Dział kadr firmy

1 TREŚĆ

Dane: o pracownikach, ich kwalifikacjach (ukończonych przez nich kursach specjalistycznych), strukturze organizacyjnej firmy, np. występujących stanowiskach pracy wraz z wymaganiami ukończenia kursów.

Przykładowe wykazy: pracownicy danego działu, pracownicy posiadający określone kwalifikacje, gotowi do objęcia stanowiska, z podziałem na działy.

1.1 OPIS PROJEKTU

1.1.1 Encje i atrybuty

- dział: id, nazwa, ilosc_stanowisk
- kwalifikacja: id, nazwa, opis
- wymagania: id, nazwa, poziom
- stanowisko: id, nazwa, dzial_id, minimamlne_wyksztalcenie, wynagrodzenie
- pracownik: nr_hr, imie, nazwisko, pesel, data_urodzenia, numer_telefonu, miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby, kwalifikacja_id, stanowisko_id
- stanowisko_wymagania: stanowisko_id, wymagania_id

1.1.2 Związki

- pracownik N:1 kwalifikacja
- wymagania N:M stanowisko
- stanowisko N:1 dzial
- pracownik N:1 stanowisko

1.1.3 Pytania

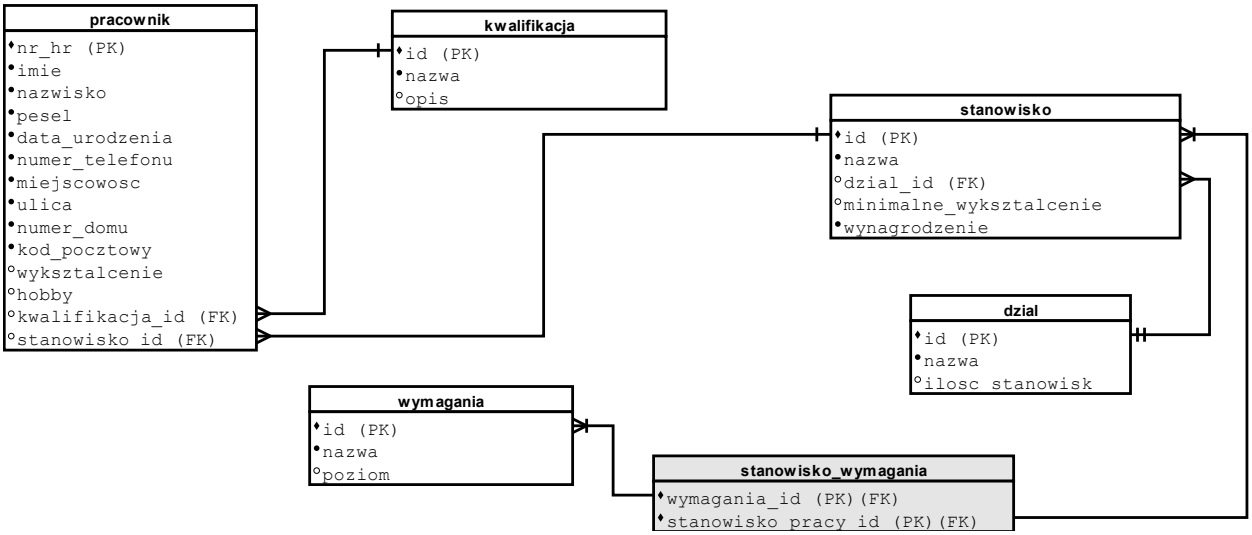
- pracownicy w działach
- ilość pracowników w działach
- wiek pracowników
- wypisanie ilości osób z kwalifikacją
- wypisanie kwalifikacji z największą ilością osób
- wypisanie wsyzstkich wymagań bez duplikatów, które zaczynają się od słowa „Znajomość”
- wypisanie pracowników bez stanowiska
- wypisanie hobby w rekordach
- wypisanie par wymagania stanowisko
- wypisanie pracowników bez stanowiska
- utworzenie widoku ilości osób na stanowisku
- skasowanie stanowiska z id 1
- ustawienie stanowiska id = 2 wszystkich pracowników, którzy mają stanowisko 4
- wypisanie ilości pracowników w działach wraz z ilością dostępnych miejsc w dziale

- wypisanie działu, który ma więcej pracowników niż powinien mieć
- wypisanie pracowników wraz z wynagrodzeniem, którzy są w dziale, w którym jest za mało stanowisk
- nadanie premii zależnej od wynagrodzenia
- nadanie premii zależnej od wynagrodzenia z wykluczeniem wynagrodzeń, które są przypisane do większej ilości stanowisk niż 2
- nadanie premii zależnej od wynagrodzenia z wykluczeniem wynagrodzeń, które są przypisane do większej ilości stanowisk niż 2 oraz dla pracowników z wykształceniem zasadniczym zawodowym i średnim, wykluczając dział kadr
- wypisanie minimalnego, średniego i maksymalnego wieku pracowników w dziale dla parzystych nr_hr
- wypisanie najmłodszych i najstarszych pracowników w każdym dziale uwzgledniając tylko parzyste nr_hr

2 DIAGRAM ZWIĄZKÓW I ENCJI

2.1 ZWIĄZKI

- pracownik N:1 kwalifikacja
- wymagania N:M stanowisko
- stanowisko N:1 dzial
- pracownik N:1 stanowisko



3 FRAGMENTY TABEL

3.1 DZIAL

	id integer	nazwa character varying (50)	ilosc_stanowisk integer
1	1	IT	5
2	2	Finanse	2
3	3	Kadry	5

3.2 KWALIFIKACJA

	id integer	nazwa character varying (50)	opis character varying (500)
1	1	Poziom 1	Kwalifikacje niezbędne do wykonywania prostych, rutynowych zadań wykonywanych pod kierunkiem przeło...
2	2	Poziom 2	Kwalifikacje niezbędne do wykonywania zadań w sytuacjach typowych. Niektóre zadania mogą być bardziej...
3	3	Poziom 3	Kwalifikacje niezbędne do wykonywania zadań złożonych, zarówno w warunkach typowych jak i problemow...

3.3 WYMAGANIA

	id integer	nazwa character varying (255)	poziom integer
1	1	Znajomość pakietu Microsoft Office	1
2	2	Znajomość pakietu Microsoft Office	2
3	3	Znajomość pakietu Microsoft Office	3

3.4 STANOWISKO

	id integer	nazwa character varying (50)	dzial_id integer	minimalne_wyksztalcenie character varying (50)	wynagrodzenie numeric (10,2)
1	1	Sekretarka	5	zasadnicze zawodowe	2200.00
2	2	Handlowiec	6	średnie	3500.00
3	3	Marketingowiec	5	wyższe	4100.00

3.5 PRACOWNIK

	nr_hr integer	imie character varying (25)	nazwisko character varying (50)	pesel bigint	data_urodzenia date	numer_telefonu bigint	miejscowosc character varying (50)	ulica character varying (50)	numer_domu character varying (25)
1	1	Joyce	Jast	07850958	1993-07-23	362456459	Caliport	Aditya Ridge	9
2	2	Norma	Kreiger	45431375	2003-01-17	770018709	New Agnes	Ivy Villages	45
3	3	Larue	Boehm	03227344	1991-07-25	656992298	Lake Annettaport	Nikolaus Ranch	83

kod_pocztowy character varying (6)	wyksztalcenie character varying (25)	hobby character varying (255)	kwalifikacja_id integer	stanowisko_id integer
85-099	zasadnicze zawodowe	kino, grafika, pisanie książe...	1	1
48-950	średnie	pisanie książek, grafika	3	4
88-364	średnie	teatr, elektronika, czytanie, ...	2	6

3.6 STANOWISKO_WYMAGANIA

stanowisko_id integer	wymagania_id integer
1	9
1	11
1	13

4 PRZYKŁADY

4.1 ZAPYTANIA

4.1.1 pracownicy w dzialach

```
-- pracownicy w dzialach
SELECT imie, nazwisko, dzial.nazwa
FROM pracownik
INNER JOIN stanowisko ON stanowisko_id = stanowisko.id
INNER JOIN dzial ON dzial_id = dzial.id;
```

	imie character varying (25)	nazwisko character varying (50)	nazwa character varying (50)
1	Cloyd	Stamm	IT
2	Maynard	Swift	IT
3	Clarissa	Robel	IT
4	Edison	Mertz	IT
5	Abelardo	Heidenreich	IT
6	Lexi	Daniel	Finanse
7	Johnson	Grimes	Finanse
8	Austin	Ferry	Finanse

4.1.2 ilosc pracownikow w dzialach

```
-- ilosc pracownikow w dzialach
SELECT dzial.nazwa, COUNT(nr_hr)
FROM dzial
INNER JOIN stanowisko ON stanowisko.dzial_id = dzial.id
INNER JOIN pracownik ON stanowisko.id = stanowisko_id
GROUP BY dzial.nazwa
ORDER BY dzial.nazwa;
```

	nazwa character varying (50)	count bigint
1	Finanse	5
2	Handlowy	4
3	IT	5
4	Kadry	5
5	Marketing	6
6	Obsługa klienta	5

