UCS310

DBMS

Assignment PL/SQL - 3

Q1.

```
CREATE TABLE emp (
  empno NUMBER PRIMARY KEY,
  ename VARCHAR2(50) UNIQUE,
  job VARCHAR2(10) CHECK(job IN ('Prof', 'AP', 'Lect')),
  sal NUMBER NOT NULL,
  deptno NUMBER DEFAULT 10
);

BEGIN
  INSERT INTO emp VALUES (1, 'John', 'Prof', 5000, 10); -- violates unique constraint on
ename
  INSERT INTO emp VALUES (2, 'Mary', 'Manager', NULL, 20); -- violates not null constraint on
sal
  INSERT INTO emp VALUES (3, 'Bob', 'Lect', 3000, 30); -- violates check constraint on job
  INSERT INTO emp VALUES (4, 'Sarah', 'AP', 4000, 40);
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Error: ' || SQLCODE || ' - ' || SQLERRM);
END;

SELECT constraint_name FROM user_constraints WHERE table_name = 'EMP';
```

Q2.

```
CREATE OR REPLACE PROCEDURE raise_salary (
  empid IN NUMBER,
  bonus IN NUMBER
)
IS
  v_salary emp.sal%TYPE;
BEGIN
  SELECT sal INTO v_salary FROM emp WHERE empno = empid;
  v_salary := v_salary + bonus;

  UPDATE emp SET sal = v_salary WHERE empno = empid;
  COMMIT;
DBMS_OUTPUT.PUT_LINE('Salary updated successfully');
END;
```

```
BEGIN
  raise_salary(1001, 1000); -- increase salary of employee with empid=1001 by 1000
END;
```

Q3.

```
CREATE OR REPLACE PROCEDURE fire_employee (
  empid IN NUMBER
)
IS
BEGIN
  DELETE FROM emp WHERE empno = empid;
  COMMIT;
  DBMS_OUTPUT.PUT_LINE('Employee with empid=' || empid || ' has been fired');
END;

BEGIN
  fire_employee(1001); -- delete employee with empid=1001
END;
```

Q4.

```
CREATE OR REPLACE FUNCTION get_employee_count
RETURN NUMBER
IS
  v_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO v_count FROM emp;
  RETURN v_count;
END;
```

Q5.

```
CREATE OR REPLACE FUNCTION add_numbers (
  num1 IN NUMBER,
  num2 IN NUMBER
)
RETURN NUMBER
IS
  v_result NUMBER;
BEGIN
  v_result := num1 + num2;
  RETURN v_result;
END;
/
DECLARE
  v_sum NUMBER;
BEGIN
```

```
  v_sum := add_numbers(10, 20);
  DBMS_OUTPUT.PUT_LINE('The sum of 10 and 20 is: ' || v_sum);
END;
/
```

Q9.

```
CREATE OR REPLACE TRIGGER no_emp_op_on_sunday
BEFORE INSERT OR UPDATE OR DELETE
ON emp
BEGIN
  IF TO_CHAR(SYSDATE, 'Dy') = 'Sun' THEN
    RAISE_APPLICATION_ERROR(-20001, 'No operations on EMP table are allowed on
Sundays');
  END IF;
END;
```

Q10.

```
CREATE OR REPLACE TRIGGER no_higher_commission
BEFORE INSERT OR UPDATE OF comm ON emp
FOR EACH ROW
BEGIN
  IF :NEW.comm > :NEW.sal THEN
    RAISE_APPLICATION_ERROR(-20001, 'Commission cannot be greater than salary');
  END IF;
END;
```

Q11.

```
CREATE OR REPLACE TRIGGER emp_pk_trigger
BEFORE INSERT OR UPDATE OF eno ON emp
FOR EACH ROW
DECLARE
  duplicate_count NUMBER;
BEGIN
  SELECT COUNT(*) INTO duplicate_count FROM emp WHERE eno = :NEW.eno;
  IF duplicate_count > 0 THEN
    RAISE_APPLICATION_ERROR(-20001, 'Duplicate value not allowed for ENO');
  END IF;
END;
/
```