Veebiteenuste ja hajussüsteemide arendus

Loeng 8: Hajus- ja Pilveandmebaasid

Pelle Jakovits, jakovits@ut.ee

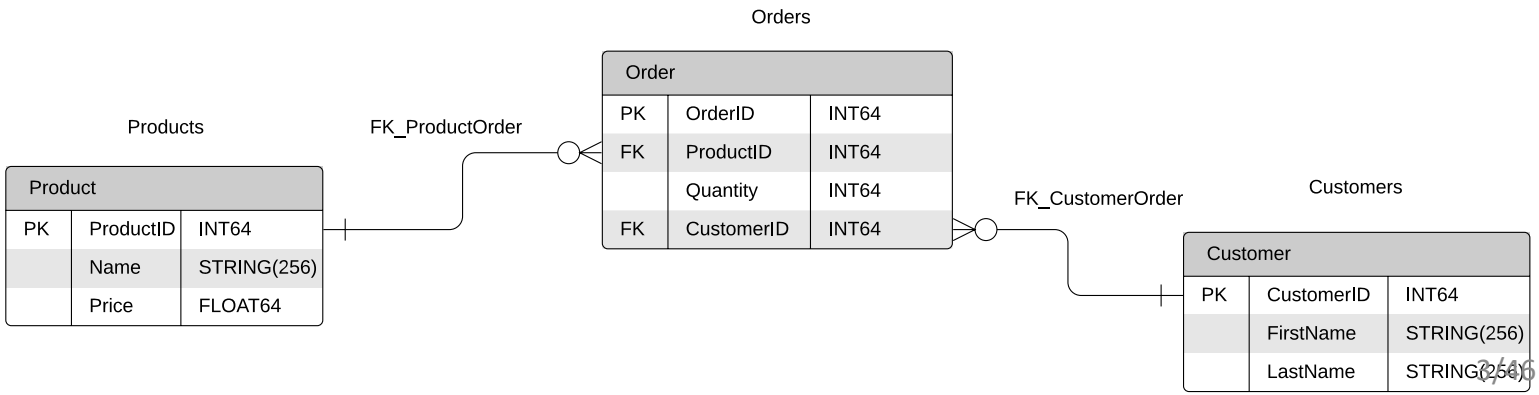
Aprill 2025

Sisukord

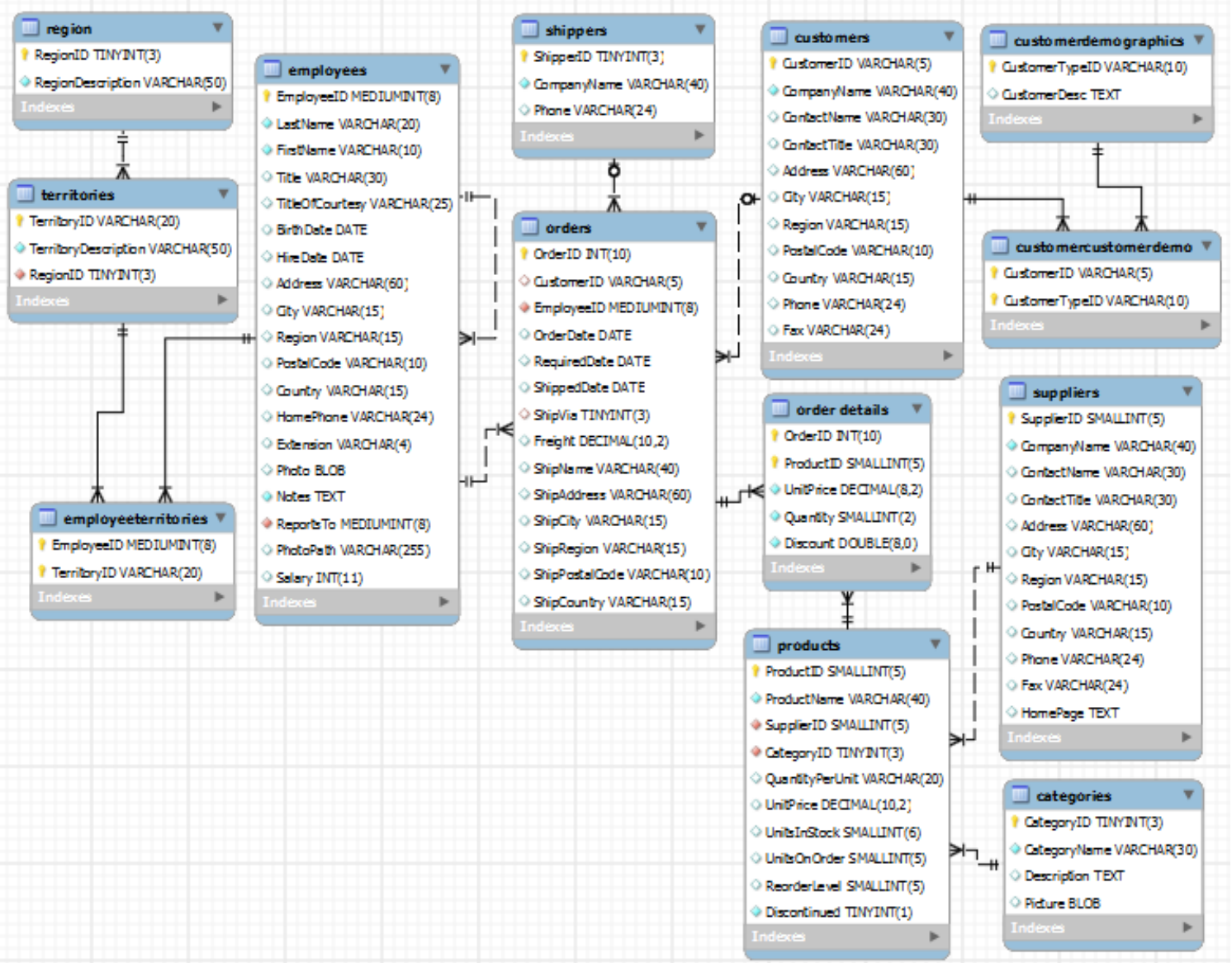
- Relatsiooniline andmebaasi mudel
- Andmebaaside skaleerimine
- Hajusandmebaaside mudelid
- Andmebaaside pilveteenused

