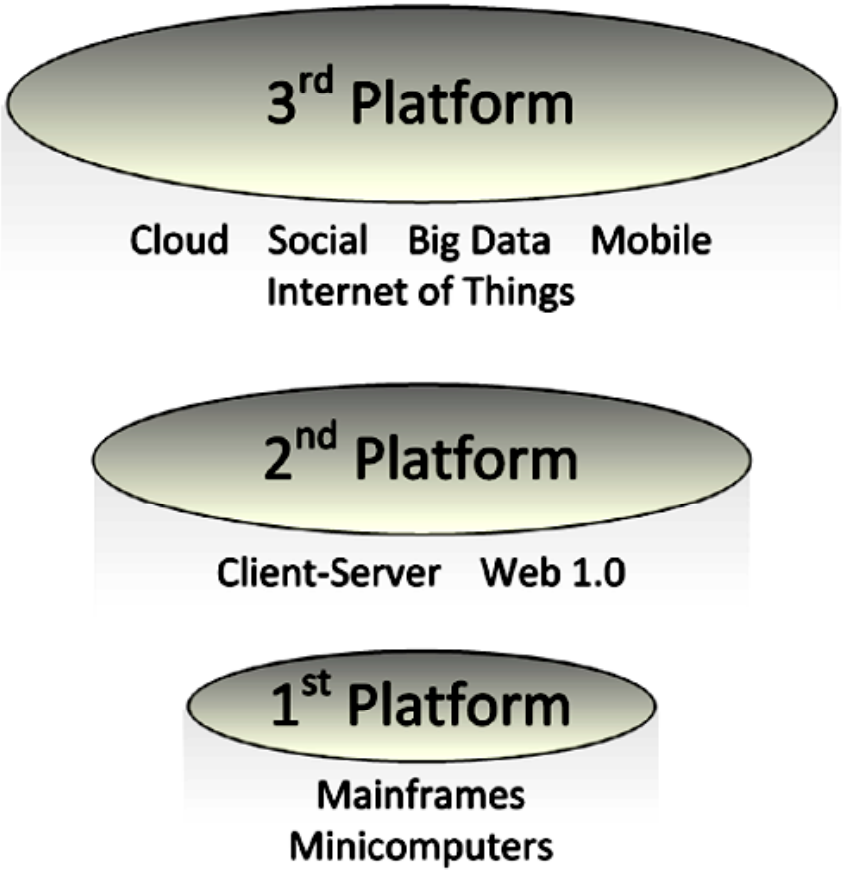
Nosql Database

عادل قاضی خانی
هیئت علمی گروه کامپیوتر دانشگاه امام رضا (ع)

تعریف

- **نوعی پایگاه داده** که از مفاهیم **پایگاه داده های رابطه ای** استفاده نمی کند

نسلهای پایگاه داده ها



Relational Database
NoSQL
NewSQL
Big Data platforms

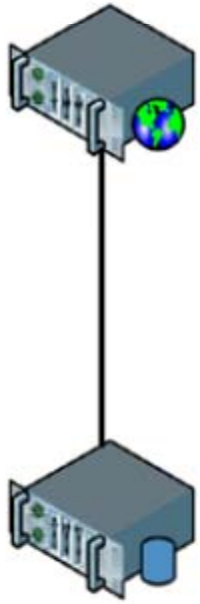
Relational Database

Hierarchical Database
Network Database
ISAM files

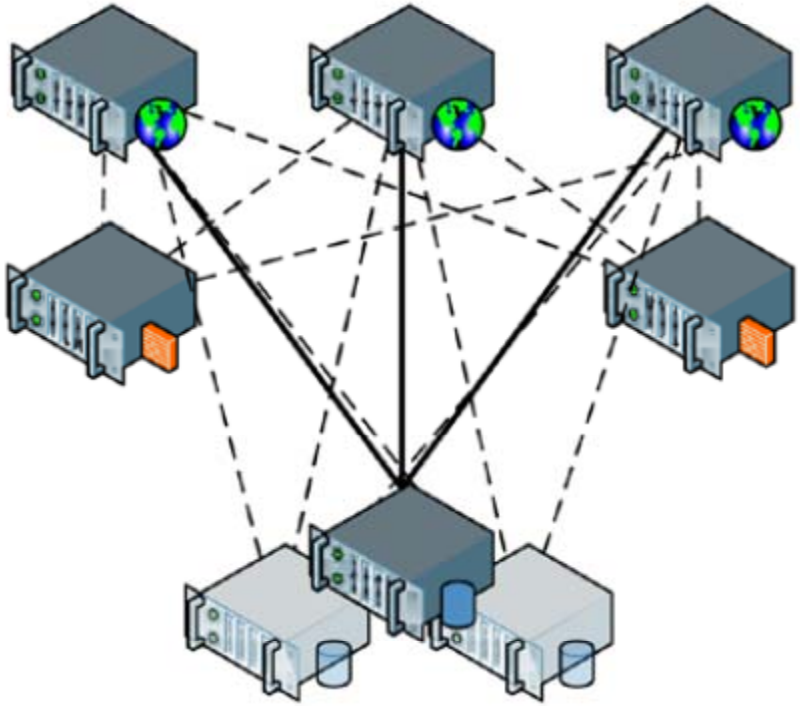
تاریخچه

- در ابتدا وب با صفحات ثابت مرتبط شروع شد (HTML)
- در ادامه سایتهای پویا مبتنی بر پایگاه داده غالب شد (CGI,ASP.NET,PHP,J2EE,...)
- در ادامه مقیاس پذیری مهم شد که با افزایش سرویس دهنده ها وب و منابع سخت افزاری و تکنولوژی های خاص مدیریت شد

