



دانشگاه صنعتی شریف

دانشگاه فنونی کامپیووتر

طراحی پایگاه داده

(فصل دهم: پایگاه‌های داده خپر را بله‌ای)

مهندی دادبخش

mahdi.dadbakhsh@sharif.edu

۴۰۳۸۴: شماره درس

یکشنبه - سه‌شنبه (۱۶:۳۰ الی ۱۸:۰۰)

۱۴۰۱ - ۱۴۰۲

مُفَاعِلَيْمٌ پَارَهَايِ وَ مِنَابِي

پَارَكَاهَهَايِ دَادَهِي رَابِطَهَايِ

پَارَكَاهَهَايِ دَادَهِي غَيْرِ رَابِطَهَايِ

نُموَنَهَهَايِي ازْ پَارَكَاهَهَايِ دَادَه

معَيَارَهَايِ اِنتَخَابِ پَارَكَاهَهَايِ دَادَهِي مَنَاسِبِ

مقَايِسَه بَرَخَى ازْ پَارَكَاهَهَايِ دَادَه

پَارَانِ

تئوری CAP



مفهوم CRUD



(Data Mining) داده کاوی



پایگاهداده (Database)



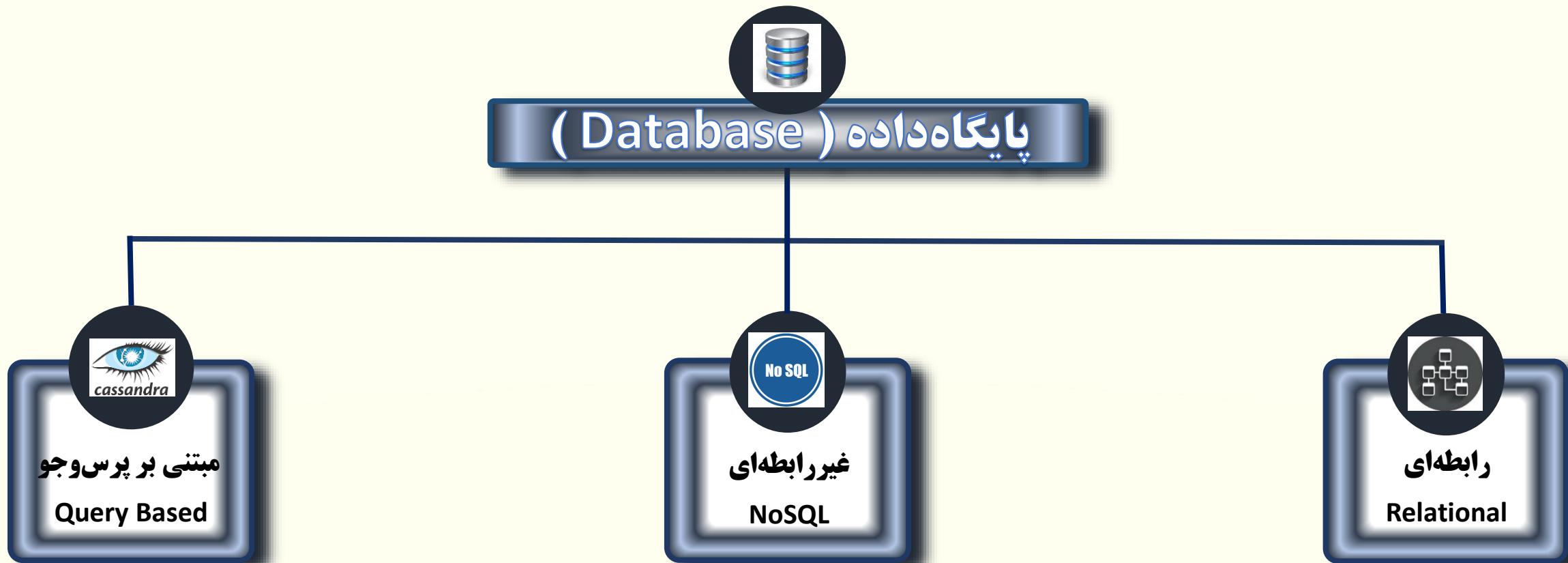
مفهوم ACID

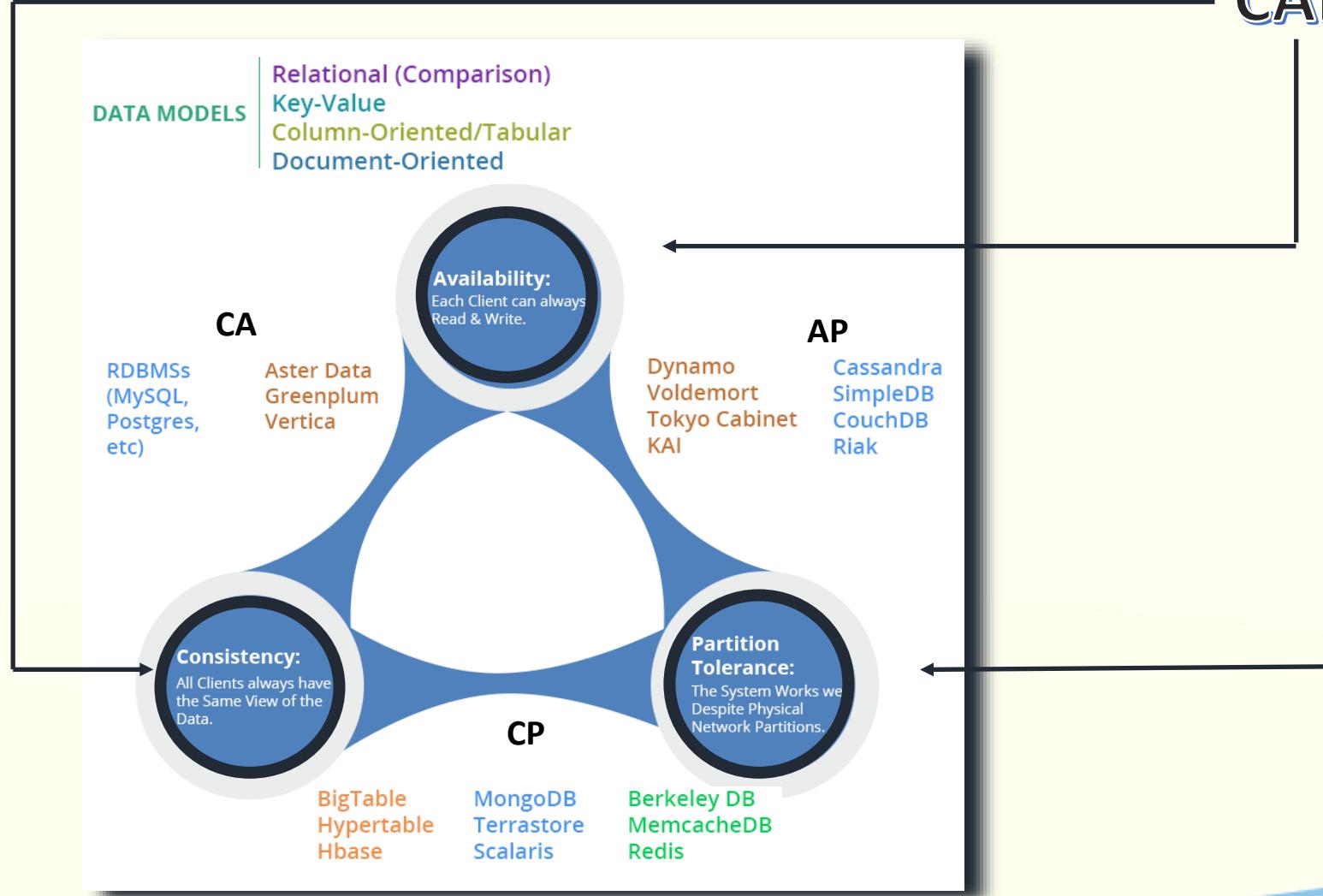


کلانداده (Big Data)



پایگاه داده محلی برای ذخیره، مدیریت و بازیابی داده‌ها می‌باشد و انواع مختلف و متعددی دارد:





a

Atomicity:
Transactions
are all or
nothing

c

Consistency:
Only valid data
is saved

i

Isolation:
Transactions
do not affect
each other

d

Durability:
Written data
will not be lost



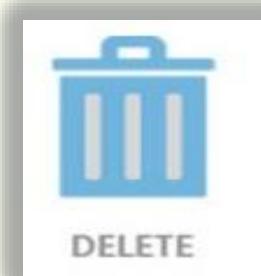
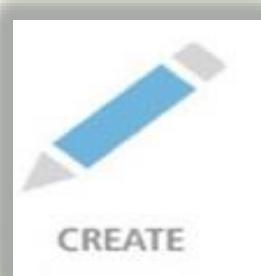
CRUD مفهوم

C

R

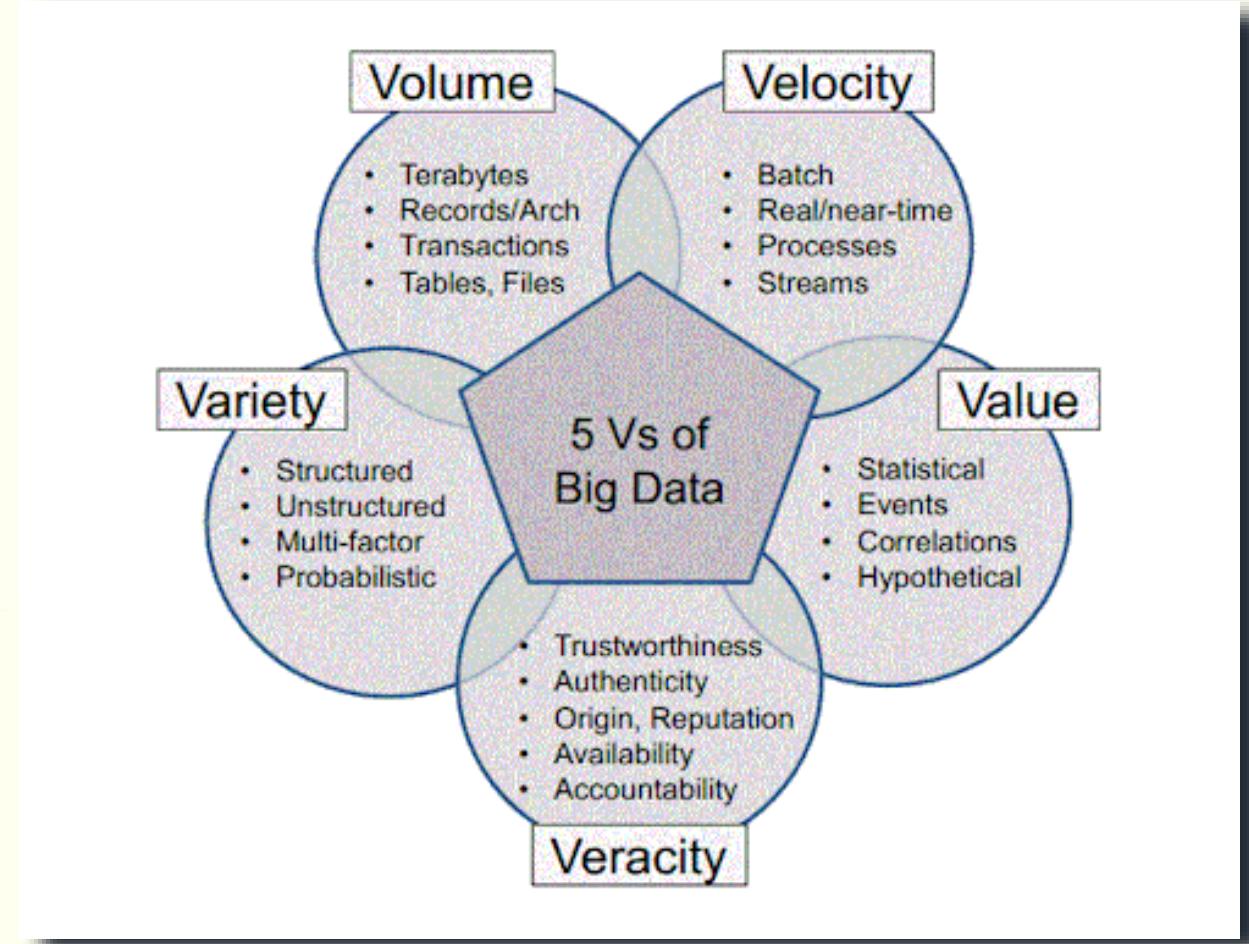
U

D



کلان داده (Big Data)

کلان داده، برای مجموعه داده های بزرگی بکار می رود که به واسطه ویژگی های زیر نمی توان با استفاده از سیستم مدیریت داده های سنتی آنها را پردازش کرد:



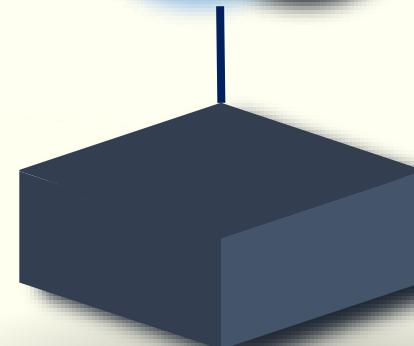
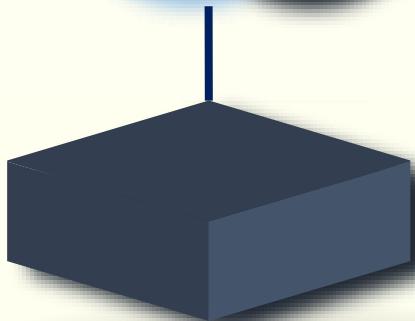
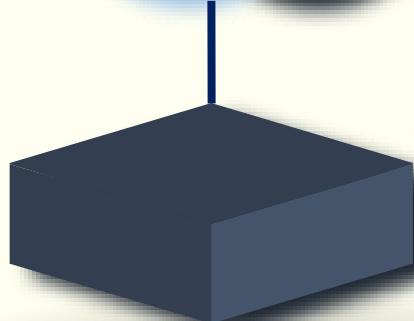
- (Volume) ۱ - حجم
- (Velocity) ۲ - سرعت
- (Variety) ۳ - تنوع
- (Value) ۴ - ارزش
- (Veracity) ۵ - صحت

