SQL MCQ EXAM

QUESTIONS : 180 MINUTES ALLOWED : 180 MARKS : 180
PASSING : 100

22ecpat093@ldce.ac.in Switch account

Draft saved

* Indicates required question

Email *

Your email

SELECT ... AS OF TIMESTAMP ... STATEMENT IS USED TO 1 point

- [ ] A. GET OLDER VERSION OF THE TABLE AT THE SPECIFIED TIMESTAMP
- [ ] B. GET ALL AVAILABLE OLDER VERSIONS OF THE TABLE
- [ ] C. UNCOMMIT THE CHANGES TO SPECIFIED TIMESTAMP
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS. 1 point

- [ ] A. SELECT TO_CHAR(HIRE_DATE,'YYYY') , MAX(SALARY) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'YYYY');
- [ ] B. SELECT TO_CHAR(HIRE_DATE,'YYYY') , MAX(SALARY) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'YY');
- [ ] C. SELECT TO_CHAR(HIRE_DATE,'RR') , MAX(SALARY) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'RRRR');
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. * HAS TWO MEANINGS
- [ ] B. * WHEN USED AS SINGLE ITEM IN SELECT, USED AS MULTIPLICATION OPERATOR.
- [ ] C. * IS EASY WAY TO DISPLAY ALL ROWS OF THE TABLE.
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , SALARY AND HIGHEST SALARY PAID TO CORRESPONDING DEPARTMENT. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , "MAXSAL" FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAXSAL FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MaxSal FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MaxSal FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) "MaxSal" FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT INITCAP('aDU1MaRcHa$KothMIR') FROM DUAL; 1 point

- [ ] A. Adu1Marcha$Kothmir
- [ ] B. Adu Marcha Kothmir
- [ ] C. Adu1marcha$Kothmir
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL LAST_NAME WHOSE SPELLING LENGTH IS UPTO 6. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE LAST_NAME LIKE '_ _ _ _ _ _';
- [ ] B. SELECT * FROM EMPLOYEES WHERE NOT LAST_NAME LIKE '_ _ _ _ _ _%';
- [ ] C. SELECT * FROM EMPLOYEES WHERE LAST_NAME NOT LIKE '_ _ _%_ _ _ _';
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS ? 1 point

- [ ] A. SELECT NVL(COMMISSION_PCT,.05) FROM EMPLOYEES;
- [ ] B. SELECT NVL(COMMISSION_PCT,NA) FROM EMPLOYEES;
- [ ] C. SELECT NVL(TO_CHAR(COMMISSION_PCT),'NA') FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

ROWS ARE LOCKED WHEN WE EXECUTE ____ TYPE COMMAND AND UNLOCKED WHEN ___ TYPE COMMAND IS USED. 1 point

- [ ] A. DRL , TCL
- [ ] B. TCL , DML
- [ ] C. DML , TCL/DDL
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY JOB_ID AND AVERAGE SALARY PAID FOR THE JOBS TO WHICH EITHER LOWEST AVERAGE SALARY OR HIGHEST AVERAGE SALARY ID PAID. 1 point

- [ ] A. SELECT JOB_ID , AVG(SALARY) FROM EMPLOYEES GROUP BY JOB_ID HAVING AVG(SALARY) IN (SELECT MIN(AVG(SALARY)) , MAX(AVG(SALARY)) FROM EMPLOYEES GROUP BY JOB_ID);
- [ ] B. SELECT JOB_ID , AVG(SALARY) FROM EMPLOYEES HAVING AVG(SALARY) = SOME (SELECT MIN(AVG(SALARY)) , MAX(AVG(SALARY)) FROM EMPLOYEES GROUP BY JOB_ID) GROUP BY JOB_ID;
- [ ] C. SELECT JOB_ID , AVG(SALARY) FROM EMPLOYEES HAVING AVG(SALARY) = SOME ((SELECT MIN(AVG(SALARY)) FROM EMPLOYEES GROUP BY JOB_ID), (SELECT MAX(AVG(SALARY)) FROM EMPLOYEES GROUP BY JOB_ID)) GROUP BY JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH IS TRUE ABOUT TIMESTAMP WITH LOCAL TIME ZONE? 1 point

- [ ] A. STORES DATA WITH LOCAL TIME ZONE
- [ ] B. STORES DATA WITH SERVER'S TIME ZONE
- [ ] C. RETURNS DATA WITH LOCAL TIME ZONE
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE LOGICALLY CORRECT STATEMENTS TO PERFORM NATURAL JOIN. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_NAME FROM EMPLOYEES NATURAL JOIN DEPARTMENTS;
- [ ] B. SELECT * FROM DEPARTMENTS NATURAL JOIN REGIONS;
- [ ] C. SELECT CITY , REGION_NAME FROM LOCATIONS NATURAL JOIN REGIONS;
- [ ] D. All of the above.
- [ ] E. None of the above.

HOW MANY SET OPERATORS REQUIRE TO CLUB RESULT OF 4 SELECT STATEMENTS WITHIN COMPOUND STAEMENT. 1 point

- [ ] A. 4
- [ ] B. 5
- [ ] C. 3
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH RULES MUST BE FOLLOWED WITHIN COMPOUND STATEMENT. 1 point

- [ ] A. NO. OF ITEMS IN ALL SELECT MUST BE SAME
- [ ] B. THEIR CORRESPONDING DATA TYPES MUST MATCH
- [ ] C. WE CAN USE ANY LITERAL VALUE OR NULL TO MAKE NO. OF ITEMS SAME IN ALL SELECT.
- [ ] D. All of the above.
- [ ] E. None of the above.

WHAT IS THE DEFAULT EXECUTION PRECEDENCE WHEN WE WRITE COMPOUND STATEMENT. 1 point

- [ ] A. TOP TO DOWN AND LEFT TO RIGHT
- [ ] B. TOP TO DOWN AND RIGHT TO LEFT
- [ ] C. RIGHT TO LEFT AND TOP TO DOWN
- [ ] D. All of the above.
- [ ] E. None of the above.

BELOW COMMAND WILL DEVELOP NEW YEARLY SALARY. 1 point

- [ ] A. SELECT SALARY + 2000 * 12 "NEW YEARLY SALARY" FROM EMPLOYEES;
- [ ] B. SELECT SALARY * 12 + 2000 "NEW YEARLY SALARY" FROM EMPLOYEES;
- [ ] C. SELECT (SALARY + 2000) * 12 "NEW YEARLY SALARY" FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH DATA TYPE REQUIRES 7 BYTES TO STORE IT'S VALUE? 1 point

- [ ] A. NUMBER
- [ ] B. TIMESTAMP
- [ ] C. DATE
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , SALARY , GRADE_LEVEL FOR EMPLOYEE AND HIS/HER MANAGER. 1 point

- [ ] A. SELECT E.FIRST_NAME , E.SALARY , EJG.GRADE_LEVEL , M.FIRST_NAME , M.SALARY , MJG.GRADE_LEVEL FROM EMPLOYEES E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID JOIN JOB_GRADES EJG ON E.SALARY BETWEEN EJG.LOWEST_SAL AND EJG.HIGHEST_SAL JOIN JOB_GRADES MJG ON M.SALARY BETWEEN LOWEST_SAL AND HIGHEST_SAL;
- [ ] B. SELECT E.FIRST_NAME , E.SALARY , EJG.GRADE_LEVEL , M.FIRST_NAME , M.SALARY , MJG.GRADE_LEVEL FROM EMPLOYEES E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID JOIN JOB_GRADES EJG ON E.SALARY BETWEEN EJG.LOWEST_SAL AND EJG.HIGHEST_SAL JOIN JOB_GRADES MJG ON M.SALARY BETWEEN MJG.LOWEST_SAL AND MJG.HIGHEST_SAL;
- [ ] C. SELECT E.FIRST_NAME , E.SALARY , EJG.GRADE_LEVEL , M.FIRST_NAME , M.SALARY , MJG.GRADE_LEVEL FROM EMPLOYEES E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID JOIN JOB_GRADES EJG ON M.SALARY BETWEEN EJG.LOWEST_SAL AND EJG.HIGHEST_SAL JOIN JOB_GRADES MJG ON E.SALARY BETWEEN MJG.LOWEST_SAL AND MJG.HIGHEST_SAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

ROLLBACK CAN BE USED TO UNDO THE CHANGES DONE THRU _____. 1 point

- [ ] A. LAST DML STATEMENT
- [ ] B. ALL CHANGES DONE INSIDE WHOLE TRANSACTION
- [ ] C. CHANGES DONE AFTER SPECIFIED SAVEPOINT
- [ ] D. All of the above.
- [ ] E. None of the above.

WHAT IS TRANSACTION? 1 point

- [ ] A. BUNCH OF DDL STATEMENTS
- [ ] B. BUNCH OF TCL STATEMENTS
- [ ] C. BUNCH OF INSERT, UPDATE, DELETE AND MERGE ON SINGLE OR MULTIPLE TABLES WITHIN SESSION.
- [ ] D. All of the above.
- [ ] E. None of the above.

SUB QUERY MAY BE REQUIRED TO SOLVE ... 1 point

- [ ] A. WHEN WE NEED TO COMPARE SUMMARY ITEMS IN 'WHERE' CLAUSE
- [ ] B. WHEN WE NEED TO COMPARE NESTED SUMMARY ITEM IN 'HAVING' CLAUSE
- [ ] C. WHEN WE WANT TO USE RESULT OF SUB QUERY AS TABLE FOR MAIN QUERY
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT ... VERSIONS ... BETWEEN ... IS USED

1 point

- [ ] A. TO GET ALL OLDER VERSIONS DATA AVAILABLE FOR THE TABLE
- [ ] B. TO GET SPECIFIC TIME OLDER VERSION OF THE TABLE
- [ ] C. UNCOMMIT THE CHANGES DONE IN LAST TRANSACTION
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , LOWEST SALARY PAID FOR CORRESPONDING DEPARTMENT, JOB_ID , AVERAGE SALARY PAID FOR CORRESPONDING JOB.