Andmebaaside relatsiooniline mudel

- Andmed salvestatakse tabelitena
  - Erinevad olemitüübid (Kasutaja, asutus, jne.) erinevates tabelites
- Ranged seosed (relatsioonid) tabelite vahel
  - Välisvõtme (Foreign key) viited tabelite veergude vahel
- Andmete tüübid on määratud range skeemi (schema) järgi
  - Fikseeritud andmetüübid, Tühjade null väärtuste lubamine/keelamine
- Andmetele pääseb tavaliselt juurde SQL päringute kaudu



Andmebaasi tabelite struktuuri näide

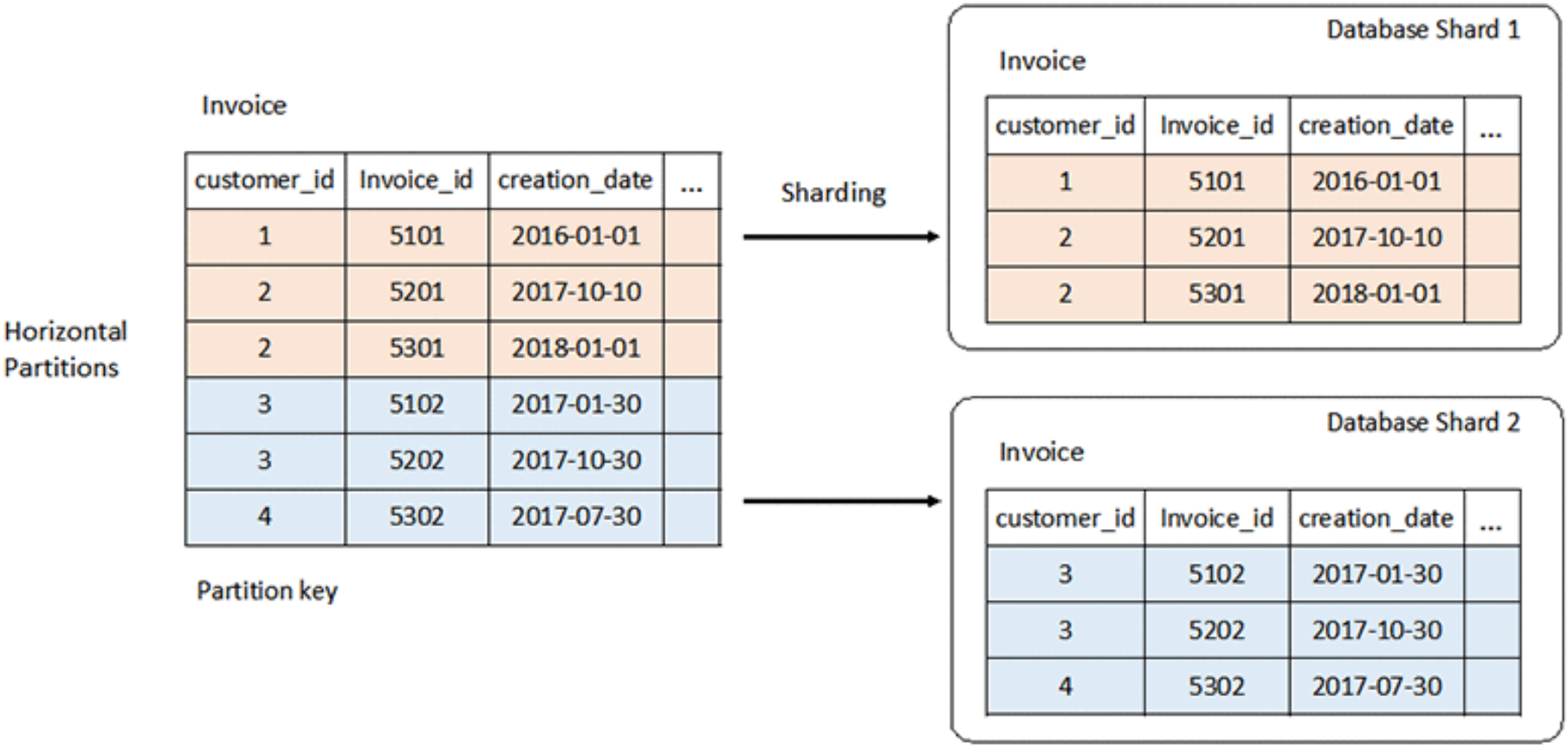


https://dev.mysql.com/doc/sakila/en/

Relatsiooniliste andmebaaside skaleerimine

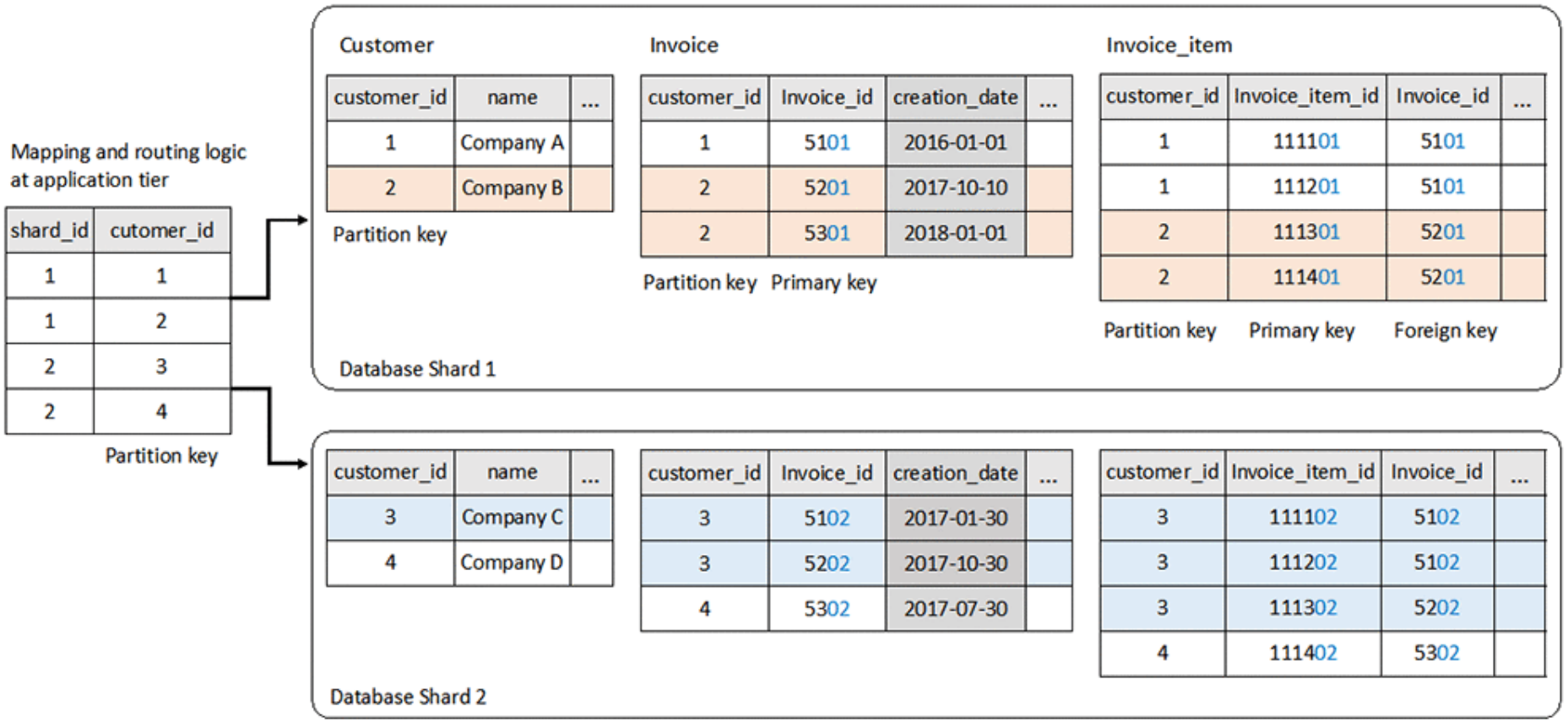
- **Vertikaalne skaleerimine** – Ühe serveri võimsuse suurendamine/vähendamine
- **Horisontaalne skaleerimine** – serverite lisamine klastrisse
- Relatsioonilised andmebaasid ei skaleeru hästi horisontaalselt
  - Relatsioonilises andmemudelis on liiga palju sõltuvusi
- Andmebaasi killustamine (sharding) on üks lähenemisviis horisontaalselt skaleerimiseks

Killustamine (sharding)



https://aws.amazon.com/blogs/database/sharding-with-amazon-relational-database-service/

Killustamine (sharding)



https://aws.amazon.com/blogs/database/sharding-with-amazon-relational-database-service/