داده‌کاوی (Data Mining)

کاربردهای
داده‌کاوی

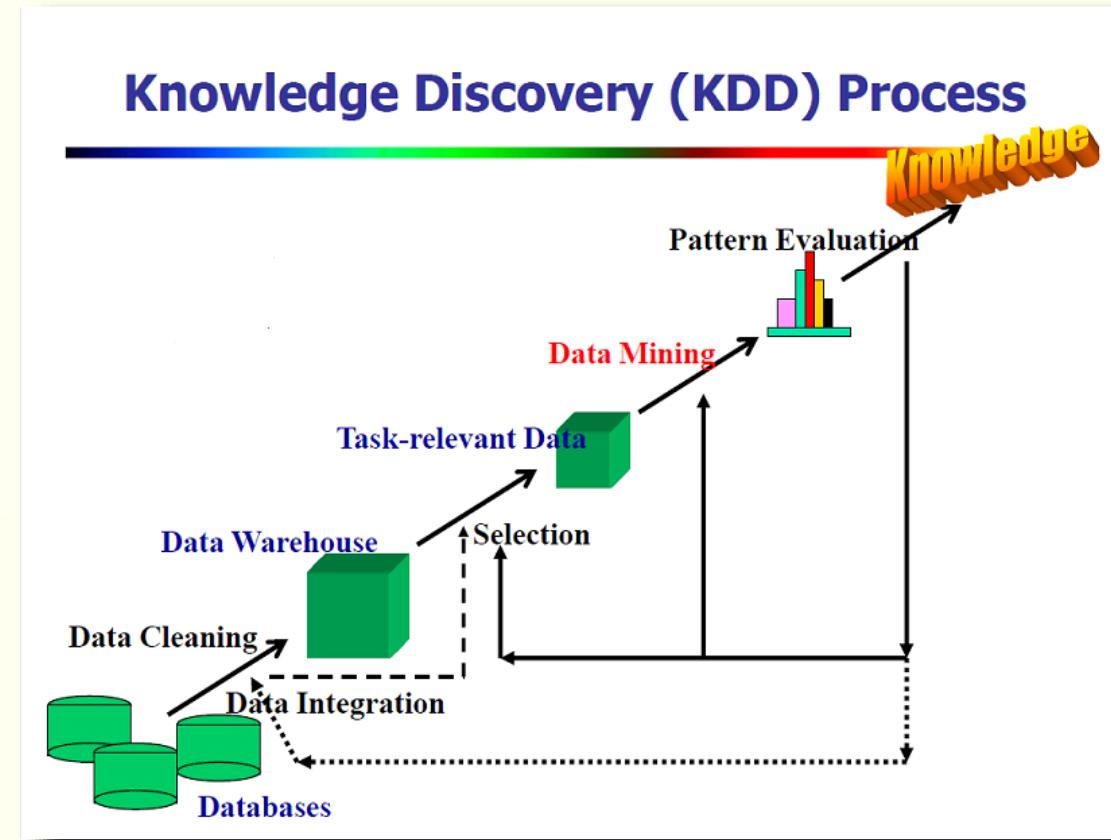
تکنیک‌های
داده‌کاوی

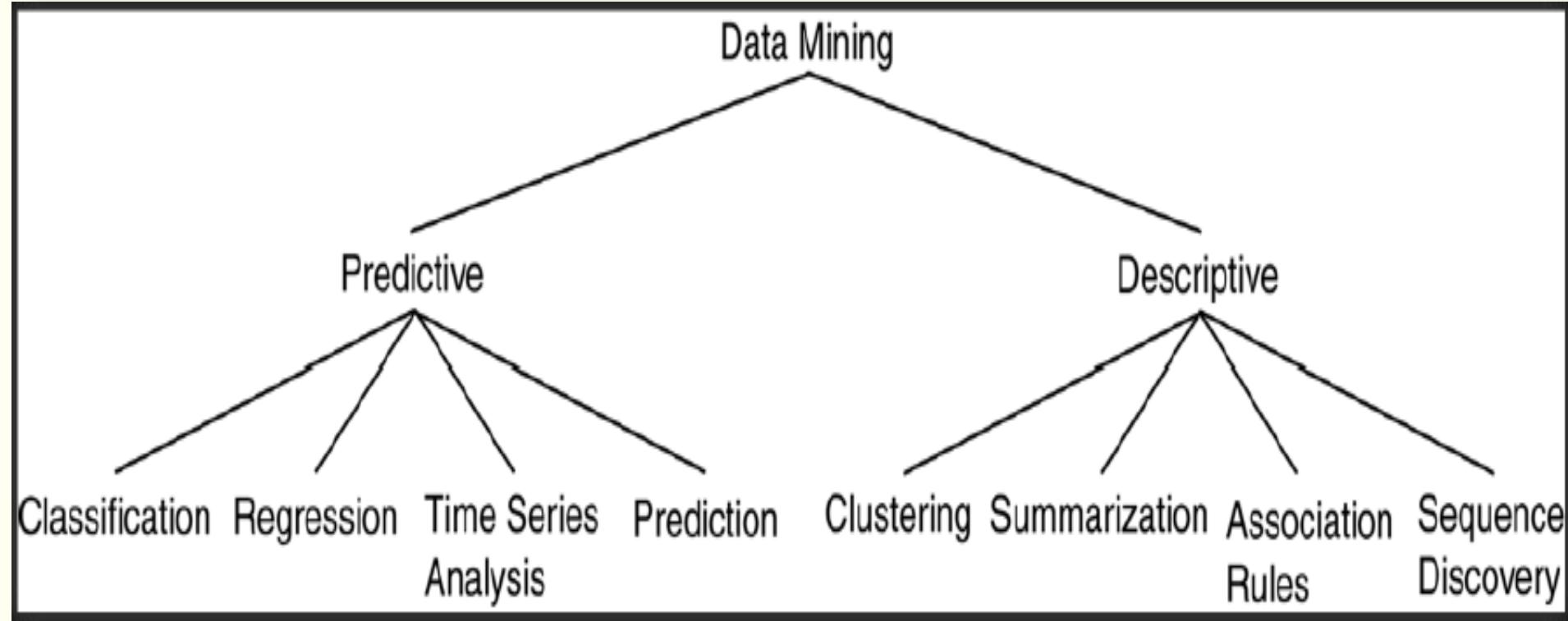
مراحل
داده‌کاوی



داده کاوی به معنای کشف دانش و استخراج الگوهای روابط پنهان موجود در داده ها می باشد.

مراحل داده کاوی مطابق شکل زیر است :





آموزش

پزشکی

بانکداری

کشاورزی

تحلیل سبد خرید

تشخیص تقلب

حمل و نقل

تحلیل شبکه‌های اجتماعی



نمونه‌هایی از پایگاه‌های داده‌ی رابطه‌ای

SQL Server
MySQL
Oracle
Access
,

ویژگی‌های پایگاه‌داده‌ی رابطه‌ای

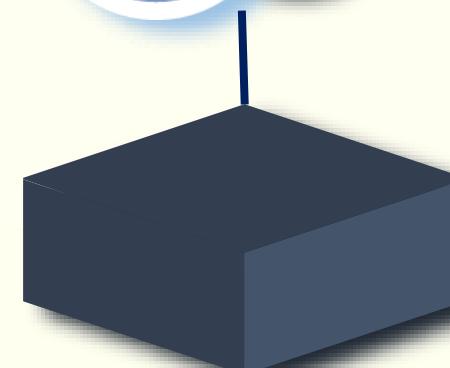
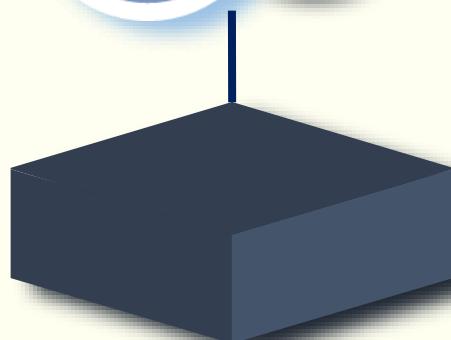
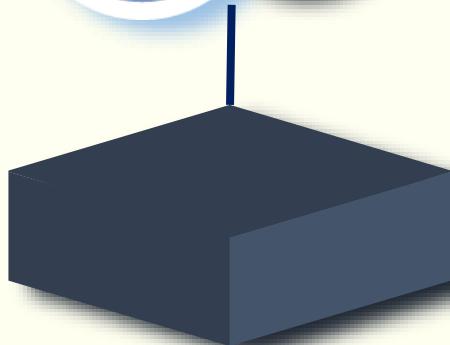
پشتیبانی کامل از ACID
پشتیبانی کامل از CRUD
مقیاس‌پذیری از نوع عمودی
Schema On Write
قرار داشتن در دسته CA
پشتیبانی از داده‌های ساخت‌یافته
پشتیبانی از زبان پرس‌وجوی قدرتمند SQL



أنواع
پایگاه‌های داده‌ی
غیرابطه‌ای

مزایا
و
معایب

معرفی
پایگاهداده‌ی
غیرابطه‌ای



معرفی پایگاهداده‌ی غیررابطه‌ای

پایگاهداده‌ی غیررابطه‌ای شامل مجموعه‌های گسترده از تکنولوژی‌های مختلف پایگاهداده است که در پاسخ به افزایش بسیار وسیع حجم داده‌های ذخیره شده و همچنین تناوب دسترسی به داده‌ها و پیچیدگی پردازش‌ها، توسعه داده شده است. در این نوع پایگاهداده عموماً داده‌ها نیمه ساخت‌یافته و یا حتی بدون ساختار می‌باشند. بنابراین داده‌ها الزاماً در قالب جدول ذخیره نمی‌شوند و روابط بین آن‌ها نسبت به آنچه در مدل رابطه‌ای وجود دارد، متفاوت است.

به معنای No SQL می‌باشد و مفهوم آن کنار گذاری کامل پایگاه داده رابطه‌ای نیست. در واقع هدف پایگاه داده غیر رابطه‌ای کمک به پایگاه داده رابطه‌ای در برابر داده‌های حجمی و پردازش‌های سنگین و همچنین کار با داده‌های نیمه ساخت‌یافته و حتی بدون ساختار، می‌باشد.



پایگاه داده رابطه ای	معیار مقایسه	پایگاه داده غیر رابطه ای
Scaled Up	Scaling	Scaled Out
CA	CAP	AP - CP
Yes	ACID	No
Low	Flexibility	High
Relational Algebra	Functionality	Variable

مزایای پایگاهداده‌ی غیر رابطه‌ای

مقیاس پذیری از نوع افقی
قابلیت دسترسی

پشتیبانی از انواع داده‌های متنوع
ارائه چندین مدل داده‌ای متنوع
انعطاف پذیری بالا
قابلیت مدیریت کلان داده‌ها

معایب پایگاهداده‌ی غیر رابطه‌ای

عدم پشتیبانی از ACID در اکثر نمونه‌ها
عدم پشتیبانی از CRUD در برخی نمونه‌ها
عدم وجود زبان پرس‌وجوی استاندارد
عدم رشد و بلوغ کافی



انواع پایکاههای دادهی غیر رابطه‌ای

۴

مبتنی بر گراف
(Graph-based)

Neo4j
HyperGraphDB
InfoGrid
AllegroGraph

۳

سندگرا
(Document-oriented)

MongoDB
CouchDB
SimpleDB
Riak
OrientDB

۲

ستون گرا
(Column-oriented)

Cassandra
BigTable
HyperTable
Hbase
Vertica

۱

کلید-مقدار
(Key-value)

Dynamo
Voldemort
Redis
BerkeleyDB
MemcacheDB

شرکت‌های استفاده کننده از NoSQL



دسته اول : کلید - مقدار (Key - Value)

پایگاهدادهی کلید-مقدار همانند پایگاهدادهی رابطه‌ای شامل جداولی از اطلاعات است، با این تفاوت که هر سطر جدول حاوی یک دیکشنری یا آرایه‌ای از اطلاعات است. سطرها می‌توانند از نظر شما کاملاً متفاوت از یکدیگر باشند.

این پایگاه داده ساده‌ترین نوع پایگاه داده غیررابطه‌ای است.

کاربرد : مدیریت سبد خرید در آمازوناز طریق DynamoDB انجام می‌شود که از این نوع است.

Key	Value (Opaque)
User:2:friends	{23, 76, 233, 11}
User:2:settings	Theme: dark, cookies: false
User:3:friends	[234, 3466, 86, 55]



دسته دوم : ستون گرا (Column Oriented)

این نوع پایگاه‌های داده همانند کلید-مقدار هستند با این تفاوت که به جای یک جفت کلید-مقدار، برای هر رکورد می‌توان چندین جفت کلید-مقدار داشت. در این نوع، نیازی به ساختار نداریم و هر رکورد می‌تواند چندین ستون با تعداد صفات متفاوت داشته باشد. معروفترین پایگاه داده‌ی ستون گرا می‌باشد Cassandra کاربرد : فیسبوک برای تحلیل داده‌های موجود در شبکه اجتماعی خود از Cassandra استفاده می‌کند.

