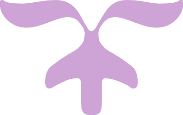


DATABASE MANAGEMENT SYSTEM LAB GUIDE

KCS 551

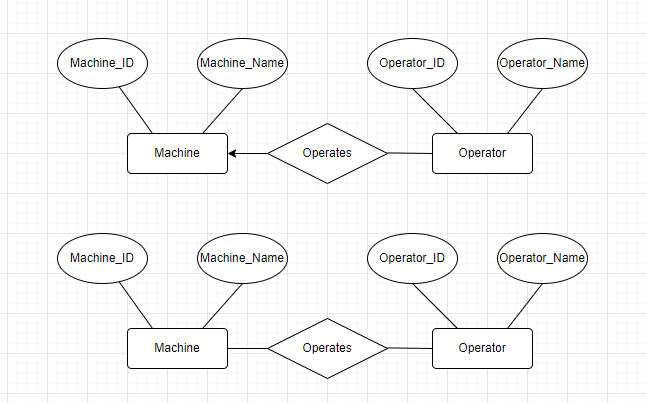




# Experiment-1

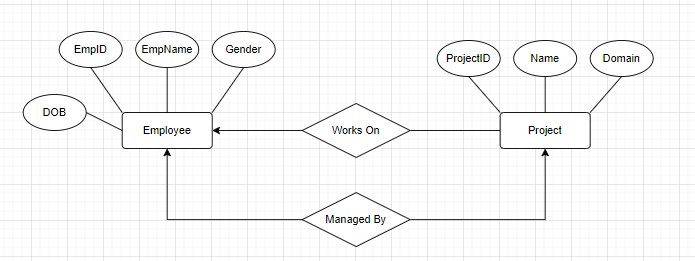
**Case Study 1:**

Draw E-R diagrams to indicate the following relationships between entity set Operator and entity set Machine: - Each Machine can be operated by many Operators but each Operator can operate only one machine. An operator can operate many machines and each machine can be operated by many Operators.



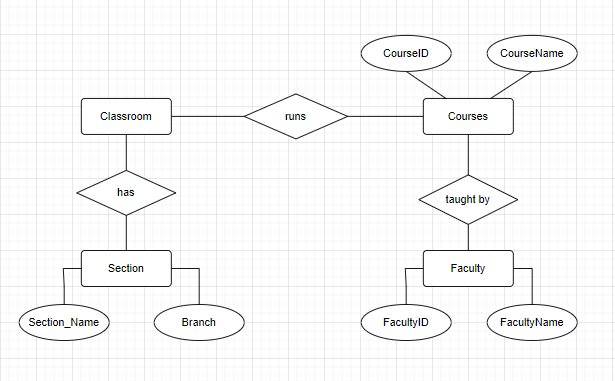
# Case Study 2:

An organization having a set of employees to execute a set of projects. Each employee may be working on more than one project, each project is managed by a manager and a manager is also one of the employees.



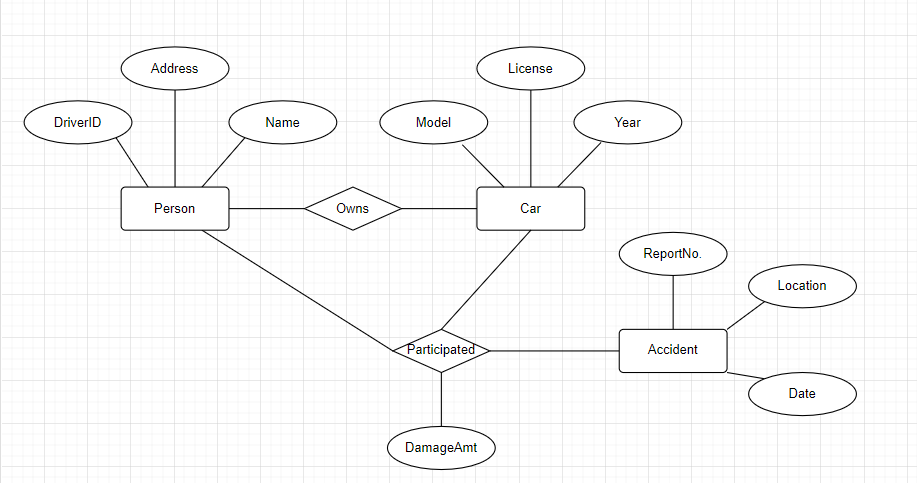
# Case Study 3:

Preparation of time table of an Engineering College, catering for a number of Sections (Year/Branch/Section), a number of courses, a numbers of faculty members teaching the courses and a number of class rooms (ignorelabs).



