2-6 Spajanje tabela

JOIN

U vježbi 2-4, pod brojem 9. (u pojednostavljenoj varijanti) tražena su imena svih studenata sa ocjenom 5. Međutim, imena se ne nalaze u tabeli ocjene. Jedini način da se ovo uradi je da se na neki način povežu tabele studenti i ocjene. U tabeli ocjene, postoji kolona *student_id* koja zapravo predstavlja studenta sa kojim je ocjena data. Ovo se može iskoristiti na sljedeći način:

```
SELECT studenti.ime, studenti.prezime, ocjene.ocjena
FROM studenti, ocjene
WHERE studenti.id = ocjene.student_id AND ocjene.ocjena=5;
```

Za razliku od prethodnih upita, sada su korištene dvije tabele u `FROM` klauzuli. Kako bi se skratilo pisanje, može se koristiti aliasing:

```
SELECT s.ime, s.prezime, o.ocjena
FROM studenti s, ocjene o
WHERE s.id = o.student_id AND o.ocjena=5;
```

Ovo je jedna vrsta JOIN-a, ili povezivanja jedne ili više tabela (u ovom slučaju dvije). Ova vrsta povezivanja je implicitno lijevo unutrašnje povezivanje (*LEFT INNER JOIN*). Razlikuje se nekoliko tipova veza, unutrašnje i vanjsko (*INNER i OUTER*) i lijevo i desno (*LEFT i RIGHT*).

Potrebno je spomenuti da se ovi nazivi djelomično razlikuju od DBMS-a do DBMS-a međutim u ovom slučaju govorimo o MySQL ili MariaDB bazi podataka. U MySQL bazi postoje ove vrste veza:

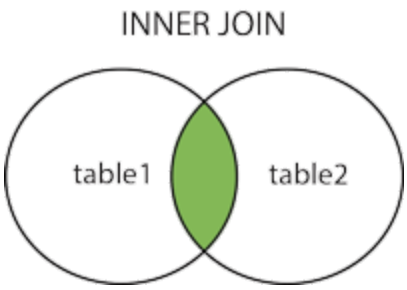
- `INNER JOIN`
- `LEFT OUTER JOIN` ili `LEFT JOIN`
- `RIGHT OUTER JOIN` ili `RIGHT JOIN`
- `FULL OUTER JOIN` ili `CROSS JOIN`
- `SELF JOIN`
- `UNION / ALL`

JOIN se koristi tako što se naznače ključne riječi koje predstavljaju vezu a potom kolone preko kojih se dvije tabele vežu. Npr.

```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;
```

INNER JOIN

Ova vrsta JOIN-a vraća redove koje imaju unose u obje tabele.



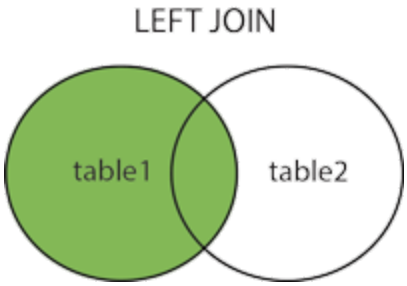
Npr. ako je potrebno pronaći sve učenike sa ocjenom 5, trebamo spojiti tabele studenti i ocjene

```
SELECT s.ime, s.prezime, o.ocjena
FROM studenti s INNER JOIN ocjene o ON s.id = o.student_id
WHERE o.ocjena=5;
```

Primjeti se da su rezultati ovog upita isti kao rezultati implicitnog upita, gdje su dvije tabele spojene definišući vezu preko `WHERE` klauzule. Potrebno je reći da je rezultat između ova dva upita isti međutim eksplicitno definisanje `INNER JOIN` pruža mogućnost bazi da bolje optimizuje izvršenje upita pa tako eksplicitni `INNER JOIN` ima bolje performanse.

LEFT JOIN

`LEFT JOIN` vraća sve rezultate iz lijeve tabele, i podudarajuće rezultate iz desne tabele.



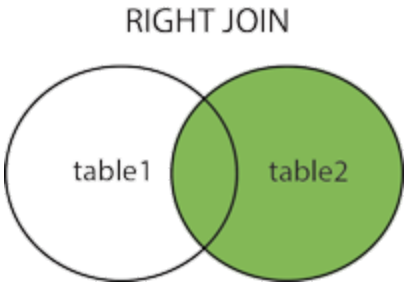
Ukoliko nema podudarajućih rezultata, na njihovom mjestu stajat će `NULL`. Npr. ukoliko je potrebno pronaći sve studente i njihove ocjene, uključujući i studente koji nisu ocijenjeni, onda se to može uraditi na sljedeći način:

```
SELECT s.id, s.ime, s.prezime, o.ocjena
FROM studenti s
LEFT JOIN ocjene o ON s.id = o.student_id;
```

Jednako isti rezultat bi bio dobiven ako bi se koristilo `LEFT OUTER JOIN`.

