1. Stworzyć blok anonimowy wypisujący zmienną numer_max równą maksymalnemu numerowi Departamentu i dodaj do tabeli departamenty – departament z numerem o 10 wiekszym, typ pola dla zmiennej z nazwą nowego departamentu (zainicjować na EDUCATION) ustawić taki jak dla pola department_name w tabeli (%TYPE)

```
DECLARE
   numer_max departments.department_id%TYPE;
   new_department departments.department_name%TYPE := 'EDUCATION';
BEGIN
   SELECT MAX(department_id) INTO numer_max FROM departments;
   INSERT INTO departments (department_id, department_name)
   VALUES (numer_max + 10, new_department);
   DBMS_OUTPUT.PUT_LINE('Dodano nowy departament: ' || new_department);
END;
```

2. Do poprzedniego skryptu dodaj instrukcje zmieniającą location_id (3000) dla dodanego departamentu

```
UPDATE departments
SET location_id = 3000
WHERE department_id = (SELECT MAX(department_id) FROM departments);
```

3. Stwórz tabelę nowa z jednym polem typu varchar a następnie wpisz do niej za pomocą pętli liczby od 1 do 10 bez liczb 4 i 6

```
CREATE TABLE LICZBY(wartosc VARCHAR(50));
INSERT INTO LICZBY (wartosc)
SELECT TO_CHAR(i)
FROM (SELECT LEVEL AS i FROM dual CONNECT BY LEVEL <= 10)
```

```
WHERE i NOT IN (4, 6);
```

4. Wyciągnąć informacje z tabeli countries do jednej zmiennej (%ROWTYPE) dla kraju o identyfikatorze ‘CA’. Wypisać nazwę i region_id na ekran

```
CREATE TABLE countries AS SELECT * FROM HR.countries;

SELECT 'Country Name: ' || country_name || ', Region ID: ' || region_id AS result

FROM countries

WHERE country_id = 'CA'
```

5. Zadeklaruj kursor jako wynagrodzenie, nazwisko dla departamentu o numerze 50. Dla elementów kursora wypisać na ekran, jeśli wynagrodzenie jest wyższe niż 3100: nazwisko osoby i tekst ‘nie dawać podwyżki’ w przeciwnym przypadku: nazwisko + ‘dać podwyżkę’

```
SELECT

   last_name,

   CASE

      WHEN salary > 3100 THEN last_name || ' nie dawać podwyżki'

      ELSE last_name || ' dać podwyżkę'

   END AS wynagrodzenie

FROM employees

WHERE department_id = 50;
```

6. Zadeklarować kursor zwracający zarobki imię i nazwisko pracownika z parametrami, gdzie pierwsze dwa parametry określają widełki zarobków a trzeci część imienia pracownika. Wypisać na ekran pracowników:
    a. z widełkami 1000- 5000 z częścią imienia a (może być również A)

```
SELECT first_name, last_name, salary

FROM employees
```

```
WHERE salary BETWEEN 1000 AND 5000
AND LOWER(first_name) LIKE '%a%';
```

b. z widełkami 5000-20000 z częścią imienia u (może być również U)

```
SELECT first_name, last_name, salary
FROM employees
WHERE salary BETWEEN 5000 AND 20000
AND LOWER(first_name) LIKE '%u%';
```

9. Stwórz procedury:

   a. dodającą wiersz do tabeli Jobs – z dwoma parametrami wejściowymi określającymi Job_id, Job_title, przetestuj działanie wrzuć wyjątki – co najmniej when others

```
CREATE OR REPLACE PROCEDURE add_job(
  p_job_id IN VARCHAR2,
  p_job_title IN VARCHAR2
) AS
BEGIN
  INSERT INTO jobs (job_id, job_title, min_salary, max_salary)
  VALUES (p_job_id, p_job_title, NULL, NULL);

  COMMIT;
EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Wystąpił błąd: ' || SQLERRM);
```

```
        ROLLBACK;
END;
/
```

b. modyfikującą title w  tabeli Jobs – z dwoma parametrami id dla którego ma być modyfikacja oraz nową wartość dla Job_title – przetestować działanie, dodać swój wyjątek dla no Jobs updated – najpierw sprawdzić numer błędu

```
CREATE OR REPLACE PROCEDURE update_job_title(
    p_job_id IN VARCHAR2,
    p_new_job_title IN VARCHAR2
) AS
BEGIN
    UPDATE jobs
    SET job_title = p_new_job_title
    WHERE job_id = p_job_id;

    IF SQL%ROWCOUNT = 0 THEN
        RAISE_APPLICATION_ERROR(-20001, 'No Jobs updated');
    END IF;

    COMMIT;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Brak rekordu do aktualizacji');
    WHEN OTHERS THEN
```

```
        DBMS_OUTPUT.PUT_LINE('Wystąpił błąd: ' || SQLERRM);
        ROLLBACK;
END;
/
```

c. usuwającą wiersz z tabeli Jobs  o podanym Job_id– przetestować działanie, dodaj wyjątek dla no Jobs deleted

```
CREATE OR REPLACE PROCEDURE delete_job(
    p_job_id IN VARCHAR2
) AS
BEGIN
    DELETE FROM jobs
    WHERE job_id = p_job_id;

    IF SQL%ROWCOUNT = 0 THEN
        RAISE_APPLICATION_ERROR(-20002, 'No Jobs deleted');
    END IF;

    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Wystąpił błąd: ' || SQLERRM);
        ROLLBACK;
END;
/
```

d. Wyciągającą zarobki i nazwisko (parametry zwracane przez procedurę) z tabeli employees dla pracownika o przekazanym jako parametr id

```
CREATE OR REPLACE PROCEDURE get_salary_and_name(
    p_employee_id IN NUMBER,
    p_salary OUT NUMBER,
    p_last_name OUT VARCHAR2
) AS
BEGIN
    SELECT salary, last_name
    INTO p_salary, p_last_name
    FROM employees
    WHERE employee_id = p_employee_id;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Brak pracownika o ID ' || p_employee_id);
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Wystąpił błąd: ' || SQLERRM);
END;
/
```

e. dodającą do tabeli employees wiersz – większość parametrów ustawić na domyślne (id poprzez sekwencję), stworzyć wyjątek jeśli wynagrodzenie dodawanego pracownika jest wyższe niż 20000

```
CREATE OR REPLACE PROCEDURE add_employee(
```

```
    p_first_name IN VARCHAR2,
    p_last_name IN VARCHAR2,
    p_salary IN NUMBER
) AS
BEGIN
    IF p_salary > 20000 THEN
        RAISE_APPLICATION_ERROR(-20003, 'Wynagrodzenie nie może być wyższe niż
20000');
    END IF;

    INSERT INTO employees (first_name, last_name, salary)
    VALUES (p_first_name, p_last_name, p_salary);

    COMMIT;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Wystąpił błąd: ' || SQLERRM);
        ROLLBACK;
END;
/
```