Programming Constructs :

1. Create a block that selects the maximum department ID in the departments table and stores it in the v_max_deptno variable. Display the maximum department ID.
2. Modify the block that you created in step 1 to insert a new department into the departments table.
    a. Declare two variables: v_dept_name of type departments.department_name and v_dept_id of type NUMBER
    b. Assign 'Education' to v_dept_name in the declarative section.
    c. You have already retrieved the current maximum department number from the departments table. Add 10 to it and assign the result to v_dept_id.
    d. Include an INSERT statement to insert data into the department_name, department_id, and location_id columns of the departments table.
    e. Use values in dept_name and dept_id for department_name and department_id, respectively, and use NULL for location_id.

3. Create a block which selects the salary of a given employee_id and raise the salary on the basis of following table:

Range	Increment
1000-5000	10%
5001-10000	8%
100001-15000	6%
15001-20000	5%
>20000	No increment

Procedure:

1. Create a procedure called ADD_JOB to insert a new job into the JOBS table. Provide the ID and job title using two parameters.
2. Create a procedure called GET_EMPLOYEE to query the EMPLOYEES table, provided with the EMPLOYEE_ID and returns SALARY and JOB_ID.

Function:

1. Create a function called GET_ANNUAL_COMP to return the annual salary computed from an employee's monthly salary and commission passed as parameters.
    a) Create the GET_ANNUAL_COMP function, which accepts parameter values for the monthly salary and commission. Either or both values passed can be NULL, but the function should still return a non-NULL annual salary. Use the following basic formula to calculate the annual salary:
    (salary*12) + (commission_pct*salary*12)

b) Use the function in a SELECT statement against the EMPLOYEES table for employees in department 30.

Trigger:

1. Create a table with following script:
   CREATE TABLE DML_LOG (log_date DATE , action VARCHAR(50));

2. Create a trigger, EVAL_CHANGE_TRIGGER, which adds a row to the table DML_LOG whenever an INSERTor DELETE statement changes the DEPARTMENTS table.
   Note: log_date column stores the date of DML operation and action column stores the event name (INSERT or DELETE).