

Julia Ruszer 247775

Dominik Gałkowski 247659

Jan Śladowski 247806

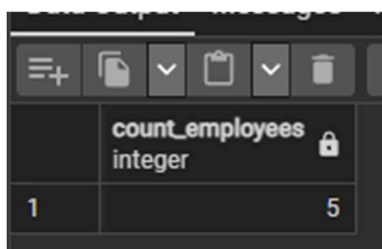
Wiktor Żelechowski 247833

Zadanie 1

Utwórz funkcję, która zwróci liczbę pracowników zatrudnionych na określonym stanowisku w określonym dziale w określonym kraju. Nazwa stanowiska, nazwa działu i nazwa kraju są przekazywane do funkcji jako parametry. Wywołaj funkcję z parametrami Sales Manager, Sales i United Kingdom.

PostgreSQL

```
CREATE OR REPLACE FUNCTION count_employees(  
    job_title_fun VARCHAR(35),  
    department_name_fun VARCHAR(30),  
    country_name_fun VARCHAR(40)  
) RETURNS INTEGER AS $$  
BEGIN  
    RETURN (  
        SELECT COUNT(e.employee_id)  
        FROM employees e  
        JOIN jobs j ON e.job_id = j.job_id  
        JOIN departments d ON e.department_id = d.department_id  
        JOIN locations l ON d.location_id = l.location_id  
        JOIN countries c ON l.country_id = c.country_id  
        WHERE j.job_title = job_title_fun  
        AND d.department_name = department_name_fun  
        AND c.country_name = country_name_fun  
    );  
END;  
$$ LANGUAGE plpgsql;  
  
SELECT count_employees('Sales Manager', 'Sales', 'United Kingdom');
```



| count_employees integer | |
|----------------------------|---|
| 1 | 5 |

MS SQL Server

```
CREATE OR ALTER FUNCTION func (@input1 NVARCHAR(100), @input2  
NVARCHAR(100), @input3 NVARCHAR(100))
```

```

RETURNS INT
AS BEGIN
    DECLARE @employeeCount INT

    SELECT @employeeCount = COUNT(*) from employees e
    JOIN departments d ON d.department_id = e.department_id
    JOIN locations l ON l.location_id = d.location_id
    JOIN countries c ON c.country_id = l.country_id
    JOIN jobs j ON j.job_id = e.job_id
    WHERE j.job_title = @input1
    AND d.department_name = @input2
    AND c.country_name = @input3;
    RETURN @employeeCount;
END

SELECT dbo.func('Sales Manager', 'Sales', 'United Kingdom') AS Employees

```

| Results | | Messages | |
|---------|-----------|----------|--|
| | Employees | | |
| 1 | 5 | | |

Zadanie 2

Utwórz wyzwalacz, który przy zmianie stanowiska danego pracownika:

- zaktualizuje jego datę zatrudnienia w tabeli employees na dzień jutrzejszy,
- zarchiwizuje dane o jego poprzednim stanowisku, tzn. doda odpowiednie informacje do tabeli job_history i ustawi datę końcową na dzień dzisiejszy,
- sprawdzi, czy jego aktualne wynagrodzenie należy do zdefiniowanego przedziału wartości wynagrodzeń dla jego nowego stanowiska. Jeżeli jego aktualne wynagrodzenie jest niższe niż minimalna wartość przedziału, to zostanie zaktualizowane do tejże wartości. Dodatkowo zostanie wypisana informacja o imieniu i nazwisku pracownika oraz kwocie jego podwyżki (różnicy pomiędzy nową i starą kwotą wynagrodzenia). Jeżeli jego aktualne wynagrodzenie jest wyższe niż maksymalna wartość przedziału, to maksymalne wynagrodzenie dla jego nowego stanowiska zostanie zaktualizowane do wartości jego aktualnego wynagrodzenia. Potwierdź działanie dla wszystkich przypadków testowych. *Uwaga!* Wyzwalacz powinien pracować także przy zmianie stanowiska dla wielu pracowników jednocześnie.

