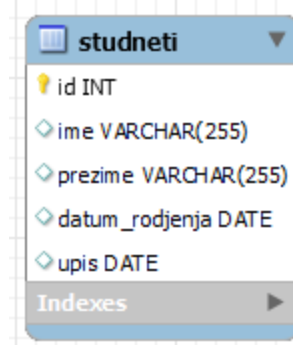


## 2-2 SQL upiti

### SQL upiti INSERT, DELETE, UPDATE

Nakon što se tabela kreira, moguće je unijeti podatke u tabelu. Recimo da imamo tabelu studenti i da želimo unijeti studente u tabelu. To se može uraditi na sljedeći način:

```
INSERT INTO studenti(ime, prezime)
VALUES ('Alina', 'Alinic');
```



Ukoliko unosimo podatke u sve kolone u tabeli, ne moramo eksplicitno definisati nazive kolona, ali u ovom slučaju moramo definisati vrijednosti za sve kolone:

```
INSERT INTO studenti
VALUES(100, 'AAlina', 'Alinic', '2002-01-02', '2020-02-02', '');
```

Prilikom ubacivanja imena potkrala se greška, pa je slovo A duplirano. Da bi smo ovo izmjenili u u bazi, koristimo UPDATE komandu. Npr.

```
UPDATE studenti SET ime='Alina' WHERE id=100;
```

Izmjene će biti primjenjene na sve pronađene redove. Ukoliko se slučajno izostavi WHERE klauzula, onda će izmjene biti primjenjene na sve redove.

I na kraju vrijedi spomenuti brisanje. Da se obriše neki red u tabeli, koristi se DELETE FROM komanda:

```
DELETE FROM studenti WHERE id=100;
```

Za ovu komandu vrijedi isto što i za update. Ukoliko se izostavi WHERE klauzula, brisanje će biti primjenjeno na sve redove u tabeli.

### SQL upiti SELECT, FROM, WHERE

Naredba SELECT se koristi za pregled podataka u tabeli. Podaci koji se dobiju se zovu rezultni skup (result-set). Za pregled svih unosa u ovoj tabeli koristit će se komanda SELECT:

```
SELECT * FROM studneti;
```

Slično kao i kod UPDATE i DELETE, možemo koristiti WHERE da bliže označimo koje redove želimo:

```
SELECT * FROM studneti WHERE id=3;
```

Ovo u prevodu znači, selektuj sve redove iz tabele studenti gdje je vrijednost kolone id, 3. Pošto je *id* primarni ključ i time jedinstveni identifikator reda, rezultat će imati samo jedan red.

Zvjezdica nakon ključne riječi **SELECT** znači da se vrate sve kolone koje su dostupne. Umjesto zvjezdice možemo naznačiti specifične kolone koje želimo da dobijemo. Npr:

```
SELECT id, ime, prezime FROM studneti WHERE id=3;
```

Ovom komandom neće se dobiti vrijednosti za *datum\_rodjenja* i upis za studenta pod id=3. Za sada, vrijedi spomenuti da se mogu koristiti operatori: >, <, =, <=, >= i <> za različito.

Generalizujući, sintaksa ove komande izgleda ovako:

```
SELECT column1, column2, ... FROM table_name WHERE condition;
```

Poslije ključne riječi **SELECT** ide lista kolona (ili zvjezdica) kao i njihovih modifikatora. Nakon toga ide **FROM** sa tabelom ili listom tabela. Klauzula **WHERE** služi za dodatno filtriranje redova po određenim uslovu ili uslovima.

## **SELECT DISTINCT**

Poseban dodatak **SELECT** komandi nam omogućava da vratimo listu jedinstvenih imena. Npr.:

```
SELECT DISTINCT ime FROM studneti;
```

će selektovati sva različita imena tj. sve jedinstvene vrijednosti iz tabele studenti za kolonu *ime*. Ovo se također može kombinovati sa **WHERE** izrazom tako da će se **DISTINCT** primjeniti samo na redove filtrirane koristeći **WHERE** izraz.

## Preimenovanje kolona

Koristeći **SELECT** upit, moguće je preimenovati izlaznu kolonu, ili izvršiti neku drugu vrstu manipulacije nad kolonom. Ovo se zove alias kolone, tj. dodjeljivanje nadimka koloni.

```
SELECT DISTINCT ime AS jedinstvena_imena FROM studneti;
```

## Preimenovanje tabela

U SQL upitima moguće je preimenovati tabelu radi lakšeg korištenja. Prije toga, potrebno je reći da su sljedeće dvije komande ekvivalentne:

```
SELECT id, ime, prezime FROM studneti;
```

i

```
SELECT studneti.id, studneti.ime, studneti.prezime FROM studneti;
```

Tj. moguće je navesti naziv tabele prije naziva kolone kako bi se naznačila tabela koja se treba upotrijebiti. Ovo će biti posebno korisno prilikom spajanja tabela kasnije. Međutim pošto je ovo poprilično dugo za pisati, pogotovo u slučaju ako tabela ima dug naziv, moguće je dodijeliti nadimak tabeli. Npr. sljedeća komanda je jednaka prethodnim:

```
SELECT s.id, s.ime, s.prezime FROM studneti s;
```

## Ograničavanje izlaznog skupa sa LIMIT i OFFSET

S obzirom da tabele mogu imati milione redova, ukoliko je potrebno ograničiti količinu redova koje nam baza vraća to se može učiniti korištenjem klauzule **LIMIT** kao dodatka **SELECT** upitu. Npr.:

```
SELECT * FROM studenti LIMIT 3;
```

će vratiti 3 reda iz tabele studenti. Ova klauzula se stavlja na kraj upita.

Postoje situacije u kojima nije potrebno samo ograničiti broj izlaznih redova. Šta ukoliko imamo web stranicu koja ima listu artikala i ta list ima svoje stranice. Tako prva stranica ima 10 artikala, druga, sljedećih 10, treća sljedećih 10, itd... U tom slučaju, može se postaviti **OFFSET** što predstavlja prvu poziciju od koje **LIMIT** počinje važiti. Npr.:

```
SELECT * FROM studenti LIMIT 3 OFFSET 2;
```

Da se primjetiti da izlazni skup kreće od broja 3, dakle prva dva su preskočena. Kao kraću verziju ovog upita, može se staviti:

```
SELECT * FROM studenti LIMIT 2, 3;
```

gdje par brojeva 2,3 jednostavno znači: preskoči prva dva reda i prebroj tri.

## Prebrojavanje redova

Ukoliko je potrebno vratiti broj redova za određen upit, tj. prebrojati redove izlaza, može se koristiti agregacijska funkcija unutar **SELECT** klauzule. Npr.

```
SELECT COUNT(*) FROM studneti;
```

Ovo znači da će baza primjeniti agregacijsku funkciju na izlazni skup podataka. Broj redova za prebrojavanje se može dodatno suziti koristeći **WHERE** klauzulu. Ili sa preimenovanom kolonom:

```
SELECT COUNT(*) AS ukupan_broj_studenata FROM studneti;
```