# Case Study 4:

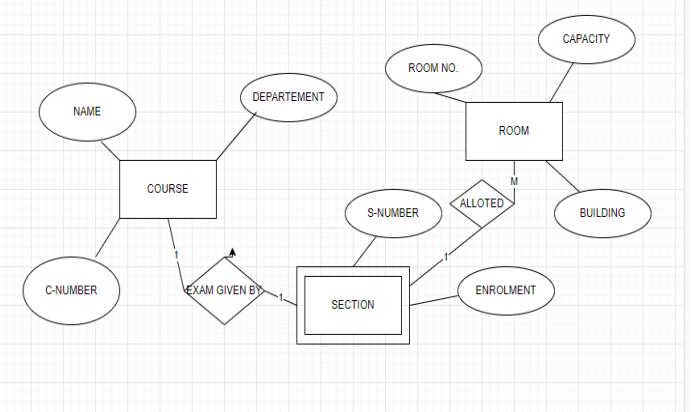
Construct an E-R diagram for a car-insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents.



# Case Study 5:

Consider a university database for the scheduling of classrooms for final exams. This database could be modelled as the single entity set exam, with attributes course-name, section- number, room-number, and time. Alternatively, one or more additional entity sets could be defined, along with relationship sets to replace some of the attributes of the exam entity set, as

* Course with attributes name, department, and c-number
* Section with attributes s-number and enrolment, and dependent as a weak entity set on course
* Room with attributes r-number, capacity, and building



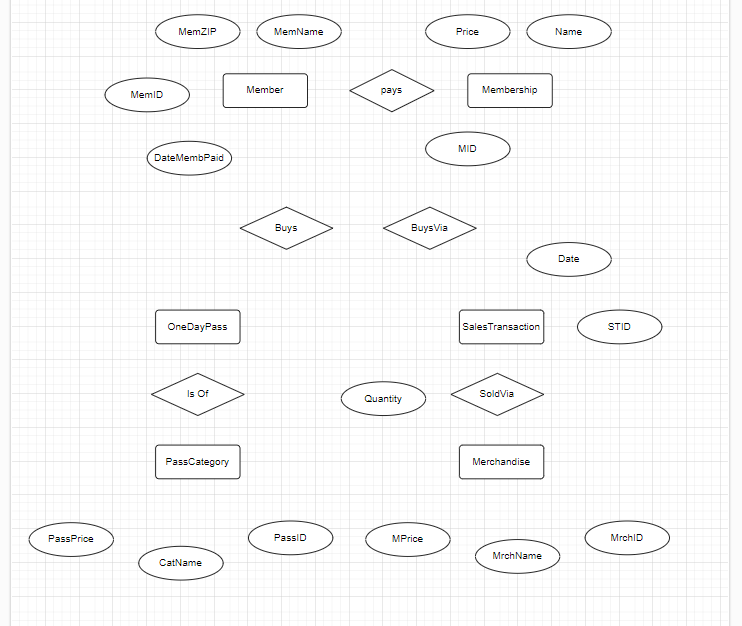
**Experiment-2**

# Case Study 18:

The following are the requirements for the Gym Fitness Database

* For each MEMBER we keep track of the unique MemdID, a well as Name, Zip, and the Date the membership was paid
* For each MEMBERSHIP type we keep track of the unique Mid, as well as MName and Price
* For each PASS CATEGORY we keep track of the unique PassCatID, as well as PCName and Price
* For each ONE DAY PASS we keep track of the unique PassID and Date
* For each MERCHANDISE item we keep track of the unique MrchID, as well as Name and Price
* For each sale TRANSACTION we keep track of the unique Tid and Date
* Each member pays for exactly one membership type; each membership type has at least one member but can have many members
* Each member can buy many day passes but does not have to buy any, each day pass was bought by exactly one member
* Each day pass belongs to exactly one pass category; a pass category can have many individual day passes issued for it but does not have to have any
* Each sale transaction involves exactly one member; each member can be involved in many sale transactions but does not have to be involved in any
* Each merchandise item is sold via at least one sale transaction but it can be sold via many sale transactions; each sale transaction involves at least one merchandise item but can involve many merchandise items
* Every time a merchandise item is sold via a sale transaction, we keep track of the quantity (how many instances of that particular merchandise item were sold via that particular sale transaction)

Your task is to create and ER Diagram based on these requirements.

