Funkcje rankingowe i agregacja na wielu poziomach

KLASYCZNE GRUPOWANIE

```
SELECT Nazwa, SUM(Brutto) AS RazemD
FROM Dzialy JOIN Osoby
ON Dzialy.Iddzialu=Osoby.Iddzialu
JOIN Zarobki
On Osoby.Idosoby=Zarobki.Idosoby
GROUP BY Nazwa
```

```
SELECT Nazwa, Nazwisko, SUM(Brutto) AS RazemO
FROM Dzialy JOIN Osoby
ON Dzialy.Iddzialu=Osoby.Iddzialu
JOIN Zarobki
On Osoby.Idosoby=Zarobki.Idosoby
GROUP BY Nazwa, Nazwisko
```

Zapytania wybierające – wyznaczające podsumowanie na dwóch poziomach

```
SELECT Nazwa, Nazwisko, SUM(Brutto)As RazemO, RazemD
 FROM Dzialy JOIN Osoby ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki ON Osoby.IdOsoby=Zarobki.IdOsoby
JOIN
(SELECT Osoby.IdDzialu, SUM(brutto) AS RazemD FROM Dzialy
JOIN Osoby ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki
ON Osoby.IdOsoby=Zarobki.IdOsoby
GROUP BY Osoby.IdDzialu) AS xxx
ON Osoby.IdDzialu=xxx.IdDzialu
GROUP BY Nazwa, Nazwisko, zarobki.IdOsoby, RazemD
```

Zapytania wybierające – wyznaczające podsumowanie na trzech poziomach

```
SELECT Nazwa, Nazwisko, SUM(Brutto)As RazemO, RazemD,
(SELECT SUM(Brutto) FROM Zarobki) AS RazemF
FROM Dzialy JOIN Osoby ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki ON Osoby.IdOsoby=Zarobki.IdOsoby
JOIN
(SELECT IdDzialu, SUM(brutto) AS RazemD FROM
Osoby JOIN Zarobki
ON Osoby.IdOsoby=Zarobki.IdOsoby
GROUP BY Osoby.IdDzialu) AS xxx
ON Osoby.IdDzialu=xxx.IdDzialu
GROUP BY Nazwa, Nazwisko, zarobki.IdOsoby, RazemD
```

Podsumowanie na wielu poziomach

```
SELECT Wojewodztwo, Miasto, NazwaProducenta, NazwaTowaru, SUM(Szt) AS IleT,
IleP, IleM
FROM Wojewodztwa JOIN Miasta
ON Wojewodztwa.IdWojewodztwa=Miasta.IdWojewodztwa
JOIN Producenci ON Miasta.IdMiasta=Producenci.IdMiasta
JOIN Towar ON Producenci.IdProducenta=Towar.IdProducenta
JOIN Transakcje ON Towar.IdTowaru=Transakcje.IdTowaru
JOIN
(SELECT IdProducenta, SUM(Szt) AS IleP
FROM Towar JOIN Transakcje ON Towar.IdTowaru=Transakcje.IdTowaru
GROUP BY IdProducenta ) AS xxxP
ON Producenci.IdProducenta=xxxP.IdProducenta
JOIN
(SELECT IdMiasta, SUM(Szt) AS IleM
FROM Producenci JOIN Towar ON Producenci.IdProducenta=Towar.IdProducenta
JOIN Transakcje ON Towar.IdTowaru=Transakcje.IdTowaru
GROUP BY IdMiasta ) AS xxxM
ON Miasta.IdMiasta=xxxM.IdMiasta
GROUP BY Wojewodztwo, Miasto, NazwaProducenta, NazwaTowaru, IleP, IleM
```

```
SELECT Wojewodztwo, Miasto, NazwaProducenta, NazwaTowaru, SUM(Szt) AS IleT,
IleP, IleM, IleW, (SELECT SUM(Szt) FROM Transakcje) AS IleF
FROM Wojewodztwa JOIN Miasta ON Wojewodztwa.IdWojewodztwa=Miasta.IdWojewodztwa
JOIN Producenci ON Miasta.IdMiasta=Producenci.IdMiasta
JOIN Towar ON Producenci.IdProducenta=Towar.IdProducenta
JOIN Transakcje ON Towar.IdTowaru=Transakcje.IdTowaru
JOIN      (SELECT IdProducenta, SUM(Szt) AS IleP
FROM Towar JOIN Transakcje ON Towar.IdTowaru=Transakcje.IdTowaru
GROUP BY IdProducenta
) AS xxxP  ON Producenci.IdProducenta=xxxP.IdProducenta
JOIN       (SELECT IdMiasta, SUM(Szt) AS IleM
FROM Producenci JOIN Towar ON Producenci.IdProducenta=Towar.IdProducenta
JOIN Transakcje ON Towar.IdTowaru=Transakcje.IdTowaru
GROUP BY IdMiasta
) AS xxxM  ON Miasta.IdMiasta=xxxM.IdMiasta
JOIN   (SELECT IdWojewodztwa, SUM(Szt) AS IleW
FROM Miasta JOIN Producenci ON Miasta.IdMiasta=Producenci.IdMiasta
JOIN Towar ON Producenci.IdProducenta=Towar.IdProducenta
JOIN Transakcje ON Towar.IdTowaru=Transakcje.IdTowaru
GROUP BY IdWojewodztwa
) AS xxxW   ON Wojewodztwa.IdWojewodztwa=xxxW.IdWojewodztwa
GROUP BY Wojewodztwo, Miasto, NazwaProducenta, NazwaTowaru,
IleP, IleM, IleW
```

