Univerza v Ljubljani

Fakulteta za družbene vede

INFORMATIKA (baze podatkov) ZA VISOKOŠOLSKI ŠTUDIJ DRUŽBOSLOVNE INFORMATIKE

pripravila: Nataša Kejžar

1 Baze podatkov

Podatki so del našega vsakdana. V raznih organizacijah (banke, trgovine, bolnišnice ...) so nepogrešljivi za normalno delo. S pomočjo podatkov ljudje ustvarjajo informacije. To so nova spoznanja, ki jih izluščimo neposredno iz podatkov. Če torej povzamemo:

- Podatki so dejstva, ki so predstavljena na nek splošno uveljavljen način in imajo pomen v določenem kontekstu (npr. dnevno temperaturo predstavimo s številko in mero, v kateri merimo stopinje – 15°C, naklon klanca predstavimo s številko in mero (stopinjami), ki ni enaka meri za temperaturo – 15°klanec).
- Informacije so nova spoznanja, oziroma nov pomen, ki ga človek pripiše podatkom s pomočjo svojega znanja.

Vsak dan se v organizacijah pojavi na milijone novih podatkov. Postavi se novo vprašanje: "Kako zagotoviti, da bodo vsi pomembni podatki spravljeni in dostopni vsakemu uslužbencu, ki jih potrebuje, in to v kar se da kratkem času?".

Z razvojem informacijske in komunikacijske tehnologije je dostop do množice relevantnih podatkov postal mogoč. Samo uporaba informacijskih tehnologij pa ni dovolj za učinkovito uporabo podatkov. Podatki morajo biti predhodno ustrezno organizirani in vzdrževani, da iz njih znamo izluščiti željene informacije in da lahko delamo potrebne analize.

Organizacija, ki želi učinkovito uporabljati podatke, mora postati informacijsko usmerjena [1]. Taka organizacija:

- uporablja podatke,
- z njimi upravlja (jih shranjuje, vzdržuje, ščiti) in
- se informacijsko obnaša.

Informacijsko obnašanje pomeni, da:

- shranjuje celovite podatke – torej podatke, ki so točni in nepristrani,
- uporablja podatke iz formalnih virov (če se le da), saj s tem poveča kakovost in zanesljivost podatkov
- s podatki upravlja transparentno.

Tranparentnost upravljanja pomeni, da si podatkov ne lasti noben del organizacije in zato ni potrebe po njihovem podvajanju (npr. če bi si kadrovski oddelek lastil podatke o zaposlenih, v računovodstvu ne bi imeli na razpolago teh podatkov in bi jih morali sami zbirati). S tem postane potvarjanje dejstev nemogoče, prav tako pa ni nekonsistentnosti v podatkih. Pri podvojenih podatkih namreč prihaja do vprašanja, ali se (pri zahtevi) podatki spremenijo na *vseh* mestih, kjer so shranjeni (npr. sprememba naslova zaposlenega v kadrovski službi in računovodstvu).

Organizacija, ki s podatki uspešno upravlja, doseže, da so podatki dostopni takrat, ko se jih potrebuje, in v obliki, v kakršni se jih potrebuje. To pomeni, da ljudje nadzorujejo kakovost podatkov, identificirajo nove zanimive vire podatkov in načrtujejo zaščito podatkov.

Najboljša rešitev za shranjevanje podatkov v skupnem zbirališču je podatkovna baza. To je celovita računalniško podprta zbirka podatkov. Če imamo več podatkovnih baz v podjetju, potem je dobro, da so le-te med sabo integrirane.

Primer integracije podatkov [1]:

- Spremembo naslova kupca, ki jo zajamemo v računovodsko podatkovno bazo, posredujemo takoj v trženjsko podatkovno bazo.
- Iz podatkov o prodaji moramo vsaj enkrat mesečno izračunati skupno prodajo in ta sumarni podatek posredovati bazo, ki se uporablja za napoved prodaje.

Celovite podatkovne rešitve (programska orodja za kreacijo in vzdrževanje podatkovne baze) ponuja veliko svetovno znanih podjetij. Naj naštejemo nekatera: Oracle, SAP, SSE ipd.

2 Načrtovanje podatkovne baze

Podatkovna baza (PB) je opis dela realnega sveta. Vse spremembe v svetu (ki ga PB zajema), se torej morajo odražati v PB. PB je neodvisna od uporabniških programov, ki jo bodo uporabljali.

Življenjski cikel podatkov, ki bi lahko bili zanimivi za določeno organizacijo, je tak:

- zaznava podatkov, ki bi bili lahko zanimivi
- zbiranje podatkov
- organiziranje in obdelava podatkov
- vzdrževanje podatkov (posodobitve, preprečiti ponovno zbiranje že vnešenih podatkov)

V tretji točki (organiziranje in obdelava podatkov) se v organizaciji odloči, kje se bodo podatki shranjevali in kdo bo te podatke uporabljal. Ta del spada pod načrtovanje podatkovne baze.

Potrebno je narediti načrt, kako podatke vključiti v proces in kako bodo podatki v bazi sploh zapisani.

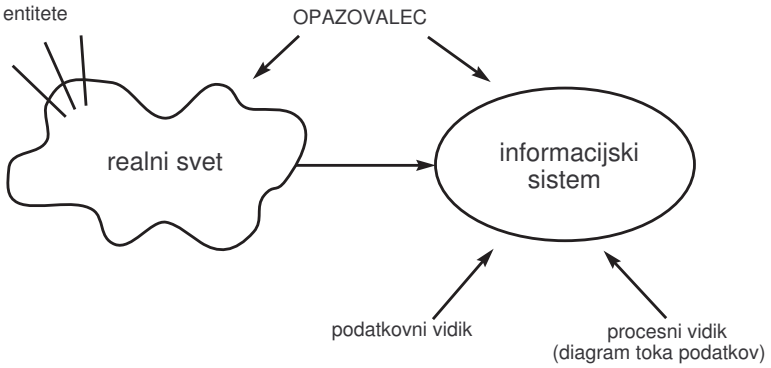
2.1 Analiza uporabnikovih potreb, ER model

Preden se lotimo kreacije podatkovne baze, je potrebno PB načrtovati. Prvi korak je izvedeti kar največ informacij o problemskem področju in o uporabnikovih potrebah. Izvedeti je potrebno stvari o podatkih, ki naj bi bili v bazi, o povezavah med njimi, o drugih odnosih (pravila, omejitve),... To zvemo iz različnih virov:

- intervjujev in neformalnih pogovorov z uporabniki
- zbiranja in pregleda dokumentacije (trenutni dokumenti za hranjenje podatkov, načrt dosedanjega vira in zbirališča podatkov)

Ker so besedni opisi problemskega področja največkrat premalo natančni, uporabljamo tehnike grafičnih predstavitev (formalnejši opis). Grafično lahko predstavimo celotno področje iz

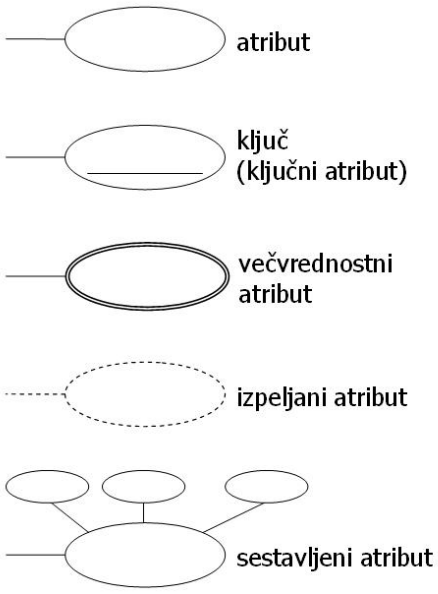
- procesnega vidika – narišemo Data Flow Diagram (oz. diagram toka podatkov); rezultat so programi
- podatkovnega vidika – narišemo Entity Relationship Model (ER model); rezultat je baza podatkov



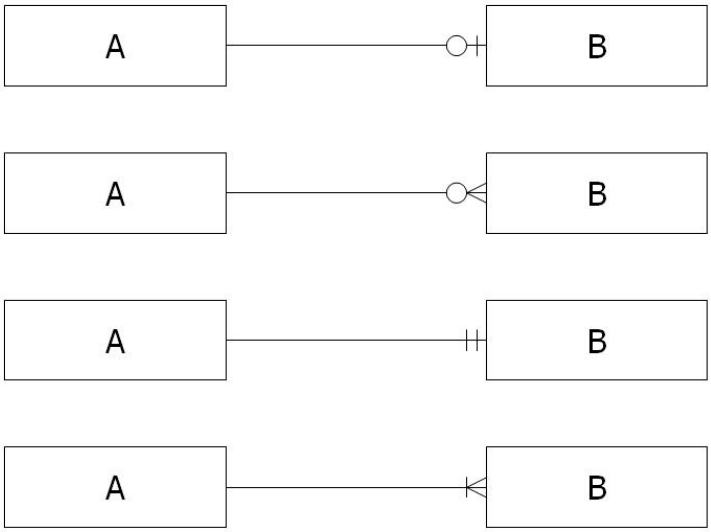
ER model je po slovensko model entitet-povezav. Predstavlja nam konceptualni načrt baze podatkov.

