to access the database. The leading RDBMS products are Oracle, IBM's DB2 and Microsoft's SQL Server.

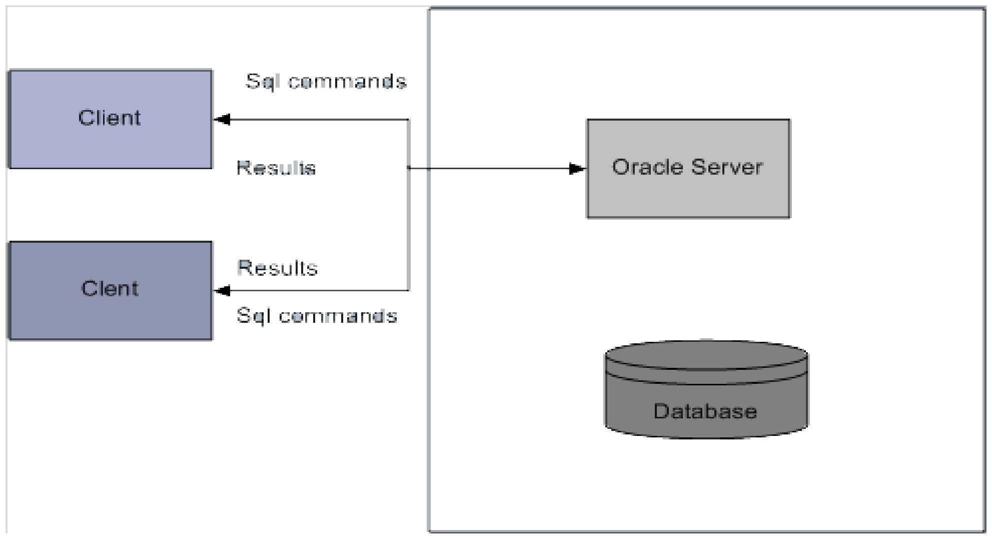
Laboratory Objective

Upon successful completion of this Lab the student will be able to:

- Create, modify and retrieve database objects from database server.
- Apply the basic concepts of conditions, joins and sub query to retrieve data.
- Demonstrate the usage of Triggers and stored procedures.
- Design and Develop forms on some scenarios.
- Illustrate data validation by using PL/SQL.
- Apply security related features like locking mechanisms and creating password to protect the database.
- Illustrate several requirement and operations that the analyst needed to analyze, design, and implement the systems.
- Performing database operations (create, update, modify, retrieve, etc.,) using front- end tools like D2K.
- Design and Develop forms and reports for student Information, library information, Pay roll etc.

Overview of Oracle

Oracle consists of a comprehensive set of application building and end-user products. Oracle applications are accessible on a wide range of platforms and operating system.Oracle provides a flexible Database Management system. Oracle application may run on the same computers as the oracle server. Alternatively application and the tools supporting them may run on system local to the user (the client system), the oracle DBMS runs on another (server system).



Oracle Server

Oracle is the name of a database management system developed by Oracle Corporation. Oracle server manages data in the database. Users access oracle server using SQL commands. So Oracle Server receives SQL commands from users and executes them on the database.

The oracle server comprises a DBMS that controls

- Storage of data in designated database area.
- Retrieval of data for applications, using approximate optimization technique.
- Database security and the tasks permitted for specific users.
- Consistency and protection of data, including archiving task and locking mechanisms.
- Communication and integrity of data when databases are distributed across a network.

Features of Oracle

- Large database support
- Data concurrence
- Industry accepted standards
- Portability

- Enforced Integrity
- Data security
- Support for client/Server environment

SQL

SQL is a powerful query language. The greatest strength of SQL is that it is a standard that most vendors of DBMS software support. Pronounce SQL as sequel. It's the interface between the Kernel and ORACLE utilities and application development tools. SQL is a language that operates on entire groups of data and treats any quantity of information extracted as a single unit. SQL consists of a small number of high level commands that left you query a database to extract and update information stored in a database. The SQL standard is defined by ANSI to which all competing SQL products are supposed to confirm.

SQL is usually provided in two modes – interactive and embedded SQL.

Interactive SQL is used to operate directly on a database. The response to any SQL command can be seen almost immediately on the same terminal.

Embedded SQL consists of SQL commands used within programs written in some other language like C, ASP, JSP, VB or Pascal.

SQL commands are categorized into the following language statement

- **DDL is Data Definition Language statements. Some examples:**

  CREATE - to create objects in the database.

  ALTER - alters the structure of the database.

  DROP - delete objects from the database.

  TRUNCATE - remove all records from a table, including all spaces allocated for the records.