Suma na wszystkich poziomach

WITH CUBE lub WITH ROLLUP

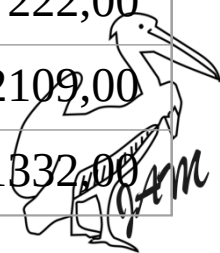
```
SELECT Nazwa, Nazwisko, SUM(Brutto)
FROM Dzialy JOIN Osoby
ON Dzialy.Iddzialu=Osoby.Iddzialu
JOIN Zarobki
On Osoby.Idosoby=Zarobki.Idosoby
GROUP BY Nazwa, Nazwisko
WITH ROLLUP
```

```
SELECT Nazwa, Nazwisko, SUM(Brutto)
FROM Dzialy JOIN Osoby
ON Dzialy.Iddzialu=Osoby.Iddzialu
JOIN Zarobki
On Osoby.Idosoby=Zarobki.Idosoby
GROUP BY Nazwa, Nazwisko
WITH CUBE
```

Grupowanie z klauzulami ROLLUP i CUBE

Administracja	Janik	555,00
Administracja	Nowak	1332,00
Administracja	NULL	1887,00
Dyrekcja	Kowalski	2109,00
Dyrekcja	NULL	2109,00
Techniczny	Adamczyk	777,00
Techniczny	Kow	222,00
Techniczny	NULL	999,00
NULL	NULL	4995,00

Administracja	Janik	555,00
Administracja	Nowak	1332,00
Administracja	NULL	1887,00
Dyrekcja	Kowalski	2109,00
Dyrekcja	NULL	2109,00
Techniczny	Adamczyk	777,00
Techniczny	Kow	222,00
Techniczny	NULL	999,00
NULL	NULL	4995,00
NULL	Adamczyk	777,00
NULL	Janik	555,00
NULL	Kow	222,00
NULL	Kowalski	2109,00
NULL	Nowak	1332,00



Nowa postać CUBE, ROLLUP

```
--ALTER DATABASE BazaRelacyjna SET COMPATIBILITY_LEVEL =
    100;

SELECT Nazwa, Nazwisko, SUM(Brutto)
FROM Dzialy JOIN Osoby
ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki ON Osoby.IdOsoby=Zarobki.IdOsoby
GROUP BY CUBE(Nazwa, Nazwisko)
-- GROUP BY ROLLUP(Nazwa, Nazwisko)
```

