# FACULTY OF COMPUTER SCIENCE
## Master of Computer Application (Sem-I)
In Effect from Academic Year 2020-21

| Subject Code : 2CS2010104 | Subject Title: DATABASE MANAGEMENT SYSTEM |
|---|---|
| Pre-requisite : | Basic knowledge of working with computers. |

**Course Objective:**
Introduces the student to the fundamental concepts necessary for designing, using and Implementing database systems and applications.

| Teaching Scheme (Hours per week) | | | | Evaluation Scheme (Marks) | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | Theory | | Practical | | |
| Lecture | Tutorial | Practical | Credit | University Assessment | Continuous Assessment | University Assessment | Continuous Assessment | Total |
| 4 | - | 3 | 7 | 60 | 40 | 30 | 20 | 150 |

| Subject Contents | | | |
|---|---|---|---|
| **Sr. No** | **Topic** | **Total Hours** | **Weight (%)** |
| **1** | **Introduction to Database System**<br>Basic Concepts : data, database, database systems, database management systems, instance, schema, Database Applications, Purpose and Advantages of Database Management System (over file systems), View of Data (Data Abstraction, Data Models), Database Languages (DML, DDL),Relational Databases (Tables, DML,DDL),Data Storage and Querying (Components, Storage Manager, Query Processor), Database Architecture , Database User and Administrators | 8 | 15 |
| **2** | **Entity Relationship Diagram**<br>Design Phases, Design Alternatives (Major Pitfalls), Entity Relational Model (Entity Sets, Relationship Sets, Attributes), Constraints (Mapping Cardinalities, Keys, Participation Constraints), Entity Relationship Diagram, Weak Entity Set, Extended E-R Features (Generalization, Specialization and Aggregation), E-R Notations, Examples of ERD<br>**Database Design**<br>Features of Good Relational Design, Atomic Domain and First Normal Form, Decomposition Using Functional Dependency (Key and Functional Dependency, BCNF,2NF, 3NF), Functional Decomposition Theory (Closure Set of Functional Dependency with Armstrong Rules, Canonical Cover and Lossless Decomposition), Dependency Preservation, Comparison of 3NF and BCNF, Decomposition Using Multi-Valued Dependencies (Multi-Valued Dependency and 4 NF) | 10 | 25 |
| **3** | **Relational Model**<br>Structure of Relational Databases (Basic Structure, Database Schema, Types of Keys), Fundamental Relational Algebra Operations (Select, Project, Union, Set Difference, Cartesian Product and Rename Operator), Additional Relational Algebra Operators (Set Intersection, Natural Join, Division Operator, Assignment Operator), Examples | 5 | 10 |
| **4** | **Structured Query Language (SQL):**<br>• Basic data types in SQL<br>• Creating and Managing Tables: CREATE TABLE and ALTER TABLE commands, INSERT, UPDATE and DELETE commands, Viewing data in the Tables, eliminating duplicate rows when using a select statement, Sorting data in a table, Creating a table from a table, | 15 | 30 |

| | | | |
|---|---|---|---|
| | Inserting data into a table from another table.<br>• Creating and Dropping Integrity Constraints: Primary key, Foreign key, Unique key, Not Null, Check<br>• Computations done on table data: Arithmetic operators, Logical operators, Range searching, Pattern matching<br>• Database Functions: Scalar and Group functions (Aggregate functions, Numeric functions, String functions), Conversion functions(To CHAR(), TO_DATE())<br>• Grouping and joining data from tables in SQL: GROUP BY Clause and HAVING Clause, Joins (Inner Join, Outer Join, Cross Join, and Self Join), Index, View, Sequence, Granting and Revoking Permissions. | | |
| **5** | **PL/SQL :**<br>Introduction, Control Structure, Cursor, Procedure, Function, Trigger, Exception Handling, Package. | 10 | 20 |

**Course Outcome:**
At the end of this course, the student would be able to

- Effective transformation of the real-world data into the relational data model of the Database system and data retrieval.
- Clear understanding for the need of a database.
- Ability to store information without unnecessary redundancy.