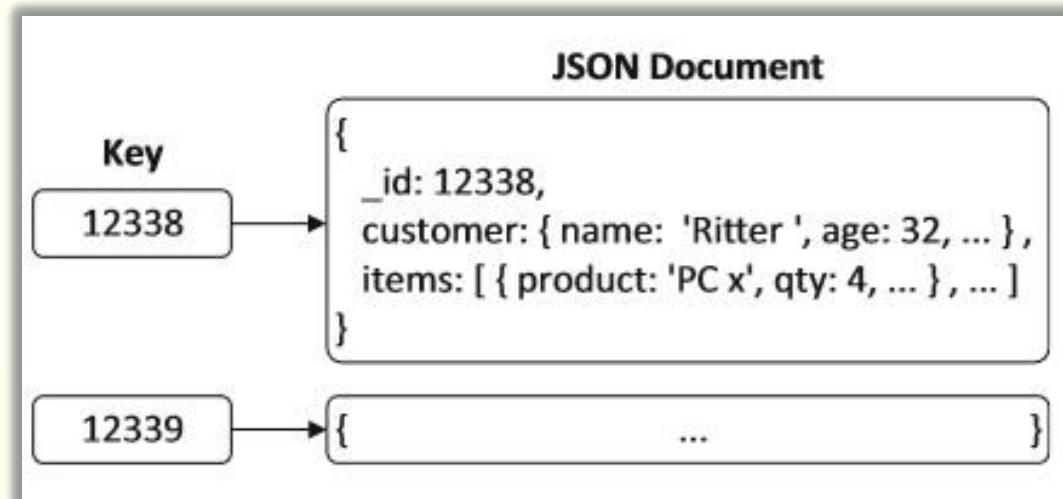


دسته سوم : سندگرا (Document Oriented)

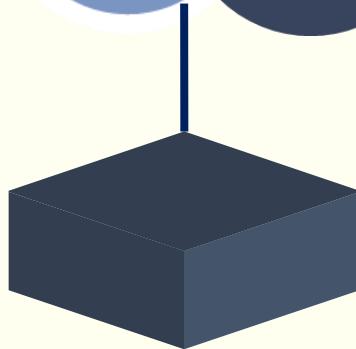
در این نوع پایگاهداده می‌توان داده‌های پیچیده‌تری را نسبت به کلید-مقدار ذخیره کرد. هر عضو از داده‌ها را سند، و مجموعه‌ای از اسناد را مجموعه می‌گویند. سندها معادل رکوردها و مجموعه‌ها معادل جداول رابطه‌ای هستند، با این تفاوت که در اینجا، یک مجموعه ممکن است دارای فیلدی مختلفی باشد.

ساختار سندها عموماً بر مبنای JSON می‌باشد.

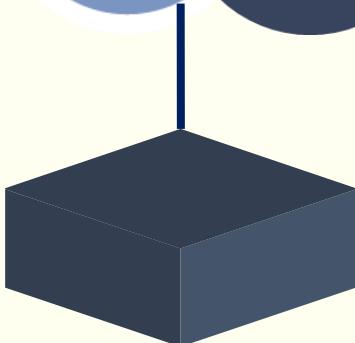
معروف‌ترین پایگاه داده‌ی سندگرا MongoDB می‌باشد.



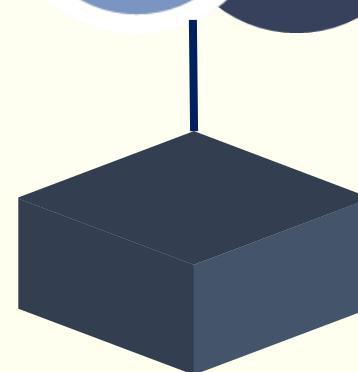
کاربرد
و
مزایا



مقایسه
با
پایگاه داده رابطه ای



تعریف گراف
و
پایگاه داده گراف



تعریف گراف و پایگاه داده گراف

گراف (Graph)

گراف ساختاری متشکل از دو عنصر می باشد :

- گره (node) : هر گره بیانگر یک موجودیت است.
 - یال (Edge) : هر یال بیانگر ارتباط بین موجودیتها می باشد.
- ساختار همه منظوره گراف این اجازه را به ما می دهد تا هر نوع سناریویی را مدل سازی کنیم.

پایگاه داده گراف (Graph Database) :

برای استفاده سودمند از روابط بین داده ها، سازمانها به تکنولوژی پایگاه داده ای نیاز دارند که اطلاعات روابط بین داده ها را به عنوان یک موجودیت ذخیره کند. چنین تکنولوژی، پایگاه داده گراف نام دارد.

پایگاه داده گراف، یک سیستم مدیریت پایگاه داده آنلاین است که عملیات CRUD را روی مدل داده ای گراف اجرا می کند.



تفاوت مدل گراف با مدل رابطه ای :

- ۱ - در مدل گراف داده Null وجود ندارد.
- ۲ - سطر های فاقد مقدار نشان داده نمی شوند.
- ۳ - ارتباطات را با جزئیات بیشتری نشان می دهد.
- ۴ - اغلب نرمال شده است.
- ۵ - مدل داده ای ساده تری دارد.

نحوه تبدیل مدل رابطه ای به مدل گراف :

- ۱ - هر سطر یا رکورد در مدل رابطه ای به یک گره یا node در گراف تبدیل می شود.
- ۲ - نام جدول به عنوان نام برچسب نشان داده می شود.
- ۳ - ستونها و فیلد های جدول به عنوان خصوصیات (Properties) گره در نظر گرفته می شوند.
- ۴ - ارتباطات جداول به ارتباطات گره ها تبدیل می شود.



موارد کاربرد:

- ۱ - تشخيص کلاه برداری
- ۲ - سامانه مدیریت مشتری
- ۳ - شبکه های اجتماعی
- ۴ - مانیتورینگ های پیچیده و پر روابط

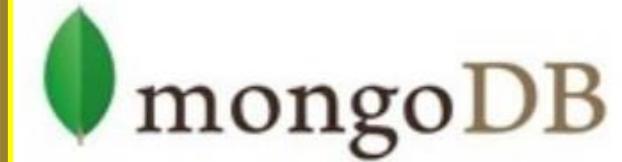
مزايا:

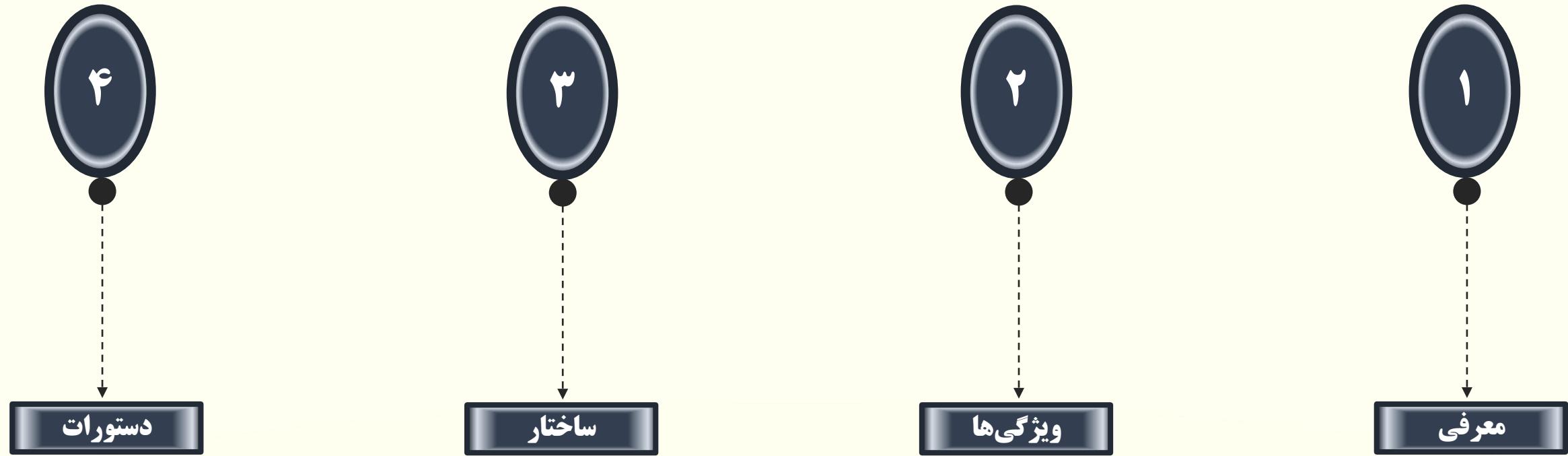
- ۱ - کارایی
- ۲ - انعطاف پذیری
- ۳ - نصب، راه اندازی، تست، اجرا و پیکربندی راحت
- ۴ - پشتیبانی از ACID
- ۵ - فاقد شما

نمونه‌هایی از پایکاههای داده



Amazon SimpleDB





MongoDB برخلاف پایگاه‌های داده‌ی رابطه‌ای که بر پایه‌ی جدول پیاده سازی می‌شوند، مبتنی بر سند (document-oriented) پیاده سازی شده و با استفاده از یک قالب پویا نظیر JSON به یکپارچه سازی داده‌ها می‌پردازد.

MongoDB Highlights :

Developers : MongoDB inc.

Initial Release : 2007

Written in : C, C++, Java Script

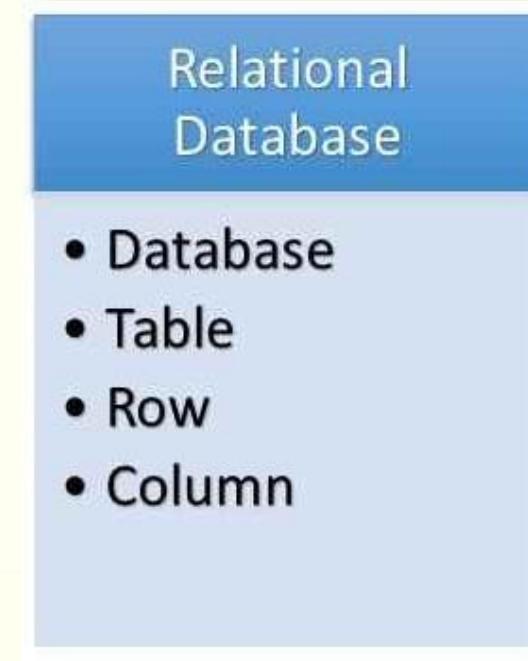
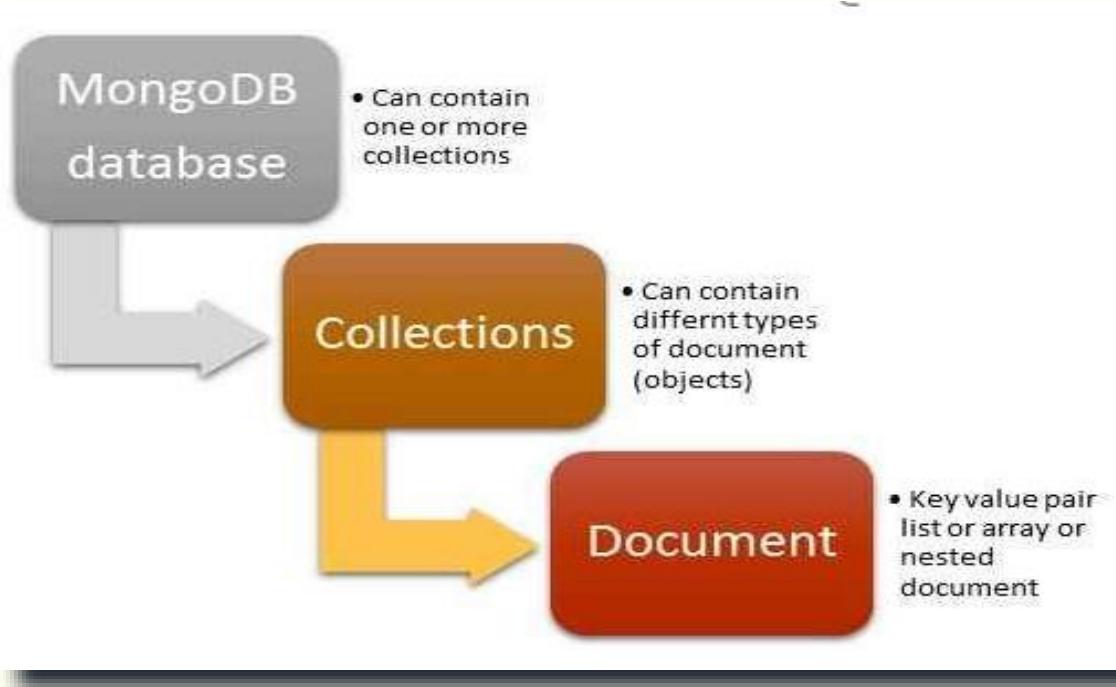
Operation System : Cross Platform

