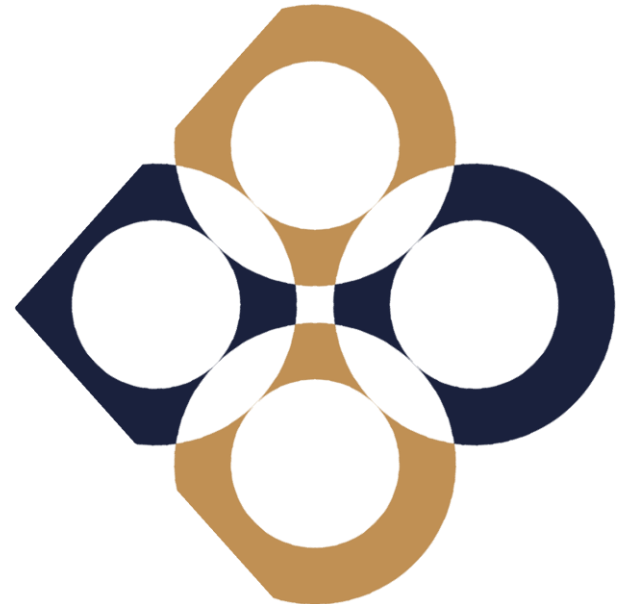


Adatbázisok előadás 11.

Oszlopalapú adatbázisok



Miről lesz szó?

- ☐ Oszlopalapú adatbázisok jellemzői
- ☐ A Cassandra adatbázis
 - ☐ Jellemzők
 - ☐ Lekérdezések
 - ☐ Terminál
 - ☐ Elérés Python-ból
 - ☐ Feladatok megoldása

Oszlopalapú adatbázisok (Oszloptárolók)

Olyan adatbázisok, amelyek az oszlopok elemeit együtt, illetve egymás melletti helyeken (blokkokban) tárolják a lemezen

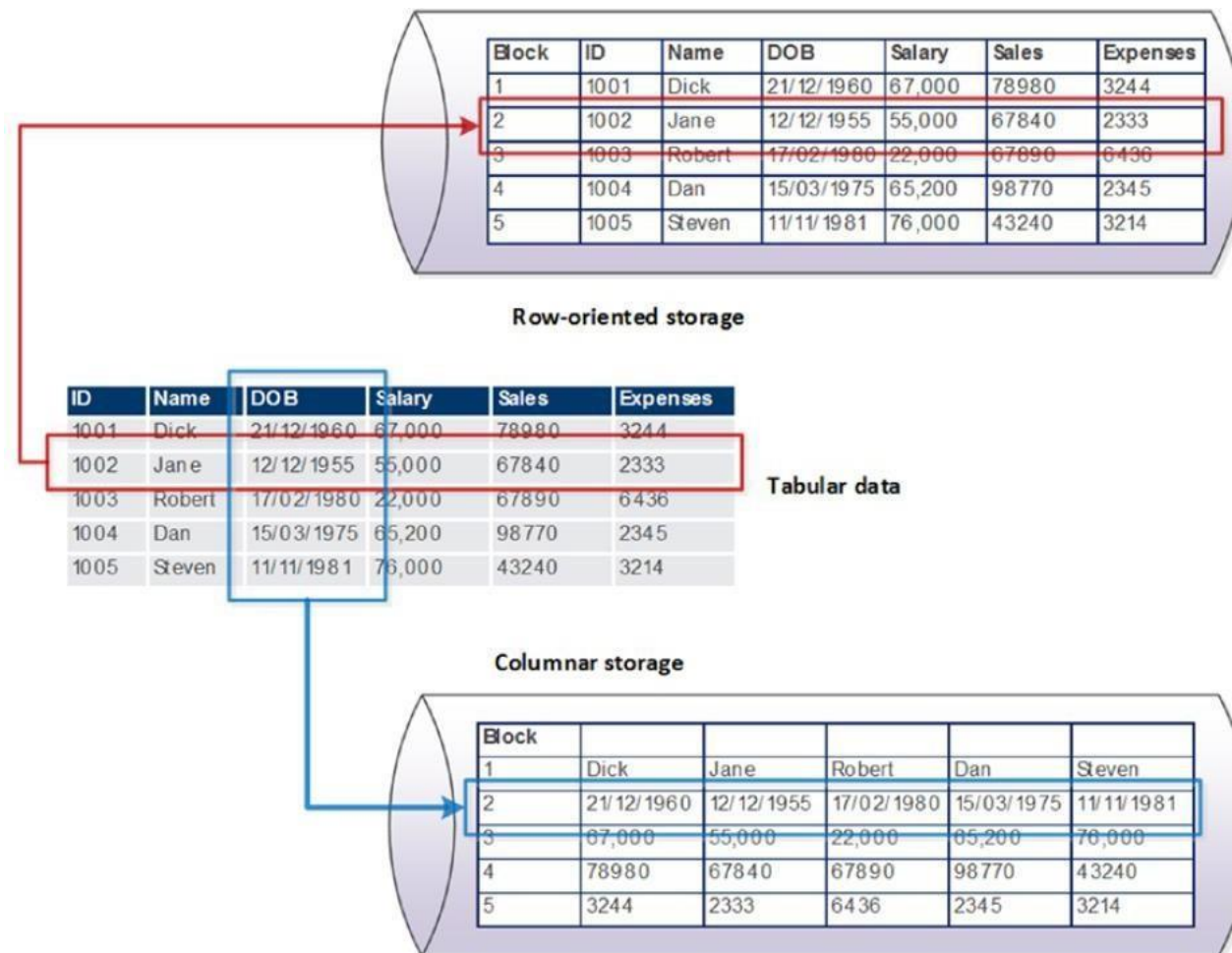
Column Oriented Database

<u>date</u>	<u>price</u>	<u>size</u>
2011-01-20	10.1	10
2011-01-21	10.3	20
2011-01-22	10.5	40
2011-01-23	10.4	5
2011-01-24	11.2	55
2011-01-25	11.4	66
...
2013-03-31	17.3	100

Felismerés:

Sok (elemző) lekérdezés többnyire oszlopokon dolgozik, így a soralapú szervezés felesleges I/O műveletekkel jár

Soralapú vs. Oszlopalapú tárolás



A mai relációs adatbázis-kezelők egy része mindkét tárolási módot támogatja

Oszloptárolók – előnyök és hátrányok

Előnyök

- Aggregálások, projekció, szelekció gyorsabb elvégzése
- Adatok tömöríthetősége
- Az adatok akár a memóriába is beférhetnek
- Egyszerre több lekérdezés is végrehajtható – diszjunkt attribútumhalmazok esetén
- Skálázható, elosztott rendszer

Hátrányok

- Írási műveletek lassúsága
- OLTP adatbázisokhoz nem optimálisak
- Kevés sorból álló táblák lekérdezése lassú lehet
- (Inkrementális) adatbetöltés lassú

Oszloptárolók – Hol használják őket?