PostgreSQL

```

CREATE OR REPLACE FUNCTION trigger_employee_job_change()
RETURNS TRIGGER AS $$
DECLARE
    new_min_salary NUMERIC(6);
    new_max_salary NUMERIC(6);
    current_salary NUMERIC(6);
    salary_increase NUMERIC;
BEGIN
    SELECT min_salary, max_salary INTO new_min_salary, new_max_salary
    FROM jobs

```

```

WHERE job_id = NEW.job_id;

NEW.hire_date := CURRENT_DATE + INTERVAL '1 day';

INSERT INTO job_history (employee_id, start_date, end_date, job_id,
department_id)
VALUES (OLD.employee_id, OLD.hire_date, CURRENT_DATE, OLD.job_id,
OLD.department_id);

current_salary := NEW.salary;

--jak mniejsze to podnosi pracowników
IF current_salary < new_min_salary THEN
    salary_increase := new_min_salary - current_salary;
    NEW.salary := new_min_salary;
    RAISE NOTICE 'Pracownik % % otrzymał podwyżkę o kwotę %',
NEW.first_name, NEW.last_name, salary_increase;

--jak większe to zmienia dla stanowiska
ELSIF current_salary > new_max_salary THEN
    UPDATE jobs
    SET max_salary = current_salary
    WHERE job_id = NEW.job_id;
ENDIF;

RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE TRIGGER job_change_trigger
BEFORE UPDATE OF job_id ON employees
FOR EACH ROW
WHEN (OLD.job_id IS DISTINCT FROM NEW.job_id)
EXECUTE FUNCTION trigger_employee_job_change();

--dobrze wynagrodzenie
INSERT INTO employees (employee_id, first_name, last_name, email,
phone_number, hire_date, job_id, salary, commission_pct, manager_id,
department_id)
VALUES (1000, 'imie1', 'nazwisko1', 'test1', '515.123.1234', '2000-01-01',
'IT_PROG', 11000, NULL, NULL, 60);

--wyzsze wynagrodzenie
INSERT INTO employees (employee_id, first_name, last_name, email,
phone_number, hire_date, job_id, salary, commission_pct, manager_id,
department_id)
VALUES (1001, 'imie2', 'nazwisko2', 'test2', '515.123.1234', '2000-01-01',
'IT_PROG', 25000, NULL, NULL, 60);

--za niskie wynagrodzenie
INSERT INTO employees (employee_id, first_name, last_name, email,
phone_number, hire_date, job_id, salary, commission_pct, manager_id,
department_id)
VALUES (1002, 'imie3', 'nazwisko3', 'test3', '515.123.1234', '2000-01-01',
'IT_PROG', 5000, NULL, NULL, 60);

```

```
UPDATE employees SET job_id = 'SA_MAN' WHERE employee_id = 1000;
UPDATE employees SET job_id = 'SA_MAN' WHERE employee_id = 1001;
UPDATE employees SET job_id = 'SA_MAN' WHERE employee_id = 1002;
```

| | | | | | | | |
|------|-------|-----------|-------|--------------|------------|--------|----------|
| 1000 | imie1 | nazwisko1 | test1 | 515.123.1234 | 2024-11-21 | SA_MAN | 11000.00 |
| 1001 | imie2 | nazwisko2 | test2 | 515.123.1234 | 2024-11-21 | SA_MAN | 25000.00 |
| 1002 | imie3 | nazwisko3 | test3 | 515.123.1234 | 2024-11-21 | SA_MAN | 10000.00 |