Type : Document-Oriented Database

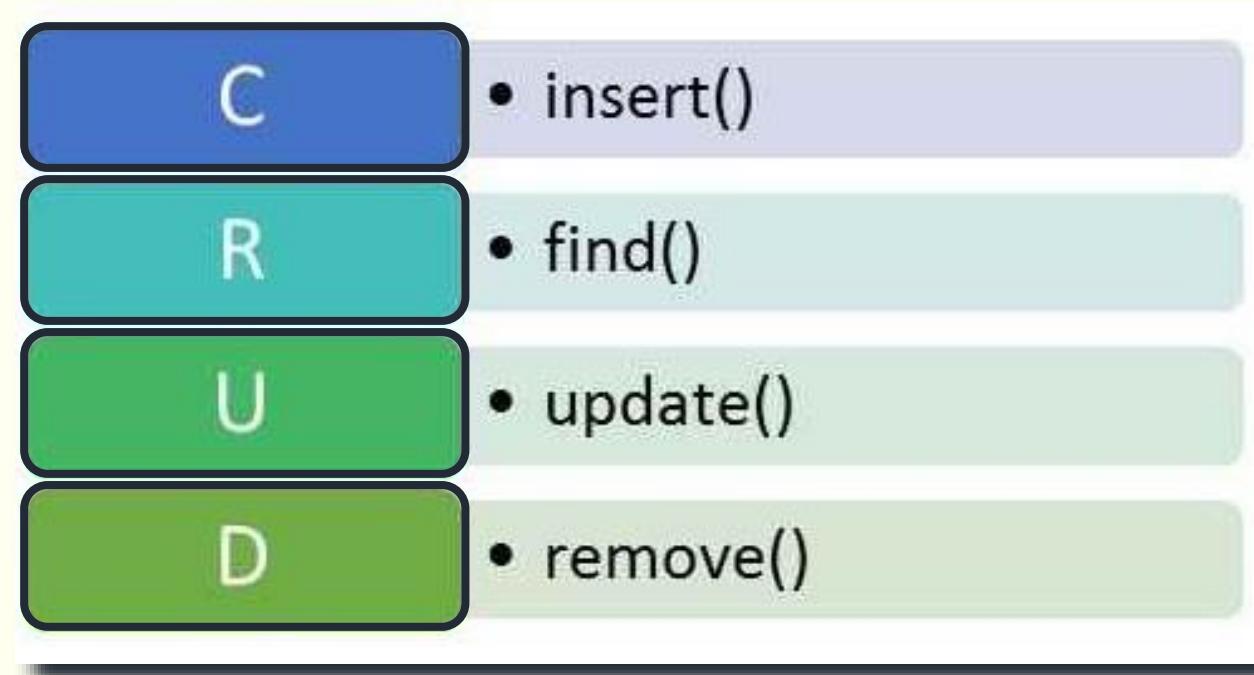
پایپکاه داده MongoDB – ویرگی‌ها

- ۱ - فاقد شمای خاصی می‌باشد.
- ۲ - بر روی سیستم‌های عامل ویندوز، مکینتاش، و سولاریس قابل اجرا می‌باشد.
- ۳ - پشتیبانی از Scalling out
- ۴ - سیستم را در برابر خاموش شدن ناگهانی حمایت می‌کند.
- ۵ - تکنیک master-slave replication (گره اصلی : خواندن و نوشتن - گره فرعی : کپی از گره اصلی)
- ۶ - نیازی به ذخیره کردن مقادیر Null ندارد.
- ۷ - Open source و رایگان است.
- ۸ - از تراکنش‌ها پشتیبانی نمی‌کند.
- ۹ - از join پشتیبانی نمی‌کند.
- ۱۰ - در برخی موارد، انعطاف‌پذیری در پرس‌وجوها کم است.





برای عملیات CRUD دستورات زیر را دارد :



برای درج رکورد از دستور insert استفاده می‌شود:

```
1 doc = {  
2   name: 'xyx',  
3   class: '12th',  
4   subjects: ['physics', 'chemistry', 'maths', 'english', 'computer'],  
5   address: {  
6     house_no: '123',  
7     sector: '50',  
8     city: 'noida'  
9   }  
10 }  
11 db.mycol.insert(doc);
```

از دستور **Find** برای جستجوی داده‌ها استفاده می‌شود. این دستور معادل دستور **Select** می‌باشد و فرمت آن به صورت زیر است:

```
1 | db.marks.find({name: 'student0'})
```

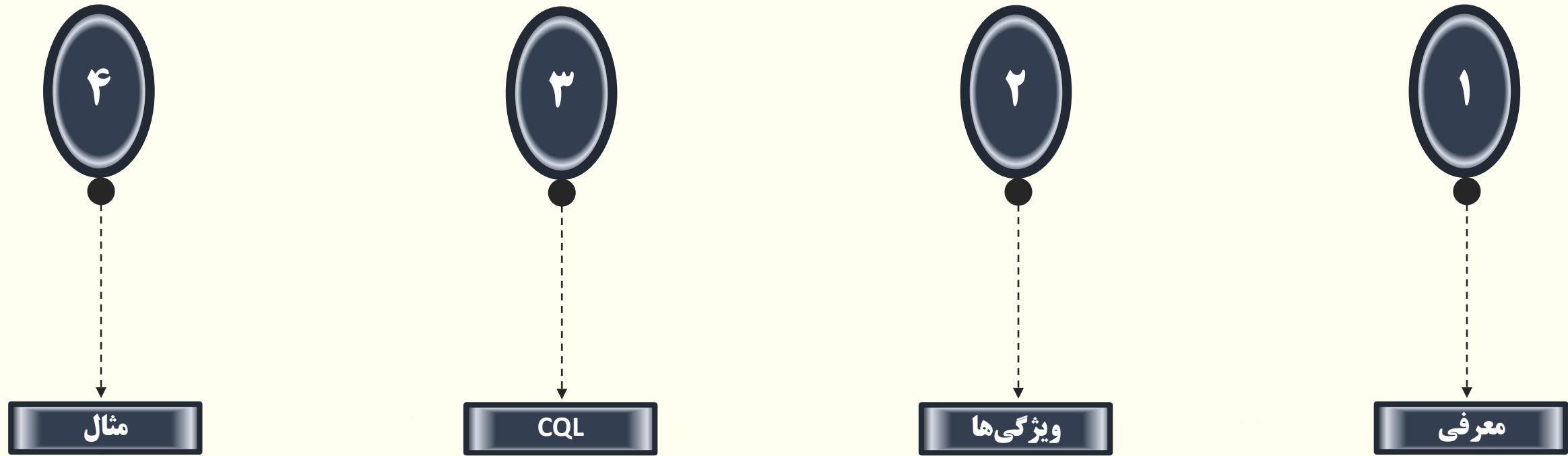
پایگاه داده MongoDB – دستور Update

برای اعمال تغییرات در داده‌ها از دستور **Update** استفاده می‌شود. فرمت این دستور به صورت زیر است :

```
1 | db.additionalsubject.update({name:'student1'}, {subject:['craft']})
```

برای حذف داده‌ها از دستور Remove استفاده می‌شود. فرمت این دستور به صورت زیر می‌باشد:

```
1 | db.additionalsubject.remove({name:'student3'})
```



Neo4j شناخته شده ترین پایگاه داده مبتنی بر گراف می باشد.

Neo4j Highlights :

Developers : Neo Technology

Initial Release : 2007

Written in : java

Operation System : Cross Platform

Type : Graph Database

Dual License : Open Source and Commercial

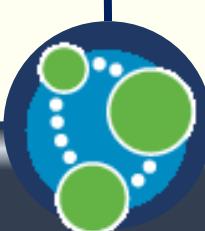
پایگاه داده Neo4j - ویژگی‌ها

- ۱ - اولین و بهترین پایگاه داده مبتنی بر گراف است.
- ۲ - بزرگ‌ترین جامعه گرافی دنیا با بیشترین جمیعت علاقمند به پایگاه داده مبتنی بر گراف می‌باشد:
 - بیش از یک میلیون دانلود
 - بیش از ۲۰۰۰۰ مطلب آموزشی
 - بیش از ۲۰۰۰۰ عضو بازدید کننده
 - بیش از ۵۰۰ رویداد در سال
 - بیش از ۱۰۰ تکنولوژی و سرویس
- ۳ - هنگام عملیات خواندن و نوشتن، بدون از دست دادن یکپارچگی داده‌ها، عملکرد سریع و کارآمدی ارائه می‌دهد.
- ۴ - از عملیات ACID پشتیبانی می‌کند.
- ۵ - بواسطه پردازش و ذخیره گراف بصورت محلی، کارایی بالایی دارد.
- ۶ - یادگیری و استفاده از آن بسیار آسان است.
- ۷ - مدل داده‌ای کاربر پسندی دارد.
- ۸ - با استفاده از زبان CQL از عملیات CRUD به خوبی پشتیبانی می‌کند.
- ۹ - برای پروژه‌های Start up و Enterprise بسیار مناسب است.

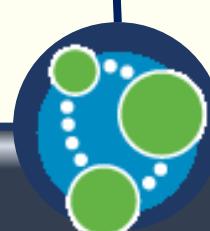




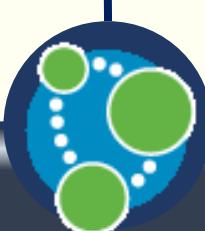
(Cypher Query Language) CQL زبان



دستورات
زبان
CQL



أنواع داده ها
در زبان
CQL



ویژگی های
زبان
CQL



- ۱. یک زبان پرس و جو برای پایگاه داده Neo4j می باشد.
- ۲. ساختار دستوری و گرامری شبیه به SQL دارد.
- ۳. گرامر و قواعد دستوری ساده و قابل فهمی دارد.
- ۴. دستورات متعددی برای انجام عملیات پایگاه داده دارد.
- ۵. با استفاده از Where و Order By می تواند Query های سنگین را به سادگی انجام دهد.
- ۶. از توابع Relationship ، String و همچنین توابع مربوط به Aggregation بهره می برد.



پایکاوه داده Neo4j – انواع دادهها در زبان CQL

نوع داده	محتوی	اندازه	نوع داده	محتوی	اندازه
boolean	داده منطقی	۸ بیت	float	عدد اعشاری	۳۲ بیت
byte	عدد صحیح	۸ بیت	double	عدد اعشاری	۶۴ بیت
short	عدد صحیح	۱۶ بیت	char	کاراکتر	۱۶ بیت
int	عدد صحیح	۳۲ بیت	string	رشته	-----
long	عدد صحیح	۶۴ بیت			



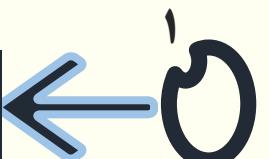
Syntax Sample

Delete



Syntax Sample

Create



Syntax Sample

Remove



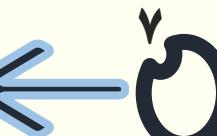
Syntax Sample

Match



Syntax Sample

Order By



Syntax Sample

Return



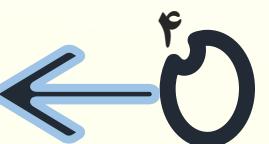
Syntax Sample

Set



Syntax Sample

Where



دستور Create برای موارد زیر بکار می رود :

Create

(
 <node-name> : <label-name>

{

 <property1-name> : <property1-value> ,
 <property2-name> : <property2-value> ,

.

.

.

 <property n-name> : <property n-value>

}

)

۱ - ایجاد گره بدون خاصیت

۲ - ایجاد گره با خاصیت های آن

۳ - ایجاد ارتباط بین گره ها بدون خاصیت آنها

۴ - ایجاد ارتباط بین گره ها با خاصیت آنها

۵ - ایجاد یک یا چند برچسب برای یک گره یا ارتباط

پایکاوه داده Neo4j - مثال ۱ دستور Create

```
$ create(p1:Contact  
{FirstName:'Reza',LastName:'Bagheri',Mobile:'09190023119',City:'Karaj',Category:'Family'})
```

The screenshot shows the Neo4j browser interface with three main panels: Graph, Rows, and Text.

