Normaliseerimine

Relatsioonilises andmebaasis kujutab tabeli iga lahtri sisu endast atomaarset andmeelementi, so. igasse lahtrisse saab salvestada ainult uhe andmeelemendi, jarelikult mitut andmeelementi (nn. korduvaid) ei ole voimalik salvestada uhte lahtrisse.

Andmestruktuure, millest on korvaldatud korduvad andmeelemendid, nimetatakse **normaliseeritud** andmestruktuurideks.

Relatsioonilistes andmebaasides peavad andmestruktuurid olema normaliseeritud. Vastasel korral pole andmeid voimalik organiseerida tabelite vormis.

Et vältida liiast andmebaasis, on vaja andmebaas normaliseerida. **Normaliseerimine** võimaldab vältida mitmeid vigu, mis võivad tekkida normaliseerimata andmete põhjal loodud andmebaasides. Viimastes tulevad sagedamini ette järgmised vead:

- Andmeliiasus, mis tähendab, et esineb ülearuseid andmeid ja samade andmete asjatut kordamist;
- Andmete vastuolulisus, s.t. andmete liiasusega kaasneb sageli segadus andmetes, kuna ei ole alati üheselt selge, milline andmevariant on õige;
- Esinevad segadused ja ebamugavused uute andmete lisamisel;
- Ei ole tagatud andmekaitse, kuna uute andmete lisamisel ja ebaoluliste kustutamisel on võimalik ka mõnede vajalike andmete kaotsiminek.

Nimetatud vigade vältimine on väga oluline tarbija-sõbraliku, usaldusväärse ja operatiivse andmekasutust võimaldava andmebaasi loomisel.

Normaliseerimine - andmestruktuuride organiseerimine sel viisil, et neist struktuuridest kaoksid **korduvad andmeelemendid**, et ei oleks **dubleeritud andmeid** süsteemis, andmeliiasuse kõrvaldamine.

Normaliseerimisprotsess koosneb mitmesest etapist, igal etapil määratakse nö **normaalkuju.** On olemas 7 normaalkuju. Selles kursuses vaatame ainul **kolme esimest**. Üldiselt on soovitav organiseerida andmed relatsioonilises baasis **vähemalt kolme** esimese normaalkuju nõuetele vastavalt.

Esimeste kolme etapppide tulemustena on

- esimene normaalkuju, 1NK (First Normal Form – 1NF),
- teine normaalkuju, 2NK (Second Normal Form – 2NF),
- kolmas normaalkuju, 3NK (Third Normal Form – 3NF).

Enamuses andmebaasiprojektides **3**NK lõpetab normaliseerimise protsessi.

Normaalkujude põhiomadused

- Iga järgmine normaalkuju mõnes mõttes on parem kui eelmine
- Üleminekul järgmisele normaalkujule, eelmiste normaalkujude omadused säilivad

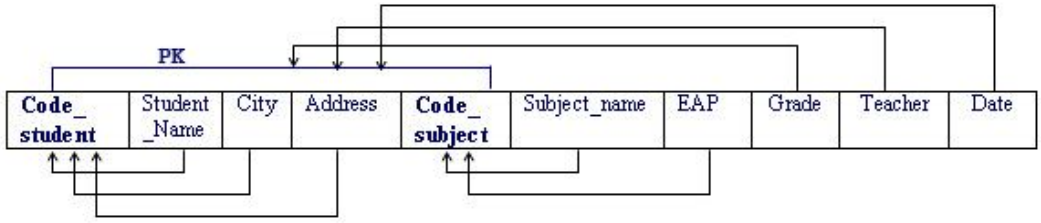
Esimene normaalkuju (1NF)

Andete normaliseerimisprotsess algab andmete viimisega **esimesele normaalkujule**. Kui andmeid on võimalik esitada tabeli kujul, siis on need andmed vähemalt esimesel normaalkujul, iga relatsioonilise andmebaasi tabel on automaatselt esimesel normaalkujul.