Adattárházak

Adatelemzések

Big data rendszerek

Párhuzamos feldolgozás

Oszloptárolók - Példák



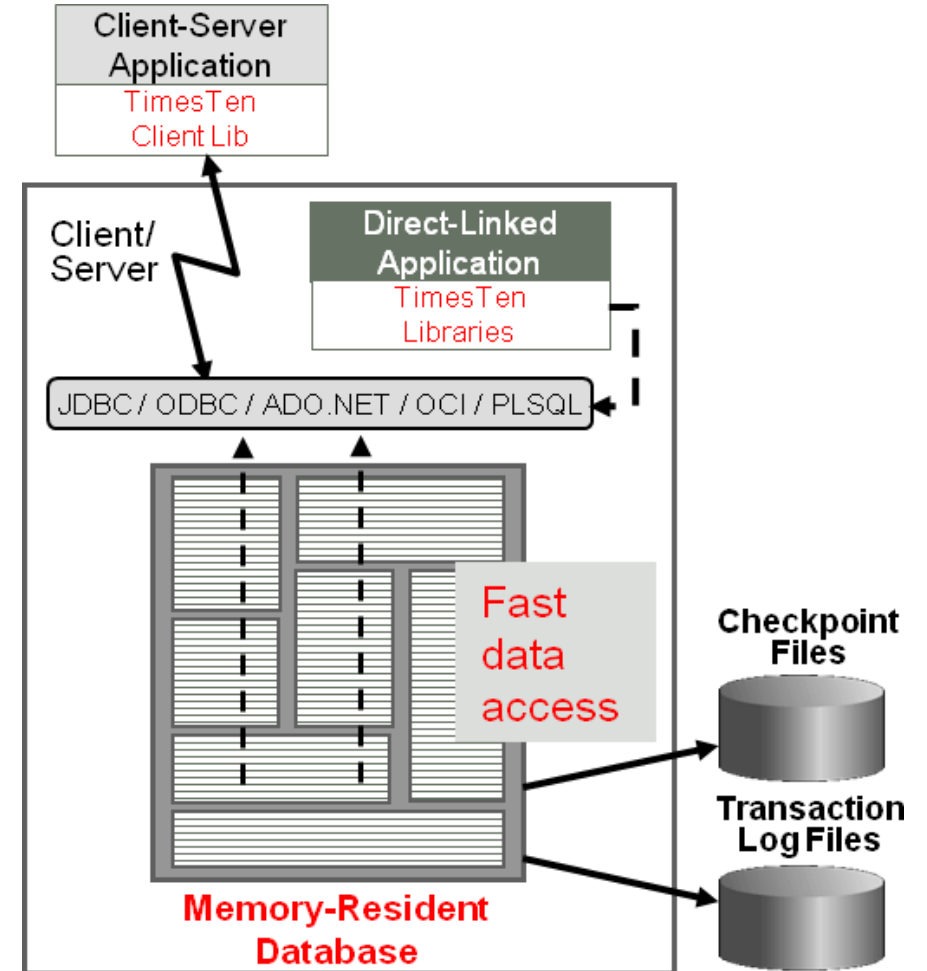
HYPERTABLE



In-memory adatbázisok

Az oszlopalapú tárolás sok esetben in-memory technikával van kombinálva

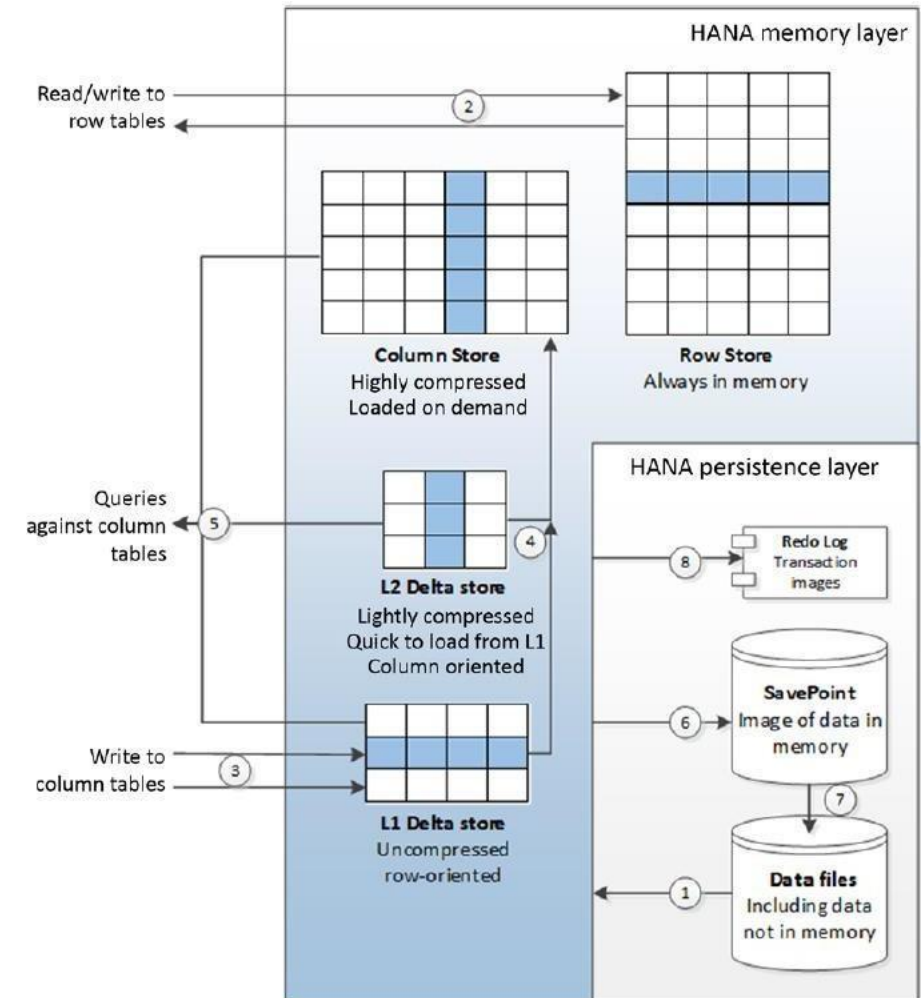
- ❑ Az elmúlt évtizedben a memóriák ára nagymértékben csökkent
- ❑ A kapacitás ugyanakkor nagymértékben nőtt
- ❑ Akár az egész adatbázis lehet memória rezidens
 - ❑ Így nincs szükség cache-re
 - ❑ A lekérdezések sebessége sokkal jobb lesz
 - ❑ Probléma: mi történik pl. áramszünet esetén?



<https://www.oracle.com/technical-resources/articles/database/timesten-imdb.html>