**List of References:**

1. Silberschatz, Korth, Sudarshan, “Database System Concepts”, 5th Edition, McGraw Hill Publication
2. “SQL,PL/SQL The programming language of oracle”, 3rd revised edition, Ivan Bayross, BPB Publication
3. C J Date, A Kannan, S Swaminathan, “An Introduction to Database Systems”, 8th Edition, Pearson Education (2006)
4. S K Singh, “Database Systems : Concepts, Design and Applications”, Pearson Education
5. Elmsari, Navathe, “Fundamentals of Database Systems”, 5th Edition, Pearson Education (2008)
6. Peter Rob, Carlos Coronel, “Database Systems : Design, Implementation and Management”, 7th Edition, Cengage Learning (2007)

# FACULTY OF COMPUTER SCIENCE

## Master of Computer Application (Sem-I)

In Effect from Academic Year 2020-21

**List of Experiments:**

**Note:** The experiment list provided beneath is for reference only. The course teacher may Change/formulate it as per his/her methodology and requirement.

**Sr.No** **Practical Exercise**

**SQL QUERIES (Based on DDL statement, constraints, DML statement, SELECT statement and Views)**

Note: In all schemas, create the table with necessary constraints(PK, FK, NotNull, Unique and Check constraints) on SQL prompt and then solve the given queries.

**1** **SQL Practical List:**

**CUST (custno, custname, addln1, addln2, city, state, phone)**
**ITEM (itemno, itemname, itemprice, qty_on_hand)**
**INVOICE (invno, invDate, custno)**
**INV_ITEM (invno, itemno, qty_used)**

1. Create the above four tables along with key constraints.
2. Write an Insert script for insertion of rows with substitution variables and insert Appropriate data.
3. Add a column – "colour" to the Item table.
4. Display the column Item name and Price in sentence form using concatenation.
5. Find the total value of each item(item price * qty).
6. Display the list of customers belonging to "Gujarat" state.
7. Display items with unit price between`100 and ` 500.
8. Find the customers from "Lalbaug" city of Ahmedabad and Baroda.
9. Find all the customers whose name starts with the letter 'P'.
10. Sort all customers alphabetically.
11. Sort all items in descending order by their prices.
12. Display invoice dates as per the format "January 16, 2012".
13. Find the total, average, highest and lowest unit price of an item.
14. Count number of items ordered in each invoice.
15. Find invoices in which three or more items have been ordered.
16. Display the details of items along with its quantity used (natural join).
17. Use outer join to display items ordered as well as items not ordered so far.
18. Find invoices with 'screw' in their itemname.
19. Display name of items ordered in invoice number 1001.
20. Find the items that are cheaper than item'Gear'.
21. Create a table(namely Gujarat cust)for all Gujarat customer based on existing customer table.
22. Copy all M.P customers to the table with Gujarat customers.
23. Rename Gujarat_cust table to MP_cust table.
24. Find the customers who are not in Gujarat or M.P.
25. Delete the rows from customer table that are also in MP_ cust table.
26. Find the first three items which has the highest price.
27. Create a read only view foritems having price less than ` 50.
28. Create a sequence and use that sequence in insert statement which can be used to enter new items into item table.

**STUDENT (rollno, name, class, birthdate)**
**COURSE (courseno, coursename, max_marks, pass_marks)**

**2** **SC (rollno, courseno, marks)**

1. Create the above three tables along with key constraints.
2. Write an Insert script for insertion of rows with substitution variables and insert

appropriate data.

3. Add a constraint that the marks entered should strictly be between 0 and 100.
4. WhilecreatingSCtable,compositekeyconstraintwasforgotten.Addthecomposite key now.
5. Display details of student who takes'Database Management System' course.
6. Displaythenamesofstudentswhohavescoredmorethan70%inComputer Networks and have not failed in any subject.
7. Display the average marks obtained by each student.
8. Select all courses where passing marks are more than 30% of average maximum mark.
9. Display details of students who are born in 1980 or 1982.
10. Create a view that displays student courseno and its corresponding marks.

**3. HOSTEL (hno, hname, haddress, total_capacity, warden_nm)**
**ROOM (hno, rno, rtype, location, no_of_students, status)**
**CHARGES (hno, rtype, charges)**
**STUDENT (sid, sname, saddr, faculty, dept, class, hno, rno)**
**FEES (sid, fdate, famount)**

The STATUS field tells us whether the room is occupied or vacant. The charges represent the term fees to be paid half yearly. A student can pay either the annual fees at one time or the half yearly fees twice a year.