- Graph Panel:** Shows three green circular nodes representing the created contact entities. One node is labeled "Moradi", another is "Bagheri", and the third is "Ahmadi".
- Rows Panel:** Displays the results of the query execution:
 - Row 1: {Category: Family, FirstName: Ali, City: Tehran, LastName: Ahmadi, Mobile: 09123239939}
 - Row 2: {Category: Family, FirstName: Reza, City: Karaj, LastName: Bagheri, Mobile: 09190023119}
 - Row 3: {Category: Friend, FirstName: Zahra, City: Tehran, LastName: Moradi, Mobile: 09371323219}
 - Row 4: {Category: Family, FirstName: Reza, City: Karaj, LastName: Bagheri, Mobile: 09190023119}
- Text Panel:** Shows the Cypher code used to create the nodes:

```
create(p1:Contact  
{FirstName:'Reza',LastName:'Bagheri',Mobile:'09190023119',City:'Karaj',Category:'Family'})
```
- Code Panel:** Shows the generated Cypher code:

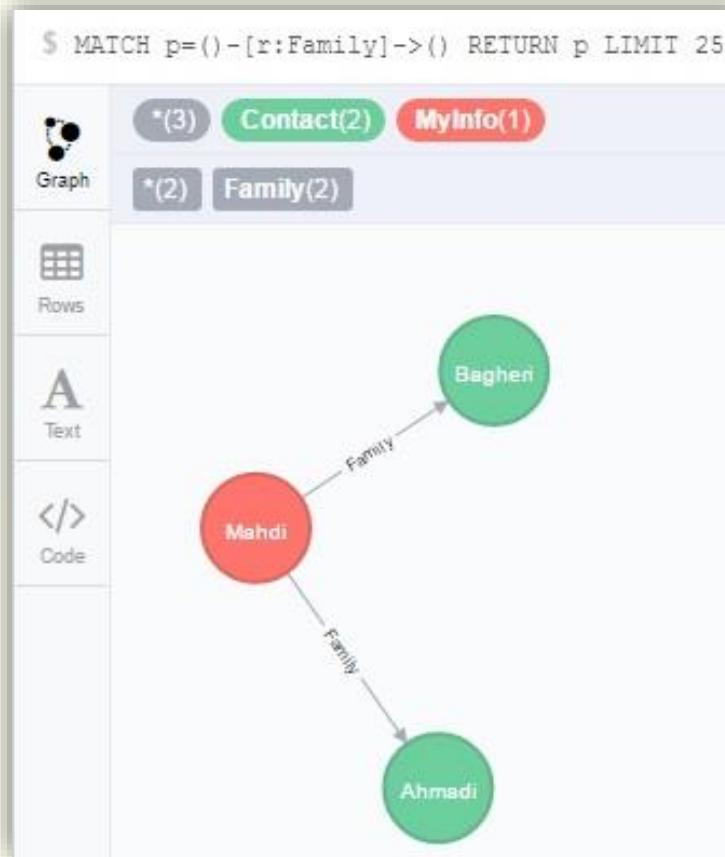
```
graph TD; n1[{"Category": "Family", "FirstName": "Ali", "City": "Tehran", "LastName": "Ahmadi", "Mobile": "09123239939"}]; n2[{"Category": "Family", "FirstName": "Reza", "City": "Karaj", "LastName": "Bagheri", "Mobile": "09190023119"}]; n3[{"Category": "Friend", "FirstName": "Zahra", "City": "Tehran", "LastName": "Moradi", "Mobile": "09371323219"}]; n4[{"Category": "Family", "FirstName": "Reza", "City": "Karaj", "LastName": "Bagheri", "Mobile": "09190023119"}]
```

مثال ۲



پایکاہ داده - مثال ۲ دستور Create - Neo4j

```
1 MATCH (a:Category { Name: 'Family' }), (b:Contact { Category: 'Family' })
2 CREATE (a)-[:Contain] -> (b)
```



Match

```
(  
    <node-name> : <label-name>  
    {  
        <property1-name> : <property1-value>,  
        <property2-name> : <property2-value>,  
        .  
        .  
        .  
        <property n-name> : <property n-value>  
    }  
)
```

دستور Match برای موارد زیر بکار می رود :

- ۱ - بازیابی اطلاعات مربوط به گره ها و خصوصیات آنها از پایگاه داده
- ۲ - بازیابی اطلاعات مربوط به گره ها، ارتباطات و خصوصیات آنها

پایکاه داده – مثال دستور Match – Neo4j

مثال زیر اشخاصی که ساکن تهران هستند را می یابد :

```
$ match (n:Contact{City:'Tehran'})
```

دستور Return برای موارد زیر بکار می رود :

Return

```
<node-name> : <property1-name>,  
.  
. .  
<node-name> : <propertyn-name>
```

۱ - بازیابی برخی از خصوصیات یک گره

۲ - بازیابی تمام خصوصیات یک گره

۳ - بازیابی برخی از خصوصیات گره ها و ارتباط بین آنها

۴ - بازیابی تمام خصوصیات گره ها و ارتباط بین آنها

در واقع Return خروجی حاصل از Match را باز می گرداند.

پایگاه داده Neo4j - مثال دستور Return

```
$ match (n:Contact{City:'Tehran'}) return n
```

```
$ match (n:Contact{City:'Tehran'}) return n
```

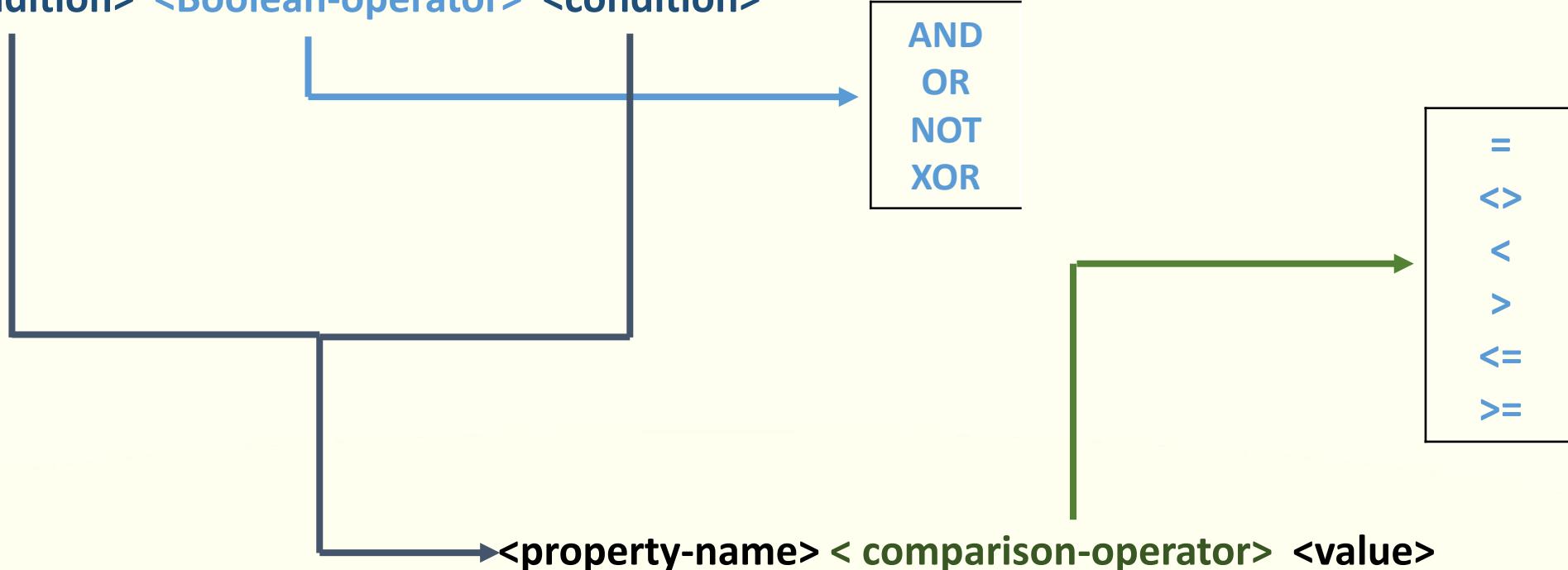
n
Category Family FirstName Ali City Tehran LastName Ahmadi Mobile 09123239939
Category Friend FirstName Zahra City Tehran LastName Moradi Mobile 09371323219

Returned 2 rows in 87 ms.



دستور Where به منظور فیلتر کردن و اعمال شرط بکار می رود :

Where <condition> <Boolean-operator> <condition>



دستور Delete به دو صورت بکار می رود :

۱ - حذف یک گره

Delete <node-name-list>

۲ - حذف یک گره و گره های مرتبط با آن و ارتباطات بین آنها

Delete <node1-name> , <node2-name> , <relationship-name>

مثال ۱ : حذف گره C با برچسب Contact

```
$ MATCH ( c: Contact ) DELETE c
```

مثال ۲ : حذف گره های C ، P و ارتباط بین آنها

```
$ MATCH ( c: Category )-[ Contain ]-( p: Contact )  
DELETE c, p, Contain
```

دستور Remove به دو صورت بکار می رود :

۱ - حذف برچسب های یک گره یا یک ارتباط

Remove <label-name-list>

۲ - حذف خصوصیات یک گره یا یک ارتباط

Remove <property-name>

مثال ۱ : حذف برچسب Category

```
$ MATCH ( c: Category ) Remove c
```

مثال ۲ : حذف خاصیت موبایل از Contact

```
$ MATCH ( c: Contact ) Remove c . mobile Return c
```

از دستور Order By برای مرتب کردن خروجی حاصل از Match استفاده می شود :

Order By <Property-name> [DESC]



مرتب سازی نزولی

مثال زیر نام ، نام خانوادگی و موبایل افراد را بر حسب نام خانوادگی مرتب کرده نمایش می دهد :

\$ MATCH (c: Contact)

Return

c . First Name , c . Last Name , c . Mobile

Order By c . Last Name

از دستور Set برای بروز رسانی و به دو صورت زیر استفاده می شود :

۱ - اضافه کردن خاصیت جدید به یک گره یا ارتباط

Set <node-label-name> . <property-name>

۲ - اضافه کردن یا تغییر مقادیر خاصیت ها

Set <node-label-name> . <property-name> = <value>

مثال زیر نام ، برای شخصی با نام خانوادگی احمدی ، خصیت سن را با مقدار ۳۰ اضافه می کند :

```
$ MATCH ( c: Contact { Last Name : ' Ahmadi ' } )
```

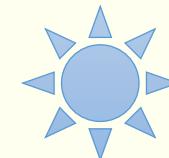
```
Set c . Age = 30
```

پایگاه داده دفتر تلفن (Phone Book) را با سه جدول زیر در نظر بگیرید:

Category	
Code (PK)	Name
100	Family
200	Friend

Contact				
First Name	Last Name	Mobile	Category Code	City Code
Ali	Ahmadi	09123239939	100	1
Reza	Bagheri	09190023119	100	2
Zahra	Moradi	09371323219	200	1

