

Hajus andmebaas

Teema 12

Informatsiooni saared

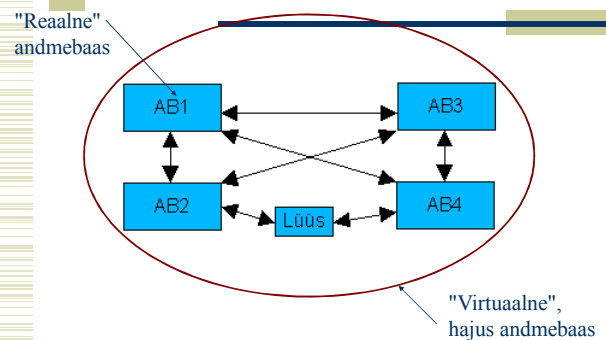
- Andmed pole vajalikul hetkel kättesaadavad.
- Andmete kvaliteet võib olla halb, sest sama juhtumi kohta võivad erinevates andmebaasides olla vastuolulised andmed.



Andmete riskasutus

- Andmete riskasutus on andmete ülekandmine ühest andmekogust teise või mitmes andmekogus sisalduvate andmete ühine infotehnoloogiline töötlemine. (Andmekogude seadus)
 - Kehtis kuni 31.12.2007. Peale selle kehtetuks muutumist reguleerib andmekogude valdkonda avaliku teabe seadus (<https://www.riigiteataja.ee/akt/1220320110107/leia?kehtiv>).

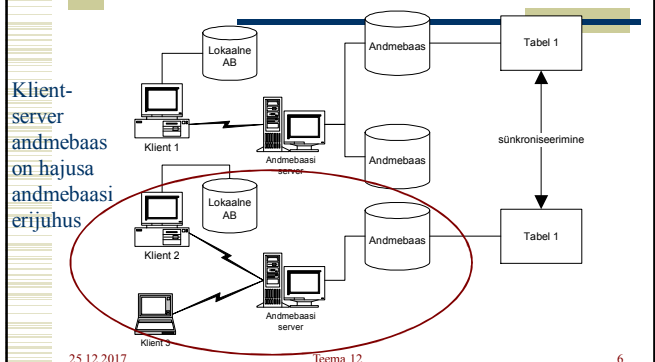
Hajus andmebaas



Hajus andmebaas (2)

- Kasutajale ühtse andmebaasina paistev võrgus mitme serveri vahel hajutatud andmebaasikogum (Vallaste e-teatmik, 2017)
- Füüsiliselt detsentraliseeritud andmebaas, mida andmebaasihaldur juhib nii, et kasutaja saab andmebaasist loogiliselt tsentraliseeritud vaate (IT terministandardi projekti (1998-2001) sõnastik)

Hajus andmebaas (3)



Andmete paiknemine

- Hajusa andmebaasi moodustavad andmebaasid (*komponent-andmebaasid*) võivad paikneda:
 - samas füüsilises arvutis (eriti arendamise perioodil),
 - erinevates serverarvutites, kuid samas geograafilises asukohas,
 - erinevates geograafilistes asukohtades.



Mis mõjutab hajusate andmebaaside kasutuselevõttu?

- Organisatsioonidel on *globaalne* haare ja nende allüksused paiknevad hajutatult.
- Rakenduste (ja andmebaasi) *kasutajate* hulk muutub üha suuremaks. *Andmemahud* suurenevad.
- Riistvara* muutub võimsamaks ja odavamaks.
- Arvutivõrgud* muutuvad kiiremaks ja kommunikatsioonikulud langevad.
- Arvutisüsteem peaks ideaalis olema *skaleeritav* – peaks olema võimalik sõlmede lisamine/eemaldamine vastavalt vajadusele.

Nt meediaettevõtteid, filmikompaniid, arvutimängude tootjad.

Hajusa andmebaasi eelised

- Andmebaas peegeldab **organisatsiooni struktuuri**.
 - Allorganisatsioonidel lokaalsed andmebaasid.
 - Juhtkond saab teha globaalseid päringuid.
- Suurendab **lokaalset autonoomiat**, sest andmeid saab hoida nende kasutajatele füüsiliselt lähedal.
 - Lokaalsetele andmetele saab rakendada kohalikult välja töötatud reegleid (nt pääsuõigused).

Hajusa andmebaasi eelised (2)

- Parandab **käideldavust**.
 - Ei ole enam ühte kesksert serverit või kommunikatsioonikanalit, mille rivist välja langemine halvab kogu süsteemi töö.
- Parandab hea disaini korral operatsioonide **töökiirust**.
 - Andmed saab paigutada oma põhikasutajate lähedale.
 - Töökoormus jaotub erinevate serverite vahel.

Hajus andmebaas suurendab süsteemi keerukust

- Hajususe varjamine kasutajate eest.
- Andmete terviklikkuse tagamine üle erinevate komponent-andmebaaside.
- Andmebaasiobjektide nimetamine.
- Andmekäitluskeele lausete töökiiruse optimeerimine.
- Transaktsioonide juhtimine.
 - lukustamine, kinnitamine, tühistamine jne.
- Erandite haldamine.

On suhtelised mõisted

Hajusa andmebaasi homogeensus/heterogeensus

- Homogeenne** hajus andmebaas
 - Kõik komponent-andmebaasid on loodud kasutades sama andmebaasisüsteemi (nt Oracle)
 - Kõik komponent-andmebaasid on loodud kasutades sama tüüpi andmebaasisüsteemi (nt SQL-andmebaasisüsteem)
- Heterogeenne** hajus andmebaas
 - Komponent-andmebaasid on loodud kasutades erinevat tüüpi andmebaasisüsteeme (nt SQL-andmebaasisüsteem ja objektorienteeritud andmebaasisüsteem)

Andmete kasutamine hajusas andmebaasis

- ♦ Iga komponent-andmebaas kasutavad:
 - *lokaalsed* rakendused, mis kasutavad ainult antud komponent-andmebaasi,
 - *globaalsed* rakendused, mis kasutavad mitmes erinevas komponent-andmebaasis asuvaid andmeid.

Semantiline heterogeensus

- ♦ Erinevates hajusasse andmebaasi kuuluvates andmebaasides võivad *sama nimega* tabelites ja veergudes olla *erineva tähendusega andmed*.
- ♦ Sama tähendusega andmed võivad erinevates süsteemides olla *erinevalt kodeeritud*.
 - ISO soo koodid: 0, 1, 2, 9
 - Soo koodid eestlaste andmebaasis: M, N, T
 - Soo koodid inglaste andmebaasis: M, F, U
 - Soo andmeid pole eraldi esitatud vaid on kodeeritud isikukoodi
 - ...

Semantiline heterogeensus (2)

- ♦ Erinevates andmebaasides võivad samatüübilised mõõtmisandmed (nt temperatuurid) olla esitatud kasutades *erinevaid mõõtühikuid* või samu mõõtühikuid, kuid erinevat *täpsust*.
 - Erinevates andmebaasides rahasummad erinevates valutades.
- ♦ Erinevates andmebaasides on samasisulised andmed *erinevate andmetüüpidega*.
 - Erinevad andmebaasisüsteemid võimaldavad kasutada erinevaid andmetüüpe.
- ♦ Erinevused *NULLide* kasutamises.

Nõuded hajusale andmebaasile

- ♦ Date, C. J. (1990). What Is a Distributed Database System? In: *Relational Database Writings 1985–1989*. Reading, Mass.: Addison-Wesley.
- ♦ Põhiprintsiip + 12 nõuet

Hajusa andmebaasi põhiprintsiip

- ♦ Kasutajale peab hajus andmebaas paistma **samasugune, kui mittehajus andmebaas**.
 - Kasutaja ei tohi aru saada, et ta kasutab hajusat andmebaasi (ingl *transparency*).
 - Kasutaja – andmekäitluskeele lausete looja/käivitaja.
- ♦ *Eesmärk*: kaitsta investeeringuid, mis on tehtud olemasolevatesse andmebaasi kasutavatesse programmidesse.

Hajusa andmebaasi põhiprintsiip (2)

Kasutaja nägemus andmetest

aine_kood	nimetus	ap	teaduskond
IDU0220	Andmebaasid I	3,5	I
IDU0230	Andmebaasid II	3,5	I
TAM3460	Aastaaruanduse analüüs	3,5	T
TAF3261	Raamatupidamisarvestus I	3,5	T
IDN5420	Andmeanalüüs	3,5	I

Kasutaja kirjutab lause:

```
SELECT aine_kood FROM
Aine WHERE ap=3.5;
```

Andmed TTÜ peamaja serveris – teaduskond="I"

aine_kood	nimetus	ap	teaduskond
IDU0220	Andmebaasid I	3,5	I
IDU0230	Andmebaasid II	3,5	I
IDN5420	Andmeanalüüs	3,5	I

Teaduskond = 'I'

Andmed TTÜ majandus- ja teaduskonna maja serveris – teaduskond="T"

aine_kood	nimetus	ap	teaduskond
TAM3460	Aastaaruanduse analüüs	3,5	T
TAF3261	Raamatupidamisarvestus I	3,5	T

Teaduskond = 'T'

Nõuded hajusale andmebaasile (2)

- Hajusa andmebaasi moodustamiseks kasutatavad serverid peavad olema nii **autonoomsed kui vähegi võimalik**.
 - Serveris X toimuv operatsioon peaks võimalikult vähe sõltuma mõnest teisest serverist.
 - Pole alati võimalik – näiteks mitut serverit hõlmav kitsenduste kontroll.
- Ei ole** kasutusel ühte **keskset** serverit.
 - Kõik serverid on võrdsed.
- Serveri lisamine/eemaldamine või andmete paigutuse muutumine ei tohiks tekitada süsteemi kui terviku **töö katkestust**.
 - Süsteemi **usaldusväärsus** ja **kättesaadavus** peaks paranema.