1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , JOB_ID , AVGSAL FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID) E2 ON E1.DEPARTMENT_ID = E2.DEPARTMENT_ID JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES GROUP BY JOB_ID) E3 ON E1.JOB_ID = E3.JOB_ID;
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , E3.JOB_ID , AVGSAL FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID) E2 ON E1.DEPARTMENT_ID = E2.DEPARTMENT_ID JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES GROUP BY JOB_ID) E3 ON E1.JOB_ID = E3.JOB_ID;
- [ ] C. SELECT FIRST_NAME , E1.DEPARTMENT_ID , "MIN(SALARY)" , E3.JOB_ID , AVGSAL FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID) E2 ON E1.DEPARTMENT_ID = E2.DEPARTMENT_ID JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES GROUP BY JOB_ID) E3 ON E1.JOB_ID = E3.JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT DEPARTMENT_ID , JOB_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID || JOB_ID;
- [ ] B. SELECT DEPARTMENT_ID || JOB_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID || JOB_ID;
- [ ] C. SELECT DEPARTMENT_ID || JOB_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID , JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

INCREMENT BASED ON COMMISSION_PCT : IF EMPLOYEE GETS COMMISSION, HE WILL GET 20% INCREMENT OTHERWISE 30% INCREMENT. 1 point

- [ ] A. SELECT CASE WHEN COMMISSION_PCT LIKE '%' THEN SALARY * 1.2 ELSE SALARY * 1.3 END FROM EMPLOYEES;
- [ ] B. SELECT CASE WHEN COMMISSION_PCT > 0 THEN SALARY * 1.2 ELSE SALARY * 1.3 END FROM EMPLOYEES;
- [ ] C. SELECT CASE WHEN COMMISSION_PCT IS NOT NULL THEN SALARY * 1.2 ELSE SALARY * 1.3 END FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

INCREMENT POLICY : FOR IT_PROG -> 40% , FOR DEPARTMENT_ID 90 -> 30%, FOR THOSE WHO REPORTS TO MANAGER_ID 149 -> 20%, FOR REMAINING 10%

1 point

- [ ] A. SELECT CASE WHEN JOB_ID = 'IT_PROG' THEN SALARY * 1.4 WHEN DEPARTMENT_ID = 90 THEN SALARY * 1.3 WHEN MANAGER_ID = 149 THEN SALARY * 1.2 ELSE SALARY * 1.1 END NEW_SALARY FROM EMPLOYEES;
- [ ] B. SELECT CASE WHEN JOB_ID = 'IT_PROG' THEN SALARY * 1.4 ELSE CASE DEPARTMENT_ID WHEN 90 THEN SALARY * 1.3 ELSE DECODE(MANAGER_ID,149,SALARY * 1.2,SALARY * 1.1) END END NEW_SALARY FROM EMPLOYEES;
- [ ] C. SELECT CASE WHEN JOB_ID = 'IT_PROG' THEN SALARY * 1.4 ELSE CASE DEPARTMENT_ID WHEN 90 THEN SALARY * 1.3 ELSE DECODE(MANAGER_ID,149,SALARY * 1.2,SALARY * 1.1 END END END NEW_SALARY FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS.

1 point

- [ ] A. SELECT MIN(AVG(SALARY)) , MAX(AVG(SALARY)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] B. SELECT DEPARTMENT_ID , MIN(AVG(SALARY)) , MAX(AVG(SALARY)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] C. SELECT MIN(AVG(SALARY)) , MAX(AVG(SALARY)) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS. 1 point

- [ ] A. SELECT CITY , COUNTRY_ID, COUNTRY_NAME , REGION_ID , REGION_NAME FROM LOCATIONS JOIN COUNTRIES C (COUNTRY_ID) JOIN REGIONS R ON C.REGION_ID = R.REGION_ID;
- [ ] B. SELECT CITY , COUNTRY_ID, COUNTRY_NAME , R.REGION_ID , REGION_NAME FROM LOCATIONS JOIN COUNTRIES C (COUNTRY_ID) JOIN REGIONS R ON C.COUNTRY_ID = R.COUNTRY_ID;
- [ ] C. SELECT CITY , C.COUNTRY_ID, COUNTRY_NAME , REGIONS.REGION_ID , REGION_NAME FROM LOCATIONS JOIN COUNTRIES C (COUNTRY_ID) JOIN REGIONS ON C.REGION_ID = REGIONS.REGION_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. SELECT COUNTRY_NAME , REGION_ID , REGION_NAME FROM COUNTRIES JOIN REGIONS USING (REGION_ID);
- [ ] B. SELECT COUNTRY_NAME , REGIONS.REGION_ID , REGION_NAME FROM COUNTRIES JOIN REGIONS USING (REGION_ID);
- [ ] C. SELECT COUNTRY_NAME , C.REGION_ID , REGION_NAME FROM COUNTRIES C JOIN REGIONS USING (REGION_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY COMMON JOBS OF DEPTNO 10 AND 20. 1 point

- [ ] A. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 UNION ALL SELECT JOB FROM MPSEMP WHERE DEPTNO = 20;
- [ ] B. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 UNION SELECT JOB FROM MPSEMP WHERE DEPTNO = 20;
- [ ] C. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 20;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. DISTINCT OR UNIQUE ONLY ONE CAN BE USED IN SINGLE SELECT STATEMENT.
- [ ] B. DISTINCT OR UNIQUE MUST BE PLACED JUST AFTER SELECT KEY WORD INSIDE COMMAND.
- [ ] C. WE CAN USE ANY ONE OF DISTINCT OR UNIQUE MULTIPLE TIMES WITHIN SINGLE SELECT STATEMENT.
- [ ] D. All of the above.
- [ ] E. None of the above.

HOW TO ADD 2 AND HALF YEARS TO DATE? 1 point

- [ ] A. SELECT ADD_YEARS(SYSDATE,2.5) FROM DUAL;
- [ ] B. SELECT ADD_MONTHS(SYSDATE,30) FROM DUAL;
- [ ] C. SELECT ADD_MONTHS(ADD_YEARS(SYSDATE,2),6) FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT COMMANDS. 1 point

- [ ] A. SELECT FIRST_NAME || Q' ('s monthly salary is Rs.(' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] B. SELECT FIRST_NAME || Q' ['s monthly salary is Rs.] ' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] C. SELECT FIRST_NAME || Q' } 's monthly salary is Rs.} ' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENT. 1 point

- [ ] A. SELECT AVG(SALARY) FROM EMPLOYEES WHERE DEPARTMENT_ID = 90;
- [ ] B. SELECT COUNT(*) , COUNT(DISTINCT JOB_ID) FROM EMPLOYEES WHERE NOT MANAGER_ID IN (100,124,149);
- [ ] C. SELECT MAX(SALARY) - MIN(SALARY) FROM EMPLOYEES WHERE HIRE_DATE NOT LIKE '%E%';
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , LOWEST SALARY PAID FOR CORRESPONDING DEPARTMENT, JOB_ID , AVERAGE SALARY PAID FOR CORRESPONDING JOB.

1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , JOB_ID , AVGSAL FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) MINSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID) NATURAL JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES GROUP BY JOB_ID);
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , JOB_ID , AVGSAL FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID) NATURAL JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES GROUP BY JOB_ID);
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , JOB_ID , AVGSAL FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) MINSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID) NATURAL JOIN (SELECT JOB_ID , AVG(SALARY) FROM EMPLOYEES GROUP BY JOB_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

INSERT ... SUB QUERY, SYNTAX CAN INSERT _____ ROWS WITH SINGLE STATEMENT EXECUTION.

1 point

- [ ] A. 1
- [ ] B. 1 OR MORE
- [ ] C. 0 OR MORE
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE LIST OF VALID NUMBER FORMAT USED FOR CONVERTING NUMBER TYPE VALUES. 1 point

- [ ] A. 9 0 , G D .
- [ ] B. $ L U S PR MI
- [ ] C. eeee EEEE Eeee
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH BELOW COMMAND(S) WILL DISPLAY 4 AS RESULT ? 1 point

- [ ] A. SELECT MOD(19,5) FROM DUAL;
- [ ] B. SELECT POWER(16,1/2) FROM DUAL;
- [ ] C. SELECT ROUND(3.5) FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH SET OPERATOS REMOVES DUPLICATES AS WELL AS SORT RESULT BY DEFAULT. 1 point

- [ ] A. UNION
- [ ] B. INTERSECT
- [ ] C. MINUS
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , MANAGER_ID , DEPARTMENT_NAME FROM EMPLOYEES NATURAL JOIN DEPARTMENTS;
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , E.MANAGER_ID , DEPARTMENT_NAME FROM EMPLOYEES E JOIN DEPARTMENTS USING (DEPARTMENT_ID);
- [ ] C. SELECT FIRST_NAME , E.DEPARTMENT_ID , E.MANAGER_ID , DEPARTMENT_NAME FROM EMPLOYEES E JOIN DEPARTMENTS USING (DEPARTMENT_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS ? 1 point

- [ ] A. SELECT NVL2(COMMISSION_PCT,COMMISSION_PCT*2,.05) FROM EMPLOYEES;
- [ ] B. SELECT NVL2(COMMISSION_PCT,'YES','NO') FROM EMPLOYEES;
- [ ] C. SELECT NVL2(COMMISSION_PCT,'YES',0) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. DUAL HAS SINGLE COLUMN AND SINGLE ROW
- [ ] B. DUAL IS GENERALLY USED TO DISPLAY SOMETHING WHICH IS NOT STORED IN ANY TABLE.
- [ ] C. WE CAN DISPLAY ONLY ONE COLUMN/ITEM USING DUAL STATEMENT IN SINGLE EXECUTION.
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS. 1 point

- [ ] A. SELECT AVG(CASE WHEN SALARY > 10000 THEN SALARY END), MIN(CASE HIRE_DATE WHEN > '31-DEC-95' THEN SALARY END) FROM EMPLOYEES;
- [ ] B. SELECT COUNT(DECODE(JOB_ID,'ST_CLERK',SALARY ELSE COMMISSION_PCT)), SUM(DECODE(DEPARTMENT_ID,90,COMMISSION_PCT,SALARY)) FROM EMPLOYEES;
- [ ] C. SELECT MAX(CASE WHEN TO_CHAR(HIRE_DATE,'YYYY') > '1995' THEN SALARY END)), AVG(DECODE(MANAGER_ID,'149',SALARY) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

HOW TO SUBTRACT 8 MONTHS FROM DATE? 1 point

- [ ] A. SELECT SYSDATE - (30*8) FROM DUAL;
- [ ] B. SELECT SUBTRACT_MONTHS(SYSDATE , 8) FROM DUAL;
- [ ] C. SELECT ADD_MONTHS(SYSDATE , -8) FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH DATA TYPE IS SUITABLE FOR STORING INTERNATIONAL FLIGHT TAKE OFF AND LANDING TIME INFO. 1 point

- [ ] A. TIMESTAMP WITH LOCAL TIME ZONE
- [ ] B. TIMESTAMP WITH TIME ZONE
- [ ] C. TIMESTAMP
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENT. 1 point

- [ ] A. SELECT MANAGER_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY MANAGER_ID HAVING DEPARTMENT_ID BETWEEN 50 AND 90;
- [ ] B. SELECT MANAGER_ID , MIN(SALARY) FROM EMPLOYEES HAVING DEPARTMENT_ID BETWEEN 50 AND 90 GROUP BY MANAGER_ID ;
- [ ] C. SELECT MANAGER_ID , MIN(SALARY) FROM EMPLOYEES WHERE DEPARTMENT_ID BETWEEN 50 AND 90 GROUP BY MANAGER_ID ;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY JOB_ID AND NO. OF EMPLOYEES WORKING FOR IT, FOR THE JOB FOR WHICH LOWEST NO. OF EMPLOYEE WORKS. 1 point

- [ ] A. SELECT JOB_ID , COUNT(*) FROM EMPLOYEES GROUP BY JOB_ID HAVING COUNT(*) = (SELECT MIN(COUNT(*)) FROM EMPLOYEES GROUP BY JOB_ID);
- [ ] B. SELECT JOB_ID , COUNT(*) FROM EMPLOYEES HAVING COUNT(JOB_ID) = (SELECT MIN(COUNT(*)) FROM EMPLOYEES GROUP BY JOB_ID) GROUP BY JOB_ID;
- [ ] C. SELECT JOB_ID , COUNT(*) FROM EMPLOYEES HAVING COUNT(*) = (SELECT MIN(COUNT(JOB_ID)) FROM EMPLOYEES GROUP BY JOB_ID) GROUP BY JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH SET OPERATOR DOESN'T REMOVE DUPLICATES BY DEFAULT. 1 point