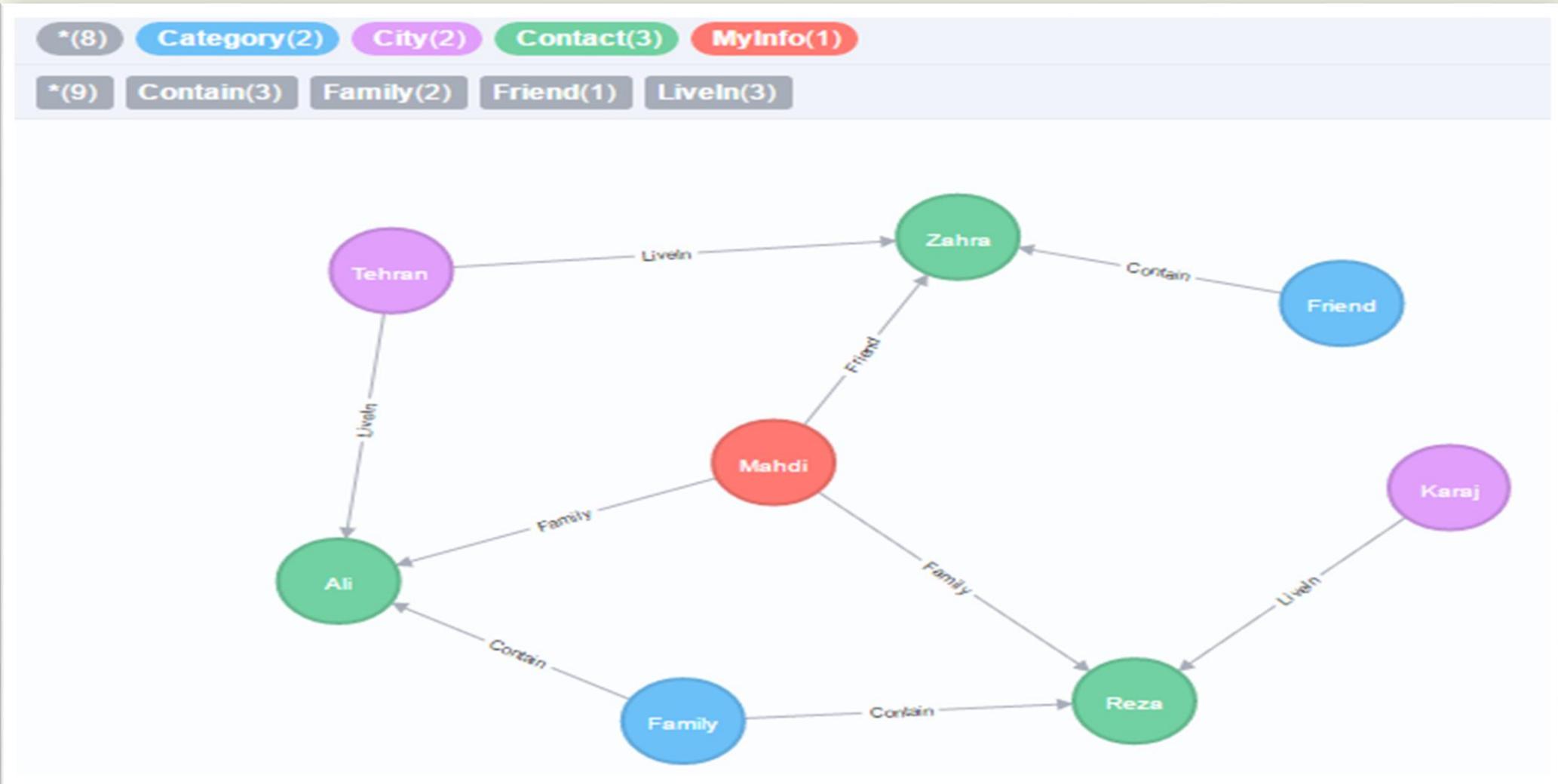
City	
Code (PK)	Name
1	Tehran
2	Karaj

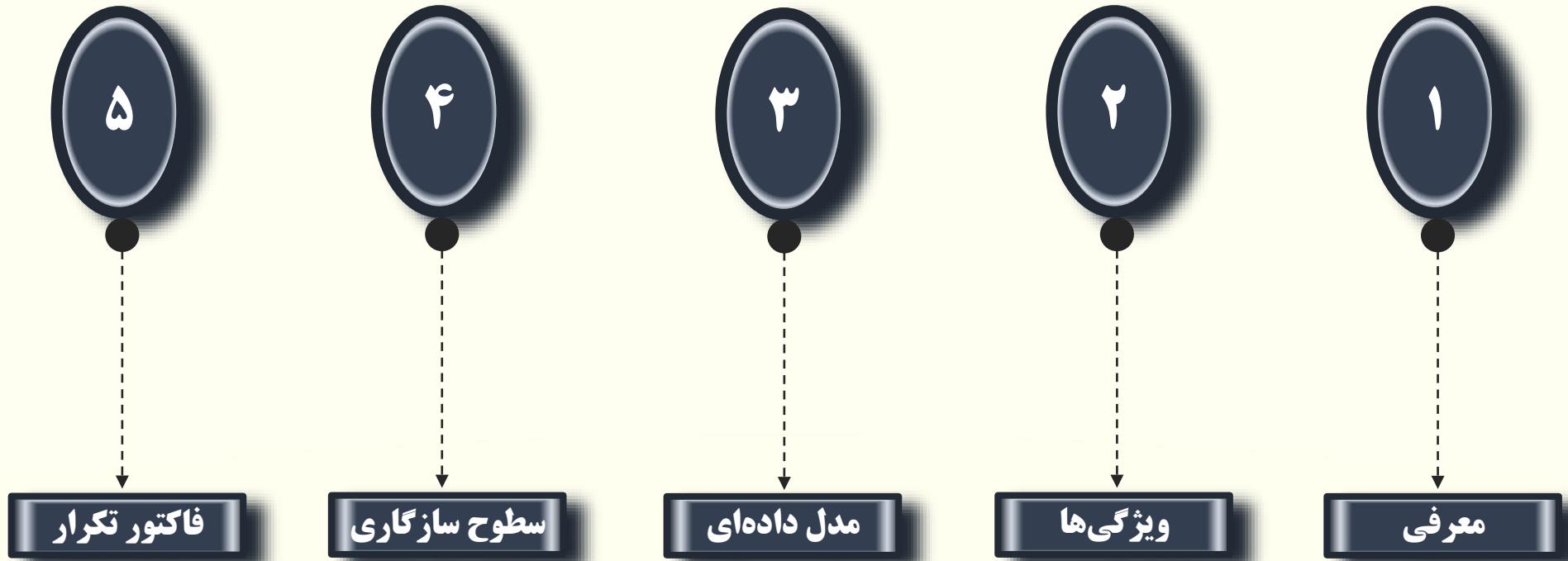


حال می توانید گراف طراحی شده از این پایگاه داده را با کلیک بر روی آیکن مقابل مشاهده فرمایید :



پایکاہ داده میال – Neo4j





در پایگاه‌های داده‌ی مبتنی بر پرس‌وجو نظیر کساندرا، طراحی پایگاه‌داده بر اساس دو معیار انجام می‌شود:

۱- موجودیت‌ها و ارتباطات آن‌ها

۲- پرس‌وجوهای مورد نیاز

بنابراین، داده‌های مرتبط با یک پرس‌وجو در یک محل و در یک سرور قرار می‌گیرند و هنگام بازیابی اطلاعات سرعت بسیار بالا خواهد بود.

ویژگی‌های پایگاه داده Cassandra

مقیاس پذیری از نوع افقی
متن باز بودن
سرعت خواندن و نوشتן سریع
انعطاف پذیری بالا
پشتیبانی از تکرار
قرار داشتن در دسته AP

قابلیت دسترسی بالا
تحمل پذیری خطأ
Schema on Read
قابلیت توزیع پذیری بالا
سازگاری قابل تنظیم
Peer to Peer
معماری



مرحله دوم : شناسایی موجودیت‌ها و ارتباط بین آن‌ها

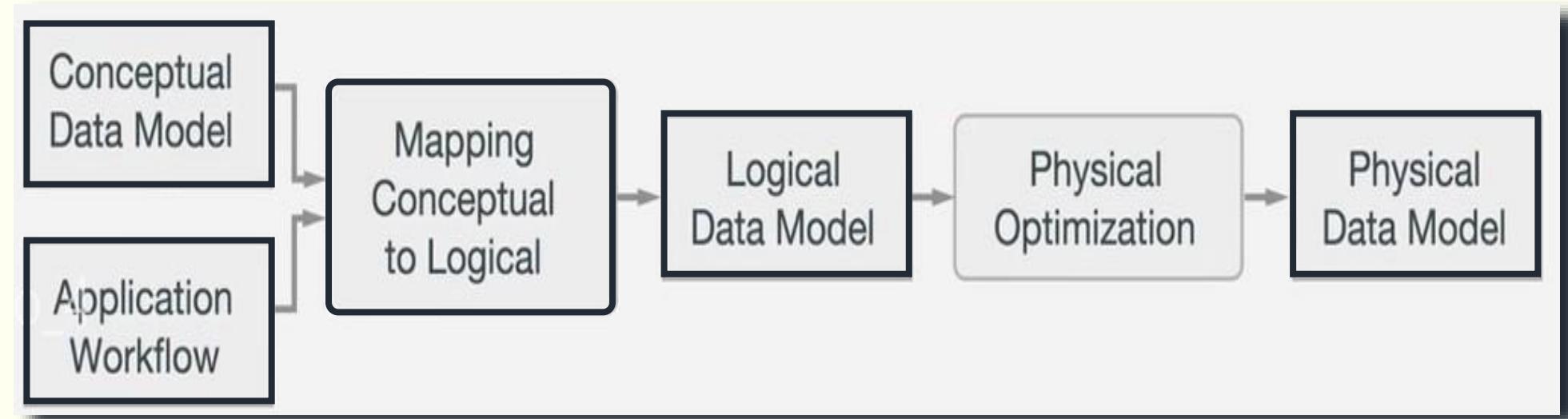
مرحله چهارم : تعیین شما (Schema)

مرحله ششم : پیاده‌سازی با زبان CQL

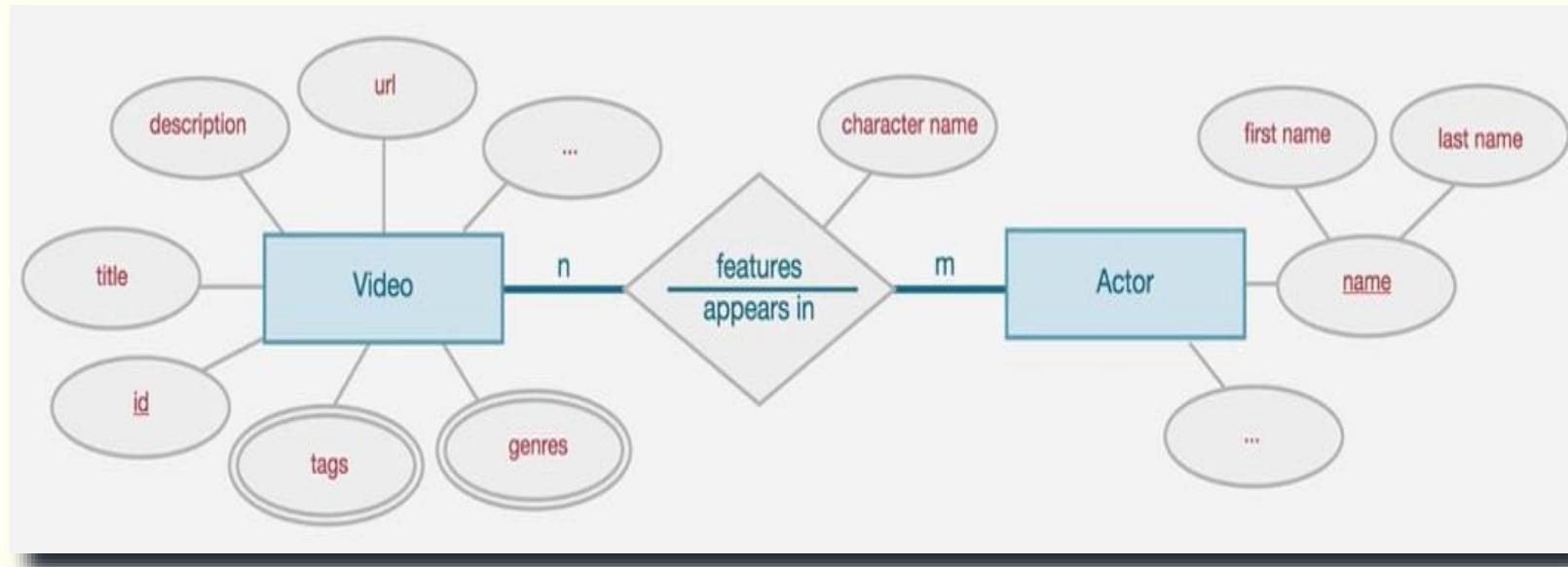
مرحله اول : تحلیل نیازمندی‌ها

مرحله سوم : شناسایی پرس‌و‌جواب‌ها

مرحله پنجم : بهینه‌سازی



پایگاه داده – مدل داده‌ای مفهومی Cassandra



پایکاہ داده – نمودار چریان کاری Cassandra

مثال :

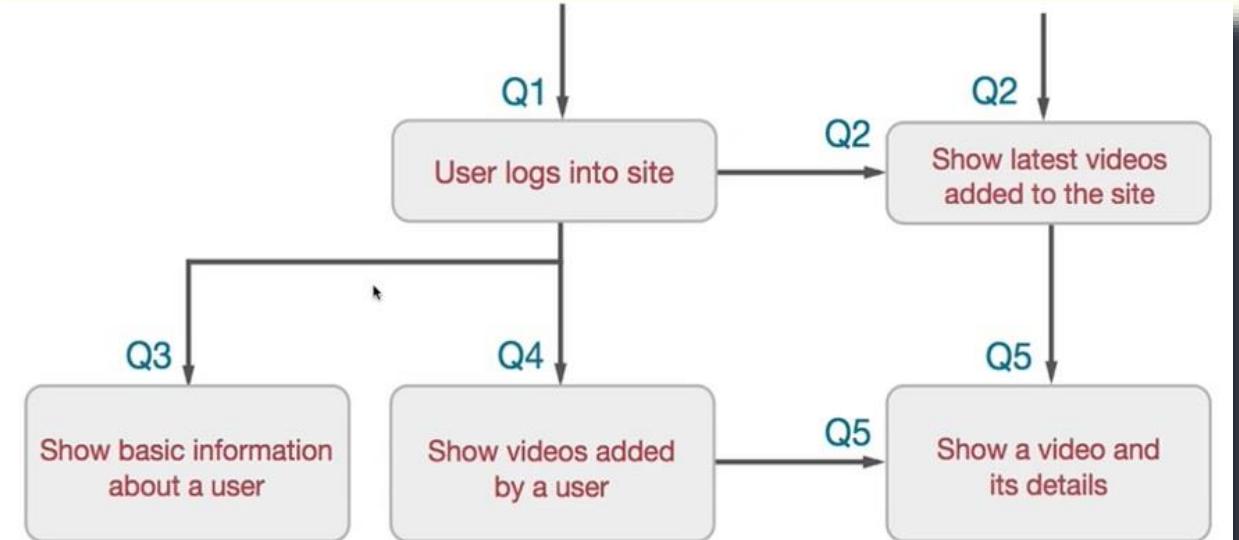
Q1: یافتن یک کاربر با ایمیل خاص.

Q2: یافتن ویدئوهایی که اخیراً در سایت آپلود شده‌اند.

Q3: یافتن یک کاربر با id خاص.

Q4: یافتن ویدئوهایی که توسط یک کاربر با id مشخص آپلود شده‌اند.

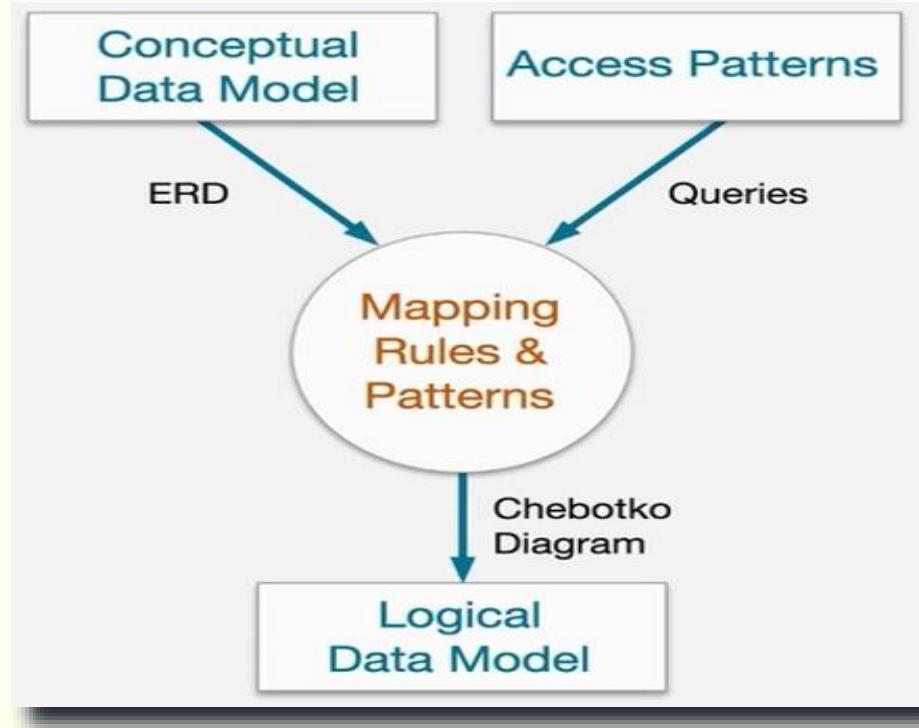
Q5: یافتن یک ویدئو با id خاص.