4.1.3 wiek pracownikow

```
-- wiek pracownikow
SELECT imie, nazwisko, DATE_PART('year', NOW()) - DATE_PART('year',
data_urodzenia) AS wiek
FROM pracownik
ORDER BY DATE_PART('year', NOW()) - DATE_PART('year', data_urodzenia)
```

	imie character varying (25)	nazwisko character varying (50)	wiek double precision
1	Raegan	Cummerata	1
2	Zoila	Simonis	1
3	Wilbert	Murray	4
4	Aryanna	Wyman	5
5	Sidney	Brown	5
6	Edison	Mertz	8

4.1.4 wypisanie ilosci osob z kwalifikacji

```
-- wypisanie ilosci osob z kwalifikacji
SELECT nazwa, COUNT(nr_hr)
FROM kwalifikacja
INNER JOIN pracownik ON pracownik.kwalifikacja_id = id
GROUP BY nazwa;
```

	nazwa character varying (50)	count bigint
1	Poziom 5	6
2	Poziom 3	16
3	Poziom 1	10
4	Poziom 2	12
5	Poziom 4	10

4.1.5 wypisanie kwalifikacji z najwieksza iloscia osob

```
-- wypisanie kwalifikacji z najwieksza iloscia osob
WITH x AS (
    SELECT nazwa, COUNT(nr_hr) AS ilosc
    FROM kwalifikacja
    INNER JOIN pracownik ON kwalifikacja_id = id
    GROUP BY nazwa
)

SELECT * FROM x
WHERE ilosc = (SELECT MAX(ilosc) FROM x);
```

	nazwa character varying (50)	ilosc bigint
1	Poziom 3	16

4.1.6 wypisanie wszystkich wymagan bez duplikatow, ktore zaczynaja sie od slowa 'Znajomość'

```
-- wypisanie wszystkich wymagan bez duplikatow, ktore zaczynaja sie od slowa
'Znajomość'
SELECT DISTINCT nazwa FROM wymagania
WHERE nazwa LIKE 'Znajomość%'
ORDER BY nazwa;
```

	nazwa character varying (255)
1	Znajomość analizy finansowej
2	Znajomość AutoCAD
3	Znajomość branży
4	Znajomość budżetowania
5	Znajomość języków obcych
6	Znajomość ogólnych zasad rachunkowości oraz ewidencji księgowej
7	Znajomość pakietu Microsoft Office
8	Znajomość Power BI
9	Znajomość prawa pracy

4.1.7 wypisanie pracownikow bez stanowiska

```
-- wypisanie pracownikow bez stanowiska
SELECT imie, nazwisko FROM pracownik
WHERE stanowisko_id IS NULL;
```

	imie character varying (25)	nazwisko character varying (50)
1	Raegan	Cummerata
2	Adam	Wisozk
3	Eloisa	Rippin
4	Will	Kemmer
5	Cristopher	Feil
6	Davonte	Denesik
7	Dejah	Swaniawski

4.1.8 wypisanie hobby w rekordach

```
-- wypisanie hobby w rekordach
SELECT TRIM(REGEXP_SPLIT_TO_TABLE(p.hobby, E',')) AS hobby
FROM pracownik AS p;
```

	hobby text
1	kino
2	grafika
3	pisanie książek
4	park
5	gry
6	programowanie
7	pisanie książek

4.1.9 wypisanie par wymagania stanowisko

```
-- wypisanie par wymagania stanowisko
WITH x AS (
    SELECT DISTINCT w.nazwa AS wymagania, s.nazwa AS stanowisko
    FROM wymagania AS w
```

```
    LEFT JOIN stanowisko_wymagania AS sw ON sw.wymagania_id = w.id
    LEFT JOIN stanowisko AS s ON s.id = sw.stanowisko_id
    ORDER BY w.nazwa
)

SELECT x.wymagania AS wymagania, STRING_AGG(x.stanowisko, ', ') as stanowiska,
COUNT(x.stanowisko)
FROM x
GROUP BY x.wymagania
ORDER BY x.wymagania;
```

	wymagania character varying (255)	stanowiska text	count bigint
1	Asertywność	[null]	0
2	Doświadczenie w prowadze...	Kadrowa	1
3	Etyka	Informatyk, K...	3
4	Inicjatywa	Handlowiec, I...	3
5	Komunikatywność	Handlowiec	1

4.1.10 wypisanie pracownikow bez stanowiska

```
-- wypisanie pracownikow bez stanowiska
SELECT p.imie, p.nazwisko FROM pracownik AS p
LEFT OUTER JOIN stanowisko AS s ON p.stanowisko_id = s.id
WHERE p.stanowisko_id IS NULL
ORDER BY p.imie, p.nazwisko;
```

	imie character varying (25)	nazwisko character varying (50)
1	Adam	Wisozk
2	Alexane	Ferry
3	Anne	Bosco
4	Aryanna	Wyman

4.1.11 utworzenie widoku ilosci osob na stanowisku

```
-- utworzenie widoku ilosci osob na stanowisku
CREATE VIEW ilosc_osob_na_stanowisku AS
SELECT s.nazwa, COUNT(p.nr_hr) AS ilosc
FROM stanowisko AS s
LEFT JOIN pracownik AS p ON p.stanowisko_id = s.id
GROUP BY s.nazwa;

SELECT * FROM ilosc_osob_na_stanowisku;
```

	nazwa character varying (50)	ilosc bigint
1	Informatyk	2
2	Doradca klienta	2
3	Marketingowiec	0
4	Sekretarka	3
5	Projektant	1
6	Analityk	2

4.1.12 skasowanie stanowiska z id 1; w tablicy pracownik ustawia NULL w kolumnie stanowisko_id

```
-- skasowanie stanowiska z id 1
-- w tablicy pracownik ustawia NULL w kolumnie stanowisko_id
DELETE FROM stanowisko
WHERE id = 1;
```

```
DELETE 1

Query returned successfully in 231 msec.
```

4.1.13 ustawienie stanowiska id = 2 wszystkim pracownikom, ktorzy maja stanowisko id = 4

```
-- ustawienie stanowiska id = 2 wszystkim pracownikom, ktorzy maja stanowisko id
= 4
UPDATE pracownik SET stanowisko_id = 2 WHERE stanowisko_id = 4;
```

```
UPDATE 2

Query returned successfully in 282 msec.
```

4.1.14 wypisanie ilosci pracownikow w dzialach wraz z iloscia dostepnych miejsc w dziale

```
-- wypisanie ilosci pracownikow w dzialach wraz z iloscia dostepnych miejsc w
dziale
SELECT
    dzial.nazwa,
    dzial.ilosc_stanowisk,
    COUNT(pracownik.nr_hr) AS "ilosc_pracownikow_w_dziale"
FROM dzial
LEFT JOIN stanowisko ON stanowisko.dzial_id = dzial.id
LEFT JOIN pracownik ON pracownik.stanowisko_id = stanowisko.id
GROUP BY dzial.nazwa, dzial.ilosc_stanowisk;
```

	nazwa character varying (50)	ilosc_stanowisk integer	ilosc_pracownikow_w_dziale bigint
1	Kadry	5	5
2	Obsługa klienta	8	5
3	IT	5	5
4	Finanse	2	3
5	Marketing	9	3
6	Handlowy	4	6

4.1.15 wypisanie dzialu, ktory ma wiecej pracownikow niz powinien miec

```
-- wypisanie dzialu, ktory ma wiecej pracownikow niz powinien miec
SELECT
    dzial.nazwa,
    dzial.ilosc_stanowisk,
    COUNT(pracownik.nr_hr) AS "ilosc_pracownikow_w_dziale"
FROM dzial
LEFT JOIN stanowisko ON stanowisko.dzial_id = dzial.id
LEFT JOIN pracownik ON pracownik.stanowisko_id = stanowisko.id
GROUP BY dzial.nazwa, dzial.ilosc_stanowisk
HAVING COUNT(pracownik.nr_hr) > dzial.ilosc_stanowisk;
```

	nazwa character varying (50)	ilosc_stanowisk integer	ilosc_pracownikow_w_dziale bigint
1	Finanse	2	3
2	Handlowy	4	6

4.1.16 wypisanie pracownikow wraz z wynagrodzeniem, ktorzy sa w dziale, w ktorym jest za malo stanowisk

```
-- wypisanie pracownikow wraz z wynagrodzeniem, ktorzy sa w dziale, w ktorym jest
za malo stanowisk
WITH x AS (
    SELECT
        dzial.id,
        dzial.nazwa,
        dzial.ilosc_stanowisk,
        COUNT(pracownik.nr_hr) AS "ilosc_pracownikow_w_dziale"
    FROM dzial
    LEFT JOIN stanowisko ON stanowisko.dzial_id = dzial.id
    LEFT JOIN pracownik ON pracownik.stanowisko_id = stanowisko.id
    GROUP BY dzial.id, dzial.nazwa, dzial.ilosc_stanowisk
    HAVING COUNT(pracownik.nr_hr) > dzial.ilosc_stanowisk
)

SELECT
    pracownik.imie,
    pracownik.nazwisko,
```

```
    stanowisko.wynagrodzenie
FROM x
LEFT JOIN stanowisko ON stanowisko.dzial_id = x.id
LEFT JOIN pracownik ON pracownik.stanowisko_id = stanowisko.id;
```

	imie character varying (25)	nazwisko character varying (50)	wynagrodzenie numeric (10,2)
1	Eldridge	West	3800.00
2	Rosina	Grimes	3220.00
3	Anibal	Bradtke	3220.00