UWAGA: Pracownik imie3 nazwisko3 otrzymał podwyżkę o kwotę 5000.00

| | | | | | | | |
|------|-------|-----------|-------|--------------|------------|--------|----------|
| 1000 | imie1 | nazwisko1 | test1 | 515.123.1234 | 2024-11-21 | SA_MAN | 11000.00 |
| 1001 | imie2 | nazwisko2 | test2 | 515.123.1234 | 2024-11-21 | SA_MAN | 25000.00 |
| 1002 | imie3 | nazwisko3 | test3 | 515.123.1234 | 2024-11-21 | SA_MAN | 10000.00 |

| | | | |
|--------|---------------|-------|-------|
| SA_MAN | Sales Manager | 10000 | 25000 |
|--------|---------------|-------|-------|

| | | | | |
|------|------------|------------|---------|----|
| 1000 | 2000-01-01 | 2024-11-20 | IT_PROG | 60 |
| 1001 | 2000-01-01 | 2024-11-20 | IT_PROG | 60 |
| 1002 | 2000-01-01 | 2024-11-20 | IT_PROG | 60 |

MS SQL Server

```
CREATE OR ALTER TRIGGER trig
ON employees
AFTER UPDATE
AS
BEGIN
    INSERT INTO job_history(employee_id, start_date, end_date, job_id,
department_id)
    SELECT
        d.employee_id,
        d.hire_date,
        CAST(GETDATE() AS DATE),
        d.job_id,
        d.department_id
    FROM deleted d
    JOIN inserted i on d.employee_id = i.employee_id
    WHERE
        i.job_id <> d.job_id;

    UPDATE employees
    SET hire_date = CAST DATEADD(DAY, 1, GETDATE()) AS DATE
    FROM inserted i
    WHERE employees.employee_id = i.employee_id
    AND i.job_id <> (SELECT job_id FROM deleted WHERE
deleted.employee_id = i.employee_id);

    DECLARE @new_min_salary NUMERIC(8,2), @new_max_salary NUMERIC(8,2),
@current_salary NUMERIC(8,2), @salary_increase NUMERIC(8,2);
    DECLARE @employee_id NUMERIC(6);
    DECLARE @first_name VARCHAR(20), @last_name VARCHAR(20), @job_id
VARCHAR(20);
    DECLARE salary_cursor CURSOR FOR
    SELECT i.salary, j.min_salary, j.max_salary, i.employee_id,
i.first_name, i.last_name, i.job_id
    FROM inserted i
    JOIN jobs j ON i.job_id = j.job_id
```

```

        WHERE i.job_id <> (SELECT job_id FROM deleted WHERE
deleted.employee_id = i.employee_id);

        OPEN salary_cursor
        FETCH NEXT FROM salary_cursor INTO @current_salary,
@new_min_salary, @new_max_salary, @employee_id, @first_name, @last_name,
@job_id;
        WHILE @@FETCH_STATUS = 0
        BEGIN
            IF @current_salary < @new_min_salary
            BEGIN
                SET @salary_increase = @new_min_salary -
@current_salary

                UPDATE employees
                SET salary = @new_min_salary
                WHERE employee_id = @employee_id

                PRINT 'Pracownik ' + @first_name + ' ' + @last_name
+ ' otrzymał odwyżkę o ' + CAST(@current_salary AS VARCHAR(10));
            END
            ELSE IF @current_salary > @new_max_salary
            BEGIN
                UPDATE jobs
                SET max_salary = @current_salary
                WHERE job_id = @job_id
            END
            FETCH NEXT FROM salary_cursor INTO @current_salary,
@new_min_salary, @new_max_salary, @employee_id, @first_name, @last_name,
@job_id;
        END;
        CLOSE salary_cursor
        DEALLOCATE salary_cursor
    END

--dobre wynagrodzenie
INSERT INTO employees (employee_id, first_name, last_name, email,
phone_number, hire_date, job_id, salary, commission_pct, manager_id,
department_id)
VALUES (1000, 'imie1', 'nazwisko1', 'test1', '515.123.1234', '2000-01-01',
'IT_PROG', 11000, NULL, NULL, 60);

--wyzsze wynagrodzenie
INSERT INTO employees (employee_id, first_name, last_name, email,
phone_number, hire_date, job_id, salary, commission_pct, manager_id,
department_id)
VALUES (1001, 'imie2', 'nazwisko2', 'test2', '515.123.1234', '2000-01-01',
'IT_PROG', 28000, NULL, NULL, 60);

--za niskie wynagrodzenie
INSERT INTO employees (employee_id, first_name, last_name, email,
phone_number, hire_date, job_id, salary, commission_pct, manager_id,
department_id)
VALUES (1002, 'imie3', 'nazwisko3', 'test3', '515.123.1234', '2000-01-01',
'IT_PROG', 5000, NULL, NULL, 60);

UPDATE employees SET job_id = 'SA_MAN' WHERE employee_id = 1000;
UPDATE employees SET job_id = 'SA_MAN' WHERE employee_id = 1001;
UPDATE employees SET job_id = 'SA_MAN' WHERE employee_id = 1002;

