Kalinga Institute of Industrial Technology
Deemed to be University,
Bhubaneswar-24

School of Computer Engineering

CS-2094

DBMS LAB Manual

L-T-P/week: 0-0-2 **Credit: 2**

Pre-requisites: NIL

Course Description: Database Management System Lab, DBMS Lab, is a laboratory subject that deals with the knowledge of creation, updating, and maintenance of the database. This lab allows the students to understand the basic concepts of the database management system and apply those concepts in creating and maintaining the database for real-life problems. This lab intends to provide in-depth knowledge in DBMS using SQL commands with different types of integrity constraints. It also focuses on the security aspects of a database system along with a brief discussion on accessing and manipulating data using PL/SQL blocks.

Course objectives

- To provide Oracle client/server Database Management System and its utilities.
- To provide an in-depth knowledge in database management using SQL commands with different types of integrity constraints.
- To provide the knowledge of using database security & authorization in order to access database for the different kinds of the user.
- To understand to access and manipulate data using PL/SQL blocks.

Course outcomes

- Able to understand the importance of database system and theiruses in real world transactions.
- Able to create and manipulate databases through SQL commands with different types of integrity constraints.
- Able to use database security & authorization in order to access database for the different kinds of the user.
- Able to access and manipulate data using PL/SQL blocks.

Mapping of CO to PO

CO \ PO	a	b	c	d	e	f	g	h	i	j	k	l
Able to understand the importance of database system and their uses in real world transactions.	M	H	M									M
Able to create and manipulate databases through SQL commands with different types of integrity constraints.	M	H	M	H	H		M					M
Able to use database security & authorization in order to access database for the different kinds of the user.			M		M	H						M
Able to access and manipulate data using PL/SQL blocks.		H		M	M							

Software Requirements

- Windows OS
- Oracle/My SQL/DB2/SQL Server

Text Book: Fundamentals of Database Systems, Elmsri & Navathe, 7th Edition

List of experiments

Assignment 1: Introduction to SQL

- Real time examples for practice purpose.

Assignment 2: Data Definition Languages

- Use SQL statements to create the following tables in College Student Database. Define NOT NULL DEFAULT, UNIQUE, and CHECK constraints wherever appropriate.
  STUDENT (StudentId, Last, First, Street, City, State, Zip, StartTerm, BirthDate, FacultyId, MajorId, Phone); **FACULTY** (FacultyId, Name, RoomId, Phone, DeptId); **COURSE** (CourseId, Title, Credits, PreReq); **CRSSECTION** (CsId, CourseId, Section, TermId, FacultyId, Day, StratTime, EndTime, RoomId, MaxCount); **TERM** (TermId, TermDesc, StartDate, EndDate); **ROOM** (RoomType, RoomDesc); **REGISTRATION** (StudentId, CsId, Midterm, Final, RegStatus); **DEPARTMENT** (DeptId, DeptName, FacultyId); **MAJOR**(MajorId, MajorDesc); **LOCATION** (RoomId, Building, RoomNo, Capacity, RoomType).

  Insert minimum 5 records in each table.

- Use SQL statements to create the following tables in Corporation Employee Database. Define NOT NULL DEFAULT, UNIQUE, and CHECK constraints wherever appropriate.
  EMPLOYEE (EmployeeId, Lname, Fname, PositionId, Supervisor, HireDate, Salary, Commission, DeptId, QualId); **POSITION** (PositionId, PosDesc); **DEPT** (DeptId, DeptName, Location, EmployeeId); **QUALIFICATION** (QualId, QualDesc); **EMPLEVEL** (LevelNo, LowSalary, HighSalary); **DEPENDENT** (EmployeeId, DependentId, DepDOB, Relation).
  Insert minimum 5 records in each table.

Assignment 3: Data Management & Retrieval

- Display all employee names (last name and first name separated by a comma and a space) and salary with appropriate column aliases.
- Display all employees who do not get any commission.
- Display unique building names from LOCATION table.
- Display all course sections offered in Winter 2003.
- Display names of faculty members who work in department 1 or 2. Use IN operator in your query.
- Find all New York and New Jersey students.
- Give a 10% raise to employee number 111.
- Delete department number 30 from the department table. If it is not successful, write down your suggestion to make it work.
- For each course ID, display the maximum count in descending order.
- Insert a new term in the TERM table.
- Create a custom prompt for user to input any value betwwen 50 and 99 into DeptId column.
- Find courses with no required prerequisite.
- Display faculty names in descending order by their department, but in alphabetical order by their name within each department.
- Find faculty members whose name start with C.
- Find students who started in year 2003. Use start term column and wild card.
- Write SQL*Plus command to display a character column in 30 columns, and a numeric column with 9,999.99 format.