مقیاس پذیری



Single database & single web server



Scaling up by adding 2 web servers, 2 Memcached servers, and 2 read-only database slaves

- – – Read only traffic
—— Read/Write traffic

مقیاس پذیری

- **Memchached server**
- تکنولوژی که دسترسی به پایگاه داده را کاهش می دهد. با توزیع کردن cache اشیاء

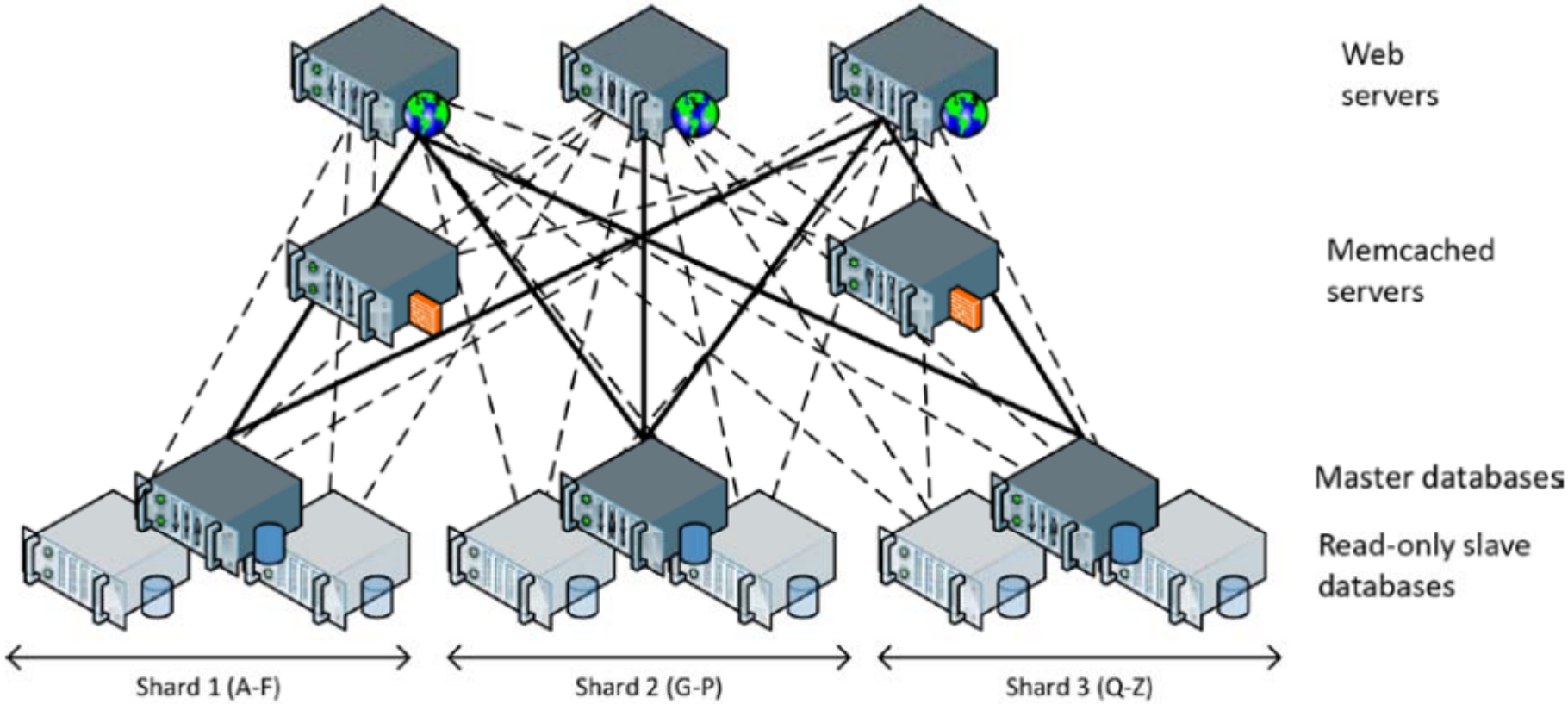
- **Replication**
- ایجاد نسخه ها کپی از داده ها برای مدیریت خواندن داده ها

- هر دو تکنولوژی **دسترسی خواندن را بهبود می دهد نه نوشتن**

مقیاس پذیری

- **Sharding**
- تکنولوژی که با آن می توان یک پایگاه داده بر روی چند سرویس دهنده فیزیکی تقسیم کرد.
- به هر بخش پایگاه داده shard گفته می شود
- بر اساس کلید مقدار
- در برنامه باید مشخص باشد داده ها در کدام سرویس دهنده وجود دارد

مقیاس پذیری



Scaling with 3 web servers, 2 Memcached servers, and 3 database shards with 2 read-only database slaves per shard

--read-only traffic --read/write traffic

مقیاس پذیری

- در سال ۲۰۱۱ ، فیسبوک اعلام کرد ۴۰۰۰ ، shard و ۹۰۰۰ سرویس دهنده Memcached دارد
- پیاده سازی sharding پیچیده
- تنها راه مقیاس پذیری پایگاه داده رابطه ای ، ***Memcached,Replcation,Sharding***

پایگاه داده Amazon

- Amazon برای حل مشکلات خود در سال ۲۰۰۷ ***Dynamo*** را بعنوان یک پایگاه داده غیررابطه ای ارائه کرد و ***Nosql*** متولد شد

پایگاه داده Nosql

- **سه نوع:**

- مبتنی بر سند
- مبتنی بر گراف
- ...

