

Normalizacija

Vsebina:

- Funkcionalne odvisnosti
- Namen normalizacije.
 - Uporaba normalizacije pri načrtovanju relacijske podatkovne baze.
 - Problemi zaradi redundance podatkov v osnovnih relacijah.
- Postopek normalizacije.
- Osnovne normalne oblike:
 - I. normalna oblika,
 - II. Normalna oblika,
 - III. Normalna oblika
 - IV. Poslovna normalna oblika

Ponovitev

Funkcionalne odvisnosti...

- Relacija je model nekega stanja v svetu → njena vsebina ne more biti poljubna.
- Realne omejitve ne omogočajo, da bi bili odnosi v svetu kakršnikoli; možna so le določena stanja.
- Odvisnosti so sredstvo, s katerim lahko v relacijskem modelu povemo, katere vrednosti relacij so veljavne in katere sploh ne morejo obstajati.
- Oglejte si video predavanje na db-class.org

Funkcionalne odvisnosti...

- Poznamo več vrst odvisnosti:
 - Funkcionalne odvisnosti (functional dependency)
 - Večvrednostne odvisnosti (multivalued dependency)
 - Stične odvisnosti (join dependency)
- Obravnavali bomo funkcionalne odvisnosti; ostale bomo obravnavali v okviru postopka normalizacije.

Funkcionalne odvisnosti...

- Predpostavimo, da obstaja relacijska shema R z množico atributov, katere podmnožici sta X in Y .
- V relacijski shemi R velja $X \rightarrow Y$ (X funkcionalno določa Y oziroma Y je funkcionalno odvisen od X), če v nobeni relaciji, ki pripada shemi R , ne obstajata dve n -terici, ki bi se ujemali v vrednostih atributov X in se ne bi ujemali v vrednostih atributov Y .

Funkcionalne odvisnosti

- Množico funkcionalnih odvisnosti, ki veljajo med atributi funkcionalne sheme R in v vseh njenih relacijah, označimo s F

$$X \rightarrow Y \in F \Leftrightarrow \forall r (Sh(r) = R \Rightarrow \forall t, \forall u (t \in r \wedge u \in r \wedge t.X = u.X \Rightarrow t.Y = u.Y))$$

kjer

t.X, u.X, t.Y in u.Y označujejo vrednosti atributov X oziroma Y v n-tericah t oziroma u.

Primeri funkcionalnih odvisnosti

- Imamo relacijo s shemo
Izpit(VpŠt, Priimek, Ime, ŠifraPredmeta, Datum izpita, OcenaPisno, OcenaUstno)
- z naslednjim pomenom:
Študent z vpisno številko VpŠt ter priimkom Priimek in imenom Ime je na DatumIzpita opravljal izpit iz predmeta s šifro ŠifraPredmeta. Dobil je oceno OcenaPisno in OcenaUstno.
- Funkcionalne odvisnosti relacijske sheme Izpit
SO:
 $F \equiv \{ VpŠt \rightarrow (Priimek, Ime), (VpŠt, ŠifraPredmeta, DatumIzpita) \rightarrow (OcenaPisno, OcenaUstno) \}$

Ključni relacije...

- Ker je relacija množica n-teric, so v njej vse n-terice ločene med seboj.
- Za sklicevanje na posamezno n-terico ni potrebno poznati vseh vrednosti atributov n-terice, če v shemi nastopajo funkcionalne odvisnosti.
- Množici atributov, ki določajo vsako n-terico, pravimo ključ relacije oziroma ključ relacijske sheme.

Ključni relacije...

- Predpostavimo, da obstaja relacijska shema z atributi $A_1 A_2 \dots A_n$ katere podmnožica je množica atributov X .
- Atributi X so ključ relacijske sheme oziroma pripadajočih relacij, če sta izpolnjena naslednja dva pogoja:
 - (1) $X \rightarrow A_1 A_2 \dots A_n$
 - (2) ne obstaja X' , ki bi bila prava podmnožica od X in ki bi tudi funkcionalno določala $A_1 A_2 \dots A_n$

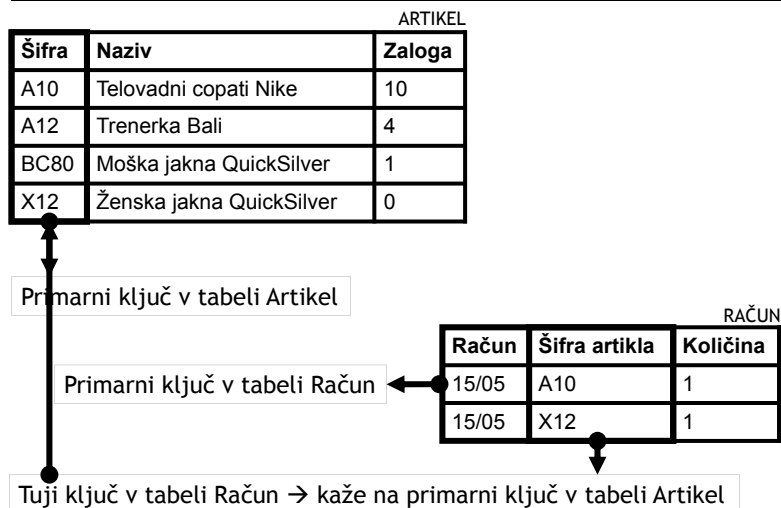
Ključni relacije...

- Poznamo več vrst ključev:
 - Kandidat za ključ (a key candidate)
 - Primarni ključ (primary key)
 - Superključ (superkey)
 - Tuji ključ (foreign key)
- Kandidat za ključ je vsaka podmnožica atributov relacije, ki relacijo enolično določa.

Ključni relacije

- Primarni ključ je tisti kandidat za ključ, ki ga izberemo za shranjevanje relacij v fizični podatkovni bazi.
- Superključ je vsaka množica atributov, v kateri je vsebovan ključ → ključ je podmnožica superključa.
- Tuji ključ je množica atributov, v okviru ene relacije, ki je enaka kandidatu za ključ neke druge ali iste relacije.

Primeri ključev



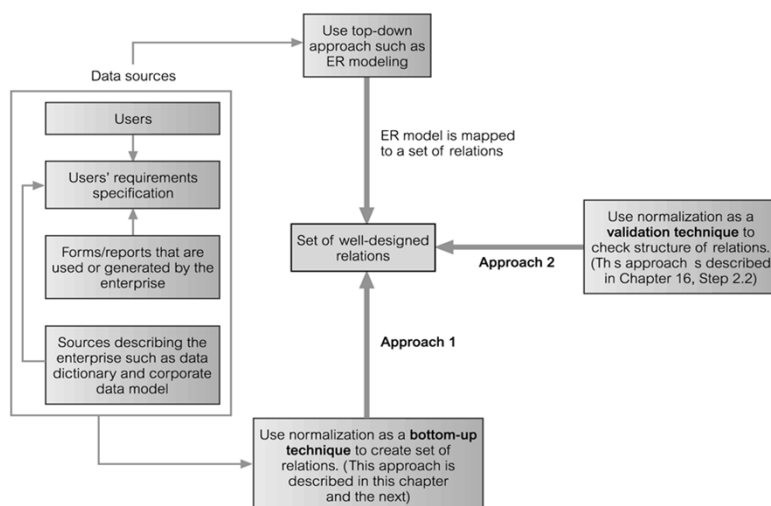
Namen normalizacije...

