OBLIKOVANJE BAZA PODATAKA

Vježbe 12-13

GRUPIRANJE

- Prema ISO definiciji, grupiranje se može raditi ovako:

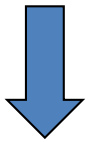
```
GROUP BY {
        column-expression
        | ROLLUP ( <group_by_expression> [ ,...n ] )
        | CUBE ( <group_by_expression> [ ,...n ] )
        | GROUPING SETS ( <grouping_set> [ ,...n ] )
        | () --calculates the grand total
} [ ,...n ]

<group_by_expression> ::=
        column-expression
        | ( column-expression [ ,...n ] )

<grouping_set> ::=
        () --calculates the grand total
        | <grouping_set_item>
        | ( <grouping_set_item> [ ,...n ] )

<grouping_set_item> ::=
        <group_by_expression>
        | ROLLUP ( <group_by_expression> [ ,...n ] )
        | CUBE ( <group_by_expression> [ ,...n ] )
```

Osnovno korištenje

- U osnovnom korištenju radimo grupiranje po jednom ili više stupaca (`column-expression`)
- Jedinstvene vrijednosti definiraju grupe
- U rezultatu će biti onoliko redaka koliko ima grupa

Country	Region	Sales
Canada	Alberta	100
Canada	British Columbia	200
Canada	British Columbia	300
United States	Montana	100

```
select
        Country,
        Region,
        SUM(sales) AS TotalSales
from Sales
group by Country, Region
```



Country	Region	TotalSales
Canada	Alberta	100
Canada	British Columbia	500
United States	Montana	100

GROUP BY ROLLUP postupak

- **GROUP BY ROLLUP primjenjuje sljedeći algoritam:**
  1. Napravi osnovno grupiranje
  2. Ukloni prvi stupac s desne strane iz liste grupiranja
     a. Napravi grupiranje prema preostalim stupcima
     b. Ako upravo nisi izračunao grand total, odi na korak 2

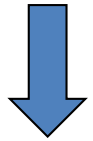
- **Algoritam staje nakon grupiranja po ničemu (grand total)**

- **U rezultatu, NULL vrijednost označava da po tom stupcu nismo grupirali**
  - Ako podaci također sadržavaju NULL vrijednosti, priča se komplicira

GROUP BY ROLLUP primjer

Country	Region	Sales
Canada	Alberta	100
Canada	British Columbia	200
Canada	British Columbia	300
United States	Montana	100

```
select
        Country,
        Region,
        SUM(Sales) AS TotalSales
from Sales
group by rollup (Country, Region)
```



	Country	Region	TotalSales
(Country, Region)	Canada	Alberta	100
	Canada	British Columbia	500
	United States	Montana	100
(Country)	Canada	NULL	600
	United States	NULL	100
()	NULL	NULL	700

GROUP BY CUBE postupak

- Kao i ROLLUP, samo primjenjuje grupiranje na sve moguće kombinacije stupaca
- Primjerice, ako napišemo GROUP BY CUBE (a, b, c), dobit ćemo jednake rezultate kao da smo napisali:
  - GROUP BY a, b, c
  - GROUP BY a, b
  - GROUP BY a, c
  - GROUP BY b, c
  - GROUP BY a
  - GROUP BY b
  - GROUP BY c
  - GROUP BY ()

GROUP BY CUBE primjer

Country	Region	Sales
Canada	Alberta	100
Canada	British Columbia	200
Canada	British Columbia	300
United States	Montana	100

```
select
        Country,
        Region,
        SUM(Sales) AS TotalSales
from Sales
group by cube (Country, Region)
```

	Country	Region	TotalSales
(Country, Region)	Canada	Alberta	100
	Canada	British Columbia	500
	United States	Montana	100
(Country)	Canada	NULL	600
	United States	NULL	100
(Region)	NULL	Alberta	100
	NULL	British Columbia	500
	NULL	Montana	100
()	NULL	NULL	700

GROUP BY GROUPING SETS ()

- Sami biramo koja sve grupiranja želimo napraviti

Country	Region	Sales
Canada	Alberta	100
Canada	British Columbia	200
Canada	British Columbia	300
United States	Montana	100

```
select
        Country, Region,
        SUM(Sales) AS TotalSales
from Sales
group by grouping sets
        (Country, Region, ())
```

	Country	Region	TotalSales
(Country)	Canada	NULL	600
(Country)	United States	NULL	100
(Region)	NULL	Alberta	100
(Region)	NULL	British Columbia	500
(Region)	NULL	Montana	100
()	NULL	NULL	700

Zadaci

1. Koristeći ROLLUP, dohvatite broj izdanih računa za kupce prezimena Adams i Simmons te broj izdanih računa za svaku jedinstvenu kombinaciju imena i prezimena tih kupaca. Koliko je ukupno računa izdano Adamsima?