Grupowanie Grouping Sets

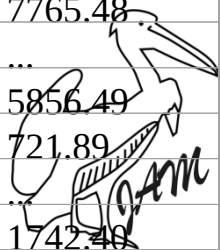
```
SELECT Nazwa, Nazwisko, SUM(Brutto)
FROM Dzialy JOIN Osoby
ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki ON Osoby.IdOsoby=Zarobki.IdOsoby
GROUP BY GROUPING SETS(Nazwa, Nazwisko)
```

NULL	Adamczyk	777,00
NULL	Janik	555,00
NULL	Kow	666,00
NULL	Kowalczyk	777,00
NULL	Kowalski	2109,00
NULL	Nowak	1332,00
NULL	Nowicki	2109,00
NULL	Zięba	333,00
Administracja	NULL	1887,00
Dyrekcja	NULL	2442,00
Handlowy	NULL	2109,00
Techniczny	NULL	2220,00

Grupowanie - Grouping Sets

```
SELECT NazwaTowaru, NazwaKategorii, NazwaProducenta, Miasto, Województwo,
SUM(Cena * szt) AS Wartosc FROM Transakcje INNER JOIN
Towar ON Transakcje.IdTowaru = Towar.IdTowaru INNER JOIN
Producenci ON Towar.IdProducenta = Producenci.IdProducenta INNER JOIN
Miasta ON Producenci.IdMiasta = Miasta.IdMiasta INNER JOIN
Wojewodztwa ON Miasta.IdWojewodztwa = Wojewodztwa.IdWojewodztwa INNER JOIN
Kategorie ON Towar.IdKategorii = Kategorie.IdKategorii
GROUP BY GROUPING SETS (NazwaTowaru, NazwaKategorii, NazwaProducenta, Miasto,
Województwo)
```

NazwaTowaru	NazwaKategorii	NazwaProducenta	Miasto	Województwo	Wartosc
NULL	NULL	NULL	NULL	łódzkie	61842.63
...	...	...	...	...	...
NULL	NULL	NULL	NULL	wielkopolskie	354.87
NULL	NULL	NULL	BIELSKO	NULL	16677.98
...	...	...	...	...	...
NULL	NULL	NULL	ZGIERZ	NULL	8691.34
NULL	NULL	ANTOFAGAS	NULL	NULL	7634.56
...	...	...	...	...	...
NULL	NULL	Zegarmistrz	NULL	NULL	4347.94
NULL	Akcesoria	NULL	NULL	NULL	7765.48
...	...	...	...	...	...
NULL	Zegary	NULL	NULL	NULL	5856.49
Akwarela	NULL	NULL	NULL	NULL	721.89
...	...	...	...	...	...
Zestaw Lux	NULL	NULL	NULL	NULL	1742.40



Funkcje OVER PARTITION

```
SELECT Idosoby, Brutto,
ROW_NUMBER() OVER (ORDER BY Brutto DESC)AS RowNumber
FROM Zarobki
```

```
SELECT Idosoby, Brutto,
ROW_NUMBER() OVER ( PARTITION BY idosoby ORDER BY Brutto DESC)AS
    RowNumber
FROM Zarobki
```

Idosoby	Brutto	RowNumber
1	999.00	1
6	999.00	2
11	888.00	3
2	888.00	4
5	777.00	5
8	777.00	6
6	666.00	7
1	666.00	8
...	...	...

Idosoby	Brutto	RowNumber
1	999.00	1
1	666.00	2
1	333.00	3
1	111.00	4
2	888.00	1
2	444.00	2
3	444.00	1
3	222.00	2
4	555.00	1
...	...	...

Ranking Functions (Transact-SQL)

Ustala numer wiersza według wzrostu w obrębie działu.

