# Bazy Danych - Laboratorium 6

Student:

```
Bartłomiej Krawczyk
```

Numer Albumu:

```
310774
```

```
-- Napisz zapytanie, które wyświetli imię, nazwisko oraz nazwy zakładów, w których
pracownicy mają większe zarobki niż minimalne zarobki na stanowisku o nazwie
‘Konsultant’.

SELECT e.name, e.surname, d.name
FROM employees e
    JOIN departments d USING(department_id)
WHERE e.salary > (
    SELECT MIN(e.salary)
    FROM employees e
        JOIN positions p USING(position_id)
    WHERE p.name = 'Konsultant'
);

-- Napisz zapytanie, które zwróci dane najmłodszego wśród dzieci pracowników.
(Skorzystaj z podzapytań. Jaki jest inny sposób na osiągnięcie tego wyniku?)

SELECT *
FROM dependents d
WHERE d.birth_date = (SELECT MIN(birth_date) FROM dependents);

-- Inny sposób:

SELECT *
FROM dependents d
ORDER BY d.birth_date
FETCH FIRST 1 ROWS ONLY;


-- Napisz zapytanie, które zwróci dane dzieci najstarszego pracownika z zakładu
102.

SELECT *
FROM dependents
WHERE employee_id = (
```

```
    SELECT employee_id
    FROM employees
    WHERE department_id = 102
    ORDER BY birth_date
    FETCH FIRST 1 ROW ONLY
);

-- Napisz zapytanie, które wyświetli wszystkich pracowników, którzy zostali
zatrudnieni nie wcześniej niż najwcześniej zatrudniony pracownik w zakładzie o id
101
-- i nie później niż najpóźniej zatrudniony pracownik w zakładzie o id 107.

SELECT *
FROM employees
WHERE date_employed >= (
    SELECT MIN(date_employed)
    FROM employees
    WHERE department_id = 101
)
AND date_employed <= (
    SELECT MAX(date_employed)
    FROM employees
    WHERE department_id = 107
);

-- Wyświetl średnie zarobki dla każdego ze stanowisk, o ile średnie te są większe
od średnich zarobków w departamencie “Administracja”.

SELECT position_id, p.name, ROUND(AVG(e.salary))
FROM employees e
    JOIN positions p USING(position_id)
GROUP BY position_id, p.name
HAVING AVG(e.salary) > (
    SELECT AVG(e.salary)
    FROM employees e
        JOIN departments d USING(department_id)
    WHERE d.name = 'Administracja'
);

--------------------------------------------------------------------------------

-- Napisz zapytanie, które zwróci informacje o pracownikach zatrudnionych po
zakończeniu wszystkich projektów (tabela projects). Zapytanie zrealizuj na 2
sposoby i porównaj wyniki

SELECT *
FROM employees e
WHERE e.date_employed > ALL (
    SELECT DISTINCT date_end
    FROM projects
    WHERE date_end IS NOT NULL
);
```

```
SELECT *
FROM employees e
WHERE e.date_employed > (
    SELECT MAX(date_end)
    FROM projects
);

-- Napisz zapytanie, które wyświetli wszystkich pracowników, których zarobki są co
najmniej czterokrotnie większe od zarobków jakiegokolwiek innego pracownika.

SELECT *
FROM employees
WHERE salary >= ANY (
    SELECT 4 * salary
    FROM employees
);

-- Korzystając z podzapytań napisz zapytanie które zwróci pracowników
departamentów mających siedziby w Polsce.

SELECT *
FROM employees e
WHERE e.department_id IN (
    SELECT department_id
    FROM departments d
        JOIN addresses USING (address_id)
        JOIN countries c USING (country_id)
    WHERE c.name = 'Polska'
);

-- Zmodyfikuj poprzednie zapytania tak, żeby dodatkowo pokazać maksymalną pensję
per departament.

SELECT d.name, max(e.salary)
FROM employees e
    JOIN departments d ON (d.department_id = e.department_id)
WHERE e.department_id IN (
    SELECT department_id
    FROM departments d
        JOIN addresses USING (address_id)
        JOIN countries c USING (country_id)
    WHERE c.name = 'Polska'
)
GROUP BY d.department_id, d.name;

--------------------------------------------------------------------------------

-- Napisz zapytanie, które zwróci pracowników zarabiających więcej niż średnia w
ich departamencie.