25.12.2017

Teema 12

19

Nõuded hajusale andmebaasile (3)

- Kasutaja ei pea olema teadlik **andmete asukohast**.
 - Kasutaja peab pääsema talle ettenähtud andmetele ligi mistahes serveri vahendusel, **sõltumata** andmete füüsilisest **asukohast**.
- Kasutaja ei pea olema teadlik sellest, millised **killud** on andmebaasis loodud.

25.12.2017

Teema 12

20

Nõuded hajusale andmebaasile (4)

- Kasutaja ei pea olema teadlik sellest, millised **koopiad** on andmeelementidest loodud.
- Andmebaasis peab olema võimalik täita **lauseid**, mis kasutavad (loevad/muudavad) mitme **erineva serveri** hallatavaid **andmeid**.

25.12.2017

Teema 12

21

Nõuded hajusale andmebaasile (5)

- Hajusas andmebaasis peab saama kasutada transaktsioone.
 - Lokaalne transaktsioon** töötab ühe serveri piires.
 - Globaalne (hajus) transaktsioon** kasutab mitme erineva serveri hallatavaid andmeid.
 - Kahefaasilise* kinnitamise protokoll
 - Kolmefaasilise* kinnitamise protokoll
 - Lokaalse ja globaalse transaktsiooni algatamine ja lõpetamine peaks käima samal viisil.
 - Andmete samaaegse kasutamise juhtimiseks kasutatakse enamasti **lukustamist**.

25.12.2017

Teema 12

22

Nõuded hajusale andmebaasile (6)

- Hajusa andmebaasi moodustamiseks kasutatavad serverid peavad olema võimelised töötama erinevatel:
 - riistvara platvormidel,
 - operatsioonisüsteemidel,
 - erinevatel tehnoloogiatel põhinevatel arvutivõrkudel,
 - andmebaasisüsteemidel.
 - Hajus andmebaas koosneb lokaalsetest andmebaasidest, mis võivad olla loodud erinevates andmebaasisüsteemides / andmebaasisüsteemi versioonides ja võimalik, et kasutavad erinevaid andmemudeleid (nt relatsiooniline, hierarhiline).

25.12.2017

Teema 12

23

Nõuded hajusatele andmebaasidele

- Hajusate andmebaasidega seoses eksisteerib kolm omavahel seotud süsteemset nõuet.
 - Terviklikkus** (*consistency*) – kõikides andmete koopiates erinevates sõlmedes (komponent-andmebaasides) on klientidel lugemiseks alati samad andmed.
 - Käideldavus** (*availability*) – iga töökorras oleva sõlme poole pöördumine peab õnnestuma.
 - Järelkult kõik kliendid peavad saama igal ajahetkel andmeid lugeda ja kirjutada, sõltumata sellest, millise töökorras sõlme poole nad pöörduvad.
 - Kuna võrgukatkestus võib kesta kuitahes kaua, ei või sõlmed vastamisega viivitada kuni katkestus kõrvaldatakse.

25.12.2017

Teema 12

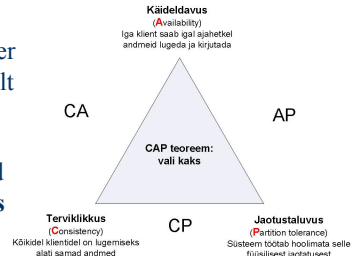
24

Nõuded hajusatele andmebaasidele (2)

- ♦ **Jaotustaluvus** (*partition tolerance*) – ükski viga, peale täieliku võrgukatkestuse, ei tohi tingida vigu andmete lugemises või muutmises. Süsteem peab töötama korrektselt ka siis, kui võrguühendus osade sõlmede vahel ei tööta.

CAP teoreem

- ♦ CAP teoreemi (sõnastas Eric Brewer 2000. aastal) kohaselt saab **hajusas** andmebaasis olla samaaegselt täidetud **maksimaalselt kaks nõuet kolmest**.



http://viktor.ld.ttu.ee/animatsioonid/animation_cap_theorem/

Terviklikkus + käideldavus (CA)

- ♦ Ühes komponent-andmebaasis muudetakse andmeid. See muudatus tuleb teha ka kõigis teistes komponent-andmebaasides, tagamaks, et kõigil klientidel on asukohast sõltumata lugemiseks samad andmed (**terviklikkuse nõue**).
- ♦ Muudatus tuleb teha kohe – ilma süsteemi üldist tööd katkestamata (**käideldavuse nõue**).
- ♦ Eelnev on võimalik vaid juhul, kui võrguühendustes ei teki katkestusi.

Terviklikkus + käideldavus (CA) (2)

- ♦ Kui tegemist on **mittehajusa** andmebaasiga, mis pole erinevate sõlmede vahel jaotunud, siis pole **jaotustaluvus** probleemiks.
 - Tagada saab nii **terviklikkuse** kui ka **käideldavuse** (CA).
- ♦ Hajusalt andmebaasilt oodatakse tüüpiliselt **jaotustaluvust** (P).
 - Tuleb otsustada kas ja kui palju ohverda **käideldavust** ja **terviklikkust**.

Terviklikkus + jaotustaluvus (CP)

- ♦ Ühes komponent-andmebaasis muudetakse andmeid. See muudatus tuleb teha ka kõigis teistes komponent-andmebaasides, tagamaks, et kõigil klientidel on asukohast sõltumata lugemiseks samad andmed (**terviklikkuse nõue**).
- ♦ Süsteem peab taluma katkestusi serverite vahelises võrguühenduses (**jaotustaluvuse nõue**).
- ♦ Eelnev tähendab, et süsteem ei saa tagada käideldavust.
 - Mittetäieliku võrgukatkestuse korral tuleb oodata katkestuse parandamist, et andmemuudatused teistesse komponent-andmebaasidesse üle kanda. Senikaua ei saa andmebaasi kasutada, sest andmed ei ole terviklikud.
 - Teine variant on loobuda algse muudatuse tegemisest.

Käideldavus + jaotustaluvus (AP)

- ♦ Andmebaas peab olema alati lugemiseks ning muutmiseks kasutatav (**käideldavuse nõue**) ning seda isegi siis, kui tekivad katkestused serverite vahelises võrguühenduses (**jaotustaluvuse nõue**).
- ♦ Eelnev on võimalik vaid juhul, kui süsteem ei pea tagama andmete terviklikkust igal ajahetkel, vaid võib seda teha viivitusega.
 - Andmemuudatus ühes komponent-andmebaasis kantakse üle teistesse komponent-andmebaasidesse siis, kui selleks on võimalus (võrguga ei ole probleeme).

Käideldavus + jaotustaluvus (AP) (2)

- ♦ Öeldakse, et andmebaas muutub lõpuks terviklikuks – *eventual consistency*.
- ♦ Analoožiaks on riigieelarve, mis ei ole tasakaalus mitte igal aastal, vaid **üle mitme aasta** (selle kohta on termin "struktuurne eelarvetasakaal").

Terviklikkus (C) vs. käideldavus (A)

- ♦ "Traditsioonilised" andmebaasisüsteemid eelistavad C-d (A hinnaga)
 - **ACID** – *Atomicity, Consistency (pole päris sama kui CAP teoreemi C), Isolation, Durability*
- ♦ NoSQL (Not Only SQL) süsteemid üritavad esmajoones tagada A-d (C hinnaga)
 - **BASE** – *Basically Available, Soft state, Eventual consistency*

Terviklikkus (C) vs. käideldavus (A) (2)

- ♦ Ühe ja sama süsteemi erinevate:
 - allsüsteemide,
 - operatsioonide,
 - andmete või
 - kasutajate korral
- ♦ võib valik **käideldavuse** ja **terviklikkuse** vahel olla erinev.

Terviklikkus (C) vs. käideldavus (A) – näide

- ♦ Gmail
 - registreerib kirja loetuks asünkroonselt (elistab A-d C-le)
 - kirja saatmine nõuab sünkroonselt andmete muutmist erinevatel serveritel (elistab C-d A-le)

Dean, J., 2009. Designs, Lessons and Advice from Building Large Distributed Systems. [WWW]
<https://www.cs.cornell.edu/projects/ladis2009/talks/dean-keynote-ladis2009.pdf> (17.12.2017)

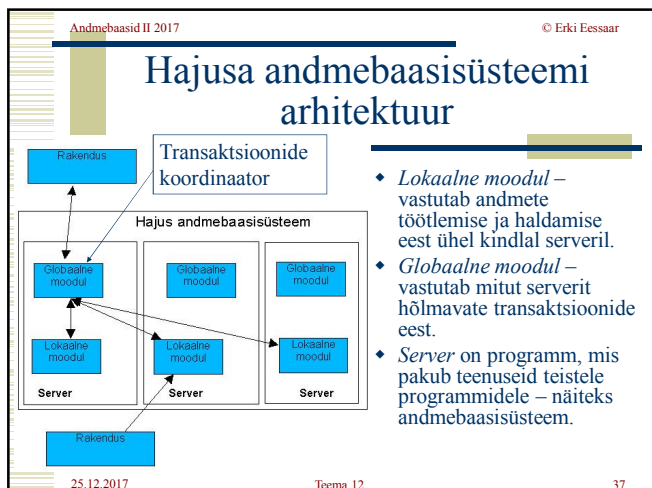
Hajus andmete paiknemine vs. andmed ainult ühes sõlmes

- ♦ Terviklikkus CAP teoreemis tähendab, et sõlmedes on kooskõlalised andmed.
 - Terviklikkus ACID järgi tähendab andmete kooskõla kitsendustega.
- ♦ Kui andmed on ainult ühes sõlmes, siis süsteemil CA omadused.
 - Samas ei pruugi süsteemil olla *kõrge käideldavus* (võib olla pikalt maas, kuid kui ärkab, siis C ja A tagatud).