Dla równych wartości numer wiersza losowo

```
SELECT IdDzialu ,Wzrost,
ROW_NUMBER() OVER (PARTITION BY IdDzialu order by Wzrost DESC) AS NUMER_WIERSZA
FROM Osoby
ORDER BY Iddzialu
```

IdDzialu	Wzrost	NUMER_WIERSZA
1	1.82	1
1	1.67	2
1	1.67	3
1	NULL	4
2	1.75	1
2	1.72	2
2	1.68	3
3	1.78	1
3	1.72	2
5	1.73	1

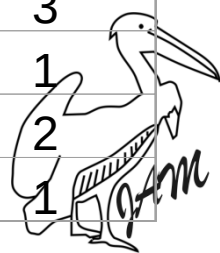
Ranking Functions (Transact-SQL)

Ustala ranking według wzrostu w obrębie działu.

Dla równych wartości taki sam ranking ale następny zwiększany o N

```
SELECT IdDzialu ,Wzrost,
RANK() OVER (PARTITION BY IdDzialu ORDER BY Wzrost DESC) AS RANK
FROM Osoby
ORDER BY Iddzialu
```

IdDzialu	Wzrost	RANK
1	1.82	1
1	1.67	2
1	1.67	2
1	NULL	4
2	1.75	1
2	1.72	2
2	1.68	3
3	1.78	1
3	1.72	2
5	1.73	1



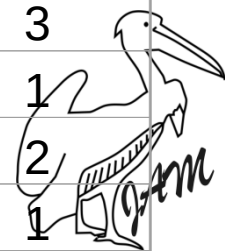
Ranking Functions (Transact-SQL)

Ustala ranking według wzrostu w obrębie działu.

Dla równych wartości taki sam ranking ale następny zwiększany o 1

```
SELECT IdDzialu ,Wzrost,
DENSE_RANK() OVER (PARTITION BY IdDzialu ORDER BY Wzrost DESC)
AS RANK
FROM Osoby
ORDER BY Iddzialu
```

IdDzialu	Wzrost	RANK
1	1.82	1
1	1.67	2
1	1.67	2
1	NULL	3
2	1.75	1
2	1.72	2
2	1.68	3
3	1.78	1
3	1.72	2
5	1.73	1



Ranking Functions (Transact-SQL)

Ustala numer wiersza według wzrostu w obrębie klastra (liczba klastrów dana parametrem) działu. Dla równych wartości numer wiersza losowo

```
SELECT IdDzialu ,Wzrost,
NTILE(2) OVER (PARTITION BY IdDzialu ORDER BY Wzrost DESC) AS NUMER_WIERSZA
FROM Osoby
ORDER BY Iddzialu
```

IdDzialu	Wzrost	NUMER_WIERSZA
1	1.82	1
1	1.67	1
1	1.67	2
1	NULL	2
2	1.75	1
2	1.72	1
2	1.68	2
3	1.78	1
3	1.72	2
5	1.73	1

Ranking Functions (Transact-SQL)

```
SELECT IdDzialu ,Wzrost,
ROW_NUMBER() OVER (ORDER BY Iddzialu,Wzrost DESC) AS
NUMER_WIERSZA,
ROW_NUMBER() OVER (PARTITION BY IdDzialu ORDER BY Iddzialu,
Wzrost DESC) AS NUMER_W_Dziale
FROM Osoby
WHERE IdDzialu IS NOT NULL
ORDER BY Iddzialu, Wzrost DESC
```

IdDzialu	Wzrost	NUMER_WIERSZA	NUMER_W_Dziale
1	1.69	1	1
1	1.67	2	2
2	1.72	3	1
2	1.68	4	2
3	1.87	5	1
3	1.79	6	2
3	1.78	7	3
3	1.72	8	4
4	1.93	9	1
4	1.85	10	2
5	0.00	11	1

Ranking Functions (Transact-SQL)

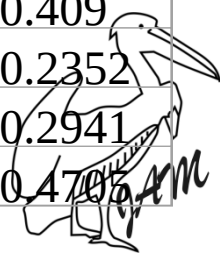
```
SELECT Nazwa, Nazwisko, Wzrost,
ROW_NUMBER() OVER(ORDER BY Wzrost DESC) AS Nr,
ROW_NUMBER() OVER(PARTITION BY Nazwa ORDER BY Wzrost DESC) AS Nr1,
RANK() OVER(PARTITION BY Nazwa ORDER BY Wzrost DESC) AS Nr2,
DENSE_RANK() OVER(PARTITION BY Nazwa ORDER BY Wzrost DESC) AS Nr3,
PERCENT_RANK() OVER(PARTITION BY Nazwa ORDER BY Wzrost DESC) AS Nr4,
NTILE(2) OVER(PARTITION BY Nazwa ORDER BY Wzrost DESC) AS Nr5

FROM Dzialy JOIN Osoby
ON Dzialy.IdDzialu=Osoby.IdDzialu
```