پایگاه داده مبتنی بر سند

- در این نوع پایگاه داده داده ها بصورت اسناد ساختیافته معمولاً XML یا JSON ذخیره می شوند
- مختار هستند که محدودیتهای پایگاه داده رابطه ای را اعمال کنند یا نه

پایگاه داده مبتنی بر سند

- **پایگاه داده XML**
- بر اساس استاندارد سند XML
- قبل از JSON
- استانداردی برای انتقال داده ها

XML

- **ابزارها و استانداردها**
- XML شامل مجموعه ای از ابزار و استانداردها برای تولید، اعتبارستجی، جستجو و تبدیل اسناد XML هستند شامل:
- Xpath
- Xquery
- XML schema
- XSLT
- DOM

- XPath: ابزاری برای دسترسی به بخشهای مختلف یک سند XML

XML

- **ابزارها و استانداردها**
- **XPath:** ابزاری برای دسترسی به بخشهای مختلف یک سند XML
- **XQuery:** یک زبان پرس و جو برای سند XML
- **XML schema:** برای اعتبارسنجی سند XML
- **XSLT:** برای تبدیل سند XML به یک فرمت دیگر
- **DOM** : یک API برای تعامل با XML

پایگاه داده XML

- مثالی از XQuery

query

```
1  xquery version "3.0";
2  collection("/db/apps/demo/data")//address[contains(city, "Berlin")]
3
```

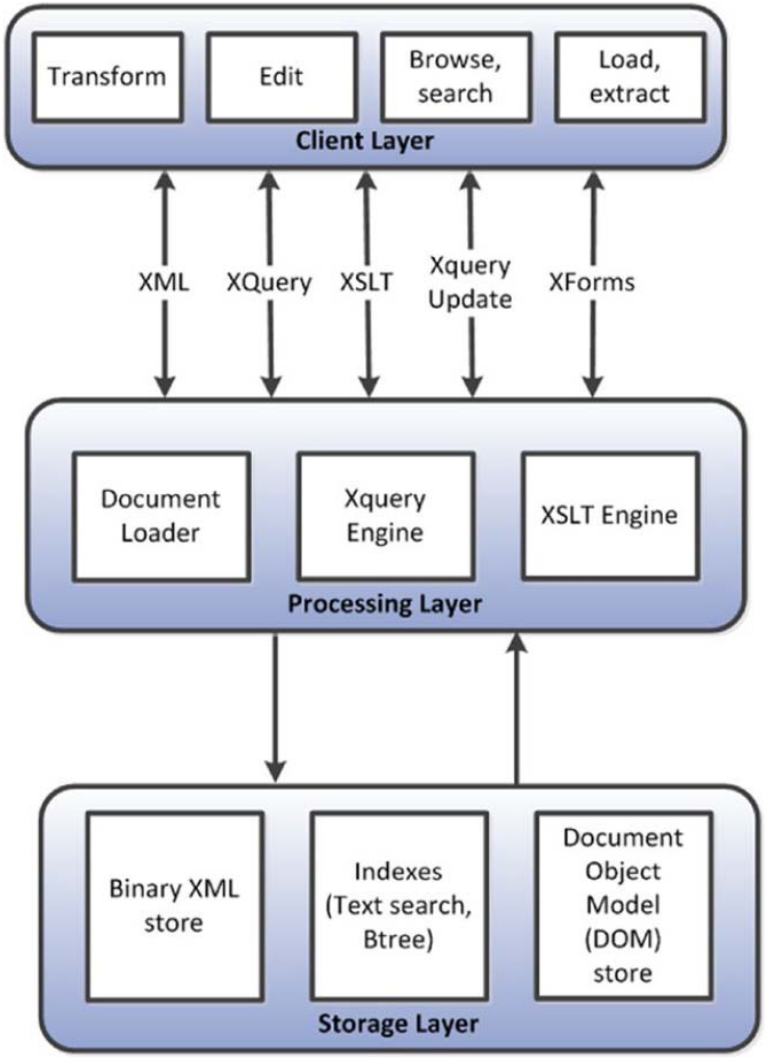
/db/query

XML Output ▾ Live Preview

```
1  <address id="0ff8612a-b998-4677-84a3-73e9ef84ba5f">
       <name>Biene Maja</name>
       <street>Wiesenweg 33</street>
       <city>Berlin</city>
   </address>
```

پایگاه داده XML

- **معماری پایگاه داده**



- **دو نمونه:**
- **eXist**
- **MarkLogic**

پایگاه داده XML

- این پایگاه داده بیشتر برای مدیریت محتوی ، سازماندهی و نگهداری مجموعه ای از فایلهای متنی در فرمت XML