Hajus andmebaasisüsteem

- ♦ Hajus andmebaasisüsteem on omavahel suhtlevate ja *partnerluses* olevate andmebaasisüsteemide kogum.
- ♦ Partnerlus tähendab, et andmebaasisüsteemi A kasutaja saab kasutada ka teiste partnerluses osalevate andmebaasisüsteemide hallatavaid andmebaase täpselt nii, nagu nad oleksid A abil loodud andmebaasid.
- ♦ Partnerluses osalevatel andmebaasisüsteemidel peab olema täiendav tarkvarakomponent, mis tagab partnerluse toimimise.



Andmebaasid II 2017 © Erki Eessaar

Andmete paigutamise strateegiad

- ♦ **Tsentraalne** – Kogu andmebaas on ühe serveri kontrolli all.
- ♦ **Killustamine** – Andmebaas on jaotatud üksteisega mittekattuvateks osadeks (kildudeks) ning need osad on erinevate serverite kontrolli all.
- ♦ **Täielikud koopiad** – Iga server haldab täielikku koopiat andmebaasist. Muudatusi tuleb sünkroniseerida.
- ♦ **Valikulised koopiad** – Kombinatsioon killustamisest, koopia tegemisest ja tsentraliseerimisest.

25.12.2017 Teema 12 38

Andmebaasid II 2017 © Erki Eessaar

Strateegiate võrdlus

	Andmete lähedus kasutajale	Käideldavus	Jõudlus (operatsioonide töökiirus)	Salvestuse maksumus	Kommunikatsiooni kulud
Tsentraalne	Madalaim	Madalaim	Mitte-rahuldav	Madalaim (sest andme-elementidest pole koopiaid)	Kõrgeim
Killustatud	Kõrge *	Rahuldav (madal üksiku andme-elementi seisukohast, kõrge süsteemi seisukohast)	Rahuldav *	Madalaim (sest andme-elementidest pole koopiaid)	Madal *
Täielikud koopiad	Kõrgeim	Kõrgeim	Lugemise puhul parim	Kõrgeim	Kõrge andmete muutmise puhul, madal andmete lugemise puhul
Valikulised koopiad	Kõrge *	Rahuldav (madal üksiku andme-elementi seisukohast, kõrge süsteemi seisukohast)	Rahuldav *	Keskmine	Madal *

* juhul, kui andmebaas on hästi disainitud

25.12.2017 Teema 12 39

Andmebaasid II 2017 © Erki Eessaar

Kommentaare andmete paigutamise strateegiate kohta

- ♦ Killud tuleks paigutada võimalikult lähedale andmete kasutajale, et minimeerida *kommunikatsioonikulusid*.
- ♦ Andmete koopia kasutamine parandab süsteemi *töökindlust*.
- ♦ Andmete koopia kasutamine suurendab süsteemi *koormust*, sest koopiaid tuleb värskendada.

25.12.2017 Teema 12 40

Andmebaasid II 2017 © Erki Eessaar

Kommentaare andmete paigutamise strateegiate kohta (2)

- ♦ Andmete killustamine ja koopia loomine muudab raskemaks andmete *terviklikkuse tagamise*.
 - Et andmed vastaksid reeglitele ja kõigis sõlmedes oleksid samad andmed.

25.12.2017 Teema 12 41

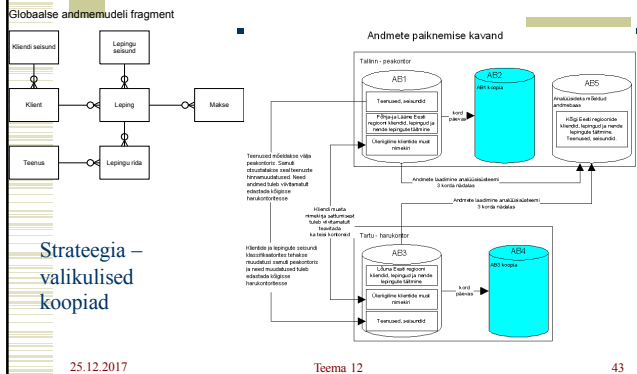
Andmebaasid II 2017 © Erki Eessaar

Kommentaare andmete paigutamise strateegiate kohta (3)

- ♦ Killustamise strateegia korral läheb erinevates geograafilistes asukohtades asuvate serverite jaoks vaja eraldi salvestusseadmeid ning andmekandjaid ja see tõstab võrreldes tsentraalse strateegiaga salvestuse hinda.

25.12.2017 Teema 12 42

Andmete paigutuse näide



25.12.2017

Teema 12

43

Horisontaalne ja vertikaalne killustamine

- ◆ **Horizontaalne.**
 - **K1**=Aktuaalses seisundis lepingute andmed (number, seisund, sõlmimise aeg, kirjeldus) *server 1*
 - **K2**=Mitteaktuaalses seisundis lepingute andmed (number, seisund, sõlmimise aeg, kirjeldus) *server 2*
- ◆ **Vertikaalne (sageli kooskasutatavad andmed).**
 - **K1**=Töötajate isikuandmed (*isikukood, eesnimi, perenimi, aadress, e_mail*) *server 1*
 - **K2**=Töötajate palgaandmed (*isikukood, põhipalk*) *server 2*

25.12.2017

Teema 12

44

Segastrategia

- ♦ **K1**= Töötajate isikuandmed (isikukood, eesnimi, perenimi, aadress, e_mail) *server 1*
- ♦ **K3**= Töötajate palgaandmed, kui põhipalk on 2000 EUR või vähem (isikukood, põhipalk) *server 1*
- ♦ **K2**= Töötajate palgaandmed, kui põhipalk on üle 2000 EUR (isikukood, põhipalk) *server 2*

25.12.2017

Teema 12

45

Tuletatud killud

- ◆ Tabeli T tuletatud kild põhineb tabeliga T 1:M seotsetüübi kaudu seotud tabeli horisontaalsel killustamisel.
- ◆ Näiteks: [Klient]-1-----0..*-[Leping]
- ◆ Tabeli *Leping* killud.
 - **K1**=Seisundis "mustas nimekirjas" olevate klientide lepingud.
 - **K2**=Seisundis "mustas nimekirjas" mitteolevate klientide lepingud.

25.12.2017

Teema 12

46

Nõuded killustamise tulemusele



- ◆ Täielikkus.
 - Iga andmeelement vähemalt ühes killus.
- ◆ Taastatavus.
 - Kildude põhjal peab saama kõik tabelid täpselt taastada.
- ◆ Kattumatus.
 - Andmeelement (v.a kandidaatvõti) ei tohi olla mitmes killus.

25.12.2017

Teema 12

47

Killustamine ja taastamine

- ◆ *Horisontaalne* killustamine.
 - Killustamiseks tuleb tabelile rakendada *piirangu* operaatorit. 
 - Kildudest terviku taastamiseks tuleb kildudele rakendada *ühendi* (UNION) leidmise operaatorit.
- ◆ *Vertikaalne* killustamine.
 - Killustamiseks tuleb tabelile rakendada *projektsiooni* operaatorit.
 - Kildudest terviku taastamiseks tuleb kildudele rakendada *ühendamise* (JOIN) operaatorit. 

25.12.2017

Teema 12

48

Veel killustamisest

- ♦ Tabeli võib loomulikult jagada ka rohkem kui kaheks killuks.
- ♦ Tabelis olevate andmete tähendust kirjeldavad andmebaasisüsteemi jaoks tabelile loodud *kitsendused*, mitte tabeli/veergude *nimed* (*L1*, *Lepingud1*, *LBP1*, *Akt_lepingud*,...).
- ♦ Ka killustatud tabelites tuleb luua tabelitele kitsendused.
- ♦ Muutes rida mingis killus, võib olla vajalik selle rea migreerumine teise kildu.

Kitsenduse näide horisontaalselt killustatud tabelis

- ♦ K1=Aktuaalses seisundis lepingute andmed (number, **seisund**, sõlmimise aeg, kirjeldus)
server 1
 - ... CONSTRAINT aktuaalsed_lepingud CHECK (seisund=1) ...
- ♦ K2=Mitteaktuaalses seisundis lepingute andmed (number, **seisund**, sõlmimise aeg, kirjeldus)
server 2
 - ... CONSTRAINT mitteaktuaalsed_lepingud CHECK (seisund=2) ...

