Raport 2

Mateusz Tereba 280556

Rozgrzewka Oracle

Zadanie 1

```
SELECT k1.imie
FROM kocury k1
         LEFT JOIN kocury k2
                   ON k1.szef = k2.pseudo
         LEFT JOIN wrogowie_kocurow w1
                   ON k1.pseudo = w1.pseudo
WHERE k1.w_stadku_od < k2.w_stadku_od
   OR w1.pseudo IS NULL;
```

#	imie
1	MICKA
2	PUCEK
3	ZUZIA
4	LUCEK

Zadanie 2

```
SELECT k1.pseudo, w1.imie_wroga, w1.opis_incydentu
FROM kocury k1
         LEFT JOIN wrogowie_kocurow w1
                   ON k1.pseudo = w1.pseudo
WHERE k1.plec = 'D'
  AND w1.imie_wroga IS NOT NULL;
```

#	pseudo	imie_wroga	opis_incydentu
1	DAMA	KAZIO	CHCIAL OBEDRZEC ZE SKORY
2	KURKA	BUREK	POGONIL
3	LASKA	KAZIO	ZLAPAL ZA OGON I ZROBIL WIATRAK
4	LASKA	DZIKI BILL	POGRYZL ZE LEDWO SIE WYLIZALA
5	MALA	CHYTRUSEK	ZALECAL SIE
6	PUSZYSTA	SMUKLA	OBRZUCILA SZYSZKAMI
7	SZYBKA	GLUPIA ZOSKA	UZYLA KOTA JAKO SCIERKI
8	UCHO	SWAWOLNY DYZIO	OBRZUCIL KAMIENIAMI

Zadanie 3

Złączenie z szefami pozwala znaleźć koty, których szefowie są z innych band niż oni sami

```
SELECT k1.pseudo, k1.nr_bandy
FROM kocury k1
         INNER JOIN kocury k2 ON k1.szef = k2.pseudo
WHERE k1.nr_bandy != k2.nr_bandy
AND k1.szef = 'TYGRYS'
;
```

#	pseudo	nr_bandy
1	ZOMBI	3
2	LYSY	2
3	RAFA	4

Zadanie 4

Złączenie kocurów z kocurami, wybranie tylko kotów płci męskiej. jeśli przełożony lub podwładny jest null wyświetla odpowiedni komunikat przy pomocy coalesce

```
SELECT COALESCE(k1.pseudo, 'Brak przelozonego') "Przelozony",
       COALESCE(k2.pseudo, 'Brak podwladnego')  "Podwladny"
FROM kocury k1
         FULL JOIN kocury k2 ON k1.pseudo = k2.szef
WHERE COALESCE(k1.plec, 'M') = 'M'
  AND COALESCE(k2.plec, 'M') = 'M'
ORDER BY "Przelozony";
```

#	Przelozony	Podwladny
1	BOLEK	Brak podwladnego
2	Brak przelozonego	TYGRYS
3	LYSY	PLACEK
4	LYSY	RURA
5	MALY	Brak podwladnego
6	MAN	Brak podwladnego
7	PLACEK	Brak podwladnego
8	RAFA	MAN
9	RAFA	MALY
10	RURA	Brak podwladnego
11	TYGRYS	LYSY
12	TYGRYS	RAFA
13	TYGRYS	BOLEK
14	TYGRYS	ZOMBI
15	ZERO	Brak podwladnego

Zadanie 5

Dla każdego kota obliczamy sumę myszy w jego bandzie w podzapytaniu, a następnie obliczamy procent dla wybranych kotów

```
SELECT DISTINCT k1.pseudo,
                k1.przydzial_myszy,
                k1."suma_bandy",
                ROUND(k1.przydzial_myszy / k1."suma_bandy" * 100, 0) "procent"
FROM (SELECT k2.*, SUM(k2.przydzial_myszy) OVER (PARTITION BY k2.nr_bandy) "suma_bandy" FROM kocury k2) k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
         LEFT JOIN wrogowie_kocurow wk1 ON k1.pseudo = wk1.pseudo
         LEFT JOIN wrogowie w1 ON wk1.imie_wroga = w1.imie_wroga
WHERE b1.teren IN ('POLE', 'CALOSC')
  AND w1.stopien_wrogosci > 5
;
```

#	pseudo	przydzial_myszy	suma_bandy	procent
1	TYGRYS	103	200	52
2	BOLEK	50	200	25
3	RURA	56	284	20
4	LASKA	24	284	8

Zadanie 6

Złączenie poziome dwóch zapytań, w których wybrani zostali prominenci i szaracy

```
SELECT pseudo, przydzial_myszy, nr_bandy, 'Prominent'
FROM kocury k1
WHERE przydzial_myszy > (SELECT AVG(przydzial_myszy) FROM kocury)
UNION
SELECT pseudo, przydzial_myszy, nr_bandy, 'Szarak'
```

```
FROM kocury k2
WHERE przydzial_myszy = (SELECT MIN(przydzial_myszy) FROM kocury WHERE k2.nr_bandy = nr_bandy)
;
```

#	pseudo	przydzial_myszy	nr_bandy	'PROMINENT'
1	PLACEK	67	2	Prominent
2	RURA	56	2	Prominent
3	TYGRYS	103	1	Prominent
4	ZOMBI	75	3	Prominent
5	LYSY	72	2	Prominent
6	SZYBKA	65	2	Prominent
7	RAFA	65	4	Prominent
8	KURKA	61	3	Prominent
9	PUSZYSTA	20	3	Szarak
10	UCHO	40	4	Szarak
11	MALY	40	4	Szarak
12	MALA	22	1	Szarak
13	LASKA	24	2	Szarak

Zadanie 7

Dla każdego kota płci męskiej w podzapytaniu obliczono średni przydział z kotów z jego bandy

```
SELECT pseudo, (SELECT AVG(przydzial_myszy) "Srednia" FROM kocury WHERE k1.nr_bandy = nr_bandy) "Srednia"
FROM kocury k1
WHERE k1.plec = 'M'
ORDER BY "Srednia";
```

#	pseudo	Srednia
1	MALY	49.4
2	RAFA	49.4
3	MAN	49.4
4	ZERO	49.75
5	ZOMBI	49.75
6	TYGRYS	50
7	BOLEK	50
8	RURA	56.8
9	PLACEK	56.8
10	LYSY	56.8

Zadanie 8a

W podzapytaniu obliczane są dla każdej bandy średni przydział i całkowity średni przydział w bandzie i dla wszystkich kotów (bandy są wybierane na podstawie całkowitego przydziału natomiast, wyświetlany jest zwykły przydział — tak jak jest w przykładzie)

```
SELECT s1.nr_bandy, s1.srednia_bandy
FROM (SELECT DISTINCT nr_bandy,
                      AVG(przydzial_myszy) OVER (PARTITION BY nr_bandy)                    srednia_bandy,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0)) OVER () c_srednia,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0))
                          OVER (PARTITION BY nr_bandy)                                     c_srednia_bandy
      FROM kocury) s1
WHERE s1.c_srednia_bandy > (s1.c_srednia);
```

#	nr_bandy	srednia_bandy
1	1	50
2	2	56.8

Zadanie 8b

Obliczony i wyświetlony jest również średni przydział

```
SELECT s1.nr_bandy, s1.srednia_bandy, s1.srednia
FROM (SELECT DISTINCT nr_bandy,
                      AVG(przydzial_myszy) OVER ()                                        srednia,
                      AVG(przydzial_myszy) OVER (PARTITION BY nr_bandy)                   srednia_bandy,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0)) OVER () c_srednia,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0))
                          OVER (PARTITION BY nr_bandy)                                    c_srednia_bandy
      FROM kocury) s1
WHERE s1.c_srednia_bandy > (s1.c_srednia);
```

#	nr_bandy	srednia_bandy	srednia
1	1	50	51.666666666666666666666666666666666666667
2	2	56.8	51.666666666666666666666666666666666666667

Zadanie 9

Zliczenie tych samych miesięcy z dat przystąpienia do stadka

```
SELECT "Miesiac", COUNT("Miesiac")
FROM (SELECT TO_CHAR(w_stadku_od, 'MONTH') "Miesiac", EXTRACT(MONTH FROM w_stadku_od) "nr" FROM kocury)
GROUP BY "nr", "Miesiac"
ORDER BY "nr";
```

#	Miesiac	COUNT("MIESIAC")
1	JANUARY	3
2	FEBRUARY	1
3	MARCH	2
4	MAY	2
5	JULY	2
6	AUGUST	1
7	SEPTEMBER	2
8	OCTOBER	2
9	NOVEMBER	2
10	DECEMBER	1

