# NoSQL, Spatial Databases

Přednáška č. 9

RDB

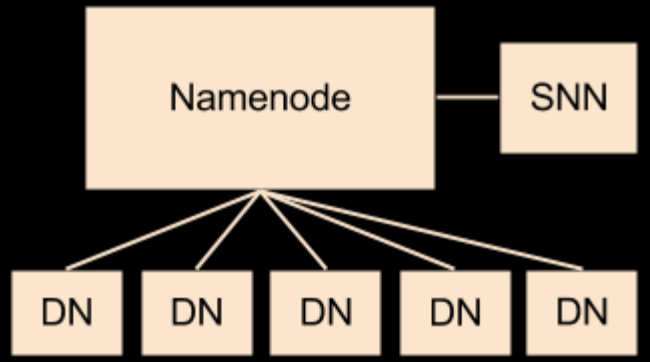
Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# Hadoop

- Hadoop poskytuje:
  - Distribuovaný systém souborů HDFS (Hadoop Distributed File System)
  - Framework a API pro vytváření a běh tzv. *MapReduce* úkonů

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# HDFS




- Strukturováno podobně jako UNIX file systém, ale je distribuované mezi uzly (počítači).
- Typy počítačů v HDFS klasteru:
  - **Datanode** - několik dle toho kolik máme uložených dat.
  - **Namenode** - ‘master’ počítač, který kontroluje meta data pro celý klaster (např. jaké bloky tvoří soubor, na jakých datanode jsou tyto bloky uloženy).
  - **Secondary Namenode** - nejedná se o backup, ale je to jiná služba, které ukládá kopie logů a filesystem image. Čas od času je spojí (merge) aby udržoval velikost na rozumné hodnotě.
- HDFS je navržen pro „non-realtime“ aplikace, které požadují vysoký výkon (propustnost).
- Díky více datanodes je jeho škálovatelnost tzv. lineární, takže je možné zvládnout zátěže, které by žádný počítač nezvládl.
- HDFS vlastnosti vhodné pro distribuované systémy:
  - **Failure tolerant** - data mohou být duplikována na mnoho počítačů (datanodes) . Obecně se mluví o míře replikace 3 (vše je uloženo na 3 počítačích)
  - **Scalability** - přenos dat je řešen přímo na datanodes, takže s větším počtem datanodes stoupá výkon celého systému
  - **Space** - stačí přidat nový datanode
  - **Industry standard** - mnoho průmyslových aplikací staví na HDFS (HBase, Map-Reduce)

