

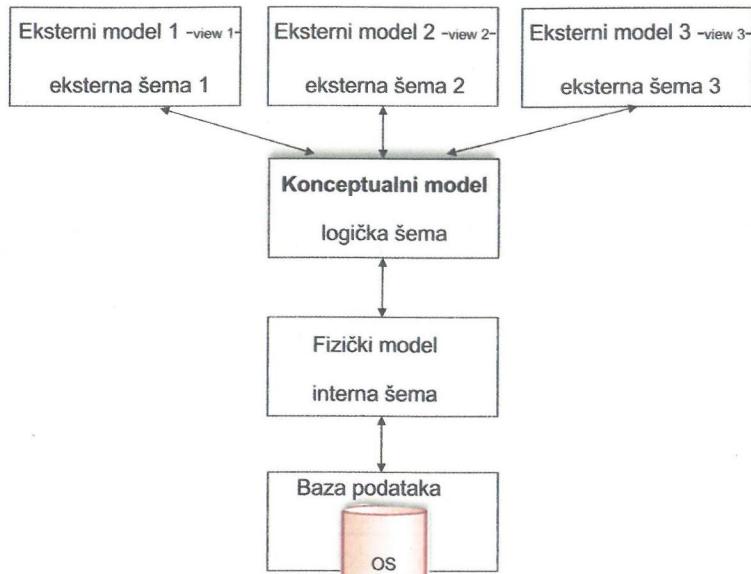
4.4. Relacioni model

Model je reprezentacija skupa entiteta (objekata) i njihovih međusobnih veza. Izbor i definisanje entiteta i veza između njih je suština procesa modeliranja.

Za sada možemo reći da postupak definisanja relacija – tabela i veza među njima znači kreiranje modela relacione baze podataka. Prije nego što definišemo relacioni model, upoznaćemo se sa modelima baza podataka uopšte, potom sa elementima i uslovima kreiranja relacionog modela.

4.4.1. Modeli baze podataka

Prilikom objašnjenja arhitekture baze objasnili smo da se na konceptualnom nivou definiše konceptualni model sa odgovarajućom logičkom šemom koja sadrži opis svih entiteta i veza, atributa, domena i integritetska ograničenja. Hiperarhijsko mjesto konceptualnog modela vidi se na slici 4.5 (između korisnika i fizičkog nivoa).



Slika 4.5. Hiperarhija modela

Eksterni modeli (1–3) nastaju na osnovu konceptualnog modela i predstavljaju samo pogled (view) koji je nastao prema potrebama korisnika.

Na slici 4.5. može se primijetiti da fizički model omogućava predstavljanje konceptualnog modela u formi zapisa podataka na neki fizički medij (najčešće je to disk, a posrednik između medija i fizičkog modela je operativni sistem – OS).

Kad se govori o modelima baze podataka, najčešće se misli na konceptualni model, pa ćemo mi ubuduće kad kažemo model smatrati da je to konceptualni model (a kad to ne bude slučaj, to ćemo posebno naglasiti).

Definisanje logičke šeme, odnosno pravljenje konceptualnog modela (uopšte, pa i kod baza podataka), podrazumijeva tri koraka:

- detekciju i selekciju objekata (entiteta) i veza između objekata,
- imenovanje objekata, njihovih atributa i veza,
- klasifikaciju, gdje se objekti svrstavaju u klase i tipove.

4.4.2. Vrste modela baza podataka

Model baze podataka je **formalni sistem** koji se sastoji od:

- skupa objekata – osnovnih elemenata (koncepta) baze podataka,
- skupa operacija koje se provode nad objektima,
- skupa integritetskih ograničenja (*integrity constraints*) kojima se definiše skup važećih pravila.

Razlikujemo nekoliko vrsta modela. Mi ćemo se baviti relacionim modelom (što je i naslov ove (4.4) sekcije), a ostale modele, koji su uglavnom istorijski, samo ćemo informativo obraditi.

