Loeng 3. Andmete disain (Oracle CASE metoodika põhjal)

Taust

Käesolevas loengus räägime disaini ühest põhilistest tegevustest, andmete disainist. Korralik andmete (andmebaasi) disain ja häälestamine on peamiseks rakenduse piisava töökiiruse ja jõudluse tagamise aluseks. Andmete disain on märksa üldisem ning omab rohkem ühiseid jooni erinevate süsteemide puhul, kui rakendusmoodulite disain, milline on väga spetsiifiline iga konkreetse rakenduse jaoks. Loengu aluseks on võetud andmete disain Oracle CASE metoodika disainifaasis. Enamus siin väljatoodud tegevustest, tulemustest ja eeldustest on omased ka teistele disaini lähenemistele ja meetoditele.

Loengu eesmärgid

1. Anda ülevaade andmete disaini põhilistest tegevustest, nende tegevuste objektidest / tulemustest ning eduka läbiviimise eeldustest
2. Näidata andmete disaini seosed talle eelnevate, järgnevate ning samaaegselt toimuvate arendustegevustega.
3. Käsitleda põhjalikumalt esmast andmete disaini

Loengu plaan

1. Andmete disaini eesmärgid, objektid ja põhitegevused
2. Esmane andmete disain
3. Esmase andmete disaini otsustuspunktid ning nende lahendusvariandid

- Olemil mitu unikaalset identifikaatorit
- Infomudelis sisalduvad supertüüpi ja alamtüüpi olemid
- Infomudel sisaldab kaarega ühendatud suhteid
- Infomudel sisaldab M:M suhteid
- Infomudel sisaldab 1:1 suhteid, mis on mõlemast otsast sama kohustuslikkuse astmega

Andmete disain

Andmete disaini tulemusena peaks loodama analüüsi etapil spetsifitseeritud nõudmistele vastav andmebaas koos kõigi sinna juurde kuuluvate komponentidega:

- segmendid
- andmefailid
- andmetabelid ja veerud
- indeksid
- kitsendused
- andmebaasi trigerid
- jne.

Põhitegevuste loetelu andmete disainis oleks järgmine:

- esmane andmete disain (first-cut data design)
- disaini täpsustamine
  - indekseerimine
  - numbrijada generaatorite kasutamine
  - denormaliseerimine
  - kitsenduste / piirangute (constraints) täpsustamine ja lisamine
  - andmebaasi trigerite lisamine
  - abitabelite lisamine
- andmebaasi häälestamine (eesmärgiks rakenduste optimaalse töökiiruse saavutamine olemasoleva andmete struktuuri juures)
  - serveris hoitavate protseduuride maksimaalne kasutamine
  - andmebaasi objektide paigutuse planeerimine (n. erinevad andmefailid / logifailid eraldi ketastele, et tõsta S/V operatsioonide kiirust)
  - andmebaasi objektide laadimise optimeerimine (seotud objektide organiseerimisega andmeblokkidesse, arvestades andmemahtude hinnanguid)
  - koodi häälestamine (SQL lausete häälestamine / ümberkirjutamine optimaalse töökiiruse saavutamiseks)
- käsufailide genereerimine (reaalse andmebaasi ja selles sisalduvate objektide (tabelid, indeksid, kitsendused, kasutajagrupid, kasutajad, …) loomiseks teadmusbaasi kirjelduste alusel
- käsufailide käivitamine

Esmane andmete disain

Esmane andmete disain on otseselt seotud CASE metoodika kasutamisega. Selle läbiviimine eeldab analüüsi etapi elementide (olemid, atribuudid, unikaalsed identifikaatorid) olemasolu teadmusbaasis, et nende alusel saaks genereerida vastavate andmebaasi elementide definitsioonid samas teadmusbaasis ning reaalsed andmebaasi elemendid produktsioonilises andmebaasis.

Esmane andmete disain on teostatav põhimõtteliselt automaatselt, spetsiaalsete töövahenditega (Oracle Database Design Wizard). Eelduseks on siin kvaliteetse infomudeli olemasolu (analüüsi etapi tulemusena).