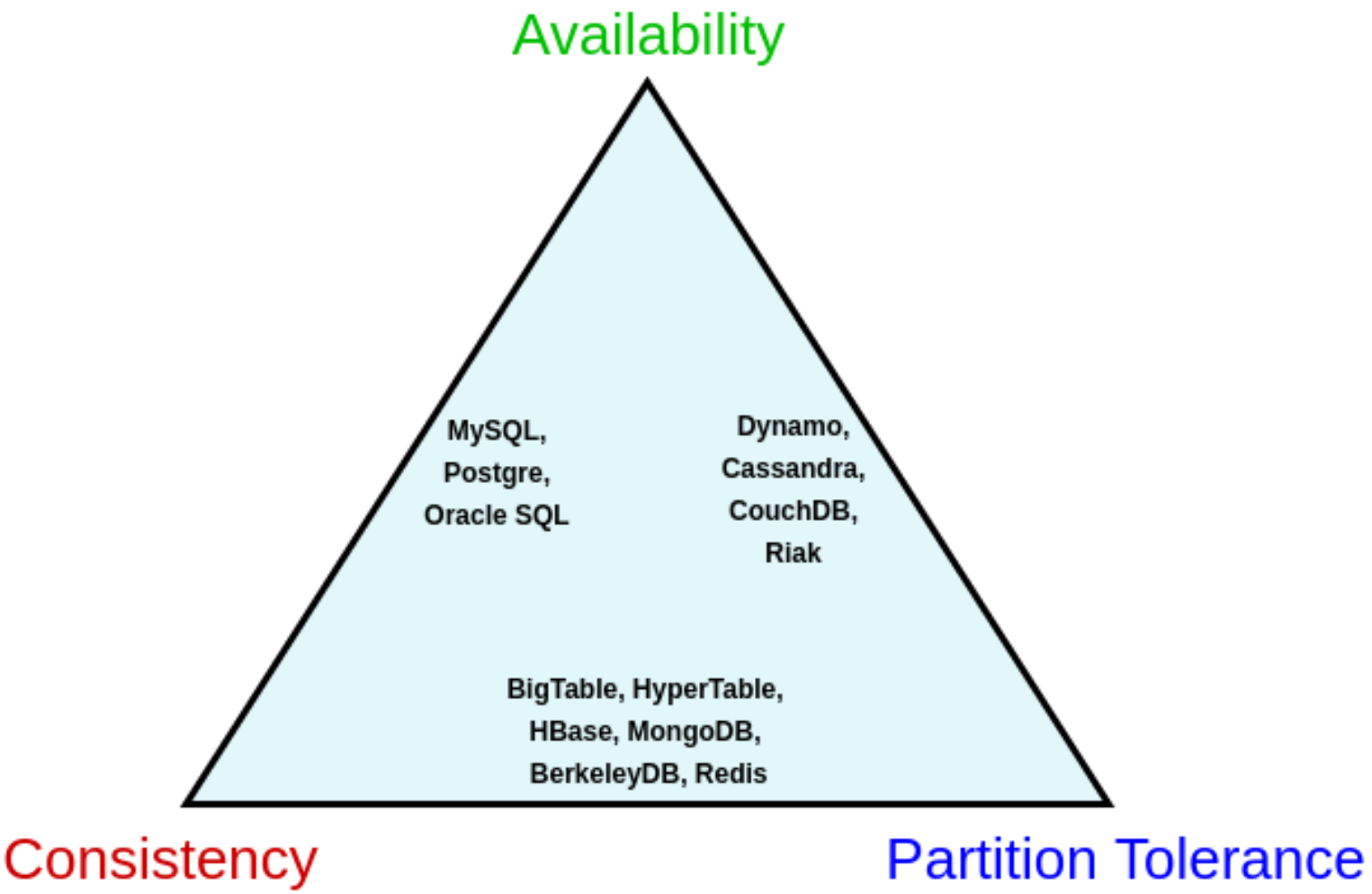
Hajusandmebaasid

- Suurandmete (Big Data) ja pilvetehnoloogiate edukus tekitas vajaduse väga suurte ja skaleeruvate andmebaaside jaoks
- NoSQL – Mitterelatsioonilised andme mudelid
  - Põhinevad lihtsal Võti-Väärtus (**Key - Value**) andmestruktuuril
  - Võti-Väärtus mudeli peale on ehitatud erinevad abstraktsioonikihid
- Kasutatakse lihtsamad, range skeemata andme(baasi)mudelid
  - Disainitud skaleeritavust silmas pidades
- Vaja hoolitseda, et:
  - andmebaas toimib tõrgeteta
  - andmed sünkroniseeritakse korrektselt olukorras kus andmebaasi päringud võivad olla jagatud paljude serverite vahel
  - Andmete lugemine ja kirjutamine toimub kiirelt

CAP Teoreem

- Samuti tuntud kui Brewer teoreem, autori Eric Brewer järgi
- Hajusarvutil (hajusandmebaasil) on **võimatu** samaaegselt **garanteerida** kõik kolm järgnevat omadust:
  1. **Järjepidevus (Consistency)** - iga lugemise operatsioon saab kas kõige uuema kirje või veateate
  2. **Kättesaadavus (Availability)** - iga päring saab korrektse vastuse
  3. **Paritisiooneerimise/tõrke taluvus** (Partition tolerance)
     - Süsteem jätkab tööd hoolimata suvalisest süsteemi partitsioneerimisest
     - Võrgurikked, pakettide väljalangemine
- Tavaliselt tuleb tõrgete korral valida kas järjepidevuse või kättesaadavuse vahel
- Lahendused, keskenduvad rohkem kättesaadavusele, püüavad saavutada **viivitusega järjepidevust** (eventual consistency)