Killustamise eelised/puudused

- ♦ Eelised.
 - Andmete lähedus kasutajatele
 - Paralleeltöö võimalus
 - Turvalisus
- ♦ Puudused.
 - Töökiiruse vähenemine
 - Keerukas andmete terviklikkuse kontroll (kitsendused üle erinevate kildude)
 - Andmeelemendi salvestamise maksumus suureneb

Hajusa andmebaasiobjektide võimalik globaalse nimetamise skeem

- ♦ Andmebaasiobjektidel peab olema globaalselt (üle erinevate komponent-andmebaaside) unikaalne nimi.
- ♦ Andmebaasiobjekti (nt baastabeli) globaalse nime *võimalikud* komponendid.
 - Objekti loonud kasutaja identifikaator (serveri piires unikaalne).
 - Serveri identifikaator (globaalselt unikaalne), kus anti käsk objekti loomiseks.
 - Objekti lokaalne identifikaator.
 - Serveri identifikaator (globaalselt unikaalne), kus objekt esialgselt loodi.
- ♦ **Mati@TTY_majandus.Aine@TTY_peamaja**
 - See nimi ei muutu kunagi

Hajusa andmebaasi objektide lokaalne nimetamine

- ♦ Lisaks globaalsele nimele võib lokaalse andmebaasi piires objektile viitamiseks kasutada lokaalset nime (sünonüümi).
- ♦ CREATE SYNONYM Aine FOR
Mati@TTY_majandus.Aine@TTY_peamaja;

Hajusa andmebaasi süsteemikataloogi võimalik asukoht

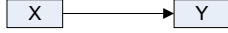
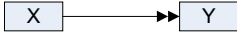
- ♦ Haldab spetsiaalne server.
 - Vastuolu keskse serveri puudumise nõudega.
- ♦ Iga server haldab kogu andmebaasi süsteemikataloogi täielikku koopiat.
 - Vastuolu komponentide autonoomsuse nõudega.
- ♦ Süsteemikataloogi andmete killustamine.
 - Igas server haldab kataloogi, kus on andmed vaid selle serveri hallatava andmebaasi andmebaasiobjektide kohta.
- ♦ Täielike koopiate ja killustamise kombinatsioon.

Vt andmete paigutamise strateegiaid


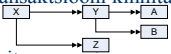
Transaktsioonide haldamine hajusas andmebaasis

- ♦ Globaalne transaktsioon (hajustransaktsioon) koosneb alamtransaktsioonidest.
- ♦ Süsteem peab tagama **ACID** omaduste täidetuse nii globaalsel, kui lokaalsel tasemel.
- ♦ Transaktsioonis osalevad serverid (osapooled):
 - üks transaktsiooni **koordineerija**,
 - üks või rohkem transaktsioonis **osalejat**.
- ♦ Osapooled vahetavad teateid.

Transaktsioonid hajusas andmebaasis

- ♦ *Remote request* 
 - Serverist X üks lause serverisse Y.
 - Lause täidetakse serveris Y ja kasutab andmeid vaid serverist Y.
- ♦ *Remote unit of work* 
 - Serverist X mitu lauset serverisse Y (üks transaktsioon).
 - Lauseid täidetakse serveris Y ja kasutab andmeid vaid serverist Y.
 - Server X otsustab transaktsiooni kinnitamise/tühistamise.

Transaktsioonid hajusas andmebaasis (2)

- ♦ *Distributed unit of work* 
 - Serverist X laused mitmesse erinevasse serverisse.
 - Iga lause kasutab andmeid vaid serverist kus ta täidetakse.
 - Server X otsustab transaktsiooni kinnitamise/tühistamise.
- ♦ *Distributed request* 
 - Serverist X laused mitmesse erinevasse serverisse.
 - Iga lause võib kasutada andmeid mitmest erinevast serverist.
 - Server X otsustab transaktsiooni kinnitamise/tühistamise.

Lukustamine osaliste või täielike koopiate korral

- ♦ Lukkude haldur ühes serveris.
 - Vastuolu keske serveri puudumise nõudega.
- ♦ Lukkude haldur mõnes, aga mitte kõigis serverites.
 - Iga andmelemendi kohta **primaarne koopia**.
 - Andmelemendi lugemiseks või muutmiseks tuleb lukustada selle primaarne koopia.
 - Primaarse koopia omanik vastutab andmete muutmise eest koopiates.

Lukustamine osaliste või täielike koopiate korral (2)

- ♦ Lukkude haldur kõigis serverites.
 - **Read-One-Write-All** e ROWA protokoll.
 - Andmelemendi lugemiseks tuleb panna *jagatud* lukk vähemalt ühele koopiale.
 - Andmelemendi muutmiseks tuleb panna *eksklusiivne* lukk kõigile koopiatele.
 - Andmelemendi lugemiseks/muutmises tuleb panna jagatud/eksklusiivne lukk **enamusele** (üle poole) antud andmelemendi koopiatele.



Hajusate transaktsioonide kinnitamine

- ♦ Globaalne transaktsioon võib muuta andmeid mitmes sõlmes.
- ♦ Kui süsteem peab tagama CAP teoreemi mõttes C, siis transaktsiooni osalised (sõlmed) peavad jõudma *konsensusele*, kas globaalne transaktsioon kinnitada või mitte.
 - Kõigis osalevates sõlmedes tuleb kas kinnitada või jätta kõigis kinnitamata.

Hajusate transaktsioonide kinnitamine (2)

- Leidub erinevaid konsensuse saavutamise protokolle.
- Transaktsiooni **osapooled** (koordinaator ja osalised) vahetavad **teateid**.
 - Protokoll võib arvestada sünkroonse või asünkroonse teadete vahetusega.
- Osapool peab enne teate välja saatmist selle **logima**, et veaolukorrast taastudes oleks "meeles", millised teated on väljastatud.

25.12.2017

Teema 12

61

Sünkroonne vs. asünkroonne teadete vahetus

- Sünkroonne
 - Teate ootamisele saab panna piiraja.
 - Lihtsustab protokoll.
 - Vigu saab avastada kella vaadates, sest teate tähtjaks mitte saamisest võib midagi järeldada.
- Asünkroonne.
 - Saatja töökoormus võis lükata teate saatmise pikalt edasi.
 - Teate tähtjaks mittesaamisest ei saa järeldada viga.
 - Realistlikum kui sünkroonne.
 - Konsensust pole mõnikord võimalik saavutada.

25.12.2017

Teema 12

62

Konsensuse saavutamise protokollid

- Two-Phase Commit (2PC)
 - Blokeeritav** protokoll, sest otsustusprotsess võib jääda toppama, kui juba üksik sõlm kättesaamatu
 - http://viktor.ld.ttu.ee/animations/animation_2PC/
- Three-Phase Commit (3PC)
 - Asünkroonne teadete vahetuse korral võib tekkida olukord, et mõne osalise veast taastumise ja töö jätkamise tulemusena transaktsioon osades sõlmedes kinnitatakse ning osades tühistatakse
 - Andmebaas läheb ebakorrektsesse seis.

25.12.2017

Teema 12

63

Konsensuse saavutamise protokollid (2)

- Paxos
 - Kolmest kõige uuem
 - Annab tõestatavalt korrektse tulemuse asünkroonne teadete vahetuse korral, mis lõpuks muutub sünkroonseks
 - Ei blokeeri ja annab tulemuse, kui *enamik* osalisi on *piisavalt* kaua kättesaadavad
 - 2PC on selle erijuhus

25.12.2017

Teema 12

64

Kahefaasilise kinnitamise protokoll (edukas stsenaarium)

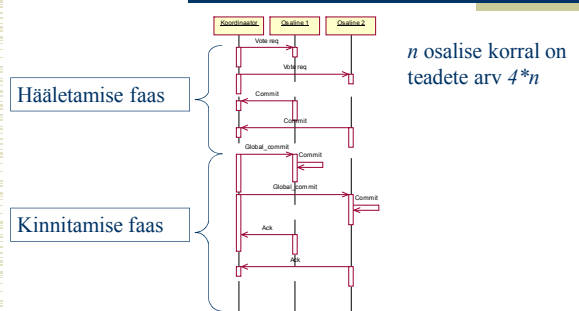
- Hääletamise faas.**
 - Koordinaator palub kõigil osalistel hääletada, kas globaalne transaktsioon kinnitada või tühistada.
 - Osalised teatavad oma otsusest.
- Kinnitamise faas.**
 - Koordinaator saadab otsuse osalistele.
 - Kui kõik poolt, siis **kinnitada**.
 - Kui vähemalt üks vastu, siis **tühistada**.
 - Kui mõni osaline korduvalt ei vasta, siis **tühistada**.
 - Osaline täidab koordinaatori otsuse ja teavitab teda täitmisest.