- [ ] A. UNION
- [ ] B. INTERSECT
- [ ] C. MINUS
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY HIRE YEAR AND NO. OF EMPLOYEES HIRED FOR THE YEAR. FOR THOSE YEARS IN WHICH HIGHEST NO. OF APPOINTMENTS WERE DONE. 1 point

- [ ] A. SELECT TO_CHAR(HIRE_DATE,'RRRR') YEAR, COUNT(*) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'RRRR') HAVING MAX(COUNT(*)) = (SELECT COUNT(*) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'RRRR'));
- [ ] B. SELECT TO_CHAR(HIRE_DATE,'RRRR') YEAR, COUNT(*) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'RRRR') HAVING COUNT(*) = (SELECT MAX(COUNT(*)) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'RRRR'));
- [ ] C. SELECT TO_CHAR(HIRE_DATE,'RRRR') YEAR, COUNT(*) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'RRRR') HAVING COUNT(*) = MAX(COUNT(*));
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY DATA OF EMPLOYEES TABLE, SORT IT ON DEPARTMENT_ID KEEPING NULLS AT THE TOP AND JOB_ID IN DESCENDING. 1 point

- [ ] A. SELECT DEPARTMENT_ID, JOB_ID , LAST_NAME FROM EMPLOYEES ORDER BY 1 NULLS TOP, JOB_ID DESC;
- [ ] B. SELECT DEPARTMENT_ID DID, JOB_ID DESI, LAST_NAME FROM EMPLOYEES ORDER BY DID NULLS FIRST, JOB_ID DESC;
- [ ] C. SELECT DEPARTMENT_ID DID, JOB_ID DESI, LAST_NAME FROM EMPLOYEES ORDER BY DID NULLS FIRST ASC, DESI DESC;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. CREATE TABLE T4(AUTHOR VARCHAR(15),BOOKNAME VARCHAR2(30),BOOK_CONTENTS BFILE,PRICE NUMBER(4));
- [ ] B. CREATE TABLE T4(AUTHOR VARCHAR(15),BOOKNAME VARCHAR2(30),BOOK_CONTENTS VARCHAR2(4000),PRICE NUMBER(4));
- [ ] C. CREATE TABLE T4(AUTHOR VARCHAR(15),BOOKNAME VARCHAR2(30),BOOK_CONTENTS CHAR(4000),PRICE NUMBER(4));
- [ ] D. All of the above.
- [ ] E. None of the above.

HOW TO FIND DIFFERENCE IN MONTHS BETWEEN TWO DATE VALUES? 1 point

- [ ] A. SELECT (SYSDATE - HIRE_DATE) / 30 MONTHS FROM EMPLOYEES;
- [ ] B. SELECT MONTHS_BETWEEN(SYSDATE - HIRE_DATE) MONTHS FROM EMPLOYEES;
- [ ] C. SELECT MONTHS_BETWEEN(SYSDATE , HIRE_DATE) MONTHS FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , SALARY AND HIGHEST SALARY PAID TO CORRESPONDING DEPARTMENT. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES O JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID) I ON O.DEPARTMENT_ID = I.DEPARTMENT_ID;
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID) I ON EMPLOYEES.DEPARTMENT_ID = I.DEPARTMENT_ID;
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES O JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID) ON O.DEPARTMENT_ID = EMPLOYEES.DEPARTMENT_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT HIRE_DATE , TO_CHAR(HIRE_DATE,'ddTH monTH YEAR') FROM EMPLOYEES;
- [ ] B. SELECT HIRE_DATE , TO_DATE(HIRE_DATE,'HH12 MI SSSSS aM') FROM EMPLOYEES;
- [ ] C. SELECT SYSDATE , TO_CHAR(SYSDATE,' "We are currently in" qth "quarter." ') FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONSTRAINT HAS DIFFERENT SYNTAX FOR COLUMN LEVEL AND TABLE LEVEL. 1 point

- [ ] A. PRIMARY KEY
- [ ] B. NOT NULL
- [ ] C. REFERENCES
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT SALARY , TO_CHAR(SALARY,'$90,909.09') FROM EMPLOYEES;
- [ ] B. SELECT SALARY , TO_CHAR(SALARY,'L90G909D09') FROM EMPLOYEES;
- [ ] C. SELECT SALARY , TO_CHAR(SALARY,'U90G909.09') FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

IF WE WANT TO DISPLAY ONLY EXTRA ROWS OF A TABLE WHAT SHOULD BE USED? 1 point

- [ ] A. LEFT / RIGHT OUTER JOIN WITH EXTRA CONDITION IN 'WHERE' CLAUSE
- [ ] B. LEFT / RIGHT / FULL OUTER JOIN WITH EXTRA CONDITON IN 'ON' CLAUSE
- [ ] C. LEFT / RIGHT / FULL OUTER JOIN WITH EXTRA CONDITION IN 'WHERE' CLAUSE
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT SUM(DECODE(DEPARTMENT_ID,90,SALARY)) DEPT_90_SUM_SAL, MIN(CASE JOB_ID WHEN 'IT_PROG' THEN SALARY END), MAX(CASE WHEN MANAGER_ID IN (100,124) THEN HIRE_DATE END) FROM EMPLOYEES;
- [ ] B. SELECT SUM(DECODE(DEPARTMENT_ID,90,SALARY)) DEPT_90_SUM_SAL, MIN(DECODE(JOB_ID,'IT_PROG',FIRST_NAME)), AVG(DECODE(MANAGER_ID,100,HIRE_DATE)) FROM EMPLOYEES;
- [ ] C. SELECT MIN(CASE WHEN HIRE_DATE BETWEEN '01-JAN-90' AND '31-DEC-95' THEN SALARY END), MAX(CASE WHEN JOB_ID LIKE 'SA%' THEN SALARY END) FROM EMPLOYEES GROUP BY MANAGER_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

FLASHBACK TABLE ... TIMESTAMP ... COMMAND CAN BE USED 1 point

- [ ] A. TO UNCOMMIT THE LAST COMMITTED TRANSACTION
- [ ] B. TO ROLLBACK THE CURRENT TRANSACTION
- [ ] C. TO UNCOMMIT ANY AVAILABLE OLDER VERSION COMMITTED TRANSACTION
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , LOWEST SALARY PAID FOR CORRESPONDING DEPARTMENT, JOB_ID , AVERAGE SALARY PAID FOR CORRESPONDING JOB.

1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , JOB_ID , AVGSAL FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID) E2 ON E1.DEPARTMENT_ID = E2.DEPARTMENT_ID JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES E3 GROUP BY JOB_ID) ON E1.JOB_ID = E3.JOB_ID;
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , JOB_ID , AVGSAL FROM EMPLOYEES JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID) USING (DEPARTMENT_ID) JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES GROUP BY JOB_ID USING (JOB_ID));;
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , "MIN(SALARY)" , JOB_ID , AVGSAL FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES E2 GROUP BY DEPARTMENT_ID) ON E1.DEPARTMENT_ID = E2.DEPARTMENT_ID JOIN (SELECT JOB_ID , AVG(SALARY) AVGSAL FROM EMPLOYEES GROUP BY JOB_ID) E2 ON E1.JOB_ID = E2.JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS.

1 point

- [ ] A. CREATE TABLE T5(COUNTRY_NAME VARCHAR2(15),BRANCH_NAME VARCHAR2(15),TRANSACTION_TIME TIMESTAMP WITH LOCAL TIME ZONE,CR_DR CHAR,AMOUNT NUMBER(10));
- [ ] B. CREATE TABLE T5(COUNTRY_NAME VARCHAR2(15),BRANCH_NAME VARCHAR2(15),TRANSACTION_TIME TIMESTAMP WITH TIME ZONE,CR_DR CHAR,AMOUNT NUMBER(10));
- [ ] C. CREATE TABLE T5(COUNTRY_NAME VARCHAR2(15),BRANCH_NAME VARCHAR2(15),TRANSACTION_TIME TIMESTAMP,CR_DR CHAR,AMOUNT NUMBER(10));
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. SELECT COUNT(*) , COUNT(COMMISSION_PCT) , COUNT(DISTINCT COMMISSION_PCT) FROM EMPLOYEES;
- [ ] B. SELECT COUNT(*) , COUNT(UNIQUE *) FROM EMPLOYEES;
- [ ] C. SELECT COUNT(*) - COUNT(COMMISSION_PCT) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY DISTINCT JOB OF DEPTNO 30 WHICH MUST NOT EXISTS IN DEPTNO 20. 1 point

- [ ] A. SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] B. SELECT JOB FROM MPSEMP WHERE DEPTNO = 30 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] C. SELECT JOB FROM MPSEMP WHERE DEPTNO = 30 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 20;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY JOBS OF DEPTNO 10 AND 20 WHICH ARE NOT PRESENT IN DEPTNO 30. (EXCLUDING DUPLICATES) 1 point

- [ ] A. SELECT JOB FROM MPSEMP WHERE DEPTNO IN (10,20) MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] B. SELECT JOB FROM MPSEMP WHERE DEPTNO =10 UNION ALL SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] C. SELECT JOB FROM MPSEMP WHERE DEPTNO =10 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] D. All of the above.
- [ ] E. None of the above.

INCREMENT POLICY : HIRED BEFORE YEAR 96 FOR GETTING SALARY UPTO 10000 40% , FOR GETTING SALARY ABOVE 10000 30% , HIRED AFTER YEAR 95 FOR GETTING SALARY UPTO 10000 +2000 , FOR GETTING SALARY ABOVE 10000 +3000 1 point

- [ ] A. SELECT CASE WHEN HIRE_DATE <= '31-DEC-95' THEN CASE WHEN SALARY <= 10000 THEN SALARY * 1.4 ELSE SALARY * 1.3 END ELSE CASE WHEN SALARY <= 10000 THEN SALARY + 2000 ELSE SALARY + 3000 END END NEW_SALARY FROM EMPLOYEES;
- [ ] B. SELECT CASE WHEN TO_CHAR(HIRE_DATE,'RRRR') <= '1995' THEN CASE WHEN SALARY <= 10000 THEN SALARY * 1.4 ELSE SALARY * 1.3 END ELSE CASE WHEN SALARY <= 10000 THEN SALARY + 2000 ELSE SALARY + 3000 END END NEW_SALARY FROM EMPLOYEES;
- [ ] C. SELECT CASE WHEN TO_CHAR(HIRE_DATE,'YY') < '96' THEN CASE WHEN SALARY <= 10000 THEN SALARY * 1.4 ELSE SALARY * 1.3 END ELSE CASE WHEN SALARY <= 10000 THEN SALARY + 2000 ELSE SALARY + 3000 END END NEW_SALARY FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH DATA TYPE SUPPORTS MAX 4000 BYTES? 1 point

- [ ] A. CHAR
- [ ] B. VARCHAR
- [ ] C. BFILE
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , SALARY AND HIGHEST SALARY PAID TO CORRESPONDING DEPARTMENT. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , "MAX(SALARY)" FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , "MAX(SALARY)" FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , max(salary) FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , "MAX(SALARY)" FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONSTRAINT IS NOT ALLOWED AT TABLE LEVEL. 1 point

- [ ] A. PRIMARY KEY
- [ ] B. REFERENCES
- [ ] C. NOT NULL
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL EXCEPT THOSE WHO DOES SAME JOB AS Jonathon DOES. 1 point

- [ ] A. SELECT FIRST_NAME , JOB_ID FROM EMPLOYEES WHERE JOB_ID NOT IN (SELECT JOB_ID FROM EMPLOYEES WHERE FIRST_NAME = 'Jonathon');
- [ ] B. SELECT FIRST_NAME , JOB_ID FROM EMPLOYEES WHERE JOB_ID |= ALL (SELECT JOB_ID FROM EMPLOYEES WHERE FIRST_NAME = 'Jonathon');
- [ ] C. SELECT FIRST_NAME , JOB_ID FROM EMPLOYEES WHERE JOB_ID # ALL (SELECT JOB_ID FROM EMPLOYEES WHERE FIRST_NAME = 'Jonathon');
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE ROW WHOSE FIRST_NAME SPELLING STARTS WITH ALPHABETS 'D' TO 'K'. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE FIRST_NAME BETWEEN 'D' AND 'K';
- [ ] B. SELECT * FROM EMPLOYEES WHERE FIRST_NAME BETWEEN 'D' AND 'L';
- [ ] C. SELECT * FROM EMPLOYEES WHERE FIRST_NAME BETWEEN 'E' AND 'K';
- [ ] D. All of the above.
- [ ] E. None of the above.