Zadanie 10

Znalezienie całkowitego przydziału dla każdego kota następnie użycie tabeli przestawnej do zsumowania przydziałów z podziałem na funkcję i wybrane bandy

```
SELECT *
FROM (SELECT k1.funkcja, b1.nazwa, COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0) "myszy"
      FROM kocury k1
               LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy)
    PIVOT (
    SUM("myszy") FOR nazwa IN ('CZARNI RYCERZE', 'BIALI LOWCY')
    )
WHERE funkcja != 'SZEFUNIO'
ORDER BY funkcja;
```

#	funkcja	'CZARNI RYCERZE'	'BIALI LOWCY'
1	BANDZIOR	93	88
2	DZIELCZY	null	null
3	KOT	null	43
4	LAPACZ	56	null
5	LOWCZY	132	61

#	funkcja	'CZARNI RYCERZE'	'BIALI LOWCY'
6	MILUSIA	52	55

Zadanie 11

W tabeli przestawnej dodano podział na płeć

```
SELECT *
FROM (SELECT k1.funkcja, k1.plec, b1.nazwa, COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0) "myszy"
      FROM kocury k1
               LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy) PIVOT (
    SUM("myszy") FOR nazwa IN ('CZARNI RYCERZE', 'BIALI LOWCY')
    )
ORDER BY funkcja;
```

#	funkcja	plec	'CZARNI RYCERZE'	'BIALI LOWCY'
1	BANDZIOR	M	93	88
2	DZIELCZY	M	null	null
3	KOT	D	null	null
4	KOT	M	null	43
5	LAPACZ	D	null	null
6	LAPACZ	M	56	null
7	LOWCZY	D	65	61
8	LOWCZY	M	67	null
9	MILUSIA	D	52	55
10	SZEFUNIO	M	null	null

Rozgrzewka MSSQL

Zadanie 1

```
SELECT k1.imie
FROM kocury k1
         LEFT JOIN kocury k2
                   ON k1.szef = k2.pseudo
         LEFT JOIN wrogowie_kocurow w1
                   ON k1.pseudo = w1.pseudo
WHERE k1.w_stadku_od < k2.w_stadku_od
   OR w1.pseudo IS NULL;
```

#	imie
1	MICKA
2	PUCEK
3	ZUZIA
4	LUCEK

Zadanie 2

```
SELECT k1.pseudo, w1.imie_wroga, w1.opis_incydentu
FROM kocury k1
         LEFT JOIN wrogowie_kocurow w1
                   ON k1.pseudo = w1.pseudo
WHERE k1.plec = 'D'
  AND w1.imie_wroga IS NOT NULL;
```

#	pseudo	imie_wroga	opis_incydentu
1	DAMA	KAZIO	CHCIAL OBEDRZEC ZE SKORY
2	KURKA	BUREK	POGONIL
3	LASKA	DZIKI BILL	POGRYZL ZE LEDWO SIE WYLIZALA

#	pseudo	imie_wroga	opis_incydentu
4	LASKA	KAZIO	ZLAPAL ZA OGON I ZROBIL WIATRAK
5	MALA	CHYTRUSEK	ZALECAL SIE
6	PUSZYSTA	SMUKLA	OBRZUCILA SZYSZKAMI
7	SZYBKA	GLUPIA ZOSKA	UZYLA KOTA JAKO SCIERKI
8	UCHO	SWAWOLNY DYZIO	OBRZUCIL KAMIENIAMI

Zadanie 3

Złączenie z szefami pozwala znaleźć koty, których szefowie są z innych band niż oni sami

```
SELECT k1.pseudo, k1.nr_bandy
FROM kocury k1
         INNER JOIN kocury k2 ON k1.szef = k2.pseudo
WHERE k1.nr_bandy != k2.nr_bandy;
```

#	pseudo	nr_bandy
1	LYSY	2
2	RAFA	4
3	ZOMBI	3

Zadanie 4

Złączenie kocurów z kocurami, wybranie tylko kotów płci męskiej. jeśli przełożony lub podwłądny jest null wyświetla odpowiedni komunikat przy pomocy coalesce

```
SELECT COALESCE(k1.pseudo, 'Brak przelozonego') "Przelozony",
       COALESCE(k2.pseudo, 'Brak podwladnego')  "Podwladny"
FROM kocury k1
         FULL JOIN kocury k2 ON k1.pseudo = k2.szef
WHERE COALESCE(k1.plec, 'M') = 'M'
  AND COALESCE(k2.plec, 'M') = 'M'
ORDER BY "Przelozony";
```

#	Przelozony	Podwladny
1	BOLEK	Brak podwladnego
2	Brak przelozonego	TYGRYS
3	LYSY	PLACEK
4	LYSY	RURA
5	MALY	Brak podwladnego
6	MAN	Brak podwladnego
7	PLACEK	Brak podwladnego
8	RAFA	MALY
9	RAFA	MAN
10	RURA	Brak podwladnego
11	TYGRYS	BOLEK
12	TYGRYS	LYSY
13	TYGRYS	RAFA
14	TYGRYS	ZOMBI
15	ZERO	Brak podwladnego

Zadanie 5

Dla każdego kota obliczamy sumę myszy w jego bandzie w podzapytaniu, a następnie obliczamy procent dla wybranych kotów

```
SELECT DISTINCT k1.pseudo,
                k1.przydzial_myszy,
                k1."suma_bandy",
                ROUND(k1.przydzial_myszy / k1."suma_bandy" * 100, 0) "procent"
```

```
FROM (SELECT k2.*, SUM(k2.przydzial_myszy) OVER (PARTITION BY k2.nr_bandy) "suma_bandy" FROM kocury k2) k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
         LEFT JOIN wrogowie_kocurow wk1 ON k1.pseudo = wk1.pseudo
         LEFT JOIN wrogowie w1 ON wk1.imie_wroga = w1.imie_wroga
WHERE b1.teren IN ('POLE', 'CALOSC')
  AND w1.stopien_wrogosci > 5
;
```

#	pseudo	przydzial_myszy	suma_bandy	procent
1	BOLEK	50	200	25.00000000000000000000000000000000
2	LASKA	24	284	8.00000000000000000000000000000000
3	RURA	56	284	20.00000000000000000000000000000000
4	TYGRYS	103	200	52.00000000000000000000000000000000

Zadanie 6

Złączenie poziome dwóch zapytań, w których wybrani zostali prominenci i szaracy

```
SELECT pseudo, przydzial_myszy, nr_bandy, 'Prominent'
FROM kocury k1
WHERE przydzial_myszy > (SELECT AVG(przydzial_myszy) FROM kocury)
UNION
SELECT pseudo, przydzial_myszy, nr_bandy, 'Szarak'
FROM kocury k2
WHERE przydzial_myszy = (SELECT MIN(przydzial_myszy) FROM kocury WHERE k2.nr_bandy = nr_bandy)
;
```

#	pseudo	przydzial_myszy	nr_bandy	
1	KURKA	61	3	Prominent
2	LYSY	72	2	Prominent
3	PLACEK	67	2	Prominent
4	RAFA	65	4	Prominent
5	RURA	56	2	Prominent
6	SZYBKA	65	2	Prominent
7	TYGRYS	103	1	Prominent
8	ZOMBI	75	3	Prominent
9	MALA	22	1	Szarak
10	LASKA	24	2	Szarak
11	PUSZYSTA	20	3	Szarak
12	UCHO	40	4	Szarak
13	MALY	40	4	Szarak

Zadanie 7

Dla każdego kota płci męskiej w podzapytaniu obliczono średni przydział z kotów z jego bandy

```
SELECT pseudo, (SELECT AVG(przydzial_myszy) "Srednia" FROM kocury WHERE k1.nr_bandy = nr_bandy) "Srednia"
FROM kocury k1
WHERE k1.plec = 'M'
ORDER BY "Srednia";
```

#	pseudo	Srednia
1	MALY	49.400000
2	MAN	49.400000
3	RAFA	49.400000
4	ZERO	49.750000
5	ZOMBI	49.750000
6	BOLEK	50.000000
7	TYGRYS	50.000000