****

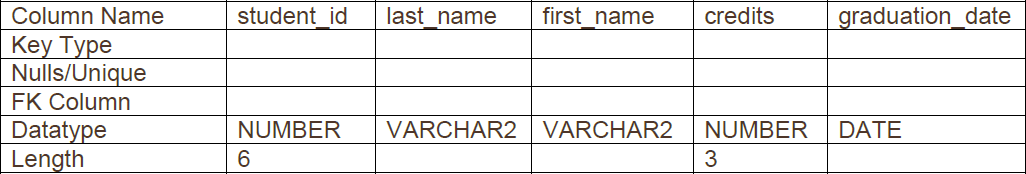
# Lab 3 & Lab 4 – DDL Commands & Constraints

**Objective**: At the end of the assignments, participants will be able to understand basic DDL, Create table with constraints, Alter, Truncate, Drop and Rename

# Student has to submit a file named as < Group No.\_Branch\_Lab Group\_Day3> with the complete solution.

## Exercise -1:

1. Complete the GRADUATE CANDIDATE table instance chart. Credits is a foreign-key column referencing the requirements table.

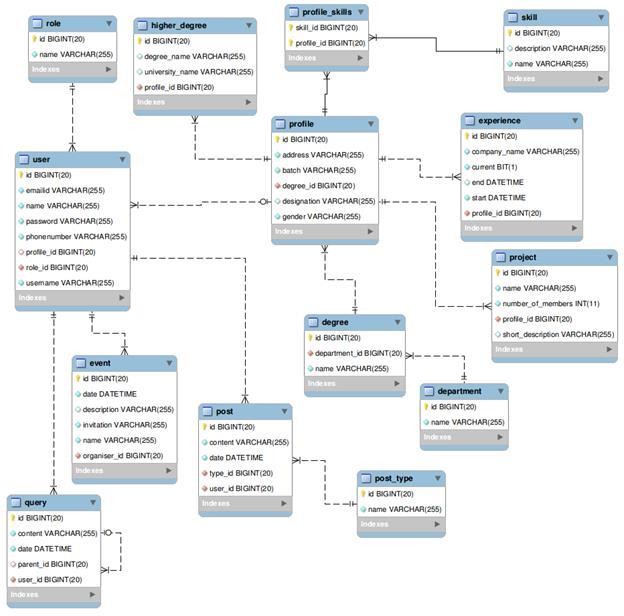


1. Write the syntax to create the grad\_candidates table.
2. Confirm creation of the table.
3. Create a new table using grad\_candidates with the following syntax: CREATE TABLE o\_grad\_candidates AS (SELECT \* FROM grad\_candidates);
4. Create a new table using a subquery. Name the new table your first name -- e.g., gaurav\_table. Using a subquery, copy grad\_candidates into gaurav\_table.
5. In your o\_grad\_candidates table, enter a new column called “adm\_date.” The datatype for the new column should be VARCHAR2. Set the DEFAULT for this column as SYSDATE.
6. In your o\_grad\_candidates table, increase the length of last\_name column by 10 and remove the credits column.
7. Create a new column in the smith\_table table called start\_date. Use the TIMESTAMP WITH LOCAL TIME ZONE as the datatype.
8. Write syntax to change the name of credit column by grad\_credit.
9. Insert 5 tuples in gaurav\_table.
10. Truncate the gaurav\_table table. Then do a SELECT \* statement. Are the columns still there?
11. What the distinction is between TRUNCATE and DROP for tables?
12. List the changes that can and cannot be made to a column.
13. Rename o\_grad\_candidates to n\_grad\_candidates.