HOW CAN WE CHANGE THE EXECUTION PRECEDENCE WITHIN COMPOUND STATEMENT? 1 point

- [ ] A. BY CHANGING THE PLACEMENT OF STATEMENTS AND CORRESPONDING SET OPERATOR
- [ ] B. BY USING MOON BRACKETS
- [ ] C. CAN'T CHANGE
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , JOB_ID , JOB_TITLE , SALARY , GRADE_LEVEL FOR EMPLOYEE AND HIS/HER MANAGER. 1 point

- [ ] A. SELECT E.FIRST_NAME , E.JOB_ID , JOB_TITLE , SALARY , GRADE_LEVEL FROM EMPLOYEES E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID JOIN JOBS EJ ON E.JOB_ID = EJ.JOB_ID JOIN JOBS MJ ON M.JOB_ID = MJ.JOB_ID JOIN JOB_GRADES EJG ON E.SALARY BETWEEN EJG.LOWEST_SAL AND EJG.HIGHEST_SAL JOIN JOB_GRADES MJG ON M.SALARY BETWEEN MJG.LOWEST_SAL AND MJG.HIGHEST_SAL;
- [ ] B. SELECT FIRST_NAME , E.JOB_ID , JOB_TITLE , SALARY , GRADE_LEVEL FROM EMPLOYEES E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID JOIN JOBS EJ ON E.JOB_ID = EJ.JOB_ID JOIN JOBS MJ ON M.JOB_ID = MJ.JOB_ID JOIN JOB_GRADES EJG ON E.SALARY BETWEEN EJG.LOWEST_SAL AND EJG.HIGHEST_SAL JOIN JOB_GRADES MJG ON M.SALARY BETWEEN MJG.LOWEST_SAL AND MJG.HIGHEST_SAL;
- [ ] C. SELECT E.FIRST_NAME , E.JOB_ID , EJ.JOB_TITLE , E.SALARY , EJG.GRADE_LEVEL FROM EMPLOYEES E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID JOIN JOBS EJ ON E.JOB_ID = EJ.JOB_ID JOIN JOBS MJ ON M.JOB_ID = MJ.JOB_ID JOIN JOB_GRADES EJG ON E.SALARY BETWEEN EJG BETWEEN LOWEST_SAL AND HIGHEST_SAL JOIN JOB_GRADES MJG ON M.SALARY BETWEEN MJG.LOWEST_SAL AND MJG.HIGHEST_SAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS ? 1 point

- [ ] A. SELECT DEPARTMENT_NAME , CITY , COUNTRY_NAME FROM DEPARTMENT NATURAL JOIN LOCATIONS NATURAL JOIN COUNTRIES;
- [ ] B. SELECT DEPARTMENT_NAME , CITY , COUNTRY_NAME FROM LOCATIONS NATURAL JOIN DEPARTMENTS NATURAL JOIN COUNTRIES;
- [ ] C. SELECT DEPARTMENT_NAME , CITY , COUNTRY_NAME FROM DEPARTMENT NATURAL JOIN COUNTRIES NATURAL JOIN LOCATIONS;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY DATA OF EMPLOYEES TABLE, SORT ROWS IN DESCENDING ORDER OF EMPLOYEES TABLE. 1 point

- [ ] A. SELECT LAST_NAME , HIRE_DATE FROM EMPLOYEES ORDER BY HIRE_DATE DESC;
- [ ] B. SELECT LAST_NAME , HIRE_DATE "Join Date" FROM EMPLOYEES ORDER BY Join Date DESC;
- [ ] C. SELECT LAST_NAME , HIRE_DATE "Join Date" FROM EMPLOYEES ORDER BY 2 DESC;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT COMMANDS. 1 point

- [ ] A. SELECT FIRST_NAME || ' ' 's monthly salary is Rs.' SALARY || '/-.' FROM EMPLOYEES;
- [ ] B. SELECT FIRST_NAME || Q '~'s monthly salary is Rs.~' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] C. SELECT FIRST_NAME || Q ' ' 's monthly salary is Rs.' ' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

UPDATE STATMENT CAN BE USED TO UPDATE ... 1 point

- [ ] A. SINGLE/MULTIPLE COLUMNS IN SINGLE ROW
- [ ] B. SINGLE COLMNN IN SINGLE/MULTIPLE ROWS
- [ ] C. SINGLE/MULTIPLE COLUMNS IN SINGLE/MULTIPLE ROWS
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE WHOSE FIRST_NAME SPELLING HAS TWO OCCURRENCES OF ALPHABET 'e'. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE FIRST_NAME LIKE 'ee';
- [ ] B. SELECT * FROM EMPLOYEES WHERE FIRST_NAME LIKE '%ee%';
- [ ] C. SELECT * FROM EMPLOYEES WHERE FIRST_NAME LIKE '%e%e%';
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL ROWS OF EMPLOYEES TABLE BETWEEN 8 AND 14 (INCLUSIVE OF BOTH 8 AND 14) 1 point

- [ ] A. SELECT * FROM EMPLOYEES FETCH FIRST 7 ROWS ONLY OFFSET 7 ROWS;
- [ ] B. SELECT * FROM EMPLOYEES OFFSET 7 ROW FETCH NEXT 7 ROW ONLY;
- [ ] C. SELECT * FROM EMPLOYEES OFFSET 7 ROWS FETCH FIRST 7 ROWS ONLY;
- [ ] D. All of the above.
- [ ] E. None of the above.

SAVEPOINT IS USED TO _____ 1 point

- [ ] A. BREAK TRANSACTION INTO PARTS.
- [ ] B. ROLLBACK CHANGES AFTER SPECIFIED SAVEPOINT
- [ ] C. COMMIT CHANGES BEFORE SAVEPOINT
- [ ] D. All of the above.
- [ ] E. None of the above.

WHAT IS DEFAULT SIZE FOR FRACTIONAL SECONDS? 1 point

- [ ] A. 3
- [ ] B. 6
- [ ] C. 9
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , SALARY AND HIGHEST SALARY PAID TO CORRESPONDING DEPARTMENT. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , "MAXSAL" FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES GROUP BY DEPARTMENT_ID) USING (DEPARTMENT_ID);
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES O JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) MAXSAL FROM EMPLOYEES I GROUP BY DEPARTMENT_ID) ON O.DEPARTMENT_ID = I.DEPARTMENT_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT COUNTRIES.COUNTRY_NAME , REGION_ID , R.REGION_NAME FROM COUNTRIES JOIN REGIONS R ON COUNTRIES.REGION_ID = R.REGION_ID;
- [ ] B. SELECT COUNTRIES.COUNTRY_NAME , COUNTRIES.REGION_ID , R.REGION_NAME FROM COUNTRIES JOIN REGIONS R ON COUNTRIES.REGION_ID = R.REGION_ID;
- [ ] C. SELECT COUNTRIES.COUNTRY_NAME , R.REGION_ID , R.REGION_NAME FROM COUNTRIES JOIN REGIONS R ON COUNTRIES.REGION_ID = R.REGION_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY DATA OF EMPLOYEES WHO WERE HIRED BEFORE YEAR 1995. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE HIRE_DATE < 1995;
- [ ] B. SELECT * FROM EMPLOYEES WHERE HIRE_DATE <= '31-DEC-1995';
- [ ] C. SELECT * FROM EMPLOYEES WHERE '01-JANUARY-1995' > HIRE_DATE
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL EXCEPT THOSE WHO REPORTS TO MANAGERS 100, 124 AND 149. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE MANAGER_ID NOT IN (100,'124',149);
- [ ] B. SELECT * FROM EMPLOYEES WHERE NOT MANAGER_ID IN (100,124,149);
- [ ] C. SELECT * FROM EMPLOYEES WHERE MANAGER_ID <> 100 AND MANAGER_ID NOT IN (124,149);
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE INCORRECT CONSTRAINT TYPE. 1 point

- [ ] A. CREATE TABLE EMP(EID NUMBER(2) PRIMARY KEY,ENM VARCHAR2(10),MID NUMBER(2) REFERENCES EMP);
- [ ] B. CREATE TABLE EMP(EID NUMBER(2) PRIMARY KEY,ENM VARCHAR2(10),MID NUMBER(2) REFERENCES EMP(EID));
- [ ] C. CREATE TABLE EMP(EID NUMBER(2) UNIQUE NOT NULL,ENM VARCHAR2(10),MID NUMBER(2) REFERENCES EMP);
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH COMMAND(S) IS/ARE CORRECT TO GET DAYS BETWEEN TWO DATES? 1 point

- [ ] A. SELECT '13-MAR-73' - '15JANUARY71' FROM DUAL;
- [ ] B. SELECT '13-03-73' - '15-01-71' FROM DUAL;
- [ ] C. SELECT TO_DATE('13MARCH73') - TO_DATE('15-01-71','DD-MM-RR') FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENT. 1 point

- [ ] A. CREATE TABLE T7(RNO NUMBER(2) PRIMARY KEY,NM VARCHAR2(10) CHECK(NM=UPPER(NM)));
- [ ] B. CREATE TABLE T7(RNO NUMBER(2) UNIQUE NOT NULL,NM VARCHAR2(10) CHECK(NM=INITCAP(NM));
- [ ] C. CREATE TABLE T7(RNO NUMBER(2),NM VARCHAR2(10) CHECK(LOWER(NM));
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY COMMON JOBS OF ALL THE 3 DEPTNOs. 1 point

- [ ] A. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 UNION SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 UNION SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] B. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 UNION ALL SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 UNION ALL SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] C. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY DATA FOR THOSE WHO WERE HIRED BETWEEN YEAR 1991 AND 1998. (EXCLUDE BOTH YEARS 1991 AND 1998) 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE HIRE_DATE > '31-DEC-91' AND HIRE_DATE < '01-JAN-98';
- [ ] B. SELECT * FROM EMPLOYEES WHERE HIRE_DATE BETWEEN '1-JAN-92' AND '31-DEC-97';
- [ ] C. SELECT * FROM EMPLOYEES WHERE NOT (HIRE_DATE >= '1-JAN-92' OR HIRE_DATE <= '31-DEC-97');
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH DATA TYPE IS SUITABLE FOR STORING FIXED DEPOSIT PERIOD ? 1 point

