

# GRUPISANJE REZULTATA

VI auditorne vježbe

# Teme

---

- ▶ **Grupisanje rezultata**
  - ▶ Postavljanje uslova nad grupom zapisa
  - ▶ Ispitivanje postojanja funkcijske zavisnosti na temelju trenutnog sadržaja relacije
- ▶ **Poredak rezultata**
- ▶ **Pohrana rezultata upita u privremenu relaciju**
- ▶ **Određivanje unije relcija pomoću naredbe SELECT**



# Agregacija i grupisanje

ispit

mbrStud	datIspit	sifPred	sifNastavnik	ocjena
21223	10.01.2017	516	1001	1
21224	14.01.2017	517	1003	3
21225	20.01.2017	520	1004	4
21223	31.01.2017	516	1002	2
21224	31.01.2017	516	1001	1
21225	31.01.2017	516	1001	5
21223	12.06.2017	520	1004	4
21224	16.06.2017	516	1002	3
21223	07.07.2017	517	1003	2

prosjeci

mbrStud	prosJOcjena
21223	2.25
21224	2.33
21225	4.50

- ▶ Kako izračunati prosjek ocjena svakog od studenata?
  - ▶ prosječnu ocjenu za studenta sa matičnim brojem 21223
  - ▶ prosječnu ocjenu za studenta sa matičnim brojem 21224
  - ▶ ... I za sve ostale studente čiji se matični broj pojavljuje u relaciji ispit

# Agregacija i grupisanje

- ▶ Loše rješenje: za svakog studenta napisati po jedan upit

- ▶  $\delta_{\text{prosjeck}(\text{prosjeOcjena21223})}(\mathcal{G}_{\text{AVG}(\text{ocjena})}(\sigma_{\text{mbrStud}=21223}(\text{ispit})))$

```
SELECT AVG(ocjena) AS prosjeOcjena21223
FROM ispit
WHERE mbrStud = 21223
```

prosjeck

prosjeOcjena21223

2.25

- ▶  $\delta_{\text{prosjeck}(\text{prosjeOcjena21224})}(\mathcal{G}_{\text{AVG}(\text{ocjena})}(\sigma_{\text{mbrStud}=21224}(\text{ispit})))$

```
SELECT AVG(ocjena) AS prosjeOcjena21224
FROM ispit
WHERE mbrStud = 21224
```

prosjeck

prosjeOcjena21224

2.33

- ▶ itd. (za svaki matični broj studenta)
- ▶ Postoji li bolje rješenje?

## Grupisanje (*grouping*)

---

- ▶ Zadana je relacija  $r(R)$ . Neka su atributi  $A_1, A_2, \dots, A_m, B_1, B_2, \dots, B_n$  atributi sheme  $R$ . Opšti oblik operacije grupisanja je sljedeći:

$$A_1, A_2, \dots, A_m \mathcal{G}_{\mathcal{AF}_1(B_1), \mathcal{AF}_2(B_2), \dots, \mathcal{AF}_n(B_n)}(r)$$

- a) određuju se grupe  $n$ -torki: u svakoj grupi se nalaze  $n$ -torke koje imaju jednake vrijednosti atributa  $A_1, A_2, \dots, A_m$
- b) za svaku grupu  $n$ -torki izračunavaju se vrijednosti agregatnih funkcija  $\mathcal{AF}_1(B_1), \mathcal{AF}_2(B_2), \dots, \mathcal{AF}_n(B_n)$
- c) za svaku grupu formira se  $n$ -torka s vrijednostima atributa  $A_1, A_2, \dots, A_m$  i izračunatim vrijednostima agregatnih funkcija