ACCESS PATTERNS

- Q1: Find a user with a **specified email**
- Q2: Find most recently uploaded **videos**
- Q3: Find a **user with a specified id**
- Q4: Find videos uploaded by a **user with a known id**
- Q5: Find a video with a **specified video id**





قوانين نگاشت مدل داده مفهومی به مدل داده منطقی

قانون اول : موجودیت‌ها و ارتباطات

قانون دوم : جستجوی صفات به عملگر مساوی

قانون سوم : جستجوی صفات با عملگر نامساوی

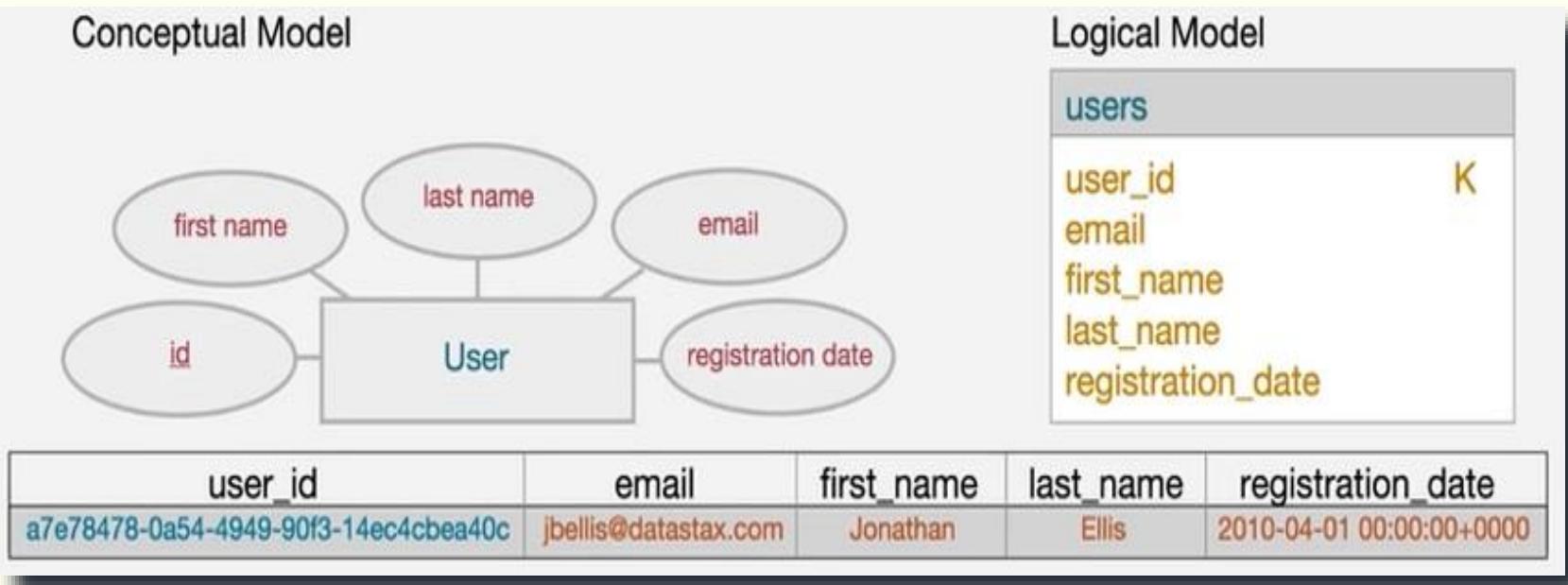
قانون چهارم : مرتب‌سازی

قانون پنجم : صفات کلیدی

مثال جامع بکارگیری قوانین نگاشت

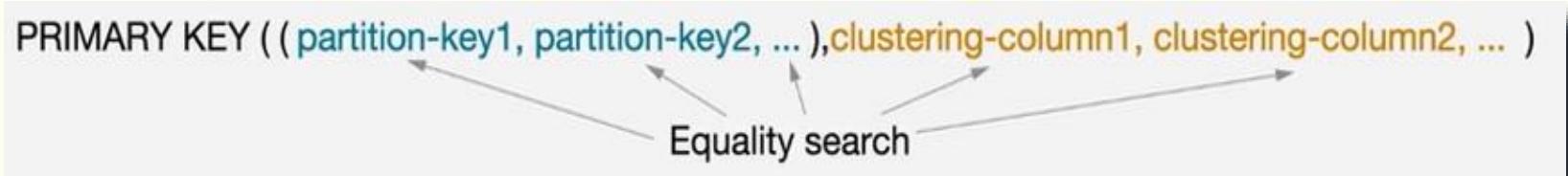
پایکاه داده – قانون اول نگاشت Cassandra

نوعهای موجودیت و رابطه‌ها به جداول و موجودیت‌ها و رابطه‌ها به پارتبیشن‌ها یا سطرها نگاشت می‌شوند.

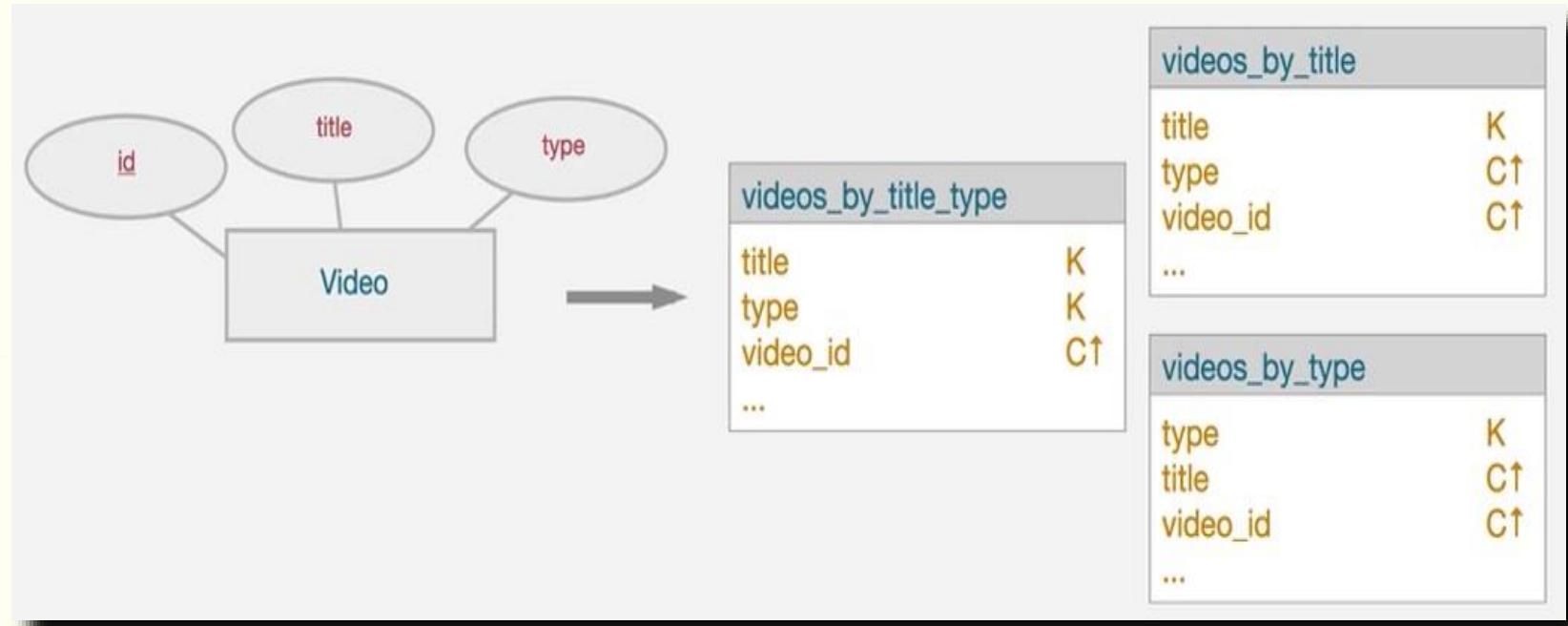


پایکاه داده – قانون دوم نگاشت Cassandra

صفاتی در پرس و جو با عملگر مساوی بکار می‌رond باید به عنوان کلید پارتیشن تعریف شوند.



مثال : نگاشت موجودیت : پرس و جو روی **title** و **type**



پاپکاه داده – قانون سوم نکاشت Cassandra

صفاتی که جستجو روی آن‌ها با عملگرهای نامساوی انجام می‌شود، کلیدهای خوشبندی هستند.



مثال : نگاشت موجودیت : پرس و جو روی ? registration_date > ? و lastname = ?



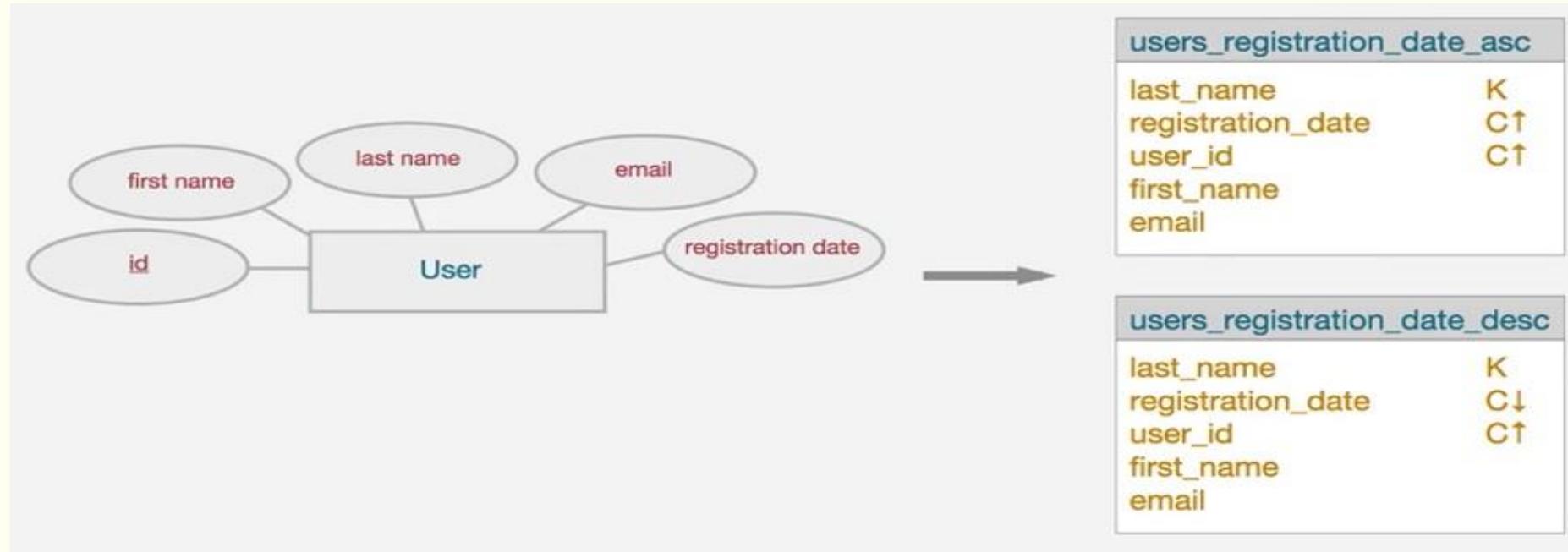
پایکاه داده – قانون چهارم نگاشت Cassandra

صفات مرتبسازی به ستون‌های خوشبندی نگاشت می‌شوند. (در هر پارتیشن داده‌ها بر اساس کلیدهای خوشبندی مرتب می‌شوند.)

PRIMARY KEY ((partition-key1, partition-key2, ...), clustering-column1, clustering-column2, ...)

ORDER BY clustering-column1ASC, clustering-column2ASC, ...

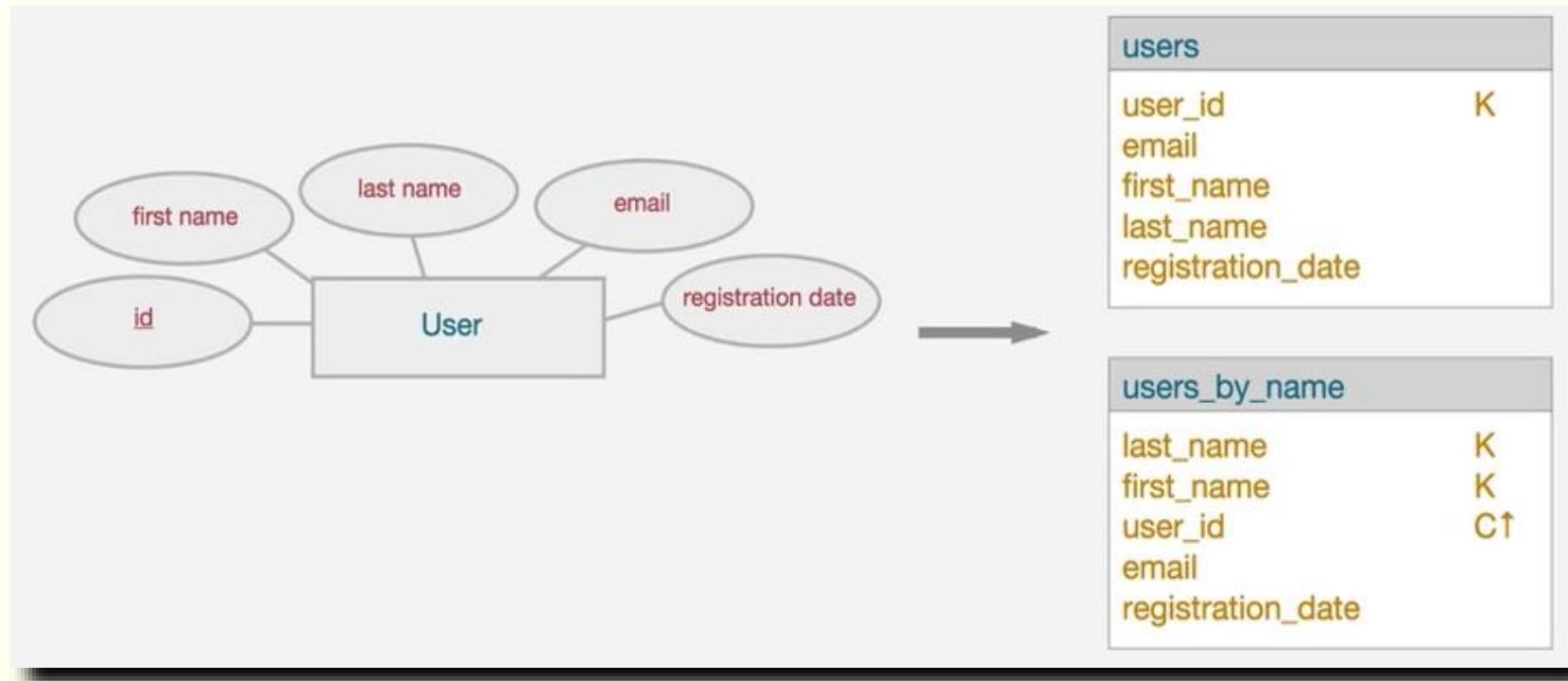
مثال: نگاشت موجودیت: پرس‌وجو روی ? و مرتبسازی صعودی بر اساس فیلد registration_data > ? و lastname = ?