- [ ] A. INTERVAL
- [ ] B. INTERVAL DAY TO SECOND
- [ ] C. INTERVAL YEAR TO SECOND
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE WHO GETS HIGHER SALARY THAN 'Abel'. 1 point

- [ ] A. SELECT LAST_NAME , SALARY FROM EMPLOYEES WHERE SALARY > (SELECT SALARY FROM EMPLOYEES WHERE LAST_NAME = 'Abel');
- [ ] B. SELECT LAST_NAME , SALARY FROM EMPLOYEES WHERE (SELECT SALARY FROM EMPLOYEES WHERE LAST_NAME = 'Abel') < SALARY;
- [ ] C. SELECT LAST_NAME , SALARY FROM EMPLOYEES WHERE LAST_NAME <> 'Abel' AND SALARY > (SELECT SALARY FROM EMPLOYEES);
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS ? 1 point

- [ ] A. SELECT MANAGER_ID , MIN(SALARY) FROM EMPLOYEES WHERE MANAGER_ID NOT IN (100,124,149) GROUP BY MANAGER_ID;
- [ ] B. SELECT MANAGER_ID , MIN(SALARY) FROM EMPLOYEES HAVING MANAGER_ID NOT IN (100,124,149) GROUP BY MANAGER_ID;
- [ ] C. SELECT MANAGER_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY MANAGER_ID HAVING MANAGER_ID NOT IN (100,124,149);
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , LOWEST SALARY PAID FOR CORRESPONDING DEPARTMENT, JOB_ID , AVERAGE SALARY PAID FOR CORRESPONDING JOB.

1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , "MinSalary" , E1.JOB_ID , AVGSALARY FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) "MinSalary" FROM EMPLOYEES GROUP BY DEPARTMENT_ID) E2 USING(DEPARTMENT_ID) JOIN (SELECT JOB_ID , AVG(SALARY) AvgSalary FROM EMPLOYEES GROUP BY JOB_ID) E3 ON E1.JOB_ID = E3.JOB_ID;
- [ ] B. SELECT FIRST_NAME , E2.DEPARTMENT_ID , "MinSalary" , JOB_ID , AVGSALARY FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) "MinSalary" FROM EMPLOYEES GROUP BY DEPARTMENT_ID) E2 ON E1.DEPARTMENT_ID = E2.DEPARTMENT_ID JOIN (SELECT JOB_ID , AVG(SALARY) AvgSalary FROM EMPLOYEES GROUP BY JOB_ID) E3 USING (JOB_ID);
- [ ] C. SELECT FIRST_NAME , E2.DEPARTMENT_ID , "MinSalary" , E3.JOB_ID , AVGSALARY FROM EMPLOYEES E1 JOIN (SELECT DEPARTMENT_ID , MIN(SALARY) "MinSalary" FROM EMPLOYEES GROUP BY DEPARTMENT_ID) E2 ON E1.DEPARTMENT_ID = E2.DEPARTMENT_ID JOIN (SELECT JOB_ID , AVG(SALARY) AvgSalary FROM EMPLOYEES GROUP BY JOB_ID) E3 ON E1.JOB_ID = E3.JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

BELOW IS/ARE EXAMPLE(S) OF PROJECTION.

1 point

- [ ] A. SELECT REGION_id FROM REGIONS;
- [ ] B. SELECT FIRST_NAME , SALARY , HIRE_DATE FROM EMPLOYEES;
- [ ] C. SELECT * FROM REGIONS;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL JOBS OF DEPTNO 20 AND 30. (EXCLUDING DUPLICATES) 1 point

- [ ] A. SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 UNION ALL SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] B. SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 UNION SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] C. SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL JOBS OF DEPTNO 20 AND 30. (INCLUDING DUPLICATES) 1 point

- [ ] A. SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 UNION ALL SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] B. SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 UNION SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] C. SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 30;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE WHO WORKS IN DEPARTMENTS 20,50,110. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE DEPARTMENT_ID IN (20,50,110);
- [ ] B. SELECT * FROM EMPLOYEES WHERE DEPARTMENT_ID LIKE '%0';
- [ ] C. SELECT * FROM EMPLOYEES WHERE DEPARTMENT_ID = 20 OR DEPARTMENT_ID IN 50 OR 110 = DEPARTMENT_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY HIRE QUARTER AND COUNT OF EMPLOYEES HIRED FOR THE QUARTER IN WHICH EITHER LOWEST OR HIGHEST EMPLOYEES WERE HIRED. (IGNORE YEAR VALUE) 1 point

- [ ] A. SELECT TO_CHAR(HIRE_DATE,'Q') , COUNT(*) FROM EMPLOYEES WHERE COUNT(*) = ANY ((SELECT MAX(COUNT(*)) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'Q')),(SELECT MIN(COUNT(*)) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'Q')));
- [ ] B. SELECT TO_CHAR(HIRE_DATE,'Q') , COUNT(*) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'Q') HAVING COUNT(*) = ANY ((SELECT MAX(COUNT(*)) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'Q')),(SELECT MIN(COUNT(*)) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'Q')));
- [ ] C. SELECT TO_CHAR(HIRE_DATE,'Q') , COUNT(*) FROM EMPLOYEES HAVING COUNT(*) = ANY ((SELECT MAX(COUNT(*)) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'Q')),(SELECT MIN(COUNT(*)) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'Q'))) GROUP BY TO_CHAR(HIRE_DATE,'Q');
- [ ] D. All of the above.
- [ ] E. None of the above.

COMMIT CAN BE USED TO SAVE _____ CHANGES TO TABLE PERMANENTLY. 1 point

- [ ] A. LAST DML STATEMENT
- [ ] B. ALL CHANGES DONE IN TRANSACTION
- [ ] C. CHANGES DONE AFTER SPECIFIED SAVEPOINT
- [ ] D. All of the above.
- [ ] E. None of the above.

HOW TO GET 'ShivRam'? 1 point

- [ ] A. SELECT SUBSTR('LaxmanShivRamKrishnan',7,7) FROM DUAL;
- [ ] B. SELECT SUBSTR('LaxmanShivRamKrishnan',7,13) FROM DUAL;
- [ ] C. SELECT SUBSTR('LaxmanShivRamKrishnan',-15,-8) FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE WHO GETS EITHER LOWEST OR HIGHEST SALARY IN COMPANY. 1 point

- [ ] A. SELECT FIRST_NAME , SALARY FROM EMPLOYEES WHERE SALARY IN (SELECT MIN(SALARY), MAX(SALARY) FROM EMPLOYEES);
- [ ] B. SELECT FIRST_NAME , SALARY FROM EMPLOYEES WHERE SALARY IN (SELECT MIN(SALARY) FROM EMPLOYEES) OR SALARY = (SELECT MAX(SALARY) FROM EMPLOYEES);
- [ ] C. SELECT FIRST_NAME , SALARY FROM EMPLOYEES WHERE SALARY = SOME ((SELECT MIN(SALARY) FROM EMPLOYEES),(SELECT MAX(SALARY) FROM EMPLOYEES));
- [ ] D. All of the above.
- [ ] E. None of the above.

INCREMENT ON THE BASES OF MANAGER_ID. FOR 100 40%, FOR 124 30%, FOR 149 20% FOR REMAINING 7.5% 1 point

- [ ] A. SELECT CASE MANAGER_ID WHEN 100 THEN SALARY * 1.4 WHEN 124 THEN SALARY + SALARY * .30 WHEN 149 THEN SALARY + SALARY * 20 / 100 ELSE SALARY * 1.075 END FROM EMPLOYEES;
- [ ] B. SELECT CASE MANAGER_ID WHEN 100 THEN SALARY * 1.4 WHEN 124 THEN SALARY + SALARY * .30 WHEN 149 THEN SALARY + SALARY * 20 / 100 , SALARY * 1.075 END FROM EMPLOYEES;
- [ ] C. SELECT CASE MANAGER_ID WHEN 100 THEN SALARY * 1.4 WHEN 124 THEN SALARY + SALARY * .30 WHEN 149 THEN SALARY + SALARY * 20 / 100 ELSE SALARY * 1.075) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH COMMAND WILL DEVELOP RESULT OF HYBRID JOIN. (COMBINATIONS OF TWO DIFFERENT TYPES OF JOINS) 1 point

- [ ] A. SELECT DEPARTMENT_ID , CITY , COUNTRY_NAME FROM DEPARTMENTS NATURAL JOIN LOCATIONS JOIN COUNTRIES USING (COUNTRY_ID);
- [ ] B. SELECT DEPARTMENT_ID , CITY , COUNTRY_NAME FROM DEPARTMENTS JOIN LOCATIONS USING (LOCATION_ID) NATURAL JOIN COUNTRIES;
- [ ] C. SELECT DEPARTMENT_ID , CITY , COUNTRY_NAME FROM DEPARTMENTS D JOIN LOCATIONS L ON D.LOCATION_ID = L.LOCATION_ID JOIN COUNTRIES USING (COUNTRY_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL DEPARTMENTAL COLLEGUES OF 'Curtis'. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID FROM EMPLOYEES WHERE DEPARTMENT_ID IN (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE FIRST_NAME = 'Curtis');
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID FROM EMPLOYEES WHERE DEPARTMENT_ID = (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE FIRST_NAME = 'Curtis');
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID FROM EMPLOYEES WHERE (SELECT DEPARTMENT_ID FROM EMPLOYEES WHERE FIRST_NAME = 'Curtis') IN DEPARTMENT_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT INSTR('NeElKaMAL','a',-2,2) FROM DUAL; 1 point

- [ ] A. 8
- [ ] B. 6
- [ ] C. 0
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE WHOSE SECOND LAST ALPHABET IS 'e' FOR BOTH FIRST_NAME AND LAST_NAME. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE FIRST_NAME LIKE '%e_' OR LAST_NAME LIKE '%e_';
- [ ] B. SELECT * FROM EMPLOYEES WHERE FIRST_NAME LIKE '%e_' AND LAST_NAME LIKE '%e_';
- [ ] C. SELECT * FROM EMPLOYEES WHERE FIRST_NAME LIKE AND LAST_NAME LIKE '%e_';
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. SELECT SUM(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] B. SELECT DEPARTMENT_ID , SUM(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] C. SELECT DEPARTMENT_ID , SUM(SALARY) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME OF EMPLOYEE AND HIS/HER MANAGER WITH PROPER COLUMN HEADINGS. 1 point

- [ ] A. SELECT E.FIRST_NAME EMP_NAME , M.FIRST_NAME MGR_NAME FROM EMPLOYEES E JOIN EMPLOYEES M ON E.MANAGER_ID = M.EMPLOYEE_ID;
- [ ] B. SELECT E.FIRST_NAME EMP_NAME , M.FIRST_NAME MGR_NAME FROM EMPLOYEES E JOIN EMPLOYEES M ON M.MANAGER_ID = E.EMPLOYEE_ID;
- [ ] C. SELECT E.FIRST_NAME MGR_NAME , M.FIRST_NAME EMP_NAME FROM EMPLOYEES E JOIN EMPLOYEES M ON M.MANAGER_ID = E.EMPLOYEE_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ONLY UNCOMMON JOBS OF ALL 3 DEPTNOs. 1 point

- [ ] A. SELECT JOB FROM MPSEMP MINUS (SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 30);
- [ ] B. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 INTERSECT SELECT JOB FROM MPSEMP WHERE DEPTNO = 30
- [ ] C. SELECT JOB FROM MPSEMP WHERE DEPTNO = 10 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 20 MINUS SELECT JOB FROM MPSEMP WHERE DEPTNO = 30
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT COUNTRIES.COUNTRY_NAME , REGION_ID , R.REGION_NAME FROM COUNTRIES JOIN REGIONS R USING (REGION_ID);
- [ ] B. SELECT C.COUNTRY_NAME , REGION_ID , REGIONS.REGION_NAME FROM COUNTRIES JOIN REGIONS R USING (REGION_ID);
- [ ] C. SELECT C.COUNTRY_NAME , C.REGION_ID , R.REGION_ID , R.REGION_NAME FROM COUNTRIES C JOIN REGIONS R USING (REGION_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT CREATE TABLE COMMAND. 1 point

- [ ] A. CREATE TABLE T1(NO NUMBER,NM VARCHAR2,AGE NUMBER,GENDER CHAR);
- [ ] B. CREATE TABLE T1(NO NUMBER(2),NM VARCHAR2(20),AGE NUMBER(3),GENDER CHAR);
- [ ] C. CREATE TABLE T1(NO NUMBER(2),NM VARCHAR2(20),AGE NUMBER(3),GENDER CHAR(1));
- [ ] D. All of the above.
- [ ] E. None of the above.