**Exercise -2:**

* 1. Convert the ER Diagram into a neat and clean relational model.
  2. Create all the tables using SQL commands.
  3. Show all the tables with the constraints of your case study. Explain the reason of implementing that constraint on the required columns.

## Bonus Exercise -3:



Q-1) Write a query to create profile\_skills table. Q-2) Write a query to create user table.

Q-3) Write a query to create role table.

Q-4) Write a query to create department table. Q-5) Write a query to create degree table.

Q-6) Write a query to create profile table.

Q-7) Write a query to create higher\_degree table. Q-8) Write a query to create experience table.

Q-9) Write a query to create skill table.

Q-10) Write a query to add a new column named description of type varchar (255) to role table.

Q-11) Write a query to change the type of field description in the role table to varchar (500). Q-12) Write a query to remove the column description from the role table.

# Lab 5 -DML Commands

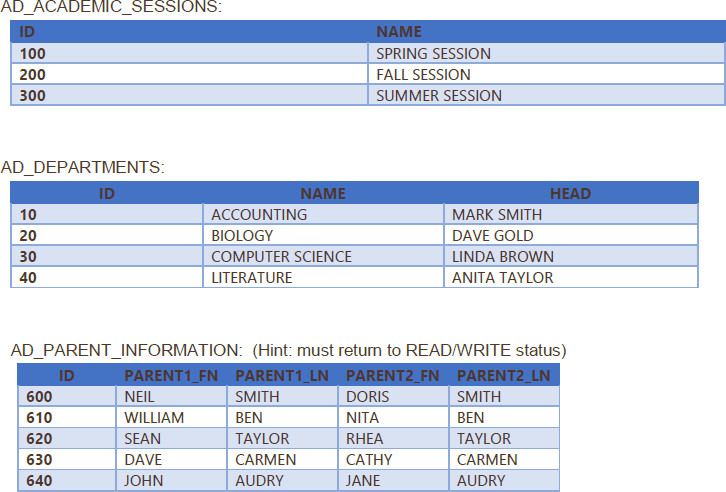
**Objective**: At the end of the assignments, participants will be able to understand basic DML, Insert rows using insert command, Update using update command and delete using delete command.

## Exercise -1:

1. Insert 5 rows into the tables for the database created of your case study.

## Exercise -2:

1. Create the tables mentioned below and insert the rows as shown. Please assume the datatype and constraints as required.



1. Add 2 new rows in AD\_ACADEMICS\_SESSIONS table with name as “Summer Break Session” and “Winter Break Session”.
2. Update the name “Computer Science” with “Computer Science and Engineering” in AD\_DEPARTMENTS table.
3. Update the PARENT1\_LN as NULL for ID 620 in AD\_PARENT\_INFORMATION table.
4. Delete the 2 new rows added in question 2.

# Lab 6 –DQL and Sorting Data

**Objective**: At the end of the assignments, participants will be able to understand basics of DQL commands, Select, Conditional retrieval, operators, pattern matching, order by clause.

**Use the default schema of EMP Table & DEPT Table of the database and implement the listed queries:**

**Exercise -1: Queries based on Conditional Retrieval of Rows**

1. List department names and location from the department table.
2. List the employees belonging to the department 20.
3. List the name and salary of the employees whose salary is more than 1000.
4. List the employee number and name of managers.
5. List the name of clerks working in the department 20.
6. List the names of analysts and salesmen.
7. List the details of the employees who have joined before the end of September 1981.
8. List the names of employees who are not managers.

**Exercise -2: Special Operators IN and BETWEEN**

1. List the name of the employee whose employee numbers are 7369,7521,7839,7934, 7788.
2. List the employee details not belonging to the department 10, 30 and 40.
3. List the employee name and salary, whose salary is between 1000 and 2000.
4. List employee names, who have joined before 30th June 81 and after December 81.

**Exercise -3: DISTINCT Clause with SELECT**

1. List the different jobs (designations) available in the EMP table.

**Exercise -4: Working with NULL Values**

1. List the employee names, who are not eligible for commission.
2. List the name of the employees and designation (job) of the employee, who does not report to anybody (manager is NULL).
3. List the employees not assigned to any department.
4. List the employees who are eligible for commission.
5. List the details of employees, whose salary is greater than 2000 and commission is NULL.