#	pseudo	Srednia
8	LYSY	56.800000
9	PLACEK	56.800000
10	RURA	56.800000

Zadanie 8a

W podzapytaniu obliczane są dla każdej bandy średni przydział i całkowity średni przydział w bandzie i dla wszystkich kotów (bandy są wybierane na podstawie całkowitego przydziału natomiast, wyświetlany jest zwykły przydział — tak jak jest w przykładzie)

```
SELECT s1.nr_bandy, s1.srednia_bandy
FROM (SELECT DISTINCT nr_bandy,
                      AVG(przydzial_myszy) OVER (PARTITION BY nr_bandy)                    srednia_bandy,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0)) OVER () c_srednia,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0))
                          OVER (PARTITION BY nr_bandy)                                     c_srednia_bandy
      FROM kocury) s1
WHERE s1.c_srednia_bandy > (s1.c_srednia);
```

#	nr_bandy	srednia_bandy
1	1	50.000000
2	2	56.800000

Zadanie 8b

Obliczony i wyświetlony jest również średni przydział

```
SELECT s1.nr_bandy, s1.srednia_bandy, s1.srednia
FROM (SELECT DISTINCT nr_bandy,
                      AVG(przydzial_myszy) OVER ()                                         srednia,
                      AVG(przydzial_myszy) OVER (PARTITION BY nr_bandy)                    srednia_bandy,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0)) OVER () c_srednia,
                      AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0))
                          OVER (PARTITION BY nr_bandy)                                     c_srednia_bandy
      FROM kocury) s1
WHERE s1.c_srednia_bandy > (s1.c_srednia);
```

#	nr_bandy	srednia_bandy	srednia
1	1	50.000000	51.666666
2	2	56.800000	51.666666

Zadanie 9

Zliczenie tych samych miesięcy z dat przystąpienia do stadka

```
SELECT "Miesiac", COUNT("Miesiac")
FROM (SELECT DATENAME(MONTH, w_stadku_od) "Miesiac", MONTH(w_stadku_od) "nr" FROM kocury) s1
GROUP BY "nr", "Miesiac"
ORDER BY "nr";
```

#	Miesiac	
1	January	3
2	February	1
3	March	2
4	May	2
5	July	2
6	August	1
7	September	2
8	October	2
9	November	2

#	Miesiac	
10	December	1

Zadanie 10

Znalezienie całkowitego przydziału dla każdego kota następnie użycie tabeli przestawnej do zsumowania przydziałów z podziałem na funkcję i wybrane bandy

```
SELECT *
FROM (SELECT k1.funkcja, b1.nazwa, COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0) "myszy"
      FROM kocury k1
               LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy) s1
    PIVOT (
    SUM(s1."myszy") FOR nazwa IN ("CZARNI RYCERZE", "BIALI LOWCY")
    ) p1
WHERE p1.funkcja != 'SZEFUNIO'
ORDER BY p1.funkcja;
```

#	funkcja	CZARNI RYCERZE	BIALI LOWCY
1	BANDZIOR	93	88
2	DZIELCZY	null	null
3	KOT	null	43
4	LAPACZ	56	null
5	LOWCZY	132	61
6	MILUSIA	52	55

Zadanie 11

W tabeli przestawnej dodano podział na płeć

```
SELECT *
FROM (SELECT k1.funkcja, k1.plec, b1.nazwa, COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0) "myszy"
      FROM kocury k1
               LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy) s1 PIVOT (
    SUM("myszy") FOR nazwa IN ("CZARNI RYCERZE", "BIALI LOWCY")
    ) p1
ORDER BY p1.funkcja;
```

#	funkcja	plec	CZARNI RYCERZE	BIALI LOWCY
1	BANDZIOR	M	93	88
2	DZIELCZY	M	null	null
3	KOT	D	null	null
4	KOT	M	null	43
5	LAPACZ	D	null	null
6	LAPACZ	M	56	null
7	LOWCZY	D	65	61
8	LOWCZY	M	67	null
9	MILUSIA	D	52	55
10	SZEFUNIO	M	null	null

Zadania 1 Oracle

Zadanie 12

```
SELECT k1.pseudo, k1.przydzial_myszy, b1.nazwa
FROM kocury k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
WHERE b1.teren IN ('POLE', 'CALOSC')
  AND k1.przydzial_myszy > 50
ORDER BY k1.przydzial_myszy DESC;
```

#	pseudo	przydzial_myszy	nazwa
1	TYGRYS	103	SZEFOSTWO
2	LYSY	72	CZARNI RYCERZE
3	PLACEK	67	CZARNI RYCERZE
4	SZYBKA	65	CZARNI RYCERZE
5	RURA	56	CZARNI RYCERZE

Zadanie 13

```
SELECT k1.imie, k1.w_stadku_od
FROM kocury k1
         INNER JOIN kocury k2 ON k1.w_stadku_od < k2.w_stadku_od
WHERE k2.imie = 'JACEK'
ORDER BY k1.w_stadku_od DESC;
```

#	imie	w_stadku_od
1	MELA	2008-11-01
2	KSAWERY	2008-07-12
3	BELA	2008-02-01
4	PUNIA	2008-01-01
5	PUCEK	2006-10-15
6	RUDA	2006-09-17
7	BOLEK	2006-08-15
8	ZUZIA	2006-07-21
9	KOREK	2004-03-16
10	CHYTRY	2002-05-05
11	MRUCZEK	2002-01-01

Zadanie 14a

```
SELECT k1.imie, k1.funkcja, k2.imie "Szef 1", k3.imie "Szef 2", k4.imie "Szef 3"
FROM kocury k1
         LEFT JOIN kocury k2 ON k1.szef = k2.pseudo
         LEFT JOIN kocury k3 ON k2.szef = k3.pseudo
         LEFT JOIN kocury k4 ON k3.szef = k4.pseudo
WHERE k1.funkcja IN ('KOT', 'MILUSIA');
```

#	imie	funkcja	Szef 1	Szef 2	Szef 3
1	LUCEK	KOT	PUNIA	KOREK	MRUCZEK
2	SONIA	MILUSIA	KOREK	MRUCZEK	null
3	BELA	MILUSIA	BOLEK	MRUCZEK	null
4	LATKA	KOT	PUCEK	MRUCZEK	null
5	DUDEK	KOT	PUCEK	MRUCZEK	null
6	MICKA	MILUSIA	MRUCZEK	null	null
7	RUDA	MILUSIA	MRUCZEK	null	null

Zadanie 14b

```
SELECT "imie", k2.funkcja, "Szef 1", "Szef 2", "Szef 3"
FROM (SELECT *
      FROM (SELECT CONNECT_BY_ROOT imie "imie", level "lvl", imie
            FROM kocury k1
            START WITH k1.funkcja IN ('KOT', 'MILUSIA')
            CONNECT BY PRIOR k1.szef = k1.pseudo)
          PIVOT (
          MAX(imie) FOR "lvl" IN (2 "Szef 1", 3 "Szef 2", 4 "Szef 3")
          )) 
```

```
        LEFT JOIN kocury k2 ON "imie" = k2.imie
;
```

#	imie	funkcja	Szef 1	Szef 2	Szef 3
1	MICKA	MILUSIA	MRUCZEK	null	null
2	LUCEK	KOT	PUNIA	KOREK	MRUCZEK
3	SONIA	MILUSIA	KOREK	MRUCZEK	null
4	LATKA	KOT	PUCEK	MRUCZEK	null
5	DUDEK	KOT	PUCEK	MRUCZEK	null
6	RUDA	MILUSIA	MRUCZEK	null	null
7	BELA	MILUSIA	BOLEK	MRUCZEK	null

Zadanie 14c

Dla każdego kota z funkcją kot lub milusia znajduję najdłuższą ścieżkę zaczynającą się od szefa wybranego kota

```
SELECT k1.imie, k1.funkcja, MAX("szefowie")
FROM kocury k1
         LEFT JOIN LATERAL (SELECT CONNECT_BY_ROOT k1.imie                                "imie",
                                   SYS_CONNECT_BY_PATH(RPAD(imie, 10, ' '), '|') || '|' "szefowie"
                           FROM kocury k2
                           START WITH k2.pseudo = k1.szef
                           CONNECT BY PRIOR k2.szef = k2.pseudo
    ) s1 ON k1.imie = s1."imie"
WHERE k1.funkcja IN ('KOT', 'MILUSIA')
GROUP BY k1.imie, k1.funkcja
;
```

	IMIE	FUNKCJA	MAX("SZEFOWIE")
1	RUDA	MILUSIA	|MRUCZEK |
2	SONIA	MILUSIA	|KOREK |MRUCZEK |
3	BELA	MILUSIA	|BOLEK |MRUCZEK |
4	LUCEK	KOT	|PUNIA |KOREK |MRUCZEK |
5	LATKA	KOT	|PUCEK |MRUCZEK |
6	MICKA	MILUSIA	|MRUCZEK |
7	DUDEK	KOT	|PUCEK |MRUCZEK |