Terminologija – iz opisa realnega stanja:

- **entiteta** je predmet, dogodek, ki ga obravnavamo (ki ga želimo imeti v bazi). Opazovalec si ustvari določeno podobo sveta. Entiteta prestavlja najmanjši del poobe sveta, ki ga lahko ločimo od drugh delov in poimenujemo. Katere stvari ločujemo, je odvisno od tega, kaj nas v danem trenutku zanima.
- **atribut** je lastnost entitete. Ta ima lahko eno ali več vrednosti (za posamezno entiteto). Tistim z več vrednostmi pravimo *večvrednostni atributi*. Če atribut lahko razdelimo na manjše dele (ki so še vedno informativni - recimo naslov v ulico, hišno številko in kraj), ga imenujemo *sestavljeni atribut*. *Izpeljani atribut* je tisti, ki ga lahko izračunamo/izpeljemo iz ostalih atributov.
  Atribut lahko zavzame vrednoti iz določene domene - npr. ocena je število med 1 in 10, poštna številka v Sloveniji ima vrednosti med številkami 1000 in 5999 ipd.
- **entitetni tip** je vzorec entitete z enakimi (podobnimi) atributi. Grafični prikaz – s pravokotnikom.



- **kandidat za ključ** je atribut entitetnega tipa, ki enolično razlikuje posamezne entitete med seboj (največkrat je umetno sestavljena šifra - recimo študentska vpisna številka). Ključ je lahko sestavljen iz več atributov, ki skupaj enolično opredelijo posamezno entiteto (*sestavljeni ključ*).
- **povezava** med entitetami
- **kardinalnost povezave** oziroma številčnost povezave; spodnja slika prikazuje različne možnosti za povezave:
  1. entiteta iz entitetnega tipa A ni nujno povezana z neko entiteto iz B (spodnja meja je 0 – krogec), če pa je, je največ z eno entiteto iz B (zgornja meja je 1 – črtica)
  2. entiteta iz A ni nujno povezana z entiteto iz B, lahko pa je tudi z več entitetami iz B (zgornja meja je mnogo – grabljice)
  3. entiteta iz A mora biti v vsakem trenutku povezana z vsaj 1 entiteto iz B (spodnja in zgornja meja je 1 – leva in desna črtica)
  4. entiteta iz A mora biti v vsakem trenutku povezana z vsaj eno entiteto iz B, lahko pa je tudi z več kot 1.



Pri povezavah med entitetami gre za *semantične* povezave (povezave imajo določen pomen). Pomen povezav včasih pripišemo ob povezavi sami. Primer:

2.1.1 Kratke vaje ER modelov

1. V slovenskih občinah zbirajo podatke o stalno živečih prebivalcih.
2. Trgovsko podjetje – stranke naročajo različne izdelke.
3. Zdravljenje: Vsak zdravnik ima svoje paciente. Le-ti obiskujejo zdravnika, ki jim na koncu izda račun.
4. Evidenca koncertov v CD. Za vsak koncert je potrebno vedeti (poleg imena), v kakšno zvrst spada in kdaj je bil opravljen.

2.1.2 Primer 1: Kinematografsko podjetje

Podjetje, ki se ukvarja s kinematografsko dejavnostjo bi rado spremljalo obiskanost filmov, ki jih odkupuje. Svoje kinematografe imajo na več različnih lokacijah. Od tega v nekaterih kinih predvajajo lahko po več filmov naenkrat, spet drugi pa so taki, ki imajo le eno filmsko platno. Kinematografi podjetja imajo različno velike filmske dvorane (od največjih, ki sprejmejo do 540 gledalcev, do najmanjše s 40 gledalci). Podjetje ima enake cene za

vstopnice vseh filmov, ki jih v danem trenutku predvaja. To pomaga napolniti tudi manjše kinodvorane, ki bi sicer večkrat ostajale prazne. V večini objektov deluje tudi prodajalna hitrih prigrizkov in osvežilnih pijač, ki pa jo večinoma oddajajo tujim najemnikom. Ti plačujejo podjetju mesečno najemnino.

Podjetje največkrat odkupuje ameriške - holivudske - filme, saj ti prinašajo največ denarja. Kljub temu pa so tudi z nekaj neodvisnimi filmi dosegli velik dobiček. Podjetje zato zanima, kakšne so karakteristike filmov, ki prinašajo največ denarja.

2.1.3 Primer 2: Založnik

Manjši knjižni založnik bi rad informatiziral pregled nad svojo dejavnostjo. Ukvarja se z izdajanjem strokovne literature. To so zaenkrat samo knjige, v daljni prihodnosti pa bi rad svojo dejavnost razširil tudi na izdajanje strokovnih revij. Knjige, ki jih založnikovo podjetje izda, so shranjene v njegovih 2 skladiščih (ki se nahajata v drugih predelih mesta) toliko časa, dokler jih ne prevzamejo naročniki (vse večje knjigarne v državi). Naročniki lahko prevzamejo blago takoj, ko ga plačajo. Največkrat se takoj odpeljejo do skladišča, pokažejo plačan račun (tega potem skladiščnik preveri s telefonskim klicem v podjetje) in naložijo knjige. Založnik bi rad spremljal plačilo računov in izdajo blaga prek računalnika. Poleg tega bi imel rad tudi možnost, da si zabeleži (on ali pa skladiščnik) morebitno število knjig, ki jih mora glede na povpraševanje nanovo natisniti.

Po nekaterih izdanih strokovnih knjigah (večinoma so to delovni zvezki in šolski učbeniki) je stalno povpraševanje, zato knjige tudi večkrat natisnejo. Posebna informacija pri novem natisu je, ali so knjigo vmes izpopolnili. To namreč pomeni, da se vsebina nove knjige razlikuje od starih, zato jih skladiščnik ne sme enačiti.

2.1.4 Primer 3: Trgovina na debelo

V trgovini prodajajo izdelke, ki imajo različne cene. Cene izdelkov se s časom spreminjajo. Nekateri izdelki imajo popust, ki je odvisen od nabavljene količine (popust velja za število kupljenih izdelkov v nekem intervalu). Trgovina dobiva izdelke od dobavitelja s pomočjo dobav. Dobavitelj za izstavljeno blago pošlje račun. Trgovina prodaja izdelke naročnikom, ki z naročilom izrazijo potrebe po željenih izdelkih.

2.1.5 Primer 4: Stanovanjsko podjetje

Stanovanjsko podjetje upravlja s stavbami. V stavbah so lahko stanovanja ali poslovni prostori. Oboji imajo lahko več lastnikov, vendar le enega najemnika. Najemnik mora za najem stanovanja podpisati najemno pogodbo. Stanovanja in poslovni prostori so točkovani s točkami, ki so odvisne od občine.

Najemnik ima lahko le eno 2 pogodbi – eno za poslovni prostor in eno za stanovanje.

2.1.6 Primer 5: Študij

Študent, ki je vpisan samo na eno smer, posluša več predmetov, ki pa se lahko predavajo na več smereh. Predmet lahko predava več različnih predavateljev (nosilci ali ne), vsak predavatelj lahko prav tako predava več različnih predmetov. Končen cilj študenta je opraviti izpit pri določenem predmetu.

2.2 Podatkovni modeli

Ko imamo narejeno dokaj natančno podobo o svetu (v našem primeru s pomočjo ER modela), pride na vrsto izbira podatkovnega modela, s pomočjo katerega bomo našo podobo pretvorili v računalniško zbirko podatkov. Podatkovne modele ločimo glede na možnost organizacije (strukturiranja) podatkov.

Podatkovni model je torej množica pravil, ki določajo, kako smejo biti podatki v PB strukturirani (npr. relacijski podatkovni model predpisuje, da so podatki shranjeni v tabelah, kjer so stolpci atributi, vrstice pa predstavljajo posamezne entitete (in povezave)).

Poznamo več podatkovnih modelov (urejeni so po času nastanka):

- hierarhični (drevesna struktura, podvajanje zapisov pri M:N relacijah)
- mrežni (temelji na grafih, precej zapleten)
- relacijski
- večdimenzionalni (je lahko realiziran z relacijskim, uporaben za analitične uporabnike, vsebuje dimenzije in mere)
- objektni model podatkov in procesov (vsak objekt vsebuje atribute in metode, ki se lahko izvršujejo nad njim).