1. Create the above five tables along with key constraints.
2. Write an Insert script for insertion of rows with substitution variables and insert appropriatedata.
3. Addacheckconstrainttotheroomtablesothattheroomtypeallowsthefollowing: values only – 's' for single, 'd' for double,'t' for triple and'f' for four-seater.
4. Display the total number of rooms that are presently vacant.
5. Display number of students who are staying in 'double' seated roomfor each hostel.
6. Display the warden name and hostel address of students of 'ComputerScience' department.
7. Display the hostel details where single seated or four-seated rooms are vacant.
8. Count total number of 'medical' students who live in the hostel.
9. Display hostels, which are totally occupied to its fullest capacity.
10. List details about students who are staying in the double-seated rooms of "Chanakya" Hostel.
11. Display the total number of students staying in each room type of each hostel.
12. Display details about students who have paid the fees in the month of November 2011.
13. For those hostels where total capacity is more than 300,display details of students studying in the Science faculty.
14. Display the hostel details where there are at least 10 vacant rooms.
15. Display the details of students who have still not paid their hostel fees.
16. Display those hostels where single-seated room is the costliest.

**4. SCREEN (screen_id, location, seating_capacity)**
**MOVIE (movie_id, movie_name, date_of_release)**
**CURRENT (screen_id, movie_id, date_of_arrival, date_of_closure)**

Value of screen_id must start with letters 'S'.
Attribute location can be any one of 'FF','SF',or'TF'(Firstfloor,secondfloororthird floor). Date_of_arrival must be less than the date_of_closure.
Solve the following queries based on the above schema:

1. Extract the name ofmovie which has run the longest in the multiplex so far.

2. Find the average duration of a movie on screen number 'S4'.
3. Get the details of the movie that closed on date 24-December-2011.
4. Movie"Body gaurd"was released in the 35th week of 2011. Find out the date of its release considering that a movie releases only on Friday.
5. Get the full outer join ofthe relations screen andcurrent.

**5. DISTRIBUTOR (dno, dname, daddress, dphone)**
**ITEM (itemno, itemname, colour, weight)**
**DIST_ITEM (dno, itemno, qty)**

1. Add a column CONTACT_PERSON to the DISTRIBUTOR table with the not null constraint.
2. Create a view LONDON_DIST on DIST_ITEM which contains only those records where distributors are from London. Make sure that this condition is checked for every DML against this view.
3. Display the details of all those items that have never been supplied.
4. Delete all those items that have been supplied only once.
5. List the names of distributors who have an 'A' and also a 'B' somewhere in their names.
6. Count the number of items having the same colour but not having weight between 20 And 100.
7. Display all those distributors who have supplied more than 1000 parts of the same type.
8. Count the number of distributors who have a phone connection and are supplying item number 'I100'.
9. Create a view on the tables in such a way that the view contains the distributor name, item name and the quantity supplied.
10. List the name, address and phone number of distributors who have the same three digits in their number as 'Mr. Talkative'.
11. List all distributor names who supply either item I1 or I7 or the quantity supplied is more than 100.
12. Display the data of the top three heaviest ITEMS.

**6. WORKER (worker_id, name, wage_per_hour, specialised_in, manager_id)**
**JOB (job_id, type_of_job, status)**
**JOB_ASSIGNED (worker_id, job_id, starting_date, number_of_days)**

1. Display the date on which each worker is going to end his presently assigned job.
2. Display how many days remain for each worker to finish his job.
3. Display the STARTING_DATE in the following format – 'The fifth day of the month of October, 2011'.
4. Change the status to 'Complete' for all those jobs, which started in year 2010.
5. Display details of all those jobs where at least 25 workers are working.
6. Display all those jobs that have already been completed.
7. Find all the jobs, which will begin within the next two weeks.
8. List all workers who have their wage per hour ten times greater than the wage of their managers.
9. List the names of workers who have been assigned the job of molding.
10. What is the total number of days allocated for packaging the goods for all the workers together.
11. Which workers receive higher than average wage per hour.
12. Display details of workers who are working on more than one job.
13. List the workers having specialization in "Polishing" and have started their job in December 2011.
14. Display details of workers who are specialized in the same field as that of Mr. Cacophonix or have a wage per hour more than that any of the workers.

**7. PUBLISHER (publ_id, publ_name, contact_person, contact_addr, contact_phone)**
**CATEGORY (cat_id, cat_details, max_books, duration)**

**BOOK_MASTER (book_id, bname, isbn_no, total_copies, publ_id)**
**MEMBER (member_id, mname, cat_id, mem_ship_dt)**
**ISSUE (ISSUE_id, member_id, book_id, issu_ret, issue_ret_dt)**

In the above tables, duration is in years and itstores the membership duration for that category.