- Normalizacija je postopek, s katerem pridemo do množice primernih relacij, ki ustrezajo potrebam poslovne domene.
- Nekaj lastnosti primernih relacij:
 - Relacije imajo minimalen nabor atributov → zgolj tiste, ki so potrebni za pokritje potreb poslovnega sistema;
 - Atributi, ki so logično povezani, so zajeti v isti relaciji;
 - Med atributi relacij je minimalna redundanca → vsak atribut (razen tujih ključev) je predstavljen samo enkrat.

Namen normalizacije...

- Prednosti uporabe podatkovnih baz, ki jih sestavljajo množice primernih (normaliziranih) relacij, so:
 - Enostavnejša dostop do podatkov ter vzdrževanje podatkov;
 - Večja učinkovitost;
 - Boljša izraba diskovnih kapacitet.

Normalizacija pri načrtovanju relac. PB



Prednosti pravilnega načrtovanja

- Osnovni cilj načrtovanja relacijske podatkovne baze je grupirati attribute v relacije tako, da bo čim manj redundance med podatki.
- Potencialne koristi pravilnega načrtovanja so:
 - Spremembe podatkov v podatkovni bazi dosežemo z minimalnim številom operacij → večja učinkovitost; manj možnosti za podatkovne nekonsistentnosti.
 - Manjše potrebe po diskovnih kapacitetah za shranjevanje osnovnih relacij → manjši stroški.

Primer

- Relacija StaffBranch ima odvečne podatke.

StaffBranch

staffNo	sName	position	salary	branchNo	bAddress
SL21	John White	Manager	30000	B005	22 Deer Rd, London
SG37	Ann Beech	Assistant	12000	B003	163 Main St, Glasgow
SG14	David Ford	Supervisor	18000	B003	163 Main St, Glasgow
SA9	Mary Howe	Assistant	9000	B007	16 Argyll St, Aberdeen
SG5	Susan Brand	Manager	24000	B003	163 Main St, Glasgow
SL41	Julie Lee	Assistant	9000	B005	22 Deer Rd, London

Atribut z odvečnimi (ponavljajočimi) podatki

Ažurne anomalije

- Relacije, ki vsebujejo odvečne podatke lahko povzročajo anomalije pri spreminjanju podatkov → govorimo o ažurnih anomalijah.
- Poznamo več vrst anomalij:
 - Anomalije pri dodajanju n-teric v relacijo
 - Anomalije pri brisanju n-teric iz relacije
 - Anomalije pri spreminjanju n-teric

Anomalije pri dodajanju

- Primeri anomalij:
 - Če želimo dodati podatke o novih članih (staff) za neko organizacijsko enoto (branch) moramo vpisati tudi vse podrobnosti o organizacijski enoti.
 - Če želimo dodati podatke o novi organizacijski enoti, ki še nima nobenega člana, moramo v vsa polja, ki člane opisujejo, vpisati Null.

StaffBranch

staffNo	sName	position	salary	branchNo	bAddress
SL21	John White	Manager	30000	B005	22 Deer Rd, London
SG37	Ann Beech	Assistant	12000	B003	163 Main St, Glasgow
SG14	David Ford	Supervisor	18000	B003	163 Main St, Glasgow
SA9	Mary Howe	Assistant	9000	B007	16 Argyll St, Aberdeen
SG5	Susan Brand	Manager	24000	B003	163 Main St, Glasgow
SL41	Julie Lee	Assistant	9000	B005	22 Deer Rd, London

Anomalije pri brisanju

▪ Primeri anomalij:

- Če iz relacije zberemo n-terico, ki predstavlja zadnjega člana v neki organizacijski enoti, zgubimo tudi podatke o tej organizacijski enoti.

StaffBranch

staffNo	sName	position	salary	branchNo	bAddress
SL21	John White	Manager	30000	B005	22 Deer Rd, London
SG37	Ann Beech	Assistant	12000	B003	163 Main St, Glasgow
SG14	David Ford	Supervisor	18000	B003	163 Main St, Glasgow
SA9	Mary Howe	Assistant	9000	B007	16 Argyll St, Aberdeen
SG5	Susan Brand	Manager	24000	B003	163 Main St, Glasgow
SL41	Julie Lee	Assistant	9000	B005	22 Deer Rd, London

- Kakšna bi bila situacija v primeru normaliziranih relacij? Ali bi problem še vedno obstajal?

Anomalije pri spreminjanju

▪ Primeri anomalij:

- Če želimo spremeniti vrednost nekega atributa določene organizacijske enote (npr. naslov), moramo popraviti vse n-terice, v katerih takšna vrednost atributa nastopa.

StaffBranch

staffNo	sName	position	salary	branchNo	bAddress
SL21	John White	Manager	30000	B005	22 Deer Rd, London
SG37	Ann Beech	Assistant	12000	B003	163 Main St, Glasgow
SG14	David Ford	Supervisor	18000	B003	163 Main St, Glasgow
SA9	Mary Howe	Assistant	9000	B007	16 Argyll St, Aberdeen
SG5	Susan Brand	Manager	24000	B003	163 Main St, Glasgow
SL41	Julie Lee	Assistant	9000	B005	22 Deer Rd, London

Postopek normalizacije

- Postopku preoblikovanja relacij v obliko, pri kateri do ažurnih anomalij ne more priti, pravimo normalizacija.
- Obstaja več stopenj normalnih oblik. Obravnavali bomo:
 - 1NO – Prva normalna oblika
 - 2NO – Druga normalna oblika
 - 3NO – Tretja normalna oblika in
 - 4PNO – Četrta poslovna normalna oblika

1NO – prva normalna oblika

- Relacija je v prvi normalni obliki, če:
 - Nima ponavljajočih skupin → atributi ne smejo biti več-vrednostni!
 - Ima definiran primarni ključ in določene funkcionalne odvisnosti
- Koraki:
 - Odstranimo ponavljajoče skupine
 - Določimo funkcionalne odvisnosti
 - Določimo primarni ključ

Primer – relacija v nenormalizirani obliki

Indeks(VŠ, priimek, ime, pošta, kraj, šifra predmeta, naziv, ocena)



Atribut, predstavljen kot ponavljajoča skupina.