- **TCL is Transaction control language**

  GRANT - gives user's access privileges to database

REVOKE - withdraw access privileges given with the GRANT command

- **DML is Data Manipulation Language statements. Some examples:**

  SELECT - retrieve data from the a database.

  INSERT - insert data into a table.

  UPDATE - updates existing data within a table.

  DELETE - deletes all records from a table, the space for the records remain

  LOCK TABLE - control concurrency.

- **DCL is Data Control Language statements. Some examples:**

  COMMIT - save the changes permanently.

  SAVEPOINT - identify a point in a transaction to which you can later roll back

  ROLLBACK - restore database to original since the last COMMIT

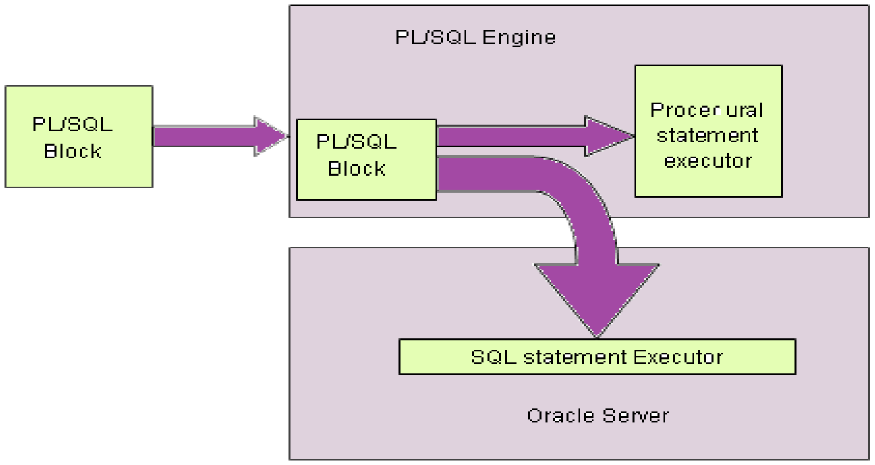
  SET TRANSACTION - Change transaction options like what rollback segment to use.

PL/SQL

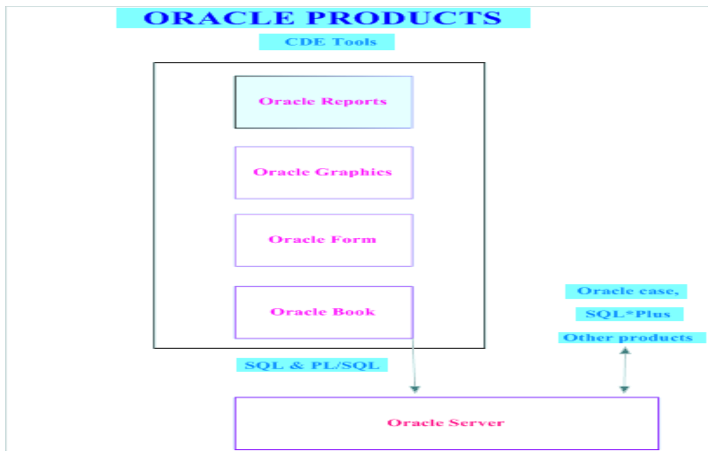
PL/SQL stands for procedural language extension to SQL. PL/SQL allows mixing SQL statements with procedural constructs like IF, looping etc. PL/SQL provides the capability to define and execute PL/SQL blocks such as procedures, functions, triggers and packages. PL/SQL is available only as an "enabling technology" within other software products; it does not exist as a standalone language. You can use PL/SQL in the Oracle relational database, in the Oracle Server, and in client-side application development tools, such as Oracle Forms. Here are some of the ways to use PL/SQL:

- To build stored procedures.
- To create database triggers.
- To implement client-side logic in your Oracle Forms application.
- To link a World Wide Web home page to an Oracle database.

PL/SQL is integrated tightly to the SQL language, yet it adds programming constructs which are not native to this standard relational database language. The programs of Pl/SQL looks like traditional 3GL modules, but can be used to include calls to SQL statements, manipulate data through cursors, and take advantage of some of the newest developments in programming languages.PL/SQL supports packages, which allow you to perform object-oriented design. PL/SQL provides a powerful mechanism for trapping and, with exception handlers, resolving errors.