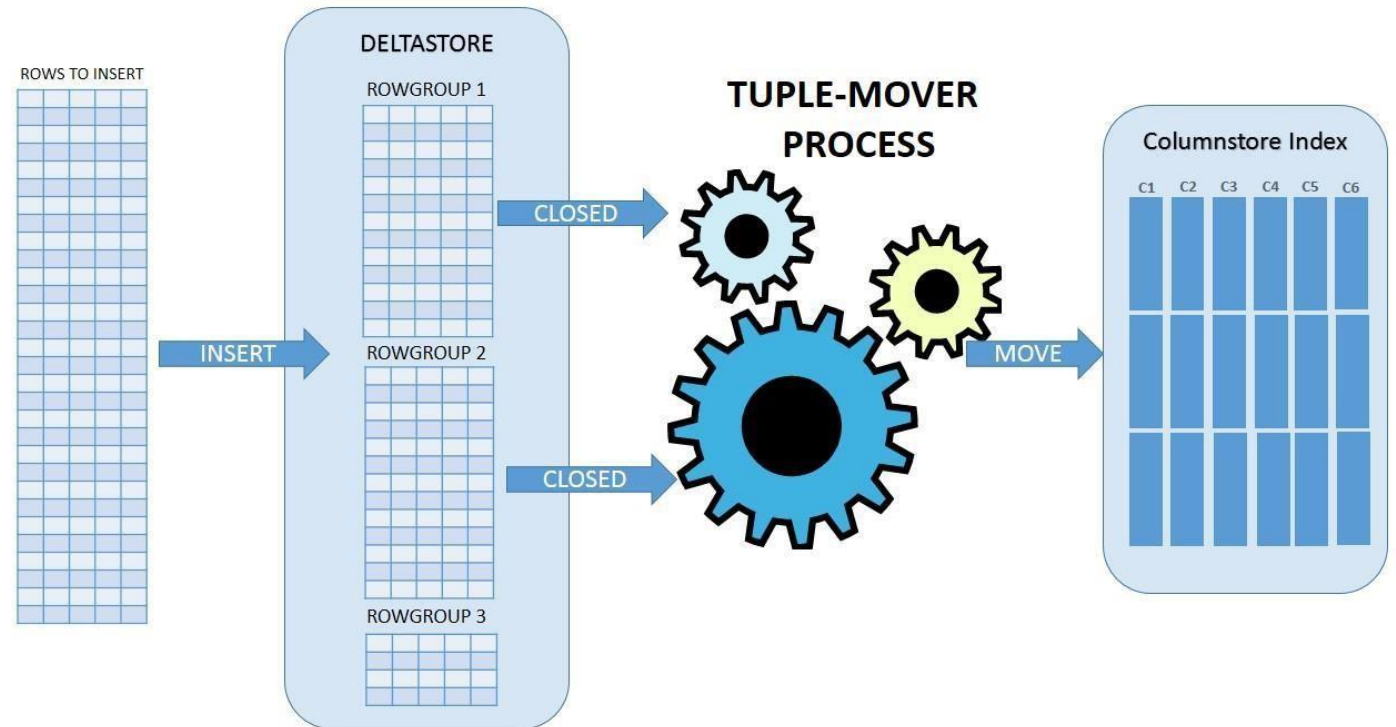
Oszloptárolók – SAP HANA

- ❑ Eredetileg relációs adatbázis
- ❑ Kombinálja az in-memory technikát az oszlop-alapú adattárolással
- ❑ Speciális hardvert igényel, elsősorban gyors SSD tárolókat
- ❑ A táblák sor-alapú, illetve oszlop-alapú tárolási modellekre is épülhetnek (konfigurálható)
- ❑ A sorok minden esetben a memóriába kerülnek, az oszlopok igény/beállítás szerint
- ❑ A rendszer így alkalmas OLTP, illetve OLAP feladatokra egyaránt



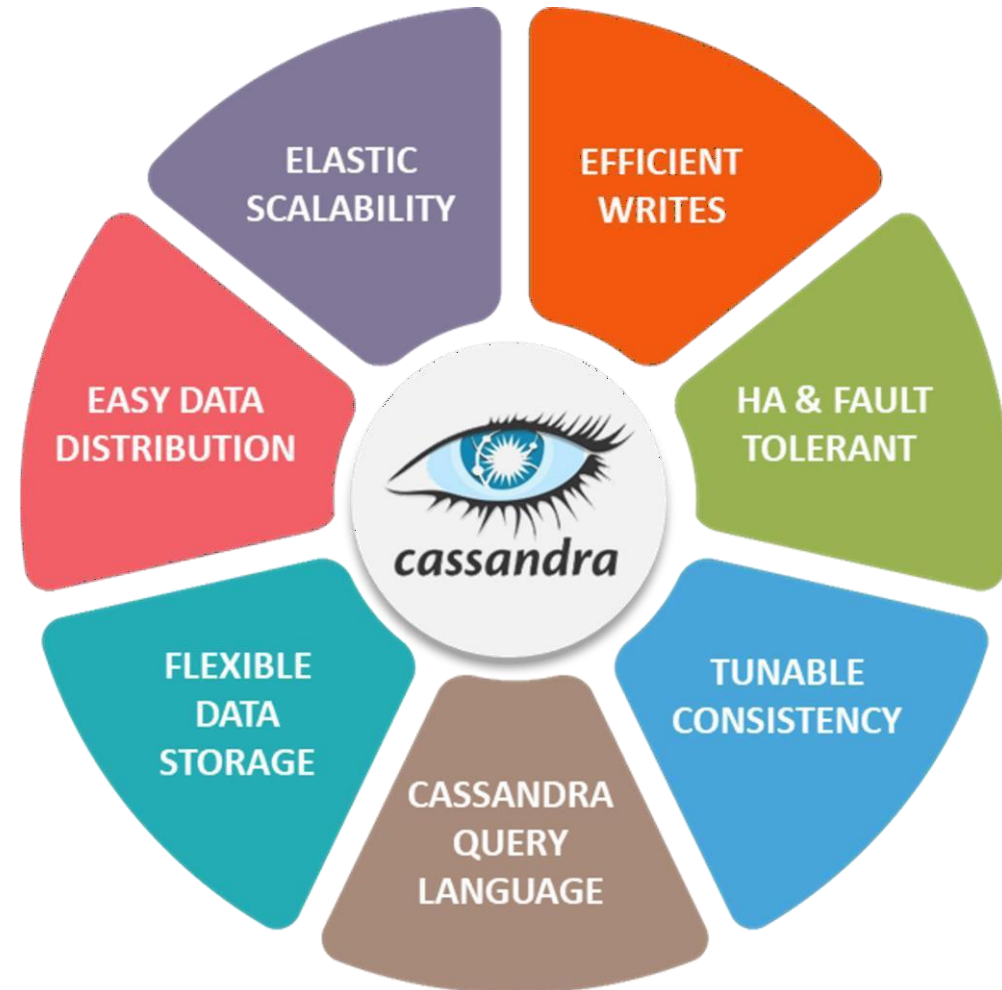
Oszloptárolók – Vertipaq (xVelocity)

- ☐ Adatbázis motor
 - ☐ Power Pivot
 - ☐ SSAS Tabular
 - ☐ Power BI
- ☐ SQL Server Columnstore index
- ☐ In-memory technikával kombinált



Oszlopalapú adatbázisok - Cassandra

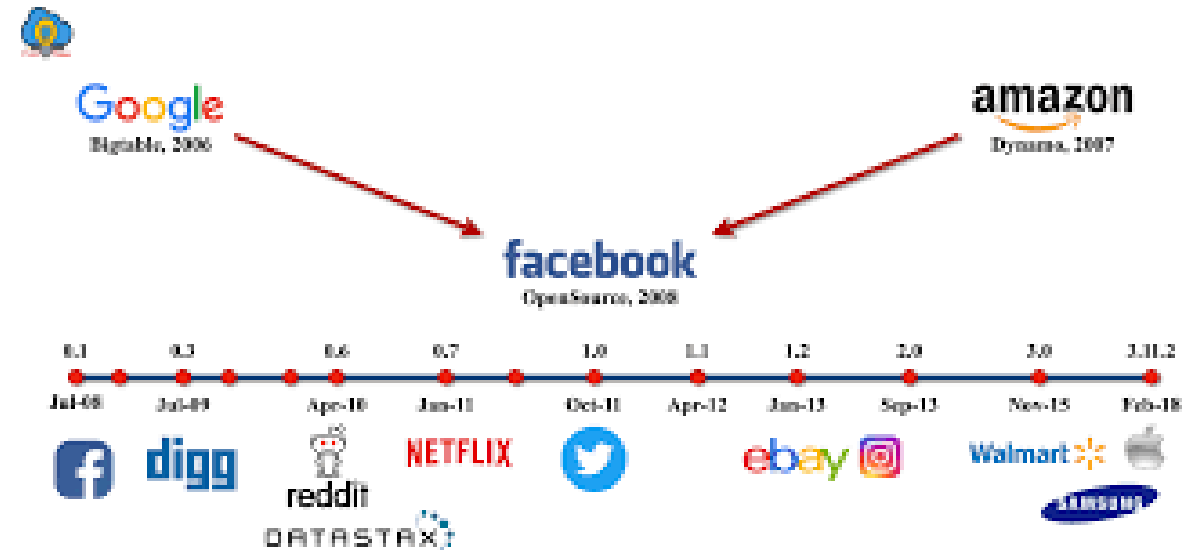
- ❑ Open-source, elosztott NoSQL adatbázis
- ❑ Magas rendelkezésre állás, hibatűrés,
- ❑ Lineáris skálázhatóság
- ❑ Saját lekérdező nyelv (CQL – Cassandra Query Language)
- ❑ Hadoop/MapReduce integráció
- ❑ Szabályozható konzisztencia



<https://www.edureka.co/blog/interview-questions/cassandra-interview-questions/>

Cassandra - történet

- ❑ 2008-ban jelent meg
(Facebook, inbox search)
- ❑ Az Apache projekt része lett
- ❑ Alapjai a Google BigTable és az Amazon DynamoDB
- ❑ Széleskörűen elterjedt (Netflix, Instagram, ebay)
- ❑ Legutolsó verzió: 4.x (2022)