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

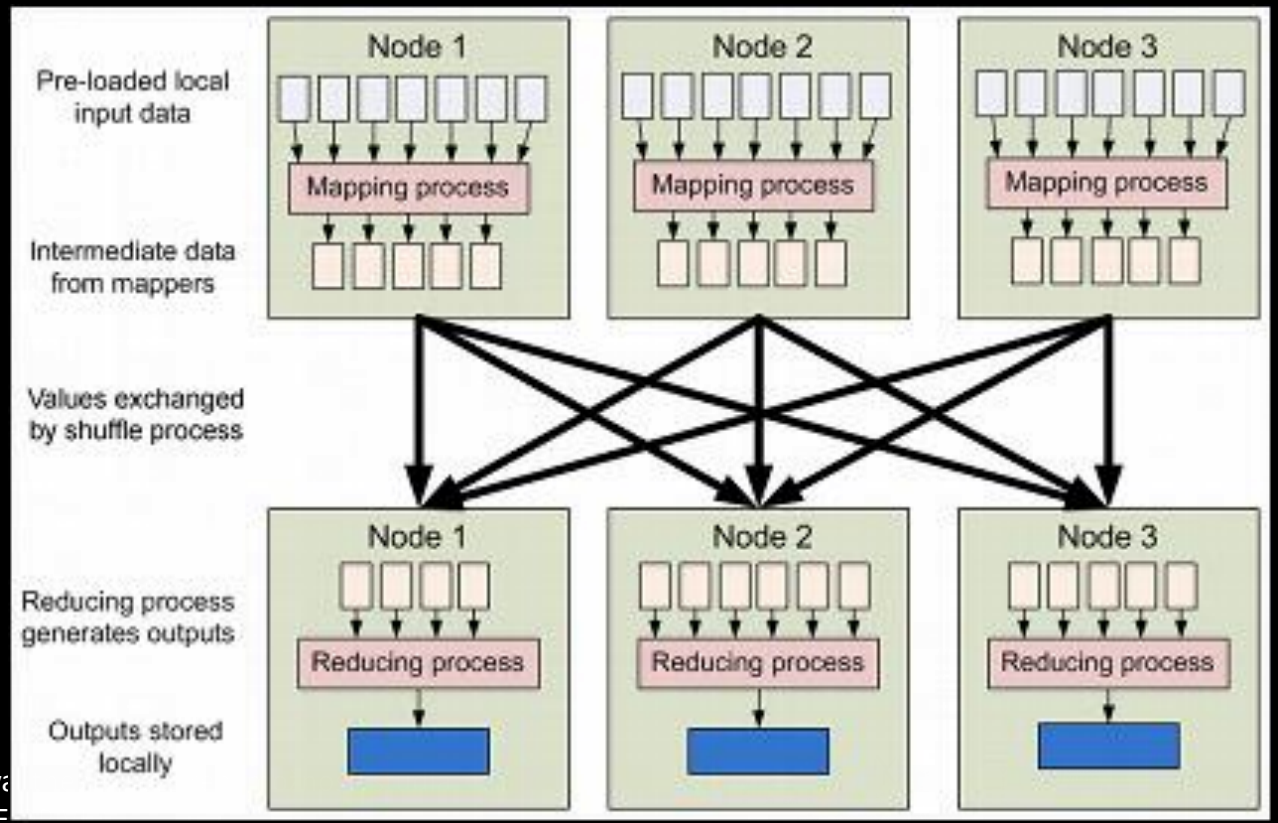
# MapReduce

- MapReduce je Framework pro zpracování paralelizování úkolů na velkých datech
- **Map:** master node vezme vstup a rozdělí jej do menších bloků (sub-problems) a ty distribuuje na další uzly, tyto uzly to mohou udělat znovu a díky tomu dostáváme stromovou strukturu, která dělá problém na podproblémy. Výsledek je pak vrácen master node.
  - Map může být prováděno paralelně na více uzlech
- **Reduce:** master node získá všechny mezi výsledky a vytvoří finální výstup
- Obsahuje v HDFS:
  - API propsaní MapReduce workflows v Javě.
  - Množinu služeb pro správu spuštění takovýchto workflows.

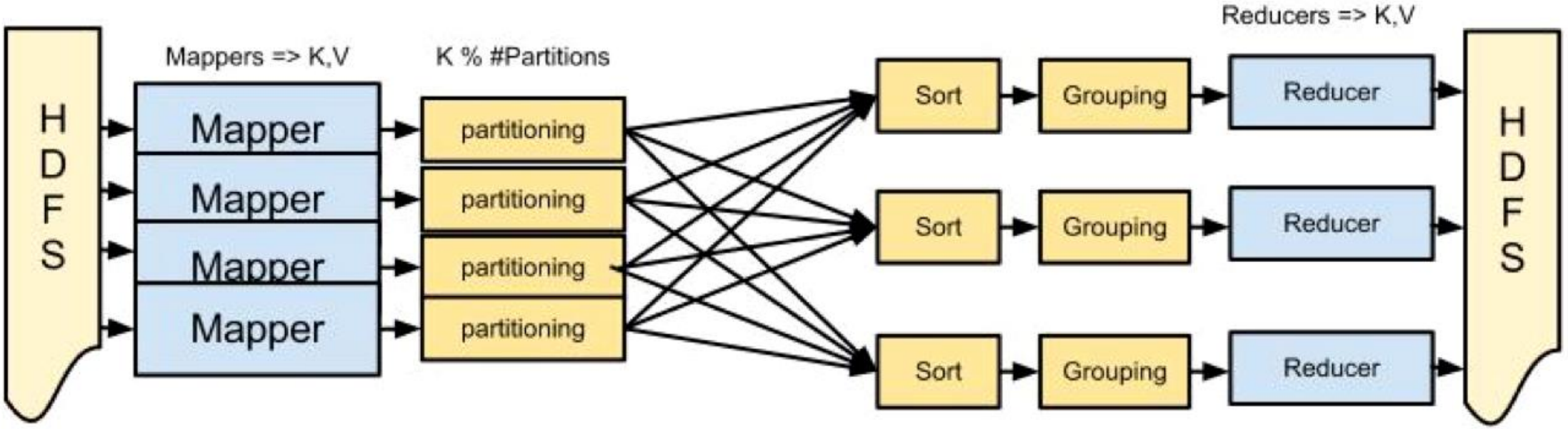
Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# MapReduce API

- Předpoklady
  - Map dělá transformaci.
  - Reduce dělá agregaci
  - Výstupem je vždy dvoje key, value
- Reduce:
  - Vstupem je KEY, ITERABLE[VALUE]
  - Např. Map nám vytvořil dvojice:
    - map1: key: foo, value: 1
    - map2: key: foo, value: 32
  - Reduce pak dostane
    - key: foo, values: [1, 32]




Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkva
na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.