Zadanie 15

```
SELECT k1.imie, b1.nazwa, wk1.imie_wroga, w1.stopien_wrogosci, wk1.data_incydentu
FROM kocury k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
         LEFT JOIN wrogowie_kocurow wk1 ON k1.pseudo = wk1.pseudo
         LEFT JOIN wrogowie w1 ON wk1.imie_wroga = w1.imie_wroga
WHERE k1.plec = 'D'
  AND wk1.data_incydentu > '2007-01-01'
ORDER BY k1.imie
;
```

#	imie	nazwa	imie_wroga	stopien_wrogosci	data_incydentu
1	BELA	CZARNI RYCERZE	KAZIO	10	2009-01-07
2	BELA	CZARNI RYCERZE	DZIKI BILL	10	2008-12-12
3	LATKA	LACIACI MYSLIWI	SWAWOLNY DYZIO	7	2011-07-14
4	MELA	LACIACI MYSLIWI	KAZIO	10	2009-02-07
5	PUNIA	BIALI LOWCY	BUREK	4	2010-12-14
6	RUDA	SZEFOSTWO	CHYTRUSEK	5	2007-03-07
7	SONIA	BIALI LOWCY	SMUKLA	1	2010-11-19

Zadanie 16

```
SELECT b1.nazwa, COUNT(DISTINCT k1.pseudo) "Koty z wrogami"
FROM koty.bandy b1
         INNER JOIN kocury k1 ON b1.nr_bandy = k1.nr_bandy
         INNER JOIN wrogowie_kocurow wk1 ON k1.pseudo = wk1.pseudo
GROUP BY b1.nazwa
;
```

#	nazwa	Koty z wrogami
1	SZEFOSTWO	3
2	LACIACI MYSLIWI	4
3	BIALI LOWCY	3
4	CZARNI RYCERZE	5

Zadanie 17

```
SELECT *
FROM (SELECT k1.funkcja, k1.pseudo, COUNT(*) "Liczba wrogow"
      FROM kocury k1
               INNER JOIN wrogowie_kocurow wk1 ON k1.pseudo = wk1.pseudo
      GROUP BY k1.pseudo, k1.funkcja)
WHERE "Liczba wrogow" > 1
;
```

#	funkcja	pseudo	Liczba wrogow
1	DZIELCZY	BOLEK	2
2	MILUSIA	LASKA	2
3	SZEFUNIO	TYGRYS	2

Zadanie 18

```
SELECT imie, "Dawka roczna", DECODE(SIGN("Dawka roczna" - 864), 1, 'POWYRZEJ 864', 0, '864', -1, 'Ponizej 864') "Dawka"
FROM (SELECT k1.imie, 12 * (COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0)) "Dawka roczna"
      FROM kocury k1)
;
```

#	imie	Dawka roczna	Dawka
1	JACEK	804	Ponizej 864
2	BARI	672	Ponizej 864
3	MICKA	864	864
4	LUCEK	516	Ponizej 864
5	SONIA	660	Ponizej 864
6	LATKA	480	Ponizej 864
7	DUDEK	480	Ponizej 864
8	MRUCZEK	1632	POWYRZEJ 864
9	CHYTRY	600	Ponizej 864
10	KOREK	1056	POWYRZEJ 864
11	BOLEK	1116	POWYRZEJ 864
12	ZUZIA	780	Ponizej 864
13	RUDA	768	Ponizej 864
14	PUCEK	780	Ponizej 864
15	PUNIA	732	Ponizej 864
16	BELA	624	Ponizej 864
17	KSAWERY	612	Ponizej 864
18	MELA	612	Ponizej 864

Zadanie 19a

```
SELECT b1.nr_bandy, b1.nazwa
FROM bandy b1
         LEFT JOIN kocury k1 ON b1.nr_bandy = k1.nr_bandy
WHERE k1.pseudo IS NULL
;
```

#	nr_bandy	nazwa
1	5	ROCKERSI

Zadanie 20

```
SELECT k1.pseudo, k1.funkcja, k1.przydzial_myszy
FROM kocury k1
WHERE k1.przydzial_myszy >=
      (SELECT 3 * k2.przydzial_myszy
       FROM kocury k2
                LEFT JOIN bandy b1 ON k2.nr_bandy = b1.nr_bandy
       WHERE k2.funkcja = 'MILUSIA'
         AND (b1.teren = 'SAD' OR b1.teren = 'CALOSC')
       ORDER BY k2.przydzial_myszy DESC
           FETCH FIRST 1 ROW ONLY)
;
```

#	pseudo	funkcja	przydzial_myszy
1	TYGRYS	SZEFUNIO	103
2	ZOMBI	BANDZIOR	75

Zadanie 21

Obliczenie średniej dla każdej funkcji, następnie wybranie funkcji, która jest mniejsza lub równa od wszystkich i większa, lub równa od wszystkich

```
WITH spozycie AS (SELECT funkcja, AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0)) "avrg"
                  FROM kocury
                  WHERE funkcja != 'SZEFUNIO'
                  GROUP BY funkcja)
SELECT *
FROM spozycie
WHERE "avrg" >= ALL (SELECT "avrg" FROM spozycie)
   OR "avrg" <= ALL (SELECT "avrg" FROM spozycie)
;
```

#	funkcja	avrg
1	KOT	41
2	BANDZIOR	90.5

Zadanie 22a

W podzapytaniu zliczana jest ilość unikalnych wierszy, następnie wybierane są tylko te spełniające warunek
define i accept pozwalają definować zmienną i przyjmować dane od użytkownika

```
DEFINE numer =  0
ACCEPT numer PROMPT 'Podaj liczbę kotów:'
SELECT pseudo, "suma1"
FROM (SELECT pseudo, COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma1"
      FROM kocury
      ORDER BY "suma1" DESC),
     LATERAL (
              SELECT COUNT(DISTINCT "suma2") "no"
              FROM (SELECT COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma2"
                    FROM kocury)
              WHERE "suma2" >= "suma1"
         )
WHERE "no" <= &numer
;
```

All rows fetched: 7 in 0.059 seconds

	PSEUDO	suma1
1	TYGRYS	136
2	LYSY	93
3	ZOMBI	88
4	LOLA	72
5	PLACEK	67
6	RAFA	65
7	SZYBKA	65

Zadanie 22b

Rownum jest evaluowany przed order by natomiast where jest ewaluowany po wiec potrzebne są 3 select-y, żeby znaleźć wartość w 6 rzędzie

```
DEFINE numer =  0
ACCEPT numer PROMPT 'Podaj liczbę kotów:'
SELECT pseudo, "suma1"
FROM (SELECT pseudo, COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma1"
      FROM kocury
      ORDER BY "suma1" DESC)
WHERE "suma1" >= (SELECT "suma2"
                  FROM (SELECT "suma2", ROWNUM "row"
                        FROM (SELECT DISTINCT COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma2"
                              FROM kocury
                              ORDER BY "suma2" DESC))
                  WHERE "row" = &numer)
;
```

All rows fetched: 7 in 0.049 seconds

	PSEUDO	suma1
1	TYGRYS	136
2	LYSY	93
3	ZOMBI	88
4	LOLA	72
5	PLACEK	67
6	RAFA	65
7	SZYBKA	65

Zadanie 22c

Dla każdego kota zliczam liczbę kotów ze spożyciem większym od danego kota, których spożycie jest unikalne (nie zajmuje tego samego miejsca) uzyskując przez to jego rangę

```
DEFINE numer =  0
ACCEPT numer PROMPT 'Podaj liczbę kotów:'
SELECT k1.pseudo, COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0) "spozycie"
FROM kocury k1
         LEFT JOIN LATERAL (
    SELECT k1.pseudo, COUNT(DISTINCT COALESCE(k2.przydzial_myszy, 0) + COALESCE(k2.myszy_extra, 0)) "rank"
    FROM kocury k2
    WHERE COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0) <
          COALESCE(k2.przydzial_myszy, 0) + COALESCE(k2.myszy_extra, 0)
    GROUP BY k1.pseudo
    ) s1 ON k1.pseudo = s1.pseudo
WHERE COALESCE(s1."rank" + 1, 1) <= &numer
;
```