4.1.17 nadanie premii zaleznej od wynagrodzenia

```
-- nadanie premii zaleznej od wynagrodzenia
WITH x AS (
    SELECT
        pracownik.imie,
        pracownik.nazwisko,
        stanowisko.nazwa,
        stanowisko.wynagrodzenie
    FROM stanowisko
    LEFT JOIN pracownik ON pracownik.stanowisko_id = stanowisko.id
)

SELECT
    x.imie,
    x.nazwisko,
    x.nazwa,
    ROUND(CASE
        WHEN x.wynagrodzenie BETWEEN 0 AND 1500 THEN x.wynagrodzenie * 0.25
        WHEN x.wynagrodzenie BETWEEN 1500.01 AND 2500 THEN x.wynagrodzenie * 0.20
        WHEN x.wynagrodzenie BETWEEN 2500.01 AND 3500 THEN x.wynagrodzenie * 0.15
        WHEN x.wynagrodzenie BETWEEN 3500.01 AND 5000 THEN x.wynagrodzenie * 0.10
    END, 2) AS "premia"
FROM x;
```

	imie character varying (25)	nazwisko character varying (50)	nazwa character varying (50)	premia numeric
1	Sidney	Brown	Handlowiec	525.00
2	Norma	Kreiger	Handlowiec	525.00
3	[null]	[null]	Marketingowiec	410.00
4	[null]	[null]	Księgowy	360.00
5	Zoila	Simonis	Kadrowa	400.00
6	Marvin	Torphy	Kadrowa	400.00
7	Kian	Hettinger	Doradca klienta	500.00

4.1.18 nadanie premii zaleznej od wynagrodzenia z wykluczeniem wynagrodzen, ktore sa przypisane; do wiekszej ilosci stanowisk niz 2

```sql
/*
nadanie premii zaleznej od wynagrodzenia z wykluczeniem wynagrodzen, ktore sa
przypisane
do wiekszej ilosci stanowisk niz 2
*/
-- wykaz_pracownikow
WITH x AS (
    SELECT
        p.imie,
        p.nazwisko,
        s.nazwa,
        s.wynagrodzenie,
        p.stanowisko_id,
        s.dzial_id,
        sw.wymagania_id
    FROM stanowisko AS s
    LEFT JOIN pracownik AS p ON p.stanowisko_id = s.id
    LEFT JOIN stanowisko_wymagania AS sw ON sw.stanowisko_id = p.stanowisko_id
    WHERE p.nr_hr IS NOT NULL
),

-- wykaz_wymagan
y AS (
    SELECT
        w.id,
        w.nazwa AS wymagania,
        COUNT(s.id) AS "ilosc_stanowisk"
    FROM wymagania AS w
    LEFT JOIN stanowisko_wymagania AS sw ON sw.wymagania_id = w.id
    LEFT JOIN stanowisko AS s ON s.id = sw.stanowisko_id
    GROUP BY w.id, w.nazwa
)

SELECT DISTINCT
    x.imie,
    x.nazwisko,
    x.nazwa,
    ROUND(CASE
        WHEN x.wynagrodzenie BETWEEN 0 AND 1500 THEN x.wynagrodzenie * 0.25
        WHEN x.wynagrodzenie BETWEEN 1500.01 AND 2500 THEN x.wynagrodzenie * 0.20
        WHEN x.wynagrodzenie BETWEEN 2500.01 AND 3500 THEN x.wynagrodzenie * 0.15
        WHEN x.wynagrodzenie BETWEEN 3500.01 AND 5000 THEN x.wynagrodzenie * 0.10
    END, 2) AS "premia",
```

```
    dzial.nazwa
FROM x
LEFT JOIN dzial ON dzial.id = x.dzial_id
LEFT JOIN y ON y.id = x.wymagania_id
WHERE y.ilosc_stanowisk < 3
```

Data Output | Explain | Messages | Query History

	imie character varying (25)	nazwisko character varying (50)	nazwa character varying (50)	premia numeric	nazwa character varying (50)
1	Abelardo	Heidenreich	Informatyk	300.00	IT
2	Anibal	Bradtke	Manager	483.00	Handlowy
3	Austin	Ferry	Analityk	360.00	Finanse
4	Cathryn	Tremblay	Manager	390.00	Marketing
5	Clarissa	Robel	Manager	380.00	IT
6	Connor	Schimmel	Manager	420.00	Obsługa klienta

4.1.19 nadanie premii zaleznej od wynagrodzenia z wykluczeniem wynagrodzen, ktore sa przypisane; do wiekszej ilosci stanowisk niz 2 oraz dla pracownikow z wyksztalceniem zasadniczym zawodowym; i srednim, wykluczajac dzial Kadr

```
/*
nadanie premii zaleznej od wynagrodzenia z wykluczeniem wynagrodzen, ktore sa
przypisane
do wiekszej ilosci stanowisk niz 2 oraz dla pracownikow z wyksztalceniem
zasadniczym zawodowym
i srednim, wykluczajac dzial Kadr
*/
-- wykaz_pracownikow
WITH x AS (
    SELECT
        p.nr_hr,
        s.nazwa,
        s.wynagrodzenie,
        p.stanowisko_id,
        s.dzial_id,
        sw.wymagania_id
    FROM stanowisko AS s
    LEFT JOIN pracownik AS p ON p.stanowisko_id = s.id
    LEFT JOIN stanowisko_wymagania AS sw ON sw.stanowisko_id = p.stanowisko_id
    WHERE p.nr_hr IS NOT NULL
),
-- wykaz_wymagan
y AS (
    SELECT
        w.id,
        w.nazwa AS wymagania,
        COUNT(s.id) AS "ilosc_stanowisk"
    FROM wymagania AS w
```

```
    LEFT JOIN stanowisko_wymagania AS sw ON sw.wymagania_id = w.id
    LEFT JOIN stanowisko AS s ON s.id = sw.stanowisko_id
    GROUP BY w.id, w.nazwa
)

SELECT DISTINCT
    p.imie,
    p.nazwisko,
    x.nazwa,
    ROUND(CASE
        WHEN x.wynagrodzenie BETWEEN 0 AND 1500 THEN x.wynagrodzenie * 0.25
        WHEN x.wynagrodzenie BETWEEN 1500.01 AND 2500 THEN x.wynagrodzenie * 0.20
        WHEN x.wynagrodzenie BETWEEN 2500.01 AND 3500 THEN x.wynagrodzenie * 0.15
        WHEN x.wynagrodzenie BETWEEN 3500.01 AND 5000 THEN x.wynagrodzenie * 0.10
    END, 2) AS "premia",
    dzial.nazwa
FROM x
LEFT JOIN pracownik AS p ON p.nr_hr = x.nr_hr
LEFT JOIN dzial ON dzial.id = x.dzial_id
LEFT JOIN y ON y.id = x.wymagania_id
WHERE y.ilosc_stanowisk < 3
AND p.wyksztalcenie IN ('zasadnicze zawodowe', 'średnie')
AND x.dzial_id <> 3;
```

	imie character varying (25)	nazwisko character varying (50)	nazwa character varying (50)	premia numeric	nazwa character varying (50)
1	Anibal	Bradtke	Manager	483.00	Handlowy
2	Clarissa	Robel	Manager	380.00	IT
3	Dallas	Satterfield	Manager	483.00	Handlowy
4	Edison	Mertz	Informatyk	300.00	IT
5	Johnson	Grimes	Analityk	360.00	Finanse
6	Lexi	Daniel	Manager	390.00	Finanse
7	Lon	Deckow	Manager	390.00	Marketing
8	Norma	Kreiger	Handlowiec	525.00	Handlowy

4.1.20 ustawienie pracownikom z pensja ponizej 2500 nowa stawke o 300 zl wieksza,; dodatkowo pracownikom z dzialu IT dodac 20 zl wyrownania do nowej stawki

```
/*
ustawienie pracownikom z pensja ponizej 2500 nowa stawke o 300 zl wieksza,
dodatkowo pracownikom z dzialu IT dodac 20 zl wyrownania do nowej stawki
*/
UPDATE stanowisko SET wynagrodzenie = wynagrodzenie + 300
WHERE wynagrodzenie < 2500;

UPDATE stanowisko SET wynagrodzenie = wynagrodzenie + 20
WHERE dzial_id = 1;
```

```
UPDATE 3

Query returned successfully in 251 msec.
```