2. Koristeći ROLLUP, dohvatite broj izdanih računa za kupce koji se zovu Rose i Lydia te broj izdanih računa za svaku jedinstvenu kombinaciju imena i prezimena tih kupaca. Koliko je ukupno računa izdano Lydiama?

3. Koristeći CUBE, dohvatite količinu izdanih računa za sve komercijaliste čije prezime započinje slovom A. Dodatno, neka rezultati uključuju i količinu po svakom imenu, svakom prezimenu te grand total.

Zadaci

4. Koristeći GROUPING SETS, dohvatite zaradu za svaki naziv proizvoda, za svaku boju proizvoda te grand total. Nedefinirane boje nemojte uključiti u rezultat.

5. Jednim upitom dohvatite sljedeće podatke: zaradu po boji proizvoda, te grand total. Nedefinirane boje nemojte uključiti u rezultat.

6. Jednim upitom dohvatite sljedeće podatke: zaradu po državi, po gradu u državi te grand total.

Window funkcija

- **Unutar OVER()-a definiramo kontekst na kojem djeluje funkcija, odnosno specifikaciju windowa**

- **Standard podržava sljedeće tipove window funkcija:**
  - Rangirajuće
  - Agregatne
  - Analitičke (distribucijske)

- **Window funkcije se najčešće koriste u SELECT dijelu upita**
  - Primjenjuju se nakon JOIN, WHERE i GROUP BY, ali prije ORDER BY

- **Korisno razmišljati ovako:**
  - Prvo pripremimo predfinalni skup stupaca i redaka
  - Napišemo window funkciju na tom skupu koja izračunava još jedan podatak za svaki od redaka

Anatomija window funkcije

```
naziv_funkcije(<parametri>)
OVER (
      [ <particioniranje> ]
      [ <redoslijed>
      [ <dodatni okvir> ] ]
 )
```

- Particioniranje: **`PARTITION BY`**
  - Dijeli retke u particije, funkcija se primjenjuje za svaku particiju neovisno o ostalim particijama
- Redoslijed: **`ORDER BY`**
  - Definira redoslijed redaka unutar svake particije
- Okvir: **`ROWS`** ili **`RANGE`**
  - Okvir je još jedan filter unutar windowa

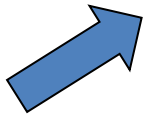
Funkcije rangiranja

- **Funkcije rangiranja su:**
  - **ROW_NUMBER()**
    - Vraća jedinstveni redni broj retka unutar particije, počevši od 1
  - **RANK()**
    - Vraća mjesto u rang listi retka unutar particije
    - Duplikati imaju jednako mjesto na rang listi
  - **DENSE_RANK()**
    - Kao RANK, samo što nakon duplikata nastavlja sa sljedećim brojem
  - **NTILE(n)**
    - Dijeli retke unutar particije u *n* skupina

- **ORDER BY je obavezan, PARTITION BY je opcionalan**

Primjer funkcija rangiranja

Country	Region	Sales
Canada	Alberta	100
Canada	British Columbia	200
Canada	British Columbia	300
United States	Montana	100

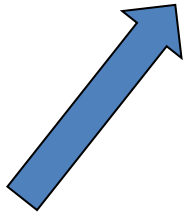


```
select
    Country,
    Region,
    Sales,
    ROW_NUMBER() over (order by Sales) as Rn,
    RANK() over (order by Sales) as R,
    DENSE_RANK() over (order by Sales) as Dr,
    NTILE(2) over (order by Sales) as N
from Sales
order by Country, Region
```

Country	Region	Sales	Rn	R	Dr	N
Canada	Alberta	100	1	1	1	1
Canada	British Columbia	200	3	3	2	2
Canada	British Columbia	300	4	4	3	2
United States	Montana	100	2	1	1	1

Primjer funkcija rangiranja s particioniranjem

Country	Region	Sales
Canada	Alberta	100
Canada	British Columbia	200
Canada	British Columbia	300
United States	Montana	100



Country	Region	Sales	Rn
Canada	Alberta	100	1
Canada	British Columbia	200	2
Canada	British Columbia	300	3
United States	Montana	100	1

```
select
    Country,
    Region,
    Sales,
    ROW_NUMBER() over (partition by Country order by Sales) as Rn
from Sales
```

Zadaci

1. Ispišite sve kupce poslagane prema prezimenu pa prema imenu. Kraj svakog kupca ispišite koji je njegov redni broj prema istom kriteriju.

2. Ispišite sve kupce poslagane prema gradu pa prema prezimenu pa prema imenu. Kraj svakog kupca ispišite koji je njegov redni broj prema prezimenu i imenu, ali za svaki grad posebno.

3. Ispišite sve stavke i uz svaku stavku ispišite njeno mjesto na rang listi. Stavka koja ima najveću količinu treba biti prva na rang listi, a ona s najmanjom količinom treba biti zadnja.