1. Change the table design of ISSUE table to add aconstraint, which will allow only 'I' or 'R' to be entered in the ISSUE_RETcolumn, which stores the action whether the book is being issued or returned.
2. Add a column to the MEMBER table,which will allow us to store the address of the member.
3. Create a table LIBRARY_USERS which has a structure similar to that of the MEMBER table but with no records.
4. Give details about members who have issued books, which contains the word 'DATA' somewhere in their titles.
5. Display the books that have been issued at the most three times in the year 2011.
6. Display the details of books issued right now that is published by "Pearson".
7. Display the details of books whose all copies are issued.
8. Display the details of the books that have been issued between 1st December 2011 and 31st December2011.The result should also contain the details of the members to Whom those books have been issued.
9. Display details of all the staff members who have issued at least two books.
10. Display the details about those publishers whose books are available in more than100 titles in the library.
11. Delete the details of all those members whose membership has expired.
12. List the details of members who have registered with the library in the last three months.
13. Display the membership period of each staff member registered.

**8. APPLICANT (aid, aname, addr, abirth_dt)**
**ENTRANCE_TEST (etid, etname, max_score, cut_score)**
**ETEST_CENTRE (etcid, location, incharge, capacity)**
**ETEST_DETAILS (aid, etid, etcid, etest_dt, score)**

This database is for a common entrance test which can be conducted at a number of centers and can be taken by an applicant on any day except holidays.

1. ModifytheAPPLICANTtablesothateveryapplicantidhasan'A'beforeitsvalue.For example, if the applicant id '1123', it should now become 'A1123'.
2. Display the test center details where no tests will be conducted.
3. Display the details about applicants who have the same score as that of "Jaydev" in "Oracle Fundamentals".
4. Display the details of applicants who have appeared for all the tests.
5. Display those tests where no applicant has failed.
6. Display details of entrance test centers which had full attendance between 1 November and 15th November 2011.
7. Display the details of those applicants who have scored more than the cut-off score in the tests they have appeared in.
8. Display the average and the maximum score, test wise of the tests conducted at Mumbai.
9. Display the number of applicants who have appeared for each test, test center wise.
10. Display the details about test centers where no tests have been conducted.
11. For tests, which have been conducted between 2-8-2011 and 23-9-2011, show details of the tests as well as the test centers.
12. List the number of applicants who had appeared in the "Oracle Fundamentals" test at Chennai in the month of September 2011.

13. Display the details about applicants who appeared for tests in the same month as the month in which they were born.
14. Display the details about APPLICANTS who have scored the highest in each test,test center wise.
15. Design a read only view, giving details about applicants and the tests that they have appeared for.

**PL/SQL**

**1** **PL/SQL Practical List:**
**Competition (Comp_code, Comp_name (Dancing, Painting, GK, etc.))**
**Participants (Part_no, Part_name, DOB, Address, EmailID, Contact_number )**
**Scorecard (Part_no, Comp_code, Judge_no [1, 2, 3], Marks)Implement the Following:**

Create a PL/ SQL block to prepare report in following format.
Display the score card in the following format, for the Participant whose ID/ Name should be provided by the user.

**Talent Winner 2011 ::: <Participant's Name>**

| Competitionname | Judge1 | Judge2 | Judge3 |
|---|---|---|---|
| **1. Painting** | | | |
| **2. Dancing** | | | |
| **Total Marks:** | | | |

**2** **Customer (Cust_Id, Cust_Name, Cust_Addr, Cust_City, EmailID,Contact_No)**
**Magazine (Mag_Id, Mag_Name, Unit_Rate, Type_of_subsciption[weekly, monthly, etc.])**
**Subscription (Cust_Id, Mag_Id, start_date, end_date)**

**Implement the following:**

**A)**

1.Create a View that displays Customer name, Magazine name along with its rate which was subscribed during 01-Sept-2010 to 01-Feb-2011.

2. Find top three magazines having the highest sale during last one month of time.

**B)**

1. Create a function to return No. of customers in city Gandhinagar who have subscribed the magazine 'Outlook' after August 2010. If no such customer exists, throw a user defined exception with appropriate message.