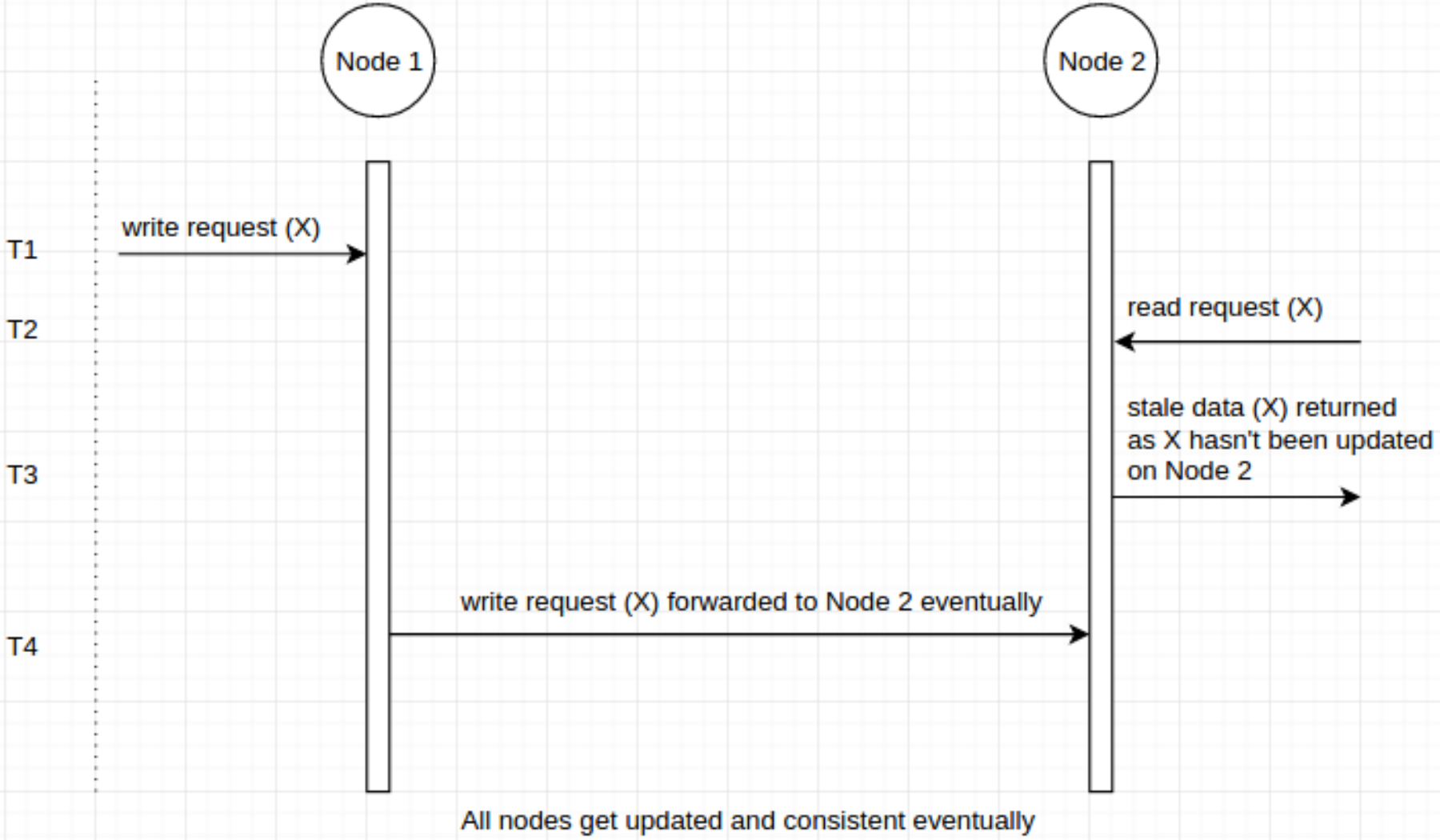
CAP Teoreem


Availability
MySQL,
Postgre,
Oracle SQL
Dynamo,
Cassandra,
CouchDB,
Riak
BigTable, HyperTable,
HBase, MongoDB,
BerkeleyDB, Redis
Consistency
Partition Tolerance

Viivitusega järjepidevus - eventual consistency

- Tavaliselt tuleb tõrgete korral valida kas järjepidevus või kättesaadavus
- Lahendused, keskenduvad rohkem kättesaadavusele, püüavad saavutada **viivitusega järjepidevust** (eventual consistency)