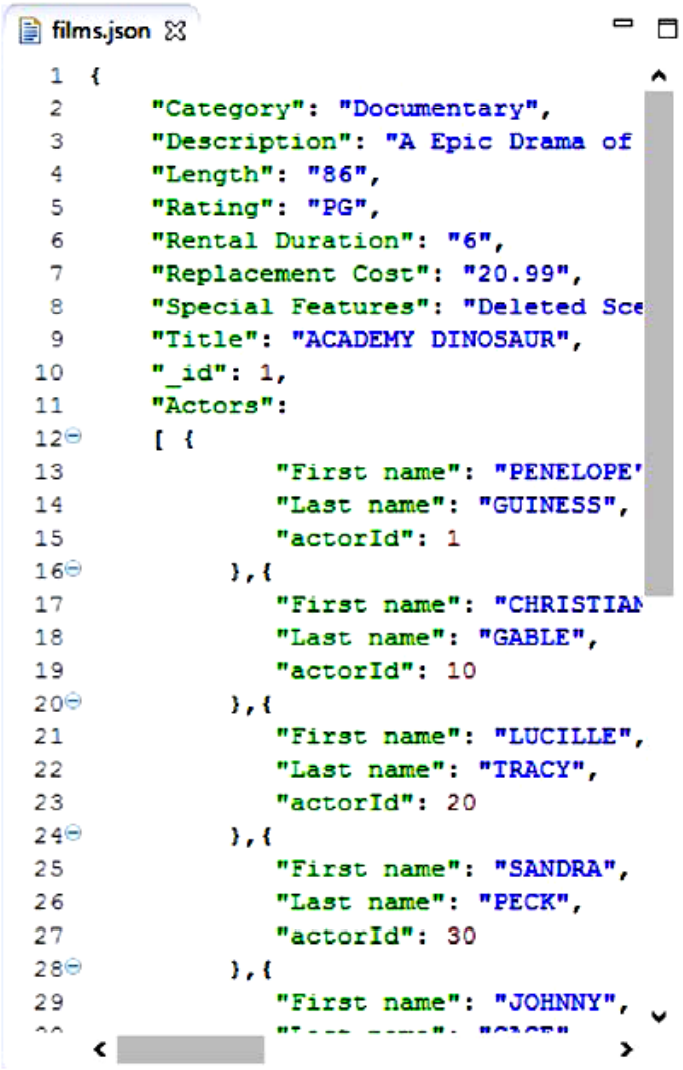
پایگاه داده مبتنی بر سند

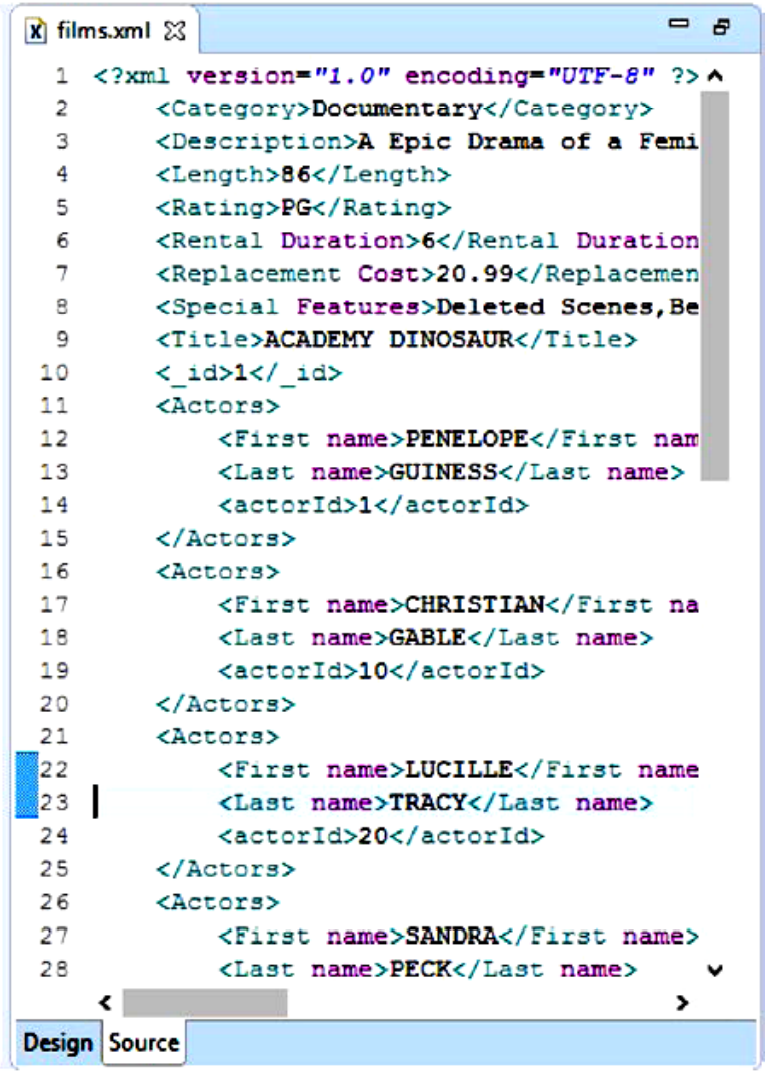
- **پایگاه داده JSON**
- در XML مصرف فضای زیاد و پردازش سنگین
- *JavaScript Object Notation(JSON)*

JSON

- در ابتدا توسط Douglas Crockford برای ایجاد برنامه های وب قویتر ارائه شد و جایگزین XML در برنامه نویسی AJAX

مقایسه ظاهر JSON و XML





پایگاه داده JSON

- در این نوع پایگاه داده مهمترین مسئله ذخیره داده ها در فرمت JSON است
- دو عنصر مهم در JSON
- *document*
- *collection*