All rows fetched: 7 in 0.048 seconds

	PSEUDO	spozycie
1	PLACEK	67
2	LOLA	72
3	TYGRYS	136
4	ZOMBI	88
5	LYSY	93
6	SZYBKA	65
7	RAFA	65

Zadanie 22d

Rank nadaje rekordom rangę (taką samą w przypadku remisów)

```
DEFINE numer =  0
ACCEPT numer PROMPT 'Podaj liczbę kotów:'
SELECT pseudo, "zjada"
FROM (SELECT k1.pseudo,
             COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0)                              "zjada",
             RANK() OVER (ORDER BY COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0) DESC) "rank"
      FROM kocury k1)
WHERE "rank" <= &numer
;
```

All rows fetched: 7 in 0.078 seconds

	PSEUDO	zjada
1	TYGRYS	136
2	LYSY	93
3	ZOMBI	88
4	LOLA	72
5	PLACEK	67
6	SZYBKA	65
7	RAFA	65

Zadanie 23

Zliczam wystąpienia następnie szereguje rekordy według odległości od średniej w podziale na 2 partycję mniejszą i większą od średniej

```
(SELECT "rok", "wstapienia"
 FROM (SELECT "rok",
              "wstapienia",
              RANK() OVER (PARTITION BY CASE
                                            WHEN "wstapienia" < "avrg" THEN 1
                                            WHEN "wstapienia" > "avrg" THEN 2
                                            ELSE 0
                  END ORDER BY ABS("wstapienia" - "avrg")
                  ) "group_rank"
       FROM (SELECT TO_CHAR(EXTRACT(YEAR FROM w_stadku_od)) "rok",
                    COUNT(*)                                "wstapienia",
                    AVG(COUNT(*)) OVER ()                   "avrg"
             FROM kocury
             GROUP BY EXTRACT(YEAR FROM w_stadku_od)))
 WHERE "group_rank" = 1)
UNION
(SELECT 'Srednia' "rok", AVG(COUNT(*)) OVER () "wstapienia"
 FROM kocury
 GROUP BY EXTRACT(YEAR FROM w_stadku_od)
     FETCH FIRST 1 ROW ONLY)
ORDER BY "wstapienia"
;
```

#	rok	wstapienia
1	2002	2
2	2009	2
3	2010	2
4	2011	2
5	Srednia	2.571428571428571428571428571428571428571
6	2006	4

Zadanie 24a

W CTE obliczenie średniego przydziału dla bandy, następnie złączenie kotów z bandami, w których średni przydział jest większy od przedziału kota

```
WITH przydzialy AS (SELECT nr_bandy,
                           AVG(COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0)) "avrg"
                    FROM kocury k1
                    GROUP BY nr_bandy)
SELECT k2.imie,
       COALESCE(k2.przydzial_myszy, 0) + COALESCE(k2.myszy_extra, 0) "zjada",
       k2.nr_bandy,
       przydzialy."avrg"
FROM przydzialy
         INNER JOIN kocury k2 ON k2.nr_bandy = przydzialy.nr_bandy AND
                                 COALESCE(k2.przydzial_myszy, 0) + COALESCE(k2.myszy_extra, 0) <= przydzialy."avrg"
WHERE k2.plec = 'M'
;
```

#	imie	zjada	nr_bandy	avrg
1	BARI	56	2	66.6
2	LUCEK	43	3	61.75
3	DUDEK	40	4	49.4
4	CHYTRY	50	1	80.5

Zadanie 24b

W podzapytaniu obliczam sumę i średnie sumy w bandach następnie wybieram tylko kocury z przydziałem mniejszym niż średnia

```
SELECT s1.imie, s1."sum", s1.nr_bandy, s1."avrg"
FROM (SELECT k1.*,
             COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0)                                    "sum",
             AVG(COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0)) OVER (PARTITION BY k1.nr_bandy) "avrg"
      FROM kocury k1) s1
         INNER JOIN kocury k2 ON s1.pseudo = k2.pseudo AND
                                 COALESCE(k2.przydzial_myszy, 0) + COALESCE(k2.myszy_extra, 0) < s1."avrg"
WHERE s1.plec = 'M'
;
```

#	imie	sum	nr_bandy	avrg
1	CHYTRY	50	1	80.5
2	DUDEK	40	4	49.4
3	BARI	56	2	66.6
4	LUCEK	43	3	61.75

Zadanie 24c

Podzapytanie w where jest niepotrzebne, bo można zrobić "sum" < "avrg" ale tak każe polecenie

```
SELECT s1.imie, s1."sum", s1.nr_bandy, s1."avrg"
FROM (SELECT k1.*,
             COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0)                                    "sum",
             AVG(COALESCE(k1.przydzial_myszy, 0) + COALESCE(k1.myszy_extra, 0)) OVER (PARTITION BY k1.nr_bandy) "avrg"
      FROM kocury k1) s1
WHERE s1.plec = 'M'
```

```
  AND "sum" <
      (SELECT AVG(COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0)) FROM kocury WHERE nr_bandy = s1.nr_bandy)
;
```

#	imie	sum	nr_bandy	avrg
1	CHYTRY	50	1	80.5
2	BARI	56	2	66.6
3	LUCEK	43	3	61.75
4	DUDEK	40	4	49.4

Zadanie 25

W cte znajduje najmniejszy i największy staż w danej bandzie, po czym wybieram koty z tymi stażami, należy użyć operatorów zbiorowych dlatego nie użyłem case

```
WITH stats AS (SELECT k1.imie,
                      b1.nazwa,
                      k1.w_stadku_od,
                      SYSDATE - k1.w_stadku_od                                          "staz",
                      MIN(SYSDATE - k1.w_stadku_od) OVER (PARTITION BY k1.nr_bandy) "mini",
                      MAX(SYSDATE - k1.w_stadku_od) OVER (PARTITION BY k1.nr_bandy) "maxi"
               FROM kocury k1
                        LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy)
SELECT s1.imie, s1.w_stadku_od, '<--- NAJMLODSZY STAZEM W BANDZIE' || s1.nazwa " "
FROM stats s1
WHERE "staz" = "mini"
UNION
SELECT s1.imie, s1.w_stadku_od, '<--- NAJSTARSZY STAZEM W BANDZIE' || s1.nazwa " "
FROM stats s1
WHERE "staz" = "maxi"
UNION
SELECT s1.imie, s1.w_stadku_od, '' " "
FROM stats s1
WHERE "staz" != "mini"
  AND "staz" != "maxi"
;
```

#	imie	w_stadku_od	
1	MICKA	2009-10-14	<--- NAJMLODSZY STAZEM W BANDZIESZEFOSTWO
2	BARI	2009-09-01	<--- NAJMLODSZY STAZEM W BANDZIECZARNI RYCERZE
3	SONIA	2010-11-18	<--- NAJMLODSZY STAZEM W BANDZIEBIALI LOWCY
4	DUDEK	2011-05-15	<--- NAJMLODSZY STAZEM W BANDZIELACIACI MYSLIWI
5	MRUCZEK	2002-01-01	<--- NAJSTARSZY STAZEM W BANDZIESZEFOSTWO
6	ZUZIA	2006-07-21	<--- NAJSTARSZY STAZEM W BANDZIECZARNI RYCERZE
7	KOREK	2004-03-16	<--- NAJSTARSZY STAZEM W BANDZIEBIALI LOWCY
8	PUCEK	2006-10-15	<--- NAJSTARSZY STAZEM W BANDZIELACIACI MYSLIWI
9	RUDA	2006-09-17	null
10	CHYTRY	2002-05-05	null
11	BOLEK	2006-08-15	null
12	JACEK	2008-12-01	null
13	BELA	2008-02-01	null
14	PUNIA	2008-01-01	null
15	LUCEK	2010-03-01	null
16	KSAWERY	2008-07-12	null
17	MELA	2008-11-01	null
18	LATKA	2011-01-01	null

Zadania 1 MSSQL

Zadanie 26

Stworzenie perspektywy

```
CREATE VIEW zadziorne_kotki AS
WITH kotki AS (SELECT *
               FROM kocury
               WHERE plec = 'D'),
     zadziorni AS (SELECT DISTINCT k1.pseudo
                   FROM kocury k1
                            LEFT JOIN wrogowie_kocurow wk1 ON k1.pseudo = wk1.pseudo
                            LEFT JOIN wrogowie w1 ON wk1.imie_wroga = w1.imie_wroga
                   WHERE w1.stopien_wrogosci > 5)
SELECT kotki.pseudo
FROM kotki
         INNER JOIN zadziorni ON kotki.pseudo = zadziorni.pseudo;
```