```

| | | | | | | | | | | | |
|-----|------|-------|-----------|-------|--------------|------------|---------|----------|------|------|----|
| 108 | 1000 | imie1 | nazwisko1 | test1 | 515.123.1234 | 2000-01-01 | IT_PROG | 11000.00 | NULL | NULL | 60 |
| 109 | 1001 | imie2 | nazwisko2 | test2 | 515.123.1234 | 2000-01-01 | IT_PROG | 28000.00 | NULL | NULL | 60 |
| 110 | 1002 | imie3 | nazwisko3 | test3 | 515.123.1234 | 2000-01-01 | IT_PROG | 5000.00 | NULL | NULL | 60 |



Messages

(1 row affected)

Pracownik imie3 nazwisko3 otrzymał odwyżkę o 5000.00

| | | | | | | | | | | | |
|-----|------|-------|-----------|-------|--------------|------------|--------|----------|------|------|----|
| 108 | 1000 | imie1 | nazwisko1 | test1 | 515.123.1234 | 2024-11-21 | SA_MAN | 11000.00 | NULL | NULL | 60 |
| 109 | 1001 | imie2 | nazwisko2 | test2 | 515.123.1234 | 2024-11-21 | SA_MAN | 28000.00 | NULL | NULL | 60 |
| 110 | 1002 | imie3 | nazwisko3 | test3 | 515.123.1234 | 2024-11-21 | SA_MAN | 10000.00 | NULL | NULL | 60 |

| | | | | | |
|----|------|------------|------------|---------|----|
| 11 | 1000 | 2000-01-01 | 2024-11-20 | IT_PROG | 60 |
| 12 | 1001 | 2000-01-01 | 2024-11-20 | IT_PROG | 60 |
| 13 | 1002 | 2000-01-01 | 2024-11-20 | IT_PROG | 60 |

| | | | | |
|----|--------|---------------|-------|-------|
| 15 | SA_MAN | Sales Manager | 10000 | 28000 |
|----|--------|---------------|-------|-------|