Funkcje agregujące nad partycją (Transact-SQL)

```
SELECT IdDzialu, Osoby.IdOsoby, Brutto,
SUM(Brutto) OVER () AS suma,
SUM(Brutto) OVER (PARTITION BY IdDzialu) AS suma_dzial,
Brutto / SUM(Brutto) OVER(PARTITION BY IdDzialu) AS udzial
FROM Osoby JOIN Zarobki
ON Osoby.IdOsoby=Zarobki.IdOsoby
```

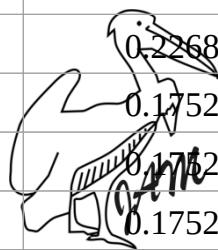
IdDzialu	IdOsoby	Brutto	suma	suma_dzial	udzial
NULL	15	555.00	10767.00	555.00	1.00
1	9	333.00	10767.00	2442.00	0.1363
1	1	666.00	10767.00	2442.00	0.2727
1	1	111.00	10767.00	2442.00	0.0454
1	1	333.00	10767.00	2442.00	0.1363
1	1	999.00	10767.00	2442.00	0.409
2	2	444.00	10767.00	1887.00	0.2352
2	4	555.00	10767.00	1887.00	0.2941
2	2	888.00	10767.00	1887.00	0.4705



Funkcje agregujące nad partycją (Transact-SQL)

```
SELECT IdDzialu, Osoby.IdOsoby, Brutto, SUM(Brutto) OVER () AS suma,
SUM(Brutto) OVER (PARTITION BY IdDzialu) AS suma_dzial,
Brutto / SUM(Brutto) OVER(PARTITION BY IdDzialu) AS udzialWyp_w_Dziale,
SUM(Brutto) OVER (PARTITION BY Osoby.IdOsoby)/
SUM(Brutto) OVER(PARTITION BY IdDzialu) AS udzialSUM_w_dziale,
Brutto / SUM(Brutto) OVER() AS udzialWyp_w_Firmie,
SUM(Brutto) OVER (PARTITION BY Osoby.IdOsoby)/SUM(Brutto) OVER() AS
udzialSUM_w_Firmie,
SUM(Brutto) OVER (PARTITION BY IdDzialu)/SUM(Brutto) OVER() AS
udzialDzialu_w_Firmie
FROM Osoby JOIN Zarobki ON Osoby.IdOsoby=Zarobki.IdOsoby  ORDER BY IdDzialu
```

dDzialu	IdOsoby	Brutto	suma	suma_dzial	UdzialWyp _Dziale	UdzialSUM w_dziale	UdzialWyp _Firmie	UdzialSUM _Firmie	UdzialDzialu Firmie
NULL	15	555.00	10767.00	555.00	1.00	1.00	0.0515	0.0515	0.0515
1	9	333.00	10767.00	2442.00	0.1363	0.1363	0.0309	0.0309	0.2268
1	1	666.00	10767.00	2442.00	0.2727	0.8636	0.0618	0.1958	0.2268
1	1	111.00	10767.00	2442.00	0.0454	0.8636	0.0103	0.1958	0.2268
1	1	333.00	10767.00	2442.00	0.1363	0.8636	0.0309	0.1958	0.2268
1	1	999.00	10767.00	2442.00	0.409	0.8636	0.0927	0.1958	0.2268
2	2	444.00	10767.00	1887.00	0.2352	0.7058	0.0412	0.1237	0.1752
2	2	888.00	10767.00	1887.00	0.4705	0.7058	0.0824	0.1237	0.1752
2	4	555.00	10767.00	1887.00	0.2941	0.2941	0.0515	0.0515	0.1752