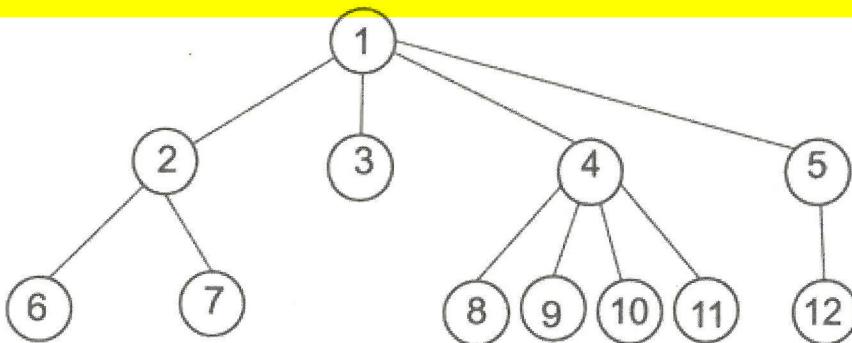
4.4.2.1. Ravni (tabelarni) model

Ravni model poznat i kao model ravnih datoteka (*flat files*) jeste model gdje se svi podaci čuvaju u samo jednoj tabeli. Glavni nedostatak jeste velika redundancija, a primjer koji bi odgovarao ovom modelu bila bi Excel tabela.

Može se smatrati pretečom relacionog modela samo zbog forme tabele kao načina organizacije podataka.

4.4.2.2. Hiperarhijski model podataka

Hiperarhijski model je najstariji model. Zasnovan je na hiperarhijskoj strukturi koja ima oblik stabla prikazan na slici 4.6. Kod hiperarhijskog modela podaci su organizovani kao čvorovi.



Slika 4.6. Hiperarhijski model predstavljen mrežnim stablom

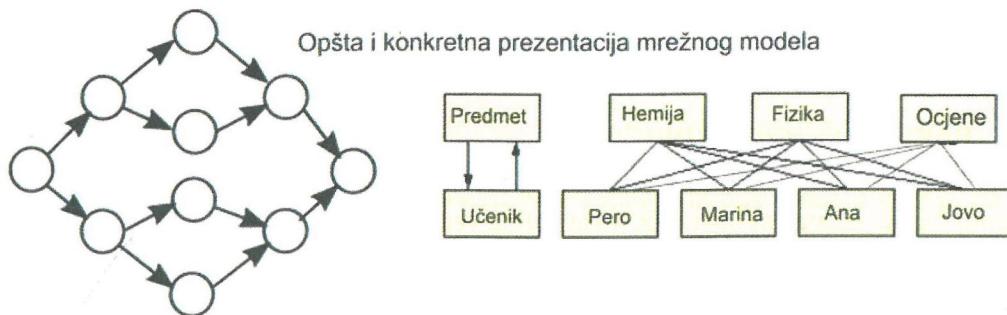
Hiperarhijska struktura odgovara stablu i postoje različiti matematički opisi i procedure pretrage mrežnog stabla⁶. Element na jednoj strani relacije *jedan-prema-više* naziva se element višeg nivoa ili roditelj, koji je povezan sa elementima nižeg nivoa ili djecom. Hiperarhijska baza podataka se predstavlja kao stablo aplikacionih elemenata, gdje relacija *jedan-prema-više* povezuje svakog roditelja i dijete. Pretraga se obavlja od osnovnog sloga (*root*) i kreće prema naniže (ako se ne pronađe, vraća se unazad i ide na iduće dijete itd.)

4.4.2.3. Mrežni model podataka

Mrežni model nastao je kao paralela hiperarhijskom modelu, ali kod njega je dopušteno da svaki slog ima više od jednog roditelja. Moglo bi se reći da je hiperarhijski model specijalni slučaj mrežnog modela.

⁶ Teorija grafova je grana matematike koja se bavi proučavanjem grafova. Graf je vrsta strukture podataka koji se sastoji od skupa čvorova i skupa grana, koje predstavljaju odnose (veze) između čvorova. Mrežno stablo je najjednostavnija klasa grafa. U računarstvu se grafovi često koriste za opis modela ili struktura podataka.

I mrežni model se može predstaviti mrežnim grafom (kako ovdje ne postoji obavezan odnos roditelj–dijete, ne govori se o stablu, već o mrežnom grafu). Kod mrežnih struktura moguće je da se svaki element povezuje sa svakim. Podaci u mrežnom modelu se prikazuju skupovima slogova, a njihovi odnosi pomoću veza. Na slici 4.7. data je ilustracija mrežnog modela predstavljena mrežnim grafom i modelom jedne konkretnе baze.



Slika 4.7. Mrežni graf kojim se može predstaviti mrežna struktura podataka

Relacioni model podataka

Ovaj model je trenutno najrasprostranjeniji i on je predmet našeg interesovanja.