SELECT e1.name, e1.surname, e1.salary
FROM employees e1
WHERE e1.salary > (
    SELECT AVG(e2.salary)
```

```
    FROM employees e2
    WHERE e2.department_id = e1.department_id
);

-- Napisz zapytanie które zwróci regiony nieprzypisane do krajów

SELECT *
FROM regions r
WHERE NOT EXISTS (
    SELECT *
    FROM countries
        JOIN reg_countries rg USING(country_id)
    WHERE rg.region_id = r.region_id
);

SELECT *
FROM regions r
WHERE r.region_id NOT IN (
    SELECT DISTINCT rg.region_id
    FROM countries
        JOIN reg_countries rg USING(country_id)
);

-- Napisz zapytanie które zwróci kraje nieprzypisane do regionów

SELECT *
FROM countries c
WHERE NOT EXISTS (
    SELECT *
    FROM regions
        JOIN reg_countries rg USING(region_id)
    WHERE rg.country_id = c.country_id AND region_id IS NOT NULL
);

SELECT *
FROM countries c
WHERE c.country_id NOT IN (
    SELECT DISTINCT country_id
    FROM regions
    JOIN reg_countries rg USING(region_id)
);

-- Napisz zapytanie, które zwróci wszystkich pracowników niebędących managerami.

SELECT *
FROM employees e1
WHERE NOT EXISTS (
    SELECT *
    FROM employees e2
    WHERE e2.manager_id = e1.employee_id
);

-- Lub:
```

```sql
SELECT *
FROM employees e1
WHERE e1.employee_id NOT IN (
    SELECT DISTINCT manager_id
    FROM employees e2
    WHERE manager_id IS NOT NULL
);

-- Napisz zapytanie, które zwróci dane pracowników, którzy zarabiają więcej niż
średnie zarobki na stanowisku, na którym pracują

SELECT *
FROM employees e1
WHERE e1.salary > (
    SELECT AVG(e2.salary)
    FROM employees e2
    WHERE e2.position_id = e1.position_id
);

-- Za pomocą podzapytania skorelowanego sprawdź, czy wszystkie stanowiska
zdefiniowane w tabeli Positions są aktualnie zajęte przez pracowników.

SELECT *
FROM positions p
WHERE NOT EXISTS (
    SELECT *
    FROM employees e
    WHERE e.position_id = p.position_id
);

-- Nieskorelowane:
SELECT *
FROM positions p
WHERE p.position_id NOT IN (
    SELECT DISTINCT e.position_id
    FROM employees e
    WHERE e.position_id IS NOT NULL
);


--------------------------------------------------------------------------------

SELECT *
FROM DUAL;

-- Napisz zapytanie, które dla wszystkich pracowników posiadających pensję zwróci
informację o różnicy między ich pensją, a średnią pensją pracowników. Różnicę
podaj jako zaokrągloną wartość bezwzględną.

SELECT e.name, e.surname, e.salary, ABS(e.salary - (SELECT ROUND(AVG(salary)) sal
FROM employees))
FROM employees e
WHERE e.salary IS NOT NULL;
```

```sql
SELECT e.name, e.surname, e.salary, ABS(e.salary - a.sal)
FROM employees e, (SELECT ROUND(AVG(salary)) sal FROM employees) a
WHERE e.salary IS NOT NULL;


-- Korzystając z poprzedniego rozwiązania, napisz zapytanie, które zwróci tylko
tych pracowników, którzy są kobietami i dla których różnica do wartości średniej
jest powyżej 1000.

SELECT e.name, e.surname, e.salary, e.gender, ABS(e.salary - a.sal) diff
FROM employees e, (SELECT ROUND(AVG(salary)) sal FROM employees WHERE gender =
'K') a
WHERE e.salary IS NOT NULL AND e.gender = 'K' AND ABS(e.salary - a.sal) > 1000;

-- Zmodyfikuj poprzednie zapytanie tak aby obliczyć liczbe pracowników.
(skorzystaj z podzapytania)

SELECT COUNT(*) FROM (
    SELECT e.name, e.surname, e.salary, e.gender, ABS(e.salary - a.sal) diff
    FROM employees e, (SELECT ROUND(AVG(salary)) sal FROM employees WHERE gender =
'K') a
    WHERE e.salary IS NOT NULL AND e.gender = 'K' AND ABS(e.salary - a.sal) > 1000
);

-- Napisz zapytanie które zwróci informacje o pracownikach zatrudnionych po
zakończeniu wszystkich projektów (tabela projects). W wynikach zapytania umieść
jako kolumnę datę graniczną.