VŠ	priimek	ime	pošta	kraj	šifra predmeta	naziv	ocena
64010632	Bratina	Simon	4100	Kranj	20020	IS	10
					20021	TPO	8
					20033	IPI	8
64016209	Bizjak	Tadeja	2250	Ptuj	20060	E1	9
					20033	IP1	6

Primer – pretvorba v 1NO...

Indeks(VŠ, priimek, ime, pošta, kraj, (šifra predmeta, naziv, ocena))



Odpravimo ponavljajoče skupine

Indeks(VŠ, priimek, ime, pošta, kraj, šifra predmeta, naziv, ocena)



Identificiramo funkcionalne odvisnosti

$F \equiv \{ V\check{S} \rightarrow (\text{priimek, ime, pošta, kraj}), \text{šifra predmeta} \rightarrow \text{naziv, pošta} \rightarrow \text{kraj}, (V\check{S}, \text{šifra predmeta}) \rightarrow \text{ocena} \}$



Določimo primarni ključ

Indeks(VŠ, priimek, ime, pošta, kraj, šifra predmeta, naziv, ocena)

Primer – pretvorba v 1NO

VŠ	priimek	ime	pošta	kraj	šifra predmeta	naziv	ocena
64010632	Bratina	Simon	4100	Kranj	20020	IS	10
					20021	TPO	8
					20033	IPI	8
64016209	Bizjak	Tadeja	2250	Ptuj	20060	E1	9
					20033	IPI	6

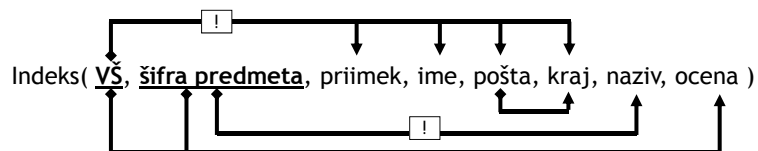


VŠ	priimek	ime	pošta	kraj	šifra predmeta	naziv	ocena
64010632	Bratina	Simon	4100	Kranj	20020	IS	10
64010632	Bratina	Simon	4100	Kranj	20021	TPO	8
64010632	Bratina	Simon	4100	Kranj	20033	IPI	8
64016209	Bizjak	Tadeja	2250	Ptuj	20060	E1	9
64016209	Bizjak	Tadeja	2250	Ptuj	20033	IPI	6

2NO – prva normalna oblika

- Relacija je v drugi normalni obliki:
 - Če je v prvi normalni obliki in
 - Ne vsebuje parcialnih odvisnosti → noben atribut, ki ni del ključa, ni funkcionalno odvisen le od dela primarnega ključa, temveč od celotnega ključa
- Druga normalna oblika je odvisna predvsem od ključa relacije. Relacija je avtomatsko v drugi normalni obliki, če:
 - Je njen primarni ključ sestavljen le iz enega atributa,
 - Je njen primarni ključ sestavljen iz vseh atributov relacije

Primer – pretvorba v 2NO...



Relacijo razbijemo

Študent(VŠ, priimek, ime, pošta, kraj)
 Predmet(šifra predmeta, naziv)
 Indeks(#VŠ, #šifra predmeta, ocena)

Primer – pretvorba v 2NO

VŠ	priimek	ime	pošta	kraj	šifra predmeta	naziv	ocena
64010632	Bratina	Simon	4100	Kranj	20020	IS	10
64010632	Bratina	Simon	4100	Kranj	20021	TPO	8
64010632	Bratina	Simon	4100	Kranj	20033	IPI	8
64016209	Bizjak	Tadeja	2250	Ptuj	20060	E1	9
64016209	Bizjak	Tadeja	2250	Ptuj	20033	IPI	6



VŠ	priimek	ime	pošta	kraj
64010632	Bratina	Simon	4100	Kranj
64016209	Bizjak	Tadeja	2250	Ptuj

VŠ	šifra predmeta	ocena
64010632	20020	10
64010632	20021	8
64010632	20033	8
64016209	20060	9
64016209	20033	6

šifra predmeta	naziv
20020	IS
20021	TPO
20033	IPI
20060	E1
20033	IPI

3NO – tretja normalna oblika

- Relacija je v tretji normalni obliki:
 - Če je v drugi normalni obliki in
 - Če ne vsebuje tranzitivnih funkcionalnih odvisnosti → med atributi, ki niso del primarnega ključa, ni odvisnosti.
- Relacija je avtomatsko v tretji normalni obliki, če:
 - Je njen ključ sestavljen iz vseh atributov relacije
 - Je njen ključ sestavljen iz vseh razen enega atributa relacije.

Primer – pretvorba v 3NO...

Študent(VŠ, priimek, ime, pošta, kraj)
Predmet(šifra predmeta, naziv)
Indeks(#VŠ, #šifra predmeta, ocena)



Relacijo razbijemo

Študent(VŠ, priimek, ime, #pošta)
Pošta(pošta, kraj)
Predmet(šifra predmeta, naziv)
Indeks(#VŠ, #šifra predmeta, ocena)

Primer – pretvorba v 3NO

VŠ	priime k	ime	pošta	kraj
64010632	Bratina	Simon	4100	Kranj
64016209	Bizjak	Tadeja	2250	Ptuj



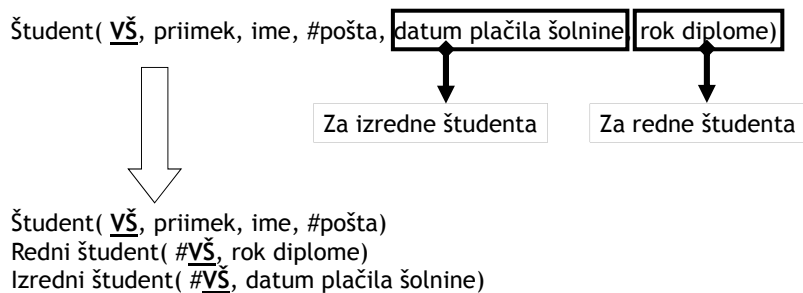
VŠ	priime k	ime	pošta
64010632	Bratina	Simon	4100
64016209	Bizjak	Tadeja	2250

pošta	kraj
4100	Kranj
2250	Ptuj

4PNO – četrta poslovna normalna oblika

- Relacija je v četrti poslovni normalni obliki, če je v tretji normalni obliki in ustreza enemu od naslednjih pogojev:
 - njeni atributi so odvisni ne samo od primarnega ključa, ampak tudi od vrednosti ključa ali
 - je bil nek atribut premeščen iz relacije, kjer je bil neobvezen v relacijo, kjer je v celoti odvisen od ključa in zato obvezen.

Primer – pretvorba v 4PNO...