4.4.3. Pojam relacije i relacione baze

Osnivač relacionog modela Kod⁷ je osnovne pojmove relacionog modela preuzeo iz matematičke teorije.

Relacija, kao osnovni koncept relacionog modela je matematička relacija⁸.

Relacija je imenovani podskup Dekartovog (Kartezijskog) proizvoda dva ili više domena. Kombinacije vrijednosti domena nazivaju se n-torce.

Pojednostavljenje objašnjenje Dekartovog proizvoda daćemo u sekciji o relacionoj algebri (sekcija 4.5), a za više detalja o ovoj problematiki preporučujemo knjigu Gordane Pavlović Lažetić, *Uvod u relacione baze podataka*, iz koje ćemo dati nešto precizniju definiciju relacije:

Neka su D_1, D_2, \dots, D_n ($n \equiv N$) domeni (ne obavezno različiti). Dekartov proizvod tih domena, u oznaci $D_1 \times D_2 \times \dots \times D_n$, jeste skup n-torki (v_1, v_2, \dots, v_n) takvih da je $v_i \in D_i$ za sve $i=1, 2, \dots, n$. Relacija R stepena n, definisana na domenima D_1, D_2, \dots, D_n , je proizvoljni konačni podskup navedenog Dekartovog proizvoda.

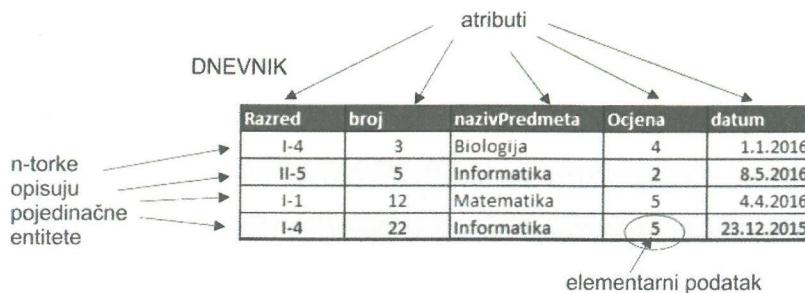
Prema tome, **relacija je skup n-torki**.

Relaciji bi u standardnoj računarskoj terminologiji odgovarala jedna datoteka, a n-torci jedan slog te datoteke.

Relacija ima jednostavnu reprezentaciju u obliku dvodimenzionalne tabele sa podacima, prikazanu na slici 4.8.

⁷ Edgar Kod, engleski matematičar i naučnik. Više detalja vidi u 4.9.

⁸ Često se može sresti pogrešna tvrdnja da se relacioni model tako zove zbog veza (*relationships*); Relacioni model je dobio ime po relacijama na kojima se model zasniva.



Slika 4.8. Tabela kao reprezentacija relacije

Napokon možemo da definišemo relacionu bazu: **Relaciona baza podataka je skup vremenski promjenljivih relacija (n-torki u relacijama).**

Sistemi relacionih baza podataka imaju sljedeće odlike:

- Svi podaci se predstavljaju preko **relacija** (*relation*) u obliku dvodimenzionalne tabele sa podacima. Osim atributa svakog entiteta, model podataka mora da definiše i **veze** ili odnose koje postoje između entiteta. **Veza** ili **odnos** (*relationship*) je udruživanje – asocijaciju između entiteta.
- Sve vrijednosti su **skalarne** (vidi 4.6.5. 1NF).
- Sve operacije obavljaju se nad cijelom relacijom, a rezultat je takođe cijela relacija. Taj koncept je poznat kao **cjelovitost** (*closure*).

Skup operacija koje se provode na relacijama, odnosno tabelama naziva se relacionom algebrom. U sekciji 4.5 upoznaćemo se sa osnovnim pojmovima iz relacione algebre i računa.

4.4.3.1. Šema relacije: Relaciona tabela i Šema relacione baze

Šema relacije predstavlja opis strukture tabele. Relaciona šema se još naziva i relaciona tabela, a nju karakterišu sljedeća svojstva:

- U relacionoj šemi može postojati samo jedan tip slogova, odnosno n-torki;
- Svaki red, odnosno slog ili n-torka (*tuple*) uključuje tačno određen broj polja podataka, tj. atributa i svaki od njih je eksplicitno imenovan;
- Relaciona šema ne sadrži dva jednakana naziva atributa, odnosno relacija ne sadrži dvije jednakake kolone;
- **Redoslijed kolona, odnosno atributa je nebitan;**
- **Redoslijed n-torki je nebitan;**

Relacija nije uređena. Relaciju možete zamisliti kao košaru u kojoj su torke nagomilane bez ikakvog posebnog redoslijeda. **Redni brojevi zapisa**, uobičajen mehanizam za pristupanje zapisima u nerelacionim bazama podataka, **ne postoji u relacijama**;

- Nove se tabele mogu stvarati povezivanjem preko vrijednosti polja podataka iz istog domena iz dvije postojeće tabele.