Oracle Products



Other Commands

1. Data Types used in SQL

Each column contains the datatype and size of the column. The list of available data types in Oracle is shown in table below

Data type	**Description**
VARCHAR2(LEN)	Variable length char. Values up to width length. Maximum width is 4000 chars.
CHAR(LEN)	Fixed length char. Values of width length. Default width is 1. Maximum length is 200.
NUMBER	Floating point with precision of 38 significant digits.
NUMBER (Precision, Scale)	Precision represents Maximum significant digits allowed, which may not exceed 38. Scale is the number of decimal places on the right of the decimal point.
DATE	Date values in the range 1-1-4712 B.C to 31-12-4712 AD.
LONG	Variable length char. Values up to 2 GB. Only one LONG col. is allowed per table. You cannot use LONG datatype in functions, WHERE clause and sub queries.
RAW AND LONG RAW	Equivalent to VARCHAR2 and LONG respectively, used for storing digital sound or graphics images.
CLOB, BLOB, NCLOB	Used to store large char. And binary objects. Each can accommodate upto 4 GB.
BFILE	Stores a pointer to an external file.
TIMESTAMP	It is an extension of Date datatype. It stores year, month, day, hour, minute and second values

2. Integrity Constraints

Constraints are used to implement standard rules such as unique, and primary key etc., constraints are normally defined at the time of creating table. But it is also possible to define after the table is created using ALTER TABLE command. Constraints are stored in the Data Dictionary.

Constraints are classified into two types.

- Table constraints
- Column constraints

Syntax Column_Constraint: [constraint name]{[NOT] NULL| {UNIQUE | PRIMARY KEY}| REFERENCEStable [(column)]| CHECK (condition)}

Ex:SQL>create table subjects(subdescvarchar(20)constraint const_nameNOT NULL..);
Ex2:SQL>create table subjects(subdesc varchar(10)constraint subject_pk PRIMARY KEY);
Syntax Table_Constraint: [constraint name]{{UNIUQE | PRIMARY KEY} (Col, [, col]…)| FOREIGN KEY (Col, [, Col}…) REFERENCES table [(Col {, Col]…)| CHECK (condition)}

Ex1: syntax for table constraint
SQL>create table studmarks (Sno Number(5),constraint studmarks_fk FOREIGN KEY(Sno) references students (Sno));
Ex2: SQL>Create table studmarks (…,constraint studmarks_chk Check(stdate<=enddate));

Integrity Constraints Description

An integrity constraint is a declarative method of defining a rule for a column of a table. Oracle supports the following integrity constraints:

- NOTNULL constraints for the rules associated with nulls in a column
- UNIQUE key constraints for the rule associated with unique column values

- PRIMARYKEY constraints for the rule associated with primary identification values
- FOREIGNKEY constraints for the rules associated with referential integrity. Oracle supports the use of FOREIGNKEY integrity constraints to define the referential integrity actions, including:
    - Update and delete No Action
    - Delete CASCADE
    - Delete SETNULL
- CHECK constraints for complex integrity rules

Referential Integrity Constraints

Different tables in a relational database can be related by common columns, and the rules that govern the relationship of the columns must be maintained. Referential integrity rules guarantee that these relationships are preserved.

The following terms are associated with referential integrity constraints.

Term	**Definition**
Foreign key	The column or set of columns included in the definition of the referential integrity constraint that reference a referenced key.
Referenced key	The unique key or primary key of the same or different table that is referenced by a foreign key.
Dependent or child table	The table that includes the foreign key. Therefore, it is the table that is dependent on the values present in the referenced unique or primary key.
Referenced or parent table	The table that is referenced by the child table's foreign key. It is this table's referenced key that determines whether specific inserts or updates are allowed in the child table.

A referential integrity constraint requires that for each row of a table, the value in the foreign key matches a value in a parent key.

3. Tables

A relational database appears to the user to be collection of data tables. When accessed through SQL. The data always appears to be stored in simple tables. Tables are of following types

- **Base Tables:**Base tables actually contain the data in the database. A database consists of a collection of base tables that describes all the kinds of entities that must be kept track of. Each row of a table is a record contains data for one entity. (INTERNAL VIEW)
- **Derived Tables:** Derived tables consist of data selected from base tables. A derived table might be just a subset of a single base table, or it could be a combination of data from two or more base tables. (CONCEPTUAL VIEW)
- **Viewed Tables:** A viewed table (also called a View/External view) provides a window onto one or more tables. A view is simply a predefined derived table. Once a view is defined, you can use it as if it were a base table by naming it in SQL statements. We cannot add, delete, or update records in it. Because a view is a derived table.