INCREMENT BASED ON JOB_ID VALUE. FOR IT_PROG +2000 , FOR ST_CLERK 30% , FOR SA_REP +750 FOR REMAINING NO INCREMENT. 1 point

- [ ] A. SELECT CASE JOB_ID WHEN 'IT_PROG' THEN SALARY + 2000 WHEN 'ST_CLERk' THEN SALARY * 1.3 WHEN 'SA_REP' THEN SALARY + 750 END FROM EMPLOYEES;
- [ ] B. SELECT CASE JOB_ID WHEN 'IT_PROG' THEN SALARY + 2000 WHEN 'ST_CLERK' THEN SALARY * 1.3 WHEN 'SA_REP' THEN SALARY + 750 ELSE SALARY END FROM EMPLOYEES;
- [ ] C. SELECT CASE JOB_ID WHEN 'IT_PROG' THEN SALARY + 2000 WHEN 'ST_CLERK' THEN SALARY * 1.3 WHEN 'SA_REP' THEN SALARY + 750 ELSE SALARY) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH IS/ARE TRANSACTION CONTROL LANGUAGE STATEMENTS? 1 point

- [ ] A. COMMIT
- [ ] B. ROLLBACK
- [ ] C. SAVEPOINT
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY ALL DATA OF REGIONS TABLE, SORT ROWS IN ASCENDING ORDER OF REGION_NAME COLUMN. 1 point

- [ ] A. SELECT * FROM REGIONS ORDER BY REGION_NAME ASCENDING;
- [ ] B. select * from regions order by 2;
- [ ] C. SeleCT REGION_NAME RNM , REGION_ID FRom REGIONS ORDER BY RNM ASC;
- [ ] D. All of the above.
- [ ] E. None of the above.

REGIONS TABLE HAS 2 COLUMNS. COUNTRIES TABLE HAS 3 COLUMNS. SELECT * FROM COUNTRIES NATURAL JOIN REGIONS; HOW MANY TOTAL COLUMNS WILL GET DISPLAYED. 1 point

- [ ] A. 5
- [ ] B. 4
- [ ] C. 3
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE LIST OF VALID NUMBER FORMAT USED FOR CONVERTING NUMBER TYPE VALUES. 1 point

- [ ] A. D DD DDD MM RM MI
- [ ] B. MON month Month Y YYY HH12 HH24 Am
- [ ] C. Q J W WW SP DD
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT DEPARTMENT_ID , JOB_ID , COUNT(*) FROM EMPLOYEES GROUP BY DEPARTMENT_ID, JOB_ID;
- [ ] B. SELECT DEPARTMENT_ID , JOB_ID , COUNT(*) FROM EMPLOYEES GROUP BY JOB_ID , DEPARTMENT_ID;
- [ ] C. SELECT JOB_ID , COUNT(*) FROM EMPLOYEES GROUP BY DEPARTMENT_ID, JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE WHO GETS LOWER SALARY THAN ANY 'SA_REP'. 1 point

- [ ] A. SELECT FIRST_NAME , JOB_ID , SALARY FROM EMPLOYEES WHERE SALARY < ANY (SELECT SALARY FROM EMPLOYEES WHERE 'SA_REP' = JOB_ID);
- [ ] B. SELECT FIRST_NAME , JOB_ID , SALARY FROM EMPLOYEES WHERE SALARY < (SELECT MAX(SALARY) FROM EMPLOYEES WHERE JOB_ID = ANY ('SA_REP'));
- [ ] C. SELECT FIRST_NAME , JOB_ID , SALARY FROM EMPLOYEES WHERE SALARY < (SELECT MIN(SALARY) FROM EMPLOYEES WHERE JOB_ID IN 'SA_REP');
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENT. 1 point

- [ ] A. SELECT DEPARTMENT_NAME , REGION_NAME FROM DEPARTMENTS NATURAL JOIN REGIONS NATURAL JOIN LOCATIONS NATURAL JOIN COUNTRIES;
- [ ] B. SELECT DEPARTMENT_NAME , REGION_NAME FROM DEPARTMENTS NATURAL JOIN LOCATIONS NATURAL JOIN COUNTRIES NATURAL JOIN REGIONS;
- [ ] C. SELECT DEPARTMENT_NAME , REGION_NAME FROM DEPARTMENTS NATURAL JOIN COUNTRIES NATURAL JOIN LOCATIONS NATURAL JOIN REGIONS;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STAEMENT. 1 point

- [ ] A. CREATE TABLE T6(INV_NO NUMBER(4),IND_DT DATE DEFAULT SYSDATE,AMOUNT NUMBER(7,2));
- [ ] B. CREATE TABLE T6(INV_NO NUMBER(4),IND_DT DATE DEFAULT CURRENT_DATE,AMOUNT NUMBER(7,2));
- [ ] C. CREATE TABLE T6(INV_NO NUMBER(4),IND_DT DATE DEFAULT TIMESTAMP,AMOUNT NUMBER(7,2));
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. SELECT MIN(SALARY), MAX(HIRE_DATE), COUNT(FIRST_NAME) FROM EMPLOYEES;
- [ ] B. SELECT MIN(HIRE_DATE) , COUNT(HIRE_DATE) , SUM(HIRE_DATE) FROM EMPLOYEES;
- [ ] C. SELECT AVG(FIRST_NAME) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS ? 1 point

- [ ] A. SELECT COUNT(FIRST_NAME || LAST_NAME) FROM EMPLOYEES;
- [ ] B. SELECT SUM(SALARY * 12) FROM EMPLOYEES;
- [ ] C. SELECT AVG(SALARY * COMMISSION_PCT) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS. 1 point

- [ ] A. SELECT FIRST_NAME || ' has joined the company on ' || TO_CHAR(HIRE_DATE,'dddth "day of year" Year'.' ') FROM EMPLOYEES;
- [ ] B. SELECT TO_CHAR(SYSDATE,' "Today is" jsp "day after Julian.") FROM DUAL;
- [ ] C. SELECT TO_DATE(SYSDATE,' "We are currently in" wwspth " week of year." ') FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH OF FOLLOWING COMMANDS WILL GENERATE CARTESIAN PRODUCT. 1 point

- [ ] A. SELECT * FROM DEPARTMENTS NATURAL JOIN LOCATIONS;
- [ ] B. SELECT * FROM DEPARTMENTS NATURAL JOIN COUNTRIES;
- [ ] C. SELECT * FROM DEPARTMENTS CROSS JOIN LOCATIONS;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SELECT STATEMENT. 1 point

- [ ] A. SELECT JOB_ID , SUM(SALARY) FROM EMPLOYEES WHERE SUM(SALARY) > 10000 GROUP BY JOB_ID;
- [ ] B. SELECT JOB_ID , SUM(SALARY) FROM EMPLOYEES HAVING SUM(SALARY) > 10000 GROUP BY JOB_ID;
- [ ] C. SELECT JOB_ID , SUM(SALARY) FROM EMPLOYEES GROUP BY JOB_ID WHERE SUM(SALARY) > 10000;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. SELECT SUBSTR(HIRE_DATE,4,3) , AVG(SALARY) FROM EMPLOYEES GROUP BY SUBSTR(HIRE_DATE,4,3);
- [ ] B. SELECT SUBSTR(HIRE_DATE,4,3) , AVG(SALARY) FROM EMPLOYEES GROUP BY SUBSTR(HIRE_DATE,-6,3);
- [ ] C. SELECT SUBSTR(HIRE_DATE,4,3) , AVG(SALARY) FROM EMPLOYEES GROUP BY TO_CHAR(HIRE_DATE,'MON');
- [ ] D. All of the above.
- [ ] E. None of the above.

ALTER TABLE ... ENABLE ROW MOVEMENT COMMAND IS USED TO 1 point

- [ ] A. ALLOW FLASHBACK OPERATION ON TABLE
- [ ] B. DISALLOW FLASHBACK OPERATION ON TABLE
- [ ] C. ALLOW/DISALLOW FLASHBACK OPERATION ON TABLE
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY DEPARTMENT_ID AND AVERAGE SALARY FOR THE DEPARTMENT TO WHICH HIGHEST AVERAGE SALARY IS PAID. 1 point

- [ ] A. SELECT DEPARTMENT_ID , AVG(SALARY) FROM EMPLOYEES WHERE AVG(SALARY) = (SELECT MAX(AVG(SALARY)) FROM EMPLOYEES);
- [ ] B. SELECT DEPARTMENT_ID , AVG(SALARY) FROM EMPLOYEES HAVING AVG(SALARY) = (SELECT MAX(AVG(SALARY)) FROM EMPLOYEES) GROUP BY DEPARTMENT_ID;
- [ ] C. SELECT DEPARTMENT_ID , AVG(SALARY) FROM EMPLOYEES HAVING AVG(SALARY) = (SELECT MAX(AVG(SALARY)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

ANY CHANGES DONE THRU DML STATEMENTS ARE LIMITED TO ... 1 point

- [ ] A. TABLE
- [ ] B. DATABASE
- [ ] C. RAM/MEMORY
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT CREATE TABLE COMMAND. 1 point

- [ ] A. CREATE TABLE T3(BANK VARCHAR2(15),AMOUNT NUMBER(10,2),FD_DATE DATE,DURATION INTERVAL YEAR TO MONTH);
- [ ] B. CREATE TABLE T3(BANK VARCHAR(15),AMOUNT NUMBER(10,2),FD_DATE DATE,DURATION INTERVAL YEAR TO MONTHS);
- [ ] C. CREATE TABLE T3(BANK VARCHAR2(15),AMOUNT NUMBER(10),FD_DATE DATE,DURATION INTERVAL YEARS TO MONTHS);
- [ ] D. All of the above.
- [ ] E. None of the above.

