Stwórz funkcje:

1. Zwracającą nazwę pracy dla podanego parametru id, dodaj wyjątek, jeśli taka praca nie istnieje

```
CREATE OR REPLACE FUNCTION get_job_name(p_job_id VARCHAR2)
RETURN VARCHAR2 IS
   v_job_name jobs.job_title%TYPE;
BEGIN
   SELECT job_title INTO v_job_name
   FROM jobs
   WHERE job_id = p_job_id;

   RETURN v_job_name;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RAISE_APPLICATION_ERROR(-20001, 'Praca o podanym ID nie istnieje.');
END;
/
```

2. zwracającą roczne zarobki (wynagrodzenie 12-to miesięczne plus premia jako wynagrodzenie * commission_pct) dla pracownika o podanym id

```
CREATE OR REPLACE FUNCTION get_annual_salary(p_employee_id NUMBER)
RETURN NUMBER IS
   v_salary employees.salary%TYPE;
   v_commission employees.commission_pct%TYPE;
BEGIN
   SELECT salary, NVL(commission_pct, 0)
   INTO v_salary, v_commission
   FROM employees
   WHERE employee_id = p_employee_id;

   RETURN v_salary * 12 + v_salary * v_commission;
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RAISE_APPLICATION_ERROR(-20002, 'Pracownik o podanym ID nie istnieje.');
END;
/
```

3. biorącą w nawias numer kierunkowy z numeru telefonu podanego jako varchar

```
CREATE OR REPLACE FUNCTION extract_area_code(p_phone VARCHAR2)
RETURN VARCHAR2 IS
   v_area_code VARCHAR2(10);
```

```
BEGIN
   v_area_code := SUBSTR(p_phone, INSTR(p_phone, '(') + 1, INSTR(p_phone, ')') -
INSTR(p_phone, '(') - 1);
   RETURN v_area_code;
END;
/
```

4. Dla podanego w parametrze ciągu znaków zmieniającą pierwszą i ostatnią literę na wielką – pozostałe na małe

```
CREATE OR REPLACE FUNCTION capitalize_first_last(p_string VARCHAR2)
RETURN VARCHAR2 IS
BEGIN
   RETURN INITCAP(SUBSTR(p_string, 1, 1)) ||
          LOWER(SUBSTR(p_string, 2, LENGTH(p_string) - 2)) ||
          INITCAP(SUBSTR(p_string, -1));
END;
/
```

5. Dla podanego peselu - przerabiającą pesel na datę urodzenia w formacie ‘yyyy-mm-dd’

```
CREATE OR REPLACE FUNCTION pesel_to_birthdate(p_pesel VARCHAR2)
RETURN DATE IS
   v_year VARCHAR2(4);
   v_month VARCHAR2(2);
   v_day VARCHAR2(2);
BEGIN
   v_year := CASE
              WHEN SUBSTR(p_pesel, 3, 1) IN ('2', '3') THEN '20' || SUBSTR(p_pesel, 1, 2)
              WHEN SUBSTR(p_pesel, 3, 1) IN ('8', '9') THEN '18' || SUBSTR(p_pesel, 1, 2)
              ELSE '19' || SUBSTR(p_pesel, 1, 2)
             END;
   v_month := MOD(TO_NUMBER(SUBSTR(p_pesel, 3, 2)), 20);
   v_day := SUBSTR(p_pesel, 5, 2);

   RETURN TO_DATE(v_year || '-' || v_month || '-' || v_day, 'YYYY-MM-DD');
END;
/
```

6. Zwracającą liczbę pracowników oraz liczbę departamentów które znajdują się w kraju podanym jako parametr (nazwa kraju). W przypadku braku kraju - odpowiedni wyjątek

```
CREATE OR REPLACE FUNCTION country_stats(p_country_name VARCHAR2)
```

```
RETURN VARCHAR2 IS
    v_emp_count NUMBER;
    v_dept_count NUMBER;
BEGIN
    SELECT COUNT(DISTINCT e.employee_id), COUNT(DISTINCT d.department_id)
    INTO v_emp_count, v_dept_count
    FROM employees e
    JOIN departments d ON e.department_id = d.department_id
    JOIN locations l ON d.location_id = l.location_id
    JOIN countries c ON l.country_id = c.country_id
    WHERE c.country_name = p_country_name;

    RETURN 'Pracownicy: ' || v_emp_count || ', Departamenty: ' || v_dept_count;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20003, 'Kraj nie istnieje.');
END;
/
```