RIGHT JOIN

`RIGHT JOIN` je obratno od `LEFT JOIN` i vraća sve rezultate iz desne tabele i podudarajuće rezultate iz lijeve tabele.

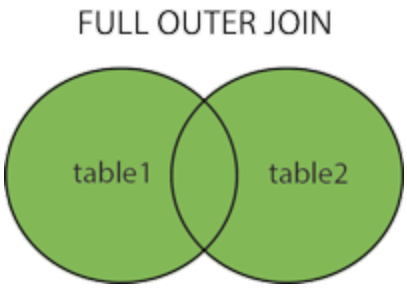


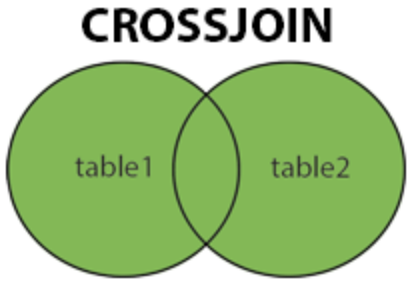
Npr. ako želimo vratiti sve ocjene i podudarajuće predmete, uključujući predmete koji nemaju ocjene, možemo reći:

```
SELECT o.ocjena, p.naziv
FROM ocjene o
RIGHT JOIN predmeti p ON p.id = o.predmet_id;
```

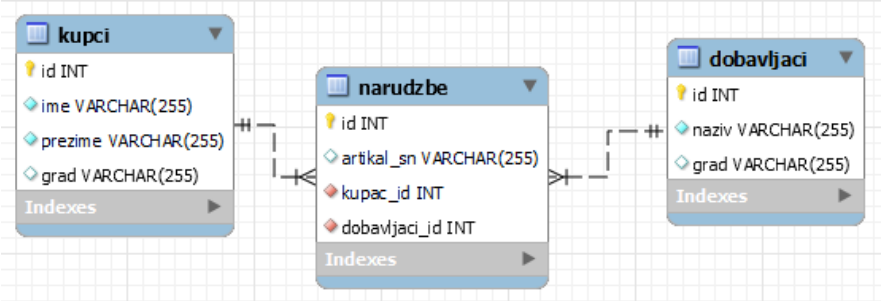
FULL JOIN ili `CROSS JOIN`

Puno spajanje dvije tabele u izlazu vraća rezultate iz obje tabele. Ovo se također zove `FULL OUTER JOIN`. U MySQL bazi podataka, ovo se zove `CROSS JOIN`:





Ovaj JOIN, vraća sve rezultate iz obje tabele, bez obzira ima li podudaranja ili ne. Npr. recimo da postoje tabele za kupce, narudžbe i dostavljače:



```
SELECT k.ime, k.prezime, n.artikal_id
FROM kupci k
CROSS JOIN narudzbe n ON k.id = n.kupac_id;
```

Pošto `CROSS JOIN` vraća vrijednosti iz obje tabele bez obzira ima li podudaranja ili ne, stoga će u izlazu biti kupci koji nemaju nikakvih narudžbi ali i narudžbe za koje nema podudarajućeg polja u tabeli kupci.

SELF JOIN

SELF JOIN je obični JOIN kao i svaki drugi, stim da se tabela spaja sama sa sobom. Na primjeru tabele kupci, koristeći SELF JOIN možemo vratiti sve kupce koji se nalaze u istom gradu:

```
SELECT a.ime, a.prezime, b.ime, b.prezime, a.grad
FROM kupci a, kupci b
WHERE a.id <> b.id AND a.grad = b.grad;
```

Ovaj upit će vratiti sve kupce koji se nalaze u istom gradu.

UNION / ALL

`UNION` operater se koristi da kombinira rezultate dva ili više `SELECT` izraza, međutim, pošto funkcioniše kao unija dva skupa, da bi funkcionisalo, mora biti ispunjeno par zahtjeva:

- Svaki `SELECT` upit mora imati isti broj kolona
- Sve kolone moraju imati podudarajuće tipove podataka
- Kolone u svakom `SELECT` izrazu moraju biti na istim pozicijama

Npr. ako želimo da pronađemo sve gradove iz tabele kupci i iz tabele dobavljači, možemo napisati:

```
SELECT grad FROM kupci
UNION
SELECT grad FROM dobavljaci;
```

Ovo će vratiti sva jedinstvena imena gradova iz dvije tabele, tj. bez duplikata. Ako želimo da vratimo sa duplikatima, možemo koristiti UNION ALL:

```
SELECT grad FROM kupci
UNION ALL
SELECT grad FROM dobavljaci;
```