4.1.21 wypisanie minimalnego, sredniego i maksymalnego wieku pracownikow w dziale; dla parzystych nr_hr

```
/*
wypisanie minimalnego, sredniego i maksymalnego wieku pracownikow w dziale
dla parzystych nr_hr
*/
WITH x AS (
    SELECT nr_hr, DATE_PART('year', NOW()) - DATE_PART('year', data_urodzenia) AS
wiek
    FROM pracownik
)

SELECT
    d.nazwa,
    MIN(x.wiek) AS "min_wiek",
    ROUND(AVG(x.wiek)) AS "avg_wiek",
    MAX(x.wiek) AS "max_wiek"
FROM x
LEFT JOIN pracownik AS p ON p.nr_hr = x.nr_hr
LEFT JOIN stanowisko AS s ON s.id = p.stanowisko_id
LEFT JOIN dzial AS d ON d.id = s.dzial_id
WHERE d.nazwa IS NOT NULL AND p.nr_hr % 2 = 0
GROUP BY d.nazwa
```

	nazwa character varying (50)	min_wiek double precision	avg_wiek double precision	max_wiek double precision
1	Finanse	34	36	39
2	Handlowy	11	24	46
3	IT	8	13	17
4	Kadry	1	24	47
5	Obsługa klienta	14	14	14

4.1.22 wypisanie najmlodszych i najstarszych pracownikow w kazdym dziale; uwzgledniajac tylko parzyste nr_hr

```
/*
wypisanie najmlodszych i najstarszych pracownikow w kazdym dziale
uwzgledniajac tylko parzyste nr_hr
*/
WITH x AS (
    SELECT
```

```
        d.id,
        MIN(DATE_PART('year', NOW()) - DATE_PART('year', p.data_urodzenia)) AS
"min_wiek",
        MAX(DATE_PART('year', NOW()) - DATE_PART('year', p.data_urodzenia)) AS
"max_wiek"
    FROM dzial AS d
    LEFT JOIN stanowisko AS s ON s.dzial_id = d.id
    LEFT JOIN pracownik AS p ON p.stanowisko_id = s.id
    WHERE d.id IS NOT NULL AND p.nr_hr % 2 = 0
    GROUP BY d.id
),
y AS (
    SELECT
        p.nr_hr,
        d.id,
        CONCAT(p.imie, ' ', p.nazwisko) AS "pracownik",
        DATE_PART('year', NOW()) - DATE_PART('year', p.data_urodzenia) AS "wiek"
    FROM pracownik AS p
    LEFT JOIN stanowisko AS s ON s.id = p.stanowisko_id
    LEFT JOIN dzial AS d ON d.id = s.dzial_id
    WHERE d.id IS NOT NULL AND p.nr_hr % 2 = 0
)

SELECT
    d.nazwa,
    y.pracownik AS "min_pracownik",
    MIN(x.min_wiek),
    y2.pracownik AS "max_pracownik",
    MAX(x.max_wiek)
FROM x
LEFT JOIN dzial AS d ON d.id = x.id
LEFT JOIN y AS y ON y.wiek = x.min_wiek AND y.id = x.id
LEFT JOIN y AS y2 ON y2.wiek = x.max_wiek AND y2.id = x.id
GROUP BY d.nazwa, y.pracownik, y2.pracownik;
```

	nazwa character varying (50)	min_pracownik text	min double precision	max_pracownik text	max double precision
1	Finanse	Lexi Daniel	34	Austin Ferry	39
2	Handlowy	Rosina Grimes	11	Dallas Satterfield	46
3	IT	Edison Mertz	8	Maynard Swift	17
4	Kadry	Zoila Simonis	1	Marilou Wolff	47
5	Obsługa klienta	Davonte Raynor	14	Davonte Raynor	14

4.2 WYZWALACZE

4.2.1 wyzwalacz do zmiany wynagrodzenia

4.2.1.1 create

```
-- wyzwalacz do zmiany wynagrodzenia
CREATE TABLE Log_zmiany_wynagrodzenia_UPDATE (
    id SERIAL,
    data_logu TIMESTAMP NOT NULL,
    stanowisko VARCHAR(255),
    stare_wynagrodzenie DECIMAL(10, 2) NOT NULL,
    nowe_wynagrodzenie DECIMAL(10, 2) NOT NULL
);

CREATE FUNCTION dopisz_do_logu()
    RETURNS TRIGGER AS $$
DECLARE
    st VARCHAR(255);
    x DECIMAL(10, 2);

BEGIN
    st := OLD.nazwa;
    x := NEW.wynagrodzenie;
    RAISE NOTICE 'zmiana wynagrodzenia dla %', st || ' na ' || x || ' brutto';

    INSERT INTO Log_zmiany_wynagrodzenia_UPDATE (data_logu, stanowisko,
stare_wynagrodzenie, nowe_wynagrodzenie)
    VALUES (NOW(), st, OLD.wynagrodzenie, NEW.wynagrodzenie);
    RETURN NEW;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER Log_zmiany_wynagrodzenia_UPDATE
AFTER UPDATE ON stanowisko
FOR EACH ROW EXECUTE PROCEDURE dopisz_do_logu();
```

```
CREATE TRIGGER

Query returned successfully in 258 msec.
```

4.2.1.2 example

```
-- zmiana wynagrodzenia
UPDATE stanowisko SET wynagrodzenie = 200 WHERE nazwa = 'Handlowiec';
UPDATE stanowisko SET wynagrodzenie = 3500 WHERE nazwa = 'Kadrowa';
UPDATE stanowisko SET wynagrodzenie = 4200 WHERE nazwa = 'Manager';
```

```
UPDATE stanowisko SET wynagrodzenie = 1200 WHERE nazwa = 'Ksiegowa';
SELECT * FROM Log_zmiany_wynagrodzenia_UPDATE;
```

```
NOTICE:  zmiana wynagrodzenia dla Manager na 4200.00 brutto
NOTICE:  zmiana wynagrodzenia dla Manager na 4200.00 brutto
NOTICE:  zmiana wynagrodzenia dla Manager na 4200.00 brutto
NOTICE:  zmiana wynagrodzenia dla Manager na 4200.00 brutto
NOTICE:  zmiana wynagrodzenia dla Manager na 4200.00 brutto
NOTICE:  zmiana wynagrodzenia dla Manager na 4200.00 brutto
```

4.2.1.3 select

	id integer	data_logu timestamp without time zone	stanowisko character varying (255)	stare_wynagrodzenie numeric (10,2)	nowe_wynagrodzenie numeric (10,2)
1	1	2018-06-02 07:01:51.765674	Handlowiec	3500.00	200.00
2	2	2018-06-02 07:01:51.765674	Kadrowa	4000.00	3500.00
3	3	2018-06-02 07:01:51.765674	Manager	2600.00	4200.00
4	4	2018-06-02 07:01:51.765674	Manager	2650.00	4200.00
5	5	2018-06-02 07:01:51.765674	Manager	3900.00	4200.00
6	6	2018-06-02 07:01:51.765674	Manager	3220.00	4200.00
7	7	2018-06-02 07:01:51.765674	Manager	2400.00	4200.00
8	8	2018-06-02 07:01:51.765674	Manager	2220.00	4200.00

4.2.2 wyzwalacz do poczekalni

4.2.2.1 create

```
-- wyzwalacz do poczekalni
CREATE TABLE Poczekalnia_INSERT (
    id SERIAL,
    nr_hr INTEGER,
    imie VARCHAR(25) NOT NULL,
    nazwisko VARCHAR(50) NOT NULL,
    pesel BIGINT NOT NULL,
    na_stanowisko INTEGER NOT NULL
);

CREATE FUNCTION dopisz_do_poczekalni()
    RETURNS TRIGGER AS $$
DECLARE
    hr INTEGER;
    imie VARCHAR(25);
    nazwisko VARCHAR(50);
    na_stanowisko INTEGER;
    ilosc_miejsc INTEGER;
    ilosc_pracownikow_w_dziale INTEGER;

BEGIN
    hr := NEW.nr_hr;
    imie := NEW.imie;
```

```
    nazwisko := NEW.nazwisko;
    na_stanowisko := NEW.stanowisko_id;
    RAISE NOTICE 'pracownik nr HR %', hr || '(' || imie || ' ' || nazwisko || ')
czeka na stanowisko od id ' || na_stanowisko;

    SELECT
        dzial.ilosc_stanowisk INTO ilosc_miejsc
    FROM dzial
    LEFT JOIN stanowisko ON stanowisko.dzial_id = dzial.id
    WHERE stanowisko.id = na_stanowisko;

    SELECT
        COUNT(pracownik.nr_hr) INTO ilosc_pracownikow_w_dziale
    FROM dzial
    LEFT JOIN stanowisko ON stanowisko.dzial_id = dzial.id
    LEFT JOIN pracownik ON pracownik.stanowisko_id = stanowisko.id
    WHERE stanowisko.id = na_stanowisko;

    IF ilosc_miejsc <= ilosc_pracownikow_w_dziale THEN
        INSERT INTO Poczekalnia_INSERT (nr_hr, imie, nazwisko, pesel,
na_stanowisko)
        VALUES (hr, imie, nazwisko, NEW.pesel, na_stanowisko);

        UPDATE pracownik SET stanowisko_id = NULL WHERE nr_hr = hr;
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER sprawdz_miejsca_pracy
AFTER INSERT ON pracownik
FOR EACH ROW EXECUTE PROCEDURE dopisz_do_poczekalni();
```

```
CREATE TRIGGER

Query returned successfully in 280 msec.
```