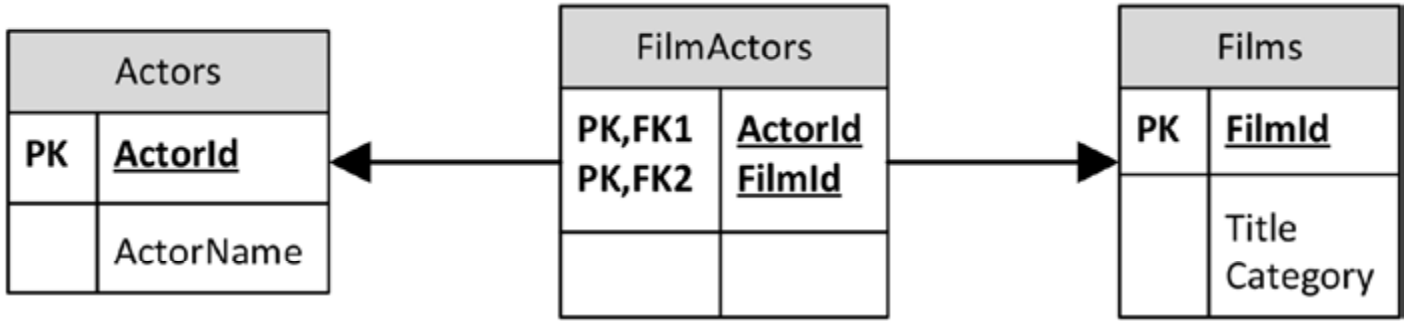
پایگاه داده JSON

- ***document :***
- واحد پایه ای ذخیره سازی، شامل یک یا چند کلید – مقدار ، همچنین می تواند شامل اسناد و آرایه های تو در تو (سطر جدول در RDBMS)
- ***collection:***
- مجموعه ای از اسناد که یک هدف مشترک دارند (جدول در RDBMS)

پایگاه داده JSON

- **مثال: پایگاه داده فیلم**

- **در RDBMS**



پایگاه داده JSON

- **مثال: پایگاه داده فیلم**
- **در JSONDB**

```
{ "_id" : 97, "Title" : "BRIDE INTRIGUE",
              "Category" : "Action",
   "Actors" :
     [ { "actorId" : 65, "Name" : "ANGELA HUDSON" } ]
}

{ "_id": 115,"Title": "CAMPUS REMEMBER",
             "Category": "Action",
   "Actors" :
      [
        { "actorId": 8,"Name": "MATTHEW JOHANSSON" },
        { "actorId": 45,"Name": "REESE KILMER" },
        { "actorId": 168,"Name": "WILL WILSON" }
      ]
}

{ "_id" : 105, "Title" : "BULL SHAWSHANK",
               "Category" : "Action",
    "Actors" :
      [ { "actorId" : 2, "Name" : "NICK WAHLBERG" },
        { "actorId" : 23, "Name" : "SANDRA KILMER" } ]
}
```

Film Document

Actor document

Array of actors

Collection

پایگاه داده JSON

- **مثال: پایگاه داده فیلم**
- **در JSONDB ، می توان اسناد را مرتبط کرد**

```
{ "_id": 115,"Title": "CAMPUS REMEMBER","Category": "Action",
    "Actors": [8,45,168]}
```

```
{ "actorId": 168,"Name": "WILL WILSON" }
```

```
{ "actorId": 45,"Name": "REESE KILMER" }
```

```
{ "actorId": 8,"Name": "MATTHEW JOHANSSON" }
```

پایگاه داده JSON

- **چند نمونه**

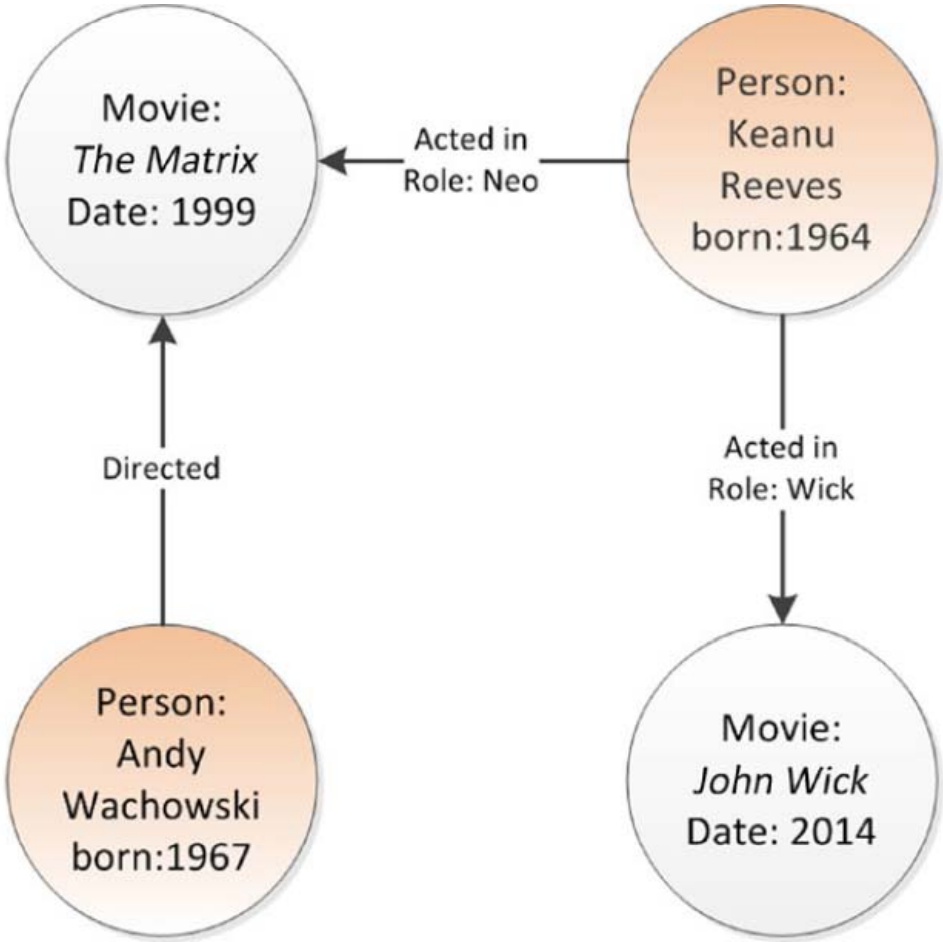
- CouchDB
- MemBase and CouchBase
- MongoDB