3 Pretvorba v relacijsko shemo

Relacijska podatkovna baza je ena izmed možnih realizacij podatkovnih baz. Uporabniku se kaže kot množica tabel. Vsaka tabela je sestavljena iz dveh delov:

- čelna vrstica (= relacijska shema)
- podatkovne vrstice (= relacija)

Primer relacijske baze podatkov:

```
Student
------------------------
VpisnaSt Ime    Priimek
------------------------
001      Janez  Novak
002      Marko  Bitenc
003      Ana    Kobal
------------------------
Predmet
--------------------------
SifraP  NazivPredmeta
--------------------------
P1      Statistika
P2      Informatika
P3      Racunalnistvo
P4      Multivariatna
--------------------------
Izpit
-------------------------------
VpisnaSt SifraP Datum     Ocena
-------------------------------
001      P3     12.2.04   9
002      P1     1.2.04    10
002      P4     11.2.04   8
-------------------------------
```

Vrstni red stolpcev in vrstic ni pomemben, pomembno pa je, da se vrstice med seboj razlikujejo v vrednosti vsaj enega atributa. Relacija je v splošnem podmnožica kartezičnega produkta domen:

$$R \subseteq D_1 \times D_2 \times \ldots \times D_n$$

Na primeru:

Janez	Novak
Marija	Kovač

$\subseteq$

Janez	Novak
Janez	Kovač
Marija	Novak
Marija	Kovač

Relacijska shema je torej logični načrt baze podatkov, ki je osnova za relacijski model baz podatkov.

V relacijski shemi bi zgornjo podatkovno bazo zapisali s 3 tabelami (Student, Predmet in Izpit) takole:

```
Student(VpisnaSt, Ime, Priimek)
Predmet(SifraP, NazivPredmeta)
Izpit(VpisnaSt, SifraP, Datum, Ocena)
```

Pri relacijski shemi nas zanima le struktura relacije in ne konkretni podatki. Do strukture lahko pridemo na različne načine.

3.1 Normalne forme

Prvi način:

ER model pretvorimo v relacijsko shemo tako, da poskušamo vsaki entiteti prirediti svojo relacijo. Dobimo nenormalizirano shemo. Le-to prek normalizacijskih pravil spravimo v normalizirano relacijsko shemo. Poleg normalizacijskih pravil, je dobro upoštevati še dodatna navodila:

- semantika naj bo jasna (že na pogled naj bo jasno, kaj kakšna relacija pomeni)
- podvajanje podatkov naj bo čimmanjše
- v atributih naj bo čimmanj vrednosti NULL (nedefiniranih vrednosti)

NULL vrednosti so nedefinirane vrednosti. Primer: V relaciji Izdelek so podatki o izdelkih. Vsake nekaj časa je na kakšnega izmed izdelkov tudi popust. Če bi v Izdelek dodali še atribut Popust, bi večina izdelkov tu imela NULL (nedefinirano vrednost). Zato je v tem primeru pametneje narediti dodatno relacijo Popust(StIzdelka, Popust).

1. Shema je v I. normalni formi (1NF), če se vsakemu objektu priredi največ ena vrednost iz domene. V shemi tako ne sme biti ponavljajočih grup (večvrednostnih atributov). Za take atribute naredimo novo relacijo.
   Primer relacije, ki ni v I. normalni formi (podatki o knjižnih izdajah):

   ```
   Knjiga(StKnjige, Avtor, Naslov, LetoIzdaje)
   ```

   V 1NF jo spravimo takole

   ```
   Knjiga(StKnjige, Avtor, Naslov)
   Izdaja(StKnjige, LetoIzdaje)
   ```

2. Shema je v II. normalni formi (2NF), če so vsi atributi polno funkcionalno odvisni od ključa. Atribut B je funkcijsko odvisen od atributa A, če vsakem trenutku k vrednosti atributa A pripada največ ena z njo povezana vrednost atributa B.

Primer relacije, ki ni v 2NF:

```
Student(VpisnaSt, StSrSole, Ime, Priimek, KoncanaSrednjaSola,
NaslovSrSole, TipSrSole)
```

V 2NF jo spravimo takole

```
Student(VpisnaSt, Ime, Priimek, StSrSole)
SrSola(StSrSole, KoncanaSrednjaSola, NaslovSrSole, TipSrSole)
```

3. Shema je v III. normalni formi, če ni tranzitivnih odvisnosti. To so funkcionalne odvisnosti med neključnimi atributi.
   Primer relacije, ki ni v 3NF:

   ```
   Delavec(SifraD, Ime, Priimek, RojstniKraj, Drzava)
   ```

   V 3NF jo spravimo takole

   ```
   Delavec(SifraD, Ime, Priimek, StKraja)
   RojstniKraj(StKraja, RojstniKraj, Drzava)
   ```

   Ime in Priimek določata RojstniKraj, le-ta pa državo. V tem primeru smo morali dodati tudi nov atribut StKraja (glavni ključ v relaciji RojstniKraj). Za glavni ključ bi lahko izbrali tudi RojstniKraj, vendar ni rečeno, da se dva kraja ne imenujeta povsem enako. V takem primeru ta atribut ne bi več enolično razločeval dveh krajev med sabo.

Podatke iz 2 ali več relacij vsebinsko povezujemo s skupnimi atributi (glavni ključ – tuj ključ).

Od sedaj naprej bomo vedno zahtevali, da je podatkovna baza v tretji normalni obliki. To ponavadi upoštevamo že pri predhodnem načrtovanju.

3.1.1 Naloge

1. V kateri normalni obliki je zapis? Po potrebi pretvori v 3NF.

   ```
   Delavec(StDel, Ime, StPodj, StOtroka, ImeOtroka)
   ```

2. ```
   Student(StStud, Ime, RojDat, Letnik)
   StudentSmer(StStud, StSmeri, ImeSmeri)
   ```

   Predpostavimo, da je študent lahko vpisan na več smeri.

3. ```
   Dobava(StDobavitelja, StArtikla, Kolicina, ImeDob, NaslovDob, ImeArt)
   ```

4. `Dobavitelj(StDob, ImeDob, NaslovDob, Kraj, Razdalja)`

   Razdalja je razdalja od nas do dobavitelja.

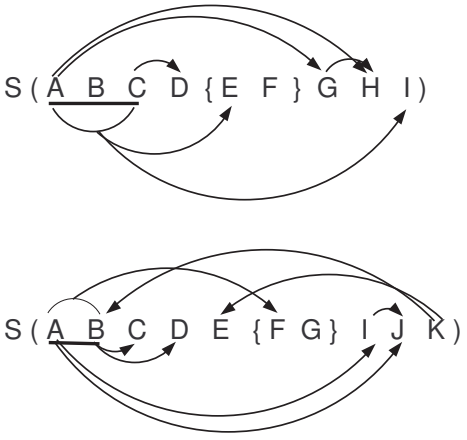
   Problemi pri shemah, ki niso v 3NF: Napake in redundanca pri vnašanju, ažuriranju, brisanju. Primer: Ko izbrišemo edinega dobavitelja iz nekega kraja, izgbimo tudi podatek o oddaljenosti do tega kraja. Pri naslednjem dodajanju dobavitelja iz tega kraja, moramo ponovno izračunati (poznati) oddaljenost do tega kraja. Poleg tega, če se oddaljenost do kraja spremeni (nova cesta), je to potrebno popraviti na veliko mestih in ne samo na enem.

5. Napišite konceptualno shemo v 3NF za primer zdravljenja. Pacient se določenega dne ob določenem času odpravi do zdravnika. Zdravnik opravi določeno vrsto pregleda, od katerega je tudi odvisna cena. Le-ta pa je tudi odvisna od trajanja obiska. Predpostavimo, da ima vsak pacient svojega izbranega zdravnika.

6. Relacija Film opisuje predvajanje filmov v kinematografih. Določi ključ, ugotovi v kateri normalni obliki je zapis. Po potrebi pretvori v 3NF. Omejitve: Vsak film ima samo enega režiserja, vsak igralec v filmu igra samo 1 vlogo.

   `Predstava(Film, Reziser, Igralec, Vloga)`

7. Normaliziraj.



3.2 Direktna prevedba ER modela z upoštevanjem normalizacije

Drugi način za prevedbo ER modela v relacijsko shemo:

1. Vsak entitetni tip je svoja relacija.
2. Večvrednostni atributi postanejo samostojen entitetni tip.
3. Ključ entitetnega tipa postane glavni ključ relacije.

4. Povezave mnogo:mnogo (M:N) pretvorimo v relacije. Glavni ključ te relacije je sestavljen iz glavnih ključev obeh entitetnih tipov, ki jih povezava povezuje. Če ima povezava lastne atribute, so to neključni atributi relacije.