BELOW COMMAND(S) WILL DISPLAY 15 SINGLE TIME ONLY. 1 point

- [ ] A. SELECT 10 + 5 FROM DUAL;
- [ ] B. SELECT DISTINCT (2 + 3) * 3 FROM REGIONS;
- [ ] C. SELECT UNIQUE 25 - 10 FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY INFO. ABOUT THOSE 'IT_PROG' WHO GETS HIGHER SALARY THAN AVERAGE SALARY PAID TO COMPANY'S ALL EMPLOYEES. 1 point

- [ ] A. SELECT FIRST_NAME , JOB_ID , SALARY FROM EMPLOYEES WHERE JOB_ID = 'IT_PROG' AND SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES);
- [ ] B. SELECT FIRST_NAME , JOB_ID , SALARY FROM EMPLOYEES WHERE JOB_ID = 'IT_PROG' AND SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES WHERE JOB_ID = 'IT_PROG');
- [ ] C. SELECT FIRST_NAME , JOB_ID , SALARY FROM EMPLOYEES WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES WHERE JOB_ID = 'IT_PROG');
- [ ] D. All of the above.
- [ ] E. None of the above.

INCREMENT BASED ON COMPLETED YEARS. FOR 0-25 YEARS 25% , FOR 26-30 YEARS +3000, FOR 31 OR MORE YEARS 50%. 1 point

- [ ] A. SELECT CASE WHEN TRUNC(MONTHS_BETWEEN(SYSDATE,HIRE_DATE)/12) <= 25 THEN SALARY * 1.25 WHEN TRUNC(MONTHS_BETWEEN(SYSDATE,HIRE_DATE)/12) BETWEEN 26 AND 30 THEN SALARY + 3000 ELSE SALARY * .5 END NEW_SALARY FROM EMPLOYEES;
- [ ] B. SELECT CASE WHEN TRUNC((SYSDATE-HIRE_DATE)/365) <= 25 THEN SALARY * 1.25 WHEN TRUNC((SYSDATE-HIRE_DATE)/365) BETWEEN 26 AND 30 THEN SALARY + 3000 ELSE SALARY * .5 END NEW_SALARY FROM EMPLOYEES;
- [ ] C. SELECT CASE WHEN TRUNC(MONTHS_BETWEEN(SYSDATE,HIRE_DATE)/12) <= 25 THEN SALARY * 1.25 WHEN TRUNC(MONTHS_BETWEEN(SYSDATE,HIRE_DATE)/12) BETWEEN 26 AND 30 THEN SALARY + 3000 ELSE SALARY * 1.5 END NEW_SALARY FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. CONSTRAINT NAME MUST BE UNIQUE PER TABLE.
- [ ] B. CONSTRAINT NAME MUST BE UNIQUE PER USER.
- [ ] C. CONSTRAINT NAME MUST BE UNIQUE PER DATABASE.
- [ ] D. All of the above.
- [ ] E. None of the above.

WHAT TYPE OF INFO. CAN BE STORED IF WE CHOOSE BFILE DATA TYPE? 1 point

- [ ] A. PHOTOGRAPH, SIGNATURE
- [ ] B. AUDIO , VIDEO FILE
- [ ] C. TEXT
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY THOSE WHO GETS LOWEST SALARY FOR THEIR DEPARTMENT. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY FROM EMPLOYEES WHERE SALARY IN (SELECT MIN(SALARY) FROM EMPLOYEES);
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY FROM EMPLOYEES WHERE SALARY IN (SELECT MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY FROM EMPLOYEES WHERE (SALARY,DEPARTMENT_ID) IN (SELECT DEPARTMENT_ID , MIN(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT RTRIM(LTRIM('12-MAY-98','18'),'9') FROM DUAL; 1 point

- [ ] A. 2-MAY-8
- [ ] B. 2-MAY-98
- [ ] C. 2-MAY-
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH DATA TYPE IS SUITABLE TO STORE TRAVELLING TIME? 1 point

- [ ] A. TIMESTAMP WITH TIME ZONE
- [ ] B. INTERVAL DAY TO SECOND
- [ ] C. INTERVAL YEAR TO MONTH
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE INCORRECT CONSTRAINT TYPE. 1 point

- [ ] A. NOT NULL
- [ ] B. UNIQUE
- [ ] C. DISTINCT
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH IS NOT DML STATEMENT? 1 point

- [ ] A. MERGE
- [ ] B. INSERT
- [ ] C. DROP
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAYMANAGER_ID AND COUNT OF EMPLOYEES REPORTING TO THEM FOR ALL MANAGERS EXCEPT THOSE TO WHOM HIGHEST OR LOWEST NO. OF EMPLOYEES REPORT. 1 point

- [ ] A. SELECT MANAGER_ID , COUNT(*) FROM EMPLOYEES GROUP BY MANAGER_ID HAVING COUNT(*) NOT IN ((SELECT MIN(COUNT(*)) FROM EMPLOYEES GROUP BY MANAGER_ID), (SELECT MAX(COUNT(MANAGER_ID)) FROM EMPLOYEES GROUP BY MANAGER_ID));
- [ ] B. SELECT MANAGER_ID , COUNT(*) FROM EMPLOYEES GROUP BY MANAGER_ID HAVING COUNT(MANAGER_ID) NOT IN ((SELECT MAX(COUNT(*)) FROM EMPLOYEES GROUP BY MANAGER_ID), (SELECT MIN(COUNT(MANAGER_ID)) FROM EMPLOYEES GROUP BY MANAGER_ID));
- [ ] C. SELECT MANAGER_ID , COUNT(*) FROM EMPLOYEES GROUP BY MANAGER_ID HAVING COUNT(*) NOT IN ((SELECT MAX(COUNT(*)) FROM EMPLOYEES GROUP BY MANAGER_ID), (SELECT MIN(COUNT(*)) FROM EMPLOYEES GROUP BY MANAGER_ID));
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT COUNT(DEPARTMENT_ID) , MIN(COUNT(*)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] B. SELECT COUNT(*) , MIN(COUNT(*)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] C. SELECT COUNT(JOB_ID) , MIN(COUNT(*)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY INFO. ABOUT THE EMPLOYEE WHO GETS LOWEST COMMISSION AMOUNT. 1 point

- [ ] A. SELECT FIRST_NAME , SALARY * COMMISSION_PCT COMM_AMOUNT FROM EMPLOYEES WHERE COMM_AMOUNT = (SELECT MIN(SALARY * COMMISSION_PCT) FROM EMPLOYEES);
- [ ] B. SELECT FIRST_NAME , SALARY * COMMISSION_PCT COMM_AMOUNT FROM EMPLOYEES WHERE SALARY * COMMISSION_PCT = (SELECT MIN(SALARY * COMMISSION_PCT) FROM EMPLOYEES);
- [ ] C. SELECT FIRST_NAME , SALARY * COMMISSION_PCT COMM_AMOUNT FROM EMPLOYEES WHERE COMM_AMOUNT = MIN(SALARY * COMMISSION_PCT);
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT upper(LOWER(iNITCAP('ShRi rAm'))) FROM DUAL; 1 point

- [ ] A. SHRI RAM
- [ ] B. Shri Ram
- [ ] C. shri ram
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT REPLACE(INITCAP(TRANSLATE('HIREN','H','V')),'n','ndra') FROM DUAL; 1 point

- [ ] A. Viren
- [ ] B. VIREndra
- [ ] C. Virendra
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONSTRAINT CAN BE USED SINGLE TIME PER TABLE. 1 point

- [ ] A. PRIMARY KEY
- [ ] B. CHECK
- [ ] C. NOT NULL
- [ ] D. All of the above.
- [ ] E. None of the above.

CREATE SEQUENCE SEQ1; CHOOSE CORRECT CREATE TABLE COMMAND. 1 point

- [ ] A. CREATE TABLE T7(RNO NUMBER(2) DEFAULT SEQ1,NM VARCHAR2(10));
- [ ] B. CREATE TABLE T7(RNO NUMBER(2) DEFAULT SEQ1,CURRVAL,NM VARCHAR2(10));
- [ ] C. CREATE TABLE T7(RNO NUMBER(2) DEFAULT SEQ1.NEXTVALUE,NM VARCHAR2(10));
- [ ] D. All of the above.
- [ ] E. None of the above.

EVEN IF, WHEN WE CHANGE PLACEMENT OF QUERIES WHICH SET OPERATORS RETURNS IDENTICAL RESULT WITH IDENTICAL SORTED ORDER.

1 point

- [ ] A. UNION
- [ ] B. UNION ALL
- [ ] C. INTERSECT
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH COMMANDS WILL PRODUCE IDENTICAL RESULTS FOR BOTH FUNCTIONS?

1 point

- [ ] A. SELECT TRUNC(3.99) , ROUND(3.49) FROM DUAL;
- [ ] B. SELECT TRUNC(1234.59,1), ROUND(1234.45,1) FROM DUAL;
- [ ] C. SELECT TRUNC(12345,-2) , ROUND(12345,-2) FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

REGIONS TABLE HAS 2 COLUMNS AND 4 ROWS. COUNTRIES TABLE HAS 3 COLUMNS AND 4 ROWS. BOTH TABLES HAS SINGLE COMMON COLUMNS. REGIONS TABLE HAS 2 EXTRA/UNMATCHED ROWS. SELECT * FROM COUNTRIES NATURAL JOIN REGIONS; HOW MANY TOTAL COLUMNS AND ROWS WILL GET DISPLAYED.

1 point

- [ ] A. 4 , 4
- [ ] B. 5 , 8
- [ ] C. 4 , 8
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH SET OPERATOR ONLY CHANGES THE SORTED ORDER BUT RETURNS IDENTICAL OUTPUT IF WE CHANGE THE QUERY PLACEMENT INSIDE COMPOUND STATEMENT. 1 point

- [ ] A. UNION
- [ ] B. MINUS
- [ ] C. UNION ALL
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STAEMENT. 1 point

- [ ] A. CREATE TABLE T11(INV_NO NUMBER(4) NOT NULL, UNIQUE ,IND_DT DATE DEFAULT SYSDATE,AMOUNT NUMBER(7,2));
- [ ] B. CREATE TABLE T11(INV_NO NUMBER(4) UNIQUE NOT NULL,IND_DT DATE DEFAULT CURRENT_DATE,AMOUNT NUMBER(7,2));
- [ ] C. CREATE TABLE T11(INV_NO NUMBER(4) PRIMARY KEY,IND_DT DATE DEFAULT SYSDATE,AMOUNT NUMBER(7,2));
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT DEPARTMENT_ID , MIN(COUNT(*)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] B. SELECT MIN(SALARY) , MIN(AVG(SALARY)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] C. SELECT AVG(SALARY) , MIN(AVG(SALARY)) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY DATA OF EMPLOYEES TABLE WHO WORKS AS IT_PROG. 1 point

- [ ] A. SELECT * FROM EMPLOYEES WHERE JOB_ID = IT_PROG;
- [ ] B. SELECT * FROM EMPLOYEES WHERE JOB_ID = "IT_PROG";
- [ ] C. SELECT * FROM EMPLOYEES WHERE 'IT_PROG' = JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONSTRAINT IS COMBINATION OF NOT NULL AND UNIQUE. 1 point

- [ ] A. PRIMARY KEY
- [ ] B. REFERENCES
- [ ] C. FOREIGN KEY
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT COMMANDS. 1 point