4.2.2.2 example

```
-- dodanie nowych pracownikow
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('N1', 'S1', '11111111111', '1993-07-23',
'111111111', 'Caliport', 'Aditya Ridge', '9', '85-099', 'zasadnicze zawodowe',
'kino, grafika, pisanie książek, park, gry, programowanie', 1, 3);
```

```
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('N2', 'S2', '22222222222', '2003-01-17',
'222222222', 'New Agnes', 'Ivy Villages', '45', '48-950', 'średnie', 'pisanie
książek, grafika', 3, 4);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('N3', 'S3', '33333333333', '1991-07-25',
'333333333', 'Lake Annettaport', 'Nikolaus Ranch', '83', '88-364', 'średnie',
'teatr, elektronika, czytanie, pisanie książek, wycieczki, czytanie, gry,
elektronika, grafika, taniec, taniec, jeżdzenie na rolkach, opera, taniec,
jeżdzenie na deskorolce', 2, 2);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('N4', 'S4', '44444444444', '1998-08-07',
'444444444', 'Schmidtberg', 'Gutmann Summit', '1', '60-121', 'wyższe',
'wycieczki', 4, 2);
SELECT * FROM Poczekalnia_INSERT;
```

```
NOTICE:  pracownik nr HR 55(N1 S1) czeka na stanowisko od id 3
NOTICE:  pracownik nr HR 56(N2 S2) czeka na stanowisko od id 4

Successfully run. Total query runtime: 245 msec.
1 rows affected.
```

4.2.2.3 select

	id integer	nr_hr integer	imie character varying (25)	nazwisko character varying (50)	pesel bigint	na_stanowisko integer
1	1	56	N2	S2	22222222	4

4.2.3 wyzwalacz, ktory przerzuca wszystkie osoby bez stanowisk do nowo utworzonego dzialu (podanego z kwery) i stanowiska (podanego z funkcji), przydziela X wynagrodzenia (podanego z funkcji) i dopisuje do logow kogo przeniosiono, gdzie i z jakim wynagrodzeniem; dodatkowo wywietlenie ostrzezenia czy pracownik ma mniej niz 18 lat; dodatkowo zliczenie przedzialow wiekowych przeniesionych pracownikow z wykorzystaniem wczesniej utworzonego widoku; jesli nie ma potrzebny tworzenia dzialu, to dzial nie jest tworzony (dzial musi byc najpierw stworzony, bo nie bedzie mozna przypisac go do stanowiska i pracownikow)

4.2.3.1 create

```
CREATE OR REPLACE VIEW przedzialy_wiekowe_praconikow AS
SELECT y.przedzialy, COUNT(y.przedzialy), y.stanowisko_id
FROM (
    SELECT
        CASE
            WHEN x.wiek > 1 AND x.wiek <= 20 THEN 'w1_20'
            WHEN x.wiek > 21 AND x.wiek <= 40 THEN 'w21_40'
```

```
            WHEN x.wiek > 41 AND x.wiek <= 60 THEN 'w41_60'
            WHEN x.wiek > 61 AND x.wiek <= 80 THEN 'w61_80'
            WHEN x.wiek > 81 AND x.wiek <= 100 THEN 'w81_100'
        END AS przedzialy,
        x.stanowisko_id AS stanowisko_id
    FROM (
        SELECT p.nr_hr, DATE_PART('year', NOW()) - DATE_PART('year',
p.data_urodzenia) AS wiek, p.stanowisko_id AS stanowisko_id
        FROM pracownik AS p
        GROUP BY p.nr_hr, p.stanowisko_id
    ) AS x
) AS y
GROUP BY przedzialy, y.stanowisko_id
HAVING COUNT(y.przedzialy) > 0;

CREATE TABLE Przerzuceni_pracownicy (
    id SERIAL PRIMARY KEY,
    nr_hr INTEGER,
    imie VARCHAR(25) NOT NULL,
    nazwisko VARCHAR(50) NOT NULL,
    pesel BIGINT NOT NULL,
    na_stanowisko INTEGER NOT NULL,
    na_dzial INTEGER NOT NULL,
    nowe_wynagrodzenie DECIMAL(10, 2) NOT NULL
);

CREATE FUNCTION przerzuc_pracownikow()
    RETURNS TRIGGER AS $$
DECLARE
    ilosc_stanowisk_tymaczasowych INTEGER;
    s_hr INTEGER;
    s_imie VARCHAR(25);
    s_nazwisko VARCHAR(50);
    s_pesel BIGINT;
    na_stanowisko INTEGER;
    na_dzial INTEGER;
    dzial_nazwa VARCHAR(50);
    ilosc_osob_bez_stanowiska INTEGER;
    czy_przerzucic BOOLEAN;
    nowe_wynagrodzenie DECIMAL(10, 2);
    stanowisko_nazwa VARCHAR(50);
    s_wiek INTEGER;
    wiek_1_20 INTEGER;
    wiek_21_40 INTEGER;
    wiek_41_60 INTEGER;
```

```
    wiek_61_80 INTEGER;
    wiek_81_100 INTEGER;

BEGIN
    nowe_wynagrodzenie := TG_ARGV[0];
    stanowisko_nazwa := TG_ARGV[1];
    dzial_nazwa := NEW.nazwa;

    -- sprawdzenie czy jest potrzeba tworzenia przerzucenia ludzi
    SELECT
        CASE WHEN COUNT(nr_hr) > 0 THEN true ELSE false END,
        COUNT(nr_hr)
    INTO czy_przerzucic, ilosc_osob_bez_stanowiska
    FROM pracownik WHERE stanowisko_id IS NULL;

    IF czy_przerzucic THEN
        -- pobranie ID dzialu
        SELECT id INTO na_dzial FROM dzial WHERE nazwa = dzial_nazwa;

        -- pobranie ilosci stanowisk w tablicy stanowisko i przypisanie kolejnego
id
        SELECT MAX(id) + 1 INTO na_stanowisko FROM stanowisko;

        -- utworzenie nowego stanowiska
        INSERT INTO stanowisko (id, nazwa, dzial_id, minimalne_wyksztalcenie,
wynagrodzenie)
        VALUES (na_stanowisko, stanowisko_nazwa, na_dzial, NULL,
nowe_wynagrodzenie);

        FOR i IN 1..ilosc_osob_bez_stanowiska LOOP
            -- pobranie informacji o pracowniku
            SELECT nr_hr, imie, nazwisko, pesel, DATE_PART('year', NOW()) -
DATE_PART('year', data_urodzenia)
            INTO s_hr, s_imie, s_nazwisko, s_pesel, s_wiek
            FROM pracownik WHERE stanowisko_id IS NULL;

            -- wrzucenie do tablicy 'Przerzuceni_pracownicy' pracownikow, ktorzy
beda przeniesieni
            INSERT INTO Przerzuceni_pracownicy (nr_hr, imie, nazwisko, pesel,
na_stanowisko, na_dzial, nowe_wynagrodzenie)
            VALUES (s_hr, s_imie, s_nazwisko, s_pesel, na_stanowisko, na_dzial,
nowe_wynagrodzenie);

            -- zaktualizowanie stanowiska przerzuconych pracownikow
```

```
            UPDATE pracownik SET stanowisko_id = na_stanowisko WHERE nr_hr =
s_hr;

            RAISE NOTICE 'pracownik nr HR %', s_hr || ' (' || s_imie || ' ' ||
s_nazwisko ||
            ') zostal przypisany do stanowiska ' || stanowisko_nazwa || ' ('
            || na_stanowisko || ') do dzialu ' || dzial_nazwa || ' (' || na_dzial
||
            ') z wynagrodzeniem ' || nowe_wynagrodzenie || ' PLN';

            IF s_wiek < 18 THEN
                RAISE WARNING 'UWAGA! Pracownik o numerze %', s_hr || ' nie jest
pelnoletni!';
            END IF;
        END LOOP;

        RAISE INFO 'przeniosiono %', ilosc_osob_bez_stanowiska || ' pracownikow';

        -- pobranie ilosci przedzialow wiekowych z widoku
        SELECT count INTO wiek_1_20 FROM przedzialy_wiekowe_praconikow WHERE
stanowisko_id = na_stanowisko AND przedzialy = 'w1_20';
        SELECT count INTO wiek_21_40 FROM przedzialy_wiekowe_praconikow WHERE
stanowisko_id = na_stanowisko AND przedzialy = 'w21_40';
        SELECT count INTO wiek_41_60 FROM przedzialy_wiekowe_praconikow WHERE
stanowisko_id = na_stanowisko AND przedzialy = 'w41_60';
        SELECT count INTO wiek_61_80 FROM przedzialy_wiekowe_praconikow WHERE
stanowisko_id = na_stanowisko AND przedzialy = 'w61_80';
        SELECT count INTO wiek_81_100 FROM przedzialy_wiekowe_praconikow WHERE
stanowisko_id = na_stanowisko AND przedzialy = 'w81_100';

        IF wiek_1_20 IS NULL THEN wiek_1_20 := 0; END IF;
        IF wiek_21_40 IS NULL THEN wiek_21_40 := 0; END IF;
        IF wiek_41_60 IS NULL THEN wiek_41_60 := 0; END IF;
        IF wiek_61_80 IS NULL THEN wiek_61_80 := 0; END IF;
        IF wiek_81_100 IS NULL THEN wiek_81_100 := 0; END IF;

        -- wypisanie przedzialow wiekowych
        RAISE INFO 'W przedziale wiekowym 1-20 jest %', wiek_1_20 || '
pracownikow';
        RAISE INFO 'W przedziale wiekowym 21-40 jest %', wiek_21_40 || '
pracownikow';
        RAISE INFO 'W przedziale wiekowym 41-60 jest %', wiek_41_60 || '
pracownikow';
        RAISE INFO 'W przedziale wiekowym 61-80 jest %', wiek_61_80 || '
pracownikow';
```

```
        RAISE INFO 'W przedziale wiekowym 81-100 jest %', wiek_81_100 || '
pracownikow';

        -- zaktualizowanie ilosci stanowisk w nowym dziale
        UPDATE dzial SET ilosc_stanowisk = ilosc_osob_bez_stanowiska WHERE id =
na_dzial;
    ELSE
        -- pobranie ID nowego dzialu
        SELECT id INTO na_dzial FROM dzial WHERE nazwa = dzial_nazwa;

        -- skasowanie utworzonego dzialu
        DELETE FROM dzial WHERE id = na_dzial;

        RAISE WARNING 'brak pracownikow bez stanowiska, skasowano utworzony
dzial';
    END IF;

    RETURN NEW;
END;
$$ LANGUAGE 'plpgsql';

CREATE TRIGGER przerzucenie_pracownikow
AFTER INSERT ON dzial
FOR EACH ROW EXECUTE PROCEDURE przerzuc_pracownikow(2200, "Stanowisko
tymczasowe");
```

```
CREATE TRIGGER

Query returned successfully in 76 msec.
```