4. Library / System Calls

a. Displaying all Tables Existing in System

```
SQL> Select • from tab;    or
SQL>Select· from cat;
```

b. To see view table structure

```
SQL>desc table_name
```

c. To see the content of table.

```
SQL> Select • from table_name;
```

d. To display the selected attributes

```
Select col1, co12 from table_ name Where (condition);
```

e. Command to drop a table

```
SQL> Drop table tablename;
```

f. Command to drop a column

```
SQL> alter table tablename drop column columnname
```

5. Security

Security is achieved by granting privileges. A privilege is a right to access an object such as a table, view etc. Privileges are classified into two categories depending upon what type of right they confer with the user.

i. Object Privileges

An object privilege is a right to perform a particular operation on an object. An object is either a table, view, sequence, procedure, function, or a package

ii. System Privileges

A system privilege is a right to perform certain operation in the system. For ex, the privilege to create a table is a system privilege.

Granting Object Privileges

In order to grant object privilege use Grant command

Syntax

GRANT {obj – priv | ALL [privileges]}[(col[,col]….)]{[, obj – priv | ALL [privileges]} [(col[,col]…)]…..ON objectTO {user | role | PUPLIC} [,{user | role | PUPLIC} [,{user | role | PUPLIC}]…[WITH GRANT OPTION]

Ex1: To grant SELECT privilege on EMP to user STEVE;

```
SQL> grant select on emp to steve;
SQL> Select * from scott.emp;
```

Ex2: To restrict UPDATE privilege to FEEPAID column of STUDENTS table

```
SQL> grant update(fee paid) on students to steve;
```

Ex3: SQL> grant select on emp to Steve with grant option:

Ex4: SQL> grant select on scott.emp to micheal

Revoking object privileges:

To revoke the privileges, granted earlier use REVOKE command

Syntax:

REVOKE {obj-priv | ALL [PRIVILEGES]}[,{obj – priv | ALL [privileges]}]…on[object] from{user | role | pupic} [,{user | role | puplic}]..[CASCADE CONSTRAINTS]
Ex: To revoke SELECT privilege fromsteve on emp table enter:

SQL> revoke select on emp from steve.

System Privileges
When DBA creates a new user, DBA assigns required system privileges to the new user.
Ex: SQL> grant create table to steve.

6. System Requirement
Hardware Configuration

Server Configuration: IBM X3400 Server, Intel Xeon 3.0 GHz CPU,512 KB Cache,2GB DDR RAM

Client Configuration: Lenovo Dual core 1.60 GHz /1 GB RAMS

Software Configuration

Data Base Server Software Version: -

Oracle Database 11g software, version 11.1.0.6.0

Data Base Client Software Version: Oracle Developer Suite 10g

Program 1

CREATION OF DATABASE

Problem Definition

Creation of database (exercising the commands for creation)

Problem Description

Creating the structure of an entity is done by using DDL commands. Data Definition Language (DDL) commands are those commands that define either the structure of an object or the database in which the objects are stored. These commands include the database creation, modification and administration.

CREATE TABLE: This command defines all characteristics of a table, including integrity and check constraints, column names and data types, storage option, and so on.

Data in relational model is stored in tables. So the first step is creating a table using SQL command CREATE TABLE. Before a table is created the following factors are to be finalized.

- The name of the table
- The number of columns in the table
- The name, data type and size of each column of the table
- The column and table constraints.

Pseudo code

Create Table Syntax

```
CREATE TABLE table_name ({column datatype [column_constraints] | table_constraints} );
CREATE TABLE table_name(column1datatype, column2datatype, column3datatype,.....
columnNdatatype, PRIMARY KEY( one or more columns ));
```

Problem Validation

Input

a) Create table dept (Deptno number(9), Dname varchar2(9), LOC varchar2(10), primary key(deptno));

b) Create table emp (Empno number(4) primary key, Mgr number(4), Empname varchar2(10), Deptno number(4));

Output

DEPT

Deptno	Dname	LOC

EMP

Empno	Mgr	Empname	Deptno

Program 2

SIMPLE AND COMPLEX QUERIES

Problem Definition

Simple to complex condition query creation using SQL Plus

Problem Description

SQL Plus commands are used fundamentally for accessing and managing data in database.

Simple Queries: Simple Queries includes insert, select, update, alter, delete, drop, savepoint, commit, rollback etc.

Complex Queries: Complex Queries includes Joins, Subqueries, constraints, views etc.