SELECT *
FROM employees, (SELECT MAX(date_end) date_end FROM projects)
WHERE date_employed > date_end;

-- Napisz zapytanie które zwróci pracowników którzy uzyskali w 2019 oceny wyższe
niż średnia w swoim departamencie. Pokaż średnią departamentu jako kolumnę.

SELECT *
FROM employees e1
    JOIN (
            SELECT AVG(e2.salary) sal_avg, e2.department_id
            FROM employees e2
            GROUP BY e2.department_id
        ) USING (department_id)
WHERE e1.salary > sal_avg;

--------------------------------------------------------------------------------


-- Skonstruuj po jednym zapytaniu, które będzie zawierać w klauzuli WHERE:
-- a. podzapytanie zwracające tylko jedną wartość;
SELECT *
FROM employees
WHERE salary > (
    SELECT AVG(salary)
    FROM employees
```

```
);

-- b. podzapytanie zwracające jeden wiersz danych, ale wiele kolumn;

SELECT *
FROM employees
WHERE (name, surname) = (
    SELECT name, surname
    FROM employees
    ORDER BY 1, 2
    FETCH FIRST 1 ROW ONLY
);

-- c. podzapytanie zwracające jedną kolumnę danych;

SELECT *
FROM employees
WHERE employee_id <= ALL (
    SELECT employee_id
    FROM employees
);

-- d. podzapytanie zwracające tabelę danych.

SELECT *
FROM employees
WHERE (name, surname) IN (
    SELECT name, surname
    FROM employees
    ORDER BY salary DESC NULLS LAST
    FETCH FIRST 5 ROWS ONLY
);

-- Napisz zapytanie, które zwróci pracowników będących kierownikami zakładów, o
ile ich zarobki są większe niż średnia zarobków dla wszystkich pracowników.

SELECT *
FROM employees
WHERE employee_id IN (
    SELECT manager_id
    FROM employees
    WHERE manager_id IS NOT NULL
)
AND
salary > (
    SELECT AVG(salary)
    FROM employees
);

-- Zmodyfikuj powyższe zapytanie tak, aby wyświetlało wszystkich pracowników
będących kierownikami zakładów, o ile ich zarobki są większe niż średnia zarobków
na stanowisku które zajmują
```

```sql
SELECT *
FROM employees e1
WHERE employee_id IN (
    SELECT manager_id
    FROM employees
    WHERE manager_id IS NOT NULL
)
AND
salary > (
    SELECT AVG(salary)
    FROM employees e2
    WHERE e2.position_id = e1.position_id
);


-- Wyszukaj informacje w Internecie, dokumentacji bazy danych Oracle lub w
dostarczonych materiałach Oracle Academy o sposobie wykonywania podzapytań
skorelowanych.

-- W których klauzulach polecenia SELECT możemy wykorzystać podzapytania
nieskorelowane?

-- SELECT

SELECT (SELECT AVG(salary) FROM employees)
FROM DUAL;

-- FROM

SELECT *
FROM (SELECT AVG(salary) FROM employees);

-- WHERE

SELECT *
FROM employees
WHERE date_employed < (SELECT SYSDATE FROM DUAL);

-- HAVING

SELECT e.department_id
FROM employees e
GROUP BY e.department_id
HAVING e.department_id IN (SELECT department_id FROM departments WHERE year_budget
> 1000);

-- W których klauzulach polecenia SELECT możemy wykorzystać podzapytania
skorelowane?

-- SELECT

SELECT e1.employee_id, (SELECT AVG(e2.employee_id) FROM employees e2 WHERE
e2.salary = e1.salary)
FROM employees e1;
```

```
-- WHERE

SELECT e1.employee_id, e1.salary
FROM employees e1
WHERE e1.employee_id > (SELECT AVG(e2.employee_id) avg_salary FROM employees e2
WHERE e2.salary = e1.salary);


-- HAVING

SELECT e1.department_id
FROM employees e1
GROUP BY e1.department_id
HAVING MIN(e1.salary) <= (SELECT AVG(e2.salary) FROM employees e2 WHERE
e2.department_id = e1.department_id);
```