**Exercise -5: Matching Pattern with Column**

1. List the employees whose names start with an “S”.
2. List the employees names ending with an “S”.
3. List the names of employees whose names have exactly 5 Characters.
4. List the employee names having “I” as the second character**.**

**Exercise -6: Using Expression with Column**

1. List the name, salary and PF amount of all the employees (PF is calculated as 10% of salary).
2. List the names of employees, who are more than 2 years old in the organization.

**Exercise -7: Ordering the Results of a query**

**ORDER BY** clause to impose an order on the result of a query,

**ORDER BY** clause used with **SELECT** statement.

**SYNTAX: SELECT [DISTINCT] <col list>| <exp> FROM table name WHERE cond ORDER BY col [ASC|DESC].**

**One or more column can be specified in ORDER BY clause.**

1. List the empno, ename, sal in ascending order of salary.
2. List the employee name and hiredate in descending order of Hiredate.
3. List the employee name, salary, job and department No. in ascending order of Department No. in ascending order of Department No and then on descending descending order of salary.

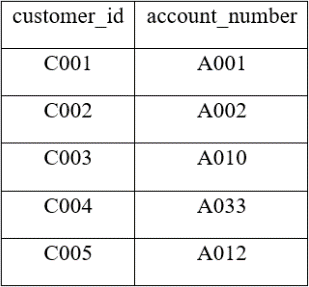
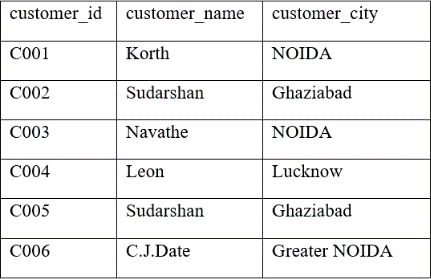
# Lab 7 –Subquery and Nested Query

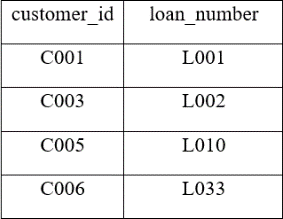
**Exercise -1:**

**To solve the below mentioned queries (1-10), please upload the SMS database script provided by your faculty.**

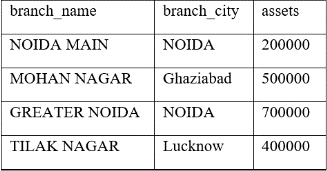
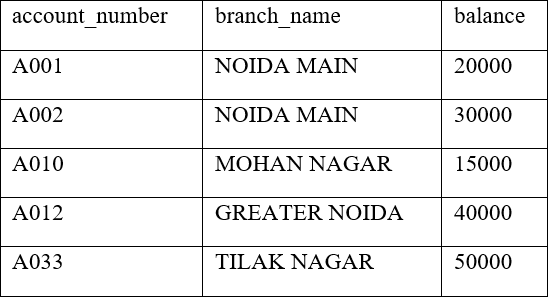
* 1. Find the names of faculty who are working in the same department in which Gagan Kumar Verma is working.
  2. Find the status of research project/s in which Ram Mohan Prasad is working.
  3. Find the names of students who have registered for mini-project.
  4. Find the names of faculty who are working in more than two research projects.
  5. Find the number of students registered in each course of a department.
  6. Create a report that displays the faculty id, last name, and salary of all faculty who earn more than the average salary. Sort the results in ascending order of salary.
  7. Write a query that displays the faculty id and last name of all employees who are working in a department with any employee whose last name includes letter **t.**
  8. Find the first name and salary of all faculty who are reporting to head of the department Ayush Giri.
  9. Find the names of students who are taking courses of faculty Mr. Ram Mohan Prasad.
  10. Find the department in which a faculty has joined most recently.