Esmase andmete disaini tulemusena luuakse:

- Olemi-suhte mudeli olemitest andmebaasi tabelite definitsioonid (tegelikud objektid reaalses andmebaasis tekivad andmete disaini käigus hilisemas faasis)
- Atribuutidest andmetabelite veergude definitsioonid
- Unikaalsetest identifikaatoritest andmetabelite primaarsed ja unikaalsed võtmed
- Suhetest välisvõtmed

Vaadeldava teisenduse tulemus pole määratud täiesti üheselt. Projekteerijapoolsed otsustuskohad tekivad esmase andmete disaini käigus järgnevatel juhtudel:

1. **Olemite korral, mille jaoks on kirjeldatud mitu unikaalset identifikaatorit**, tuleks otsustada, millisest tekitatakse vastava andmetabeli primaarne võti ja millised saavad unikaalseteks võtmeteks. Lähtuda tuleks järgnevast:
   1) Primaarsed võtmed ei luba neis sisalduvates veergudes määramatusi, unikaalsed lubavad. Seega ei saa primaarset võtit luua unikaalsest identifikaatorist, mis sisaldab mittekohustuslikke atribuute.
   2) Primaarse võtme veergude väärtused antakse edasi sõltuva tabeli välisvõtme veergudesse. Seetõttu tuleks primaarne võti luua unikaalsest identifikaatorist, mis koosneb võimalikult vähestest ja/või võimalikult lühikestest atribuutidest (tabeli pikkus minimaalseks)

2. **Infomudelis sisalduvad supertüüpi ja alamtüüpi olemid.**
   Sellise konstruktsiooni lahendamiseks on mitu teed ning tuleb leida spetsifitseeritud nõudmistega kõige paremini sobiv lahendus. Valik on järgmine:

2.1. Realiseerida supertüüpi olem ja tema kõik alamtüüpi olemid ühe andmetabelina:

- , milline nimetatakse lähtudes supertüüpi olemist,
- , sisaldab veergudena kõiki supertüüpi olemi atribuute ja kõigi alamtüüpi olemite atribuute
- , mille kõik alamtüüpi olemite atribuutidest tekitatud veerud on mittekohustuslikud (võivad olla määramata, tühjad), vaatamata sellele, kas vastav atribuut teda hõlmavas olemis oli kohustuslik või mitte.
- , millesse lisatakse automaatselt veerg, mis näitab iga rea kuulumist kindlat alamtüüpi olemi esinemiste hulka. Veeru väärtus tuletatakse vastavat alamtüüpi olemi lühendnimest.

2.2. Tekitada iga alamtüüpi olemi kohta üks andmetabel. Siis hakkab iga tekkinud tabel sisaldama veerge, millised vastavad selle alamtüüpi olemi atribuutidele ning veerge, mis on tekkinud supertüüpi olemi atribuutidest (on ühised kõigi alamtüüpi olemite jaoks).

2.3. Luua eraldi andmetabel supertüüpi olemi ning iga alamtüüpi olemi baasil. Antud juhul:

- sisaldab supertüüpi olemist tekkinud andmetabel vaid supertüüpi olemi atribuutidest loodud veerge,
- alamtüüpi olemitest tekkinud andmetabelid sisaldavad nii veerge, mis on loodud vastavat alamtüüpi olemi atribuutidest, kui ka veerge, mis on loodud supertüüpi olemi atribuutidest (on ühised kõigile alamtüüpi olemitele).

2.4. Luua eraldi tabelid supertüüpi olemi ning iga alamtüüpi olemi baasil, kuid nii, et iga tabel saab kaasa ainult talle vastava olemi atribuudid. Selleks:

- ühendatakse alamtüüpi olemitest tekkinud tabelid 1:1 suhete abil supertüüpi olemist tekkinud tabeliga,
- tekitatud suhted ühendatakse kaarega supertüüpi olemi poolses otsas, s.t., et reaalselt tohib korraga eksisteerida vaid üks kaarega seotud suhetest (supertüüpi olemist tekkinud tabelis tekitatakse iga suhte kohta välisvõtme veergude komplekt, milledest peab korraga väärtust omama 1 ja ainult 1 veergude komplekt). Kaare loogika, mis garanteeriks korraga vaid 1 suhte esinemise, tuleb kirjutada projekteerijal.

CASE metoodika annab soovitused selle kohta, mida võiks pidada silmas toodud variantide hulgast valiku tegemisel :

Varianti 2.1. soovitatakse kasutada juhul, kui:

- alamtüüpi olemitel on palju ühiseid ja vähe erinevaid atribuute (enamus atribuute tuleneb supertüüpi olemi tasandilt),
- enamus suhteid lähtub supertüüpi olemi tasandilt (on ühised kõigile alamtüüpi olemitele,
- enamus kontrolle, väärtustamisi tabelite tasandil hõlmab kõiki (või enamust) alamtüüpi olemitest tekkinud tabeleid,
- enamus aruandeid käib ühtviisi kõigi alamtüüpi olemite kohta,
- andmete uuendamine alamtüüpi olemitest tekkinud tabelites toimub sarnastes tingimustes, samadel töökohtadel, samadel põhjustel.