25.12.2017

Teema 12

65

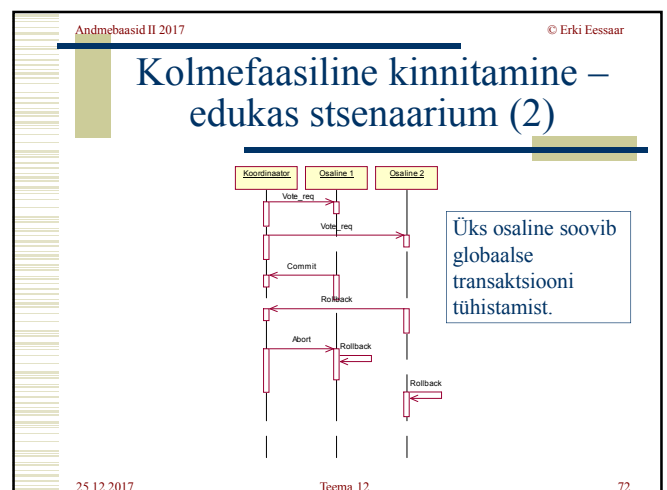
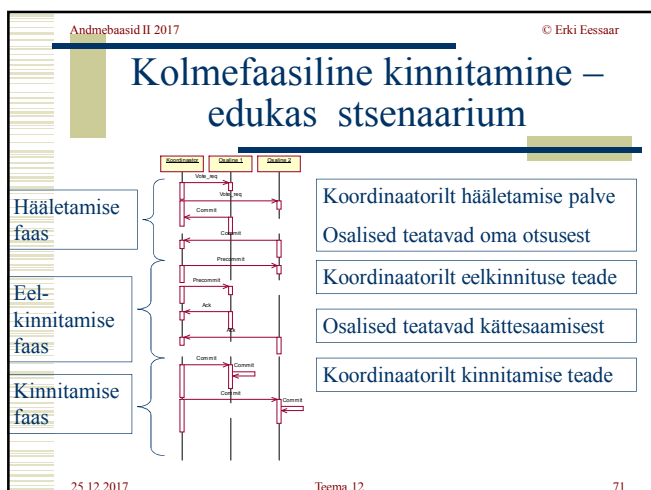
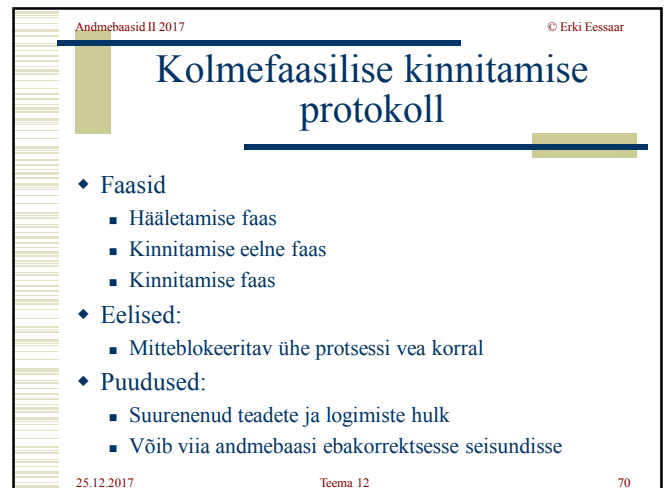
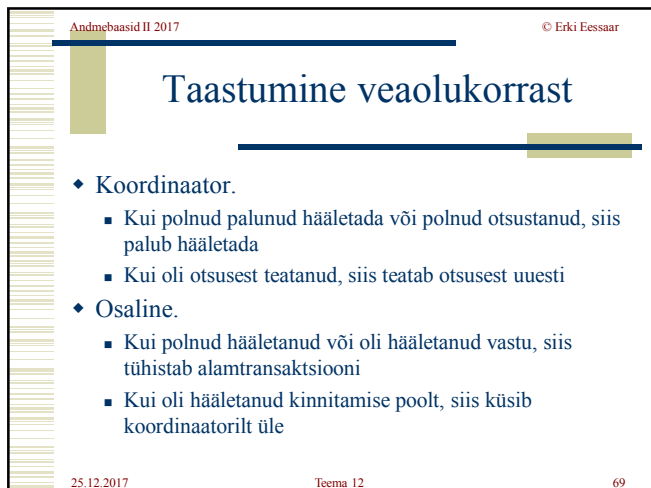
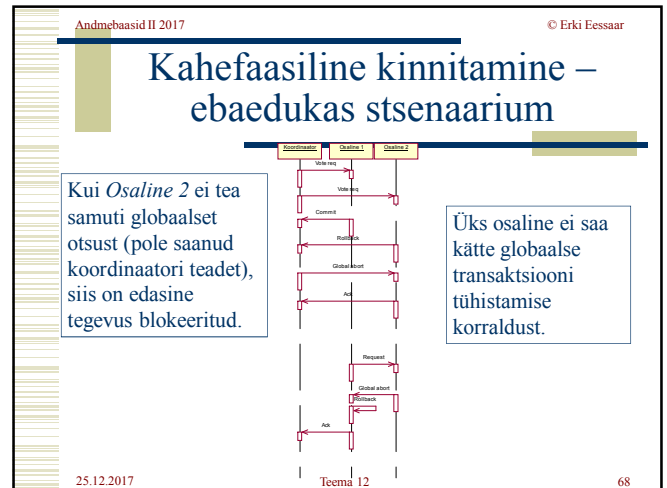
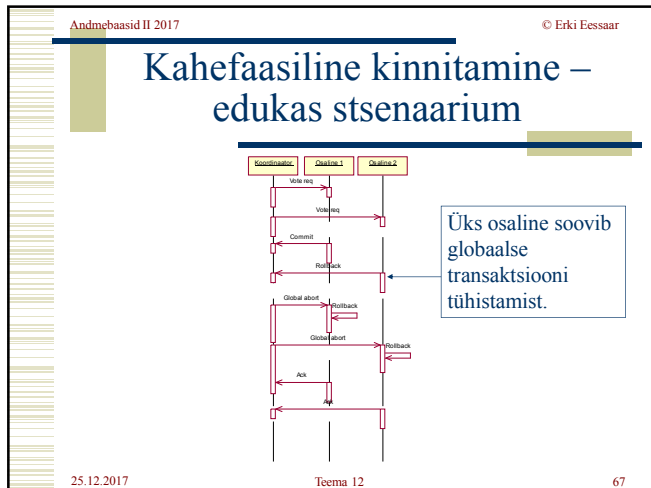
Kahefaasiline kinnitamine – edukas stsenaarium



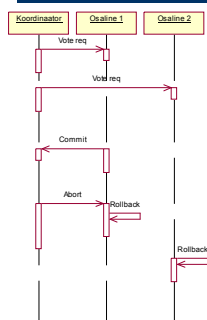
25.12.2017

Teema 12

66



Kolmefaasiline kinnitamine – ebaedukas stsenaarium



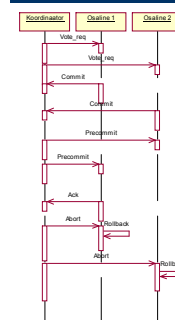
Ühelt osaliselt ei saada otsust, kas globaalne transaktsioon kinnitada või tühistada.

25.12.2017

Teema 12

73

Kolmefaasiline kinnitamine – ebaedukas stsenaarium (2)



Ühelt osaliselt ei saada vastust esialgset kinnitamist nõudvale teatele.

25.12.2017

Teema 12

74

Vigade haldamine hajusas andmebaasis

- ♦ Lisapõhjused vigade tekkimiseks hajusas andmebaasis võrreldes mittehajusa andmebaasiga.
 - Teadete kaotsimine.
 - Rike arvutivõrgus.
 - Rike arvutivõrku kuuluvates arvutites.
 - Arvutisüsteem on nii hõivatud, et sel pole aega teatele vastata.

25.12.2017

Teema 12

75

Vigade haldamine hajusas andmebaasis (2)

- ♦ Katkestada kõik transaktsioonid, mida viga mõjutab.
- ♦ Tähistada server kui "vigane".
- ♦ Kontrollida perioodiliselt serveri taastumist.
- ♦ Serveri käivitumisel taastamisprotseduur.
- ♦ Serveri andmete koopia täiendamine.

25.12.2017

Teema 12

76

Päringud hajusas andmebaasis

- ♦ Täitmisplaani koostamine keerulisem, kui tsentraalse andmebaasi puhul.
- ♦ Andmebaasisüsteemi eesmärgiks on koostada ja täita täitmisplaan, mis tagab andmekäitluskeele lause võimalikult kiire täitmise.

25.12.2017

Teema 12

77

Päringute tegemine hajusas andmebaasis – näide

- ♦ *Property*(#propertyNo,city) **10 000** rida Londonis
- ♦ *Client*(#clientNo, maxPrice) **100 000** rida Glasgows
- ♦ *Viewing*(#propertyNo, #clientNo) **1 000 000** rida Londonis
- ♦ `SELECT p.propertyNo FROM Property p INNER JOIN (Client c INNER JOIN Viewing v ON c.clientNo=v.clientNo) ON p.propertyNo=v.propertyNo WHERE p.city='Aberdeen' AND c.maxPrice>200000;`

25.12.2017

Teema 12

78

Andmebaasid II 2017 © Erki Eessaar

Näite eeldused

- Iga rida igas tabelis on 100 sümbolit pikk.
- Eksisteerib 10 klienti, kelle puhul $\text{maxPrice} > 200\,000$.
- Aberdeenis paiknevat kinnisvara on vaadatud 100 000 korda.
- Kommunikatsioonilingi läbilaskevõime on 10 000 sümbolit sekundis.
- Teate saatmisel ühest asukohast teise on viide 1 sekund.
- Ühe rea edastamiseks kulub $100/10\,000 = 0.01$ sekundit.

25.12.2017 Teema 12 79

Andmebaasid II 2017 © Erki Eessaar

Erinevate täitmisplaanide täitmisaja võrdlus

Jrk.	Strateegia	Ligikaudne kulumud aeg
1	Tabel Client edastatakse Londonisse ja päring teostatakse seal. $\text{aeg} = 1 + (100\,000 \cdot 0.01)$	16,7 minutit
2	Tabelid Property ja Viewing edastatakse Glasgowsse ja päring teostatakse seal. $\text{aeg} = 2 + (1\,000\,000 + 10\,000) \cdot 0.01$	2,8 tundi
3	Londonis ühendatakse tabelid Property ja Viewing. Latakse Aberdeenis leiduvad varad ja igatse kohta kontrollitakse Glasgows, kas $\text{maxPrice} > 200\,000$. Ühe rea kontrolli tähendab kahte teadet – päring + vastus. $\text{aeg} = 100\,000 \cdot (1 + 0.01) + 100\,000 \cdot 1$	2,3 päeva
4	Glasgows latakse klientid, kelle $\text{maxPrice} > 200\,000$. Igatse kohta kontrollitakse Londonist, kas ta on vaadatud Aberdeenis paiknevat vara. Ühe rea kontrolli tähendab kahte teadet – päring + vastus. $\text{aeg} = 10 \cdot (1 + 0.01) + 10 \cdot 1$	20 sekundit <i>(Adjust table row)</i>
5	Londonis ühendatakse tabelid Property ja Viewing. Latakse Aberdeenis leiduvad varad ja moodustatakse projektsioon, mis sisaldab veerge propertyId ja clientId. See edastatakse Glasgowsse, et leida tingimisel $\text{maxPrice} > 200\,000$ vastavad read. Lõpuks hõlmed eeldatakse, et ka projektsioon on endiselt 100 märgi pikk. $\text{aeg} = 1 + (100\,000 \cdot 0.01)$	16,7 minutit
6	Glasgows latakse klientid, kelle $\text{maxPrice} > 200\,000$ ning need edastatakse Londonisse päringu lõpetamiseks. $\text{aeg} = 1 + (10 \cdot 0.01)$	1 sekund

Allikas
Connolly & Begg, 2001.