## The MapReduce Pipeline

A mapper receives (Key, Value) & outputs (Key, Value)
A reducer receives (Key, Iterable[Value]) and outputs (Key, Value)
Partitioning / Sorting / Grouping provides the Iterable[Value] & Scaling

# FRAMEWORK A API PRO VYTVÁŘENÍ A BĚH MAPREDUCE ÚKONŮ

- Základní úkoly:
  - *Job Tracker (JT) – odpovídá za alokaci TT a plánování úkolů z globálního hlediska*
  - *Task Tracker (TT)*. Odpovídá za spuštění vlastních MapReduce
- Jak se řeší případné problémy:
  - **Automatické znovuspuštění** - pokud úkol N x selže (obecně se doporučuje 3x), pak se stejný úkol spustí na jiném TT.
  - **Vyřadit špatné TT** – pokus TT má příliš mnoho chyb, pak je vyřazen z použití
  - **Vícenásobné spuštění** - JT naplánuje stejný úkol na více počítačů, aby v případě, že některé neprojdou (nebo budou pomalé), tak dostal alespoň od některých výsledek (v případě pomalých pc jsou po přijetí prvního výsledku ostatní ukončeny).

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# HBase

- HBase je sloupcově orientovaná NoSQL databáze
- Jinými slovy*:*
  - ***Sparse, Consistent, Distributed, Multidimensional, Sorted map*.**
- Řádkově orientované (Row-oriented)

  1,Smith,Joe,40000;
  2,Jones,Mary,50000;
  3,Johnson,Cathy,44000;
- Sloupcově orientované (Column-oriented)

  1,2,3;
  Smith,Jones,Johnson;
  Joe,Mary,Cathy;
  40000,50000,44000;

| EmpId | Lastname | Firstname | Salary |
|---|---|---|---|
| 1 | Smith | Joe | 40000 |
| 2 | Jones | Mary | 50000 |
| 3 | Johnson | Cathy | 44000 |

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

- Map:
  - HBase mapuje klíč na hodnotu.
  - Každé takové mapování se nazývá KeyValue nebo buňka.
  - Na základě klíče najdeme hodnotu
- Setříděné (Sorted)
  - Buňky jsou setříděny podle klíčů, abychom mohli efektivně vyhledávat (např. všechny hodnoty jenž mají klíč mezi X a Y"), spíše než vrať hodnotu pro daný klíč.
- Multidimensional
  - Klíče mají sami o sobě strukturu:
  - row-key, column family, column, time-stamp.
  - Pak tedy platí:
    - (rowkey, column family, column, timestamp) -> value
    - *rowkey a value* jsou byte (*column family* musí být tisknutelné ), takže je možné uložit do buňky cokoli co se dá serializovat do pole byte[].

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

- Sparse
  - Hbase ukládá vše jako dvojici klíč-hodnota a tedy řádek není nic víc než jen seskupení takovýchto záznamů.
  - Není tedy třeba žádných null pro nedostupné hodnoty (prostě tam ta buňka nebude).
- Distributed
  - Hbase je připraven rozdělit celé uložiště na stovky až tisíce serverů.
  - Hbase se pak stará samo o rozdělování zátěže.
- Consistent
  - HBase garantuje:
    - Všechny změny se stejným *rowkey* jsou atomické.
    - Při čtení hodnoty je vždy vrácena poslední zapsaná hodnota (potvrzená – commited).

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# Uložiště

- Každé partition se říká tabulka.
- Každá tabulka definuje jednu nebo více tzv. *column families*.
- *Column families* definují vlastnosti uložiště a libovolnou množinu sloupců (*columns)*.
- *Columns* nejsou deklarovány, v principu jsou jen další název pro hodnotu.
- Základní operace:
  - Put (přidat data),
  - Delete (smazat data),
  - Scan (získat buňky),
  - Get (speciální případ scan).