Cassandra – alapfogalmak

Cluster (klaszter) – Egy rendszert alkotó node-ok gyűjteménye

Data Center (adatközpont) – Egymással kapcsolatban lévő csomópontok halmaza

Node (csomópont) – Egy számítógép (szerver) tárolási és számítási kapacitással

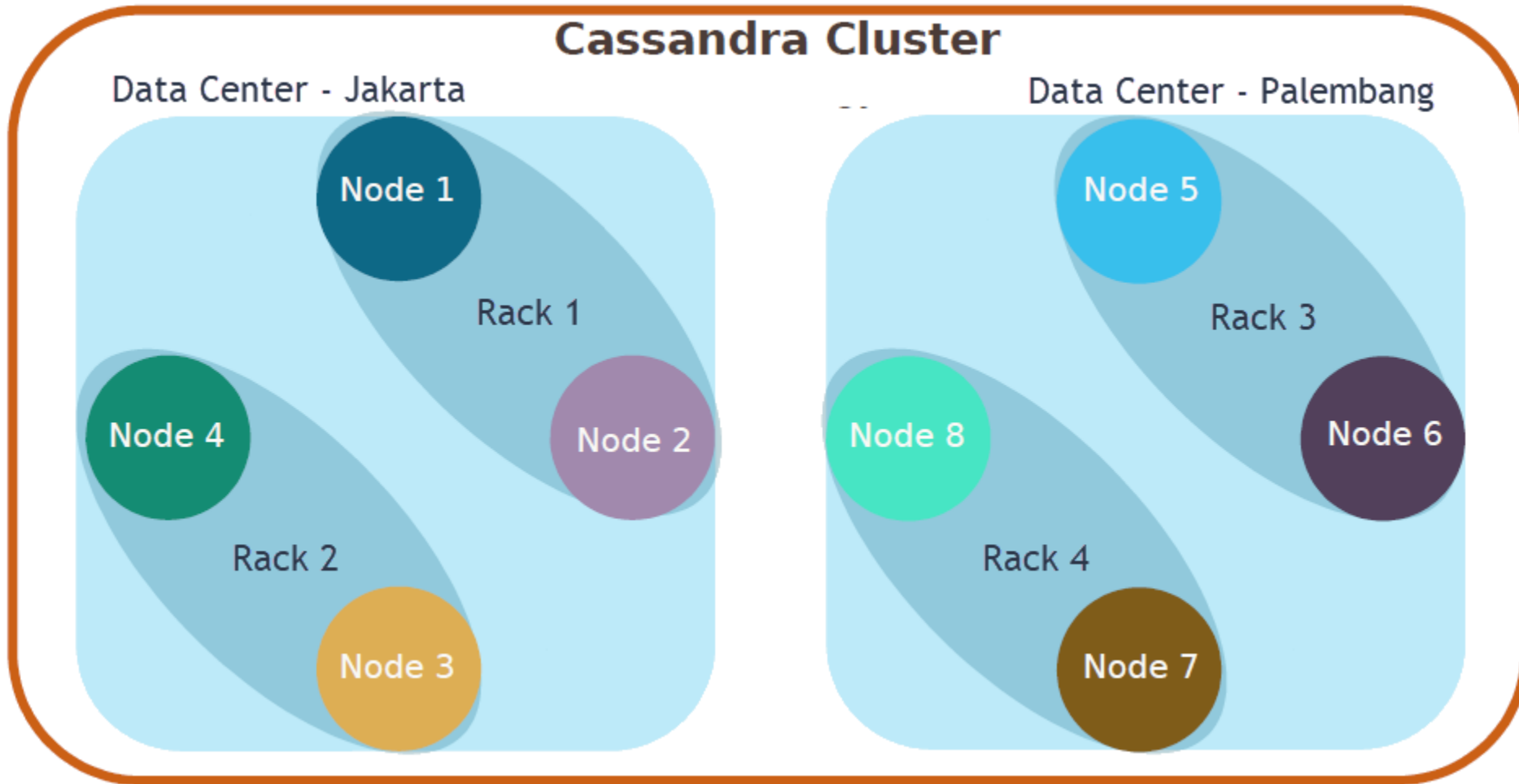
Keyspace (kulcstér) – Adatbázishoz hasonló konténer objektum

Column family (oszlopcsalád) – Tábláknak megfelelő objektumok, rendezett oszlopok gyűjteménye

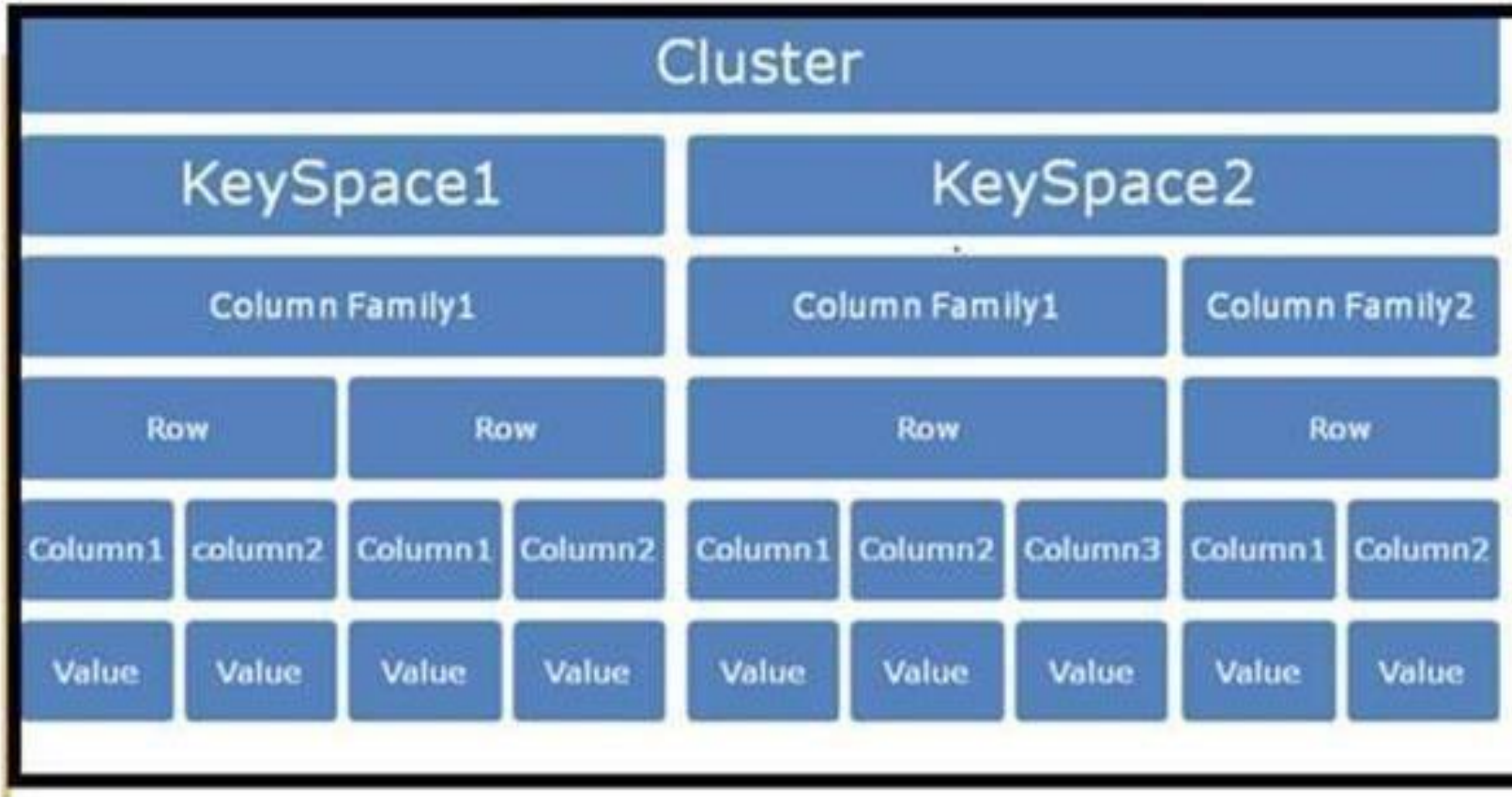
Memtable (memória tábla) – Memória rezidens adatstruktúra, a commit log után ide kerülnek az adatok

SSTable – Fájl, ahova az adatok kerülnek, ha a Memtable megtelik

Cassandra – Node, Cluster, Data Center



Cassandra – architektúra



Cassandra – Column family példa 1.