- Viivitusega järjepidevus: iga sõlme andmed muutuvad lõpuks järjepidevaks!
- Võimaldab pakkuda madalat latentsust - aga koos suurema riskiga aegunud andmete tagastamiseks
- Kui püüda saavutada nii järjepidevust kui ka kättesaadavust
  - Siis saavutame selle asemel väga suure latentsuse (ootame kuni andmed on täielikult sünkroniseeritud).

Viivitusega järjepidevus - eventual consistency

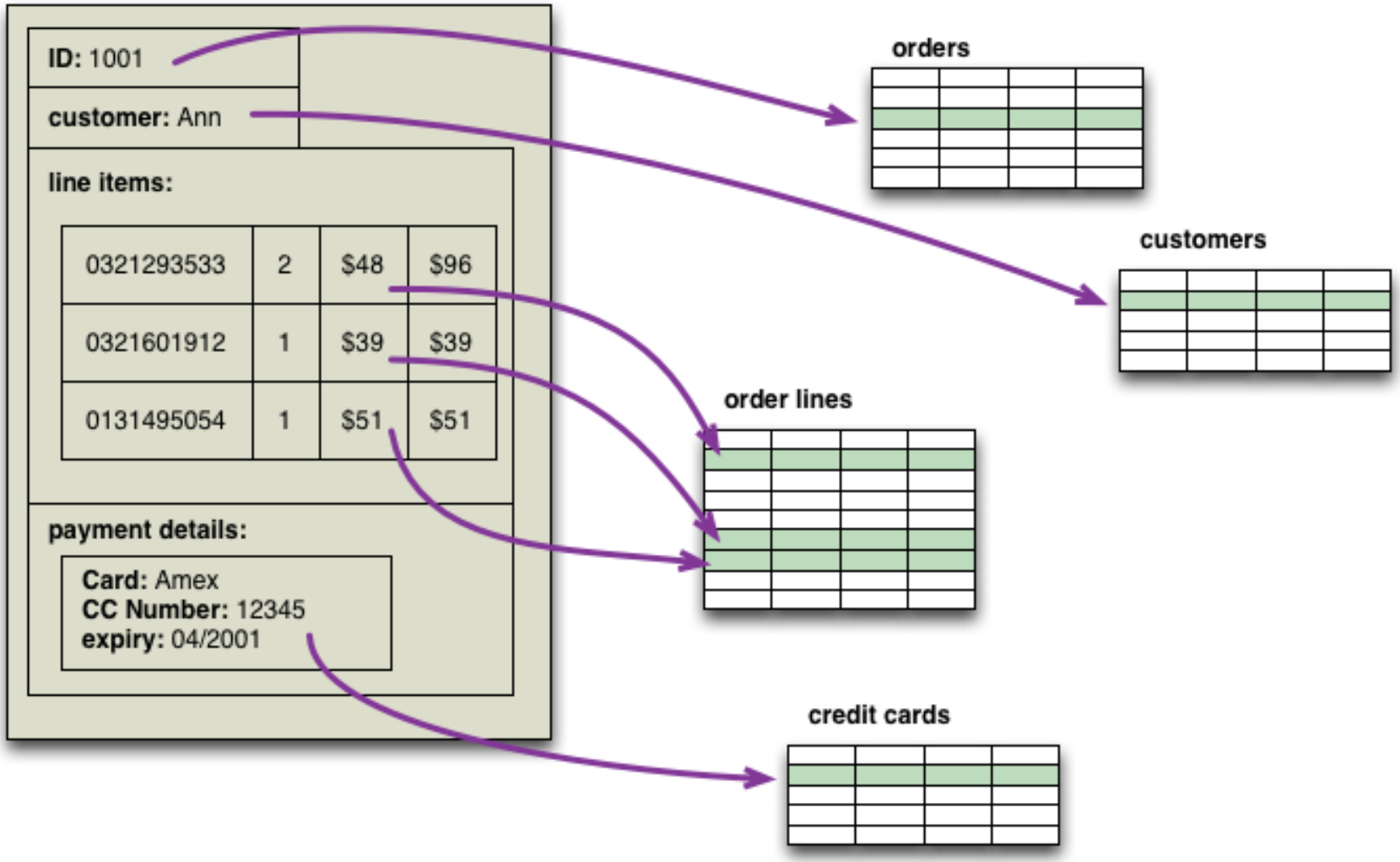


https://hackernoon.com/eventual-vs-strong-consistency-in-distributed-databases-282fdad37cf7