4.2.3.2 example 1

```
INSERT INTO dzial (id, nazwa, ilosc_stanowisk) VALUES (7, 'Dzial tymczasowy',
NULL);
```

```
NOTICE:  pracownik nr HR 7 (Raegan Cummerata) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 7 nie jest pelnoletni!
NOTICE:  pracownik nr HR 8 (Adam Wisozk) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 8 nie jest pelnoletni!
NOTICE:  pracownik nr HR 9 (Eloisa Rippin) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 10 (Will Kemmer) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 11 (Cristopher Feil) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 12 (Davonte Denesik) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 20 (Dejah Swaniawski) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 21 (Victoria Collins) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 21 nie jest pelnoletni!
NOTICE:  pracownik nr HR 22 (Natalia Shields) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 23 (Roger Connelly) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 23 nie jest pelnoletni!
NOTICE:  pracownik nr HR 24 (Aryanna Wyman) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 24 nie jest pelnoletni!
NOTICE:  pracownik nr HR 25 (Cyril Kihn) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 26 (Dangelo Bauch) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 34 (Alexane Ferry) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 35 (Katharina Parker) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 36 (Wilbert Murray) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 36 nie jest pelnoletni!
NOTICE:  pracownik nr HR 37 (Tyrell Wuckert) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 38 (Percival Lemke) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 38 nie jest pelnoletni!
NOTICE:  pracownik nr HR 39 (Emanuel Fay) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 46 (Milan Morar) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING:  UWAGA! Pracownik o numerze 46 nie jest pelnoletni!
NOTICE:  pracownik nr HR 47 (Bette Shields) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 48 (Hermina Leannon) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 49 (Bret Douglas) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE:  pracownik nr HR 50 (Anne Bosco) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
INFO:  przeniosiono 24 pracownikow
INFO:  W przedziale wiekowym 1-20 jest 9 pracownikow
INFO:  W przedziale wiekowym 21-40 jest 11 pracownikow
INFO:  W przedziale wiekowym 41-60 jest 2 pracownikow
INFO:  W przedziale wiekowym 61-80 jest 0 pracownikow
INFO:  W przedziale wiekowym 81-100 jest 0 pracownikow
INSERT 0 1

Query returned successfully in 69 msec.
```


NOTICE: pracownik nr HR 7 (Raegan Cummerata) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING: UWAGA! Pracownik o numerze 7 nie jest pelnoletni!
NOTICE: pracownik nr HR 8 (Adam Wisozk) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING: UWAGA! Pracownik o numerze 8 nie jest pelnoletni!
NOTICE: pracownik nr HR 9 (Eloisa Rippin) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE: pracownik nr HR 10 (Will Kemmer) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE: pracownik nr HR 11 (Cristopher Feil) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE: pracownik nr HR 12 (Davonte Denesik) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE: pracownik nr HR 20 (Dejah Swaniawski) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE: pracownik nr HR 21 (Victoria Collins) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING: UWAGA! Pracownik o numerze 21 nie jest pelnoletni!
NOTICE: pracownik nr HR 22 (Natalia Shields) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
NOTICE: pracownik nr HR 23 (Roger Connelly) zostal przypisany do stanowiska Stanowisko tymczasowe (18) do dzialu Dzial tymczasowy (7) z wynagrodzeniem 2200.00 PLN
WARNING: UWAGA! Pracownik o numerze 23 nie jest pelnoletni!


```
INFO:  przeniosiono 24 pracownikow
INFO:  W przedziale wiekowym 1-20 jest 9 pracownikow
INFO:  W przedziale wiekowym 21-40 jest 11 pracownikow
INFO:  W przedziale wiekowym 41-60 jest 2 pracownikow
INFO:  W przedziale wiekowym 61-80 jest 0 pracownikow
INFO:  W przedziale wiekowym 81-100 jest 0 pracownikow
INSERT 0 1

Query returned successfully in 69 msec.
```

4.2.3.3 select 1

	id integer	nr_hr integer	imie character varying (25)	nazwisko character varying (50)	pesel bigint	na_stanowisko integer	na_dzial integer	nowe_wynagrodzenie numeric (10,2)
1	1	7	Raegan	Cummerata	13221703	18	7	2200.00
2	2	8	Adam	Wisozk	17241760	18	7	2200.00
3	3	9	Eloisa	Rippin	62232113	18	7	2200.00
4	4	10	Will	Kemmer	37223603	18	7	2200.00
5	5	11	Cristopher	Feil	30670581	18	7	2200.00
6	6	12	Davonte	Denesik	94749558	18	7	2200.00
7	7	20	Dejah	Swaniawski	15627672	18	7	2200.00
8	8	21	Victoria	Collins	32913452	18	7	2200.00
9	9	22	Natalia	Shields	45835720	18	7	2200.00
10	10	23	Roger	Connelly	71277935	18	7	2200.00
11	11	24	Aryanna	Wyman	19753433	18	7	2200.00
12	12	25	Cyril	Kihn	25222332	18	7	2200.00
13	13	26	Dangelo	Bauch	38281436	18	7	2200.00
14	14	34	Alexane	Ferry	66050278	18	7	2200.00

4.2.3.4 example 2

```
INSERT INTO dzial (id, nazwa, ilosc_stanowisk) VALUES (8, 'Dzial tymczasowy',
NULL);
```

4.2.3.5 select 2

```
WARNING:  brak pracownikow bez stanowiska, skasowano utworzony dzial
INSERT 0 1

Query returned successfully in 52 msec.
```

4.3 Reguły

4.3.1 historia zatrudnien

4.3.1.1 create

```
-- historia zatrudnien
CREATE TABLE historia_zatrudnien (
    id SERIAL PRIMARY KEY,
    data_aktualizacji DATE NOT NULL,
    nr_hr INTEGER NOT NULL,
    imie VARCHAR(25) NOT NULL,
    nazwisko VARCHAR(50) NOT NULL,
    pesel BIGINT NOT NULL,
    data_urodzenia DATE NOT NULL,
    stanowisko_id INTEGER,
    opis VARCHAR(25) NOT NULL
);
```

```
CREATE OR REPLACE RULE historia_zatrudnien_INSERT AS
ON INSERT TO pracownik
DO (
    INSERT INTO historia_zatrudnien (data_aktualizacji, nr_hr, imie, nazwisko,
pesel, data_urodzenia, stanowisko_id, opis)
    VALUES (CURRENT_TIMESTAMP, NEW.nr_hr, NEW.imie, NEW.nazwisko, NEW.pesel,
NEW.data_urodzenia, NEW.stanowisko_id, 'zatrudnienie');
);

CREATE OR REPLACE RULE historia_zatrudnien_DELETE AS
ON DELETE TO pracownik
DO (
    INSERT INTO historia_zatrudnien (data_aktualizacji, nr_hr, imie, nazwisko,
pesel, data_urodzenia, stanowisko_id, opis)
    VALUES (CURRENT_TIMESTAMP, OLD.nr_hr, OLD.imie, OLD.nazwisko, OLD.pesel,
OLD.data_urodzenia, OLD.stanowisko_id, 'wypowiedzenie');
);

CREATE OR REPLACE RULE historia_zatrudnien_UPDATE AS
ON UPDATE TO pracownik WHERE OLD.stanowisko_id <> NEW.stanowisko_id
DO (
    INSERT INTO historia_zatrudnien (data_aktualizacji, nr_hr, imie, nazwisko,
pesel, data_urodzenia, stanowisko_id, opis)
    VALUES (CURRENT_TIMESTAMP, OLD.nr_hr, OLD.imie, OLD.nazwisko, OLD.pesel,
OLD.data_urodzenia, NEW.stanowisko_id, 'zmiana stanowiska');
);
```