25.12.2017 Teema 12 80

Andmebaasid II 2017 © Erki Eessaar

Strateegia

- Liiguta üle võrgu võimalikult vähe andmeid.
- Rakenda enne andmete ülekandmist teise serverisse andmetele piirangu ja projektsiooni operatsioonid.

25.12.2017 Teema 12 81

Andmebaasid II 2017 © Erki Eessaar

Andmebaasi koopiad

- Koopia kättesaamatuks muutumisel saab süsteem hakata kasutama mõnda teist ja tööd jätkata
- Andmed saab paigutada kasutajale lähedale
- Erinevate koopiate peal saab täita erinevaid päringuid – korraga saab täita rohkem päringuid

25.12.2017 Teema 12 82

Andmebaasid II 2017 © Erki Eessaar

Probleemid andmebaasi koopiatega

Kokku on arvel 100 ühikut raha

Aeg	Koopia 1	Koopia 2
1	start transaction	start transaction
2	$\text{summa}_x = \text{summa}_x - 60$	$\text{summa}_x = \text{summa}_x - 50$
3	kirjuta(summa_x)	kirjuta(summa_x)
4	commit	commit

Arvelt võeti 110 ühikut raha

Hajusa andmebaasi terviklikkuse (consistency) nõue – hajusa andmebaasi kõikides sõlmedes peavad olema klientidel lugemiseks alati samad andmed.

25.12.2017 Teema 12 83


Andmebaasid II 2017 © Erki Eessaar

Probleemid andmebaasi koopiatega – CAP teoreem

- Kui soovida tagada andmete kohene terviklikkus, siis võrgukatkestuste korral ei saa raha välja võtta.
- Kui soovida tagada käideldavus (raha väljavõtmise võimalus ei katke), siis võrgukatkestuste korral on erinevates koopiates kontoseisu kohta erinevad andmed.

25.12.2017 Teema 12 84

Andmebaasid II 2017 © Erki Eessaar

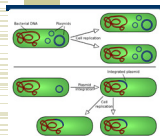


Probleemid andmebaasi koopiatega (2)

- ♦ *Käideldavuse* tagamine võib olla **äriliselt kasulik** kui *terviklikkuse* tagamine.
- ♦ Võimalik disainilahendus.
 - Kui tekib võrgukatkestus sõlmede vahel rakendab süsteem piirangu väljavõetava raha hulgale (nt 100EUR) ja sagedusele (nt kord tunnis).
 - Peale katkestuse lõppu ja andmete terviklikkuse taastamist küsitakse kliendilt vajadusel negatiivse kontojäägi kompenseerimiseks raha.

25.12.2017 Teema 12 85

Andmebaasid II 2017 © Erki Eessaar



Replikeerimine

- ♦ *Replikeerimine* e *tiražeerimine* (ingl *replication*) on protsess, mille käigus värskendatakse andmebaasi (täielikke või osalisi) koopiaid.
- ♦ Tüüpiliselt on need koopiaid erinevate serverite hoole all.

25.12.2017 Teema 12 86

Andmebaasid II 2017 © Erki Eessaar

Replikeerimiseks vajalik tarkvara

- ♦ Funktsionaalsus võib olla andmebaasisüsteemi sisse ehitatud
- ♦ Leidub eraldiseisvat tarkvara
 - <https://www.postgresql.org/download/products/3-clusteringreplication/>
- ♦ Arendajad võivad ise realiseerida
 - Andmebaasisüsteem peaks toetama *trigereid*, et oleks võimalik andmemuudatusi logida

25.12.2017 Teema 12 87

Andmebaasid II 2017 © Erki Eessaar

Kus saab teha andmemuudatusi?

- ♦ *Primaarserveri* poolne replikeerimine – ainult primaarserveris olevas primaarses koopias (emaeksemplaris).
 - Lihtsaim realiseerida, levinuim.
- ♦ *Mitme liidriga replikeerimine*
 - Mitu primaarse koopia + sekundaarsete koopiade komplekti (nt erinevates andmekeskustes).
 - Muudatusi saab teha kõigis primaarsetes koopiates.

25.12.2017 Teema 12 88

Andmebaasid II 2017 © Erki Eessaar

Kus saab teha andmemuudatusi? (2)

- ♦ *Jaosreplikeerimine* – igal ajahetkel on andmeelementi õigus muuta vaid ühes koopias.
- ♦ *Partnerreplikeerimine* – igal ajahetkel on andmeelementi õigus muuta kõikides koopiates.

25.12.2017 Teema 12 89

Andmebaasid II 2017 © Erki Eessaar

Kus saab teha andmemuudatusi? (3)

- ♦ Viimase kolme meetodi puhul võib *sünkroniseerimisel* tekkida vajadus *konfliktide* lahendamiseks (vastuolulised andmed erinevates koopiates).
- ♦ Konfliktide lahendamine võib toimuda:
 - lõppkasutajate poolt käsitsi
 - automaatselt

25.12.2017 Teema 12 90

Kuidas andmemuudatusi üle kanda?

- ♦ *Loogilisel tasemel* – teistesse andmebaasidesse kantakse muudatus loogiliste struktuuride (nt read) tasemel.
 - Need struktuurid on andmebaasi loogilisel e kontseptuaalsel tasemel.
 - Näiteks edastatakse logi, lisatud muudetud ja kustutatud ridade kohta.
 - Eesmärgiks tagada, et andmebaasides samad andmed.

25.12.2017

Teema 12

91

Kuidas andmemuudatusi üle kanda? (2)

- ♦ *Lausete tasemel* – teistes andmebaasides käivitatakse samad andmekäitluskeele laused, mis selles andmebaasis, kus algselt andmeid muudeti.
 - Loogilise taseme muudatuste ülekande üks viis.
 - Trigerite käivitumise ja mittedeterministlike funktsioonide poole pöördumise tulemusel ei pruugi erinevatesse koopiatesse jõuda ühesugused andmed.

25.12.2017

Teema 12

92

Kuidas andmemuudatusi üle kanda? (3)

- ♦ *Füüsilisel tasemel* – teistesse andmebaasidesse kantakse muudatus füüsiliste struktuuride (failid, plokid) tasemel.
 - Need struktuurid on andmebaasi füüsilisel e sisemisel tasemel.
 - Eesmärgiks tagada, et andmebaasid oleksid füüsiliselt ühesugused.

25.12.2017

Teema 12

93

Koopiate üldine probleem – muudatuste ülekanne vales järjekorras

- ♦ Andmebaasi primaarne koopia
 - Lisa rida r tabelisse X
 - Kustuta rida r tabelist X
- ♦ Muudatused kantakse üle andmebaasi sekundaarsesse koopiasse, kuid jõuavad kohale *vales järjekorras*
 - Kustuta rida r tabelist X
 - Lisa rida r tabelisse X
- ♦ Tulemus – koopiad ei ole kooskõlas

25.12.2017

Teema 12

94

Sünkroonne replikeerimine

- ♦ Koopiate värskendamine toimub kohe andmeid muutva transaktsiooni käigus.
- ♦ Eelised.
 - Kõigis koopiates on ühesugused värsked andmed.
- ♦ Probleemid.
 - Andmemuudatusi sisaldava transaktsiooni täitmine võtab rohkem aega ja võib ka ebaõnnestuda (näiteks võrguühenduse probleemide tõttu).

25.12.2017

Teema 12

95

Sünkroonne replikeerimine (2)

- Kui muudatused kantakse koopiasse üle füüsilisel tasemel, et hoida andmebaase **füüsiliselt identsetena**, siis muudatus mille tulemusel riknes põhiandmebaas viib ka koheselt koopiate riknemiseni.
 - Taastamine eraldiseisvast koopiast (kui selline on)
- Kui tuleb välja, et muudatus oli sisuliselt ebakorrektn, siis jällegi vajalik kas andmete parandamine kõigis koopiates või nii originaali kui koopia taastamine varukoopiast.

25.12.2017

Teema 12

96

Asünkroonne replikeerimine

- ♦ Koopiaid ei värskendata andmemuudatuse käigus.
- ♦ Koopiate värskendamine toimub perioodiliselt.
- ♦ Palju levinum kui sünkroonne.
- ♦ Koopiate kooskõla (andmete terviklikkus hajusa andmebaasi mõttes) saavutatakse viiteajaga (*eventual consistency*).

25.12.2017

Teema 12

97

Asünkroonne replikeerimine (2)

- ♦ Eelised.
 - Andmete muutmine toimub kiiremini kui sünkroonne replikeerimise korral.
 - Koopiate värskendamiseks saab valida aja, millal see kõige vähem süsteemi koormab.
 - Jääb aega avastada probleeme, et need ei kanduks koopiatesse
- ♦ Probleemid.
 - Koopiates ei pruugi olla värsked andmed – põhjustab *andmete lugemise anomaaliaid*.