Agregeeritusele orienteeritus

- Agregatsioon on andmete kogum, mida käsitletakse ühe üksusena
  - Nt. klient ja kõik tema tellimused
- Normaliseeritud relatsioonilistes andmebaasides arvutatakse agregatsioonid GroupBy (ja JOIN) operatsioonide abil
- Eesmärk on **de-normaliseerida** andmed, et oleks vaja minimaalset (või üldse mitte) grupeerimist ja JOIN operatsioone!
- Hoiame andmeid andmebaasis juba "ühendatud" ja "grupeeritud" kujul
  - Kasutajate järgi grupeeritud
  - Ettevõtete järgi grupeeritud
- Võtmete-põhised agregatsioonid moodustavad naturaalsed andmete killustamise ühikud/grupid

Agregatsioonidele-orienteeritus

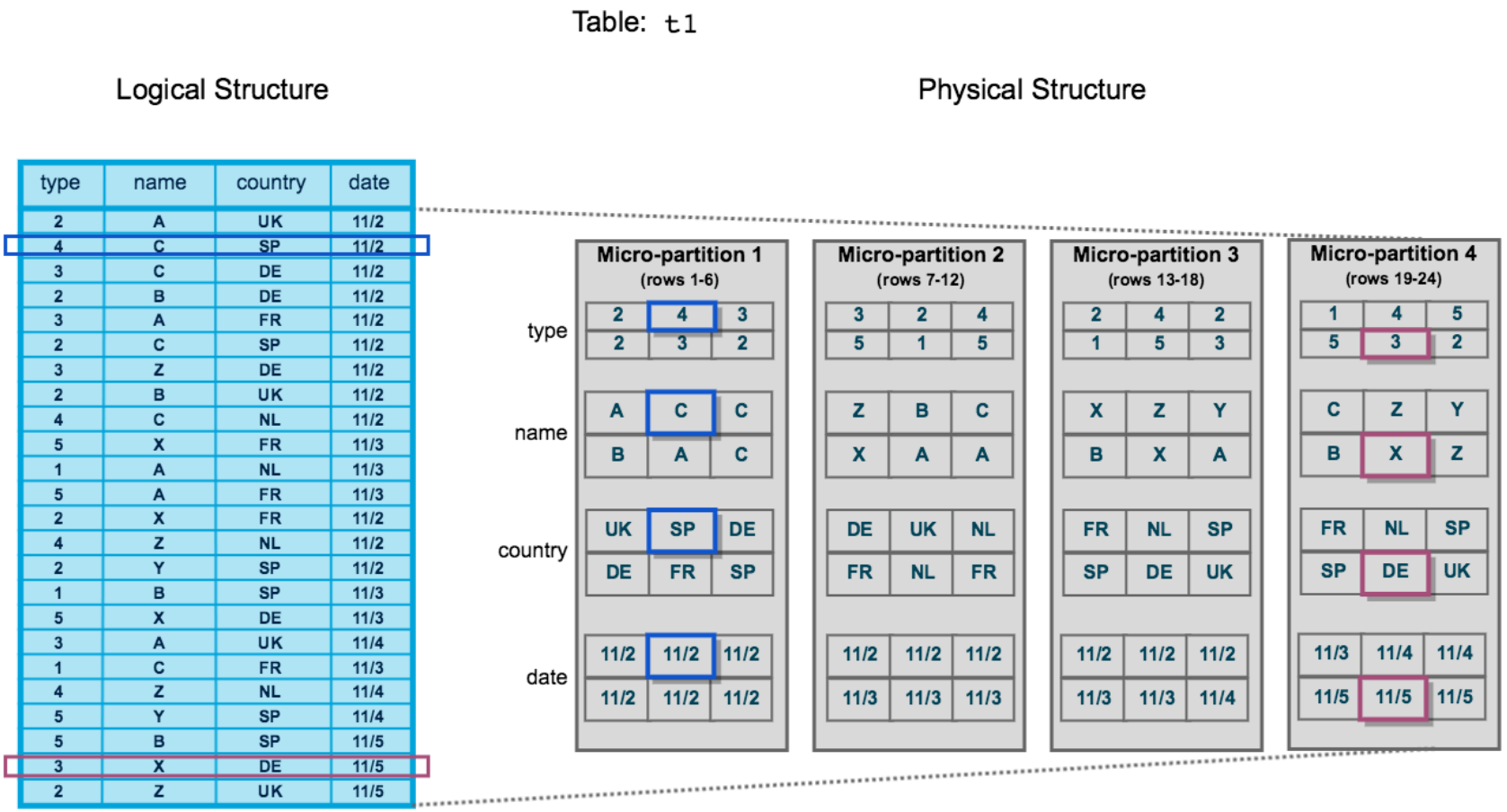


https://martinfowler.com/bliki/AggregateOrientedDatabase.html

Partitsioneerimine

- Võtme-väärtuse mudelis toimib võti indeksina
- Osades lahendustes saab luua sekundaarseid indekseid
- Andmed jaotatakse klastris olevate erinevate masinate vahel
  - Sarnaselt killustamisele
- Tavaliselt on andmed jaotatud ridade ja/või veeru plokkide kaupa
- Kasutajad saavad sageli dünaamiliselt partitsioneerimise parameetreid
  - Annab kontrolli andmete hajutamise üle
  - Väga oluline päringute kiiruse optimeerimiseks
  - Kasutusjuhtum: **Igakuine aruandlus**. Andmed jaotatakse kuude või nädalate kaupa.
  - Partitsioneerimise viis tuleb paika panna rakenduse disainimise käigus!