- Table with single-row partitions

partition key

columns

performer	born	country	died	founded	style	type
John Lennon	1940	England	1980		Rock	artist
Paul McCartney	1942	England			Rock	artist
The Beatles		England		1957	Rock	band

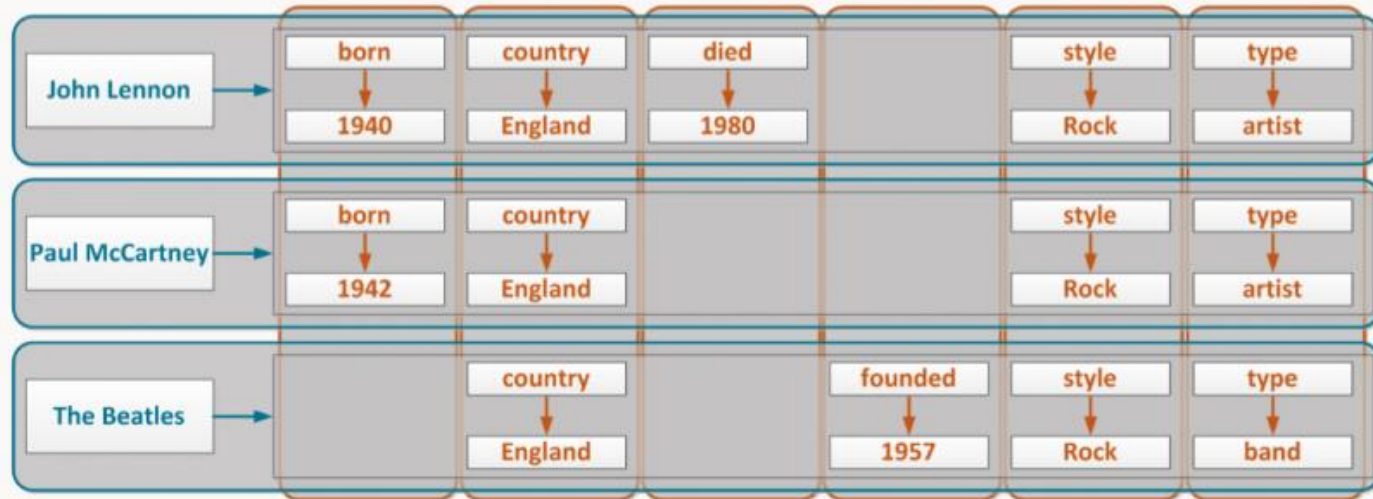
partitions

rows

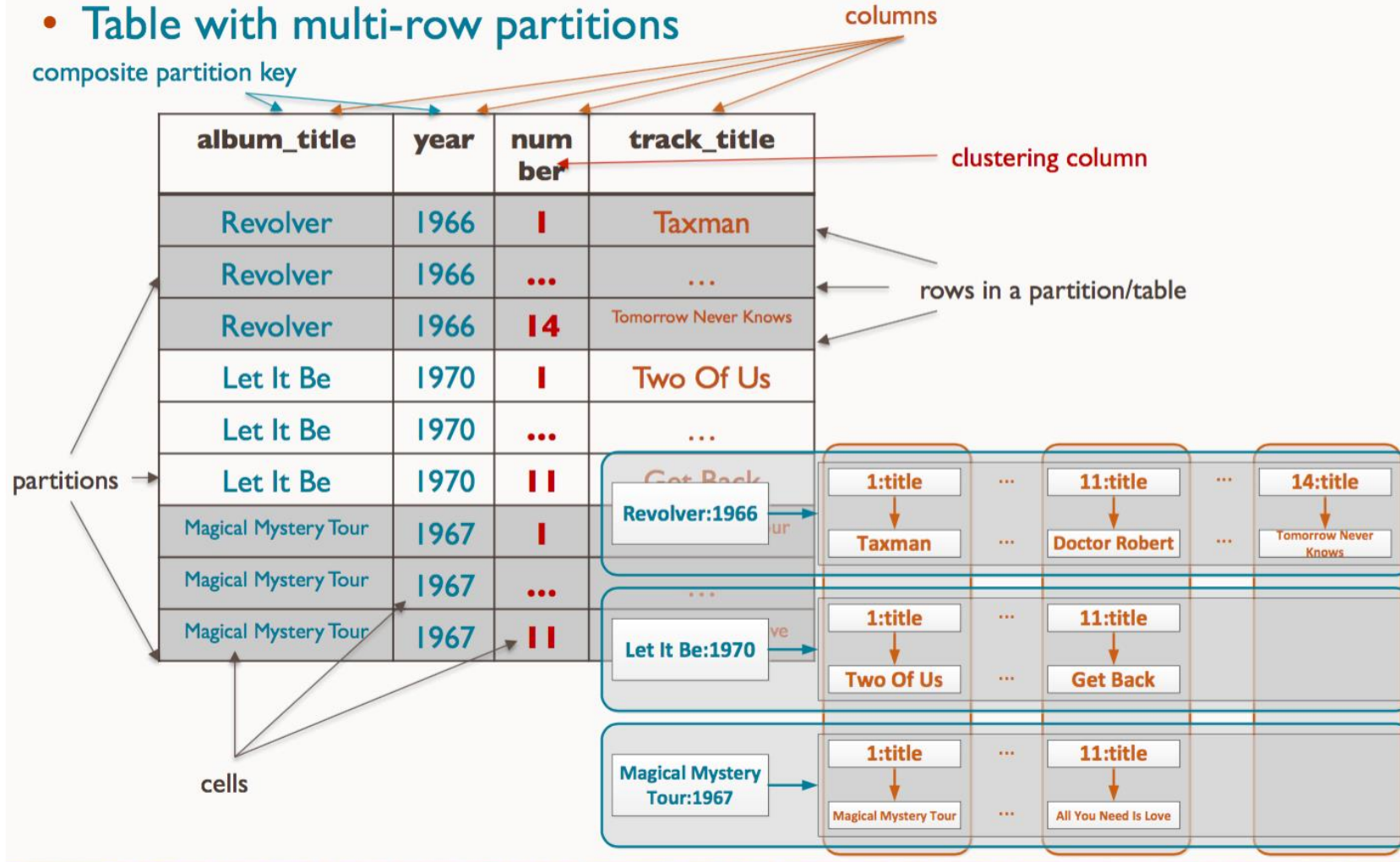
cells

Row key = Partition key

- Column family view



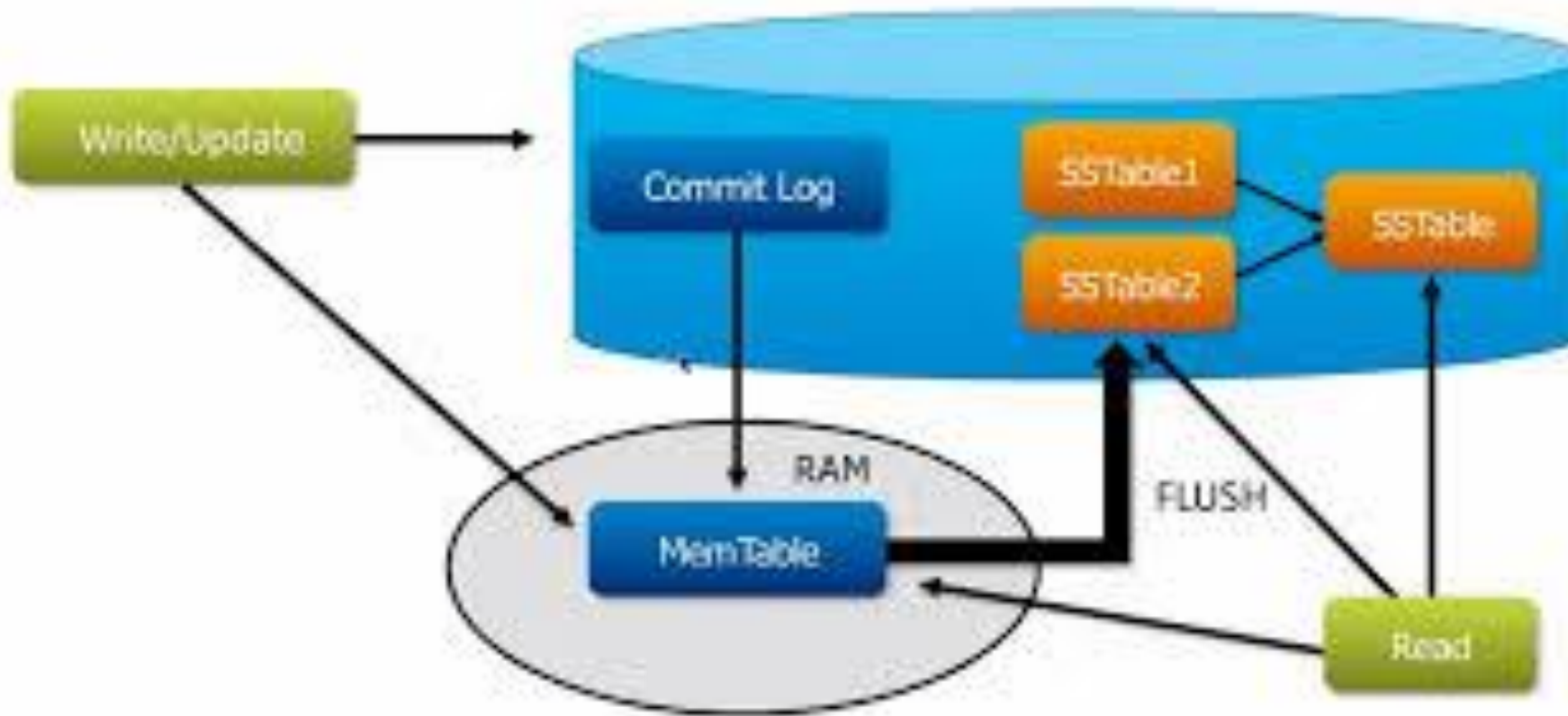
Cassandra – Column family példa 2.



Row key =
Partition key + Clustering key

Cassandra | Mauricio Poppe

Cassandra – Memtable és SSTable



Timestamp (időbélyeg)

- ☐ Minden írási művelet esetén automatikusan képződik minden egyes érték esetén
- ☐ Használatával az esetleges ütközések (konfliktusok) feloldhatók
- ☐ Az időbélyeget a kliens is generálhatja

Time To Live (TTL)

- ❑ Az adat lejáratási idejét határozza meg
- ❑ Adott idő után az adat törlődik
- ❑ A felhasználó szabályoztatja a TTL-eket (retention policy)

```
cqlsh> insert into University.Student(rollno,name,dept,semester) values(3,'Guru9  
9','CS',7) using ttl 100;  
cqlsh>
```

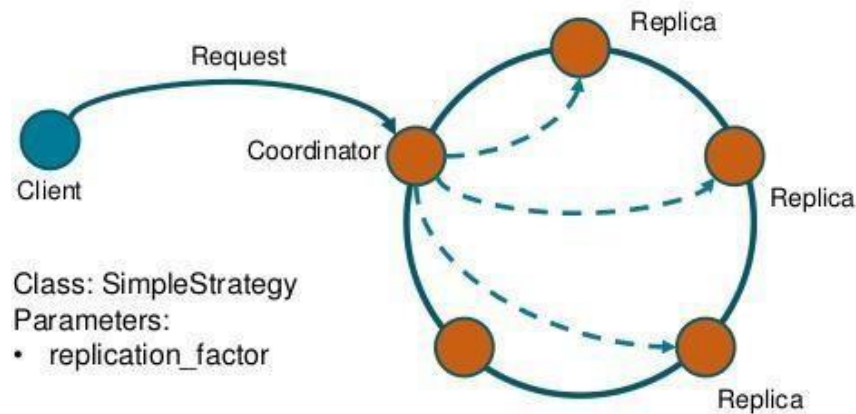
specified ttl value
in using clause

Cassandra – Replikációs stratégiák

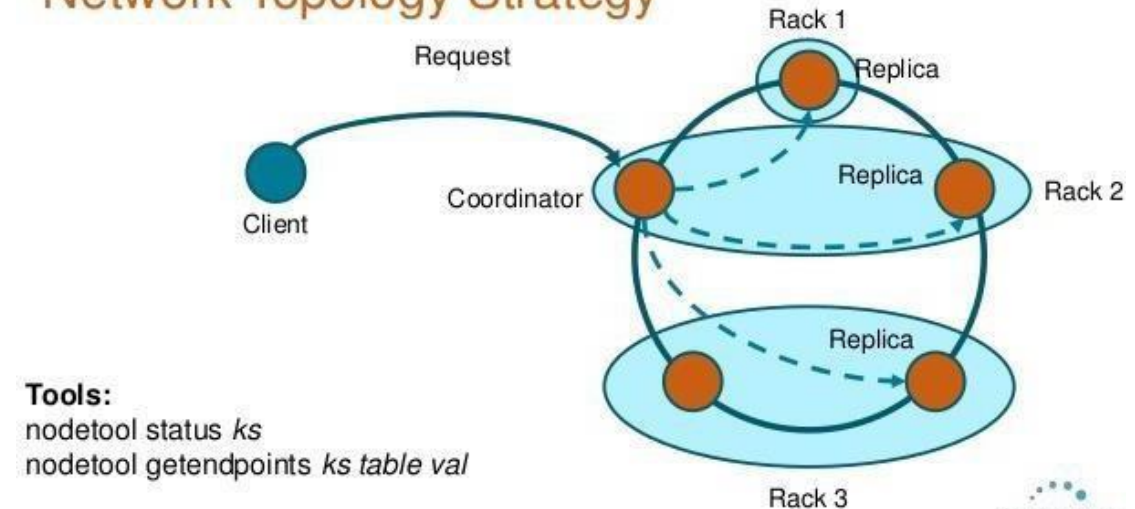
A replikáció során az adatokról több node-on is másolat készül az adatvesztés elkerüléséért.

- A másolatok számát replikációs faktornak nevezzük.
- A másolatok létrehozása többféle stratégia alapján történhet.

Simple Strategy