5. Povezave 1:N – če ima lastne atribute, jih pretvorimo v relacije (enako kot pri M:N).

6. Če povezava 1:N nima lastnih atributov, jo realiziramo s tujim ključem: ključni atribut entitetnega tipa na strani 1 dodamo kot tuj ključ v relacijo, ki pripada entitetnemu tipu na strani N. S tujim ključem tako vzpostavimo povezavo med relacijama.

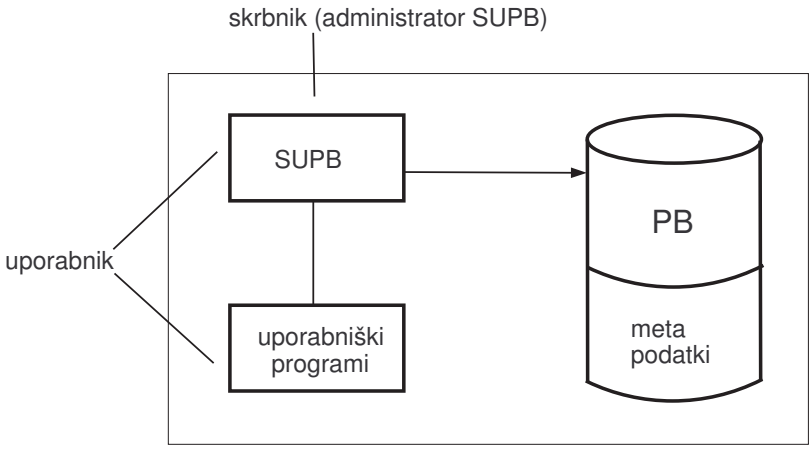
Podpora razvoju PB so programska orodja CASE (Computer Aided Software Engineering). Ti omogočajo po večini vse – od gradnje ER modela, do fizične podatkovne baze. Boljša orodja omogočajo tudi avtomatsko pretvorbo med modeli.

4 Kreiranje računalniške baze podatkov

Podatkovno bazo najlažje skreiramo s pomočjo sistema za upravljanje podatkovnih baz (SUPB). Le-ta omogoča:

- tvorbo PB
- uporabo PB (vnos, spreminjanje, ažuriranje podatkov, brisanje, poizvedovanje)
- vzdrževanje PB

Podatkovne baze, do katerih dostopa več ljudi (tudi sočasno), ponavadi upravlja in vzdržuje skrbnik podatkovne baze (DataBase Administrator – DBA).



Le-ta s pomočjo SUPB skrbi za naslednje operacije:

- izvajanje zaščit

- spremembe na strukturi podatkov
- dela zahtevnejše poizvedbe
- skrbi za optimizacijo PB
- skrbi za čiščenje baze (stari, neuporabni podatki)

Do velikih podatkovnih baz lahko uporabniki dostopajo na 2 načina:

- z vnaprej pripravljenimi programi (uporabniški programi, vnosni obrazci)
- neposredno prek SUPB

4.1 MS Access

MS Access je SUPB. Omogoča nam:

- vnašanje podatkov (tvorba baze)
- ažuriranje, brisanje, poizvedovanje (doseganje podatkov)
- povezovanje podatkov (pomembno v vlogi manjšega podvajanja podatkov, večje ažurnosti podatkov)
- vzdrževanje (zaščita, čiščenje, varovanje podatkov)

MS Access je prirejen za relacijski podatkovni model, ki je trenutno najbolj uporabljani podatkovni model (podatkovni model so pravila, kako naj bodo podatki v bazi strukturirani).

Relacijski model je sestavljen iz relacij (dvodimenzionalnih tabel). Ena vrstica posamezne tabele predstavlja eno entiteto (en primerek te relacije). Stolpci pa predstavljajo atribute – lastnosti, ki jih posamezna entiteta ima.
TABELA:

	atribut 1	atribut 2	atribut 3	...	atribut n
entiteta 1					
entiteta 2					
entiteta 3					
...					
entiteta m					

Terminologija:

- tabela ali relacija (entitetni tip)
- stolpec ali polje (atribut)
- vrstica ali podatkovni zapis (entiteta)

Program MS Access je del MS Office-a in tako zgrajen na podoben način (orodne vrstice, vrstica ikonc). Operacije, ki jih izvajamo na trenutno aktivni podatkovni bazi so dosegljive prek okenca, ki se nam odpre v sredini z naslovom "ime baze":Database. S pomočjo tega okenca lahko kreiramo:

- tabele
- poizvedbe = povpraševanja (SQL, QBE); angl. queries
- obrazce (za vnos novih podatkov v bazo ...)
- poročila (za lažji pregled, tiskanje podatkov)
- internetne strani, makroje, module (Visual Basic)

4.1.1 Tabela – relacija

Preden začnemo s tvorjenjem tabele (po možnosti iz prej zgrajene relacijske sheme), je potrebno vedeti še nekaj stvari. Še pred kreacijo je dobro poznati vse:

- omejitve območja, oziroma dopustne vrednosti za atribute(*domene*)
- vedeti, ali obstaja na atributih kakšna referenčna celovitost; ali je domena atributa vezana na domeno ustreznega ključa (glej poglavje Povezovanje tabel)
- ali obstaja kakšna eksplicitna celovitost/dodatna omejitev (npr. pogoj za napredovanje v 3. letnik so opravljeni vsi izpiti iz 1. letnika in 7 izpitov iz 2. letnika)
- ali bomo atribut indeksirali; *indeksi* imajo vlogo stvarnega kazala; poizvedovanje bo postalo hitrejše, potrebno pa je paziti na 2 stvari:
  - indeksi zavzamejo dodaten prostor na disku
  - vnos podatkov v tabelo postane zahtevnejši, saj vsak vnos povzroči tudi spremembo pri indeksih

Postopki za kreiranje tabele v *Design View*-u (Tables>New>Design View).

V vrstice vpišemo atribute, ki jih ima tabela, ki jo želimo kreirati (z vsemi glavnimi in tujimi ključi – torej vse atribute, ki jih ima tabela, ko je pretvorjena v relacijsko shemo).

Izberemo zaloge vrednosti oziroma **tipe podatkov**:

- alfanumerični/tekstovni Text, Memo
- številski Number, Currency
- datum Date/Time
- ostalo OLE Object (za slike ...)

V zavihku (spodaj) *General* lahko za vsako polje določimo še dodatne lastnosti in kontrole pri vnosu:

- velikost
- format
- število decimalnih mest (za številske atribute)
- vhodno obliko (input mask):

&	obvezna črka ali presledek
C	neobvezna črka ali presledek
<	male črke
>	velike črke
0	prostor za številko (številka ali 0)
9	prostor za številko (številka ali prazen prostor)

- naslov (caption)
- privzeta vrednost (default value)
- omejitve (Validation Rule)
- tekst, ki se izpiše na monitorju, če se omejitev krši (Validation Text)
- ali je vrednost atributa obvezna (Required)
- ali je atribut indeksiran (Indexed) (glavni ključi in atributi, po katerih se veliko poizveduje)

Omejitve atributov se pišejo v taki obliki kot pri dodatnih omejitvah pri SQL poizvedbah (primeri):

- atribut mora biti pozitiven: > 0
- atribut se mora začeti na črko "K" ali končati na črko "B": Like "K*" Or Like "*B" (za tekstovne/alfanumerične podatke se vedno uporabljajo *navednice*!)
- atribut mora imeti vrednosti med 1.1.2005 in 1.2.2006: >= #1.1.05# And <=#1.2.06# (v primeru angleških nastavitev pa: >= #1/1/05# And <=#2/1/06#)
- atribut ne sme imeti vrednosti ali pa mora imeti vrednost zavrnjen: Is Null Or "zavrnjen"
- atribut ne sme imeti vrednosti pred današnjim dnem: >= Now()

Glavni ključ

Glavni ključ izberemo tako, da v orodni vrstici poiščemo ikonco ključ ali pa z desnim klikom z miško kliknemo na področje pred imenom atributa (tako odpremo meni) in izberemo Primary Key.

Za kreiranje sestavljenega glavnega ključa s pomočjo <Ctrl> + levi klik z miško na polja za sestavljeni ključ najprej označimo ta polja. Potem z desnim klikom z miško odpremo meni in izberemo Primary Key.

Ko zapremo okno, nas program vpraša za ime datoteke (tabele), ki se nato shrani v našo bazo podatkov.

NASVET: Priporočljivo je kreiranje vseh tabel in povezav med njimi in šele kasneje vnašanje dejanskih podatkov vanje.