Primer – pretvorba v 4PNO

VŠ	Priimek	Ime	Datum plačila šolnine	Rok diplome
64010632	Bratina	Simon		15.3.2005
64016209	Bizjak	Tadeja	19.4.2002	
64010670	Berce	Marjan	12.4.2004	
64620010	Mele	Silvana		1.4.2005
65120987	Leban	Tibor		15.7.2005

VŠ	Priimek	Ime
64010632	Bratina	Simon
64016209	Bizjak	Tadeja
64010670	Berce	Marjan
64620010	Mele	Silvana
65120987	Leban	Tibor

VŠ	Datum plačila šolnine
64016209	19.4.2002
64010670	12.4.2004

VŠ	Rok diplome
64010632	15.3.2005
64620010	1.4.2005
65120987	15.7.2005

Uporaba nenormaliziranih relacij...

- Včasih zavestno uporabljamo relacije, ki ne ustrezajo najvišjim normalnim oblikam.
- Prve in druge normalne oblike nikoli ne kršimo.
- Višjim normalnim oblikam se včasih odrečemo na račun doseganja boljše učinkovitosti.

Uporaba nenormaliziranih relacij

- Primer:
 - Rezultat (športnik, tekmovanje, čas prvega teka, čas drugega teka, čas skupaj)
 - Relacija **ni** v tretji normalni formi.
 - Čas skupaj je izpeljan atribut → ni odvisen od ključa, temveč je seštevek časov obeh tekov.
 - Skupen čas računamo ob vpisu v bazo, zato izboljšamo učinkovitost pri nadaljnji obdelavi podatkov.

Vaja

- Spodnjo relacijo pretvorite v 4PNO

Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk, (datum izplačila, plača))

Pomen relacije: delavec s šifro (**šifra delavca**), priimkom (**priimek**) ter imenom (**ime**) je zaposlen v natanko enem podjetju (**podjetje**). To podjetje se nahaja v natanko enem mestu (**mesto**). Vsi delavci imajo sklenjene delovne pogodbe (**številka pogodbe**), s to razliko, da imajo vodilni delavci sklenjene individualne pogodbe, ostali delavci pa kolektivne pogodbe, na osnovi katerih so tudi točkovani (**število točk**).

V relaciji so zajeti tudi atributi, ki povedo, kakšno plače je prejemal delavec (**datum izplačila, plača**)

Vaja

- Prva normalna oblika – odprava ponavljajočih skupin

Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk, (datum izplačila, plača))



Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk, datum izplačila, plača)

Vaja

- Prva normalna oblika – identifikacija funkcionalnih odvisnosti

Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk, datum izplačila, plača)

šifra delavca → priimek, ime, podjetje, mesto, številka pogodbe, število točk

podjetje → mesto

šifra delavca, datum izplačila → plača

$F \equiv \{ \text{šifra delavca} \rightarrow (\text{priimek, ime, podjetje, mesto, številka pogodbe, število točk}), \text{podjetje} \rightarrow \text{mesto}, (\text{šifra delavca, datum izplačila}) \rightarrow \text{plača} \}$

Vaja

- Prva normalna oblika – določitev ključa

$F \equiv \{ \text{šifra delavca} \rightarrow (\text{priimek, ime, podjetje, mesto, številka pogodbe, število točk}), \text{podjetje} \rightarrow \text{mesto}, (\text{šifra delavca, datum izplačila}) \rightarrow \text{plača} \}$

Ključ = {šifra delavca ,datum izplačila}

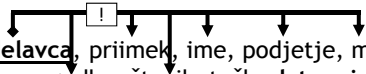
1NO

Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk, datum izplačila, plača)

Vaja

▪ Druga normalna oblika

Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk, datum izplačila, plača)



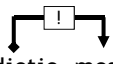
2NO

Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk)
Plača (datum izplačila, #šifra delavca, plača)

Vaja

▪ Tretja normalna oblika

Delavec (šifra delavca, priimek, ime, podjetje, mesto, številka pogodbe, število točk)



3NO

Delavec (šifra delavca, priimek, ime, #podjetje, številka pogodbe, število točk)
Podjetje (podjetje, mesto)
Plača (datum izplačila, #šifra delavca, plača)

Vaja

▪ Četrta poslovna normalna oblika

Delavec (šifra delavca, priimek, ime, #podjetje, številka pogodbe,

število točk)

Število točk nas zanima samo za delavce, ki nimajo individualnih pogodb

Podjetje (podjetje, mesto)

Plača (šifra delavca, datum izplačila, plača)

4PNO

Delavec (šifra delavca, priimek, ime, #podjetje, številka pogodbe)

Točke (#šifra delavca, število točk)

Podjetje (podjetje, mesto)

Plača (datum izplačila, #šifra delavca, plača)

Vaja 2

▪ Spodnjo relacijo pretvorite v 3NO

(KletkaID, KletkaVelikost, KletkaTipID, KletkaTipOpis,
ŽivalID, ŽivalIme, ŽivalStarost, VrstaŽivaliID, VrstaŽivaliLokacija,
ČasPrihoda, ŽivalStanje)

Pomen relacije:

- V živalskem vrtu vodijo evidenco kletk in živali.
- Za vsako kletko poznajo identifikacijsko številko (*KletkaID*) in velikost (*KletkaVelikost*). Vsaka kletka je enega tipa (*KletkaTipID*), istega tipa pa je seveda lahko več kletk. Za vsak tip kletke je podan kratek opis (*KletkaTipOpis*).
- Za vsako žival poznajo njeno identifikacijsko številko (*ŽivalID*), ime (*ŽivalIme*) in starost (*ŽivalStarost*).
- Vsaka žival pripada eni živalski vrsti (*VrstaŽivaliID*), za vsako živalsko vrsto pa vedo na kateri lokaciji živi v naravi (*VrstaŽivaliLokacija*). Imajo lahko več živali iste živalske vrste.
- Živali niso vedno v isti kletki, pač pa jih med letom selijo. Ob selitvi vsakokrat zabeležijo čas prihoda v novo kletko (*ČasPrihoda*) in zapišejo stanje živali (*ŽivalStanje*). Ob istem času lahko preselijo tudi več živali, neko žival pa lahko preselijo tudi v kletko v kateri je bivala že kdaj prej.
- Shranjujejo zgodovino vseh selitev med kletkami.

Logično načrtovanje

Vsebina:

- Sistematična izvedba logičnega načrtovanja
- Podrobnosti pri pretvorbi iz konceptualnega v logični model:
 - entitetni tipi, nadtipi in podtipi, povezave,...
- Preverba modela z normalizacijo
- Združevanje lokalnih shem
- Dokumentiranje logičnega modela

Metoda logičnega načrtovanja...