# Exercise -2:

**To solve the below mentioned queries (1-10), please create the below mentioned database. depositor: borrower: customer:**



**account: branch:**

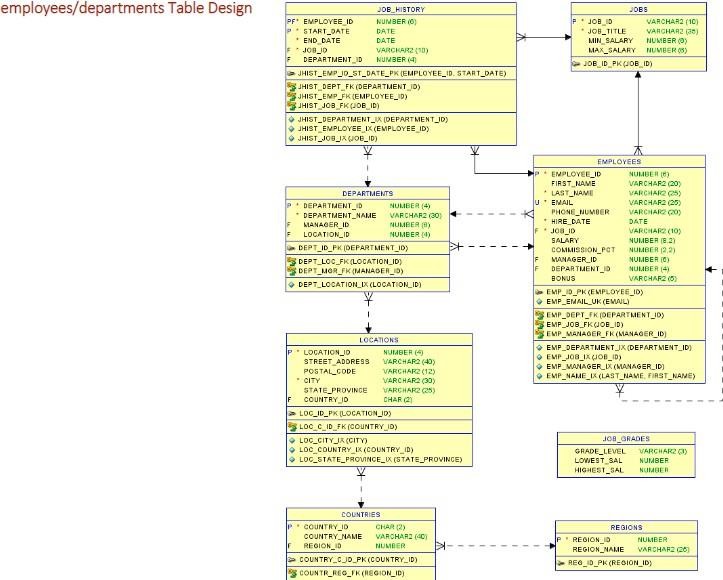


1. Find customer ids of those customers who are borrower from the banks and who appear in the list of account holders.
2. Find those customer names who are borrower.
3. Find the name of the customers who have a loan from the bank, but do not have an account at the bank. (Hint: use NOT IN)
4. Get the Customer Id and name of those customers who have both account and loan from the bank.
5. Get Branch Name of the branch having highest average balance amongst all branches.
6. Find the names of all branches that have assets greater than those of at least one branch located in NOIDA. Use some
7. Find the names of all branches that have assets greater than that of each branch located in NOIDA. (Use All)
8. Get the names of the customers who have account in each branch located in Noida.

# Lab 8 – Joins using multiple tables

**To solve the below mentioned queries, please upload the employee\_department script provided by your faculty.**

**To understand the employees/departments database, please refer below mentioned table design**



# Exercise -1: CROSS-JOIN and NATURAL JOIN

1. Create a cross-join that displays the last name and department name from the employees and departments tables.
2. Create a query that uses a natural join to join the departments table and the locations table. Display the department id, department name, location id, and city.
3. Create a query that uses a natural join to join the departments table and the locations table. Restrict the output to only department IDs of 20 and 50. Display the department id, department name, location id, and city.

# Exercise -2: INNER JOIN

1. Join the database locations and departments table using the location\_id column. Limit the results to location 1400 only.
2. Display the city, department name, location ID, and department ID for departments 10, 20, and 30 for the city of Seattle.
3. Display country name, region ID, and region name for Americas.
4. Write a statement joining the employees and jobs tables. Display the first and last names, hire date, job id, job title, and maximum salary. Limit the query to those employees who are in jobs that can earn more than $12,000.
5. Display job title, employee first name, last name, and email for all employees who are stock clerks.
6. Write a statement that displays the employee ID, first name, last name, manager ID, manager first name, and manager last name for every employee in the employees table. Hint: this is a self-join.
7. Display the location ID, city, and department name for all Canadian locations.
8. Query and display manager ID, department ID, department name, first name, and last name for all employees in departments 80, 90, 110, and 190.
9. Display employee ID, last name, department ID, department name, and hire date for those employees whose hire date was June 7, 1994.

# Exercise – 3: OUTER JOIN

1. Return the first name, last name, and department name for all employees including those employees not assigned to a department.
2. Return the first name, last name, and department name for all employees including those departments that do not have an employee assigned to them.
3. Return the first name, last name, and department name for all employees including those departments that do not have an employee assigned to them and those employees not assigned to a department.

# Exercise – 4: SELF JOIN

1. Display the employee’s last name and employee number along with the manager’s last name and manager number. Label the columns: Employee, Emp#, Manager, and Mgr#, respectively.
2. Modify question 1 to display all employees and their managers, even if the employee does not have a manager. Order the list alphabetically by the last name of the employee.
3. Display the names and hire dates for all employees who were hired before their managers, along with their managers’ names and hire dates. Label the columns Employee, Emp Hired, Manager and Mgr Hired, respectively.