Użycie perspektywy

```
SELECT *
FROM zadziorne_kotki;
```

#	pseudo
1	DAMA
2	LASKA
3	UCHO

Zadanie 27

Użycie rekursywnego cte, aby przechodzić po drzewie

```
WITH hierarhia AS (SELECT 1 AS lvl, kocury.*
                   FROM kocury
                   WHERE funkcja = 'BANDZIOR'
                   UNION ALL
                   SELECT lvl + 1, kocury.*
                   FROM kocury
                            INNER JOIN hierarhia ON hierarhia.pseudo = kocury.szef)
SELECT lvl, pseudo, funkcja, nr_bandy
FROM hierarhia
WHERE plec = 'M'
ORDER BY lvl;
```

#	lvl	pseudo	funkcja	nr_bandy
1	1	LYSY	BANDZIOR	2
2	1	ZOMBI	BANDZIOR	3
3	2	PLACEK	LOWCZY	2
4	2	RURA	LAPACZ	2
5	3	ZERO	KOT	3

Zadanie 28

Ponowne użycie rekursywnego cte do rekursywnego przechodzenia drzewa, replicate pozwala na duplikowanie ciągów znaków n razy

```
WITH hierarhia AS (SELECT 0 AS lvl, kocury.*
                   FROM kocury
                   WHERE szef IS NULL
                   UNION ALL
                   SELECT lvl + 1, kocury.*
                   FROM kocury
                            INNER JOIN hierarhia ON hierarhia.pseudo = kocury.szef)
SELECT REPLICATE('===>', lvl) + CAST(lvl AS VARCHAR) + ' ' + imie "Hierarchia", COALESCE(szef, 'Sam sobie panem') "Pseudo
szefa", funkcja
FROM hierarhia
WHERE COALESCE(myszy_extra, 0) != 0
;
```

#	Hierarchia	Pseudo szefa	funkcja
1	0 MRUCZEK	Sam sobie panem	SZEFUNIO
2	===>1 MICKA	TYGRYS	MILUSIA
3	===>1 BOLEK	TYGRYS	BANDZIOR
4	===>1 RUDA	TYGRYS	MILUSIA
5	===>1 KOREK	TYGRYS	BANDZIOR
6	===>===>2 SONIA	ZOMBI	MILUSIA
7	===>===>2 BELA	LYSY	MILUSIA

Zadanie 29

Złączenie 3 tabel i wybranie kotów zgodnie z formułą

```
SELECT DISTINCT k1.pseudo, b1.nazwa
FROM kocury k1
         LEFT JOIN funkcje f1 ON k1.funkcja = f1.funkcja
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
         INNER JOIN wrogowie_kocurow wk1 ON k1.pseudo = wk1.pseudo
WHERE k1.pseudo NOT IN (SELECT szef FROM kocury WHERE szef IS NOT NULL)
  AND f1.min_myszy + (f1.max_myszy - f1.min_myszy) / 3 <= k1.przydzial_myszy
;
```

#	pseudo	nazwa
1	BOLEK	SZEFOSTWO
2	LASKA	CZARNI RYCERZE
3	MALA	SZEFOSTWO
4	PLACEK	CZARNI RYCERZE
5	RURA	CZARNI RYCERZE
6	SZYBKA	CZARNI RYCERZE

Zadania 1 Oracle

Zadanie 30

Utworzenie perspektywy, zliczenie ile kotów dostaje dodatkowe myszy, przy użyciu sum i case

```
CREATE VIEW przydzialy AS
SELECT b1.nazwa,
       AVG(przydzial_myszy)                                          "avrg",
       MIN(przydzial_myszy)                                          "mini",
       MAX(przydzial_myszy)                                          "maxi",
       COUNT(*)                                                      "suma",
       SUM(CASE WHEN COALESCE(myszy_extra, 0) != 0 THEN 1 ELSE 0 END) "suma_dod"
FROM kocury k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
GROUP BY b1.nazwa
;
```

Użycie perspektywy do wykonania zapytania z parametrem

```
DEFINE kot = ''
ACCEPT kot PROMPT 'Podaj pseudonim kota:'
SELECT pseudo,
       imie,
       funkcja,
       przydzial_myszy,
       'OD ' || TO_CHAR(p1."mini") || ' DO ' || TO_CHAR(p1."maxi") "granice",
       w_stadku_od
FROM kocury k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
         LEFT JOIN przydzialy p1 ON b1.nazwa = p1.nazwa
WHERE pseudo = '&kot'
;
```

All rows fetched: 1 in 0.064 seconds

	PSEUDO	IMIE	FUNKCJA	PRZYDZIAL_MYSZY	granice	W_STADKU_OD
1	PLACEK	JACEK	LOWCZY	67	OD 24 DO 72	01/12/2008 12:00:00

Zadanie 31

Zdefiniowanie perspektywy znajdującej koty z band czarni rycerze i łaciaci myśliwi o 3 najwyższych stażach

```
CREATE VIEW przydzialy_kotow AS
(
SELECT r1.pseudo,
       r1.plec,
       r1.przydzial_myszy,
       r1.myszy_extra,
       r1."przydzial_mini",
       r1."avrg_extra"
FROM (SELECT k1.*,
             RANK() OVER (PARTITION BY k1.nr_bandy ORDER BY SYSDATE - k1.w_stadku_od DESC) "staz_rank",
             MIN(przydzial_myszy) OVER ()                                                  "przydzial_mini",
             AVG(COALESCE(myszy_extra, 0)) OVER (PARTITION BY k1.nr_bandy)                 "avrg_extra"
      FROM kocury k1
               LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
      WHERE b1.nazwa = 'CZARNI RYCERZE'
         OR b1.nazwa = 'LACIACI MYSLIWI') r1
WHERE "staz_rank" <= 3)
;
```

Wyświetlenie przydziałów przed podwyżką

```
SELECT pseudo, plec, przydzial_myszy, COALESCE(myszy_extra, 0)
FROM przydzialy_kotow;
```

#	pseudo	plec	przydzial_myszy	COALESCE(MYSZY_EXTRA,0)
1	SZYBKA	D	65	0
2	LYSY	M	72	21
3	LASKA	D	24	28
4	RAFA	M	65	0
5	MAN	M	51	0
6	DAMA	D	51	0

Aktualizacja danych z perspektywy przy pomocy zadanych formuł, wszelkie zmiany w bazie nie są trwałe do polecenia COMMIT. Oracle nie pozwala modyfikować kolumn, które nie odwołują się bezpośrednio do tabeli

```
UPDATE kocury k1
SET przydzial_myszy = CASE
                          WHEN plec = 'D' THEN COALESCE(przydzial_myszy, 0) +
                                               (SELECT "przydzial_mini" FROM przydzialy_kotow WHERE k1.pseudo = pseudo) *
                                               0.1
                          WHEN plec = 'M' THEN COALESCE(przydzial_myszy, 0) + 10 END,
    myszy_extra     = COALESCE(myszy_extra, 0) +
                      (SELECT "avrg_extra" FROM przydzialy_kotow WHERE k1.pseudo = pseudo) * 0.15;
```

Wyświetlenie zaktualizowanych danych po podwyżce

```
SELECT pseudo, plec, przydzial_myszy, COALESCE(myszy_extra, 0)
FROM przydzialy_kotow;
```

