1. **Consider the following relational schema:**

STAFF (**STAFFNO**, NAME, DOB, GENDER, DOJ, DESIGNATION, BASIC_PAY, DEPTNO)

GENDER must take the Value 'M' or 'F'

DEPT (**DEPTNO, NAME**)

SKILL (**SKILL_CODE**, DESCRIPTION, CHARGE_OUTRATE)

STAFF_SKILL (**STAFFNO , SKILL_CODE**)

PROJECT (**PROJECTNO**, PNAME, START_DATE, END_DATE, BUDGET, PROJECT_MANAGER_STAFFNO)

WORKS (**STAFFNO, PROJECTNO, DATE WORKED ON**, IN_TIME, OUT_TIME)

The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema specifying appropriate data types for each attribute and enforcing primary key, check constraint and foreign key constraints. (25)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the department number and number of staff in each department. (5)

d. Develop a SQL query to list the details of staff who earn less than the average basic pay of all staff. (5)

e. Develop a SQL query to list the details of departments which has more than five staff working in it. (5)

f. Develop a SQL query to list the details of staff who have more than three skills. (5)

g. Develop a SQL query to list the details of staff who have skills with a charge out rate greater than 60 per hour. (5)

h. Create a view that will keep track of the department number, the department name, the number of employees in the department, and the total basic pay expenditure for each department. (10)

i. Develop a database trigger that will not permit a staff to work on more than three projects on a day. (15)

j. Develop a procedure INCR that will accept staff number and increment amount as input and update the basic pay of the staff in the staff table. Include exception in the procedure that will display a message 'Staff has basic pay null' if the basic pay of the staff is null and display a message 'No such staff number' if the staff number does not exist in the staff table. (15)

******

1. **Consider the following relations for an order-processing database application in a company:**

CUSTOMER (**CUSTOMERNO** VARCHAR2 (5), CNAME VARCHAR2 (30), CITY VARCHAR2 (30))

Implement a check constraint to check CUSTOMERNO starts with 'C'

CUST_ORDER (**ORDERNO** VARCHAR2 (5), ODATE DATE, CUSTOMERNO REFERENCES CUSTOMER, ORD_AMT NUMBER (8))

Implement a check constraint to check ORDERNO starts with 'O'.

ITEM (**ITEMNO** VARCHAR2 (5), ITEM_NAME VARCHAR2 (30), UNIT_PRICE NUMBER (5))

Implement a check constraint to check ITEMNO starts with 'I'

ORDER_ITEM (**ORDERNO** REFERENCES CUST_ORDER, **ITEMNO** REFERENCES ITEM, QTY NUMBER (3))

SHIPMENT (**ORDERNO** REFERENCES CUST_ORDER, **ITEMNO** REFERENCES ITEM, SHIP_DATE DATE)

Here, ORD_AMT refers to total amount of an order (ORD_AMT is a derived attribute); ODATE is the date the order was placed; SHIP_DATE is the date an order is shipped. The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema enforcing primary key, check constraints and foreign key constraints. (25)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the details of customers who have placed more than three orders. (5)

d. Develop a SQL query to list the details of items whose price is less than the average price of all items. (5)

e. Develop a SQL query to list the orderno and number of items in each order. (5)

f. Develop a SQL query to list the details of items that are present in 25% of the orders. (5)

g. Develop an update statement to update the value of ORD_AMT. (10)

h. Create a view that will keep track of the details of each customer and the number of orders placed. (10)

i. Develop a database trigger that will not permit to insert more than six records in the CUST_ORDER table for a particular order. (An order can contain a maximum of six items). (15)

j. Develop a procedure DISP that will accept an order number and display the order details from CUST_ORDER. Include exception in the procedure that will display a message 'No such order' if the order number does not exist in the CUST_ORDER relation. (10)

---

1. **Consider the following relational schema for a company database application:**

EMPLOYEE (**ENO**, NAME, GENDER, DOB, DOJ, DESIGNATION, BASIC, DEPT_NO, PAN, SENO)