4.3.1.2 example 1

```
-- historia zatrudnien
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie1', 'Nazwisko1', '11111111121',
'2017-06-25', '111111121', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie2', 'Nazwisko2', '11111111122',
'2017-06-25', '111111122', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie3', 'Nazwisko3', '11111111123',
'2017-06-25', '111111123', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
```

```
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie4', 'Nazwisko4', '11111111124',
'2017-06-25', '111111124', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie5', 'Nazwisko5', '11111111125',
'2017-06-25', '111111125', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie6', 'Nazwisko6', '11111111126',
'2017-06-25', '111111126', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie7', 'Nazwisko7', '11111111127',
'2017-06-25', '111111127', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie8', 'Nazwisko8', '11111111128',
'2017-06-25', '111111128', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie9', 'Nazwisko9', '11111111129',
'2017-06-25', '111111129', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie10', 'Nazwisko10', '11111111130',
'2017-06-25', '111111130', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
```

```
INSERT 0 1

Query returned successfully in 68 msec.
```

4.3.1.3 select 1

```
SELECT * FROM historia_zatrudnien;
```

	id integer	data_aktualizacji date	nr_hr integer	imie character varying (25)	nazwisko character varying (50)	pesel bigint	data_urodzenia date	stanowisko_id integer	opis character varying (25)
1	1	2018-06-06	56	Imie1	Nazwisko1	11111121	2017-06-25	[null]	zatrudnienie
2	2	2018-06-06	58	Imie2	Nazwisko2	11111122	2017-06-25	[null]	zatrudnienie
3	3	2018-06-06	60	Imie3	Nazwisko3	11111123	2017-06-25	[null]	zatrudnienie
4	4	2018-06-06	62	Imie4	Nazwisko4	11111124	2017-06-25	[null]	zatrudnienie
5	5	2018-06-06	64	Imie5	Nazwisko5	11111125	2017-06-25	[null]	zatrudnienie
6	6	2018-06-06	66	Imie6	Nazwisko6	11111126	2017-06-25	[null]	zatrudnienie
7	7	2018-06-06	68	Imie7	Nazwisko7	11111127	2017-06-25	[null]	zatrudnienie
8	8	2018-06-06	70	Imie8	Nazwisko8	11111128	2017-06-25	[null]	zatrudnienie
9	9	2018-06-06	72	Imie9	Nazwisko9	11111129	2017-06-25	[null]	zatrudnienie
10	10	2018-06-06	74	Imie10	Nazwisko10	11111130	2017-06-25	[null]	zatrudnienie

4.3.1.4 example 2

```
DELETE FROM pracownik WHERE pesel = 11111111121;
DELETE FROM pracownik WHERE pesel = 11111111122;
DELETE FROM pracownik WHERE pesel = 11111111123;
DELETE FROM pracownik WHERE pesel = 11111111124;
DELETE FROM pracownik WHERE pesel = 11111111125;
```

```
DELETE 1

Query returned successfully in 47 msec.
```

4.3.1.5 select 2

```
SELECT * FROM historia_zatrudnien;
```

	id integer	data_aktualizacji date	nr_hr integer	imie character varying (25)	nazwisko character varying (50)	pesel bigint	data_urodzenia date	stanowisko_id integer	opis character varying (25)
1	1	2018-06-06	56	Imie1	Nazwisko1	11111121	2017-06-25	[null]	zatrudnienie
2	2	2018-06-06	58	Imie2	Nazwisko2	11111122	2017-06-25	[null]	zatrudnienie
3	3	2018-06-06	60	Imie3	Nazwisko3	11111123	2017-06-25	[null]	zatrudnienie
4	4	2018-06-06	62	Imie4	Nazwisko4	11111124	2017-06-25	[null]	zatrudnienie
5	5	2018-06-06	64	Imie5	Nazwisko5	11111125	2017-06-25	[null]	zatrudnienie
6	6	2018-06-06	66	Imie6	Nazwisko6	11111126	2017-06-25	[null]	zatrudnienie
7	7	2018-06-06	68	Imie7	Nazwisko7	11111127	2017-06-25	[null]	zatrudnienie
8	8	2018-06-06	70	Imie8	Nazwisko8	11111128	2017-06-25	[null]	zatrudnienie
9	9	2018-06-06	72	Imie9	Nazwisko9	11111129	2017-06-25	[null]	zatrudnienie
10	10	2018-06-06	74	Imie10	Nazwisko10	11111130	2017-06-25	[null]	zatrudnienie
11	11	2018-06-06	55	Imie1	Nazwisko1	11111121	2017-06-25	[null]	wypowiedzenie
12	12	2018-06-06	57	Imie2	Nazwisko2	11111122	2017-06-25	[null]	wypowiedzenie
13	13	2018-06-06	59	Imie3	Nazwisko3	11111123	2017-06-25	[null]	wypowiedzenie
14	14	2018-06-06	61	Imie4	Nazwisko4	11111124	2017-06-25	[null]	wypowiedzenie
15	15	2018-06-06	63	Imie5	Nazwisko5	11111125	2017-06-25	[null]	wypowiedzenie

4.3.1.6 example 3

```
UPDATE pracownik SET stanowisko_id = 2 WHERE pesel = 11111111126;
UPDATE pracownik SET stanowisko_id = 3 WHERE pesel = 11111111127;
UPDATE pracownik SET stanowisko_id = 4 WHERE pesel = 11111111128;
```

```
UPDATE 1

Query returned successfully in 46 msec.
```

4.3.1.7 select 3

Data Output | Explain | Messages | Query History

	id integer	data_aktualizacji date	nr_hr integer	imie character varying (25)	nazwisko character varying (50)	pesel bigint	data_urodzenia date	stanowisko_id integer	opis character varying (25)
8	8	2018-06-06	70	Imie8	Nazwisko8	11111128	2017-06-25	[null]	zatrudnienie
9	9	2018-06-06	72	Imie9	Nazwisko9	11111129	2017-06-25	[null]	zatrudnienie
10	10	2018-06-06	74	Imie10	Nazwisko10	11111130	2017-06-25	[null]	zatrudnienie
11	11	2018-06-06	55	Imie1	Nazwisko1	11111121	2017-06-25	[null]	wypowiedzenie
12	12	2018-06-06	57	Imie2	Nazwisko2	11111122	2017-06-25	[null]	wypowiedzenie
13	13	2018-06-06	59	Imie3	Nazwisko3	11111123	2017-06-25	[null]	wypowiedzenie
14	14	2018-06-06	61	Imie4	Nazwisko4	11111124	2017-06-25	[null]	wypowiedzenie
15	15	2018-06-06	63	Imie5	Nazwisko5	11111125	2017-06-25	[null]	wypowiedzenie
16	16	2018-06-06	65	Imie6	Nazwisko6	11111126	2017-06-25	2	zmiana stanowiska
17	17	2018-06-06	67	Imie7	Nazwisko7	11111127	2017-06-25	3	zmiana stanowiska
18	18	2018-06-06	69	Imie8	Nazwisko8	11111128	2017-06-25	4	zmiana stanowiska

4.3.2 logi

4.3.2.1 create

```sql
-- logi
CREATE TABLE logi (
    id SERIAL PRIMARY KEY,
    uzytkownik VARCHAR(25) NOT NULL,
    data_logu TIMESTAMP NOT NULL,
    tablica VARCHAR(50) NOT NULL,
    akcja VARCHAR(10) NOT NULL
);

-- pracownik
CREATE OR REPLACE RULE logi_pracownik_INSERT AS
ON INSERT TO pracownik
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'pracownik', 'INSERT');
);

CREATE OR REPLACE RULE logi_pracownik_DELETE AS
ON DELETE TO pracownik
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'pracownik', 'DELETE');
);

CREATE OR REPLACE RULE logi_pracownik_UPDATE AS
ON UPDATE TO pracownik
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'pracownik', 'UPDATE');
);
```

```sql
-- stanowisko
CREATE OR REPLACE RULE logi_stanowisko_INSERT AS
ON INSERT TO stanowisko
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'stanowisko', 'INSERT');
);

CREATE OR REPLACE RULE logi_stanowisko_DELETE AS
ON DELETE TO stanowisko
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'stanowisko', 'DELETE');
);

CREATE OR REPLACE RULE logi_stanowisko_UPDATE AS
ON UPDATE TO stanowisko
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'stanowisko', 'UPDATE');
);

-- dzial
CREATE OR REPLACE RULE logi_dzial_INSERT AS
ON INSERT TO dzial
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'dzial', 'INSERT');
);

CREATE OR REPLACE RULE logi_dzial_DELETE AS
ON DELETE TO dzial
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'dzial', 'DELETE');
);

CREATE OR REPLACE RULE logi_dzial_UPDATE AS
ON UPDATE TO dzial
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'dzial', 'UPDATE');
);
```

```sql
-- wymagania
CREATE OR REPLACE RULE logi_wymagania_INSERT AS
ON INSERT TO wymagania
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'wymagania', 'INSERT');
);

CREATE OR REPLACE RULE logi_wymagania_DELETE AS
ON DELETE TO wymagania
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'wymagania', 'DELETE');
);

CREATE OR REPLACE RULE logi_wymagania_UPDATE AS
ON UPDATE TO wymagania
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'wymagania', 'UPDATE');
);

-- kwalifikacja
CREATE OR REPLACE RULE logi_kwalifikacja_INSERT AS
ON INSERT TO kwalifikacja
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'kwalifikacja',
'INSERT');
);

CREATE OR REPLACE RULE logi_kwalifikacja_DELETE AS
ON DELETE TO kwalifikacja
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'kwalifikacja',
'DELETE');
);

CREATE OR REPLACE RULE logi_kwalifikacja_UPDATE AS
ON UPDATE TO kwalifikacja
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'kwalifikacja',
'UPDATE');
```

```
);

-- stanowisko_wymagania
CREATE OR REPLACE RULE logi_stanowisko_wymagania_INSERT AS
ON INSERT TO stanowisko_wymagania
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'stanowisko_wymagania',
'INSERT');
);

CREATE OR REPLACE RULE logi_stanowisko_wymagania_DELETE AS
ON DELETE TO stanowisko_wymagania
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'stanowisko_wymagania',
'DELETE');
);

CREATE OR REPLACE RULE logi_stanowisko_wymagania_UPDATE AS
ON UPDATE TO stanowisko_wymagania
DO (
    INSERT INTO logi(uzytkownik, data_logu, tablica, akcja)
    VALUES ((SELECT USER db_details), CURRENT_TIMESTAMP, 'stanowisko_wymagania',
'UPDATE');
);
```

```
CREATE RULE

Query returned successfully in 147 msec.
```