25.12.2017

Teema 12

98

Andmete lugemise anomaalia – ei näe enda muudatust

- ♦ *Millal võib ilmned?*
 - Kasutaja muudab andmeid ja soovib seejärel neid lugeda
- ♦ *Kuidas ilmneb?*
 - Kasutaja ei näe enda andmemuudatuse tulemust
- ♦ *Mis on põhjus?*
 - Muudatus läks ühte koopiasse; lugemine toimub teise koopia põhjal, mida pole veel värskendatud

25.12.2017

Teema 12

99

Andmete lugemise anomaalia – ajas tagasiliikumine

- ♦ *Millal võib ilmned?*
 - Teeb mitu korda sama päringu (nt vaatab kommentaare)
- ♦ *Kuidas ilmneb?*
 - Hilisem päring ei too välja kõiki neid andmeid, mis varasem päring
- ♦ *Mis on põhjus?*
 - Päringud täideti erineva värskusastmega koopia põhjal

25.12.2017

Teema 12

100

Andmete lugemise anomaalia – vales järjekorras lugemine

- ♦ *Millal võib ilmned?*
 - Kasutaja teeb päringu
- ♦ *Kuidas ilmneb?*
 - Päringu tulemuses on hiljem kirjutatud tulemused eespool kui varem kirjutatud tulemus (nt kommentaari vastus enne kommentaari)
- ♦ *Mis on põhjus?*
 - Muudatused koopiasse kanti üle vales järjekorras

25.12.2017

Teema 12

101

Replikeerimise kasutusvaldkonnad

- ♦ Andmete *laialijagamine* (paiskamine) erinevatesse andmebaasidesse
- ♦ Andmete *koondamine* erinevatest andmebaasidest tervikpildi saamiseks
- ♦ Mobiilne andmebaas
- ♦ Varuandmebaas

25.12.2017

Teema 12

102

Hajusa SQL-andmebaasi disain

- ♦ Leia olemitüübid, seosetüübid, atribuudid ja loo globaalne andmemudel.
- ♦ Uuri süsteemi topoloogiat
 - Süsteemi alamosade, nende hallatavate andmete ja nendevaheliste suhtluskanalite paigutus.
- ♦ Analüüsi süsteemi tähtsaimaid transaktsioone
 - Milliseid andmeid kasutatakse üheskoos tähtsamates transaktsioonides?

25.12.2017

Teema 12

103

Hajusa SQL-andmebaasi disain (2)

- ♦ Uuri üks-mitu seosetüübi põhjal loodud tabeleid.
 - Kas kasutada tuletatud killustamist?
- ♦ Leia tabelid, mida ei killustata ja dubleeritakse.
 - Näiteks klassifikaatorid.
- ♦ Planeeri kildude ja koopiate paigutamine.

25.12.2017

Teema 12

104

Hajusad andmebaasid ja SQL standard

- ♦ SQL standard, ei paku hetkel mingit toetust **tõelistele hajusatele andmebaasisüsteemidele**.
- ♦ Standardi osa nr. 9 SQL/MED (*Management of External Data*) pakub toetust **föderatiivsetele andmebaasisüsteemidele**.

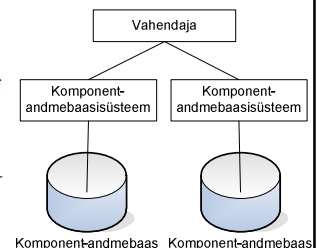
25.12.2017

Teema 12

105

Vahendajal põhinev hajussüsteem

- ♦ Hajussüsteemi võib ka luua võttes kasutusele *vahendaja e multiandmebaasisüsteemi*
- ♦ Komponent-andmebaasisüsteemid ei tea üksteise olemasolust – pole võrdsed partnerid
- ♦ Eristatakse
 - Mitteföderatiivne süsteem
 - Föderatiivne süsteem



25.12.2017

Teema 12

106

Mitteföderatiivne hajussüsteem – totaalne integratsioon

- ♦ Komponent-andmebaasisüsteemid ei ole autonoomsed.
- ♦ Ei eristata lokaalseid ja globaalseid kasutajaid ning operatsioone.

25.12.2017

Teema 12

107

Föderatiivne hajussüsteem – kompromiss

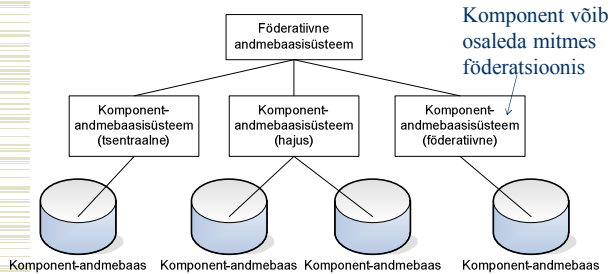
- ♦ Komponent-andmebaasisüsteemid on autonoomsed ja võimalik, et heterogeensed.
- ♦ Tarkvara, mis kontrollib ja koordineerib komponent-andmebaasisüsteeme, nimetatakse *föderatiivseks andmebaasisüsteemiks*.
- ♦ Föderatiivne andmebaasisüsteem pakub hajusa andmebaasi globaalsetele kasutajatele.
- ♦ Komponent-andmebaasisüsteemid pakuvad lokaalse andmebaasi lokaalsetele kasutajatele.

25.12.2017

Teema 12

108

Föderatiivne hajussüsteem (2)



25.12.2017

Teema 12

109

Föderatiivne hajussüsteem (3)

- ♦ Komponent-andmebaasisüsteemid pakuvad välja *eksport skeemi*, mis määrab, millised komponent-andmebaasi osad on kättesaadavad globaalsetele kasutajatele
- ♦ Kuidas sellist süsteemi luua?
 - Kasutada andmebaasisüsteeme
 - Oracle, PostgreSQL, IBM DB2
 - Kasutada spetsiaalset vahendaja süsteemi
 - InfoSphere Federation Server

25.12.2017

Teema 12

110



Analoogia

- ♦ Mitteföderatiivne hajussüsteem kui näide *nukuteatrist* – osalised (nukud) toimivad täpselt nii nagu nukujuht määrab.
- ♦ Ilma nukujuhita ei tee nukud midagi.

25.12.2017

Teema 12

111



Analoogia (2)

- ♦ Föderatiivne hajussüsteem kui näide *orkestreerimisest* – osalised (orkestrandid) toimivad tsentraliseeritud juhi (dirigendi) taktikepi ja etteantud nootide järgi.
- ♦ Dirigent ei kirjuta orkestrantidele kõike ette.
 - Kuni orkestri kõla ei kannata on neil vabadus mängida pilli omal viisil.
 - Dirigent ei juhi neid väljaspool lava või proovisaali.

25.12.2017

Teema 12

112



Analoogia (3)

- ♦ Tõeliselt hajus süsteem kui näide *koreograafiast* – osalised (tantsupartnerid) suhtlevad omavahel detsentraliseeritult, üldiste omavaheliste kokkulepete alusel.

25.12.2017

Teema 12

113



Analoogia (4)

- ♦ Tõeline hajussüsteem.
 - Rahvusriigid teevad omavahel koostööd.
- ♦ Föderatiivne hajussüsteem.
 - Leidub keskus, kust riike juhitakse.
 - Riikidel säilinud kohalike küsimuste üle otsustamise tasand.
- ♦ Mitteföderatiivne hajussüsteem.
 - Riigid asendunud provintsidega, mida juhitakse täielikult keskusest.

25.12.2017

Teema 12

114

Oracle kasutamine hajussüsteemide loomiseks

- ♦ Oracle võimaldab andmebaasi *sünkroonset* ja *asünkroonset* replikeerimist.
 - Oracle 12c soovib selleks kasutada Oracle GoldenGate tarkvara
- ♦ *Andmebaasi linke* kasutades saab Oracle andmebaasisüsteemi abil realiseerida *föderatiivse süsteemi*.

Replikeerimise tüübid Oracles

- ♦ *Read-only snapshots*
 - Sõltuvates andmebaasides on koopia tabelist T või selle alamosast.
 - Vastavaid andmeid saab muuta vaid juhtandmebaasis.
- ♦ *Updatable snapshots*
 - Sõltuvates andmebaasides on koopia tabelist T või selle alamosast.
 - Vastavaid andmeid saab muuta nii juhtandmebaasis kui sõltuvates andmebaasides.

Tabeli koopia värskendamise meetodid

- ♦ Täielik (COMPLETE) – kogu koopia tehakse uuesti.
- ♦ Kiire (FAST) – logitakse andmebaasis toimunud muudatusi. Koopiaid värskendatakse vastavalt logile.

Replikeerimise tüübid Oracles (2)

- ♦ *Multimaster replication* (võib toimuda nii sünkroonselt kui asünkroonselt)
 - Partnerreplikeerimine.
- ♦ *Procedural replication* (võib toimuda nii sünkroonselt kui asünkroonselt)
 - Juhtandmebaasis protseduuri poolt tehtud muudatuste ülekandmiseks sõltuvasse andmebaasi käivitatakse seal sama protseduur samade argumentidega.

Andmebaasi link

- ♦ Andmebaasi lingi abil saab kasutada teises andmebaasis olevaid skeemiobjekte.
 - Avalik (*public*) andmebaasi link on nähtav kõigile andmebaasi kasutajatele.
 - Privaatne (*private*) andmebaasi link on nähtav ühele konkreetsele andmebaasi kasutajale.
- ♦ Teine andmebaas võib, kuid ei pea, olema Oracle andmebaas.

Andmebaasi link (2)

- ♦ Andmebaasi link on viit, mis defineerib ühesuunalise kommunikatsioonikanali ühelt serverilt teisele.
- ♦ Kui andmebaasis **A** luua link **I** andmebaasis **B**, siis andmebaasis **A** saab selle abil kasutada andmebaasis **B** olevaid andmeid. Samas ei võimalda link **I** pöörduda andmebaasist **B** andmebaasi **A** poole.