پایگاه داده مبتنی بر گراف

- مبتنی بر تئوری گراف
- گراف شامل: رئوس، یالها که می توانند صفاتی داشته باشند

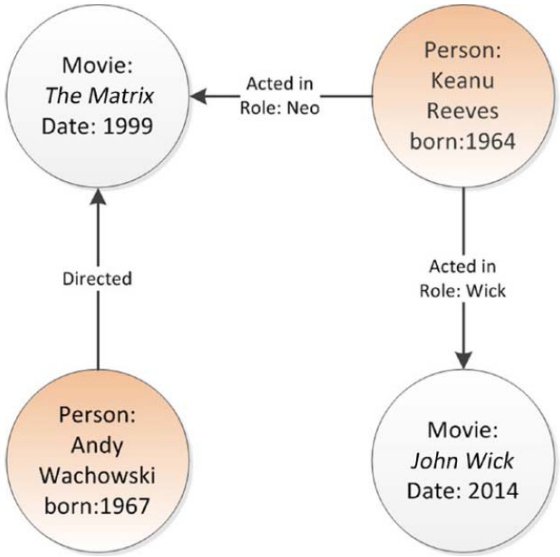
پایگاه داده مبتنی بر گراف

- مثال

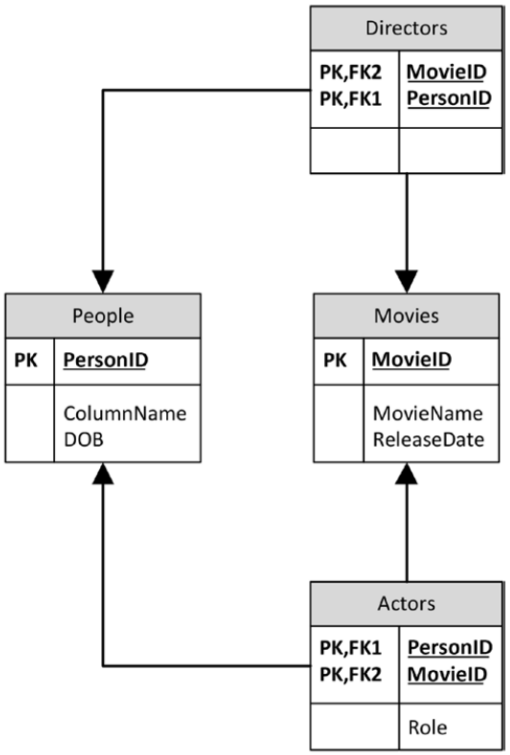


پایگاه داده مبتنی بر گراف

- **مثال در RDBMS؟**



=



?

- **عملیات گراف در SQL مشکل است**

پایگاه داده مبتنی بر گراف

- انواع
  - RDF
  - Neo4j

پایگاه داده RDF

- در RDF اطلاعات با سه گانه موجودیت ، صفت و مقدار بیان می شوند

- **مثال**

```
TheMatrix: is :Movie
Kenau: is :Person
Kenau: starred in: TheMatrix
```

پایگاه داده RDF

- انواع

- AllegroGraph
- Ontotext GraphDB
- StarDog
- Oracle Spatial

پایگاه داده RDF

- زبان پرس و جو : ***SPARQL***

- **مثال:**

Virtuoso SPARQL Query Editor

About | Namespace Prefixes | Inference rules | iSPARQL

Default Data Set Name (Graph IRI)

Query Text

```
SELECT ?object
  FROM <http://dbpedia.org>
 WHERE { <http://dbpedia.org/resource/Edgar_F._Codd>
         <http://purl.org/dc/terms/subject> ?object }
```

(Security restrictions of this

Results Format:

Execution timeout:

Options:

(The result can only be sent

Run Query | Reset

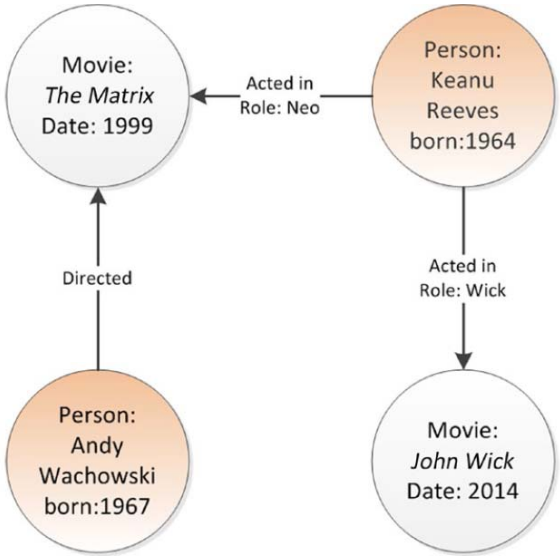
object
http://dbpedia.org/resource/Category:1923_births
http://dbpedia.org/resource/Category:2003_deaths
http://dbpedia.org/resource/Category:Alumni_of_Exeter_College,_Oxford
http://dbpedia.org/resource/Category:British_computer_scientists
http://dbpedia.org/resource/Category:Cellular_automatists
http://dbpedia.org/resource/Category:Deaths_from_myocardial_infarction
http://dbpedia.org/resource/Category:Fellows_of_the_Association_for_Computing_M
http://dbpedia.org/resource/Category:IBM_Fellows
http://dbpedia.org/resource/Category:IBM_employees
http://dbpedia.org/resource/Category:Turing_Award_laureates
http://dbpedia.org/resource/Category:University_of_Michigan_alumni
http://dbpedia.org/resource/Category:Database_researchers
http://dbpedia.org/resource/Category:People_educated_at_Poole_Grammar_School
http://dbpedia.org/resource/Category:People_from_San_Jose,_California