4.3.2.2 example

```
-- logi
INSERT INTO pracownik (imie, nazwisko, pesel, data_urodzenia, numer_telefonu,
miejscowosc, ulica, numer_domu, kod_pocztowy, wyksztalcenie, hobby,
kwalifikacja_id, stanowisko_id) VALUES ('Imie10', 'Nazwisko10', '11111111131',
'2017-06-25', '111111131', 'Sharonburgh', 'Jerry Creek', '73', '91-657',
'zasadnicze zawodowe', NULL, 3, NULL);
UPDATE pracownik SET stanowisko_id = 6 WHERE pesel = 11111111131;
DELETE FROM pracownik WHERE pesel = 11111111131;

INSERT INTO stanowisko (id, nazwa, dzial_id, minimalne_wyksztalcenie,
wynagrodzenie) VALUES (100, 'Stanowisko testowe do Reguly', 6, 'średnie', 3800);
UPDATE stanowisko SET wynagrodzenie = 1000 WHERE id = 100;
```

```sql
DELETE FROM stanowisko WHERE id = 100;

INSERT INTO dzial (id, nazwa, ilosc_stanowisk) VALUES (100, 'Dzial testowy do
reguly', 2);
UPDATE dzial SET ilosc_stanowisk = 10 WHERE id = 100;
DELETE FROM dzial WHERE id = 100;

INSERT INTO wymagania (id, nazwa, poziom) VALUES (100, 'Wymaganie testowe do
reguly', NULL);
UPDATE wymagania SET poziom = 1 WHERE id = 100;
DELETE FROM wymagania WHERE id = 100;

INSERT INTO kwalifikacja (id, nazwa, opis) VALUES (100, 'Kwalifikacja testowa do
reguly', 'Jakieś tam kwalifikacje');
UPDATE kwalifikacja SET opis = 'Nowy opis do testowej kwalifikacji' WHERE id =
100;
DELETE FROM kwalifikacja WHERE id = 100;

INSERT INTO stanowisko_wymagania (stanowisko_id, wymagania_id) VALUES (1, 1);
UPDATE stanowisko_wymagania SET wymagania_id = 2 WHERE stanowisko_id = 1 AND
wymagania_id = 1;
DELETE FROM stanowisko_wymagania WHERE stanowisko_id = 1 AND wymagania_id = 2;
```

```
DELETE 1

Query returned successfully in 206 msec.
```

4.3.2.3 select

```sql
SELECT * FROM logi ORDER BY id;
```

	id integer	uzytkownik character varying (25)	data_logu timestamp without time zone	tablica character varying (50)	akcja character varying (10)
31	31	postgres	2018-06-06 14:04:18.523909	pracownik	UPDATE
32	32	postgres	2018-06-06 14:04:18.523909	pracownik	UPDATE
33	33	postgres	2018-06-06 14:04:18.523909	pracownik	UPDATE
34	34	postgres	2018-06-06 14:04:18.523909	dzial	UPDATE
35	35	postgres	2018-06-06 14:04:18.523909	dzial	INSERT
36	36	postgres	2018-06-06 14:04:18.523909	dzial	UPDATE
37	37	postgres	2018-06-06 14:04:18.523909	dzial	DELETE
38	38	postgres	2018-06-06 14:04:18.523909	stanowisko	UPDATE
39	39	postgres	2018-06-06 14:04:18.523909	wymagania	INSERT
40	40	postgres	2018-06-06 14:04:18.523909	wymagania	UPDATE
41	41	postgres	2018-06-06 14:04:18.523909	wymagania	DELETE
42	42	postgres	2018-06-06 14:04:18.523909	kwalifikacja	INSERT
43	43	postgres	2018-06-06 14:04:18.523909	kwalifikacja	UPDATE
44	44	postgres	2018-06-06 14:04:18.523909	kwalifikacja	DELETE
45	45	postgres	2018-06-06 14:04:18.523909	stanowisko_wymagania	INSERT
46	46	postgres	2018-06-06 14:04:18.523909	stanowisko_wymagania	UPDATE
47	47	postgres	2018-06-06 14:04:18.523909	stanowisko_wymagania	DELETE

4.3.3 monitorowanie wynagrodzenia w firmie po dzialach, korzystajac z widoku

4.3.3.1 create

```
-- monitorowanie wynagrodzenia w firmie po dzialach, korzystajac z widoku
CREATE OR REPLACE VIEW statystyki_wynagrodzenia AS
SELECT d.id, d.nazwa, MAX(s.wynagrodzenie), AVG(s.wynagrodzenie),
MIN(s.wynagrodzenie)
FROM stanowisko AS s
RIGHT JOIN dzial AS d ON d.id = s.dzial_id
GROUP BY d.id, d.nazwa;

CREATE TABLE monitor_wynagrodzenia (
    id SERIAL PRIMARY KEY,
    data_logu DATE NOT NULL,
    dzial_id INTEGER NULL,
    maksymalne_wynagrodzenie DECIMAL(10, 2) NULL,
    srednie_wynagrodzenie DECIMAL(10, 2) NULL,
    minimalne_wynagrodzenie DECIMAL(10, 2) NULL
);

CREATE OR REPLACE RULE zmiana_wynagrodzenia AS
ON UPDATE TO stanowisko
DO (
    INSERT INTO monitor_wynagrodzenia (data_logu, dzial_id,
maksymalne_wynagrodzenie, srednie_wynagrodzenie, minimalne_wynagrodzenie)
```

```
    VALUES (CURRENT_TIMESTAMP, OLD.dzial_id,
        (SELECT COALESCE(x.max, 0) FROM statystyki_wynagrodzenia AS x WHERE x.id
= OLD.dzial_id),
        (SELECT COALESCE(ROUND(x.avg, 2), 0) FROM statystyki_wynagrodzenia AS x
WHERE x.id = OLD.dzial_id),
        (SELECT COALESCE(x.min, 0) FROM statystyki_wynagrodzenia AS x WHERE x.id
= OLD.dzial_id)
    );
);

CREATE OR REPLACE RULE sledz_zmiany_wynagrodzenia AS
ON UPDATE TO stanowisko
DO (
    SELECT * FROM monitor_wynagrodzenia AS x
    WHERE x.dzial_id = (SELECT id FROM dzial WHERE id = NEW.dzial_id);
);
```

```
CREATE RULE

Query returned successfully in 69 msec.
```

4.3.3.2 example

```
-- monitorowanie wynagrodzenia w firmie po dzialach, korzystajac z widoku
UPDATE stanowisko SET wynagrodzenie = 1000 WHERE id = 1;
UPDATE stanowisko SET wynagrodzenie = 2000 WHERE id = 2;
UPDATE stanowisko SET wynagrodzenie = 3000 WHERE id = 3;
UPDATE stanowisko SET wynagrodzenie = 1500 WHERE id = 4;
UPDATE stanowisko SET wynagrodzenie = 1800 WHERE id = 5;
UPDATE stanowisko SET wynagrodzenie = 3500 WHERE id = 6;
UPDATE stanowisko SET wynagrodzenie = 4200 WHERE id = 7;
```

	id integer	data_logu date	dzial_id integer	maksymalne_wynagrodzenie numeric (10,2)	srednie_wynagrodzenie numeric (10,2)	minimalne_wynagrodzenie numeric (10,2)
1	1	2018-06-06	5	4120.00	3580.00	2200.00
2	2	2018-06-06	6	3800.00	3506.67	3220.00
3	3	2018-06-06	5	4120.00	3280.00	1000.00
4	4	2018-06-06	2	2600.00	2066.67	1800.00
5	5	2018-06-06	3	4000.00	3325.00	2650.00
6	6	2018-06-06	4	5000.00	3550.00	2100.00
7	7	2018-06-06	1	4290.00	2463.33	1200.00