Zadanie 3

Utwórz procedurę, która zmieni stanowisko na podane u pracowników zatrudnionych na określonym stanowisku w określonym kraju i poprzez parametr wyjściowy zwróci liczbę zmodyfikowanych rekordów oraz wyświetli id, imię, nazwisko i nazwę departamentu pracowników, których stanowisko zostało zmienione. Dodatkowo wypisze informacje o wszystkich departamentach z danego kraju razem z listą ich pracowników (id, imię i nazwisko), których stanowisko zostało zmienione. Jeżeli w jakimś departamencie w danym kraju nie pracuje żaden pracownik na danym stanowisku, wywoła jak najwyższy priorytetowo komunikat, który nie przerwie wykonywania kodu i wypisze: "Brak pracowników na stanowisku X w departamencie Y w kraju Z!", gdzie X jest nazwą poprzedniego stanowiska, Y jest nazwą aktualnie sprawdzanego departamentu, Z jest nazwą podanego kraju. Procedura ma także weryfikować podane dane. Jeżeli podany kraj nie istnieje, wywoła wyjątek, który przerwie wykonywanie kodu i wypisze: "Brak kraju X!", gdzie X to nazwa podanego kraju. Jeżeli w podanym kraju nie ma żadnego departamentu, wywoła wyjątek, który przerwie wykonywanie kodu i wypisze: "Brak departamentów w kraju X!", gdzie X to nazwa podanego kraju. Jeżeli w podanym kraju nie pracuje żaden pracownik, wywoła wyjątek, który przerwie wykonywanie kodu i wypisze: "Brak pracowników zatrudnionych w kraju X!", gdzie X to nazwa podanego kraju. W swoim rozwiązaniu wykorzystaj funkcję z zadania 1 oraz wyzwalacz z zadania 2. Wywołaj procedurę z nazwami stanowisk Sales Manager i Sales Representative oraz odpowiednią nazwą kraju, żeby przetestować wszystkie możliwe przypadki.

PostgreSQL

```
CREATE OR REPLACE PROCEDURE change_job_title_procedure(
    IN current_job_title VARCHAR(100),
    IN new_job_title VARCHAR(100),
    IN country_name_param VARCHAR(100),
    OUT employee_count INTEGER
)
LANGUAGE plpgsql
AS $$
DECLARE
```

```

department_name_var VARCHAR(100);
employee_id NUMERIC(6, 0);
first_name VARCHAR(100);
last_name VARCHAR(100);
department_id var VARCHAR(100);
dupa NUMERIC(6, 0);
BEGIN
    IF NOT EXISTS (
        SELECT 1 FROM countries c WHERE c.country_name = country_name_param
    ) THEN
        RAISE EXCEPTION 'Brak kraju %!', country_name_param;
    END IF;

    IF NOT EXISTS (
        SELECT 1
        FROM departments d
        JOIN locations l ON d.location_id = l.location_id
        JOIN countries c ON l.country_id = c.country_id
        WHERE c.country_name = country_name_param
    ) THEN
        RAISE EXCEPTION 'Brak departamentów w kraju %!',
country_name_param;
    END IF;

    IF NOT EXISTS (
        SELECT 1
        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 = country_name_param
    ) THEN
        RAISE EXCEPTION 'Brak pracowników zatrudnionych w kraju %!',
country_name_param;
    END IF;

    IF EXISTS (SELECT FROM pg_tables WHERE tablename =
'temp_updated_employees') THEN
        DROP TABLE temp_updated_employees;
    END IF;

    CREATE TEMP TABLE temp_updated_employees AS
    SELECT
        e.employee_id,
        e.first_name,
        e.last_name,
        d.department_name,
        new_job_title AS simulated_job_title
    FROM employees e
    JOIN jobs j ON e.job_id = j.job_id
    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 j.job_title = current_job_title AND c.country_name =
country_name_param;

    SELECT SUM(count_employees(current_job_title, d.department_name,
country_name_param))
    INTO employee_count
    FROM departments d
    JOIN locations l ON d.location_id = l.location_id

```

```

JOIN countries c ON l.country_id = c.country_id
WHERE c.country_name = country_name_param;
EXECUTE 'SELECT e.first_name FROM employees e';

UPDATE employees e
SET job_id = (
    SELECT j_new.job_id
    FROM jobs j_new
    WHERE j_new.job_title = new_job_title
)
WHERE e.employee_id IN (
    SELECT te.employee_id
    FROM temp_updated_employees te
);

FOR department_name_var IN
    SELECT d.department_name
    FROM departments d
    JOIN locations l ON d.location_id = l.location_id
    JOIN countries c ON l.country_id = c.country_id
    WHERE c.country_name = country_name_param
LOOP
    IF EXISTS (
        SELECT 1
        FROM temp_updated_employees te
        WHERE te.department_name = department_name_var
    ) THEN
        RAISE NOTICE 'Departament: %', department_name_var;

        FOR employee_id, first_name, last_name IN
            SELECT te.employee_id, te.first_name, te.last_name
            FROM temp_updated_employees te
            WHERE te.department_name = department_name_var
        LOOP
            RAISE NOTICE ' Employee ID: %, Name: % %', employee_id,
first_name, last_name;
        END LOOP;
    ELSE
        RAISE NOTICE 'Brak pracowników na stanowisku % w departamencie
% w kraju %!',
current_job_title, department_name_var, country_name_param;
    END IF;
END LOOP;

END;
$$;