Tabelis on jaotatud võti (**Code_student, Code_subject)**. Kui jälgida selle tabeli välju, siis näeme, et osa nendest väljadest on seotud tugevamini ühe võtme komponendiga ja osa võtme

teise komponendiga. Tabelis on olemas **funktsionaale** (functional dependencies) ja **ülekanduv** (transitive dependency) sõltuvus.

Andmetabelites on tihti üks veerg (B) **funktsionaalses sõltuvuses** teisest veerust (A), st. veeru A igale väärtusele vastab mingi väärtus veerus B. Veerg A **identifitseerib** veeru B.



Teine normaalkuju, 2NF

- Andmete viimine teisele normaalkujule tähendab kõigi veergude omavahelise funktsionaalse sõltuvuse väljaselgitamist ning nende sõltuvuste jaoks **eraldi tabeli loomist**, st. ühe tabeli "lõhkumist" mitmeks väiksemaks.
- Esialgu tuleks identifitseerida võimalikud võtmed ja vaadata, millised veerud nn. "suurest tabelist" sellest võtmest sõltuvad ning eraldada välja iseseisvad tabelid (joonisel on üks võimalikke võtmeid ilmselt **tudengi_kood**, millest sõltuvad **Nimi** ja **aadress**).

Korduvad veerud (duplicated columns) tudengi_kood on salvestatud nüüd korraga kahte tabelisse. Selline andmeveergude kordamine on vajalik, et säilitada tabelites sisalduvate andmete õigsust peale teisele normaalkujule viimist. Tudengi_kood "Subject" tabelis loob informatsiooni selle kohta, millised ained vastavad igale tudengile, nii säilib esimeses tabelis toodud informatsioon.

Andmete liiasuse vähendamine

Loodud relatsioonid andmetabelite vahel võimaldavad säilitada andmete **terviklikkust**, n. pole võimalik lisada uut aineid tudengile, mida ei eksisteeri "Student" tabelis

Võime lisada tabelitesse andmeid, ilma teisi tabeleid muutmata, andmeuuendus on suunatud konkreetsematele objektidele. (n. uue tudengi korral uuendatakse andmeid ainult "Student" tabelis)

Andmete viimine teisele normaalkujule tähendab kõigi veergude omavahelise funktsionaalse sõltuvuse väljaselgitamist ning nende sõltuvuste jaoks eraldi tabeli loomist, st. ühe tabeli "lõhkumist" mitmeks väiksemaks.

Student

Code_ student	Student _Name	City	Address
091235	Pavel	Kohtla-Järve	No address
095421	Anna	Jõhvi	Uus tn. 5

Subject

Code_ subject	Subject_name	EAP
RAR0212	Informaatika II	4
RAR0190	Multimeedia	5
RAR0211	Informaatika I	4
RAR3112	Informaatika II	5

Results

Code_ student	Code_ subject	Grade	Teacher	Date
091235	RAR0212		Kiir	
091235	RAR0190		Peet	
091235	RAR0211	A	Sill	14.01.2011
095421	RAR3112		Kiir	
095421	RAR0190		Peet	

Kolmas normaalkuju, 3NF

Eraldades tabelid teise normaalkuju reeglite järgi võib tekkida võimalus sellest tabelist eraldada veel eraldi tabeleid, juhul kui on tegemist nn. **ülekantud sõltuvusega** (*transitive dependencies*)

St. tekivad järjestikused **üks-ühesed sõltuvused** veergude vahel - veerg **B** on üks-üheselt sõltuvuses veerust **A**, veerg **C** on omakorda üks-üheses sõltuvuses veerust **B**.

Kolmandal normaalkujul olevas andmetabelis sõltuvad kõik veerud võtmeveerust (primary key) ja ainult võtmeveerust.

Andmete organiseerimine kolmandale normaalkujule tähendab ülekanduvate sõltuvuste likvideerimist ühe tabeli mitmeks tabeliks jaotamise teel, üks-üheselt teineteisest sõltuvad veerud on paigutatud eraldi tabelitesse.

Kolmanda normaalkuju pluss – paremini kindlustatud andmete õigsus ja terviklikkus kuna väheneb andmete dubleerimine (n. rühma andmed on nüüd salvestatud ainul ühes tabelis, aga mitte igale tudengile vastavas kirjes).