# Agregacija i grupisanje

ispit	mbrStud	datIspit	sifPred	sifNastavnik	ocjena
	21223	10.01.2017	516	1001	1
	21223	31.01.2017	516	1002	2
	21223	12.06.2017	520	1004	4
	21223	07.07.2017	517	1003	2
	21224	14.01.2017	517	1003	3
	21224	31.01.2017	516	1001	1
	21224	16.06.2017	516	1002	3
	21225	20.01.2017	520	1004	4
	21225	31.01.2017	516	1001	5

prosjeci	mbrStud	prosJocjena
	21223	2.25
	21224	2.33
	21225	4.50

- ▶ Za svakog studenta ispisati prosječnu ocjenu (ispravno rješenje):

$$\delta_{\text{prosjeci}(\text{mbrStud}, \text{prosJocjena})}(\text{mbrStud } G_{\text{AVG}(\text{ocjena})}(\text{ispit}))$$

- ▶ grupisati po matičnom broju studenta (mbrStud)
- ▶ za svaku grupu izračunati AVG(ocjena)
- ▶ za svaku grupu formirati po jednu n-torku s vrijednošću atributa mbrStud i izračunatim prosjekom

- ▶ obaviti operaciju preimenovanja

# Agregacija i grupisanje

ispit				
mbrStud	datIspit	sifPred	sifNastavnik	ocjena
21223	10.01.2017	516	1001	1
21223	31.01.2017	516	1002	2
21223	12.06.2017	520	1004	4
21223	07.07.2017	517	1003	2
21224	14.01.2017	517	1003	3
21224	31.01.2017	516	1001	1
21224	16.06.2017	516	1002	3
21225	20.01.2017	520	1004	4
21225	31.01.2017	516	1001	5

zbirno			
sifPred	sifNastavnik	prosJOcjena	maxOcjena
516	1001	2.33	5
516	1002	2.50	3
517	1003	2.50	3
520	1004	4.00	4

- ▶ Ispisati prosječnu i najveću ocjenu za svaki predmet i nastavnika
  - ▶ U istu grupu ulaze n-torke koje imaju jednake vrijednosti atributa sifPred i sifNastavnik

$$\delta_{\text{zbirno}}(\text{sifPred}, \text{sifNastavnik}, \text{prosJOcjena}, \text{maxOcjena})(\text{sifPred}, \text{sifNastavnik} \mathcal{G}_{\text{AVG}}(\text{ocjena}), \text{MAX}(\text{ocjena})(\text{ispit}))$$



# Grupisanje rezultata

---

- ▶ Grupisanje se obavlja prema jednom ili više atributa iz relacije (ili više relacija ukoliko su navedene u *FROM clause*).
- ▶ *GROUP BY clause* sadrži popis atributa ili izraza prema kojima se obavlja grupisanje
- ▶ Grupisanje se obavlja tako da se n-torke koje imaju jednake vrijednosti atributa navedenih u listi za grupisanje, kombiniraju u zajedničku grupu.
- ▶ Za svaku dobivenu grupu u rezultatu se pojavljuje samo jedna n-torka.
- ▶ Grupisanje je vrlo korisno u kombinaciji sa agregatnim funkcijama.





# Grupisanje rezultata

---

**Primjer:** Za svaki predmet posebno, odrediti ukupan broj mjesta u rezervisanim dvoranama

```
SELECT sifPred, SUM(kapacitet)
FROM rezervacija INNER JOIN dvorana
ON rezervacija.oznDvorana = dvorana.oznDvorana
GROUP BY sifPred
```

<b>sifPred</b>	<b>oznVrstaDan</b>	<b>oznDvorana</b>	<b>sat</b>	<b>kapacitet</b>
MAT	PO	B - 1	8	70
MAT	PO	B - 1	9	70
MAT	UT	A - 101	9	30
FIZ	SR	A - 202	11	40
FIZ	CE	B - 1	11	70
RAC	UT	A - 102	14	30
RAC	SR	A - 201	18	40
TEH	PE	B - 5	12	50



# Grupisanje rezultata

---

- ▶ Rezultat je:

<b>sifPred</b>	<b>(SUM)</b>
MAT	170
FIZ	110
RAC	70
TEH	50

- ▶ **Važno je slijedeće pravilo:** svi atributi i izrazi koji se nalaze u listi za selekciju, a koji nisu unutar agregatnih funkcija, moraju biti navedeni u GROUP BY listi.
- ▶ Međutim, dozvoljeno je u GROUP BY listi koristiti i one attribute koji se ne nalaze u listi za selekciju, npr.

```
SELECT SUM(kapacitet) FROM rezervacija INNER JOIN  
dvorana ON rezervacija.oznDvorana =  
dvorana.oznDvorana  
GROUP BY sifPred
```



# Grupisanje rezultata

---

- ▶ **Sljedeći upit:**

```
SELECT sifPred, oznDvorana, SUM(kapacitet)
FROM rezervacija INNER JOIN dvorana
    ON rezervacija.oznDvorana = dvorana.oznDvorana
GROUP BY sifPred
```

- ▶ **NE VRIJEDI** jer se za svaki predmet dobije samo jedna grupa (dakle jedan red izlaznog rezultata).

- ▶ U listi atributa prema kojima se obavlja grupisanje mogu se umjesto imena atributa navoditi redni brojevi atributa iz liste za selekciju

```
SELECT sifPred, sifNastavnik, AVG(ocjena) FROM ispit
GROUP BY 1, 2
```

- ▶ Osim prema atributima, grupisanje se može obaviti i prema izrazima (*Expression*), ali se u tom slučaju u listi atributa za grupisanje **moraju** koristiti redni brojevi atributa/izraza iz liste za selekciju

```
SELECT ocjena*10, COUNT(*) FROM ispit
GROUP BY 1 -- ne smije se napisati GROUP BY ocjena*10
```



# Postavljanje uslova nad grupom zapisa

---

- ▶ Koristi se *HAVING clause*.
- ▶ Dok se uz pomoć *WHERE clause* izdvajaju one n-torke koje će formirati grupe definisane u *GROUP BY clause*, *HAVING clause* služi za postavljanje uslova kojeg dobivene grupe moraju zadovoljiti da bi se pojavile u rezultatu.
- ▶ U *HAVING* dijelu naredbe dozvoljeno je koristiti agregatne funkcije.
- ▶ U *HAVING* dijelu naredbe dozvoljeno je koristiti samo one attribute koji se nalaze u *GROUP BY* listi.
- ▶ Atributi koji se ne nalaze u *GROUP BY* listi smiju se koristiti jedino kao argumenti agregatnih funkcija.



# Postavljanje uslova nad grupom zapisa

---

- ▶ **Primjer:** Ispis šifri predmeta i broja položenih ispita, ali samo onih predmeta za koje je položeno više od dva ispita.

```
SELECT sifPred, COUNT(*) FROM ispit
WHERE ispit.ocjena > 1
GROUP BY sifPred
HAVING COUNT(*) > 2
```

- ▶ WHERE dio naredbe određuje koje n-torke će formirati grupe (samo položeni ispiti).
- ▶ GROUP BY lista određuje strukturu grupa tj. po kojim atributima se obavlja grupisanje n-torki (sve n-torke koje imaju jednaku šifru predmeta ulaze u jednu grupu).
- ▶ HAVING dio naredbe određuje koje od nastalih grupa će biti prihvaćene kao rezultat (samo one grupe u kojima je broj n-torki, tj. COUNT(\*) veći od dva).
- ▶ Ispisuje se po jedan zapis za svaku grupu koja zadovoljava taj uslov.



# Postavljanje uslova nad grupom zapisa

```
SELECT sifPred, COUNT(*)  
FROM ispit  
WHERE ispit.ocjena > 1  
GROUP BY sifPred  
HAVING COUNT(*) > 2
```

<u>sifPred</u>	<u>ocjena</u>
1071	3
1071	2
1095	4
1095	1
1095	1
2091	1
2091	3
2091	5
2091	2
2091	3
5048	4
5048	1
5048	5
5048	4