Implement a Check Constraint for GENDER

PAN – Permanent account Number

SENO – Supervisor Employee Number

DEPARTMENT (**DEPT_NO**, NAME, MENO)

MENO - Manager Employee Number

PROJECT (**PROJ_NO**, NAME, DEPT_NO)

WORKSFOR (**ENO, PROJ_NO, DATE_WORKED, HOURS**)

Each department has a manager managing it. There are also supervisors in each department who supervise a set of employees. A department can control any number of projects. But only one department can control a project. An employee can work on any number of distinct projects on a given day. The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema specifying appropriate data types for each attribute enforcing primary key, check constraints and foreign key constraints. (25)

b. Populate the database with a rich data set. (10)

c. Develop an SQL query to list the departments and the details of manager in each department. (5)

d. Develop an SQL query to list details of all employees and the details of their supervisors. (5)

e. Develop an SQL query to list the department number, department name and the number of employees in each department. (5)

f. Develop a SQL query to list the details of employees who earn less than the average basic pay of all employees. (5)

g. Develop a SQL query to list the details of departments which has more than six employees working in it. (5)

h. Create a view that will keep track of the department number, department name, number of employees in the department, and total basic pay expenditure for each department. (10)

i. Develop a database trigger that will not permit an employee to work on more than three projects on a day. (15)

j. Develop a procedure INCR that will accept employee number and increment amount as input and update the basic pay of the employee in the employee table. Include exception in the procedure that will display a message 'Employee has basic pay null' if the basic pay of the employee is null and display a message 'No such employee number' if the employee number does not exist in the employee table. (15)

*****

1. **Consider the following relational schema for a company database application:**

EMPLOYEE (**ENO**, NAME, GENDER, DOB, DOJ, DESIGNATION, BASIC, DEPT_NO, PANNO, SENO)

Implement a Check Constraint for GENDER

PAN – Permanent account Number

SENO – Supervisor Employee Number

DEPARTMENT (**DEPT_NO**, NAME, MENO, NOE)

MENO - Manager Employee Number

NOE - Number of Employees

The default value of NOE is 0

PROJECT (**PROJ_NO**, NAME, DEPT_NO)

WORKSFOR (**ENO, PROJ_NO, DATE_WORKED,** HOURS)

Each department has a manager managing it. There are also supervisors in each department who supervise a set of employees. A department can control any number of projects. But only one department can control a project. An employee can work on any number of distinct projects on a given day. The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema specifying appropriate data types for each attribute enforcing primary key, check constraints and foreign key constraints. (25)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the details of employees who earn less than the average basic pay of all employees. (5)

d. Develop a SQL query to list the details of departments which has more than six employees working in it. (5)

e. Create a view that will keep track of the department number, department name, number of employees in the department, and total basic pay expenditure for each department. (10)

f. Develop a database trigger which will increment the NOE attribute in DEPARTMENT relation by 1 when a new record is

inserted in the employee relation. (This situation occurs when a new employee is appointed) (10)

g. Develop a database trigger which will decrement the NOE attribute in DEPARTMENT relation when a tuple in the employee relation is deleted. (This situation occurs when an employee working in a department quits his / her job) (10)

h. Develop a database trigger that will fire when an updation is performed on the DEPT_NO attribute of the EMPLOYEE relation. The NOE attribute value of the department in which the employee is currently working must be decremented by 1 and the NOE attribute value of the department to which an employee is transferred to must be incremented by 1. (This situation occurs when an employee working in one department is transferred to another department) (10)

i. Develop a procedure INCR that will accept employee number and increment amount as input and update the basic pay of the employee in the employee table. Include exception in the procedure that will display a message 'Employee has basic pay null' if the basic pay of the employee is null and display a message 'No such employee number' if the employee number does not exist in the employee table. (15)

****

1. **Consider the following relational schema for a banking database application:**

CUSTOMER (**CID**, CNAME)

ACCOUNT (**ANO**, ATYPE, BALANCE, CID)

An account can be a savings account or a current account. Check ATYPE in 'S' or 'C'. A customer can have both types of accounts.