```

Data Output [Messages](#) [Notifications](#)

```

UWAGA: Brak pracowników na stanowisku Sales Manager w departamencie Human Resources w kraju United Kingdom!
UWAGA: Departament: Sales
UWAGA: Employee ID: 145, Name: John Russell
UWAGA: Employee ID: 146, Name: Karen Partners
UWAGA: Employee ID: 147, Name: Alberto Errazuriz
UWAGA: Employee ID: 148, Name: Gerald Cambrault
UWAGA: Employee ID: 149, Name: Eleni Zlotkey
UWAGA: Liczba pracowników: 5
DO

```


Data Output Messages Notifications

ERROR: Brak departamentów w kraju Mexico!

Data Output Messages Notifications

ERROR: Brak kraju Poland!

Data Output Messages Notifications

ERROR: Brak pracowników zatrudnionych w kraju Australia!

MS SQL Server

```
CREATE OR ALTER PROCEDURE changeJobTitle
    @CurrentJobTitle NVARCHAR(100),
    @NewJobTitle NVARCHAR(100),
    @Country NVARCHAR(100),
    @UpdatedCount INT OUTPUT
AS
BEGIN
    SET NOCOUNT ON;
    DECLARE @message NVARCHAR(max);

    IF NOT EXISTS (
        SELECT 1 FROM countries WHERE country_name = @Country
    )
    BEGIN
        SET @message = CONCAT('Brak kraju ', @Country, '!');
        THROW 50001, @message, 1;
    END;

    IF NOT EXISTS (
        SELECT 1
        FROM departments d
        JOIN locations l ON d.location_id = l.location_id
        JOIN countries c ON l.country_id = c.country_id
        WHERE c.country_name = @Country
    )
    BEGIN
        SET @message = CONCAT('Brak departamentów w kraju ', @Country,
        '!');
        THROW 50002, @message, 1;
    END;

    IF NOT EXISTS (
        SELECT 1
        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 = @Country
    )
```

```

BEGIN
    SET @message = CONCAT('Brak pracowników zatrudnionych w kraju ',
@Country, '!');
    THROW 50003, @message, 1;
END;

SELECT @UpdatedCount = dbo.func(@CurrentJobTitle, d.department_name,
@Country)
FROM departments d
JOIN locations l ON d.location_id = l.location_id
JOIN countries c ON l.country_id = c.country_id
WHERE c.country_name = @Country

CREATE TABLE #UpdatedEmployees (
    employee_id NUMERIC(6, 0),
    first_name NVARCHAR(100),
    last_name NVARCHAR(100),
    department_name NVARCHAR(100)
);

INSERT INTO #UpdatedEmployees (employee_id, first_name, last_name,
department_name)
SELECT e.employee_id, e.first_name, e.last_name, d.department_name
FROM employees e
JOIN jobs j ON e.job_id = j.job_id
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 j.job_title = @CurrentJobTitle AND c.country_name = @Country;

UPDATE employees
SET job_id = (SELECT job_id FROM jobs WHERE job_title = @NewJobTitle)
WHERE employee_id IN (SELECT employee_id FROM #UpdatedEmployees);

SELECT employee_id, first_name, last_name, department_name
FROM #UpdatedEmployees;

DECLARE @msg NVARCHAR(MAX) = '';

PRINT 'Informacje o departamentach i pracownikach:';

DECLARE @DepartmentName NVARCHAR(100);
DECLARE departmentCursor CURSOR FOR

SELECT d.department_name
FROM departments d
JOIN locations l ON d.location_id = l.location_id
JOIN countries c ON l.country_id = c.country_id
WHERE c.country_name = @Country;

OPEN departmentCursor;
FETCH NEXT FROM departmentCursor INTO @DepartmentName;

WHILE @@FETCH_STATUS = 0
BEGIN
    IF EXISTS (
        SELECT 1
        FROM #UpdatedEmployees ue
        JOIN employees e ON ue.employee_id = e.employee_id