- Možni koraki logičnega načrtovanja:
 - K2.1: Za entitetne tipe kreiraj relacije
 - K2.2: Preveri relacije z normalizacijo
 - K2.3: Preveri relacije s pregledom uporabniških transakcij
 - K2.4: Preveri omejitve integritete
 - K2.5: Preveri model z uporabnikom
 - K2.6: Združi lokalne modele v globalni model (opcijsko)
 - K2.7: Preveri zmožnosti modela za razširitve

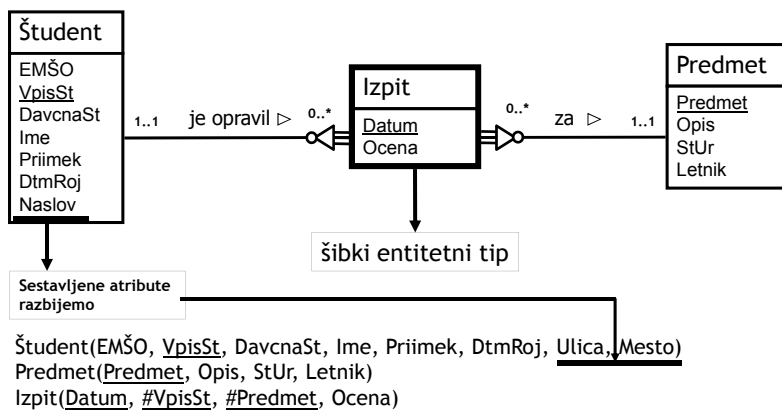
K2.1 – Za entitetne tipe kreiraj relacije...

- Namen
 - Izdelati relacije za logični model, ki bo predstavljal entitete, povezave in attribute, ki smo jih identificirali v okviru konceptualnega modeliranja.
- Ta korak je navadno avtomatiziran → pretvorba iz konceptualnega v logični model je podprta s strani številnih CASE orodij.
- Prikaz z uporabo orodja PowerDesigner.

K2.1 – Za entitetne tipe kreiraj relacije...

- Ročna pretvorba:
 - Močni entitetni tipi > Enolični identifikator sestavljajo le atributi entitete (identifikacijski atributi)
 - Za vsak močni entitetni tip kreiraj relacijo, ki vključuje vse enostavne attribute tega entitetnega tipa. Namesto sestavljenih atributov vključi njihove attribute, ki jih sestavljajo. (Npr.: atribut naslov razstavi na attribute: ulica, hišna_št, kraj, pošt_št)
 - Šibki entitetni tipi
 - Za vsak šibki entitetni tip kreiraj relacijo, ki vključuje vse enostavne attribute tega entitetnega tipa. Primarni ključ šibkega entitetnega tipa je delno ali v celoti sestavljen iz atributov, ki so ključ v entitetnih tipih, s katerimi je opazovani entitetni tip povezan. Da bi lahko določili ključ, moramo najprej pretvoriti vse povezave.

Primer: močni in šibki entitetni tipi



K2.1 – Za entitetne tipe kreiraj relacije...

▪ Ročna pretvorba (nadaljevanje):

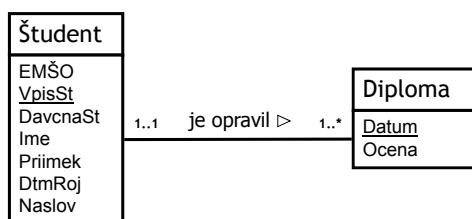
- Povezave 1:*
 - Za vsako povezavo 1:* prenesi ključ entitetnega tipa, ki nastopa v povezavi na strani 1 (oče) v entitetni tip, ki nastopa v povezavi na strani * (otrok). Dobimo tuji ključ.
- Povezave 1:1
 - Pri števnosti 1:1 ne moremo vedno enostavno določiti očeta in otroka. Za odločitev, ali bomo entitetna tipa, ki sta povezana s povezavo 1:1, povezali v eno relacijo ali ju predstavili z dvema, preverimo predvsem, kako je z obveznostjo povezav. Možne omejitve so: obveznost na obeh straneh, neobveznost na eni in obveznost na drugi, neobveznost na obeh straneh.

K2.1 – Za entitetne tipe kreiraj relacije...

▪ Ročna pretvorba (nadaljevanje):

- Povezave 1:1 (nadaljevanje...)
 - Obveznost na obeh straneh 1:1 povezave
 - Entitetna tipa združi v eno relacijo. Za primarni ključ izberi enega izmed primarnih ključev originalnih entitetnih tipov.
 - Obveznost na eni strani 1:1 povezave
 - Entitetni tip, ki ni obvezen v povezavi, naj bo oče, entitetni tip z obvezno povezavo pa naj bo otrok. Kopija primarnega ključa entitetnega tipa očeta se prenese na entitetni tip otroka.
 - Neobvezna povezava na obeh straneh 1:1 povezave
 - V primerih, ko sta oba entitetna tipa neobvezna, je težko določiti očeta in otroka povezave. Ko pridobimo dovolj podatkov, določimo ključ.

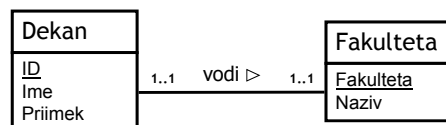
Primer: povezave 1:*



Študent(EMŠO, VpisSt, DavcnaSt, Ime, Priimek, DtmRoj, Ulica, Mesto)
Diploma(Datum, #VpisSt, Ocena)

↓
Prenos tujega ključa v smeri ena "proti mnogo"

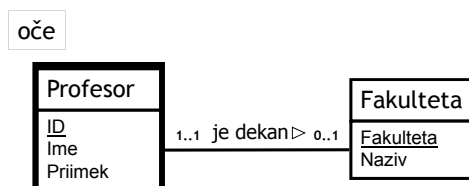
Primer: povezave 1:1



Dekan(ID, Ime, Priimek, Naziv_fakultete)

Obveznost na obeh straneh:
entitetna tipa predstavimo z eno relacijo; izberemo ključ

Primer: povezave 1:1



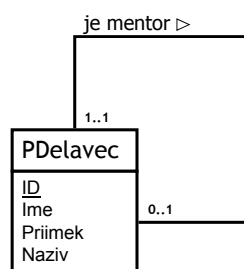
Fakulteta(Fakulteta, Naziv, #ID)
Profesor(ID, Ime, Priimek)

Obveznost na eni strani:
entitetni tip, ki igra vlogo očeta, da primarni ključ → drugi entitetni tip
dobi tuji ključ.

K2.1 – Za entitetne tipe kreiraj relacije...