# Lab 9 – Restricting Data Using Group By Clause Exercise -1:

**Write the answers of following questions in practical file**

1. What is the difference between WHERE clause and HAVING clause?
2. Why do we need to use the same columns that are selected in the SELECT list in the GROUP BY Clause?
3. Why do we need to have columns that are not part of GROUP BY Clause as part of aggregate function?
4. Will fixing the error by adding the columns to Group By clause, result in correct output?
5. In SQL, what’s the difference between the having clause and the group by statement?

# Exercise -2:

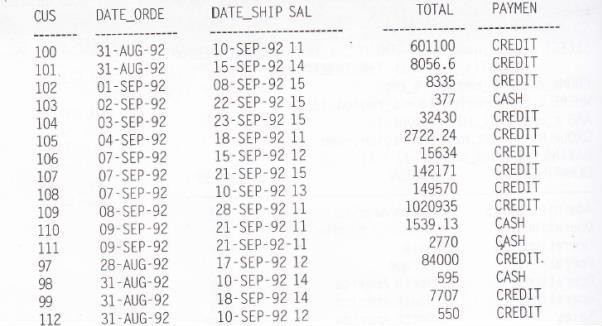
**Solve the following queries on the basis of SMS database.**

1. Write a query in SQL to find the number of faculty in each department along with the department id.
2. Write a query in SQL to find the sum of the allotment of salary amount of all departments.
3. Write a SQL query to display the average salary amount in each department along with their id.
4. Write a SQL query to find the number of faculty getting salary more than or equal to Rs.60000.
5. Write a SQL query to find the number of faculty teaching a course and number of students registered in that course.
6. Write a SQL query to find the number of faculty of each department who are working in any research project.
7. Write a SQL query to find the first name, faculty id and department of all faculty who are working in more than two research projects.
8. Write a SQL query to display the name and id of faculty who has maximum experience.
9. Write a SQL query to find the first name and department of all faculty who have completed any research project.
10. Write a SQL query to find the student names and their departments who have registered in mini projects of Artificial Intelligence domain.

# Bonus Exercise – 3:

**Solve the following queries on the basis of schema given.**

**SCHEMA: S\_ORD (CUS\_ID, DATE\_ORDER, DATE\_SHIPMENT, SP\_NO, TOTAL, PAYMENT)**



1. Write SQL query that displays the total and average payments of all the credit orders.
2. Write SQL query that displays the total and average payments grouped by type of payment.
3. How many order dates are represented compared to the total number of orders?
4. How many customers and sales representative are represented compared to the total number of orders?
5. Write SQL query that displays the lowest and highest payments of all the orders.
6. What is the average amount of the order for each sales representative?
7. Write an SQL query to display the order dates and how many orders were on each date.
8. Write SQL query to display the order amount by payment type for each sales representative
9. Query to display the highest and lowest order for each order date where more than one order was placed.
10. SQL query to display the average order for each order date where more than one order was placed and the average order is greater than 1000. Display them in order of average order.
11. Display the customer number with more than one order. Arrange alphabetically by customer id.

# Lab10 –SingleRowFunction

ThefollowingquerieswillbeexecutedusingDUALtableoremployeesdatabase.

1. Using the three separate words “Oracle,” “Internet,” and “Academy,” use one command toproduce thefollowingoutput:



1. Usethestring“Oracle InternetAcademy”toproducethefollowingoutput:



1. Whatisthelengthofthestring“Oracle Internet Academy”?
2. What’sthepositionof“I”in“OracleInternetAcademy”?
3. Startingwiththestring“OracleInternetAcademy”,padthe stringtocreate

\*\*\*\*Oracle\*\*\*\*Internet\*\*\*\*Academy\*\*\*\*

1. DisplayOracledatabaseemployeelast\_nameandsalaryforemployee\_idsbetween100and 102.Includeathirdcolumnthatdivideseachsalaryby1.55androundstheresulttotwodecimalplaces.
2. Display employee last\_name and salary for those employees who work in department 80.Giveeachof themaraiseof5.333%andtruncatethe resultto twodecimalplaces.