4. Ispišite sve račune za komercijalista 284. Podijelite te račune u skupine po 3 računa prema starosti, a zatim prema broju računa. Neka svi računi iz prve skupine imaju broj 1, iz sljedeće skupine neka imaju broj 2, itd.

Agregatne funkcije

- Jednake kao kod grupiranja, nama su zanimljive sljedeće:
  - AVG(col)
  - COUNT(col)
  - COUNT(*)
  - MAX(col)
  - MIN(col)
  - SUM(col)
- ORDER BY je opcionalan (tzv. *running total*), PARTITION BY je opcionalan
- Obično grupiranje gubi individualne retke; window agregatne funkcije ih ostavljaju

Primjer agregatnih funkcija

- Osnovno korištenje:

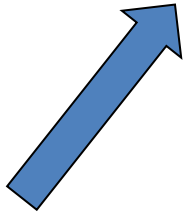
```
select *,
    SUM(s.Kolicina) over ()
from Racun as r inner join Stavka s on s.RacunID = r.IDRacun
order by r.DatumIzdavanja, r.IDRacun

select *,
    SUM(s.Kolicina) over (order by DatumIzdavanja)
…


select *,
    SUM(s.Kolicina) over (partition by year(DatumIzdavanja)
                          order by DatumIzdavanja)
…
```

- Ako koristimo ORDER BY, dobijemo tzv. *running total*
  - U obzir se uzimaju vrijednosti svih redaka koji imaju trenutnu vrijednost i svih onih prije njih, unutar iste particije
  - Prijelaz particije resetira brojač

Primjer *running totala*

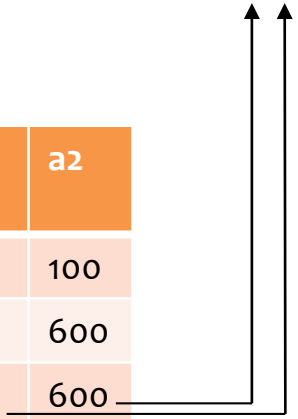
Country	Region	Sales
Canada	Alberta	100
Canada	British Columbia	200
Canada	British Columbia	300
United States	Montana	100

```
select
    Country,
    Region,
    Sales,
    AVG(Sales) over (order by Region) as a1,
    SUM(Sales) over (order by Region) as a2
from Sales
```



Country	Region	Sales	a1	a2
Canada	Alberta	100	100	100
Canada	British Columbia	200	200	600
Canada	British Columbia	300	200	600
United States	Montana	100	175	700

Ovo označava da British Columbia i regije prije nje imaju prosjek 200 i sumu 600



Zadaci

1. Ispišite sve stavke, a uz svaku stavite i datum izdavanja računa kojemu pripada. Uz svaki redak ispišite i *running total* ukupne zarade. Resetirajte vrijednost svake godine.

2. Ispišite sve proizvode i pokraj svakog ispišite koliko ima proizvoda njegove boje.

3. Ispišite sve proizvode. Pokraj svakog ispišite koliko je njegova cijena manja od najskupljeg proizvoda. Uzmite u obzira samo proizvode koji imaju cijenu veću od 0.

Analitičke (distribucijske) funkcije

- **Najvažnije analitičke funkcije su:**
  - FIRST_VALUE(col)
    - Vraća prvu vrijednost zadanog stupca (u particiji)
  - LAST_VALUE(col)
    - Vraća prvu vrijednost zadanog stupca (u particiji)
  - LAG(col, offset)
    - Vraća podatak iz nekog prethodnog retka (u particiji)
  - LEAD(col, offset)
    - Vraća podatak iz nekog sljedećeg retka (u particiji)

- **ORDER BY je obavezan (tzv. *running total*), PARTITION BY je opcionalan**

Primjeri analitičkih funkcija

```
select *,
    FIRST_VALUE(Naziv) over (order by Naziv) as a1
from Proizvod where Boja is not null
order by Naziv
```

```
select *,
    FIRST_VALUE(Naziv) over (partition by Boja order by Naziv) as a1
from Proizvod
where Boja is not null order by Boja, Naziv
```

```
select *,
    LAG(Naziv, 3) over (order by IDProizvod) as a1
from Proizvod
where Boja is not null order by IDProizvod
go
```

```
select *,
    LAG(Naziv, 3) over (partition by Boja order by IDProizvod) as a1
from Proizvod
where Boja is not null order by Boja, IDProizvod
```

Zadaci

1. Ispišite sve proizvode s cijenom većom od 0, od najjeftinijeg prema najskupljem. Uz svaki proizvod ispišite koliko je skuplji od prethodnog.

2. Napravite tablicu VozniRed sa stupcima ID, Stanica, Polazak. Ubacite četiri retka s podacima: Zagreb, 11:00; Dugo Selo, 11:35; Kutina, 13:15; Novska 13:45. Ispišite vozni red i uz svaku stanicu napišite koliko minuta traje vožnja do iduće stanice.