- Ročna pretvorba (nadaljevanje):
 - Rekurzivne povezave 1:1
 - Pri rekurzivnih povezavah tipa 1:1 upoštevaj pravila, ki izhajajo iz obveznosti povezav.
 - Obveznost na obeh straneh: rekurzivno povezavo predstavi z eno relacijo in dvema kopijama primarnega ključa.
 - Obveznost na eni, neobveznost na drugi strani: kreiraj eno relacijo z dvema kopijama primarnega ključa ali kreiraj novo relacijo. Nova relacija naj ima samo dva atributa – kopiji primarnega ključa.
 - Neobveznost na obeh straneh: kreiraj novo relacijo. Nova relacija naj ima samo dva atributa – kopiji primarnega ključa.
 - Kopije primarnih ključev je v rekurzivnih povezavah potrebno ustrezno poimenovati, da lahko ločimo med njimi!

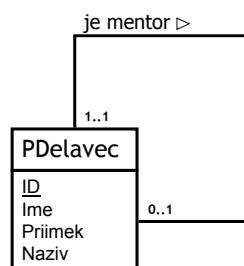
Primer: rekurzivne povezave



PDelavec(ID, Ime, Priimek, Naziv, #IDmentorja)

Obveznost na eni strani:
kreiramo relacijo z dvema kopijama primarnega ključa. Eden igra vlogo primarnega drugi pa tujega ključa.

Primer: rekurzivne povezave



PDelavec(ID, Ime, Priimek, Naziv)
Mentor(#IDmentorja, #IDdelavca)

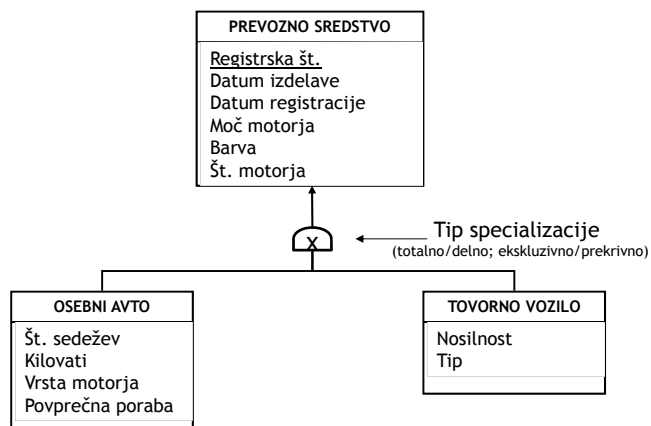
Obveznost na eni strani:

Rekurzivno povezavo pretvorimo v relacijo. Relacija za vsakega pedagoškega delavca pove, kdo je njegov mentor.

K2.1 – Za entitetne tipe kreiraj relacije...

- Ročna pretvorba (nadaljevanje):
 - Povezave med nadtipi in podtipi
 - Identificiraj nadtype kot očete ter podtype kot otroke. Obstajajo različne možnosti, kako takšne povezave predstaviti z relacijami.
 - Izbira najbolj ustrezne opcije je odvisna od številnih faktorjev: izključevanje, obveznost povezav, število entitet v povezavi....
- Prikaz preslikave dedovanja s PD

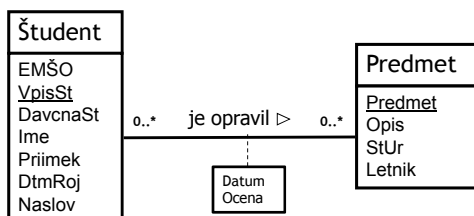
Primer: nadtipi, podtipi



K2.1 – Za entitetne tipe kreiraj relacije...

- Ročna pretvorba (nadaljevanje):
 - Povezave *:*
 - Kreiraj relacijo, ki predstavlja povezavo ter vse njene atribute. Primarne ključe entitetnih tipov, ki sta povezana s tako povezavo, vključi v novo relacijo kot tuji ključ. Tuji ključi bodo obenem tudi primarni ključi – samostojno ali v kombinaciji z drugimi atributi relacije.

Primer: povezave *:*



Študent(EMŠO, VpisSt, DavcnaSt, Ime, Priimek, DtmRoj, Ulica, Mesto)
Predmet(Predmet, Opis, StUr, Letnik)
Izpit(Datum, #VpisSt, #Predmet, Ocena)

K2.1 – Za entitetne tipe kreiraj relacije

▪ Ročna pretvorba (nadaljevanje):

– Več-vrednostni atributi

- Za predstavitev več-vrednostnih atributov kreiraj novo relacijo. V novo relacijo vključi tudi ključ entitetnega tipa, iz katerega izhajajo več-vrednostni atributi. V novi relaciji predstavlja tuji ključ. Primarni ključ v novi relaciji je kombinacija tujega ključa in več-vrednostnih atributov. Če več-vrednostni atributi sami predstavljajo kandidata za ključ, potem ni potrebno, da primarni ključ zajema tudi tuji ključ.
- Če je število vrednosti za večvrednostni atribut znano in ni veliko (npr. je manjše d 5), lahko tak atribut predstavimo z več atributi v relaciji. Za vsako vrednost svoj atribut.

Primer: večvrednostni atributi...

Študent
<u>VpisSt</u>
Ime
Priimek
DtmRoj
Naslov
Telefon

Študent(VpisSt, Ime, Priimek, DtmRoj, Mesto, Ulica, GSM, StcTelefon)

Večvrednostni atribut:

Število vrednosti za Telefon je znano, zato za vsako določimo svoj atribut v isti relaciji.

Primer: večvrednostni atributi...

Konferenca
<u>ID</u>
Datum
NazivKonf
Kraj
Sponzor

Konferenca(ID, Datum, NazivKonf, Kraj)

Sponzor(Sponzor, Naziv)

SponzorKonference(#ID, #Sponzor)

Večvrednostni atribut:

Število vrednosti za atribut sponzor ni znano, zato kreiramo novi relaciji.

K2.2 – Preveri relacije z normalizacijo...

- Namen tega koraka je preveriti, če so vse pridobljene relacije v ustrezni normalni obliki. To zagotavlja:
 - Da imajo relacije minimalno, vendar zadostno število atributov za potrebe problemske domene;
 - Da ni odvečnih podatkov (razen za potrebe povezovanja → tuji ključi)
- Prevedba konceptualnega modela v logični model navadno da relacije, ki ustrezajo 3NO.
 - Če to ne drži, so v konceptualnem modelu ali v postopku prevedbe napake.

K2.2 – Preveri relacije z normalizacijo...

- Včasih se zdi, da normalizirane relacije ne omogočajo zadovoljive učinkovitosti podatkovne baze.
- Upoštevati je potrebno:
 - V normaliziranih relacijah so podatki organizirani v skladu s funkcionalnimi odvisnostmi.
 - Logični podatkovni model ni dokončen. Predstavlja le, kako načrtovalec razume pomen in naravo podatkov, potrebnih za obravnavano problemsko domeno; Specifične potrebe v zvezi z učinkovitostjo lahko zahtevajo drugačen fizični model.