Assignment 4: Functions and Groupings

- Display all employee names (last name and first name separated by a comma and a space) with proper case, and salary with currency format.
- Display all employees with their commission value. Display 0 commission for employees who do not get any commission.
- Count the total number of rooms in LOCATION.
- Count the distinct building names in LOCATION.
- Display all student names and birth dates. Display birth dates with the format '20 OCTOBER, 1970'.
- Find the average, highest, and lowest age for students.
- Display the total number of dependents for each employee for employees who have at least two dependents.
- Display only the year value from each employee's hire date.
- Find average employee commission.
- Ignore nulls.
- Do not ignore nulls.
- Find sum of Maximum count by term by course. (GROUP BY two columns)
- Find 2 to the power 10.
- Display courses and prerequisites. If there is no prerequisite, display 'none' else display 'one'.
- Count number of faculty members by each department.
- Display average employee salary by department, but do no include departments with average salary less than $75,000.
- Find number of years employees have been working for. Display integer part of value only.
- Find students who are born in the month of May.
- Display employee's last name and first name, followed by salary+commission if commission is not null, else display salary only.
- Display employee's full name followed by a message based on salary. If salary is above $100,000, display "HIGH". If salary is between $50,000 and $100,000, display "MEDIUM". If salary is below $50,000, display "LOW".
-

Assignment 5: Joins and Set Operations

- Use the Corporation database tables to design the following queries.
    - Display all employee names and their department names.
    - Find name of the supervisor for employee number 433.
    - Find all employees' full names (lastname, firstname format) with salary, and their supervisor's name with salary.
    - Find each employee's salary information and level based on the salary.
    - Display each employee's name, department name, position description, and qualification description. Which employee is missing? Why?
    - Find all employees in the sales department.
    - Display employee names and dependent information using an outer join.
        - Find out the names and number of years worked along with their department names in descending order by number of years worked.

- Who works in the same department in which John Smith works?

- Use the College database tables to design the following queries.
    - Display a student's full name along with his/her major's description.
    - Get names of students, who received final grade 'F' in Winter 2003.
    - Display student names, their faculty advisors names, and faculty's office location.
    - Get Spring 2003 course sections with the faculty member assigned to teach the class. Include course sections without any faculty assigned to them.
    - Display course titles along with their prerequisite names. Display courses without prerequisite also.
    -

Assignment 6: Nested Queries

- Use the Corporation database tables to design the following subqueries.
    - Display employee Jinku Shaw's department name.
    - Find name of the supervisor for employee number 433.
    - Who has same qualification as Stanley Garner?
    - Which department has more employees than department 20?
    - Which employees are working in the company longer than Larry Houston?
    - Find all employees in the sales department by using a nested query.
    - Create a new table, EMP30, and populate it with employees in department 30, using an existing table and a subquery. Use EmployeeId, Lname, Fname, HireDate and Salary columns.
    - Add more rows to EMP30 table with employee in department 40. Do not transfer employee's salary.
    - Update salary of newly transferred employee from EMPLOYEE table to EMP30 table with MERGE statement, and INSERT employees who are not in EMP30 table.
    - Find employees with minimum salary in their own department with the use of correlated subquery.
    - Use multiple level subquery to display dependent information for employees, who belong to FINANCE department.
    - Use set operator and subquery to find employees, who do not have any dependents.
    - Write a subquery that finds average salary by each department. Check to find if employee 543's salary satisfies =ANY, <ANY, >ANY, <ALL, or >ALL condition against those departmental average salaries.
- Use the College database tables to design the following subqueries.
    - Display student Jose Diaz's faculty advisor's name and phone number.
    - Find rooms with bottom-2 capacities. Do not include office rooms.
    - Find Spring 2003 course sections with the top three maximum count numbers.
    - Find all information regarding classrooms (RoomType = 'C').
    - Create a new table, SP03SECT, for Spring 2003 semester course sections using a subquery. Include CourseId, Section, FacultyId and RoomId columns only.
    - Delete rows from SP03SECT table for faculty Mobley.
    - Find faculty members who do not teach any course in Spring 2003 semester. Use correlated subquery with NOT EXISTS operator on SP03SECT table.