Stworzyć następujące wyzwalacze:

1. Stworzyć tabelę archiwum_departamentów (id, nazwa, data_zamknięcia, ostatni_manager jako imię i nazwisko). Po usunięciu departamentu dodać odpowiedni rekord do tej tabeli

```
CREATE OR REPLACE TRIGGER trg_archive_departments
```

```
AFTER DELETE ON departments

FOR EACH ROW

BEGIN

   INSERT INTO archiwum_departamentów (id, nazwa, data_zamknięcia, ostatni_manager)

   VALUES (:OLD.department_id, :OLD.department_name, SYSDATE,

         (SELECT first_name || ' ' || last_name

          FROM employees

          WHERE employee_id = :OLD.manager_id));

END;

/
```

2. W razie UPDATE i INSERT na tabeli employees, sprawdzić czy zarobki łapią się w widełkach 2000 - 26000. Jeśli nie łapią się - zabronić dodania. Dodać tabelę złodziej(id, USER, czas_zmiany), której będą wrzucane logi, jeśli będzie próba dodania, bądź zmiany wynagrodzenia poza widełki.

```
CREATE OR REPLACE TRIGGER trg_check_salary
BEFORE INSERT OR UPDATE ON employees
FOR EACH ROW
BEGIN
   IF :NEW.salary NOT BETWEEN 2000 AND 26000 THEN
      INSERT INTO zlodziej (id, USER, czas_zmiany)
      VALUES (zlodziej_seq.NEXTVAL, USER, SYSDATE);
      RAISE_APPLICATION_ERROR(-20004, 'Zarobki poza dozwolonym zakresem!');
   END IF;
END;
/
```

3. Stworzyć sekwencję i wyzwalacz, który będzie odpowiadał za auto_increment w tabeli employees.

```
CREATE OR REPLACE TRIGGER trg_auto_increment
BEFORE INSERT ON employees
FOR EACH ROW
BEGIN
```

```
   :NEW.employee_id := employees_seq.NEXTVAL;
END;
/
```

4. Stworzyć wyzwalacz, który zabroni dowolnej operacji na tabeli JOD_GRADES (INSERT, UPDATE, DELETE)

```
CREATE OR REPLACE TRIGGER trg_no_operations
BEFORE INSERT OR UPDATE OR DELETE ON job_grades
BEGIN
   RAISE_APPLICATION_ERROR(-20005, 'Operacje na tabeli JOB_GRADES są zabronione.');
END;
/
```

5. Stworzyć wyzwalacz, który przy próbie zmiany max i min salary w tabeli jobs zostawia stare wartości.

```
CREATE OR REPLACE TRIGGER trg_lock_salary

BEFORE UPDATE OF min_salary, max_salary ON jobs

FOR EACH ROW

BEGIN

   :NEW.min_salary := :OLD.min_salary;

   :NEW.max_salary := :OLD.max_salary;

END;

/
```

Stworzyć paczki:

1. Składającą się ze stworzonych procedur i funkcji

```
CREATE OR REPLACE PACKAGE pkg_hr_management AS
   FUNCTION get_job_name(p_job_id VARCHAR2) RETURN VARCHAR2;
   FUNCTION get_annual_salary(p_employee_id NUMBER) RETURN NUMBER;
   FUNCTION extract_area_code(p_phone VARCHAR2) RETURN VARCHAR2;
   FUNCTION capitalize_first_last(p_string VARCHAR2) RETURN VARCHAR2;
```

```
    FUNCTION pesel_to_birthdate(p_pesel VARCHAR2) RETURN DATE;
    FUNCTION count_employees_departments(p_country_name VARCHAR2) RETURN
VARCHAR2;

    PROCEDURE add_job(p_job_id VARCHAR2, p_job_title VARCHAR2);
    PROCEDURE update_job(p_job_id VARCHAR2, p_new_title VARCHAR2);
    PROCEDURE delete_job(p_job_id VARCHAR2);
    PROCEDURE get_salary_and_name(p_employee_id NUMBER, p_salary OUT NUMBER,
p_last_name OUT VARCHAR2);
    PROCEDURE add_employee(
        p_first_name VARCHAR2,
        p_last_name VARCHAR2,
        p_salary NUMBER,
        p_job_id VARCHAR2
    );
END pkg_hr_management;
/
```

2. Stworzyć paczkę z procedurami i funkcjami do obsługi tabeli REGIONS (CRUD), gdzie odczyt z różnymi parametrami