Variandi 2.2. kasutamist soovitatakse eelmisele vastupidiste tingimuste korral, eelkõige juhul, kui erinevat alamtüüpi olemi ridade muutmine toimub erinevates olukordades, erinevatel töökohtadel.

Varianti 2.3. soovitatakse kasutada juhul, kui eksisteerib supertüüpi olemi esinemisi, mida ei saa klassifitseerida ühegi alamtüüpi olemi juurde (ei ole välja toodud alamtüüpi MUU…).

Variandist 2.4. on soovitatud lähtuda juhul, kui tahetakse alamtüüpi olemi kohustuslikest atribuutidest disaini käigus kohe luua ka kohustuslikud veerud vastavates andmetabelites.

3. **Infomudel sisaldab suhteid, mis on seotud kaarega** (s.t. ühes suhte kaarepoolse olemi esinemises saab korraga määratud olla maksimaalselt üks kaarega seotud suhetest).

Infomudeli kaarte lahendamiseks andmetabelites on projekteerijal 2 alternatiivi:

3.1.Täieliku / tõelise kaare (explicit arc) lahendus. Sel juhul:

- Tekitatakse iga kaarega hõlmatud suhte kohta eraldi veergude grupp suhte kaarepoolses otsas tekkivas tabelis. Veerud tekitatakse välisvõtmetest, millised vastavad suhte teises otsas olevate tabelite primaarvõtmetele. Vastavad välisvõtmed ei pea kuuluma samasse andmegruppi (sama andmetüübiga, sama pikkusega, samade omadustega veerud).
- Kaare loogika (korraga tohib olla määratud vaid 1 veergude komplekt) tuleb projekteerijal eraldi kirjutada (kontrollkitsendused, andmebaasi trigerid võimaldavad seda teha)

3.2. Ühise / üldise kaare (generic arc) lahendus:

- Tuleb täielikult projekteerijal ise luua (Database Design Wizard seda tüüpi kaare lahendust ei toeta)
- Sel juhul tekitatakse kõigi kaarega hõlmatavate suhete kohta 1 ühine veergude komplekt. Eeldatakse, et hõlmatavate välisvõtmete veerud kuuluvad ühte ja samasse andmegruppi.
- Tuleb lisada suhte kaarepoolsest otsast tekitatud tabelisse 2 uut kohustuslikku tüüpi (peavad alati omama väärtust) veergu:
  - Neist 1 sisaldab viidet suhte teises otsas oleva andmetabeli nimele.
  - Teine sisaldab sama andmetabeli primaarvõtit (kui primaarvõtmed on unikaalsed üle kõigi kaarega hõlmatud suhetega seotud andmetabelite, võib tabeli nime ära).

Kuna viimases lahendusvariandis tekkivad andmetabelid ei sisalda tegelikke välisvõtme veerge ja nendega seotud kitsendusi, mis lubaksid andmete täielikkust kontrollida automaatselt serveri tasemel, on soovitav infomudeli kaared lahendada esimesest variandist lähtudes.

4. **Olemi-suhte mudel sisaldab mitu mitmele (M:M) suhteid**.
Sellisel juhul on võimalik:

4.1.Lasta esmase andmete disaini käigus automaatselt tekitada M:M suhte asemele vahetabel, mis sisaldab kummagi M:M suhtega seotud tabeli primaarvõtmetest tekitatud välisvõtme veerge.

Seda meetodit on soovitav kasutada juhul, kui tekkiv vahetabel ise mingit lisainformatsiooni ei kanna, vaid on ainult siduvaks lüliks kahe M:M suhtega seotud andmetabeli vahel. Vastasel juhul tuleks lähtuda soovitusest 4.2.

4.2.Pöörduda tagasi olemite juurde (analüüsi etapp) ning asendada seal M:M suhted üks mitmele suhetega, lisada puuduvad atribuudid.

5. **Infomudel sisaldab üks ühele (1:1) suhteid**, millised on mõlemast otsast sama kohustuslikkuse astmega (kohustuslikud või mittekohustuslikud). Siin ei ole esmase andmete disaini tulemus üheselt ette määratud ning soovitud lõpptulemuse saamine eeldab projekteerija poolset otsustust, kumba vaadeldava suhtega seotud olemitest lugeda primaarseks (peremees-olemiks) ja kumba sõltuvaks olemiks ehk kumma andmetabeli (peremees) primaarvõtmest tekitatakse teise tabeli välisvõtme veerud (sõltuv tabel).