Network Topology Strategy



(Apache) Cassandra – kipróbálási lehetőségek

- ☐ Online proba ([Try It Out | Datastax](#))
- ☐ On-premise ([Download \(apache.org\)](#))
 - ☐ Telepíthető Linux, Windows és Mac alá is
- ☐ Cloud
 - ☐ Azure Cosmos DB
 - ☐ DataStax Astra

Cassandra – Online próba

Try It Out: Cassandra Query Language (CQL):

< STEP 1 OF 7 >

Create a keyspace

Let's first start learning CQL by creating a keyspace, using the `CREATE KEYSPACE` command.

```
CREATE KEYSPACE demo WITH  
replication = {'class':  
'SimpleStrategy',  
'replication_factor': 1}; ↵
```

A keyspace is a way to logically group a collection of database objects together, such as:

- tables

Terminal



Your Interactive Bash Terminal.

```
$ cqlsh
```

```
Connected to Test Cluster at 127.0.0.1:9042.
```

```
[cqlsh 5.0.1 | Cassandra 4.0-beta2 | CQL spec 3.4.5 | Native protocol v4]
```

```
Use HELP for help.
```

```
cqlsh>
```

```
cqlsh>
```

```
cqlsh>
```

Aktiválja a Wind

Aktiválja a Windows r

Po

Cassandra – Azure Cosmos DB

olgáltatások és dokumentumok keresése (G+ /)

geza.molnar@stud.uni-...
CORVINUS UNIVERSITY OF BUD...

cas1 | Adatkezelő
Azure Cosmos DB-fiók

Keresés (Ctrl+ /)

Áttekintés
Tevékenységnapló
Hozzáférés-vezérlés (IAM)
Címkék
Problémák diagnosztizálása é...
Első lépések
Értesítések
Adatkezelő
Beállítások
Kapcsolati sztring
Funkciók
Adatok globális replikálása
Alapértelmezett konzisztencia
Biztonsági mentés és visszaállí...