<u>sifPred</u>
1071
1071
1095
2091
2091
2091
2091
2091
5048
5048
5048



<u>sifPred</u>	<u>(count)</u>
1071	2
1095	1
2091	4
5048	3



<u>sifPred</u>	<u>(count)</u>
2091	4
5048	3

## Postavljanje uslova nad grupom zapisa

---

- ▶ U HAVING dijelu naredbe dozvoljeno je koristiti uslove sa podupitima na isti način kao što se koriste u WHERE clause.
- ▶ **Primjer:** Ispisati šifre predmeta za koje je prosjek ocjena veći od ukupnog prosjeka ocjena.

```
SELECT sifPred FROM ispit
GROUP BY sifPred
HAVING AVG(ocjena) > (SELECT AVG(ocjena)
                      FROM ispit)
```



# Ispitivanje postojanja funkcijske zavisnosti na temelju trenutnog sadržaja relacije

---

- ▶ Upotrebom agregatnih funkcija jednostavno se ispituje vrijedi li neka funkcijska zavisnost u relaciji
- ▶ POZOR: može se odrediti jedino vrijedi li funkcijska zavisnost za trenutni sadržaj relacije, ali ne i vrijedi li za relacijsku shemu.
- ▶ Iz definicije funkcijske zavisnosti proizilazi da funkcijska zavisnost  $X \rightarrow Y$  ne vrijedi u relaciji ukoliko u nekoj grupi zapisa s jednakom  $X$  vrijednošću postoji više od jedne različite  $Y$  vrijednosti.
- ▶ Kada se određuje funkcijska zavisnost  $X \rightarrow Y$ , formira se upit u kojem se zapisi grupišu prema atributima iz  $X$ . Ispisuju se one grupe za koje postoji više od jedne različite vrijednosti od  $Y$ .





# Ispitivanje postojanja funkcijske zavisnosti na temelju trenutnog sadržaja relacije

---

- ▶ Sljedećom naredbom se u relaciji ispit ispituje funkcijska zavisnost **{mbrStud, datIspit} → sifPred**

```
SELECT mbrStud, datIspit FROM ispit
```

```
GROUP BY 1, 2
```

```
HAVING COUNT(DISTINCT sifPred) > 1
```

<b>mbrStud</b>	<b>datIspit</b>
----------------	-----------------

36251744	13.01.1996
----------	------------

36251744	21.11.1995
----------	------------

- ▶ Dvije dobivene n-torke pokazuju da ispitivana funkcijska zavisnost ne vrijedi.



# Ispitivanje postojanja funkcijske zavisnosti na temelju trenutnog sadržaja relacije

---

- ▶ Budući se sljedećom naredbom ne dobija niti jedna n-torka rezultata, pokazano je da za trenutni sadržaj relacije ispit vrijedi funkcijska zavisnost **{mbrStud, datIspit, sifPred} → ocjena**

```
SELECT mbrStud, datIspit, sifPred FROM ispit  
      GROUP BY 1, 2, 3  
      HAVING COUNT(DISTINCT ocjena) > 1
```

- ▶ Funkcijske zavisnosti oblika  $X \rightarrow Y$  u kojima skup  $Y$  sadrži više od jednog atributa, npr  $Y = \{A, B\}$  nije moguće ispitati na ovaj način, budući da nije dozvoljena upotreba `HAVING COUNT(DISTINCT A, B)`
- ▶ Međutim, u takvim se slučajevima mogu pojedinačno ispitati funkcijske zavisnosti  $X \rightarrow A$  i  $X \rightarrow B$ , te ukoliko one vrijede, korištenjem pravila o aditivnosti (unije), zaključiti da vrijedi i funkcijska zavisnost  $X \rightarrow Y$ .



# Poredak rezultata

---

- ▶ Rezultat SELECT naredbe se može poredati (sortirati) prema atributima iz liste za selekciju.
- ▶ Iza ključne riječi ORDER BY navode se atributi prema kojima se obavlja sortiranje.
- ▶ Opcionalno se uz svaki izraz može navesti smjer sortiranja – ključna riječ ASC ili DESC.
- ▶ ASC (skraćenica za *ascending*) znači da će se poredak obaviti u uzlaznom smjeru, a DESC (skraćenica za *descending*) u silaznom.
- ▶ Ukoliko se smjer sortiranja ne navede, smatra se da se sortiranje obavlja uzlazno.