Funkcje agregujące nad partycją (Transact-SQL)

Suma bieżąca

```
SELECT Nazwa, Nazwisko, Brutto,
SUM(Brutto) OVER(ORDER BY Brutto ASC) AS SumaF,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC) AS SumaD
FROM Dzialy Join Osoby
ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki
ON Osoby.IdOsoby=Zarobki.IdOsoby
```

Suma ze zmienną definicją końców okna

```
SELECT Nazwa, Nazwisko, Brutto,
SUM(Brutto) OVER(ORDER BY Brutto ASC) AS SumaF,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ) AS SumaD,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ROWS BETWEEN
2 PRECEDING   AND 2 FOLLOWING ) AS SumaD1
FROM Dzialy Join Osoby
ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki
ON Osoby.IdOsoby=Zarobki.IdOsoby
```

Funkcje agregujące nad partycją (Transact-SQL)

Suma bieżąca ze zmienną definicją końców okna

```
SELECT Nazwa, Nazwisko, Brutto,
SUM(Brutto) OVER(ORDER BY Brutto ASC) AS SumaF,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ) AS SumaD,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ROWS BETWEEN
UNBOUNDED PRECEDING  AND CURRENT ROW ) AS SumaD1
FROM Dzialy Join Osoby ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki ON Osoby.IdOsoby=Zarobki.IdOsoby
```

Suma ze zmienną definicją końców okna

```
SELECT Nazwa, Nazwisko, Brutto,
SUM(Brutto) OVER(ORDER BY Brutto ASC) AS SumaF,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ) AS SumaD,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ROWS BETWEEN
CURRENT ROW  AND UNBOUNDED FOLLOWING ) AS SumaD1

FROM Dzialy Join Osoby
ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN Zarobki
ON Osoby.IdOsoby=Zarobki.IdOsoby
```

Funkcje agregujące nad partycją (Transact-SQL)

Suma bieżąca ze zmienną definicją końców okna

```
SELECT Nazwa, Nazwisko, Brutto,
SUM(Brutto) OVER(ORDER BY Brutto ASC) AS SumaF,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ) AS SumaD,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC RANGE BETWEEN
UNBOUNDED PRECEDING  AND CURRENT ROW ) AS SumaD1
FROM Dzialy Join Osoby ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN ZarobkiON Osoby.IdOsoby=Zarobki.IdOsoby
```

```
SELECT Nazwa, Nazwisko, Brutto,
SUM(Brutto) OVER(ORDER BY Brutto ASC) AS SumaF,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC ) AS SumaD,
SUM(Brutto) OVER(PARTITION BY Nazwa ORDER BY Brutto ASC RANGE BETWEEN
CURRENT ROW  AND UNBOUNDED FOLLOWING ) AS SumaD1
FROM Dzialy Join Osoby ON Dzialy.IdDzialu=Osoby.IdDzialu
JOIN ZarobkiON Osoby.IdOsoby=Zarobki.IdOsoby
```

RANGE obsługuje tylko CURRENT oraz UNBOUNDED
Nie można podać wartości numerycznej przesunięcia

Zapytania wybierające - operacje na zbiorach

UNIA

Suma dwóch zapytań z eliminacją powtarzających się rekordów

```
SELECT Nazwisko, Imie FROM Osoby
UNION
SELECT Nazwisko, Imie FROM ttt;
```

Suma dwóch zapytań

```
SELECT Nazwisko, Imie FROM Osoby
UNION ALL
SELECT Nazwisko, Imie FROM ttt;
```

Zapytania wybierające - operacje na zbiorach

Część wspólna dwóch zapytań

```
SELECT Nazwisko, Imie FROM Osoby
INTERSECT
SELECT Nazwisko, Imie FROM ttt;
```

Różnica dwóch zapytań

```
SELECT Nazwisko, Imie FROM Osoby
EXCEPT
SELECT Nazwisko, Imie FROM ttt;
```

Następne zapytania dynamiczne

Dla hurtowni danych

podstawowa postać hurtowni