Primer 1: tabela Študent

Polje	**Omejitev**
vpisna_stevilka	enolično določena, 8-mestna, ključ
ime	
priimek	
datum_vpisa	od leta 1990 do danes
letnik	vrednosti 1 do 4
izredni	vrednosti DA/NE
ocena_mature	od 9 do 34

Primer 2: tabela Delavec

Polje	**Omejitev**
sifra	enolično določena, 5-mestna, ključ
ime	
priimek	
datum_zaposlitve	do danes
placa	od 70.000 naprej
sifra_oddelka	2-mestna

4.1.2 Povezovanje tabel

Ikona **Relationships** oziroma Tools > Relationships...

Iz nabora tabel, ki smo jih kreirali, izberemo tabele, ki jih želimo povezati (z gumbom Add). Povezujemo glavni ključ ene tabele s pripadajočim tujim ključem druge tabele. Tuji ključi niso posebej označeni. Pazimo: sestavljeni glavni ključi imajo večkrat tudi vlogo tujega ključa! Imena glavnih in tujih ključev v tabelah, ki jih želimo povezavti, niso nujno enaka (odvisno je od tega, kako smo si stvar prej zamislili)!

Z levim gumbom na miški primemo glavni ključ in ga prenesemo do pripadajočega tujega ključa v drugi tabeli, kjer ga tudi spustimo. Izberemo še Enforce Referential Integrity.

Referenčna celovitost:

Prepove spremembe na podatkih, ki bi naredile povezave med tabelami neveljavne. Nek atribut zavzame le vrednosti, ki jih ima ustrezni ključ.

Primer: Imamo tabeli lastnik in pes. En lastnik ima lahko enega ali več psov, pes pa mora imeti enega in ima največ enega lastnika. Če ima entiteta Janko psa Flokija, potem mora biti v zapisu Flokija atribut – tuj ključ z isto vrednostjo, ki jo ima glavni ključ v zapisu Janka. Ta povezuje Janka s Flokijem. Referenčna celovitost nam prepove spreminjanje glavnega ključa v zapisu Janka, saj bi to pomenilo, da Janko ni več lastnik Flokija (tuji ključ pri Flokiju bi namreč še vedno imel tisto staro vrednost) in Floki kar naenkrat ne bi imel več lastnika. To pa je prepovedano stanje te baze podatkov.

Referenčna celovitost zato omogoča dve dodatni možnosti:

- cascade update: če posodobimo polje glavnega ključa, se posodobijo tudi vsa polja tujih ključev, ki se vežejo na ta glavni ključ
- cascade delete: če izbrišemo podatkovni zapis z določenim glavnim ključem, se izbrišejo tudi vsi podatkovni zapisi (v drugih tabelah), katerih vrednost pripadajočega tujega ključa je enaka izbrisanem glavnem ključu

Pazljivost pri izbiri teh dodatnih možnosti ni odveč!

5 Poizvedovanje

Do podatkov dostopamo na različne načine (z relacijsko algebro, z nepostopkovnim jezikom SQL, na grafični način - QBE). Rezultati poizvedb so vedno nove relacije! Spoznali bomo drugi in tretji način kreiranja poizvedb.

SQL in QBE spadata v 4. generacijo računalniških jezikov. Generacije jezikov si sledijo takole:

- 1LG (1st language generation) – strojni jezik
- 2LG – zbirnik (assembler)
- 3LG – visokonivojski jeziki (Pascal, Fortran, Cobol, C, ...)
- 4LG – SPSS, SQL, Excel ipd.

Jeziki četrte generacije so lahko:

- enostavni povpraševalni jeziki (QBE)
- kompleksni povpraševalni jeziki (SQL)
- generatorji poročil
- grafični jeziki (za risanje diagramov)
- jeziki za podporo odločanju
- aplikacijski generatorji

- specifikacijski jeziki (orodja CASE)
- visokonivojski matematični jeziki (Mathematica, Matlab, Derive, R)
- preglednice (Excel)
- razvojni sistemi

Načela jezikov 4. generacije so:

- minimalni vložek
- minimalna poklicna izobrazba
- minimalno učenje sintakse in mnemonike
- minimalni čas izdelave aplikacije
- enostavno vzdrževanje
- maksimalni rezultati

Osnovne operacije nad relacijami (glej podpoglavje SQL).

- projekcija (PROJECT)
  - izločimo določene stolpce iz tabele
  - vertikalna podmnožica relacije
  - rezultat mora biti relacija, zato je potrebno še izločanje duplikatov
- restrikcija, izbira (SELECT)
  - izločimo vrstice, ki ustrezajo nekemu pogoju
  - horizontalna podmnožica relacije
  - rezultat je vedno relacija
- stik (JOIN)
  - rezultat je nova relacija, ki jo dobimo tako, da eni od relacij pripišemo še atribute druge relacije – določiti moramo še atribut ali množico atributov, ki se mora ujemati v obeh relacijah

5.1 SQL poizvedbe – Structured Query Language

SQL je neproceduralni jezik. Povemo le KAJ želimo in ne točno, kako naj se stvar izvrši. SQL je namenjen kreiranju:

1. kreiranju poizvedb
2. kreiranju baze podatkov (del DDL – data definition language)
3. popravljanju definicije podatkovne baze (del DML – data manipulation language).

SQL jezik za poizvedbe je sestavljen iz naslednjih besed:

```
SELECT <seznam atributov>
FROM <seznam tabel, v katerih se nahajajo zgoraj izbrani atributi>
WHERE <dodatne omejitve, povezovalni pogoji med tabelami>
```

Rezultat SELECT stavka je vedno relacija (nova tabela). SELECT naredimo projekcijo, z WHERE naredimo izbiro in stik.

Primer relacijske sheme:

```
Podjetje(StPod, Ime, Kraj)
Delavec(StDel, Naziv, Poklic, OD, StPod)
Projekt(StProj, Ime, Sredstva)
DelProj(StDel, StProj, Funkcija)
```

Narišite povezave med tabelami.

5.1.1 Projekcija

Izpiši poklice po podjetjih.

```
SELECT StPod, Poklic
FROM Delavec
```

Za rezultat lahko dobimo ponovitve poklicev, zato je potrebno še izločiti duplikate:

```
SELECT DISTINCT StPod, Poklic
FROM Delavec
```

Izpiši imena vseh delavcev in njihovih podjetij urejene po imenih delavcev.

```
SELECT DISTINCT Ime, StPod
FROM Delavec
ORDER BY Ime
```

5.1.2 Izbira

Izpiši tisti del relacije Delavec, ki se nanaša na tajnice.

```
SELECT *
FROM Delavec
WHERE Poklic='tajnica'
```

...samo tistih z OD višjim od 200000 (kombinacija pogojev).

```
SELECT *
FROM Delavec
WHERE Poklic='tajnica' AND OD>200000
```

Operatorji: =; <>; >; >=; <; <=; AND; OR; NOT; LIKE; IN; IS NULL; IS NOT NULL

Poišči imena delavcev iz Podjetja 3 ali 4.

```
SELECT Ime
FROM Delavec
WHERE StPod=3 OR StPod=4
```

...ali s pripadnostjo množici:

```
SELECT Ime
FROM Delavec
WHERE StPod IN (3,4)
```

Vprašanja nad več relacijami.

Poišči imena delavcev iz ljubljanskih podjetij.

```
SELECT Delavec.Ime
FROM Delavec, Podjetje
WHERE Delavec.StPod = Podjetje.StPod AND Kraj = 'Ljubljana'
```

Ker se "Ime" nahaja v dveh relacijah (Podjetje in Delavec), moramo eksplicitno napisati, iz katere relacije jemljemo atribut s tem imenom (npr. Delavec.Ime).

Ali drugače:

```
SELECT Delavec.Ime
FROM Delavec, Podjetje
WHERE Delavec.StPod = Podjetje.StPod
  AND StPod IN (SELECT StPod
                FROM Podjetje
                WHERE Kraj = 'Ljubljana')
```

5.1.3 Stik

Poišči imena delavcev in kraje njihovih podjetij.

```
SELECT Delavec.Ime, Podjetje.Kraj
FROM Delavec, Podjetje
WHERE Delavec.StPod = Podjetje.StPod
```

Poišči imena tajnic iz Ljubljane in imena njihovih podjetij.

```
SELECT Delavec.Ime, Podjetje.Ime
FROM Delavec, Podjetje
WHERE Delavec.Poklic = 'tajnica'
  AND Delavec.StPod = Podjetje.StPod
  AND Podjetje.Kraj='Ljubljana'
```

Grupiranje pomeni združevanje vrstic v skupine na osnovi enake vrednosti nekega atributa. Za grupiranje uporabljamo GROUP BY (ga dodamo na koncu WHERE dela (oziroma na koncu ORDER BY, če še sortiramo)). V tem delu naštejemo *vsa* polja, po katerih bomo grupirali. Polja, katerega vrednost želimo izraziti z grupirno/agregatno funkcijo (avg, sum, count, max, min...) ne napišemo. V SELECT delu temu polju na levi strani pripišemo grupirno funkcijo (npr. AVG(placa)).