# Poredak rezultata

---

```
SELECT nazMjesto, prezStud FROM stud JOIN mjesto  
ON stud.pbrStan = mjesto.pbr  
ORDER BY nazMjesto DESC, prezStud ASC
```

<b>nazMjesto</b>	<b>prezStud</b>
------------------	-----------------

Zagreb	Abram
--------	-------

Zagreb	Kolar
--------	-------

Rijeka	Jambrek
--------	---------

Rijeka	Kolar
--------	-------

Rijeka	Novak
--------	-------

- ▶ **ORDER BY** lista se može napisati skraćeno, tako da se umjesto imena atributa ili izraza koriste redni brojevi izraza iz liste za selekciju. **ORDER BY** lista u prethodnom upitu bi se mogla napisati ovako:

```
ORDER BY 1 DESC, 2 ASC
```

- ▶ Sortiranje se može obaviti jedino po atributima ili izrazima navedenim u listi za selekciju



# Poredak rezultata

---

- ▶ Sortiranje se može obaviti prema podnizu nekog atributa koji je tipa niza znakova

```
SELECT nazMjesto, prezStud FROM stud JOIN mjesto
      ON stud.pbrStan = mjesto.pbr
      ORDER BY SUBSTRING(nazMjesto, 2, 5)
```

- ▶ Sortiranje se može obaviti i prema izrazima (*Expression*) iz liste za selekciju, ali se u tom slučaju u listi atributa za poredak moraju koristiti ili redni brojevi ili *alias* imena atributa/izraza iz liste za selekciju.

```
SELECT sifPred, ocjena*10 FROM ispit
      ORDER BY 2
```

```
SELECT sifPred, ocjena*10 umnozak FROM ispit
      ORDER BY umnozak
```



# Pohrana rezultata upita u privremenu relaciju

---

- ▶ `CREATE TEMPORARY TABLE` naredba koja prethodi `SELECT` naredbi služi za istovremeno kreiranje i punjenje privremene relacije
- ▶ Privremena relacija ima sve karakteristike trajne relacije kreirane naredbom `CREATE TABLE`, osim što se ne vidi u riječniku podataka, te biva uništena u trenutku kada završava program u kojem je kreirana
- ▶ Može se uništiti i upotrebom naredbe `DROP TABLE`.
- ▶ Shema privremene relacije određena je sadržajem liste za selekciju `SELECT` naredbe kojom je kreirana.
- ▶ Tipovi podataka privremene relacije određeni su tipovima podataka iz liste za selekciju, a nazivi atributa odgovaraju nazivima selektiranih atributa ili *alias* imenima koja su im pridružena.
- ▶ Članovi liste za selekciju koji nisu atributi (dakle izrazi – *Expression*) **moraju** biti imenovani uz pomoć *alias* imena.



# Pohrana rezultata upita u privremenu relaciju

---

- ▶ **Primjer:** Kreirati privremenu relaciju koja sadrži matične brojeve studenata i pripadne prosječne ocjene, te je istovremeno napuniti s podacima koji su rezultat **SELECT** naredbe.

```
CREATE TEMPORARY TABLE prosjeci AS
SELECT mbrStud, AVG(ocjena) prosjecnaOcjena
FROM ispit
WHERE ocjena > 1
GROUP BY mbrStud
```

- ▶ Korisnik koji je izveo prethodnu naredbu može privremenu relaciju **prosjeci** sa atributima **mbrStud** i **prosjecnaOcjena** koristiti kao bilo koju drugu relaciju, sve do trenutka kada program terminira ili dok relaciju **prosjeci** ne uništi pomoću naredbe **DROP TABLE**. Npr.

```
SELECT MAX(prosjecnaOcjena) FROM prosjeci
```