#	pseudo	plec	przydzial_myszy	COALESCE(MYSZY_EXTRA,0)
1	SZYBKA	D	67	1
2	LYSY	M	82	22
3	LASKA	D	26	29
4	RAFA	M	75	0

#	pseudo	plec	przydzial_myszy	COALESCE(MYSZY_EXTRA,0)
5	MAN	M	61	0
6	DAMA	D	53	0

Cofnięcie update

```
ROLLBACK;
```

Zadanie 32a

Zdefiniowanie dwóch CTE całkowitego spożycia dla wszystkich kotów oraz przydziału według funkcji i płci z podziałem na bandę, a także całkowitej sumy w bandzie. Następnie złączenie z sumami przydziałów ze względu na funkcje.

```
WITH spozycie AS (SELECT funkcja, plec, nr_bandy, COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma"
                  FROM kocury),
     podzial AS (SELECT b1.nazwa,
                        CASE WHEN s1.plec = 'M' THEN 'Kocor' WHEN s1.plec = 'D' THEN 'Kotka' END "plec",
                        COUNT(*)                                                                "ile",
                        SUM(CASE WHEN s1.funkcja = 'SZEFUNIO' THEN s1."suma" ELSE 0 END)        "SZEFUNIO",
                        SUM(CASE WHEN s1.funkcja = 'BANDZIOR' THEN s1."suma" ELSE 0 END)        "BANDZIOR",
                        SUM(CASE WHEN s1.funkcja = 'LOWCZY' THEN s1."suma" ELSE 0 END)          "LOWCZY",
                        SUM(CASE WHEN s1.funkcja = 'LAPACZ' THEN s1."suma" ELSE 0 END)          "LAPACZ",
                        SUM(CASE WHEN s1.funkcja = 'KOT' THEN s1."suma" ELSE 0 END)             "KOT",
                        SUM(CASE WHEN s1.funkcja = 'MILUSIA' THEN s1."suma" ELSE 0 END)         "MILUSIA",
                        SUM(CASE WHEN s1.funkcja = 'DZIELCZY' THEN s1."suma" ELSE 0 END)        "DZIELCZY",
                        SUM(s1."suma")                                                          "SUMA"
                 FROM bandy b1
                          INNER JOIN spozycie s1 ON b1.nr_bandy = s1.nr_bandy
                 GROUP BY b1.nazwa, s1.plec)
SELECT *
FROM podzial p1
UNION
SELECT 'ZJADA RAZEM',
       '',
       SUM(p2."ile"),
       SUM(p2."SZEFUNIO"),
       SUM(p2."BANDZIOR"),
       SUM(p2."LOWCZY"),
       SUM(p2."LAPACZ"),
       SUM(p2."KOT"),
       SUM(p2."MILUSIA"),
       SUM(p2."DZIELCZY"),
       SUM(p2."SUMA")
FROM podzial p2
;
```

#	nazwa	plec	ile	szefunio	bandzior	lowczy	lapacz	kot	milusia	dzielczy	suma
1	SZEFOSTWO	Kocor	2	136	0	0	0	0	0	50	186
2	SZEFOSTWO	Kotka	2	0	0	0	0	0	136	0	136
3	CZARNI RYCERZE	Kocor	3	0	93	67	56	0	0	0	216
4	CZARNI RYCERZE	Kotka	2	0	0	65	0	0	52	0	117
5	BIALI LOWCY	Kotka	2	0	0	61	0	0	55	0	116
6	BIALI LOWCY	Kocor	2	0	88	0	0	43	0	0	131
7	LACIACI MYSLIWI	Kocor	3	0	0	65	51	40	0	0	156
8	LACIACI MYSLIWI	Kotka	2	0	0	0	51	40	0	0	91
9	ZJADA RAZEM	null	18	136	181	258	158	123	243	50	1149

Zadanie 32b

Znalezienie całkowitego spożycia dla kotów, zdefiniowanie tabeli przestawnej z tabeli spożycia i złączenie jej z suma w bandach z podziałem na płeć. Następnie złączamy tabelę przestawną z sumami z podziałem na funkcję

```
WITH spozycie AS (SELECT funkcja, plec, nazwa, COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma"
                  FROM kocury k1
                           INNER JOIN bandy b1 ON b1.nr_bandy = k1.nr_bandy),
```

```
    podzial AS (SELECT p1.nazwa,
                       p1.plec,
                       s2."ile",
                       COALESCE(p1."SZEFUNIO", 0) "SZEFUNIO",
                       COALESCE(p1."BANDZIOR", 0) "BANDZIOR",
                       COALESCE(p1."LOWCZY", 0)   "LOWCZY",
                       COALESCE(p1."LAPACZ", 0)   "LAPACZ",
                       COALESCE(p1."KOT", 0)      "KOT",
                       COALESCE(p1."MILUSIA", 0)  "MILUSIA",
                       COALESCE(p1."DZIELCZY", 0) "DZIELCZY",
                       s2."s"                     "SUMA"
                FROM spozycie s1 PIVOT (
                         SUM(s1."suma") FOR funkcja IN (
                        'SZEFUNIO' AS szefunio,
                        'BANDZIOR' AS bandzior,
                        'LOWCZY' AS lowczy,
                        'LAPACZ' AS lapacz,
                        'KOT' AS kot,
                        'MILUSIA' AS milusia,
                        'DZIELCZY' AS dzielczy
                        )
                         ) p1
                         LEFT JOIN (SELECT SUM("suma") "s", COUNT(*) "ile", nazwa, plec
                                    FROM spozycie
                                    GROUP BY nazwa, plec) s2
                                   ON p1.nazwa = s2.nazwa AND p1.plec = s2.plec)
SELECT *
FROM podzial
UNION
(SELECT 'ZJADA RAZEM',
        NULL,
        NULL,
        SUM(p1."SZEFUNIO"),
        SUM("BANDZIOR"),
        SUM("LOWCZY"),
        SUM("LAPACZ"),
        SUM("KOT"),
        SUM("MILUSIA"),
        SUM("DZIELCZY"),
        SUM("SUMA")
 FROM podzial p1)
;
```

#	nazwa	plec	ile	szefunio	bandzior	lowczy	lapacz	kot	milusia	dzielczy	suma
1	SZEFOSTWO	M	2	136	0	0	0	0	0	50	186
2	SZEFOSTWO	D	2	0	0	0	0	0	136	0	136
3	CZARNI RYCERZE	M	3	0	93	67	56	0	0	0	216
4	CZARNI RYCERZE	D	2	0	0	65	0	0	52	0	117
5	BIALI LOWCY	D	2	0	0	61	0	0	55	0	116
6	BIALI LOWCY	M	2	0	88	0	0	43	0	0	131
7	LACIACI MYSLIWI	M	3	0	0	65	51	40	0	0	156
8	LACIACI MYSLIWI	D	2	0	0	0	51	40	0	0	91
9	ZJADA RAZEM	null	null	136	181	258	158	123	243	50	1149

Zadania 2 MSSQL

Zadanie 30

Utworzenie widoku, zliczenie lie kotów dostaje dodatkowe myszy, przy użyciu sum i case

```
CREATE VIEW przydzialy AS
SELECT b1.nazwa,
       AVG(przydzial_myszy)                                          "avrg",
       MIN(przydzial_myszy)                                          "mini",
       MAX(przydzial_myszy)                                          "maxi",
       COUNT(*)                                                      "suma",
       SUM(CASE WHEN COALESCE(myszy_extra, 0) != 0 THEN 1 ELSE 0 END) "suma_dod"
FROM kocury k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
```

```
GROUP BY b1.nazwa
;
```

Użycie widoku do wykonania zapytania z parametrem (nie znalazłem sposobu na przyjęcie wartości bezpośrednio od użytkownika)

```
DECLARE @kot VARCHAR(15)
SET @kot = 'RURA'
SELECT pseudo,
       imie,
       funkcja,
       przydzial_myszy,
       'OD ' + CAST(p1."mini" AS VARCHAR) + ' DO ' + CAST(p1."maxi" AS VARCHAR) "granice",
       w_stadku_od
FROM kocury k1
         LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
         LEFT JOIN przydzialy p1 ON b1.nazwa = p1.nazwa
WHERE pseudo = @kot
;
```

#	pseudo	imie	funkcja	przydzial_myszy	granice	w_stadku_od
1	RURA	BARI	LAPACZ	56	OD 24 DO 72	2009-09-01

Zadanie 31

Zdefiniowanie perspektywy znajdującej koty z band czarni rycerze i łaciaci myśliwi o 3 najwyższych stażach

```
CREATE VIEW przydzialy_kotow AS
(
SELECT r1.pseudo,
       r1.plec,
       r1.przydzial_myszy,
       r1.myszy_extra,
       r1."przydzial_mini",
       r1."avrg_extra"
FROM (SELECT k1.*,
             RANK() OVER (PARTITION BY k1.nr_bandy ORDER BY DATEDIFF(DAY, k1.w_stadku_od, SYSDATETIME()) DESC) "staz_rank",
             MIN(przydzial_myszy) OVER ()
"przydzial_mini",
             AVG(COALESCE(myszy_extra, 0)) OVER (PARTITION BY k1.nr_bandy)                                    "avrg_extra"
      FROM kocury k1
               LEFT JOIN bandy b1 ON k1.nr_bandy = b1.nr_bandy
      WHERE b1.nazwa = 'CZARNI RYCERZE'
         OR b1.nazwa = 'LACIACI MYSLIWI') r1
WHERE "staz_rank" <= 3)
;
```

Wyświetlenie przydziałów przed podwyżką

```
SELECT pseudo, plec, przydzial_myszy, COALESCE(myszy_extra, 0) "myszy_ekstra"
FROM przydzialy_kotow;
```

#	pseudo	plec	przydzial_myszy	myszy_ekstra
1	SZYBKA	D	65	0
2	LYSY	M	72	21
3	LASKA	D	24	28
4	RAFA	M	65	0
5	MAN	M	51	0
6	DAMA	D	51	0