Naglasimo: relacija i tabela nisu isto. Kod tabela je bitan redoslijed redova i kolona, dok kod relacije nije bitan redoslijed atributa i n-torki.

Šemu relacije možemo shvatiti dvojako: kao definiciju strukture neke datoteke (kako smo naveli u definiciji), ali i kao predstavu svojstava klase objekata i veza unutar nekog sistema.

Šema relacione baze podataka je skup šema njenih relacija.

4.4.3.2. Predstavljanje relacija i šema relacija

Relaciju možemo predstaviti tako da navedemo njeno **ime i skup atributa** koji je čine:

R(A₁:D₁, A₂:D₂, ..., A_n:D_n), ako domeni nisu bitni dovoljno je navesti samo attribute:
R(A₁, A₂, ..., A_n).

Primjer: Učenik(Razred, BrojDnevnik, Ime, Prezime)

Uobičajena konvencija za predstavljanje relacione šeme je **dijagram relacije**. Relaciju predstavljamo pravougaonom tabelom koja ima onoliko ćelija koliko je atributa u relaciji.

Ime relacije se ispisuje iznad tabele, a imena atributa u ćelijama⁹.

Kao primjer, na slici 4.9 daćemo dijagram relacione šeme baze podataka Generacija2016.

L_Podaci								
JMBG	Odjeljenje	Br_dnevnik	Prezime	Ime	Dat_rod	Pol	Adresa	Grad
Odjeljenja								
Odjeljenje	Smjer							

Slika 4.9. Primjer dijagrama relacija

4.4.4. Osnovne osobine koje treba da ima relacioni model

SUBP (DBMS) se naziva relacionim ako podržava relacione operacije (u idućoj sekciji 4.5. objasnićemo relacione operacije). Osim što podržava relacione operacije, da bi neki model mogao da se nazove relacionim, on treba da ima osnovne osobine kao što su:

- Sistemski tretman *NULL* vrijednosti.
 - *NULL* je univerzalna vrijednost *nepoznato*. Koristi se kad iz bilo kog razloga u momentu unosa podataka nije poznata vrijednost nekog ulaznog podatka. *NULL* može biti vrijednost svake kolone bez obzira na njen tip, **osim primarnog ključa** i u slučaju kad se izričito zahtijeva da vrijednosti u nekoj koloni ne smiju biti nedefinisane (dodatno ograničenje **NOTNULL**).
 - Neprekidan pristup dinamičkom katalogu relacionog modela.
 - Katalog relationalnog modela predstavlja zapis o strukturi baze podataka. Spada u tzv. metapodatke, koje smo spomenuli u uvodu. Zapisan je u istoj formi (relacije-tabele) kao i sami podaci. Pravo pristupa ovom katalogu imaju samo ovlašćeni korisnici – administratori. Naziva se dinamički, jer definicije podataka koje se mogu vidjeti odgovaraju upravo tekućem stanju u bazi podataka.
 - Fizička nezavisnost znači da korisničke aplikacije ostaju neizmijenjene kada se promijeni fizička organizacija baze ili fizički metod pristupa podacima (vidi Arhitektura baze podataka). Praktično, fizička organizacija podataka ima uticaj samo na performanse (brzinu pristupa podacima, odziv sistema).
 - Logička nezavisnost.
- Ekvivalentno prethodnom uslovu, samo u obrnutom smjeru.
- Integritetska nezavisnost:
 - Integritet entiteta definiše da je primarni ključ jedinstven na nivou cijele relacije.
 - Integritet domena određuje skup dozvoljenih vrijednosti kolone.
 - Relacioni integritet.
- Svaki atribut u relaciji vezuje se za određeni domen, pa povezivanjem atributa ne smije da se naruši (analiziraj kasnije kardinalnost).

⁹ Pri čemu se atribut koji je primarni ključ podvlači, a spoljni ključevi pišu italicikom, a šta to znači vidi 4.4.5.