- [ ] A. SELECT FIRST_NAME || Q' A 's monthly salary is Rs.a ' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] B. SELECT FIRST_NAME || Q' | 's monthly salary is Rs.! ' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] C. SELECT FIRST_NAME || Q' 6 's monthly salary is Rs.9 ' || SALARY || '/-.' FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_NAME FROM EMPLOYEES NATURAL JOIN DEPARTMENTS;
- [ ] B. SELECT * FROM REGIONS NATURAL JOIN COUNTRIES;
- [ ] C. SELECT CITY , REGION_NAME FROM LOCATIONS NATURAL JOIN REGION_NAME;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SELECT STATEMENT. 1 point

- [ ] A. SELECT SUM(SALARY)/COUNT(SALARY) FROM EMPLOYEES WHERE NOT SALARY > 10000;
- [ ] B. SELECT AVG(SALARY) , MAX(DISTINCT COMMISSION_PCT) FROM EMPLOYEES WHERE JOB_ID LIKE '%MAN';
- [ ] C. SELECT MIN(SALARY) , MAX(HIRE_DATE) , AVG(EMAIL) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH SET OPERATOR RETURNS DIFFERENT RESULT IF WE CHANGE PLACE OF FIRST QUERY OF COMPOUND STATEMENT. 1 point

- [ ] A. UNION ALL
- [ ] B. INTERSECT
- [ ] C. MINUS
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT SYSDATE , TO_CHAR(SYSDATE,'D DY DAy') FROM DUAL;
- [ ] B. SELECT SYSDATE , TO_CHAR(SYSDATE,'MM MI rm') FROM DUAL;
- [ ] C. SELECT SYSDATE , TO_CHAR(SYSDATE,'ss sSSSs ww') FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT LOWER(JOB_ID) , AVG(SALARY) FROM EMPLOYEES GROUP BY LOWER(JOB_ID);
- [ ] B. SELECT JOB_ID , AVG(SALARY) FROM EMPLOYEES GROUP BY LOWER(JOB_ID);
- [ ] C. SELECT LOWER(JOB_ID) , AVG(SALARY) FROM EMPLOYEES GROUP BY JOB_ID;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONSTRAINT(S) CAN BE USED SINGLE TIME PER COLUMN. 1 point

- [ ] A. PRIMARY KEY
- [ ] B. UNIQUE
- [ ] C. REFERENCES
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT LENGTH(TRIM(' R A M ')) FROM DUAL; (STRING HAS 2 SPACES EACH AS PREFIX AS WELL AS SUFFIX) 1 point

- [ ] A. 3
- [ ] B. 7
- [ ] C. 9
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT STATEMENTS. 1 point

- [ ] A. SELECT DEPARTMENT_ID , COUNT(*) FROM EMPLOYEES GROUP BY DEPARTMENT_ID HAVING COUNT(*) <= 3;
- [ ] B. SELECT DEPARTMENT_ID , COUNT(*) FROM EMPLOYEES GROUP BY DEPARTMENT_ID HAVING AVG(SALARY) > 8000;
- [ ] C. SELECT DEPARTMENT_ID , COUNT(*) FROM EMPLOYEES GROUP BY DEPARTMENT_ID HAVING DEPARTMENT_ID IN (80,90,50);
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONVERSION FUNCTION IS MOST SUITED WHEN DATA TYPES OF CORRESPONDING ITEMS DO NOT MATCH. 1 point

- [ ] A. TO_NUMBER
- [ ] B. TO_DATE
- [ ] C. TO_CHAR
- [ ] D. All of the above.
- [ ] E. None of the above.

INCREMENT POLICY ON DEPARTMENT_ID VALUE. FOR 20 +2000 , FOR 50 +1500 , FOR 80 +1000 REMAINING +500 1 point

- [ ] A. SELECT DECODE(DEPARTMENT_ID,20,SALARY+2000,50;1500+SALARY,80,SALARY+1000,500 +SALARY) FROM EMPLOYEES;
- [ ] B. SELECT DECODE(DEPARTMENT_ID,20,SALARY+2000,50,1500+SALARY,80,SALARY+1000 ELSE 500+SALARY) FROM EMPLOYEES;
- [ ] C. SELECT DECODE(DEPARTMENT_ID,20,SALARY+2000,50,1500+SALARY,80,SALARY+1000,500 +SALARY) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHAT IS CONCURRENCY ? 1 point

- [ ] A. ALLOWING DRL ON SAME TABLE TO MULTIPLE SESSION AT THE SAME TIME.
- [ ] B. ALLOWING DML ON SAME TABLE TO MULTIPLE SESSION AT THE SAME TIME.
- [ ] C. ALLOWING DDL ON SAME TABLE TO MULTIPLE SESSION AT THE SAME TIME.
- [ ] D. All of the above.
- [ ] E. None of the above.

WHY WE CAN'T USE SUMMARY FUNCTION DIRECTLY IN 'WHERE' CLAUSE. 1 point

- [ ] A. BECAUSE SUMMARY ITEM IS GENERATED AFTER 'WHERE' CLAUSE EXECUTION COMPLETES.
- [ ] B. BECAUSE 'WHERE' CLAUSE SUPPORTS ONLY NON-SUMMARY ITEMS.
- [ ] C. BECAUSE 'WHERE' CLAUSE WORKS ON TABLE'S ORIGINAL ROWS.
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CLAUSE IF USED IN COMPOUND STATEMENT MUST BE PLACED AT THE END OF COMPOUND STATEMENT ONCE ONLY. 1 point

- [ ] A. ORDER BY
- [ ] B. OFFSET
- [ ] C. FETCH
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SYNTACTICAL COMMANDS. 1 point

- [ ] A. SELECT SALARY , TO_CHAR(SALARY,'L99D000G99','NLS_DUAL_CURRENCY=@ NLS_NUMERIC_CHARACTERS=~!') FROM EMPLOYEES;
- [ ] B. SELECT SALARY , TO_CHAR(SALARY,'U99G000D99','NLS_DUAL_CURRENCY=@ NLS_NUMERICAL_CHARACTERS=~!') FROM EMPLOYEES;
- [ ] C. SELECT SALARY , TO_CHAR(SALARY,'U99G000D99','NLS_DUAL_CURRENCY=@ NLS_NUMERIC_CHARACTERS=~!') FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT CREATE TABLE COMMAND. 1 point

- [ ] A. CREATE TABLE T2(FROM VARCHAR2(20),TRAVEL_DURATION INTERVAL DAY TO SECONDS,TO VARCHAR(20));
- [ ] B. CREATE TABLE T2(FROM_CITY VARCHAR2(20),TRAVEL_DURATION INTERVAL DAY TO SECOND,TO_CITY VARCHAR(20));
- [ ] C. CREATE TABLE T2(FROM_CITY VARCHAR(20),TRAVEL_DURATION INTERVAL DAYS TO SECOND,TO_CITY VARCHAR2(20));
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT COMMANDS. 1 point

- [ ] A. SELECT "SHRI" || (210 + 210) FROM DUAL;
- [ ] B. SELECT DISTINCT SHRI || 210 * 2 FROM REGIONS;
- [ ] C. SELECT UNIQUE 'SHRI' || (840 / 2) FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH COMMAND(S) ARE CORRECT ? 1 point

- [ ] A. SELECT FIRST_NAME 'works as' || job_id FROM EMPLOYEES;
- [ ] B. SELECT FIRST_NAME || " works as " || job_id FROM EMPLOYEES;
- [ ] C. SELECT FIRST_NAME || ' works as ' || job_id || '.' "Employee Info." FROM EMPLOYEES;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY FIRST_NAME , DEPARTMENT_ID , SALARY AND HIGHEST SALARY PAID TO CORRESPONDING DEPARTMENT. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES GROUP BY DEPARTMENT_ID;
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES GROUP BY FIRST_NAME , DEPARTMENT_ID, SALARY;
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY , MAX(SALARY) FROM EMPLOYEES NATURAL JOIN (SELECT DEPARTMENT_ID , MAX(SALARY) GROUP BY DEPARTMENT_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE CORRECT SELECT STATEMENT. 1 point

- [ ] A. SELECT DEPARTMENT_NAME , CITY FROM DEPARTMENTS INNER JOIN LOCATIONS USING (LOCATION_ID);
- [ ] B. SELECT DEPARTMENT_NAME , CITY FROM DEPARTMENTS INNER JOIN LOCATIONS USING (DEPARTMENT_ID);
- [ ] C. SELECT DEPARTMENT_NAME , CITY FROM DEPARTMENTS JOIN LOCATIONS USING (LOCATION_ID);
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH SUB TYPE WILL DISPLAY EXTRA ROWS OF ONLY EMPLOYEES TABLE. SELECT FIRST_NAME , DEPARTMENT_NAME FROM EMPLOYEES _____ OUTER JOIN DEPARTMENTS USING (DEPARTMENT_ID); 1 point

- [ ] A. LEFT
- [ ] B. RIGHT
- [ ] C. FULL
- [ ] D. All of the above.
- [ ] E. None of the above.

CHOOSE SYNTACTICALLY CORRECT STATEMENTS ? 1 point

- [ ] A. SELECT TO_DATE('13-MAR-1973') - TO_DATE('71-JAN-15','DD-MM-RR') FROM DUAL;
- [ ] B. SELECT TO_DATE('13MARCH1973') - TO_DATE('JAN-15-71','MON-DD-RR') FROM DUAL;
- [ ] C. SELECT TO_DATE('13MARCH1973') - TO_DATE('15-JANUARY-71','DD-MON-RR') FROM DUAL;
- [ ] D. All of the above.
- [ ] E. None of the above.

DISPLAY INFO. ABOUT THE EMPLOYEES WHO GETS HIGHER SALARY THAN THE AVERAGE SALARY PAID TO DEPARTMENT 80's EMPLOYEES. 1 point

- [ ] A. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY FROM EMPLOYEES WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES) AND DEPARTMENT_ID = 80;
- [ ] B. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY FROM EMPLOYEES WHERE SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES WHERE DEPARTMENT_ID = 80);
- [ ] C. SELECT FIRST_NAME , DEPARTMENT_ID , SALARY FROM EMPLOYEES WHERE DEPARTMENT_ID = 80 AND SALARY > (SELECT AVG(SALARY) FROM EMPLOYEES WHERE DEPARTMENT_ID = 80;
- [ ] D. All of the above.
- [ ] E. None of the above.

SELECT CONCAT('Amar Akbar',Anthony) FROM DUAL; 1 point

- [ ] A. Amar AkbarAnthony
- [ ] B. Amar Akbar Anthony
- [ ] C. AmarAkbarAnthony
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONSTRAINT COMBINATION CAN'T BE USED WITH SINGLE COLUMN. 1 point

- [ ] A. PRIMARY KEY , REFERENCES
- [ ] B. PRIMARY KEY , NOT NULL
- [ ] C. PRIMARY KEY , UNIQUE
- [ ] D. All of the above.
- [ ] E. None of the above.

WHICH CONSTRAINT(S) CAN BE USED MULTIPLE TIMES PER COLUMN. 1 point

- [ ] A. PRIMARY KEY
- [ ] B. UNIQUE
- [ ] C. REFERENCES
- [ ] D. All of the above.
- [ ] E. None of the above.

Submit Clear form

This form was created outside of your domain. Report Abuse - Terms of Service - Privacy Policy

Google Forms