2 Create a trigger that is fired after an INSERT statement is executed for the Customer table. The trigger writes the new customer's code, name and the sysdate in a table called Customer_Log. ( create the table Customer_Log)

**3** **Account (ac_no, ac_name, act_type)**
**Transaction (ac_no, trans_date, tran_type, tran_amount, balance)**
**Note: Act_type may be 'S' for saving or 'C' for current and tran_type may be 'D' for deposit or 'W' for withdrawal.**
**Implement the following:**

**A).**

1. Find out those saving transactions that took place between 10th January201120th January 2011 and have withdrawn an amount greater than Rs. 50,000.

2.Create a Sequence that can be used to enter new account number into the account table. Add a new record into Account table using the created sequence.

**B)**

1. Create a trigger not allowing insertion, deletion or updation on Saturday and before 8:00 AM & after 6:00 PM on Account table.
2. Create a package for the following :
   Create a function to return the current balance for a given account number.

**4** **Supplier (sid, sname, contactnum)**
**Parts (pid, pname, color, unit rate)**
**Catalog (sid, pid, qty)**
**Implement the following:**
**A)**

1. Find the top three Parts been ordered and have the highest sale till date.
2. Find those suppliers who charge more for some part that the average cost of that part.

**B)**

**Create a PL/ SQL block to prepare invoice in following format. Display the invoice in the following format. Use parameterized cursor.**

**Invoice ::: <Supplier's Name>**

| **Part Id** | **Part Name** | **Quantity** | **Unit Price** | **Total Price** |
|---|---|---|---|---|

---

**Total: _____**

---

**5** **Sailor (sid, sname, rating (0-10), DOB)**
**Boat (bid, bname, color)**
**Reserve (sid, bid, date)**
**Implement the following:**
**A)**

1. Find the sailor(s) whose birthday fall in a leap year.
2. Find the name of the sailor who has reserved either the red or green colored boat.

**B)**

1. Create a parameterized cursor to display the sailor details who have reserved any boat after November 2010. If no record found, throw an user defined exception with appropriate message.
2. Create a function that get the Boat code from the user. Display the sailor_code who have reserved this boat code. Raise an exception if no information for boat/sailor exists.

**6** **Movie (movie_id, movie_name, date_of_release)**
**Screen (screen_id, location, max_capacity)**
**Current (movie_id,screen_id, date_of_arrival, date_of_closure)**
**Note:**
**Value of screen_id must with letter 'S'.**
**Screen location can by any one of 'FF', 'SF', and 'TF'.**
**Date_of_arrival must be less than Date_of_closure.**
**Max_capacity attribute should have a value greater than 0.**
**Implement the following:**
**A)**

1. Find the top three movies which have the highest screened record.
2. Create a View which displays the movie details along with the information about the screen on which it is currently screened.

**B)**

1. Create a trigger that is fired after an INSERT statement is executed for the Movie table. The trigger writes the new movie's code, movie name and the sysdate in a table called Movie_Log.(create the table Movie_Log)
2. Create a function that get the Screen Code from the user and displays the movie name currently screened on it. If the given screen code does not exist, throw a user defined exception with appropriate message.

**7** **Employee_master(EmpCode , Emp_Name , Dept_Id, Emp_Address , DOB , Basic_Salary)**
**Department_master(Dept_Code ,Dept_Name)**

**Implement the following:**

**A)**

1. Create a View that displays some Employee details such as Employee code, Employee name, Department Name and their Basic Salary.
2. Find those employees who do not belong to Department D102 or D105. (Note: Use set operator)

**B)**

**Create a PL/ SQL block to prepare report in the following format:**
**Display the salary slip for the employee in the following format, whose Employee Code is provided by the user.**

**Salary Slip for the month March 2012.**

Employee Code: <E102> Employee Name: <JohnSmith>

Department Name: <Finance>

_____

**Basic Salary** **DA** **HRA** **Medical** **P.F.**

Deductions:

---

Total Salary : _____

**Note:**

HRA is 15% of basic salary
DA is 30% of basic salary
Medical is 1% of basic salary
P.F. is 10% of basic salary

**8** **Competition (Comp_code, Comp_name (Dancing, Painting, GK, etc. Participants (Part_no, Part_name, DOB, Address, EmailID, Contact_number ) Scorecard (Part_no, Comp_code, Judge_no [1, 2, 3], Marks)**

**Implement the following: A)**

1. Create a sequence that allows entering new 'Competition Code' that must start with 'CMP', whenever an insertion is tried to be done.
2. Find the event names which have scored the maximum score by the each judge in total.

**B)**

1. Create a parameterized cursor to display the total score scored by each student with the competition details, the competition event name have to be supplied as the parameter. If

the given event does not exist, throw a user defined exception with appropriate message.