TRANSACTION (**TID**, ANO, TTYPE, TDATE, TAMOUNT)

TTYPE CAN BE 'D' OR 'W'

D- Deposit; W – Withdrawal

The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema specifying appropriate data types for each attribute enforcing primary key, check constraints and foreign key constraints. (20)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the details of customers who have a savings account and a current account. (5)

d. Develop a SQL query to list the details of customers who have balance less than the average balance of all customers. (5)

e. Develop a SQL query to list the details of customers with the sum of balance in their account (s) (5)

f. Develop a SQL query to list the details of customers who have performed three transactions on a day. (5)

g. Create a view that will keep track of customer details and the number of accounts each customer has. (10)

h. Develop a database trigger that will not permit a customer to perform more than three transactions on a day. (15)

i. Develop a database procedure that will accept transaction id, account number, transaction type as input and insert a record into TRANSACTION table subject to the following conditions:

   i. If TTYPE ='D' the value of BALANCE in the ACCOUNT table must be incremented by the value of TAMOUNT (10)

ii. If TTYPE ='W' the value of BALANCE in the ACCOUNT table must be decremented by the value of TAMOUNT if a minimum balance of Rs. 2000/- will be maintained for a savings account and a minimum balance of Rs. 5000/- will be maintained for a current account else appropriate messages may be displayed. (15)

****

1. **Consider the following relational schema for the office of the controller of examinations application:**

STUDENT (**ROLLNO**, NAME, DOB, GENDER, DOA, BCODE)

Implement a check constraint for GENDER

DOA – Date of admission

BRANCH (**BCODE**, BNAME, DNO)

DEPARTMENT (**DNO**, DNAME)

COURSE (**CCODE**, CNAME, CREDITS, DNO)

BRANCH_COURSE (**BCODE, CCODE**, SEMESTER)

ENROLLS (**ROLLNO, CCODE, SESS**, GRADE)

For Example: SESS can take the values APRIL2013, NOVEMBER2013

Implement a check constraint for GRADE

VALUE SET ('S', 'A', 'B', 'C', 'D', 'E', 'U')

Students are admitted to branches. Branches are offered by departments. A branch is offered only by one department. Each branch has a set of courses (subjects) each student must enroll during a semester. Courses are offered by departments. A course is offered only by one department. If a student is unsuccessful in a course he / she must enroll for the course during the next session. A student has successfully completed a course if the grade obtained is from the value set ('S', 'A', 'B', 'C', 'D', 'E'). A student is unsuccessful if he / she have obtained a 'U' grade in a course.

The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema specifying appropriate data types for each attribute enforcing primary key, check constraints and foreign key constraints. (25)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the details of departments that offer more than three branches. (5)

d. Develop a SQL query to list the details of departments that offer more than six courses. (5)

e. Develop a SQL query to list the details of courses that are common for more than three branches. (5)

f. Develop a SQL query to list the details of students who have got 'S' grade in more than two courses during a single enrollment. (5)

g. Create a view that will keep track of the roll number, name and number of courses a student has successfully completed. (5)

h. Develop a database trigger that will not permit a student to enroll for more than six courses during a session. (15)

1. **Consider the following relational schema for a banking database application:**

CUSTOMER (**CID**, CNAME)

BRANCH (**BCODE**, BNAME)

ACCOUNT (**ANO**, ATYPE, BALANCE, CID, BCODE)

An account can be a savings account or a current account. Check ATYPE in 'S' or 'C'. A customer can have both types of accounts.

TRANSACTION (**TID**, ANO, TTYPE, TDATE, TAMOUNT)

TTYPE CAN BE 'D' OR 'W'

D- Deposit; W – Withdrawal

The primary keys are underlined.