K2.2 – Preveri relacije z normalizacijo

- Upoštevati je potrebno (nadaljevanje):
 - Normaliziran načrt je robusten in odporen na podatkovne anomalije.
 - Moderni računalniki so veliko zmogljivejši → včasih je upravičeno uporabiti rešitve, ki omogočajo enostavnejšo obdelavo na račun več procesiranja.
 - Normalizacija načrtovalca prisili, da se natanko spozna z vsakim atributom relacije.
 - Z normalizacijo pridemo do fleksibilnega načrta, ki ga brez težav razširimo.

K2.3 – Preveri relacije z vidika transakcij

- Podobno kot konceptualni model preverimo tudi logični model z vidika podpore transakcij, ki jih uporabnik specificira (glej [K1.8](#)).
- Če vseh transakcij ni moč izvesti ročno, smo pri pretvorbi naredili napako, ki jo je potrebno odpraviti.

K2.4 – Preveri omejitve integritete...

- V tem koraku preverimo pravila za zagotavljanje celovitosti podatkov:
 - Obveznost atributov
 - Omejitve domen atributov
 - Števnost
 - Omejitve entitet (celovitost entitet)
 - Omejitve povezav (celovitost povezav) – Referencialna integriteta
 - Splošne omejitve

Primeri omejitev povezav

```
Staff (staffNo, fName, lName, position, sex, DOB, supervisorStaffNo)
Primary Key staffNo
Foreign Key supervisorStaffNo references Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL

Client (clientNo, fName, lName, telNo, prefType, maxRent, staffNo)
Primary Key clientNo
Foreign Key staffNo references Staff(staffNo) ON UPDATE CASCADE ON DELETE NO ACTION

PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo)
Primary Key propertyNo
Foreign Key ownerNo references PrivateOwner(ownerNo) and BusinessOwner(ownerNo)
ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key staffNo references Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL

Viewing (clientNo, propertyNo, dateView, comment)
Primary Key clientNo, propertyNo
Foreign Key clientNo references Client(clientNo) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key propertyNo references PropertyForRent(propertyNo)
ON UPDATE CASCADE ON DELETE CASCADE

Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo)
Primary Key leaseNo
Alternate Key propertyNo, rentStart
Alternate Key clientNo, rentStart
Foreign Key clientNo references Client(clientNo) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key propertyNo references PropertyForRent(propertyNo)
ON UPDATE CASCADE ON DELETE NO ACTION
```

Vir: Connoly

K2.5 – Preveri model z uporabnikom...

- Namen tega koraka je preveriti model z uporabnikom ter ugotoviti, če ustreza vsem uporabniškim zahtevam.
- Model lahko zajema več uporabniških pogledov. Pri pregledu lahko nastopa več uporabnikov.
- Odličen način za pregled celovitosti podatkovnega modela je specifikacija podatkovnih tokov s pomočjo diagrama podatkovnih tokov.

K2.6 – Združi lokalne modele...

- Namen tega koraka je združiti vse lokalne modele v en globalni model, ki predstavlja vse uporabniške vidike podatkovne baze.
- Čeprav so lokalni modeli preverjeni, lahko pri njihovem združevanju pride do prekrivanja in neskladnosti.
- Globalni model preverimo podobno kot smo preverjali lokalne modele.
- Če pri načrtovanju nismo zajeli več uporabniških vidikov, lahko korak preskočimo.

K2.6 – Združi lokalne modele

- Pri združevanju lokalnih modelov uporabimo naslednje korake:
 - K2.6.1 – Lokalne modele združi v globalni model
 - K2.6.2 – Preveri globalni model
 - K2.6.3 – Globalni model preveri z uporabniki

Pravila združevanja lokalnih modelov...

- Pravila združevanja lokalnih modelov:
 1. Preveri imena in vsebino relacij ter njihove kandidate za ključ. Pomagamo si lahko z okvirno shemo. Preveriti je potrebno entitetne tipe in povezave na spojih.
 2. Preveri imena in vsebino povezav in tujih ključev.
 3. Združi relacije z lokalnih podatkovnih modelov.
 4. Brez združevanja vključi relacije, ki so unikatne v posameznih podatkovnih modelih.
 5. Združi povezave in tuje ključe iz lokalnih podatkovnih modelov.

Pravila združevanja lokalnih modelov

- Pravila združevanja lokalnih modelov (nadaljevanje):
 6. Brez združevanja vključi povezave in tuje ključe, ki so unikatni v posameznih podatkovnih modelih.
 7. Preveri, če morda manjkajo relacije, povezave in tuji ključi.
 8. Preveri tuje ključe.
 9. Preveri pravila za zagotavljanje celovitosti podatkov.
 10. Nariši globalni podatkovni model.
 11. Ažuriraj dokumentacijo.

Primer

Združevanje lokalnih modelov:

Lokalni model 1:
Branch user views
(na sliki)

Lokalni model 2:
Staff user views

Branch (branchNo, street, city, postcode, mgrStaffNo) Primary Key branchNo Alternate Key postcode Foreign Key mgrStaffNo references Manager(staffNo)	Telephone (telNo, branchNo) Primary Key telNo Foreign Key branchNo references Branch(branchNo)
Staff (staffNo, name, position, salary, supervisorStaffNo, branchNo) Primary Key staffNo Foreign Key supervisorStaffNo references Staff(staffNo) Foreign Key branchNo references Branch(branchNo)	Manager (staffNo, mgrStartDate, bonus) Primary Key staffNo Foreign Key staffNo references Staff(staffNo)
PrivateOwner (ownerNo, name, address, telNo) Primary Key ownerNo	BusinessOwner (bName, bType, contactName, address, telNo) Primary Key bName Alternate Key telNo
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo, bName, branchNo) Primary Key propertyNo Foreign Key ownerNo references PrivateOwner(ownerNo) Foreign Key bName references BusinessOwner(bName) Foreign Key staffNo references Staff(staffNo) Foreign Key branchNo references Branch(branchNo)	Client (clientNo, name, telNo, prefType, maxRent) Primary Key clientNo
Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo) Primary Key leaseNo Alternate Key propertyNo, rentStart Alternate Key clientNo, rentStart Foreign Key clientNo references Client(clientNo) Foreign Key propertyNo references PropertyForRent(propertyNo) Derived deposit (PropertyForRent.rent*2) Derived duration (rentFinish - rentStart)	Registration (clientNo, branchNo, staffNo, dateJoined) Primary Key clientNo Foreign Key clientNo references Client(clientNo) Foreign Key branchNo references Branch(branchNo) Foreign Key staffNo references Staff(staffNo)
Advert (propertyNo, newspaperName, dateAdvert, cost) Primary Key propertyNo, newspaperName, dateAdvert Foreign Key propertyNo references PropertyForRent(propertyNo) Foreign Key newspaperName references Newspaper(newspaperName)	Newspaper (newspaperName, address, telNo, contactName) Primary Key newspaperName Alternate Key telNo