New Table
Enable Azure Synapse Link
New Notebook
Connect to GitHub

CASSANDRA API

DATA
NOTEBOOKS
Gallery
My Notebooks

Welcome to Cosmos DB

Globally distributed, multi-model database service for any scale

New Table
Create a new container for storage and throughput

New Notebook
Create a notebook to start querying, visualizing, and modeling your data

Common Tasks
New Keyspace

Recents

Tips
Data Modeling
Learn more about modeling
Cost & Throughput

Aktiválja a Windows...
Aktiválja a Windows rendszert a Gépházban.

Cassandra – DataStax Astra

DataStaxAlready have an account? [Sign in](#)

Get Your Free Astra Database

Email

Password

How are you using DataStax Astra?

Select an option

☐ I agree to the [DataStax MSA](#), including the [Astra Supplement](#).

Create Account

DataStax Astra Cassandra-as-a-Service

OPEN, MULTI-CLOUD STACK FOR MODERN DATA APPS

- Start in minutes, no credit card required, use for free.
- Eliminate the overhead to install, operate, and scale Cassandra clusters.
- Build faster with REST, GraphQL, CQL, and JSON/Document APIs.
- Built on open-source Apache Cassandra™ used by the best of the internet.
- Scale elastically — apps are viral ready from Day 1.
- Deploy multi-cloud, multi-tenant or dedicated clusters on AWS, Azure, or GCP.
- Ensure enterprise-level reliability, security, and management.

Still have questions? [Check out the FAQ!](#)

Cassandra – On premise – CQL Shell

```
Cassandra CQL Shell
Connected to Test Cluster at 127.0.0.1:9042.
[cqlsh 5.0.1 | Cassandra 2.2.3 | CQL spec 3.3.1 | Native protocol v4]
Use HELP for help.
WARNING: pyreadline dependency missing. Install to enable tab completion.
cqlsh> help

Documented shell commands:
=====
CAPTURE  CLS      COPY  DESCRIBE  EXPAND  LOGIN  SERIAL  SOURCE
CLEAR    CONSISTENCY  DESC  EXIT      HELP    PAGING  SHOW    TRACING

CQL help topics:
=====
ALTER                CREATE_TABLE_TYPES  PERMISSIONS
ALTER_ADD            CREATE_USER         REVOKE
ALTER_ALTER          DATE_INPUT          REVOKE_ROLE
ALTER_DROP           DELETE              SELECT
ALTER_RENAME         DELETE_COLUMNS      SELECT_COLUMNFAMILY
ALTER_USER           DELETE_USING        SELECT_EXPR
ALTER_WITH           DELETE_WHERE        SELECT_LIMIT
APPLY               DROP                SELECT_TABLE
ASCII_OUTPUT        DROP_AGGREGATE      SELECT_WHERE
BEGIN              DROP_COLUMNFAMILY  TEXT_OUTPUT
BLOB_INPUT          DROP_FUNCTION       TIMESTAMP_INPUT
BOOLEAN_INPUT       DROP_INDEX          TIMESTAMP_OUTPUT
COMPOUND_PRIMARY_KEYS  DROP_KEYSPACE      TIME_INPUT
CREATE              DROP_ROLE           TRUNCATE
CREATE_AGGREGATE    DROP_TABLE          TYPES
CREATE_COLUMNFAMILY DROP_USER           UPDATE
CREATE_COLUMNFAMILY_OPTIONS  GRANT              UPDATE_COUNTERS
```

Cassandra adattípusok

☐ Egyszerű adattípusok

- ☐ Szöveges: ascii, varchar, inet, text, date, time
- ☐ Numerikus: int, varint, bigint, counter, float , decimal, double
- ☐ Logikai: boolean
- ☐ Egyéb: timestamp, uuid, blob

☐ Összetett adattípusok

- ☐ list - elemek rendezett gyűjteménye
- ☐ map – kulcs-érték párok gyűjteménye
- ☐ set – elemek halmaza

☐ Felhasználó által definiált adattípusok

Cassandra – CQL nyelv

- ☐ Deklaratív nyelv
- ☐ SQL nyelvhez hasonló szintaktika, de más elvekre épül (az adatmodell nem relációs!)
- ☐ A SQL-nél limitáltabb lehetőségek
 - ☐ Nincsenek tábla összekapcsolások, beágyazott lekérdezések
 - ☐ Nincs tranzakciókezelés, tárolt eljárás, trigger
 - ☐ WHERE feltétel csak elsődleges kulcs oszlopra vagy indexelt oszlopra alkalmazható
 - ☐ Adatot módosítani (frissíteni) csak elsődleges kulcs alapján lehet

Verziótól függően további korlátozások is lehetnek, pl. 3.1 verzió előtt nem volt GROUP BY sem

Cassandra – Keyspace létrehozása

CREATE KEYSPACE kulcstérnév WITH tulajdonságok;

☐ Tulajdonságok

- ☐ replication – a replikációs stratégia és a replikák számának megadása, a stratégia lehet: SimpleStrategy, NetworkTopologyStrategy
- ☐ durable_writes – a commitlog használatban legyen-e módosítások esetén?
Logikai változó, alapértelmezett értéke true

☐ Példa

☐ CREATE KEYSPACE webshop

WITH replication = {'class': 'SimpleStrategy', 'replication_factor': 3};

Létrehoz egy webshop nevű kulcsteret, ahol a replikák száma 3, a replikációs stratégia pedig SimpleStrategy

Cassandra – Keyspace kezelő parancsok

USE kulcstérnév; - az aktuális keyspace megadása

PI: USE webshop;

DESC KEYSPACES; - a létező keyspace-ek listázása

PI: DESC
KEYSPACES;

ALTER KEYSPACE kulcstérnév WITH tulajdonságok; - kulcstér módosítása

PI: ALTER KEYSPACE webshop
WITH replication = {'class':
'SimpleStrategy', 'replication_factor':
1};

DROP KEYSPACE kulcstérnév; – adott keyspace törlése

PI: DROP
KEYSPACES webshop;

Cassandra – Tábla létrehozása

CREATE TABLE | COLUMNFAMILY táblanév (oszlop definíciók)
[WITH opciók AND opció];

- ❑ Elsődleges kulcs megadása a PRIMARY KEY kulcsszóval történik az oszlopdefiníciónál vagy a tábladefiníció végén
- ❑ Minden táblánál kötelező elsődleges kulcs létrehozása
- ❑ A WITH után tábla opciók adhatók meg pl. tömörítés vagy klaszter oszlopok (CLUSTERED ORDER BY (oszloplista))
- ❑ Az AND után egyéb opciók adhatók meg, pl. oszlop szerinti rendezettség
- ❑ Példa
CREATE TABLE Termek (ID INT PRIMARY KEY, Name VARCHAR, Price INT);
Létrehozza a Termek táblát három mezővel, ahol az ID az elsődleges kulcs