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# *Column Families*

| RowKey | Timestamp | ColumnFamily |
|---|---|---|
| Student1 | t1 | courses:history=”H0112” |
| Student1 | t2 | courses:math=”M0212” |
| Student2 | t3 | courses:history=”H0112” |
| Student2 | t4 | courses:geography=”G0112” |
| Student2 | t5 | courses:geography=”G0212” |

- *složeny z*
  - *prefix*
  - *qualifier*.
- *prefix* je vždy konstantní (fixovaný) a měl by být specifikovaný, když se zakládá tabulka
- *qualifier* je dynamický a nový může být vytvořen za běhu.
- Díky tomu je možné vytvořit nekonečnou množinu kolekcí sloupečků v rámci jedné column family.
- Příklad:
  - Je možná uložit např. nové kurzy pro studenty tím, že se uloží nové „column families“ s novým prefixem *courses*.
  - Abychom získali historii toho, kam chodil studnet1 musíme mít tři hodnoty:
    - values:*Student1;*
    - *courses:history;*
    - *timestamp*.
  - V tabulce výše je vidět, že *Student2* byl zapsán na kurz geologie (timestamp=t4, courses:geography=”G0112”)
  - Nicméně se zapsal znovu (timestamp=t5; courses:geography=”G0212”).
- *Column Families* fungují jako schéma pro tabulky a musí být specifikované, když se vytváří tabulka a je velmi těžké je pak modifikovat
- Díky dynamičnosti *qualifier* je velmi snadné přidávat nové sloupečky do již existujících *column families*.

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# Multidimenzionální

- ***rowkey*** je definován aplikací
- ***rowkey*** také poskytuje způsob jak seskupit buňky
- HBase navíc zaručí, že stejné buňky se stejným rowkey jsou na stejném serveru (RegionServer), díky tomu je možné zaručit ACID pro aktualizace nad stejným rowkey bez nutnosti pro komplikovaný 2LP.
- ***Column families*** jsou deklarovány při vytváření tabulky.
  - Definují: kompresy (compression), počet verzí k uložení, TTL (time to live) a minimální počet verzí.
- ***Columns*** jsou libovolná jména přiřazená aplikací
- ***timestamp*** je long dat typ, který identifikuje, kdy byla daná buňky vytvořena.
- Každá buňka (na rozdíl od řádku) je verzovaná. Místo update se vloží nová verze buňky.
- Představa řádku (key, column value 1, column value 2, NULL, column value 4, ...) není správná, lépe je:
  (rowkey, column family, column1, timestamp) -> column value 1
  (rowkey, column family, column2, timestamp) -> column value 2
  (rowkey, column family, column4, timestamp) -> column value 4
- *Konzistence?*
- HBase zajistí, že všechny nové verze vytvořené jednotlivými Put operacemi pro určitý rowkey jsou buď:
  - Všechny dostupné ostatním klientům nebo nejsou dostupné žádným
  - Operace Get nebo Scan vrátí pouze kombinace verzí buněk pro daný řádek, které existovaly ve stejný časový okamžik. Díky tomu žádný klient neuvidí částečně provedený update nebo delete.
- HBase to dosáhne pomocí tzv. Multi Version Currency Control (mvcc).

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

- *Storage size (velikost uložiště)*
  - *Hbase* ukládá buňky a ty jsou seskupeny pomocí *rowkey* do čehosi co připomíná řádek.
  - Díky tomu, že buňky jsou uloženy individuálně, tak je uložiště fragmentované(sparse).
  - Pokud jsou data přirozeně fragmentovaná (obsahují mnoho prázdných atributů), pak vše funguje dobře.
  - Pokud je ale model hustý (dense) je mnoho místa ztraceno díky tomu, že se ukládá plná identifikace (která obsahuje řadu opakujících se údajů), která může být v extrémním případě větší než vlastní data.
  - Díky tomu, že hodnota je pouze pole bytů, pak je ale možné husté sloupce uložit jako serializované hodnoty do jedné buňky.
- *Jak jsou data fyzicky uložena*
  - Puts a Delete jsou se skupeny do in-memory struktury - MemStore.
  - Než je MemStore aktualizovaný, tak se všechny změny zapíšou do Write Ahead Log (WAL) pro případ pádu a nutné obnovy
  - Když MemStore dosáhne určité velikosti, tak je uložena do StoreFile.
  - StoreFiles jsou pravidelně seskupeny do menšího počtu StoreFiles.

