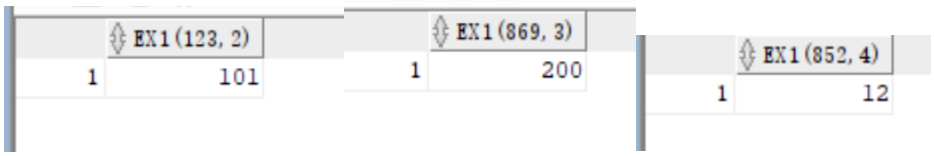
/**********question1**********/

```
CREATE OR REPLACE FUNCTION ex1(num IN NUMBER, divisor IN NUMBER) RETURN NUMBER IS
    result NUMBER := 0;
    digit NUMBER;
    power NUMBER := 1;
    temp_num NUMBER;
BEGIN
    IF divisor = 0 THEN
        RETURN NULL;
    END IF;

    temp_num := num;
    WHILE temp_num > 0 LOOP
        digit := MOD(temp_num, 10);
        result := result + MOD(digit, divisor) * power;
        temp_num := TRUNC(temp_num / 10);
        power := power * 10;
    END LOOP;

    RETURN result;
END ex1;

SELECT ex1(852, 4) FROM dual;
SELECT ex1(869, 3) FROM dual;
SELECT ex1(123, 2) FROM dual;
```



	EX1(123, 2)
1	101

	EX1(869, 3)
1	200

	EX1(852, 4)
1	12

/**********question2**********/

```
CREATE OR REPLACE PROCEDURE get_median_salary (p_category IN NIKOVITS.SAL_CAT.category%TYPE)
IS
    TYPE t_emp_record IS RECORD (
        ename NIKOVITS.EMP.ename%TYPE,
        sal     NIKOVITS.EMP.sal%TYPE
    );
    TYPE t_emp_table IS TABLE OF t_emp_record INDEX BY PLS_INTEGER;
    v_emp_data t_emp_table;
    v_median_salary NUMBER;
    v_median_index NUMBER;
BEGIN
    SELECT e.ename, e.sal
    BULK COLLECT INTO v_emp_data
    FROM NIKOVITS.EMP e
    JOIN NIKOVITS.SAL_CAT c ON e.sal BETWEEN c.lowest_sal AND c.highest_sal
    WHERE c.category = p_category
    ORDER BY e.sal;
    IF v_emp_data.COUNT = 0 THEN
        DBMS_OUTPUT.PUT_LINE('No employees found in this category.');
        RETURN;
    END IF;
    v_median_index := CEIL(v_emp_data.COUNT / 2);
    v_median_salary := v_emp_data(v_median_index).sal;
    DBMS_OUTPUT.PUT_LINE('Median Salary: ' || TO_CHAR(v_median_salary));
    DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_emp_data(v_median_index).ename);
END;
/
call    get_median_salary('2');
```

```
Procedure GET_MEDIAN_SALARY 已编译

Median Salary: 1250
Employee Name: MARTIN
```

/***/

/**********************************question3***/

```
create table exam as select * from NIKOVITS.EMP;
CREATE OR REPLACE PROCEDURE assign_commission (p_manager_name IN exam.ename%TYPE)
IS
    CURSOR emp_cursor IS
    SELECT empno, ename, mgr, comm
        FROM exam
        ORDER BY ename FOR UPDATE;
```

```
    v_counter NUMBER := 0;
    v_mgr_empno exam.empno%TYPE;
BEGIN
    SELECT empno INTO v_mgr_empno FROM exam WHERE ename = p_manager_name;
    FOR rec IN emp_cursor LOOP
        v_counter := v_counter + 1;
        IF MOD(v_counter, 2) = 0 THEN
            IF rec.mgr = v_mgr_empno AND rec.comm IS NULL THEN
                UPDATE exam SET comm = 100 WHERE CURRENT OF emp_cursor;
            END IF;
        END IF;
    END LOOP;
    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Manager not found.');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An error occurred: ' || SQLERRM);
        ROLLBACK;
END;
/
call assign_commission('BLAKE');
```