a. Develop DDL to implement the above Schema specifying appropriate data types for each attribute enforcing primary key, check constraints and foreign key constraints. (20)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the details of customers who have a savings account and a current account. (5)

d. Develop a SQL query to list the details of branches and the number of accounts in each branch. (5)

e. Develop a SQL query to list the details of branches where the number of accounts is less than the average number of accounts in all branches. (5)

f. Develop a SQL query to list the details of customers who have performed three transactions on a day. (5)

g. Create a view that will keep track of branch details and the number of accounts in each branch. (10)

h. Develop a database trigger that will not permit a customer to perform more than three transactions on a day. (15)

i. Develop a database trigger that will update the value of BALANCE in ACCOUNT table when a record is inserted in the transaction table. Consider the following cases:

    i. If TTYPE ='D' the value of BALANCE in the ACCOUNT table must be incremented by the value of TAMOUNT (10)

ii. If TTYPE ='W' the value of BALANCE in the ACCOUNT table must be decremented by the value of TAMOUNT if a minimum balance of Rs. 2000/- will be maintained for a savings account and a minimum balance of Rs. 5000/- will be maintained for a current account else appropriate messages may be displayed. (15)

****

1. Consider the following relational schema for an order processing application:

CUSTOMER (**CUSTOMER_ID** VARCHAR2 (5) PRIMARY KEY, CUSTOMER_NAME VARCHAR2 (30))

CHECK CUSTOMER_ID STARTS WITH 'C'

PRODUCT (**PRODUCT_CODE** VARCHAR2 (5) PRIMARY KEY, PRODUCT_NAME VARCHAR2 (30), UNIT_PRICE NUMBER (5))

CHECK PRODUCT_CODE STARTS WITH 'P'

CUST_ORDER (**ORDER_CODE** VARCHAR2 (5), ORDER_DATE DATE, ORDER_AMT NUMBER (8), CUSTOMER_ID REFERENCES CUSTOMER)

CHECK ORDER_CODE STARTS WITH 'O'

ORDER_AMT IS A DERIVED ATTRIBUTE

ORDER_PRODUCT (**ORDER_CODE** REFERENCES CUST_ORDER, **PRODUCT_CODE** REFERENCES PRODUCT, NO_OF_UNITS NUMBER (3), PRIMARY KEY (ORDER_CODE, PRODUCT_CODE))

The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema enforcing primary key, check constraints and foreign key constraints. (25)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the details of the product whose price is greater than average price of all products. (5)

d. Develop a SQL query to list the order code and the no of products in each order. (5)

e. Develop a SQL query to list the details of the products which is contained in more than 30% of the orders. (5)

f. Develop an update statement to update the ORDER_AMT in CUST_ORDER table. (10)

g. Develop a SQL query to list the details of the customers have placed more than 5% of the orders. (5)

h. Develop a SQL query to list the details of the products which is contained in less than 10% of the orders. (5)

i. Create a view that will keep track of details of customers and the number of orders placed by each customer. (5)

j. Develop a database trigger that will not permit to insert more than six records in the CUST_ORDER table for a particular order. (An order can contain a maximum of six items). (15)

k. Develop a procedure DISP that will accept an order number and display the order details from CUST_ORDER. Include exception in the procedure that will display a message 'No such order' if the order number does not exist in the CUST_ORDER relation. (10)

******

1. **Consider the following relational schema for the office of the controller of examinations application:**

STUDENT (**ROLLNO**, NAME, DOB, GENDER, DOA, BCODE)

Implement a check constraint for GENDER

DOA – Date of admission

BRANCH (**BCODE**, BNAME, DNO)

DEPARTMENT (**DNO**, DNAME)

COURSE (**CCODE**, CNAME, CREDITS, DNO)

BRANCH_COURSE (**BCODE, CCODE**, SEMESTER)

PREREQUISITE_COURSE (**CCODE**, **PCCODE**)

A course can have prerequisite courses. For example Database Management Systems is a prerequisite course for Advanced Databases.

ENROLLS (**ROLLNO, CCODE, SESS**, GRADE)

For Example: SESS can take the values APRIL2011, NOVEMBER2011

Implement a check constraint for GRADE

VALUE SET ('S', 'A', 'B', 'C', 'D', 'E', 'U')

For a student to enroll for a course he / she should have completed the prerequisite courses.

Students are admitted to branches. Branches are offered by departments. A branch is offered only by one department. Each branch has a set of courses (subjects) each student must enroll during a semester. Courses are offered by departments. A course is offered only by one department. If a student is unsuccessful in a course he / she must enroll for the course during the next session. A student has successfully completed a course if a the grade student obtained is from the value set ('S', 'A', 'B', 'C', 'D', 'E'). A student is unsuccessful if he / she have obtained a 'U' grade in a course.