Näide: partitsioneerimata, sorteeritud tabel



type	name	country	date
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	SP	11/5
3	X	DE	11/5
2	Z	UK	11/5

	Micro-partition 1 (rows 1-6)	Micro-partition 2 (rows 7-12)	Micro-partition 3 (rows 13-18)	Micro-partition 4 (rows 19-24)
type	2 4 3 2 3 2	3 2 4 5 1 5	2 4 2 1 5 3	1 4 5 5 3 2
name	A C C B A C	Z B C X A A	X Z Y B X A	C Z Y B X Z
country	UK SP DE DE FR SP	DE UK NL FR NL FR	FR NL SP SP DE UK	FR NL SP SP DE UK
date	11/2 11/2 11/2 11/2 11/2 11/2	11/2 11/2 11/2 11/3 11/3 11/3	11/2 11/2 11/2 11/3 11/3 11/4	11/3 11/4 11/4 11/5 11/5 11/5

https://docs.snowflake.com/en/user-guide/tables-clustering-micropartitions#benefits-of-micro-partitioning

Partitsioneeritud ja klasterdatud tabel

Table: `t1`

Logical Structure

type	name	country	date
2	A	UK	11/2
4	C	SP	11/2
3	C	DE	11/2
2	B	DE	11/2
3	A	FR	11/2
2	C	SP	11/2
3	Z	DE	11/2
2	B	UK	11/2
4	C	NL	11/2
5	X	FR	11/3
1	A	NL	11/3
5	A	FR	11/3
2	X	FR	11/2
4	Z	NL	11/2
2	Y	SP	11/2
1	B	SP	11/3
5	X	DE	11/3
3	A	UK	11/4
1	C	FR	11/3
4	Z	NL	11/4
5	Y	SP	11/4
5	B	SP	11/5
3	X	DE	11/5
2	Z	UK	11/5

```
SELECT name, country FROM t1
WHERE type = 2
AND date = '11/2';
```

Physical Structure

Pilveplatvormide omadused

- Lõpmatute ressursside illusioon
- Ilma ettemaksuta
- Väikese granulaarsusega arveldus (h, m, s, ms)
- Massiivselt skaleeruvad teenused
- Ressursside ja teenuste jooksvalt (on-demand) ja dünaamiliselt tellimine
- Kasutame ainult seda, mida vaja – Elastne
  - Ettemakseid ei ole vaja, kasutada saab lühiajaliselt
- Juurdepääs Interneti kaudu, asukohast sõltumatu
- Läbipaistev - kasutajate eest on peidetud süsteemide keerukus. Need on hallatud, virtualiseeritud, abstraheeritud.

Pilve ja Hajusandmebaaside tüübid

- **Hallatud** salvestusteenuste pakkumine
  - Storage as a Service (STaaS)
- Teenuse pakkuja hoolitseb installeerimise, konfigureerimise, skaleerimise, partitsioneerimise, varukoopiate jms eest
- Erinevad hinnastamise mudelid olenevalt objektile/failile juurdepääsu sagedusest, eeldatavast latentsusest ja salvestuse kestusest
- Tüübid:
  1. **Bucket/Blob** andmebaasid
  2. **Hallatud SQL** andmebaasid
  3. **Hallatud NoSQL** andmebaasid

Hallatud Pilve SQL andmebaasid

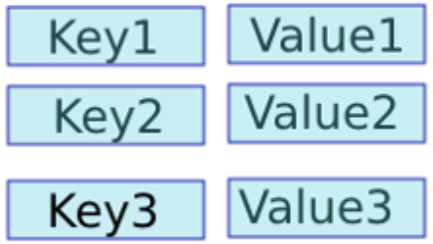
- Kaks peamist tüüpi:
  - Lihtne SQL-server nõudmisel (on-demand)
  - Täielikult hallatav, killustatud (sharded) SQL-klaster
- **Amazon (RDS)** - teenus SQL-serverite dünaamiliseks üles seadmiseks ja skaleerimiseks (Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database ja SQL Server)
- **IBM Db2** – IBM'i enda lahendus aastast 83.
  - SQL, BigSQL (Hadoop), Data Warehouse, Analytics
- **Google Cloud SQL** - Hallatud MySQL, PostgreSQL, or SQL Server

Hallatud NoSQL andmebaasid

- **Mitte-Relatsioonilised andmebaasid**
  - **Võti-Väärtus (Key-Value) mudel**
    - AWS DynamoDB, Google Cloud Datastore
  - **Dokumedi-põhised andmebaasid**
    - AWS DocumentDB, IBM Cloudant (CouchDB), Google Cloud Firestore
  - **Veergude perekondade** (Column Family) mudel
    - AWS Managed Apache Cassandra Service, Google BigQuery

Võti-Väärtus mudel

- Andmed salvestatakse võtme-väärtuste paaridena
- Väärtus on läbipaistmatu objekt andmebaasis
- Näited: Dynamo, Riak, Apache Ignite, ArangoDB, Berkeley DB, Couchbase
- Horisontaalne skaleeritavus
  - Erineva võtmega andmed võib partitsioneerida erinevatele sõlmedele
  - Sama võtmega andmed salvestatakse üksteise lähedal
  - Sobib pilvandmetöötluseks
- Paindlikud skeemavabad mudelid, mis sobivad struktureerimata andmete jaoks
- Päringud: Get, Put and Delete (REST)
- Andmete pärimine võtme järgi
  võib olla väga kiire



Key-Value Mudel: Võtmete struktuur

- AWS S3 võtmed (keys):
  - **https://s3.Region.amazonaws.com/bucket-name/KeyName**
  - **https://s3.us-west-2.amazonaws.com/mybucket/Directory/puppy.jpg**
- Võtmed võivad olla komplektsed:
  - `employee:1:firstName = "Martin"`
  - `employee:2:firstName = "John"`
  - `payment:1:1:amount = "10000"`
  - `payment:1:1:date = "01/12/2019"`
  - `payment:2:1:amount = "5000"`
  - `payment:2:1:date = "01/12/2019"`

Object / Bucket / Blob Storage

- Jälgib Võti-väärtus (Key-value) mitterelatsioonilist (NoSQL) mudelit
- Suure hulga struktureerimata andmete salvestamine
  - Andmebaasi struktuur/skeema ei ole paigas
  - Pildid, videod, logifailid, varukoopiafailid jne
- Amazon S3, Azure Blob storage, Google Cloud Storage, IBM Cloud Object Storage
- Säilitamiseks võib olla erinevaid režiime:
  - Reaalajas vs vähesagedane vs arhiveerimine