	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7369	SMITH	CLERK	7902	17-12月-80	800	100	20
2	7499	ALLEN	SALESMAN	7698	20-2月 -81	1600	300	30
3	7521	WARD	SALESMAN	7698	22-2月 -81	1250	500	30
4	7566	JONES	MANAGER	7839	02-4月 -81	2975	(null)	20
5	7654	MARTIN	SALESMAN	7698	28-9月 -81	1250	1400	30
6	7698	BLAKE	MANAGER	7839	01-5月 -81	4250	(null)	30
7	7782	CLARK	MANAGER	7839	09-6月 -81	2450	200	10
8	7788	SCOTT	ANALYST	7566	09-12月-82	3000	(null)	20
9	7839	KING	PRESIDENT	(null)	17-11月-81	5000	(null)	10
10	7844	TURNER	SALESMAN	7698	08-9月 -81	1500	10	30
11	7876	ADAMS	CLERK	7788	12-1月 -83	1100	(null)	20
12	7900	JAMES	CLERK	7698	03-12月-81	950	100	30
13	7902	FORD	ANALYST	7566	03-12月-81	3000	700	20
14	7934	MILLER	CLERK	7782	23-1月 -82	1300	600	10
15	8001	COOK	MANAGER	7839	09-6月 -81	3800	(null)	50
16	8002	HART	SALESMAN	8001	09-5月 -82	1600	200	50
17	8003	WOLF	CLERK	8001	09-4月 -83	1000	(null)	50

```
/*******************************************************************************/
```

```
/***********************************question4*********************************************/
CREATE OR REPLACE PROCEDURE print_employees_at_location (p_location IN NIKOVITS.DEPT.loc%TYPE)
IS
    location_not_found EXCEPTION;
    no_employees EXCEPTION;
    v_ename NIKOVITS.EMP.ename%TYPE;
    v_job NIKOVITS.EMP.job%TYPE;
    CURSOR emp_cursor IS
        SELECT e.ename, e.job
        FROM NIKOVITS.EMP e
        JOIN NIKOVITS.DEPT d ON e.deptno = d.deptno
        WHERE d.loc = p_location;

    v_dept_count NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_dept_count
    FROM NIKOVITS.DEPT
    WHERE loc = p_location;

    IF v_dept_count = 0 THEN
        RAISE location_not_found;
    END IF;
    OPEN emp_cursor;
    FETCH emp_cursor INTO v_ename, v_job;

    IF emp_cursor%NOTFOUND THEN
        RAISE no_employees;
    ELSE
        LOOP
            DBMS_OUTPUT.PUT_LINE('Employee Name: ' || v_ename || ', Job: ' || v_job);
            FETCH emp_cursor INTO v_ename, v_job;
            EXIT WHEN emp_cursor%NOTFOUND;
        END LOOP;
    END IF;
    CLOSE emp_cursor;

EXCEPTION
    WHEN location_not_found THEN
        DBMS_OUTPUT.PUT_LINE('Error: Location not found.');
    WHEN no_employees THEN
        DBMS_OUTPUT.PUT_LINE('Error: No employees at this location.');
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('An unexpected error occurred: ' || SQLERRM);
```

```
END;
/
call print_employees_at_location('DALLAS');
```

```
Employee Name: SCOTT, Job: ANALYST
Employee Name: FORD, Job: ANALYST
Employee Name: SMITH, Job: CLERK
Employee Name: ADAMS, Job: CLERK
Employee Name: JONES, Job: MANAGER

PL/SQL 过程已成功完成。
```

```
/*******************************************************************************************/

/***********************************question5*********************************************/
WITH RECURSIVE AncestorCTE AS (
    SELECT e.empno, e.ename, e.sal AS Money, CAST(NULL AS DECIMAL(10,2)) AS AncestorMoney, 0 AS AncestorsCount
    FROM NIKOVITS.EMP e

    UNION ALL
    SELECT e.empno, e.ename, e.sal, a.AncestorMoney + p.sal, a.AncestorsCount + 1
    FROM AncestorCTE a
    JOIN NIKOVITS.PARENTOF po ON a.empno = po.Child
    JOIN NIKOVITS.EMP p ON po.Parent = p.empno
),
CREATE OR REPLACE PROCEDURE CalculateAncestorMoney
AS
BEGIN
    WITH AncestorCTE (empno, ename, Money, AncestorMoney, AncestorsCount) AS (
        SELECT e.empno, e.ename, e.sal, CAST(NULL AS DECIMAL(10,2)), 0
        FROM NIKOVITS.EMP e

        UNION ALL
        SELECT e.empno, e.ename, e.sal, a.AncestorMoney + p.sal, a.AncestorsCount + 1
        FROM AncestorCTE a
        JOIN NIKOVITS.PARENTOF po ON a.empno = po.Child
        JOIN NIKOVITS.EMP p ON po.Parent = p.empno
    )
    SELECT a.ename, a.Money, CASE WHEN a.AncestorsCount = 0 THEN NULL ELSE a.AncestorMoney / a.AncestorsCount END AS AvgMoneyOfAncestors
    FROM AncestorCTE a
    WHERE a.Money > a.AncestorMoney / a.AncestorsCount
    ORDER BY a.ename;
END;
//*******************************************************************************************/
```