Izpiši povprečni OD tajnic.

```
SELECT AVG(OD)
FROM Delavec
WHERE Poklic='tajnica'
```

Koliko različnih funkcij ima delavec Kovac na svojih projektih?

```
SELECT COUNT(DISTINCT Funkcija)
FROM DelProj, Delavec
WHERE Delavec.Ime = 'Kovac'
  AND Delavec.StDel =Del_Proj.StDel
```

Izpiši število delavcev po posameznih podjetjih.

```
SELECT Podjetje.Ime, COUNT(StDel)
FROM Podjetje, Delavec
WHERE Delavec.StPod = Podjetje.StPod
GROUP BY Podjetje.Ime
```

Uporabili smo Podjetje.Ime, ker sta v relacijah Delavec in Podjetje dva atributa z nazivom Ime.

Za pogoj na grupirni funkciji (npr. izpiši učence, katerih povprečna ocena je višja od 3), uporabimo HAVING, ki ga napišemo za GROUP BY. Primer uporabe GROUP BY in HAVING: Izpiši imena, priimke in povprečne ocene učencev, katerih povprečna ocena je višja od 3.

```
SELECT ime, priimek, AVG(ocena) AS [povprecna ocena]
FROM ucenec, predmet
ORDER BY priimek, ime ASC
GROUP BY ime, priimek
HAVING AVG(ocena) > 3;
```

SQL del v MS Accessu dosežemo takole: odpremo novo poizvedbo izberemo SQL View (najbolj leva ikonca oziroma View > SQL View).

5.1.4 Vaje za poizvedbe SQL

Baza podatkov PODJETJE:

Baza podatkov Delavci je sestavljena na podlagi naslednjih pravil: Vsak delavec pripada enemu in samo enemu oddelku, vsak oddelek pa zaposluje enega ali več delavcev. Delavci delajo na nič ali več projektih, prav tako pa je na projekte razporejenih nič ali več delavcev.

NASVET: Če ime atributa vsebuje presledek ali kak drug ne-alfanumerični znak, ime atributa obdamo z oglatimi oklepaji (npr. delavci.[ime delavca]).

1. Izpiši vse projekte.
2. Imena in priimki vseh delavcev, ki delajo v marketingu.
3. Delavci (imena in priimki), ki so se zaposlili pred letom 2000.
4. Delavci, ki trenutno delajo na projektih.
5. Število delavcev, ki trenutno delajo na projektih.
6. Delavci s povprečno oceno vsaj 9.
7. Izpiši stimulacijo za delavce. Delo na projektu je vrednoteno + 2000 SIT.
8. Izpiši 10% dodatek za delavce s povprečno oceno vsaj 9.

Baza podatkov KINO:

1. Izpiši imena vseh kinodvoran.
2. Izpiši vse dvorane, ki sprejmejo več kot 200 gledalcev.
3. Izpiši vse umetnike, njihovo vlogo in filme, ki so jih soustvarjali.
4. Izpiši število dvoran, ki so predvajale filme od 8.4.2005 naprej.
5. V koliko filmih (ki so jih vrteli) je nastopal Robert De Niro? (Kaj pa ostali?)
6. Izpiši povprečno in skupno število sedežev po kinih. Izpiši tudi naziv kina.
7. Izračunaj število nezasedenih sedežev na predstavo. Izračunaj število nezasedenih sedežev po posameznih filmih.

8. Izpiši tiste predstave in dvorane, kjer je bilo prodanih več vstopnic kot je kapaciteta dvorane.

Delavci in Izobrazbe.

```
Delavec
-------------------------------------
ImeD   LetoRoj   DelM           OD
-------------------------------------
Ana    55        vodja razvoja  150
Ivo    51        hisnik         110
Jure   51        VD             170
Metka  45        snazilka       100
Tine   60        praktik        90
-------------------------------------
Izobrazba
------------------------
ImeD      Izobrazba
------------------------
Ana       FER
Ana       elektrotehnik
Ivo       osnovna sola
Jure      FDV
Metka     osnovna sola
Tine      ekonomist
------------------------
```

1. Za vsakega delavca izpiši njegovo delovno mesto.
2. Za vsakega delavca izpiši njegov OD in izobrazbo.

Izpit in Koledar.

```
---------------------------
Student   Predmet   Ocena
---------------------------
S1        P1        7
S1        P2        8
S1        P3        7
S2        P2        9
S2        P4        10
S3        P2        6
---------------------------
Koledar
-----------------------------
Predmet  Datum      Student
-----------------------------
```

```
P1       10.1.2005  S1
P1       10.1.2005  S2
P1       30.1.2005  S3
P2       2.2.2005   S1
P4       3.2.2005   S2
----------------------------
```

Kaj vrne naslednje povpraševanje?

```
SELECT Izpit.Student, Koledar.Predmet, Ocena, Datum
FROM Izpit, Koledar
WHERE Izpit.Student = Koledar.Student
  AND Izpit.Predmet = Koledar.Predmet
```

Tabele za dobavitelje (D), za izdelke (I) in dobave (DI).

D(StD, ImeD, Status, Kraj) I(StI, ImeI, Barva, Teza) DI(StD, StI, Kolicina)

```
D
----------------------------
StD   ImeD     Status Kraj
----------------------------
D1    Vema     20     MB
D2    Emona    10     LJ
D3    Nama     30     LJ
----------------------------
I
----------------------------
StI   ImeI    Barva   Teza
----------------------------
I1    obleka  rdeca   0.5
I2    obleka  zelena  1.5
I3    suknjic modra   0.75
I4    hlace   rdeca   0.3
----------------------------
DI
-----------------------
StD   StI    Kolicina
-----------------------
D1    I1     300
D1    I2     200
D1    I3     400
D2    I1     300
D2    I2     400
D3    I2     200
-----------------------
```

1. Poišči številke vseh izdelkov, ki so v dobavi. (brez ponavljanj)

2. Želimo vse podatke o dobaviteljih.
3. Poišči številke dobaviteljev iz MB, ki imajo status večji od 20.
4. Izpiši številke in statuse dobaviteljev iz LJ po padajočem statusu.
5. Za vsak dobavljeni izdelek poišči številko izdelka in ime kraja dobavitelja.
6. Poišči imena dobaviteljev, ki dobavljajo I2. (na 2 različna načina)
7. Poišči imena dobaviteljev s statusom, ki je manjši od tekočega maksimuma v tabeli dobaviteljev.
8. Poišči številke dobaviteljev, ki dobavljajo vsaj en izdelek dobavitelja D2.
9. Poišči imena dobaviteljev, ki ne dobavljajo izdelka I1.
10. Poišči imena dobaviteljev, ki dobavljajo vse izdelke. (imena dobaviteljev, za katere ne obstaja izdelek, ki ga ne dobavljajo) NOT EXISTS (podoben operator kot IN)
11. Poišči številke izdelkov, ki so težki več kot 0.5 kg ali pa jih trenutno dobavlja dobavitelj D2. (uporabi operator UNION)
12. Za vse izdelke izpiši njihove številke in njihovo težo v gramih.
13. Poišči številke izdelkov, ki jih dobavlja več kot 1 dobavitelj.