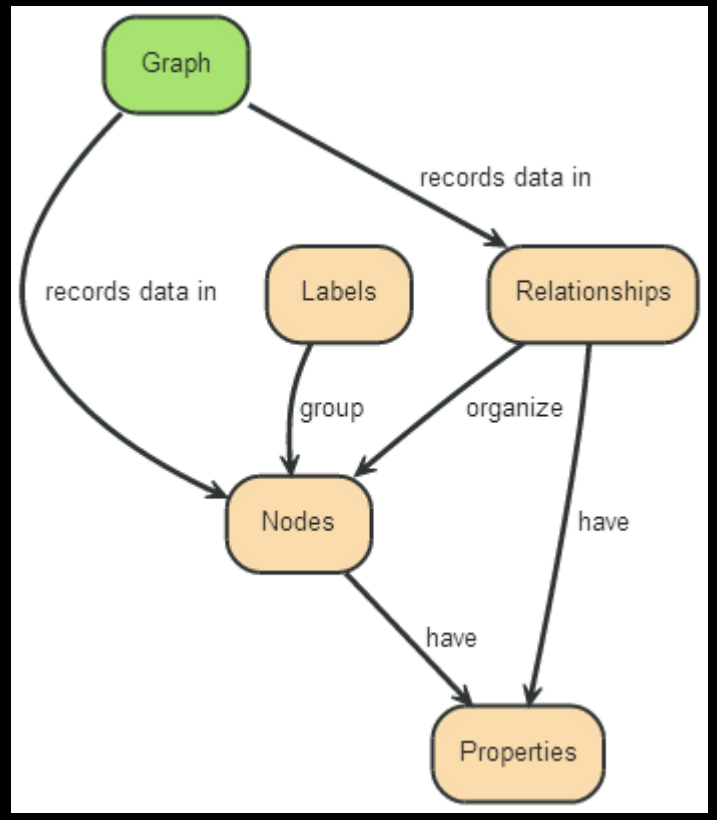
Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# Neo4j

- Neo4j je open source databáze pro uložení grafů
- Neo4j ukládá data jako uzly spojené orientovanou nebo neorientovanou hranou s případnými dalšími vlastnostmi
- **Hlavní vlastnosti:**
  - *Používá grafy pro reprezentaci dat*
  - *Podporuje plný ACID*
  - *Rychlá díky* custom disk-based a native storage engine
  - *Rozšiřitelná*, až biliony uzlů/vztahů/vlastností
  - *Podpora distribuce na více serverů*
  - *Používá graph query language*
  - *embeddable*, pomocí jars souborů
  - *Přístup pomocí REST interface nebo objektově orientovaném API (Java)*

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# Graf obsahující uzly a vztahy




Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# graph query language

- Vrať jeden uzel s id  (The Matrix)

```
MATCH (movie:Movie {title:"The Matrix"})
RETURN movie;
```

- Vrať titul a id uzlu matrix

```
MATCH (movie:Movie {title:"The Matrix"})
RETURN movie.id, movie.title;
```

- Vrať všechny herce (actors)

```
MATCH (m:Movie {title:"The Matrix"})<-[:ACTS_IN]-(actor)
RETURN actor;
```

- Vrať jméno a seřaď dle jména

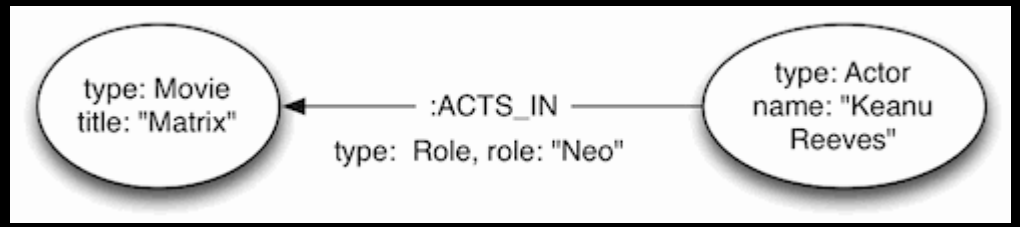
```
MATCH (m:Movie {title:"The Matrix"})<-[:ACTS_IN]-(actor)
RETURN actor.name order by actor.name;
```

- Počet herců

```
MATCH (m:Movie {title:"The Matrix"})<-[:ACTS_IN]-(actor)
RETURN count(*);
```

- Vrať pouze herce jejichž jména končí na "s"

```
MATCH (m:Movie {title:"The Matrix"})<-[:ACTS_IN]-(actor)
WHERE actor.name =~ ".*s$"
RETURN actor.name ;
```




Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# SQL X NoSQL

- Zamyšlení nad SQL a NoSQL

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“

# Budoucí vývoj NoSQL

- V roce 2013 se objevují zprávy, že Google se vrací zpět k SQL.
- Může to znít velmi zvláštně, když to byl Google a jeho MapReduce a BigTable, které dali sílu pro vývoj NoSQL.
- Nicméně tu vždy byly pokusy pracovat s daty v NoSQL databázi pomocí SQL.

Realizováno za finanční podpory ESF a státního rozpočtu ČR v rámci v projektu " Zkvalitnění a rozšíření možností studia na TUL pro studenty se SVP Registrační číslo projektu: CZ.1.07/2.2.00/29.0011 – ESF OP VK.“