Primer

Združevanje lokalnih modelov:

Primerjava imen in vsebine (1)

Branch user views		Staff user views	
Entity/Relation	Candidate keys	Entity/Relation	Candidate keys
Branch	branchNo postcode telNo		
Telephone	telNo		
Staff	staffNo	Staff	staffNo
Manager	staffNo		
PrivateOwner	ownerNo	PrivateOwner	ownerNo
BusinessOwner	bName telNo	BusinessOwner	bName telNo ownerNo
Client	clientNo	Client	clientNo
PropertyForRent	propertyNo	PropertyForRent	propertyNo
Lease	leaseNo propertyNo, rentStart clientNo, rentStart	Viewing	clientNo, propertyNo
Registration	clientNo	Lease	leaseNo propertyNo, rentStart clientNo, rentStart
Newspaper	newspaperName telNo		
Advert	(propertyNo, newspaperName, dateAdvert)		

Vir: Connoly

Primer

Združevanje lokalnih modelov: preverjanje vsebine tujih ključev (2)

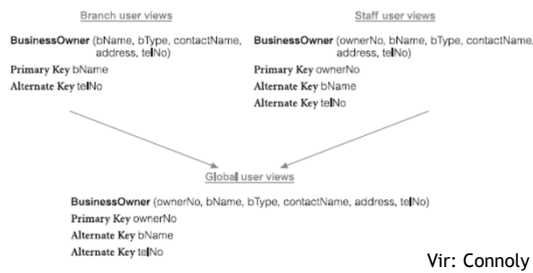
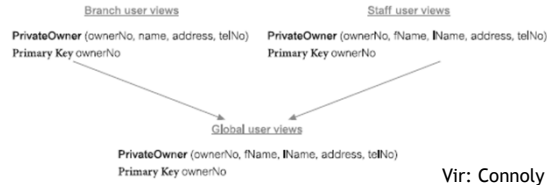
Branch user views			Staff user views		
Child relation	Foreign keys	Parent relation	Child relation	Foreign keys	Parent relation
Branch	mgrStaffNo →	Manager(staffNo)			
Telephone ^a	branchNo →	Branch(branchNo)			
Staff	supervisorStaffNo →	Staff(staffNo)	Staff	supervisorStaffNo →	Staff(staffNo)
	branchNo →	Branch(branchNo)			
Manager	staffNo →	Staff(staffNo)			
PrivateOwner			PrivateOwner		
BusinessOwner			BusinessOwner		
Client			Client	staffNo →	Staff(staffNo)
PropertyForRent	ownerNo →	PrivateOwner(ownerNo)	PropertyForRent	ownerNo →	PrivateOwner(ownerNo)
	bName →	BusinessOwner(bName)		ownerNo →	BusinessOwner(ownerNo)
	staffNo →	Staff(staffNo)		staffNo →	Staff(staffNo)
	branchNo →	Branch(branchNo)			
Lease	clientNo →	Client(clientNo)	Viewing	clientNo →	Client(clientNo)
	propertyNo →	PropertyForRent(propertyNo)	Lease	propertyNo →	PropertyForRent(propertyNo)
Registration ^b	clientNo →	Client(clientNo)		clientNo →	Client(clientNo)
	branchNo →	Branch(branchNo)		propertyNo →	PropertyForRent(propertyNo)
	staffNo →	Staff(staffNo)			
Newspaper					
Advert	propertyNo →	PropertyForRent(propertyNo)			
	newspaperName →	Newspaper(newspaperName)			

^a The Telephone relation is created from the multi-valued attribute telNo
^b The Registration relation is created from the ternary relationship Registers
^c The Advert relation is created from the many-to-many (**) relationship Advertisers

Vir: Connoly

Primer

Združevanje lokalnih modelov: združevanje relacij (3)



Primer

Združevanje lokalnih modelov:

Rezultat: Globalni logični podatkovni model

Branch (branchNo, street, city, postcode, mgrStaffNo) Primary Key branchNo Alternate Key postcode Foreign Key mgrStaffNo references Manager(staffNo)	Telephone (telNo, branchNo) Primary Key telNo Foreign Key branchNo references Branch(branchNo)
Staff (staffNo, fName, lName, position, sex, DOB, salary, supervisorStaffNo, branchNo) Primary Key staffNo Foreign Key supervisorStaffNo references Staff(staffNo) Foreign Key branchNo references Branch(branchNo)	Manager (staffNo, mgrStartDate, bonus) Primary Key staffNo Foreign Key staffNo references Staff(staffNo)
PrivateOwner (ownerNo, fName, lName, address, telNo) Primary Key ownerNo	BusinessOwner (ownerNo, bName, bType, contactName, address, telNo) Primary Key ownerNo Alternate Key bName Alternate Key telNo
PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo, branchNo) Primary Key propertyNo Foreign Key ownerNo references PrivateOwner(ownerNo) and BusinessOwner(ownerNo) Foreign Key staffNo references Staff(staffNo) Foreign Key branchNo references Branch(branchNo)	Viewing (clientNo, propertyNo, dateView, comment) Primary Key clientNo, propertyNo Foreign Key clientNo references Client(clientNo) Foreign Key propertyNo references PropertyForRent(propertyNo)
Client (clientNo, fName, lName, telNo, prefType, maxRent) Primary Key clientNo	Registration (clientNo, branchNo, staffNo, dateJoined) Primary Key clientNo Foreign Key clientNo references Client(clientNo) Foreign Key branchNo references Branch(branchNo) Foreign Key staffNo references Staff(staffNo)
Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo) Primary Key leaseNo Alternate Key propertyNo, rentStart Alternate Key clientNo, rentStart Foreign Key clientNo references Client(clientNo) Foreign Key propertyNo references PropertyForRent(propertyNo) Derived deposit (PropertyForRent.rent*2) Derived duration (rentFinish - rentStart)	Newspaper (newspaperName, address, telNo, contactName) Primary Key newspaperName Alternate Key telNo
Advert (propertyNo, newspaperName, dateAdvert, cost) Primary Key propertyNo, newspaperName, dateAdvert Foreign Key propertyNo references PropertyForRent(propertyNo) Foreign Key newspaperName references Newspaper(newspaperName)	Vir: Connoly

K2.7 – Preveri možnosti za razširitve...

- Če model ustreza le trenutnim zahtevam, je njegovo življenje lahko relativno kratko.
- V primeru, *da so predvidene bodoče razširitve* sistema, moramo preveriti, če logični model take razširitve podpira.
- Podatkovni model mora biti prilagodljiv; omogočati mora razširitve skladno z novimi zahtevami ter z minimalnim vplivom na obstoječe uporabnike.
- Popolnoma odprt sistem za razširitve je težko doseči.