Assignment 7: Objects, Transactions and Data Control

- Create a view to include all employee information, but hide salary and commission.
- Create a view to include department name and average salary by department.
- Create an index to search students faster based on their major ID.
- Create a sequence to add room IDs and then insert a new room into LOCATION table using the newly created sequence. What is the CURRVAL after the new row is inserted?
- GRANT only SELECT privilege to another user on your TERM table.
- INSERT a new Winter 2004 term in the TERM table. Use a SELECT query to see the result. Ask the user with the SELECT privilege (from activity 5) to view your TERM table. COMMIT your transaction and ask the same user to view the table again.
- Lock student ID 00101 for update of major to 600. Update the row and commit it.
- Create a view that will display name, department number, and total income (salary + commission) of each employee in department 10. Prevent change of department through the view.
- Create a view that will display department names and sum of all employee income by department.
- Create a sequence deptid_seq to generate department Id (in dept table) and another sequence empid_seq to generate employee Id (in the employee table). Use deptid_seq to add a new department in the dept table. Now, add yourself as a new employee with empid_seq in the department you just added.

Assignment 8: PL/SQL: A Programming Language

- Create a program script that uses a PL/SQL anonymous block to perform the following: Use a host variable AREA to store the result. Declare a local variable RADIUS with numeric data type. Declare a constant PI with value 3.14. Assign a value to the variable RADIUS by using a substitution variable. Calculate area of a circle by using formula

- **AREA = PI * RADIUS * RADIUS**

- Then print result in SQL*Plus.

- Write a PL/SQL block to find the square, cube, and double of a number inputted with a substitution variable, and print results using built-in package DBMS_OUTPUT.

- Write a PL/SQL block to swap the values of two variables. Print variables before and after swapping.

- Write a PL/SQL program to input hours and rate. Find gross pay and net pay. The tax rate is 28%. Print your results. (No need to perform overtime calculations)

- Write a PL/SQL program with two variables for the first name and the last name. Print the full name with last name and first name separated by comma and a space.

Assignment 9: Control Structures and Embedded SQL

- Write a PL/SQL block to find out if a year is a leap year. A leap year is divisible by 4, but not divisible by 100, or it is divisible by 400. For example, 2000 and 2004 are leap years, but 1900 and 2001 are not leap years. (*Hint:* The function MOD (n, d) divides *n* by *d* and returns the integer remainder from the operation)

- Write a PL/SQL block to print all odd numbers between 1 and 10 using a basic loop.

- Using a FOR loop, print the values 10 to 1 in reverse order.

- Create a table called ITEM with one column ItemNum with NUMBER type. Write a PL/SQL program to insert values of 1 to 5 for ItemNum.

- Input a number with a substitution variable and then print its multiplication table using a WHILE loop.
- Input a month number between 1 and 12 and a 4-digit year, and print number of days in that month. For February (month = 2), check for leap year to display number of days equal to 28 or 29.
- Use a PL/SQL block to delete item number 4 from the ITEM table.
- Write a PL/SQL block to ask a user to input a valid employee Id. Retrieve employees name, qualification description, salary and commission. Print name, qualification and sum of salary and commission.
- You went to a video store and rented a DVD that is due in 3 days from the rental date. Input rental date, rental month and rental year. Caclculate and print return date, return month and return year. For example,

RentalDate	Rental Month	Rental Year	Return Date	Return Month	Return Year
2	12	2003	5	12	2003
27	2	2000	1	3	2000
30	12	2003	2	1	2004