FIS tekme – primer velike datoteke

V datoteki FIS so shranjeni podakti o alpskih smučarjih in njihovih FIS točkah. Pomembna polja v datoteki FIS:

```
FISCODE
GENDER (M/W)
SURNAME
NAME
NAT
BIRTHDATE (za vse je datum 1.1., pomembna je samo letnica)
DHPOINTS (DownHill points, \v stevilka 9999 pomeni, da to\v ck \v se nima)
SLPOINTS (SLalom points)
GSPOINTS (Grand Slalom points)
SGPOINTS (Super G points)
```

V tabeli t_moski so shranjeni podatki o moški slalomski FIS tekmi, v tabeli t_zenske pa podatki o ženski slalomski FIS tekmi v Kranjski Gori, 15.3.2002. V tabeli so naslednja polja:

```
KODA (FIS koda tekmovalca)
ST_STEV (\v startna \v stevilka na tekmi)
GRUPA (jakostna skupina na tekmi)
REZULTAT1 (\v cas prvega teka)
REZULTAT2 (\v cas drugega teka)
TOTAL (skupni \v cas)
TOCKE (to\v cke na tekmi, ki se izra\v cunajo glede na zaostanek za zmagovalcem)
```

1. Koliko moških in koliko žensk je v tabeli FIS?
2. Koliko Slovenk in Slovencev je v tabeli FIS?
3. Za slovenske tekmovalce izpiši njihov priimek, ime, letnico rojstva in slalomske točke. (sortiraj jih pos spolu in nato še po slalomskih točkah)
4. Izpiši vrstni red pri moških/ženskah po FIS točkah v slalomu.
5. Izpiši samo tiste, ki imajo manj kot 100 točk.
6. Izpiši koliko tekmovalcev imajo posamezne države v tabeli FIS. Izpis naj bo urejen po padajočem številu tekmovalcev.
7. Izpiši ime, priimek, letnik rojstva in državo tekmovalk in jih uredi po skupnem številu točk v vseh 4 disciplinah.
8. Izpiši rezultate tekme v Kranjski Gori (priimek, ime, država, čas 1.teka, čas 2.teka, skupni čas in točke) – urejene po mestih.
9. Izpiši število nastopajočih po državah na tej tekmi.
10. Izpiši povprečno število točk tekme po posameznih državah (upoštevaj le uvrščene tekmovalce – tekmovalce s točkami).
11. Izpiši imena tekmovalcev, ki se pojavijo vsaj 2x. Uredi jih po številu pojavitev danega imena.
12. Izpiši najmlajše tekmovalce.

Kreiranje relacij

```
CREATE TABLE Podjetje (StPod INTEGER CONSTRAINT stpodpk PRIMARY KEY,
                       Ime VARCHAR (36),
                       Kraj VARCHAR (20))
```

Brisanje zapisa (vrstice)

Briši delavca s številko 15.

```
DELETE Delavec
WHERE StDel = 15
```

Brisanje vseh zapisov

```
DELETE Delavec
```

Ažuriranje tabele

Delavcem na projektu 36 povečaj osebni dohodek za faktor 1.2.

```
UPDATE Delavec
SET OD = OD * 1.2
WHERE StDel IN (SELECT StDel
                FROM DelProj
                WHERE StProj = 36)
```

Vnos novega zapisa

Vnesi podjetje številka 25 – Valjarno Jesenice.

```
INSERT INTO Podjetje
  (25, "Valjarna", "Jesenice")
```

Naloge (primer dobaviteljev, izdelkov in dobav):

- Ažuriranje zapisov:
  1. Spremeni barvo izdelka I2 v rumeno , povečaj njegovo težo za 0.2 in ime na neznano.
  2. Podvoji vrednost statusa dobaviteljev iz LJ.
  3. Za vse dobavitelje iz LJ postavi količine na 0.
- Ažuriranje več relacij (oziroma tabel):
  1. Zamenjaj številko dobavitelja iz D2 v D9.
- Vstavljanje posamezne vrstice:
  1. Dodaj izdelek I5 z imenom cevlji, (rdece barve) in tezo 0.2 kg.
- Brisanje več vrstic iz več tabel:
  1. Izbriši vse dobave dobaviteljev iz Ljubljane in prav tako dobavitelje same.

5.2 QBE – Query By Example

V MS Accessovem okencu, ki nam omogoča različne operacije nad bazo podatkov z dvoklikom izberemo možnost *Queries* - Poizvedbe. Odpreta se nam 2 novi okni. V prvem izberemo tabele, iz katerih želimo poizvedovati. Ko smo to naredili, prvo okno zapremo in v drugem naklikamo željeno poizvedbo.

V zgornjem delu drugega okna imamo sedaj tabele, ki smo jih izbrali. Preverimo, če so med seboj *povezane* (sicer jih moramo najprej povezati, oziroma izbrati dodatne tabele, ki bodo posredno povezale trenutne tabele). V spodnjem delu imamo tabelo. Vanjo vpišemo željeno poizvedbo. Stolpci nam predstavljajo atribute, ki jih želimo v rezultatu. V vrsticah pa lahko določamo:

- polje
- tabelo (katere polje smo izbrali)
- Total – vrstica, ki nam omogoča določiti grupiranje po posameznih poljih (npr. povprečna plača delavcev po oddelkih). Dosežemo jo z ikonco $\sum$ ali v meniju View > Totals
- Crosstab – le v primeru, ko želimo križno poizvedbo (posebnost MS Office)
- Sort – sortiranje

- Criteria – dodatni kriterij, po katerih izbiramo entitete. Tu izhodno relacijo še dodatno omejimo.

Dodatne omejitve: Vpisujemo lahko funkcije in različne operatorje (podobno kot pri omejitvah polj, ko tabelo kreiramo). Uporabljamo torej: And, Or, Like, Not, In, Is Null, <, >, "", +, -, *, /, div, mod.

Ko želimo v poizvedbi kaj izračunati za vsako entiteto, potem naredimo *izraz – Expression*. V ime polja zapišemo:

```
"ime izraza" : izraz (npr. cena * kolicina)
```

Pomemben izraz, ki se uporablja v MS Accessu je *Iif* izraz: **Iif(pogoj; stavek 1; stavek 2)**, ki je enak stavku v Pascalu: **if** pogoj **then** stavek 1 **else** stavek 2

5.2.1 Vaje za poizvedbe QBE

NASVET: Uporabljajte toliko tabel, kot jih morate (tabele morajo biti med seboj povezane). Dodatne tabele lahko dodatno omejijo rezultat, kar seveda ni pravilno!

Baza podatkov PODJETJE:

Baza podatkov Delavci je sestavljena na podlagi naslednjih pravil: Vsak delavec pripada enemu in samo enemu oddelku, vsak oddelek pa zaposluje enega ali več delavcev. Delavci delajo na nič ali več projektih, prav tako pa je na projekte razporejenih nič ali več delavcev.

NASVET: Če ime atributa vsebuje presledek ali kak drug ne-alfanumerični znak, ime atributa obdamo z oglatimi oklepaji (npr. delavci.[ime delavca]).

1. Delavci, katerih priimek se začne na črko B.
2. Imena in priimki vseh delavcev naj bodo urejeni po naraščajoči višini plače.
3. Delavci, ki so se zaposlili pred letom 1999.
4. Povprečna plača delavcev po oddelkih.
5. Frekvenčna porazdelitev delavcev po oddelkih.
6. Izračunajte 10% stimulacijo k plači za delavce iz oddelka proizvodnja.

Baza podatkov KINO:

1. Razvrsti dvorane po padajočem številu sedežev.
2. Izpiši najemnike in kinodvorane, katerim se pogodba konča po 1.1.2006.
3. Izpiši vse umetnike, ki imajo črko R v svojem priimku.
4. Izpiši vse igralce, ki imajo črko R v svojem priimku.
5. Izpiši vse predstave v Kinu Ena in jih uredi po času začetka predvajanja.
6. Koliko dvoran ima posamezen kino?

7. Izpiši povprečno in skupno število sedežev po kinih. Izpiši tudi naziv kina.
8. V koliko filmih, ki jih je kinematografsko podjetje odkupilo, so sodelovali režiserji v bazi?
9. Izračunaj število nezasedenih sedežev na predstavo.
10. Izpiši skupno število nezasedenih sedežev.

Studenti, Predmeti in Izpiti.

Student

SifraS	ImeS	Letnik	Kraj
S1	Miha	2	LJ
S2	Tone	3	CE
S3	Ana	1	KR
S4	Peter	4	LJ
S5	Nada	3	KR

Predmet

SifraP	ImeP	Semester	Ure
P1	MA	3	2
P2	FI	2	4
P3	RAC	1	3
P4	STAT	4	3
P5	EKON	7	2
P6	ELEK	3	3

Izpit

SifraS	SifraP	Ocena
S1	P1	7
S1	P2	6
S1	P3	8
S1	P6	9
S1	P4	10
S2	P1	7
S2	P4	6
S3	P2	8
S3	P3	8
S4	P1	9
S4	P6	10
S4	P5	8

```
S5     P6     6
-------------------------
```

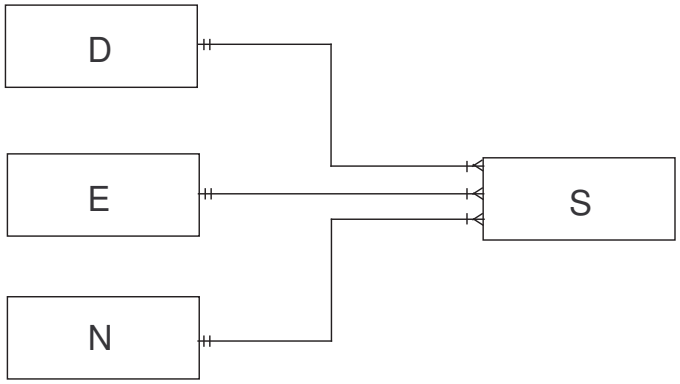
1. Poišči kode vseh predmetov, pri katerih so se opravljali izpiti.
2. Izpiši kode vseh študentov iz Ljubljane.
3. Izpiši kode vseh študentov iz Ljubljane, ki hodijo v 4.letnik.
4. Izpiši kode študentov, ki so opravili izpit pri predmetu P2.
5. Izpiši imena študentov, ki so opravili izpit pri predmetu P2.
6. Izpiši imena vseh študentov, ki so opravili vsaj 1 izpit pri predmetih, ki se predavajo v 2.semestru.
7. Za vsak predmet poišči njegovo kodo in letnike študentov, ki so ta izpit opravljali.
8. Izračunaj povprečno oceno vseh opravljenih izpitov.
9. Koliko študentov je opravilo izpit pri predmetu P1?
10. Izračunaj povprečno oceno vsakega študenta. (izpiši še ime študenta)
11. Izpiši imena študentov, ki so opravili več kot 2 izpita.

