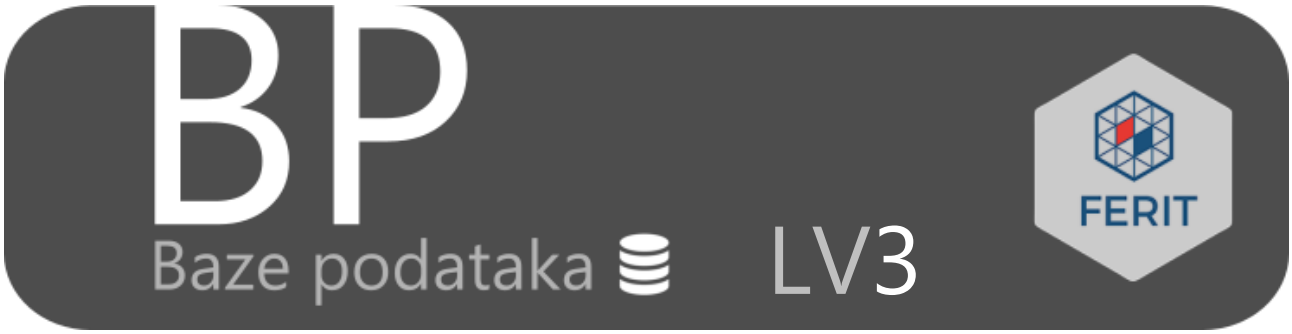

1. Ugrađene funkcije

SQL definira ugrađene funkcije koje se u osnovi dijele na:

- **Skalarne funkcije**
  - djeluju nad jednom vrijednošću i kao rezultat daju jednu vrijednost.
  - mogu se koristiti uvijek tamo gdje se očekuje izraz (*expression*)
- **Agregatne funkcije**
  - djeluju nad skupom vrijednosti, a kao rezultat daju jednu, skupnu vrijednost

1.1. Skalarne funkcije

Postoje sljedeće skalarne funkcije:

- Matematičke funkcije
  - ABS, SIN, COS, EXP, POWER, LOG, SQUARE, SQRT, RAND, ROUND, ...
- Datumske funkcije
  - DAY, MONTH, YEAR, DATEDIFF...
- Tekstualne funkcije
  - SUBSTRING, LEN, LOWER, UPPER, ...

SQL skalarne funkcije:

ABS (*numeric_expression*) - apsolutna vrijednost zadanog broja,
SIN (*numeric_expression*) - trigonometrijska funkcija sinus,
COS (*numeric_expression*) - trigonometrijska funkcija kosinus,
EXP(*numeric_expression*) - eksponencijalna vrijednost broja,
LOG (*numeric_expression*) - prirodni logaritam,
POWER (*numeric_expression*, *power*) - zadani broj potenciran na vrijednost *power*,
SQUARE (*numeric_expression*) - kvadrat zadanog broja,
RAND (*numeric_expression*) - slučajni broj između 0 i 1,
ROUND (*numeric_expression*, *length*) - vrijednost *numeric_expression* zaokružena na *length* decimala,
SQRT (*numeric_expression*) - korijen zadanog broja,
DAY (*date*) - cjelobrojna vrijednost koja predstavlja dan iz zadanog datuma,
MONTH (*date*) - cjelobrojna vrijednost mjeseca iz zadanog datuma,

YEAR (*date*) - cjelobrojna vrijednost godine iz zadanog datuma,
DATEDIFF(*datepart*, *startdate*, *enddate*) - vraća proteklo vrijeme između dva datuma, gdje *datepart* (year, month, day, second, milisecond itd.) predstavlja oblik u kojemu će se razlika predstaviti,
SUBSTRING (*expression* , *start*, *length*) - dio zadanog stringa (*expression*) od pozicije *start* u ukupnoj dužini *length*,
LEN (*string_expression*) - broj znakova u stringu bez završnih praznina (ako ih ima),
LOWER (*character_expression*) - zadani izraz napisan malim slovima,
UPPER (*character_expression*) - zadani izraz napisan velikim slovima.

Primjer (`stuslu.stuslu`):

```
SELECT SQUARE (3);

SELECT UPPER (ime_stud)
FROM student
WHERE spol = 'M';

SELECT DISTINCT SUBSTRING(ime_stud, 1, 3) AS krace_ime
FROM student;
```

1.2. Agregatne funkcije

Najčešće se koriste uz **GROUP BY** izraz SELECT naredbe, te su kao izrazi dozvoljene samo u SELECT i HAVING izrazima SELECT naredbe. Ne uzimaju obzir NULL vrijednosti, iznimka je jedino COUNT funkcija.