# Lab 11 – Implementation of Views & Indexing

Objective -Students will be able to implement the concept of View and Index Create a table Employee

1. Insert 5 records
2. Create a view having Ename and Ecity
3. Update the view& set the Ecity to ‘Delhi’ where Ename is ‘John’
4. Insert a row in the view.
5. Update the view and increase the salary of employees in IT dept
6. Create view having details of employee working in deptno=10
7. Create a view having grouping functions like max(sal)and min(sal)
8. Update the above view and set the max salary to 90000
9. Delete the view created

For the Employee table created above perform the following

1. Create an Index on LastName of Employee and allow duplicate value.
2. Create another Index on FirstName of Employee and disallow duplicate values.
3. Now ,create and Index named “idx\_FL” on the combination of columns ,LastName and FirstName.
4. Now drop the index “idx\_FL” created above.

# Lab 12 – Implementation of Procedures/Functions/Cursors/Triggers

**Theory**

PL/SQL supports the two type of programming:

* 1. Procedure.
  2. Function.

Procedures are usually used to perform any specific task and functionsare used to compute a value.

PROCEDURE:

The basic syntax for the creating procedure is:

CREATE OR REPLACE PRODURE procedure \_name (arguments) AS/ISprocedure body;

The body of procedure is block of statements with declarative executableand exception sections.

The declarative section is located between the IS /AS keyword andBEGIN keyword.

The executable section is located between BEGIN and EXCEPTION keywords or between the BEGIN and END keywords if there is no EXCEPTION handling section.

If EXCEPTION handling is present, it is located between exception andEND keywords.

***Steps for Creating Procedure:***

CREATE OR REPLACE PRODURE procedure \_name (parameter list)AS/IS(Declarative section )

BEGIN

(Executable section ) EXCEPTION

(Error handling or exception section).

END Procedure\_name;

To execute the procedure we have to write a block of statement: Begin

Procedurename(data); End;

**FUNCTION:**

**Steps for Creating function:**

CREATE OR REPLACE FUNCTION function \_name (parameter list)AS/IS Return datatype is/as

(local declaration)

BEGIN

(Executable section )

EXCEPTION

(Error handling or exception section ). END function\_name;

A function has two parts, namely function specification and function body. The function specification begins with the keyword function and end with return clause. The function bodies begins with the keyword is/as and end with keyword end

**THEORY**

* + 1. Cursor
    2. Triggers

There are two types of cursor

* + - 1. Explicit cursor
      2. Implicit cursor

1. **Explicit cursor:**

An explicit cursor is one in which cursor name is explicitly assigned to select statement. An implicit cursor is used for all other sql statements.Processing of an explicit cursor involves four steps. Processing of an implicit cursor is taken care by PL/SQL .the declaration of the cursor is done in the declarative part of the block.

A cursor variable is a reference type. A reference type is similar topointer.

***Explicit cursor:***

The set of rows return by query can contain zero or multiple rows depending upon the query defined. The rows are called active set. Thecursor will point to the current row in the active set.

After declaring a cursor, we can use the following commands to controlthe cursor.

* 1. Open
  2. Fetch
  3. Close

***Triggers:***

Types of triggers

1. before
2. after
3. for each row
4. for each statement.

***SYNTAX FOR TRIGGERS:***

CREATE OR REPLACE TRIGGER trigger\_nameBEFORE

/AFTER INSERT/UPADTE/DELETE ON

table\_name REFERENCING {OLD AS OLD/NEW AS NEW}

FOR EACH STATEMENT/ FOR EACH ROW when conditionPL/SQL\_BLOCK

/

Before/After options:

The before/after options can be used to specify when the trigger body should be fired with respect to the triggering statement. If the user include a before option, then Oracle fires the triggers before executing the triggering statement. On the other hand if AFTER is used then , oracle files the trigger after executing the triggering statement.

For each row/ statement:

For each row/ statement option included in the ‘create trigger ‘ syntax specifies that the triggers fires once per row . By defaults, database triggers a database triggers fires for each statement .

create or replace trigger orders before insert on order\_detail for

each row declare

or no order\_detail .orderno%type

begin

select orderno into orno from order\_detail where qty\_ord

<qyt\_deld;

if orno =’001’ then raise\_application\_error( 200001,’enter some other name’); end if; End;

/

In this trigger before inserting order table it check the condition orno(order number) equal to 001

and condition is satisfied then the value updated else error is occurred.

create or replace trigger up\_trig

before update order\_detail for each row

begin

update order\_detail set itemcode =’i200’ where qty\_ord=50; end;