Pseudo code

```
INSERT INTO TABLE_NAME VALUES (value1,value2,value3,...valueN);
SELECT column1, column2, columnN FROM table_name;
SELECT * FROM table_name;
DELETE FROM table_name WHERE [condition];
DROP TABLE table_name;
UPDATE table_name SET column1 = value1, column2 = value2…, columnN = valueN
WHERE [condition];
ALTER TABLE table_name ADD column_name datatype;
ALTER TABLE table_name MODIFY COLUMN column_name datatype;
ALTER TABLE table_name DROP COLUMN column_name;
SAVEPOINT savepoint-name;
ROLLBACK;
COMMIT;
ALTER TABLE table_name ADD CONSTRAINT MyPrimaryKey PRIMARY KEY
(column1, column2..);
```

ALTER TABLE table_name ADD CONSTRAINT MyCheckConstraint CHECK (CONDITION);

ALTER TABLE table_name ADD CONSTRAINT MyUniqueConstraint UNIQUE(column1, column2...);

ALTER TABLE table_name DROP CONSTRAINT constraint-name;

SELECT column_name [, column_name] FROM table1 [, table2] WHERE column_name OPERATOR (SELECT column_name [, column_name] FROM table1 [, table2] [WHERE]);

SELECT column1, column2 FROM table1, table2 WHERE [conditions] GROUP BY column1, column2 HAVING [conditions] ORDER BY column1, column2;

SELECT column_name(s) FROM table1 INNER JOIN table2
ON table1.column_name=table2.column_name;

SELECT column_name(s) FROM table1 LEFT JOIN/RIGHT JOIN table2 ON
table1.column_name=table2.column_name;

SELECT column_name(s) FROM table1 FULL OUTER JOIN
table2 ON table1.column_name=table2.column_name;

CREATE VIEW view_name AS SELECT column_name(s) FROM table_name
WHERE condition;

Problem Validation

Input

a) select empname from employee where job='analyst' and sal>2000;

b) update employee set sal=(sal*5/100+sal) where sal<(select avg(sal) from emp);

Output

a)
```
empname
--------------------
  Smith
```

b) 4 rows updated.

Program 3

TRIGGERS

Problem Definition

Using Triggers and stored procedures.

Problem Description

Database Triggers are procedures that are stored in the database and are explicitly executed (fired) where the contents of a table are changed.

Triggers have 3 basic parts.

1. Trigger event: It is a SQL statement that causes the trigger to be fired. It can be insert, delete or update statement for a specified table.
2. Trigger restriction: It specifies a Boolean (logical) expression that must be true for the trigger to fire. It is specified using a where clause.
3. Trigger action: It is a procedure that contains the SQL statement and Pl / SQL code to be executed when a triggering statement is issued and the trigger restriction evaluation to true.

Types of triggers

Row trigger: A row trigger is fired each time the table is affected by triggering statement. Eg: if an update statement updates multiple rows of a table, a row trigger is fired once for each row affected by the update statement.

Statement trigger: A row trigger is fired once on behalf of the triggering statement independent of the number of rows the triggering statement affects.

Eg: If the trigger makes the security check on the time or the user.

Before trigger: It executes the trigger action before the triggering statement is executed. Eg: Before trigger are used to derive specific column values before completing a triggering INSERT or UPDATE statement.

After trigger: It executes the trigger after the triggering statement is executed. After trigger are used when you want the triggering statement to complete before executing the triggering action.

Pseudo code

Create or replace trigger [Schema.]

Triggername

{ Before | After }

{ Delete, Insert, Update [of column, -----] } On

[Schema.] tablename

[Referencing { OLD as old, NEW as new }] for

each row [when condition]

Raise_application_error (error-**no, ‘message’);**

Syntax of deleting a trigger

Drop trigger trigger-name;

Problem Validation

Input

Write a trigger to ensure that min salary is as follows

Tab – 1

Job	Salary
PRESIDENT	15000
MANAGER	10000
SALESMAN	5000
ANALYST	8000

Create or Replace trigger STRIG before update on emp

for each row

Begin

If upper (:new.job) = ‘PRESIDENT’ and :new.sal < 15000 then Raise-application-error (-20000, ‘sorry!’ || :new.sal || ‘is less than min salary’);

Elsif

Upper (:new.job) = ‘MANAGER’ and :new.sal < 10000 then Raise-application-error (-20000, ‘sorry!’ || ‘ Minimum salary for Manager is 10,000’);

Elsif

Upper (:new.job) = ‘SALESMAN’ and :new.sal < 5000 then Raise-application-

```
          error (-20000, ‘sorry!’ || ‘ Minimum salary for Salesman is 5,000’ );
Elsif
          Upper (:new.job) = ‘ANALYST’ and :new.sal < 8000 then Raise-application-
          error (-20000, ‘sorry!’ || ‘ Minimum salary for Analyst is 8,000’ );
Endif;
End;
/
```

Output

Tab-1

Program 4

FORM DESIGNING

Problem Definition

Creation of Forms using Oracle Forms Developer.

Problem Description

Oracle Forms is a powerful application-development tool for building client-server database applications that are portable to a variety of GUI and character mode platforms. Oracle Forms is part of Developer/2000, a comprehensive set of application development tools that supports the complete application development life-cycle.