Rozpoczęcie transakcji

```
BEGIN TRANSACTION podwyzka;
```

Aktualizacja danych z perspektywy przy pomocy zadanych formuł

```
UPDATE przydzialy_kotow
SET przydzial_myszy = CASE
                          WHEN plec = 'D' THEN COALESCE(przydzial_myszy, 0) + przydzial_mini * 0.1
                          WHEN plec = 'M' THEN COALESCE(przydzial_myszy, 0) + 10 END,
    myszy_extra     = COALESCE(myszy_extra, 0) + avrg_extra * 0.15;
```

Wyświetlenie zaktualizowanych danych po podwyżce

```
SELECT pseudo, plec, przydzial_myszy, COALESCE(myszy_extra, 0) "myszy_ekstra"
FROM przydzialy_kotow;
```

#	pseudo	plec	przydzial_myszy	myszy_ekstra
1	SZYBKA	D	67	1
2	LYSY	M	82	22
3	LASKA	D	26	29
4	RAFA	M	75	0
5	MAN	M	61	0
6	DAMA	D	53	0

Cofnięcie transakcji

```
ROLLBACK TRANSACTION podwyzka;
```

Zadanie 32a

Zdefiniowanie dwóch cte całkowitego spożycia dla wszystkich kotów oraz przydziału według funkcji i płci z podziałem na bandę, a także całkowitej sumy w bandzie. następnie złączenie z sumami przydziałów ze względu na funkcje.

```
WITH spozycie AS (SELECT funkcja, plec, nr_bandy, COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma"
                  FROM kocury),
     podzial AS (SELECT b1.nazwa,
                        CASE WHEN s1.plec = 'M' THEN 'Kocor' WHEN s1.plec = 'D' THEN 'Kotka' END "plec",
                        COUNT(*)                                                                "ile",
                        SUM(CASE WHEN s1.funkcja = 'SZEFUNIO' THEN s1."suma" ELSE 0 END)        "SZEFUNIO",
                        SUM(CASE WHEN s1.funkcja = 'BANDZIOR' THEN s1."suma" ELSE 0 END)        "BANDZIOR",
                        SUM(CASE WHEN s1.funkcja = 'LOWCZY' THEN s1."suma" ELSE 0 END)          "LOWCZY",
                        SUM(CASE WHEN s1.funkcja = 'LAPACZ' THEN s1."suma" ELSE 0 END)          "LAPACZ",
                        SUM(CASE WHEN s1.funkcja = 'KOT' THEN s1."suma" ELSE 0 END)             "KOT",
                        SUM(CASE WHEN s1.funkcja = 'MILUSIA' THEN s1."suma" ELSE 0 END)         "MILUSIA",
                        SUM(CASE WHEN s1.funkcja = 'DZIELCZY' THEN s1."suma" ELSE 0 END)        "DZIELCZY",
                        SUM(s1."suma")                                                          "SUMA"
                 FROM bandy b1
                          INNER JOIN spozycie s1 ON b1.nr_bandy = s1.nr_bandy
                 GROUP BY b1.nazwa, s1.plec)
SELECT *
FROM podzial
UNION
SELECT 'ZJADA RAZEM',
       '',
       SUM(podzial.ile),
       SUM(podzial.szefunio),
       SUM(podzial.bandzior),
       SUM(podzial.lowczy),
       SUM(podzial.lapacz),
       SUM(podzial.kot),
       SUM(podzial.milusia),
       SUM(podzial.dzielczy),
       SUM(podzial.suma)
FROM podzial
;
```

#	nazwa	plec	ile	SZEFUNIO	BANDZIOR	LOWCZY	LAPACZ	KOT	MILUSIA	DZIELCZY	SUMA
1	BIALI LOWCY	Kocor	2	0	88	0	0	43	0	0	131
2	BIALI LOWCY	Kotka	2	0	0	61	0	0	55	0	116
3	CZARNI RYCERZE	Kocor	3	0	104	67	56	0	0	0	227

#	nazwa	plec	ile	SZEFUNIO	BANDZIOR	LOWCZY	LAPACZ	KOT	MILUSIA	DZIELCZY	SUMA
4	CZARNI RYCERZE	Kotka	2	0	0	68	0	0	55	0	123
5	LACIACI MYSLIWI	Kocor	3	0	0	75	61	40	0	0	176
6	LACIACI MYSLIWI	Kotka	2	0	0	0	53	40	0	0	93
7	SZEFOSTWO	Kocor	2	136	0	0	0	0	0	50	186
8	SZEFOSTWO	Kotka	2	0	0	0	0	0	136	0	136
9	ZJADA RAZEM		18	136	192	271	170	123	246	50	1188

Zadanie 32b

Znalezienie całkowitego spożycia dla kotów, zdefiniowanie tabeli przestawnej z tabeli spożycia i złączenie jej z suma w bandach z podziałem na płeć. następnie złączamy tabelę przestawną z sumami z podziałem na funkcję

```
WITH spozycie AS (SELECT funkcja, plec, nazwa, COALESCE(przydzial_myszy, 0) + COALESCE(myszy_extra, 0) "suma"
                  FROM kocury k1
                           INNER JOIN bandy b1 ON b1.nr_bandy = k1.nr_bandy),
     podzial AS (SELECT p1.nazwa,
                        p1.plec,
                        s2."ile",
                        COALESCE(p1."SZEFUNIO", 0) "SZEFUNIO",
                        COALESCE(p1."BANDZIOR", 0) "BANDZIOR",
                        COALESCE(p1."LOWCZY", 0)   "LOWCZY",
                        COALESCE(p1."LAPACZ", 0)   "LAPACZ",
                        COALESCE(p1."KOT", 0)      "KOT",
                        COALESCE(p1."MILUSIA", 0)  "MILUSIA",
                        COALESCE(p1."DZIELCZY", 0) "DZIELCZY",
                        s2."s"                     "SUMA"
                 FROM spozycie s1 PIVOT (
                          SUM(s1."suma") FOR funkcja IN (
                         "SZEFUNIO",
                         "BANDZIOR",
                         "LOWCZY",
                         "LAPACZ",
                         "KOT",
                         "MILUSIA",
                         "DZIELCZY"
                         )
                          ) p1
                          LEFT JOIN (SELECT SUM("suma") "s", COUNT(*) "ile", nazwa, plec
                                     FROM spozycie
                                     GROUP BY nazwa, plec) s2
                                    ON p1.nazwa = s2.nazwa AND p1.plec = s2.plec)
SELECT *
FROM podzial
UNION
(SELECT 'ZJADA RAZEM',
        NULL,
        NULL,
        SUM(p1."SZEFUNIO"),
        SUM("BANDZIOR"),
        SUM("LOWCZY"),
        SUM("LAPACZ"),
        SUM("KOT"),
        SUM("MILUSIA"),
        SUM("DZIELCZY"),
        SUM("SUMA")
 FROM podzial p1)
;
```

#	nazwa	plec	ile	SZEFUNIO	BANDZIOR	LOWCZY	LAPACZ	KOT	MILUSIA	DZIELCZY	SUMA
1	BIALI LOWCY	D	2	0	0	61	0	0	55	0	116
2	BIALI LOWCY	M	2	0	88	0	0	43	0	0	131
3	CZARNI RYCERZE	D	2	0	0	68	0	0	55	0	123
4	CZARNI RYCERZE	M	3	0	104	67	56	0	0	0	227
5	LACIACI MYSLIWI	D	2	0	0	0	53	40	0	0	93
6	LACIACI MYSLIWI	M	3	0	0	75	61	40	0	0	176

#	nazwa	plec	ile	SZEFUNIO	BANDZIOR	LOWCZY	LAPACZ	KOT	MILUSIA	DZIELCZY	SUMA
7	SZEFOSTWO	D	2	0	0	0	0	0	136	0	136
8	SZEFOSTWO	M	2	136	0	0	0	0	0	50	186
9	ZJADA RAZEM	null	null	136	192	271	170	123	246	50	1188

Podsumowanie

Różnice pomiędzy zapytaniami w Oracle SQL i T-SQL nie są znaczne, jednakże w T-SQL nie znalazłem możliwości wyświetlania okien dialogowych i przyjmowania danych bezpośrednio od użytkownika. Co według polecenia było wymagane w zadaniu 30. Najcięższe według mnie zadanie 23 ze względu na potrzebę podzielenia znalezionej liczby wystąpień na grupy mniejszą i większą od średniej, czego na początku nie wiedziałem jak zrobić. Zadanie 8 również spowodowało mi kłopot ponieważ wyświetlane dane są inne od tych według których wybierane są rekordy.