Dobavitelji, Elementi, Naprave in povezava med njimi

```
Dobavitelji
-----------------------------
SD   ImeD    Stat   MestoD
-----------------------------
D1   Tine    20     Kr
D2   Jure    10     Ce
D3   Ana     30     Ce
D4   Ivo     20     Kr
D5   Joze    30     Mb
-----------------------------
Elementi
-----------------------------
SE  ImeE     Barva    Cena
-----------------------------
E1  NAND     rdeca    3
E2  NOR      zelena   4
E3  JK       modra    10
E4  JK       rdeca    10
E5  RS       modra    15
E6  INV      rdeca    3
-----------------------------
Naprave
```

```
----------------------------
SN  ImeN       MestoN
----------------------------
N1  Terminal   Ce
N2  Mag. trak  Lj
N3  Disk       Mb
N4  CPU        Mb
N5  Tiskalnik  Ce
N6  Modem      Kp
N7  Tipkovnica Kr
----------------------------
Potrebni elementi za naprave in njihovi dobavitelji
----------------------------
SD   SE   SN    Kolic
----------------------------
D1   E1   N1    20
D1   E1   N4    7
D2   E3   N1    4
D2   E3   N2    3
D2   E3   N3    5
D2   E3   N4    8
D2   E3   N5    9
D2   E3   N6    6
D2   E3   N7    7
D2   E5   N2    3
D3   E3   N1    2
D3   E4   N2    1
D4   E6   N3    4
D4   E6   N7    5
D5   E2   N2    2
D5   E2   N4    3
D5   E5   N5    4
D5   E5   N7    4
D5   E6   N2    5
----------------------------
```

1. Najdi vse podatke o napravah, ki se delajo v Kranju.
2. Poišči številke dobaviteljev, ki dobavljajo elemente za napravo N1.
3. Poišči številke dobaviteljev, ki dobavljajo element E1 za napravo N1.
4. Poišči številke dobaviteljev, ki dobavljajo rdeče elemente za N1.
5. Poišči številke dobaviteljev, ki dobavljajo elemente za naprave, sestavljene v Kranju ali Celju.
6. Poišči številke elementov, ki so potrebni za naprave iz Kranja.



7. Poišči številke elementov, ki se dobavljajo katerikoli napravi v Kranju in ki jih dobavljajo dobavitelji iz Kranja.
8. Poišči številke elementov, katerih dobavitelji in naprave so iz istega kraja.
9. Poišči skupno dobavljeno količino v tabeli S.
10. Poišči skupno dobavljeno količino v tabeli S po posameznih dobaviteljih.
11. Poišči skupno dobavljeno količino posameznih elementov v tabeli S po posameznih dobaviteljih.

6 Uporaba podatkov v organizaciji

Pri kreiranju podatkovne baze – predvsem pri kreiranju uporabniških vmesnikov nas zanimajo naslednji podatki:

- ali se bodo v PB uporabljali tudi zunanji podatkovni viri
- kako bo organizirana zaščita podatkov
- kakšno bo ravnanje s podatki – kdaj bodo podatki postali zgodovinski (od kdaj naprej jih bomo hranili le še v sumarni obliki)
- kakšne vrste uporabnikov bo PB imela

Zunanji podatkovni viri:
Pri izmenjavi podatkovnih virov med organizacijami se dandanes najbolj uporablja računalniško podprta izmenjava. V preteklosti so se uveljavili standardi EDI, ki so določali enotno opredeljene strukture za posamezne vrste dokumentov. Sedaj se vedno bolj uveljavlja standard XML (EXtensible Markup Language). Ta omogoča označevanje podatkov s pomočjo *značk* in izmenjavo prek interneta. Npr.

```
<delavec> Janez Novak </delavec>
```

S tem smo povedali, da je poslan podatek Janez Novak, ki spada med podatke z oznako (značko) "delavec".

Zaščita podatkov:
Pri zaščiti podatkovne baze ločimo več dejavnosti:

- ščitenje pred tem, da bi uporabnik *nenamerno* pripeljal podatkovno bazo v nekonsistentno stanje: nadzor celovitosti (opredelitev podatkov, dopustne vrednosti, pogoji in pravice uporabe – podatki o podatkih (*metapodatki*))
- zaščita pred *namernimi* poskusi škodovanja (varnost PB)
- odprava posledic nesreč
  - obnavljanje PB
  - nadgradnja PB

Življenjska doba podatkov:
Podatke se shrajuje v podatkovno bazo v *operativni obliki*. V tej (najpodrobnejši) obliki pa jih ne potrebujemo vedno (npr. podatki o nakupih strank v supermarketu na določen dan v letu 2005 so aktualni le kakšno leto, kasneje nas zanima le še, kakšna je bila prodaja določenih izdelkov po mesecih v letu 2005). Ko takih podatkov ne potrebujemo več, se podatke agregira (spravi v *sumarno obliko* podatkov). Ponavadi se jih ne shranjuje več v najhitreje dostopni del PB. Shrani se jih v podatkovno skladišče (angl. data warehouse), kjer za dostop do njih porabimo več časa, vendar poizvedovanja po takih podatkih niso zelo pogosta.

Vrste uprabnikov PB:
Uporabniki se delijo na dva dela:

- operativni uporabniki (npr. trgovke v supermarketu). Ti uporabniki direktno dostopajo do PB (npr. trgovke preko vnajprej pripravljenega obrazca vnašajo podatke o nakupih)
- analitični uporabniki (za podporo odločanju). Ti uporabniki uporabljajo bazo za razvoj strategije organizacije, za pregled poslovanja ipd. Delajo se dolgoročnejše analize (npr. primerjava celotne prodaje po četrtletjih v letu 2005). Analitični uporabniki pogosto uporabljajo podatke v sumarni obliki.

Razliko med obema vrstama uporabnikov PB prikazuje spodnja tabela.

6.1 Obrazci in poročila

Obrazce in poročila v Accessu lahko kreiramo s pomočjo čarovnika (wizard).
Kreiramo lahko obrazce iz

- ene same tabele (za vnos novih podatkovnih zapisov)
- več tabel (form and subform).

Operativni uporabniki	Analitični uporabniki
zanimive trenutne vrednosti	zanimivi zgodovinski podatki
pogost dostop do PB	neenakomeren dostop do PB
mala količina podatkov naenkrat	velike količine podatkov
način uporabe je rutinski (obrazci, programi)	način uporabe zelo spremenljiv
kratek čas dostopa	čas dostopa ni tako pomemben (pridobitev velike količine (sumarnih) podatkov)

Primer: tabela studentov in njihovih ocen pri različnih izpitih. V subformi je tista tabela, ki ima za eno vrednost druge tabele možnih več vrednosti (torej ima na povezavi z drugo tabelo kardinalnost N (mnogo)). V primeru študentov in ocen so to ocene, saj ima en študent običajno več ocen. Glavna tabela je torej tista, ki enolično določa vnešene podatke.

NASVET: Pri obrazcih pazimo, da smo dodali vsa OBVEZNA polja za vnos, sicer ne moremo vnesti v tabelo ničesar!

6.2 Primeri

- Obrazec s filmom in možnostjo vpisovanja vseh umetnikov, ki so pri njem sodelovali (seveda z vpisom tudi njihovih vlog).
- Obrazec z možnostjo vnosa nove predstave, z izbiro filma za to predstavo in določitvijo dvorane.
- Poročilo o kinodvoranah, za vsako posamezno dvorano izpis njenih karakteristik, število gledalcev na predstavah in skupno število nezasedenih stolov.

References

[1] J. Jaklič, *Upravljanje in uporaba podatkov*, Založništvo Ekonomske fakultete, Ljubljana 2002

[2] T. Mohorič, *Uvod v podatkovne baze*, BI-TIM d.o.o., Ljubljana 1995