Assignment 10: PL/SQL Cursors and Exceptions

- Create a PL/SQL block to declare a cursor to select last name, first name, salary, and hire date from the EMPLOYEE table. Retrieve each row from the cursor and print the employee's information if the employee's salary is greater than $50,000 and the hire date is before 31-DEC-1997 (explicit cursor problem).
- Create a PL/SQL block that declares a cursor. Pass a parameter to the cursor of the type that is the same as the Salary column in EMPLOYEE table to the cursor. Open the cursor with a value for the parameter. Retrieve information into the cursor for a salary higher than the parameter value. Use a loop to print each employee's information from the cursor (cursor with parameter problem).
- Create a PL/SQL block to increase salary of employees in department 10. The salary increase is 15% for the employees making less than $100,000 and 10% for the employees making $100,000 or more. Use a cursor with a FOR UPDATE clause. Update the salary with a WHERE CURRENT OF clause in a cursor FOR loop (cursor FOR loop problem).
- Write a PL/SQL block to retrieve employees from the EMPLOYEE table based on a qualification Id. If the qualification Id returns more than one row, handle the exception with the appropriate handler and print the message 'More than one

employee with such qualification'. If the qualification Id returns no employee, handle the exception with the appropriate handler and display the message 'No employees with such qualification'. If the qualification Id returns one employee, then print that employee's name, qualification and salary (predefined server exception problem).

- Write a PL/SQL block that retrieves entire COURSE table into a cursor. Then, ask user to input a courseId to search. If course exists then print its information, but if course does not exist throw a user-defined exception and handle it with a message 'Course does not exist' in the COURSE table when you execute the block with a course ID such as CIS555. (user-defined exception problem).

- Write PL/SQL block that asks user to input first number, second number and an arithmetic operator (+, -, *, or /). If operator is invalid, throw and handle a user-defined exception. If second number is 0 and the operator is /, handle ZERO_DIVIDE predefined server exception.

Assignment 11: PL/SQL Named Blocks: Procedure, Function, Packagesand Trigger

- Write a procedure that is passed a student's identification number and returns back the student's full name and phone number from the STUDENT table to the calling program. Also write an anonymous block with the procedure call.

- Write a function and pass a department number to it. If the DEPT table does not contain that department number, return a FALSE value, otherwise return a TRUE value. Print the appropriate message in the calling program based on the result.

- Write a package that contains a procedure and a function. The procedure is passed a room number. If the room number exists, the procedure gets the capacity of the room and the building name from the LOCATION table. If the room number does not exist, the procedure performs the appropriate exception-handling routine. The function is passed a *csid* and returns maximum number of seats available in the course section.

- Write a trigger that is fired before the DML statement's execution on the EMPLOYEE table. The trigger checks the day based on SYSDATE. If the day is Sunday, the trigger does not allow the DML statement's execution and raises an exception. Write the appropriate message in the exception-handling section.

- Write a trigger that is fired after an INSERT statement is executed for the STUDENT table. The trigger writes the new student's ID, user's name, and system's date in a table called TRACKING. (Note: You must create the TRACKING table first).

- Create a complex view EMP_DEP_VIEW using an outer join between EMPLOYEE and DEPENDENT table with employee names, dependent's birthdate and relation. The outer join will also return employees without any dependents. Now, create an INSTEAD OF trigger based on EMP_DEP_VIEW to enable you to delete employee 433 through view.

Guidelines to be followed:

1. Students should be regular and come prepared for each laboratory class.
2. Students should bring their lab record and practice note books to every class.
3. The prescribed text and reference books and class notes can be kept ready for reference if required.
4. Students have to complete their lab experiments in the lab and be capable to explain and show the modifications, output results as and when required by the Faculty/Lab Programmers/Teaching Assistants responsible for that lab. All these are to ensure that the students' capability is built up to understand, debug and modify codes as per the requirements.
5. In case a student misses a class, it is his/her responsibility to complete the missed experiment(s).
6. Once the experiment(s) get executed, they should show the program and results to the Faculty/Lab Programmers/Teaching Assistants responsible for that lab. and copy the same in their observation book and get signed which ensures him/her with day to day performance marks.
7. Student should submit his/her record by/in the next lab session to ensure the lab record marks gets evaluated. Failing to do so reduces the lab record marks.
8. Lab examination questions need not necessarily be limited to the questions in the laboratory manual, rather it could include some variations and/or combinations of questions.

Scheme of Evaluation

The DBMS Lab follows the evaluation of total marks **100**, that is divided into Continuous Internal Assessment and **End-Sem** Final Evaluations. The details are as below:

Internal Assessment (Continuous evaluation over the semester):	**60 Marks**
End-Tem Evaluation (At the End of the Semester):	**40 Marks**
Minimum Marks for passing in the Lab:	**50 Marks**