AWS S3

- Amazon Simple Storage Service (Amazon S3)
- Jagab andmed salvedesse (buckets)
  - Loogiline salvestusüksus – unikaalse asukohaga veebikaust
  - Salv võib sisaldada praktiliselt piiramatul hulgal kaustu ja faile
  - Lubatud kuni 100 salve kasutaja kohta
- Salvestuse klassid:
  - S3 Standard
  - Intelligent-Tiering (Automatic storage class selection, min 30 päeva)
  - Standard-Infrequent Access (Less frequent, but rapid access, min 30 päeva)
  - One Zone-Infrequent Access (Single availability Zone, min 30 päeva)
  - S3 Glacier (data archiving, access in minutes to hours, min 90 päeva)
  - S3 Glacier Deep Archive (access in 12 hours, min 180 päeva)

AWS S3 hinnad

Mode	Storage TB/Month	1M INFO requests	Data Retrieval per TB
S3 Standard	$23.55	$5.00	$0.00
S3 Intelligent, standard	$23.55	$5.00	$0.00
S3 Intelligent, Infrequent Access	$12.80		
S3 Standard - Infrequent Access	$12.80	$10.00	$10.00
S3 One Zone - Infrequent Access	$10.24	$10.00	$10.00
S3 Glacier	$4.10	$50.00	$30.00
S3 Glacier Deep Archive	$1.01	$50.00	$20.00

Dokumedi-põhine mudel

- Andmeid salvestatakse võtme-väärtuste paaridena
  - Väärtus on "dokument" ja sellel on täiendav struktuur
- Ranget skeemat/struktuuri ei ole
  - Ei kontrollita andmebaasi poolt
- Andmeid päritakse dokumendi struktuuri alusel
- Näited: CouchDB, MongoDB, ArangoDB, BaseX, Clusterpoint, Couchbase

Näite JSON document


```
{
  "name":"Asheville Veedub Volkswagon Repair & Restoration",
  "address":"851 Charlotte Hwy",
  "attributes":{
      "BusinessAcceptsCreditCards":"True",
      "ByAppointmentOnly":"False",
      "GoodForKids":"True"
  },
  "business_id":"0KwutFa520HgPLWtFv02EQ",
  "categories":"Auto Repair, Automotive",
  "city":"Fairview",
  "is_open":1,
  "latitude":35.5431561,
  "longitude":-82.4419299,
  "neighborhood":"",
  "postal_code":"28730",
  "review_count":7,
  "stars":4.5,
  "state":"NC"
}
```


Agregaate kirjeldatakse JSON-is listi andmestruktuuri kasutades

Veergude perekondade mudel

- Column Family model - Veergude perekonnad (grupid)
- Andmed on salvestatud suurtes hõredates tabeli struktuurides
- Veerud on grupeeritud veergude perekondadesse
  - Veergude perekond on loogiline veergude grupp
  - Sarnane kontseptsioon nagu tabel relatsioonilises andmebaasis
- Andmebaasi kirjet võib pidada kahe tasandiliseks kujutiseks (Map)
  - Column Family -> Column -> Value
  - Veeru perekond -> Veerg -> Väärtus
- Uusi veerge saab igal ajal lisada
- Näited: BigTable, Cassandra, HBase, Accumulo

Column Family näide

Static column families — **Dynamic column family**

_id	names			contacts		messages	
	username	firstname	lastname	phone	email	item1	item2
a001	jsmith01	John	Smith	555 0001	jsmith@example.com	Message 1	Message 2
b014	pauljones					new message	

a001	Names	username	jsmith
		firstname	John
		lastname	Smith
	Contacts	phone	5 550 001
		email	jsmith@example.com
	Messages	item1	Message1
		item2	Message2
		...	...
		itemN	MessageN
b014	Names	username	pauljones
	Contacts		
	Messages	item1	new Message

Dynamic column family

Millal kasutada relatsioonilisi andmebaase?

- Vanad, proovitud ja testitud tehnoloogiad
- Hästi optimeeritud jõudluse jaoks
- Range andmete struktuur tähendab vähem vigu
  - Lihtsam kontrollida andmete kvaliteeti
  - Lihtsam vältida andmete puudumist, valesid andmetüüpe jne.
- Väga hea jõudlus seni, kuni andmed mahuvad ühte masinasse

Millal kasutada mitterelatsioonilisi andmebaase?

- Kui andmemaht kasvab liiga suureks
  - Lihtsam ja odavam skaleerida
- Kui andmete struktuur pole fikseeritud
  - Struktureerimata andmete salvestamine ja päringute tegemine
  - Prototüüpimine: Rakenduste loomine ilma andmete struktuuri paika panemata
  - Andmete struktuuri muutmine on lihtne
- Struktureeritud dokumentide (JSON, XML) kasutamisel
  - Sarnane skeem dokumentides, kuid võib oluliselt erineda
  - Mitme tasemelised alamstruktuurid
- Paljud avatud lähtekoodiga valikuid

Pilveandmebaaside eelised

- Saab valida vastavalt vajadusele sobiva mahu ressursse andmebaasi jaoks kasutades IaaS mudelit
  - Näiteks 256 GB RAM, 64 vCPU
- SQL andmebaasid või vabavaralised NoSQL andmebaasid teenusena
  - Ei pea ise virtuaalmasinaid haldama.
  - Klasteramine/killustamine konfigureeritakse automaatselt
    - Andmebaasi skaleerimine tihti kasutaja kontrolli all
  - Teenuse kvaliteet oleneb tarkvarast, eraldatud ressursside mahust.
- Täishallatud andmebaasi süsteemid
  - Skaleerimine peidetud: hallatud platvormi poolt
  - Minimaalselt haldust
  - Ei ole vaja muretseda selle eest, kas on andmebaasis piisavalt mahtu
  - Platvormi poolt garanteeritud Teenuse kvaliteet (QoS)

Selle nädala Praktikum

- (Pilve-) Andmebaaside kasutamine veebiteenuste loomisel