```

```

        JOIN departments d ON e.department_id = d.department_id
        WHERE d.department_name = @DepartmentName
    )
    BEGIN
        PRINT 'Departament: ' + @DepartmentName;
        SELECT @msg = STRING_AGG('Employee ID: ' + CAST(ue.employee_id
AS NVARCHAR(10)) + ', Name: ' + ue.first_name + ' ' + ue.last_name,
CHAR(13) + CHAR(10))
        FROM #UpdatedEmployees ue
        JOIN employees e ON ue.employee_id = e.employee_id
        JOIN departments d ON e.department_id = d.department_id
        WHERE d.department_name = @DepartmentName;
        PRINT @msg;
    END
    ELSE
    BEGIN
        PRINT 'Brak pracowników na stanowisku ' + @CurrentJobTitle + '
w departamencie ' + @DepartmentName + ' w kraju ' + @Country + '!';
    END;

    FETCH NEXT FROM departmentCursor INTO @DepartmentName;
END;

CLOSE departmentCursor;
DEALLOCATE departmentCursor;

DROP TABLE #UpdatedEmployees;
END
GO

DECLARE @UpdatedCount INT;

EXEC changeJobTitle
    @CurrentJobTitle = 'Sales Manager',
    @NewJobTitle = 'Sales Representative',
    @Country = 'United Kingdom',
    @UpdatedCount = @UpdatedCount OUTPUT;

PRINT 'Liczba zmodyfikowanych rekordów: ' + CAST(@UpdatedCount AS
NVARCHAR(10));

```

| | employee_id | first_name | last_name | department_name |
|---|-------------|------------|-----------|-----------------|
| 1 | 145 | John | Russell | Sales |
| 2 | 146 | Karen | Partners | Sales |
| 3 | 147 | Alberto | Errazuriz | Sales |
| 4 | 148 | Gerald | Cambrault | Sales |
| 5 | 149 | Eleni | Zlotkey | Sales |

Results Messages

Informacje o departamentach i pracownikach:
 Brak pracowników na stanowisku Sales Manager w departamencie Human Resources w kraju United Kingdom!
 Departament: Sales
 Employee ID: 145, Name: John Russell
 Employee ID: 146, Name: Karen Partners
 Employee ID: 147, Name: Alberto Errazuriz
 Employee ID: 148, Name: Gerald Cambrault
 Employee ID: 149, Name: Eleni Zlotkey
 Liczba zmodyfikowanych rekordów: 5

110 %

Messages

Msg 50001, Level 16, State 1, Procedure changeJobTitle, Line 18 [Batch Start Line 201]
Brak kraju Polska!

Completion time: 2024-11-20T23:33:13.5928328+01:00

Messages

Msg 50002, Level 16, State 1, Procedure changeJobTitle, Line 31 [Batch Start Line 201]
Brak departamentów w kraju Mexico!

Completion time: 2024-11-20T23:31:54.6814548+01:00

110 %

Messages

Msg 50003, Level 16, State 1, Procedure changeJobTitle, Line 44 [Batch Start Line 201]
Brak pracowników zatrudnionych w kraju Australia!

Completion time: 2024-11-20T23:32:52.2751860+01:00