پایگاه داده Neo4j

- بر مبنای گراف خاصیت
- زبان پرس و جو : ***Cypher***
- نتایج دستورات Cypher گراف است

پایگاه داده Neo4j

- **مثال**

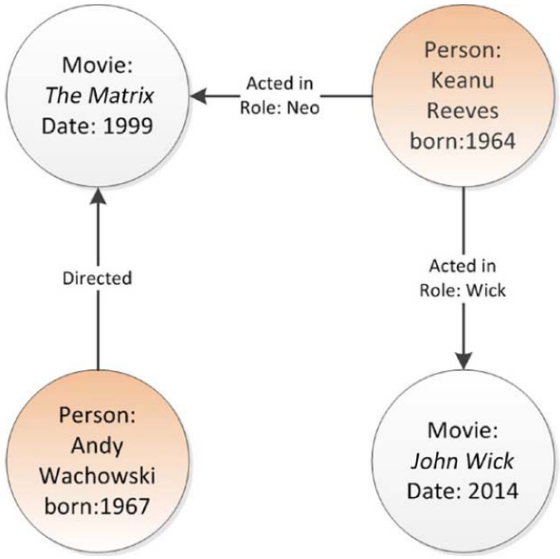


Cypher Query

```
CREATE (TheMatrix:Movie {title:'The Matrix', released:1999,
        tagline:'Welcome to the Real World'})
CREATE (JohnWick:Movie {title:'John Wick', released:2014,
        tagline:'Silliest Keanu movie ever'})
CREATE (Keanu:Person {name:'Keanu Reeves', born:1964})
CREATE (AndyW:Person {name:'Andy Wachowski', born:1967})
CREATE
  (Keanu)-[:ACTED_IN {roles:['Neo']}]->(TheMatrix),
  (Keanu)-[:ACTED_IN {roles:['John Wick']}]->(JohnWick),
  (AndyW)-[:DIRECTED]->(TheMatrix)
```