Applications built with Oracle Forms and the other Developer/2000 tools are completely scalable, and are suitable for deployment at every level of your enterprise, from decision support applications for small workgroups, to mission-critical, transaction-intensive projects that must support hundreds of users.

Oracle Forms and the other Developer/2000 tools are optimized to take complete advantage of the powerful features in the Oracle7 Server, Oracle's leading database management server.

Oracle Forms Components

When you build applications with Oracle Forms, you use three components:

- Oracle Forms Designer
- Oracle Forms Generate
- Oracle Forms Runform

The Designer is an application development environment where you work with three types of Oracle Forms modules--forms, menus, and libraries. The Designer includes a set of visual tools that allow you to create objects, set their properties, and write code for your applications.

The Generate component is used to generate application files to create executable runfiles for runtime deployment. Generating a form module compiles all of its code objects and creates an .FMX runfile.

The Runform component is the runtime engine that form operators use to run a finished Oracle Forms application.

Oracle Forms Modules

Oracle Forms applications include three types of modules:

Forms are collection of objects and code, including windows, text items, check box, trigger, procedure and buttons.

Form can include any number of seperate windows.

Menus are collection of menu objects (main menu, pop_up menu, menu items) and menu commands code.

Libraries are collection of pl/sql procedures, functions and packages that can be called from other modules.

Form Designing

Form designing using Oracle Forms (Developer 2000) is a Front-end tool. The default Backend for developer 2000 is Oracle. The client operates on the Front-end, and any operations performed are finally reflected to Backend.

The primary tools used to develop customized forms are the Object Navigator, the Layout Editor and the Object Property sheets.When a file is stored, the source code filename will have an extension .Fmb (Form module binary). From the .fmb file an executable is generated which will have the extension .fmx.

Steps for creating a Form

a. Form may be created using File > New > Form menu choice or by pressing (Ctrl +Y). Get connected to Oracle by selecting connect option from file.

b. Enter the same user name and password which you have entered at SQL prompt while creating required table.

c. Go to block and check '+' to create or select a block from database. Click current users, select the block, select columns (Items), set the style to form layout and then click OK. A block will be created for you.

d. You can run your Form by pressing 'Control R'.

e. You can add the buttons from tool box to perform the requested action. Go to the layout editor, select the push buttons by double clicking it. Right click it go to the properties, change the name and label of the buttons.

Eg : Select Push buttons

- Change name and label as Save
- Right click Save button, select PL / SQL editor
- Select the trigger as when_button pressed for save button
- Write PLSQL code as

  ```
  Begin
          Commit-form
  End;
  ```
- Write PLSQL code for exit button if selected as

  ```
  Begin
          Exit_form
  End;
  ```

Pseudo code

Create table student (stud_name varchar2(20), stud_no number(9), collegename varchar2(20),
Class varchar2(9), avg number(9,2), marks1 number(5), marks2 number(5), Marks3 number(5), total number(9), grade or division varchar(6));

<u>Write a PL/SQL code on student</u>

```
Create or replace trigger updstudent after insert on student for each row
begin
:new.total := :new.m1 + : new.m2 + :new.m3;
:new.avg := :new.total/3;
if :new.m1 < 35 or :new.m2 < 35 or :new.m3 < 35 then
:new.division := ‘Fail’;
elsif :new.avg >= 70 then
:new.div := ‘Dist.’;
elsif :new.avg >= 60 and :new.avg <= 69 then
:new.division = ‘First’;
elsif :new.avg >= 50 and :new.avg <= 59 then
:new.division := ‘Second’;
```

```
else
:new.division:= 'Third';
endif;
end;
/
```

--If you execute the above trigger without entering values in Division, avg, total then you will get values in those columns by using the condition given in PL/SQL code.

Program Validation

Input

- Connect to the database
- Select the required table
- Change the layout and other desired properties
- Design the form by using buttons from the toolbox
- Run your form to see the output.

Output

Studno	
Studname	
College Name	
Marks 1	
Marks 2	
Marks 3	
Total	
Grade	

Program 5

PL/SQL PROCEDURE

Problem Definition

Writing a PL/SQL procedure for Data Validation.

Problem Description

Using data manipulation language commands we can perform data validation. It bridges the gap between database technology and procedural Language. It can be thought of as development tool that extends the facility of Oracle SQL database. It allows us to use all the SQL data manipulation statements as well as the cursor control operation and transaction processing.

With PL/SQL an entire block of statement can be sent to RDBMS engine at any one time.