SQL agregatne funkcije:

AVG ([ALL | DISTINCT] *expression*) - Prosječna vrijednost skupa elemenata
COUNT ({ [ALL | DISTINCT] *expression*] | * }) - Broj elemenata u skupu
MAX ([ALL | DISTINCT] *expression*) - Maksimalna vrijednost skupa
MIN ([ALL | DISTINCT] *expression*) - Minimalna vrijednost skupa
SUM ([ALL | DISTINCT] *expression*) - Ukupna suma skupa

Primjer:

```
SELECT mbr_stud, AVG(ocjena) AS 'prosjek'
FROM ispit
WHERE ocjena > 1
GROUP BY mbr_stud;
```

2. CREATE VIEW naredba

Naredba za kreiranje pogleda. Pogledi su u načelu prethodno snimljeni SELECT upiti nad bazom podataka. Pogledi mogu imati određene prednosti nad tablicama:

- Pogledi mogu vraćati samo određeni podskup podataka u tablicama, drugim riječima, mogu ograničiti izloženost podataka u tablicama određenim korisnicima te time povećati zaštitu i sigurnost podataka u bazi.
- Mogu skrivati složenost baze podataka.
- Mogu vraćati spojeve više tablica u jednu unificiranu tablicu te koristiti agregacije za prikazivanje dodatnih informacija.

Sintaksa:

```
CREATE VIEW [< owner > . ] view_name [ ( column [ ,...n ] ) ]
AS select_statement
```

***owner* -** Naziv korisnika koji je vlasnik pogleda

***view_name* -** Naziv pogleda, eksplicitno navođenje naziva vlasnika je opcionalno

***select_statement* -** SELECT naredba koja definira pogled. SELECT naredba može biti kombinacija više SELECT naredbi povezanih sa UNION ili UNION ALL. Ograničenja koja postoje na SELECT naredbu su slijedeća:

- ne smije sadržavati ORDER BY parametar (osim u specijalnim slučajevima)
- ne smije sadržavati INTO parametar

Primjeri:

```
CREATE VIEW prosjek_po_studentu AS
SELECT mbr_stud, AVG(ocjena) AS 'prosjek'
FROM ispit
WHERE ocjena > 1
GROUP BY mbr_stud;
```

```
CREATE VIEW zenski_studenti AS
SELECT mbr_stud, ime_stud, prez_stud
FROM student
WHERE spol = 'Ž';
```

3. Zadaci

Zadatak 1. (**baza podataka 'stuslu'**)
Ispisati inicijale te godinu rođenja studenata iz tablice 'student'. Ispisati sve studente i njihove podatke te ih posložiti po starosti počevši od najstarijeg prema najmlađem. Ispisati ime i prezime najstarijeg ženskog studenta.

Zadatak 2. (**baza podataka 'stuslu'**)
Ispisati ukupni broj studenata iz tablice student. Ispisati iz koliko različitih mjesta dolaze studenti fakulteta.

Zadatak 3. (**baza podataka 'stuslu'**)
Ispisati prosječnu ocjenu svih položenih predmeta u tablici 'ispit.'

Zadatak 4. (**baza podataka 'stuslu'**)
Ispisati ime i prezime i prosječnu ocjenu svih uspješno položenih ispita za svakog pojedinog studenta, posloženo od najveće do najmanje prosječne ocjene. Ispisati matični broj studenata koji imaju prosječnu ocjenu položenih ispita veću od 2.5

Zadatak 5. (baza podataka 'stuslu')
Kreirati pogled koji će imati atribute slične kao u tablici ispit, ali će umjesto matičnog broja studenta imati ime i prezime studenta, a umjesto šifre predmeta – naziv predmeta.

4. Assignments

Assignment 1. (database 'stuslueng')
Get the initials and year of birth of students from the table 'student'. Get all the students from the same table and order them by their age starting from the oldest. Get the first and last name of the oldest female student.

Assignment 2. (database 'stuslueng')
Get the total number of students from the table 'student'. Get the number of distinct cities that the students come from.

Assignment 3. (database 'stuslueng')
Get the average grade for all the passed courses (exams) in the table 'exams'.

Assignment 4. (database 'stuslueng')
Get the first name, last name and the average grade for all the passed courses for each student and sort the results by the average grade in a descending order. Get the IDs of all the students that have the average grade of 2.5 (of all the passed courses).

Assignment 5. (database 'stuslueng')

Create a view that has similar attributes as the table 'exam', but instead of the student ID the view should have students' first and last name, and instead of the course ID there should be a course name.