پایگاه داده Neo4j

- **مثال**

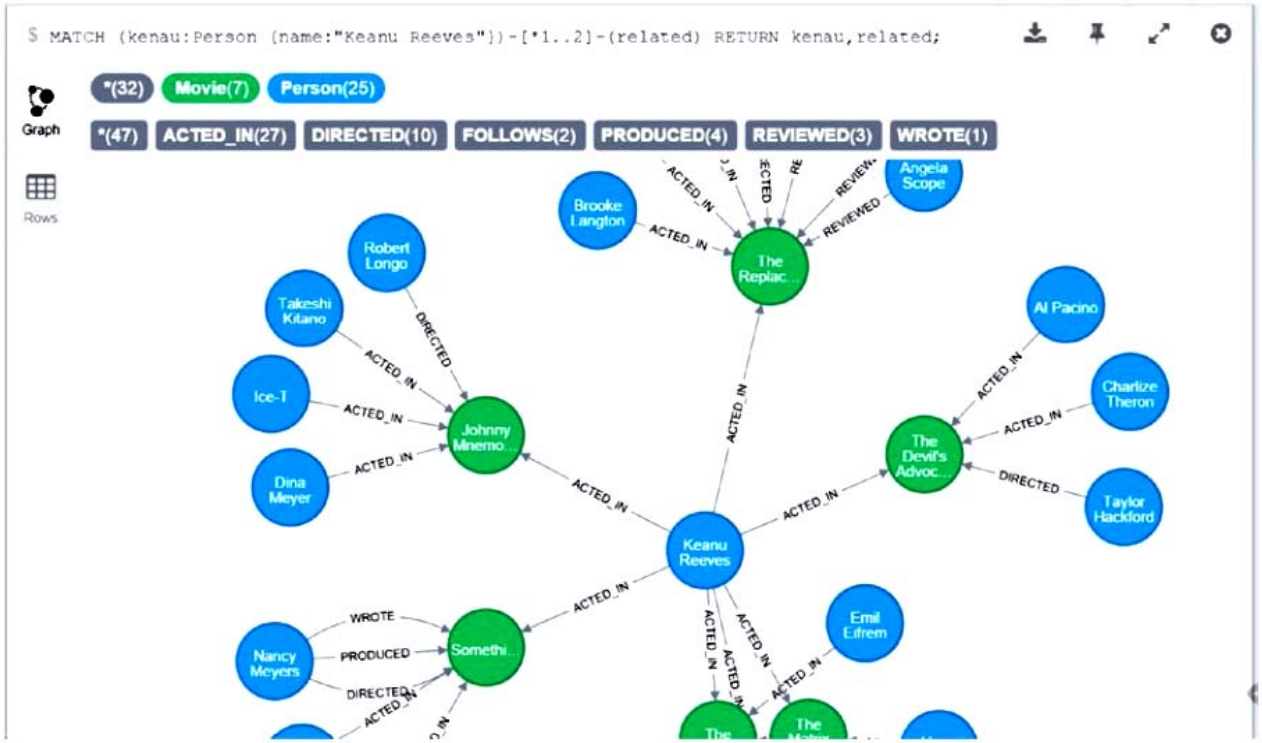


Cypher Query

```
neo4j-sh (?)$ MATCH (kenau:Person {name:"Keanu Reeves"})
>             RETURN kenau;
+-------------------------------------+
| kenau                               |
+-------------------------------------+
| Node[1]{name:"Keanu Reeves",born:1964} |
+-------------------------------------+
```

پایگاه داده Neo4j

- خروجی گراف Cypher

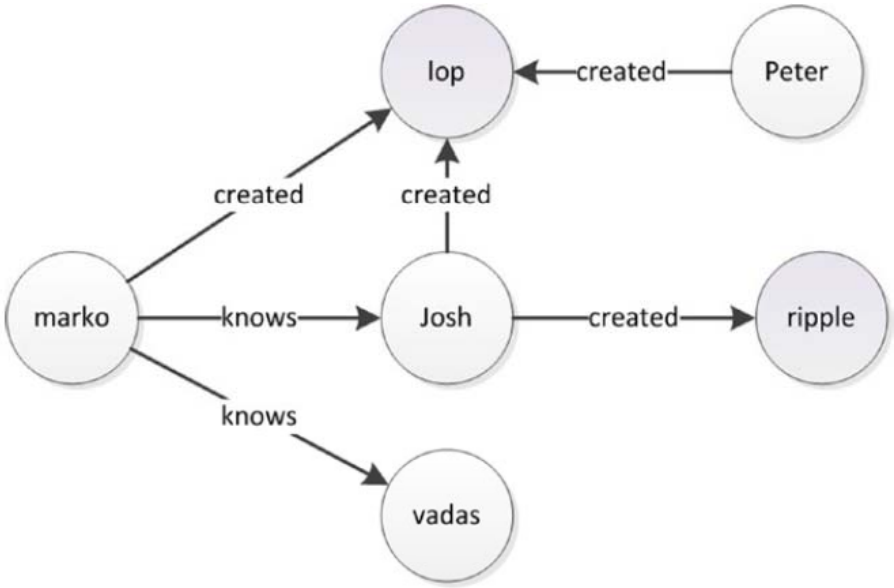


پایگاه داده Neo4j

- یک زبان پرس و جوی دیگر : ***Gremlin***
- نسبت به Cypher رویه ای تر است

پایگاه داده Neo4j

- **مثال**



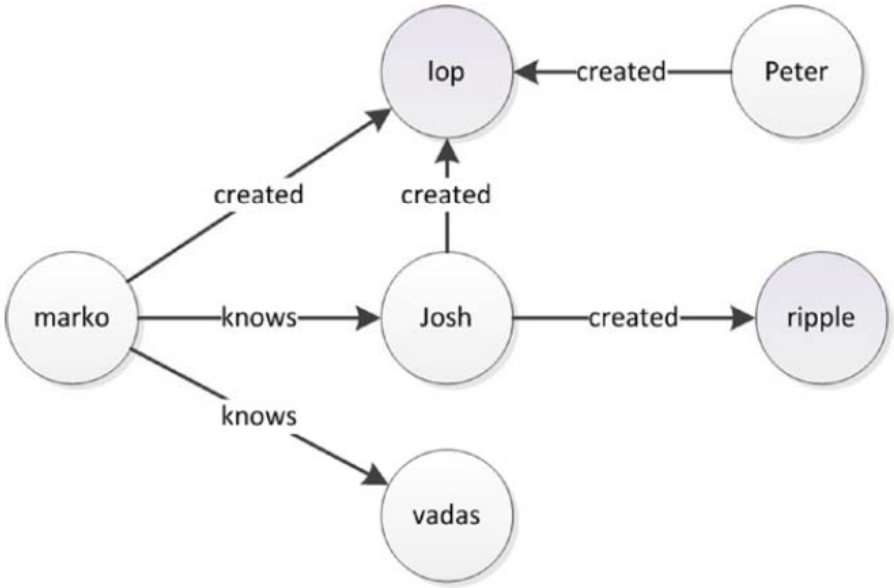
Gremlin Query

```
gremlin> myGraph.V.map
==>{name=marko, age=29}
==>{name=vadas, age=27}
==>{name=lop, lang=java}
==>{name=josh, age=32}
==>{name=ripple, lang=java}
==>{name=peter, age=35}

gremlin> myGraph.E
==>e[11][4-created->3]
==>e[12][6-created->3]
==>e[7][1-knows->2]
==>e[8][1-knows->4]
==>e[9][1-created->3]
==>e[10][4-created->5]
```

پایگاه داده Neo4j

- مثال



Gremlin Query

```
gremlin> marko=myGraph.v(1)
==>v[1]
gremlin> marko.out('knows').name
==>vadas
==>josh
```

موتورهای محاسباتی گراف

- برای پردازش گرافهای حجیم از این موتورها استفاده می شود:
- Apache Giraph **in Hadoop**
- GraphX **in Spark**
- GraphX **in Flink**
- Titan in Big Data storage engines like **Hbase, Cassandra,..**