---



# Određivanje unije relcija pomoću naredbe SELECT

---

- ▶ Moguće je dvije ili više SELECT naredbi povezati pomoću UNION operatora.
- ▶ Rezultati SELECT naredbi moraju pri tome biti unijski kompatibilni na nivou tipova podataka
- ▶ Samo posljednja SELECT naredba u nizu smije sadržavati ORDER BY *clause*.

```
SELECT pbr, nazMjesto FROM mjestoRod  
UNION
```

```
SELECT pbr, nazMjesto FROM mjestoStan
```

- ▶ Ukoliko se UNION u prethodnom primjeru zamijeni sa UNION ALL, upit ne bi obavio operaciju unije na ispravan način, jer iz rezultata ne bi eliminirao n-torke koje se javljaju dva ili više puta.





# Primjeri

---

- ▶ Ispisati rang listu organizacionih jedinica prema broju položenih ispita iz predmeta koji se predaju na tim organizacionim jedinicama. Organizacione jedinice sa eventualno istim brojem položenih ispita poredati abecedno prema nazivu. Ispisivati naziv organizacione jedinice i broj položenih ispita

```
SELECT nazOrgjed, COUNT(*) polozenoIspita
FROM orgjed INNER JOIN pred
ON orgjed.sifOrgjed = pred.sifOrgjed
INNER JOIN ispit ON pred.sifPred = ispit.sifPred
WHERE ocjena > 1
GROUP BY sifOrgjed, 1
ORDER BY 2 DESC, 1
```



# Primjeri

---

- ▶ Ispisati rang listu mjesta prema prosječnoj ocjeni položenih ispita u tekućoj godini svih studenata koji stanuju u tom mjestu. Ispisivati naziv mjesta i prosječnu ocjenu, a mjesta koja eventualno imaju istu prosječnu ocjenu poredati abecedno prema nazivu.

```
SELECT nazMjesto, AVG(ocjena)
FROM ispit INNER JOIN stud
ON ispit.mbrStud = stud.mbrStud
INNER JOIN mjesto ON stud.pbrStan = mjesto.pbr
WHERE ocjena > 1
AND YEAR(datIspit) = YEAR(CURRENT_DATE)
GROUP BY pbr, 1
ORDER BY 2 DESC, 1
```



# Primjeri

---

- ▶ Ispisati nazive, kratice predmeta i prosječnu ocjenu položenih ispita iz tih predmeta za sve predmete čija je ta prosječna ocjena veća od prosječne ocjene ispita iz svih predmeta sa iste organizacione jedinice.

```
SELECT nazPred, kratPred, AVG(ocjena) prosjek
FROM ispit INNER JOIN pred ON ispit.sifPred = pred.sifPred
WHERE ocjena > 1
GROUP BY 1, 2
HAVING AVG(ocjena) > (SELECT AVG(ocjena)
                        FROM ispit INNER JOIN pred pred1
                        ON pred1.sifPred = ispit.sifPred
                        WHERE pred1.siforgjed = pred.sifOrgjed
                        AND ocjena > 1)
```



# Primjeri

---

- Kreirati i napuniti privremenu relaciju **nastTmp** čija je shema jednaka shemi relacije nastavnik uz dodatak atributa **polIspit** i **prosjek** koji sadrže broj položenih ispita i prosječnu ocjenu položenih ispita svakog nastavnika, respektivno. Relaciju napuniti samo podacima o nastavnicima kod kojih je broj položenih ispita veći od broja studenata rođenih u istom gradu u kojem stanuje nastavnik.

```
CREATE TEMPORARY TABLE nastTmp AS
SELECT nastavnik.*, COUNT(*) polIspit, AVG(ocjena) prosjek
FROM nastavnik INNER JOIN ispit
    ON ispit.sifNastavnik = nastavnik.sifNastavnik
WHERE ocjena > 1
GROUP BY 1, 2, 3, 4, 5, 6
HAVING COUNT(*) > (SELECT COUNT(*) FROM stud
    WHERE stud.pbrRod = nastavnik.pbrStan)
```