The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema specifying appropriate data types for each attribute enforcing primary key, check constraints and foreign key constraints. (25)

b. Populate the database with a rich data set. (10)

c. Develop a SQL query to list the details of departments that offer more than three branches. (5)

d. Develop a SQL query to list the details of courses that do not have prerequisite courses. (5)

e. Develop a SQL query to list the details of courses that are common for more than three branches. (5)

f. Develop a SQL query to list the details of students who have got a 'U' grade in more than two courses during a single enrollment. (5)

g. Create a view that will keep track of the course code, name and number of prerequisite courses. (5)

h. Develop a database trigger that will not permit a student to enroll for a course if he / she have not completed the prerequisite courses. (15)

i. Develop a procedure DISP that will accept a ROLLNO of a student as input and print the roll number, name and number of courses a student has successfully completed. (15)

j. Develop a procedure DISP_NOE that will accept a CCODE of a COURSE as input and print the roll number, name of students who have enrolled for the course more than twice. (10)

****

1. **Consider the following relational schema for an order processing application:**

CUSTOMER (**CUSTOMERNO** VARCHAR2 (5), CNAME VARCHAR2 (30), CCODE REFERENCES CITY)

Implement a check constraint to check CUSTOMERNO starts with 'C'

CUST_ORDER (**ORDERNO** VARCHAR2 (5), ODATE DATE, CUSTOMERNO REFERENCES CUSTOMER, ORD_AMT NUMBER (8))

Here, ORD_AMT refers to total amount of an order (ORD_AMT is a derived attribute); ODATE is the date the order was placed.

Implement a check constraint to check ORDERNO starts with 'O'

ITEM (**ITEMNO** VARCHAR2 (5), ITEM_NAME VARCHAR2 (30), UNIT_PRICE NUMBER (5))

Implement a check constraint to check ITEMNO starts with 'I'

ORDER_ITEM (**ORDERNO** REFERENCES CUST_ORDER, **ITEMNO** REFERENCES ITEM, QTY NUMBER (3))

DRIVER (**DCODE** VARCHAR2 (5), DNAME VARCHAR2 (30))

Implement a check constraint to check DCODE starts with 'D'

CITY (**CCODE** VARCHAR2 (5), CNAME VARCHAR2 (30))

Implement a check constraint to check CCODE starts with CT

TRUCK (**TRUCKCODE** VARCHAR2 (5), TTYPE VARCHAR2 (30))

Implement a check constraint to check TRUCKCODE starts with 'T'

Implement a check constraint to check TTYPE in ('L','H')

L-Light

H- Heavy

Each truck is assigned a unique truck code. There can be many trucks belonging to the same truck type.

DRIVE_TRUCK (**TRUCKCODE** REFERENCES TRUCK, **DCODE** REFERENCES DRIVER, **DOT** DATE, CCODE REFERENCES CITY)

DOT – Date of Trip

SHIPMENT (**ORDERNO** REFERENCES CUST_ORDER, **ITEMNO** REFERENCES ITEM, SHIP_DATE DATE)

SHIP_DATE is the date an order is shipped.

The Primary Key of each relation is underlined.

a. Develop DDL to implement the above Schema enforcing primary key, check constraints and foreign key constraints. (35)

b. Populate the database with a rich data set. (15)

c. Develop a SQL query to list the details of customers who have placed more than three orders. (5)

d. Develop a SQL query to list the details of items whose price is less than the average price of all items. (5)

e. Develop an update statement to update the value of ORD_AMT. (10)

f. Create a view that will keep track of the details of each driver and the number of trips travelled. (10)

g. Develop a database trigger that will not permit to insert more than two records in the DRIVE_TRUCK table for a particular city on the same day. (15)

h. Develop a procedure DISP that will accept driver code and display the total number of trips travelled. Include exception in the procedure that will display a message 'No such driver' if the driver number does not exist in the DRIVER relation. (10)