PL / SQL Block Structure

```
Declare
declaration of memory variables used later
Begin
SQL executable statement for manipulating table data
Exception
SQL & PL/SQL code to handle errors that may crop up
during the execution of the above block
```

PL/SQL Basics

- Literal
- A literal is a numerical value or a character string used to represent itself.
- Comments

  --entire line will be treated as comment

  /* ---- */ the block is treated as comment
- Data types

Number, char, date, boolean, % type declares a variable or constant of the same data type as defined in table or view.

- Variables in PL/SQL block are named variables
- Assign values to variables
- Pick up a variable from table cell

Title book-info.book-title %type

- Declaring a constatnt

Eg : bonus-multiple constant number (3,2) := 0.10

- Displaying a user message on the screen

DBMS_OUTPUT is a package that includes a number of procedure and function that accumulate info in a buffer. So that it can be retrieved later.

PUT_LINE put a piece of info in the buffer followed by an end-of-line mark.

Eg : Setting server output on (or)Set serveroutput on

Eg : DBMS_OUTPUT.Put_line ("Display user message ");

Pseudo code

```
Create or replace procedure [ schema .] procedure name
(argument {IN, OUT, INOUT} data types, …) {IS,
AS} var declaration;
constant declaration;
begin
PL/SQL sub program body;
Exception
        Exception PL/SQL block;
End;
```

Program Validation

Input

Procedure update-item-cost (iItemId INTEGER, fnewcost NUMBER)

IS Fcurcost NUMBER(10,2);

Missing-cost exception;

```
Begin
Select item-cost into fcurcost from item Where
        item-id = iItemId;
If fcurcost is null then Raise
        missing-cost;
Else
        Update item set item-cost = fnewcost Where
        item-id = iItemId;
End            if;
Exception
When no-data-found then
        Insert into item-audit values (iItemId, ‘Invalid item identifier’);
When missing-cost then
        Insert into item-audit Values ( iItemId, ‘ Item code is null’);
When others then
        Insert into item-audit Values (iItemId, ‘ Miscellaneous errors’);
End update item-cost;
```

Output

Procedure created.

Program 6

REPORT DESIGNING

Problem Definition

Generation of reports using Reports Builder.

Problem Description

Oracle report consists of a design module, a conversion module and a runtime module. The report server runs as a process in the background and queues up reports for execution. Oracle Reports uses the same object navigator paradigm as Oracle form. The available object and structure differs, but the Object navigator performs the same function that it does in Oracle form.

Steps for Creating Report

a). Double click on Reports designer icon to start report designer

b). In the object navigator, double click on the data model button to open the Data model screen. In the data model screen, click on the Query tool. The cursor changes to plus (+) sign. Drag the cursor into the workspace and click in the workspace to create query box.

c). Double click on the Query Q-1 to bring up the query dialog box. Type the query in the query dialog box in the

select clause write

select name, state, zip from customer; then

close it.

If you are selecting from tables then click apply button of dialog box.

d). Return to data model.

e). Click on formula column icon on the tool palette to define a formula field. Drag the cursor to the workspace and click inside the Query dialog box. A field CF-1 appears when the mouse is released.

f). Double click CF-1, change the data type to char in program unit of CF-1.

Write Function CF-1 formula return char is

```
Begin
Return (:state || ‘-‘ || :zip);
End;
```

Compile it, close formula column dialog box.

g). Click on apply button and close button to return to Data model screen. h). Click on default layout tool icon, select one of the style.

i). Save the report by pressing ctrl+s.

j). Click on Run icon to run the report.

Pseudo code

```
select name, state, zip from customer;
```

Problem validation

Input

Query to select customer name, zipcode, state from customer table.

Output

Name	Zipcode	State
---- ----	------ ------	----- -----

Program 7
PASSWORD AND SECURITY FEATURES

Problem Definition

Creating password and security features of application.

Problem Description

All the core of security in the Oracle database user account, without an Oracle username there is no way that one can break into Oracle database. An Oracle database typically has Number of Oracle user accounts that were either created when the database was created or created later by database administrator. Once created an Oracle user has a number, and privileges that allow him to create and modify database object.

Once an Oracle user is given a username and password he can connect to Oracle database using any to

Pseudo code

Create User: The create user is a SQL command that can be used to define an Oracle account in database.

CREATE USER user_name IDENTIFIED {BY password | EXTERNALLY [AS 'certificate_DN'] | GLOBALLY [AS '[directory_DN]'] } [DEFAULT TABLESPACE tablespace | TEMPORARY TABLESPACE { tablespace | tablespace_group } | QUOTA integer [K | M | G | T | P | E] | UNLIMITED }ON tablespace [QUOTA integer [K | M | G | T | P | E] | UNLIMITED } ON tablespace];

SQL > grant CREATE SESSION TO user_name;

The create session system privilege must also be given to a new user account.