Andmebaasi link (3)

- ♦ *Oracle Heterogeneous Services* ja *agendid* võimaldavad Oracle andmebaasi linkida ka teiste andmeallikatega (mõne teise firma andmebaasisüsteemis loodud andmebaas, tabelarvutuse programmi fail jne.)

Andmebaasidele globaalse nime määramine

- ♦ Serveris *apex.ttu.ee* Oracle andmebaasile globaalse nime andmine.
 - ALTER DATABASE RENAME GLOBAL_NAME TO orcl.apex.ttu.ee;
- ♦ Serveris *hektor3.ttu.ee* Oracle andmebaasile globaalse nime andmine.
 - ALTER DATABASE RENAME GLOBAL_NAME TO orcl.hektor3.ttu.ee;

tnsnames.ora

- ♦ Serveris *apex.ttu.ee* muudetakse faili *tnsnames.ora*:

```
ORCL.HEKTOR3.TTU.EE =
(DESCRIPTION =
  (ADDRESS_LIST =
    (ADDRESS = (PROTOCOL = TCP)(HOST = hektor3.ttu.ee) (PORT = 1521))
  )
  (CONNECT_DATA =
    (SERVER = DEDICATED)
    (GLOBAL_NAME = orcl.hektor3.ttu.ee)
    (SID = orcl)
  )
)
```

Andmebaasi lingi loomine serveris *apex.ttu.ee* asuvas andmebaasis

- ♦ CREATE PUBLIC DATABASE LINK orcl.hektor3.ttu.ee CONNECT TO tud1 IDENTIFIED BY hahaha USING 'ORCL.HEKTOR3.TTU.EE';
- ♦ Eldused:
 - failis *tnsnames.ora* on sissekanne ORCL.HEKTOR3.TTU.EE,
 - serveris *hektor3.ttu.ee* on kasutaja *tud1*, kelle parool on *hahaha*.
- ♦ Loodud lingi kaudu saab teha samu tegevusi, mis on lubatud kasutajale *tud1* serveris *hektor3.ttu.ee*.

Teises andmebaasis asuvale objektile viitamine

- ♦ skeemi nimi.objekti nimi@andmebaasi lingi nimi
- ♦ Serveris *hektor3* asub tabel *emp* skeemis *tud1*.
 - [tud1.emp@orcl.hektor3.ttu.ee](#)
 - Kui kontekst lubab võib skeemi nime ära jätta.

Sünonüümi kasutamine serveril *apex.ttu.ee* asuvas andmebaasis

- ♦ Päring teises serveris asuvast andmebaasist:
 - SELECT * FROM emp@orcl.hektor3.ttu.ee;
- ♦ Asukoha nähtamatus tagamiseks võiks serveris *apex.ttu.ee* luua sünonüümi:
 - CREATE SYNONYM emp_hektor3 FOR emp@orcl.hektor3.ttu.ee;
 - SELECT * FROM emp_hektor3;
 - DELETE FROM emp_hektor3;

SQL laused ja transaktsioonid hajusas andmebaasis

- ♦ *SQL lausete süntaks* on ühesugune nii hajusalt paiknevas, kui hajusalt mittepaiknevas andmebaasis.
- ♦ *Transaktsioonide lõpetamine* toimub kasutaja jaoks ühtemoodi nii hajusalt paiknevas, kui hajusalt mittepaiknevas andmebaasis.

Päringu täitmisplaan ja statistika

```
SQL> SELECT * FROM emp_hektor3;
```

14 rows selected.

Execution Plan

Plan hash value: 3956160932

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	Inst
0	SELECT STATEMENT REMOTE		14	532	3 (0)	00:00:01	ORCL
1	TABLE ACCESS FULL	EMP	14	532	3 (0)	00:00:01	ORCL

Note

- fully remote statement

Statistics

```
0 recursive calls
0 db block gets
0 consistent gets
0 physical reads
0 redo size
1590 bytes sent via SQL*Net to client
543 bytes received via SQL*Net from client
2 SQL*Net roundtrips to/from client
0 sorts (memory)
0 sorts (disk)
14 rows processed
```

Globaalne unikaalne identifikaator

- ♦ Kildude ühendamisel peaks kildudes võtmeväärtused olema unikaalsed ka moodustunud tabelis.
 - Sisulise tähendusega võtmed – pole probleem.
 - Süsteemi poolt genereeritav globaalselt unikaalne identifikaator.
 - Arvujada generaatorite kasutamine.

Globaalne unikaalne identifikaator (Oracle)

- ♦ 16 baidine RAW väärtus, mis on globaalselt unikaalne. Üheski andmebaasis ei genereerita samasugust väärtust.
- ♦ Genereerimine nõuab CPU-lt rohkem vaeva, kui unikaalse arvu genereerimine arvujada generaatori abil.
- ♦ Suurem, kui süsteemi poolt genereeritud unikaalne arv ja salvestamiseks kulub rohkem ruumi.
- ♦ Raske meeles pidada.
- ♦ Pigem mitte kasutada.

Globaalne unikaalne identifikaator (Oracle) (2)

- ♦ CREATE TABLE Test(test_id\$ RAW(16) DEFAULT sys_guid() CONSTRAINT pk_test primary key);
- ♦ INSERT INTO Test(test_id\$) VALUES (DEFAULT);
- ♦ SELECT * FROM Test;
- ♦ Tulemus:
EE214761E9C17647E04341F428C1F49A

Arvujada generaatorite kasutamine (Oracle)

- ♦ Tabelist Tellimus kaks horisontaalset kildu
- ♦ CREATE SEQUENCE seq_tellimus_1 START WITH 1 INCREMENT BY 100;
- ♦ CREATE SEQUENCE seq_tellimus_2 START WITH 2 INCREMENT BY 100;
- ♦ seq_tellimus_1 – 1, 101, 201, 301, 401
- ♦ seq_tellimus_2 – 2, 102, 202, 302, 402

PostgreSQL kasutamise hajussüsteemide loomiseks

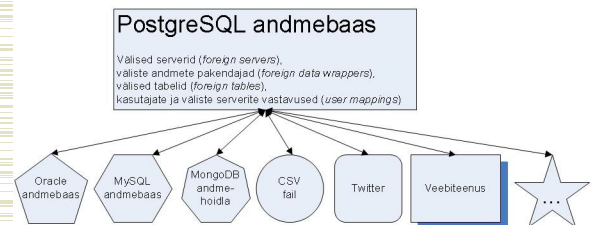
- ◆ PostgreSQL võimaldab andmebaasi *sünkroonset* ja *asünkroonset replikeerimist*.
- ◆ Lisamooduli *dblink* kaudu saab muuhulgas käivitada funktsioone teises PostgreSQL andmebaasis.
- ◆ PostgreSQL realiseerib küllaltki suure osa SQL/MED spetsifikatsioonist. PostgreSQLi abil saab realiseerida *föderatiivse süsteemi*.

25.12.2017

Teema 12

133

PostgreSQL kui föderatiivne andmebaasisüsteem



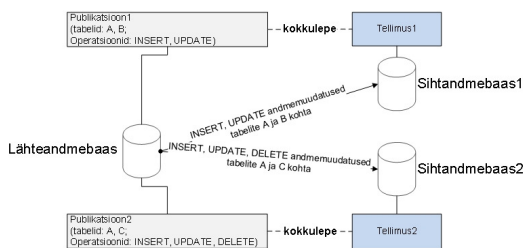
Välistest allikatest pärit andmeid saab PostgreSQL andmebaasis lugeda ja alates PostgreSQL 9.3 on ka võimalik neid andmeid muuta.

25.12.2017

Teema 12

134

Loogiline replikeerimine PostgreSQLis



Andmebaasimootori osa alates PostgreSQL 10.

25.12.2017

Teema 12

135

P2P (*Peer-to-peer*) and mebaasid

- ◆ Andmebaasid erinevate serverite kontrolli all.
- ◆ Puudub keskne süsteem, mis päringuid vahendaks.
- ◆ Server ei tea kõiki teisi servereid, vaid ainult oma naabreid.
- ◆ Päringule vastamiseks edastab server päringu oma naabritele, kes omakorda edastavad naabritele jne.
- ◆ Serverile on teada tõlkereeglistik oma andmebaasiskeemi ja naabrite andmebaasiskeemide vahel.

25.12.2017

Teema 12

136

Hajussüsteemi näide

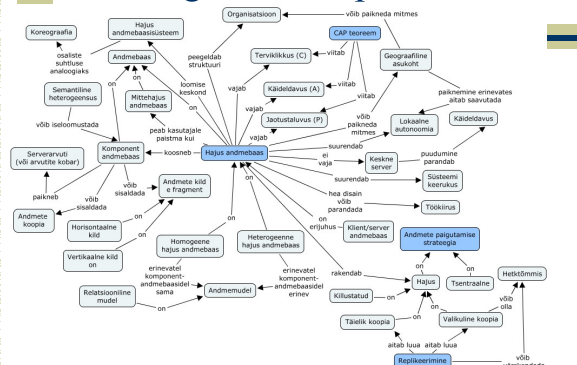
- ◆ Krüptoraha nagu Bitcoin on näide hajussüsteemist.
- ◆ Kuidas Bitcoin toimib, selle kohta saab lugeda head selgitust siit:
 - <http://www.michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works/>

25.12.2017

Teema 12

137

Mõningad teema põhimõisted



25.12.2017

Teema 12

138