2. Create a trigger that checks the 'Competition Code' must start with 'CMP' whenever an insertion is tried to be done. Raise an user defined exception if the rule is violated.

9 **Customer (Cust_Id, Cust_Name, Cust_Addr, Cust_City, EmailID,Contact_No)**
**Magazine (Mag_Id, Mag_Name, Unit_Rate, Type_of_subsciption[weekly, monthly, etc.])**
**Subscription (Cust_Id, Mag_Id, start_date, end_date)**
**Implement the following: A)**

1. Find those customers who haven't subscribed 'PCQuest' or 'Chip India'. (Using set operator).
2. Find top two magazines having the highest sale during last one month.

**B)**

1. Create a package for the following :
   Create a function to return No. of customers in city 'Ahmedabad' who have subscribed the magazine 'PCQuest' after July 2010. If no such customer exists, throw a user defined exception with appropriate message.
2. Create a function, which accept the Magazine Code and return the Magazine's name and its rate. If the magazine code does not exist, throw a user defined exception with appropriate message.

10 **Account (ac_no, ac_name, act_type)**
**Transaction (ac_no, trans_date, tran_type, tran_amount, balance)**

**Note: Act_type may be 'S' for saving or 'C' for current and tran_type may be 'D' for deposit or 'W' for withdrawal.**

**Implement the following:**

**A)**

1. Find out those saving transactions that took place between 10th January 2012 and 20th January 2012 and have withdrawn an amount greater than ` 50,000.
2. Create a View that display the account information having a balance greater than ` 1,00,000.

**B)**

1. Create a trigger not allowing insertion, deletion or updation on Saturday and before 8:00 AM & after 6:00 PM on Account table.
2. After every 6 months all the customers are given 5% interest. So for current date, give interest of 6% to all the customers whose balance are greater than or equal to 2000 and interest of, on their balance.

11 **Sailor (sid, sname, rating (0-10), DOB)**
**Boat (bid, bname, color)**
**Reserve (sid, bid, date)**
**Implement the following: A)**

1. Find the name of the sailor who has not reserved the red colored boat.
2. Find the name of the sailor who is youngest among all.

**B)**

3. Create a trigger that checks the 'Boat Code' must start with 'B' whenever an insertion is tried to be done. Raise a user defined exception if the rule is violated.
4. Create a procedure that get the Sailor Code from the user and check whether that Sailor was born in a leap year or not. If the given sailor code does not exist, throw a user defined exception with appropriate message.

**12** **Book_catalog (book_code, title, Publisher_Name, Category_Name, yr_of_release, total_copies )**
**Member (member_code, member_name,mem_ship_dt)**
**Issue (Issue_id, member_code, book_code, issu_ret, issue_date, issue_ret_dt)**
**Note:**
**Add a constraint to Issue table, which will allow only 'I' or 'R' to be entered in the ISSUE_RET column, which stores the action whether the book is being issued or returned. Implement the following:** A)

1. Find the book details which are currently issued to the members and have crossed the return date, get details starting with the current date.
2. How many members have registered in the last three months ? Display their details.

B)

1. Create a function which provides the total number of copies available for the issue for a given book. Book Code to be provided by the user.
2. Create a package for the following:
   Create a function to print the book title when Book code is been supplied by the user.

**13** **Item_master(Item_Cd, Item_Name, Item_Price)**
**Item_received( Item_Cd, Month, Year, Day, Rec_Qty)**
**Item_stock(Item_Cd, Month,Year, Open_Stock, Rec_Qty, Close_Stock)**
**Implement the following:**

**A)**

1. Create a sequence that can be used to enter new items into item table.
2. List items whose range lies between ` 250 and ` 500.

**B)**

1. Write triggers that affect Item_stock table for the insert, update and delete on Item_received table.
2. Write a procedure to accept Item Name as input. If it exists, then display the Item Price otherwise display the proper message through the use of exception.

**14** **Employee Master (Emp_Code, Emp_Name, Birth_Date)**
**Department Master (Dept_Code, Dept_Name, Budget)**
**Salary (Dept_Code, Emp_Code, Salary)**
**Implement the following:**

**A)**

1. Count the number of employee in each department.
2. Create a view to display employee name & its salary.

**B)**

1. Write a trigger before insert new row into salary table for constraint "Total salary for department is not exceeding the budget".
2. Write a function which accept the Employee Name as an argument and return the salary of that employee. If employee name does not exists than raise the exception.