The following code eg shows the quota currently assigned to user fowler and the account of storage used.

Select * from dba_ts_quota where username = 'FOWLER';

DCL commands for granting and revoking privileges

a). Select username from all-users;

b). Grant command

GRANT privileges ON object to users;

Object name may refer to table, view, sequence, synonym, procedure, function, package. c). Revoke command

REVOKE Privileges ON TABLE OR VIEW FROM USERS;

Drop user

Drop user username;

Problem Validation

Input

Reveal details about oracle user

a. Select user-id From dba-users Where username = ‘xxxx’;

a. Select username, userid, password, default tablespace, temporary tablespace, created from dba-users where username = ‘FOWLER’;

Output

```
a)      User-id
            14
b) Username   userid   password   default-tablespace   temporary-tablespace   created
   FOWLER       16       ABCD        user-data               userdata         8-May-2000
```

Program 8

TABLE LOCKING

Problem Definition

Usage of file locking, table locking facilities in application

Problem Description

Oracle uses lock to control concurrent access to data. Locks are mechanisms intended to prevent destructive interaction between users accessing the same data. Table locks lock the entire tables, while row locks lock just selected rows. Thus locks are used to ensure data integrity while allowing max concurrent access to data by unlimited users.

Locks are used to achieve two important goals. 1. Data concurrency 2. Read consistency

Oracle lock is fully automatic and requires no user action .DBA locks the oracle data while executing SQL statement. This type of locking is called implicit locking. When a lock is put by user it is called explicit locking.

Types of locks

Two levels of lock that a DBA can apply. They are

1. Shared : Multi user can hold various share lock on a single resource
2. Exclusive: It prohibits all sharing of resources i.e. only one use has the sole ability to alter the resources until locks are released.

Pseudo code

```
LOCK TABLE [Table name] IN { ROW SHARE | ROW EXCLUSIVE | SHARE
UPDATE | SHARE | SHARE ROW EXCLUSIVE | EXCLUSIVE } MODE [
NOWAIT]
```

ROW SHARE Row share locks all concurrent access to a table.

SHARE UPDATE They prohibit other users to lock entire table exclusively

ROW EXCLUSIVE Row exclusive locks the same as row share locks, but also prohibit locking in share mode. These locks are acquired when updating, inserting or deleting.

SHARE ROW EXCLUSIVE They are used to lock whole table to selective update and to allow other users to lock at row in the table but not lock the table in share mode or to update rows.

NO WAIT Indicates that you do not wish to wait if resources are unavailable.

All locks are released under the following circumstances:

- The transaction is committed successfully
- A rollback is performed
- A rollback to a save point will release locks set after specified save point
- Row-level-locks are not released by rolling back to a savepoint
- Data locks are released by log off

Problem Validation

Input

Lock table emp in exclusive mode nowait;

Output

Table Locked

Program 9

CREATION OF FULL FLEDGED DATABASE

Problem Definition

Designing of Databases and its usage.

Problem Description

Creation of tables, Designing of forms and generation of reports.

Pseudo code

Refer to the previous exercises

Problem Validation

Creation of full fledged database

ANNEXURE - I

1.	Create tables' employee and department and insert data in them.
2.	Write simple SQL Queries to demonstrate DQL Command
3.	Write SQL Queries to demonstrate DQL Command using operators (logical and set operators).
4.	Write SQL Queries to demonstrate DQL Command using aggregate and character functions
5.	Write SQL Queries to demonstrate DQL command using string and date functions
6.	Write SQL Queries using clauses
7.	Write SQL Queries using subqueries
8.	Write SQL Queries using joins
9.	Write SQL Queries on integrity constraints.
10.	Write SQL Queries for creating views.
11.	Write SQL Queries to demonstrate DDL command.
12.	Write SQL Queries to demonstrate DML command
13.	Write SQL Queries to demonstrate TCL command.
14.	Write SQL Queries to implement database security using table locking
15.	Write simple PL/SQL programs.
16.	Write PL/SQL program for exception handling.
17.	Write PL/SQL programs for implementing cursors
18.	Write PL/SQL programs for implementing functions, procedures and packages.
19.	Write PL/SQL programs for creating triggers.
20.	Create forms and generate reports using oracle for emp and dept tables.

Suggested Reading:

1. Nilesh Shah , Database System Using Oracle, PHI, 2007.
2. Rick F Vander Lans, Introduction to SQL, Fourth edition, Pearson Education,2007.
3. Benjamin Rosenzweig, Elena Silvestrova, Oracle PL/SQL by Example, Third edition, Pearson Education, 2004.
4. Albert Lulushi, Oracle Forms Developer23's Handbook, Pearson Education, 2006.