Cassandra – Kulcsok

- ❑ PRIMARY KEY – Egyedi azonosító
 - ❑ Egyszerű, ha csak egy oszlopból áll
 - ❑ Összetett (composite), ha több oszlopból áll
- ❑ PARTITION KEY – Ez alapján osztódnak el az adatok a node-ok között
- ❑ CLUSTERING KEY – A partíción belül az adatok sorrendjét határozza meg
- ❑ Egyszerű kulcs esetén PRIMARY KEY = PARTITION KEY
- ❑ Összetett kulcs esetén PRIMARY KEY első mezője a PARTITION KEY, a többi a CLUSTERING KEY, pl:
 - ❑ PRIMARY KEY (x) – a PARTITION KEY x
 - ❑ PRIMARY KEY (x, y, z) – a PARTITION KEY x, CLUSTERING KEY y, z
 - ❑ PRIMARY KEY ((x, y), v, w) – a PARTITION KEY (x, y), CLUSTERING KEY (v, w)

Cassandra – Tábla kezelő parancsok

ALTER TABLE | COLUMNFAMILY táblanév
művelet; - Módosítja az adott táblát. A művelet
lehet ADD és DROP

PI: ALTER TABLE
Termek ADD
description TEXT;

DROP TABLE táblanév; - Törli az adott táblát

PI: DROP TABLE
Termek;

TRUNCATE táblanév; - Törli a tábla tartalmát,
a szerkezet megmarad

PI: TRUNCATE Termek;

```
CREATE INDEX [IF NOT EXISTS] [indexnév] ON táblanév [KEYS]  
oszlopnév;
```

- ☐ Új indexet hoz létre az adott oszlop szerint
- ☐ Egy táblához több index is készíthető
- ☐ Az egyszerű típusok többségéhez készíthető index
- ☐ A Cassandra 2.1-től az index készülhet kollekciókhoz (map, set, list) is
- ☐ Példák:
 - ☐ CREATE INDEX i_raktar_nev ON Raktar (nev);
 - ☐ DESC TABLE Raktar; -- mutatja a létrehozott indexet is
 - ☐ DROP INDEX i_raktar_nev;

Cassandra – CRUD utasítások - INSERT

INSERT INTO táblanév (oszlopnevek) VALUES (értékek) [USING opció]

- ❑ Új értékeket szúr be a tábla adott oszlopaiba
- ❑ Az oszlopnevek megadás kötelező, vagy helyette a JSON formátum használható
- ❑ Példák
 - ❑ INSERT INTO Termek (id, name, price) VALUES (1, 'tej', 250);
 - ❑ INSERT INTO Termek JSON '{"id": 2, "name": "kakao", "price": 300}'; *

* Cassandra 2.2 verziótól alkalmazható

Cassandra – CRUD utasítások - UPDATE

UPDATE táblanév SET értékadás(ok) WHERE feltétel(ek);

- ❑ Módosítja a tábla feltételnek megfelelő adatait
- ❑ A WHERE feltételben hivatkozni kell az elsődleges kulcs oszlopra
- ❑ Példák
 - ❑ UPDATE Termek SET price = 200 WHERE id = 1;
 - ❑ UPDATE Termek SET price = 200 WHERE id IN (1, 2);
 - ❑ UPDATE Termek SET price = 200 WHERE name = 'tej';
-- Ez hibás, mert nincs kulcs hivatkozás

DELETE FROM táblanév WHERE feltétel(ek);

- ❑ Törli a tábla feltételnek megfelelő adatait
- ❑ A WHERE feltételben itt is kell kulcsra hivatkozni
- ❑ Példák
 - ❑ DELETE FROM Termek WHERE id = 2;
 - ❑ DELETE FROM Termek WHERE price = 200;
-- Ez hibás, mert nincs kulcs hivatkozás

Cassandra – Map, List, Set példák

```
CREATE TABLE Raktar(id INT, nev VARCHAR, aruk SET<TEXT>, készlet  
MAP<TEXT, INT>, dolgozok LIST<TEXT>, PRIMARY KEY(id));
```

- ❑ INSERT INTO raktar (id, aruk, dolgozok, készlet, nev)
VALUES (1, {'kifli', 'tej', 'kenyer'}, ['Kiss Bela', 'Nagy Ivett'], {'kifli':
200, 'tej': 50, 'kenyer': 130}, 'Raktar01');
- ❑ UPDATE Raktar
SET aruk = aruk + {'zsemle'},
dolgozok = dolgozok + ['Kozepes Jeno'],
készlet['kenyer'] = 500
WHERE id = 1;

Cassandra - Lekérdezések

- ☐ A DISTINCT kulcsszó használható
- ☐ Az oszlopokat elnevezhetjük az AS után
- ☐ A WHERE utáni oszlopok lehetnek elsődleges kulcs oszlopai, indexelt oszlopok vagy clustered oszlopok
- ☐ Az ORDER BY után csak clustered oszlopok adhatók meg, a WHERE után ilyenkor szükséges a kulcsra hivatkozás
- ☐ A GROUP BY oszlopai csak az elsődleges kulcs oszlopai lehetnek megfelelő sorrendben*
- ☐ A LIMIT után megadható, hogy az eredménysorok közül hány jelenjen meg

```
SELECT ...  
FROM ...  
WHERE ...  
GROUP BY ...  
ORDER BY ...  
LIMIT ...
```

*A partition key mindegyik oszlopának szerepelnie kell

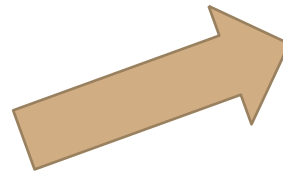
Cassandra adatmodell vs. Relációs adatmodell

A Cassandra esetén a táblák nem kapcsolhatók össze, megoldási lehetőség:

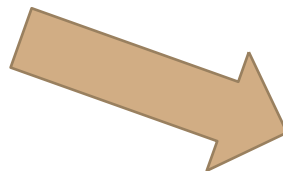
- ☐ Denormalizálás
- ☐ Kollektiók (map, list, set) használata

Tanulo		
tid	tnev	tszulido
1	Kiss Béla	1999.01.01

Vizsga		
datum	tid	eredmeny
2021.06.01	1	5
2021.06.04	1	3

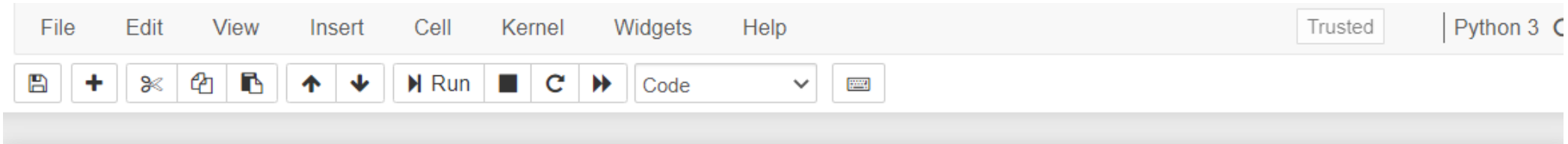


TanuloVizsga			
datum	tid	tnev	eredmeny
2021.06.01	1	Kiss Béla	5
2021.06.04	1	Kiss Béla	3



TanuloVizsga2			
tid	tnev	tszulido	vizsga
1	Kiss Béla	1999.01.01	{'2021.06.01':5 '2021.06.04':3}

Cassandra elérés Python-ból



```
In [ ]: !pip install cassandra-driver
```

```
In [2]: from cassandra.cluster import Cluster
```

```
In [3]: cluster = Cluster(['127.0.0.1'])  
session = cluster.connect()
```

```
In [23]: session.execute('use system;')  
r= session.execute('select * from schema_keyspaces;')  
print(r.current_rows)
```

```
[Row(keyspace_name='kproba', durable_writes=True, strategy_class='org.apache.cassandra.locator.Simple  
Strategy', strategy_options='{"replication_factor":"3"}'), Row(keyspace_name='system', durable_writes  
=True, strategy_class='org.apache.cassandra.locator.LocalStrategy', strategy_options='{}'), Row(keysp  
ace_name='system_traces', durable_writes=True, strategy_class='org.apache.cassandra.locator.SimpleStr  
ategy', strategy_options='{"replication_factor":"2"}')]
```

```
In [24]: cluster.shutdown()
```



**Köszönöm
a figyelmet!**