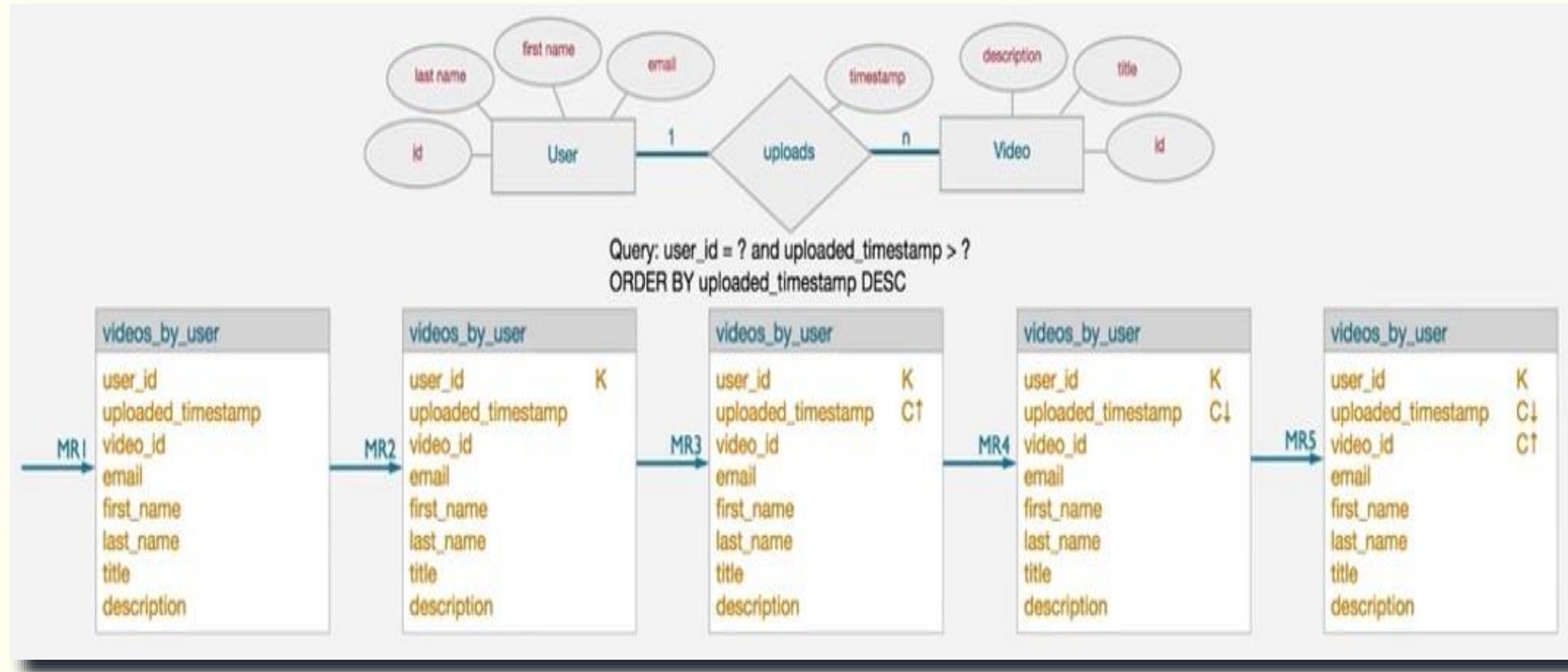
پایکاه داده – قانون پنجم نگاشت Cassandra

صفات کلیدی به ستون‌های کلید اصلی نگاشت می‌شوند.

مثال : نگاشت موجودیت :



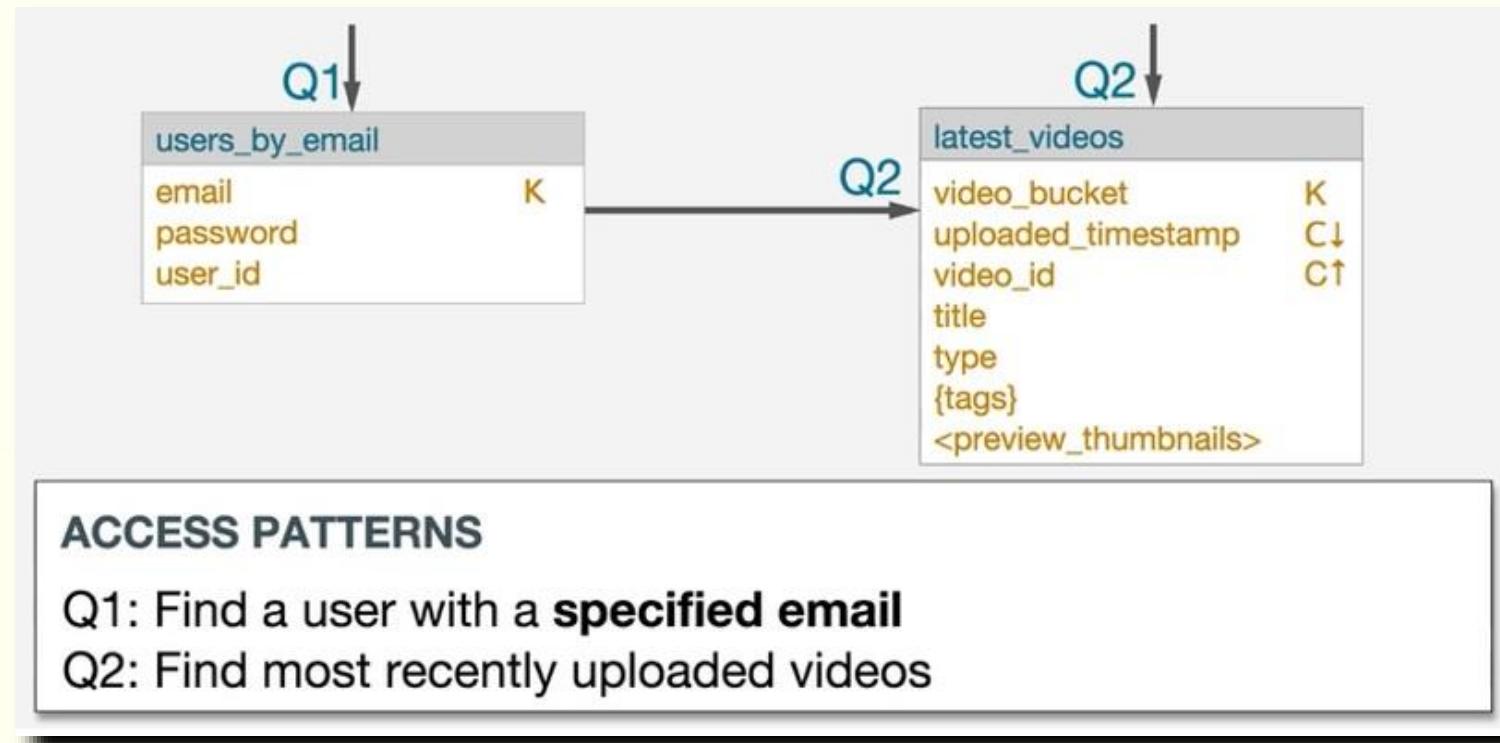
پایگاه داده مثال جامع بکارگیری قوایین نکاشت Cassandra



پایکاه داده‌ای منطقی – مدل داده‌ای Cassandra

مدل داده منطقی، نام و صفات مربوط به ستون‌ها را نمایش می‌دهد. برای نمایش مدل داده منطقی از "دیاگرام چبوتوکو" استفاده می‌کنیم. در این دیاگرام موجودیت‌ها و پرس‌وجوها جای خود را به جداول می‌دهند.

مثال ۱ : نمونه‌ای از مدل داده منطقی برای پرس‌وجوهای Q1 و Q2



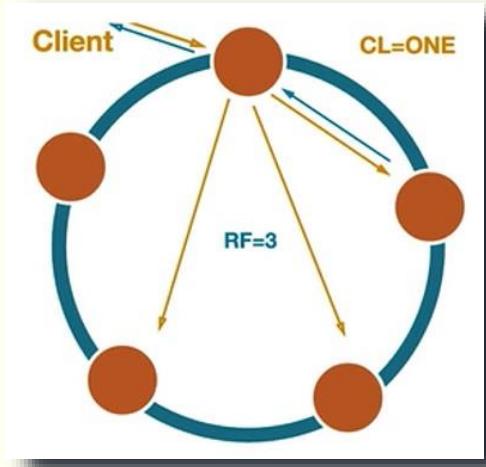
پایکاه داده‌ای فیزیکی – مدل داده Cassandra

در مدل داده فیزیکی علاوه بر نام و صفات ستون‌ها، نوع داده آن‌ها نیز نشان داده می‌شود:

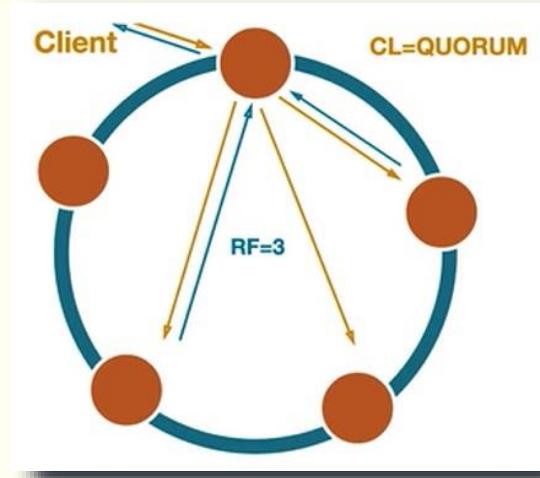
table_name	CQL Type	Definition
column_name_1	K	Partition key column
column_name_2	C↑	Clustering key column (ASC)
column_name_3	C↓	Clustering key column (DESC)
column_name_4	S	Static column
column_name_5	IDX	Secondary index column
column_name_6	++	Counter column
[column_name_7]		Collection column (list)
{column_name_8}		Collection column (set)
<column_name_9>		Collection column (map)
column_name_10	UDT Name	UDT column
(column_name_11)	CQL Type	Tuple column
column_name_12	CQL Type	Regular column



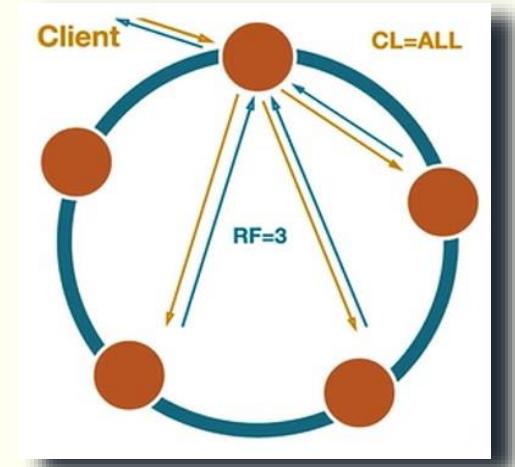
One



Quorum



All



Two

Local Quorum

Three

Each Quorum

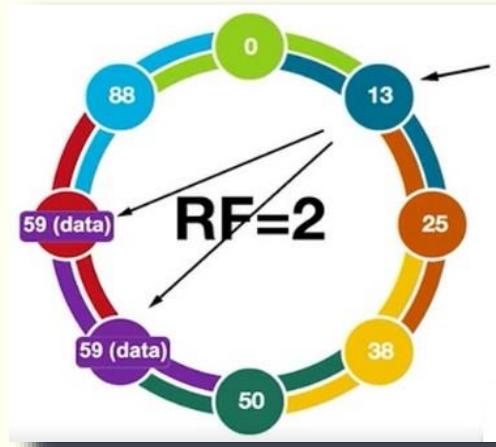


Simple Strategy

Create Keyspace sample1

With Replication =

('class': 'simple strategy', 'Replication Factor'=2)

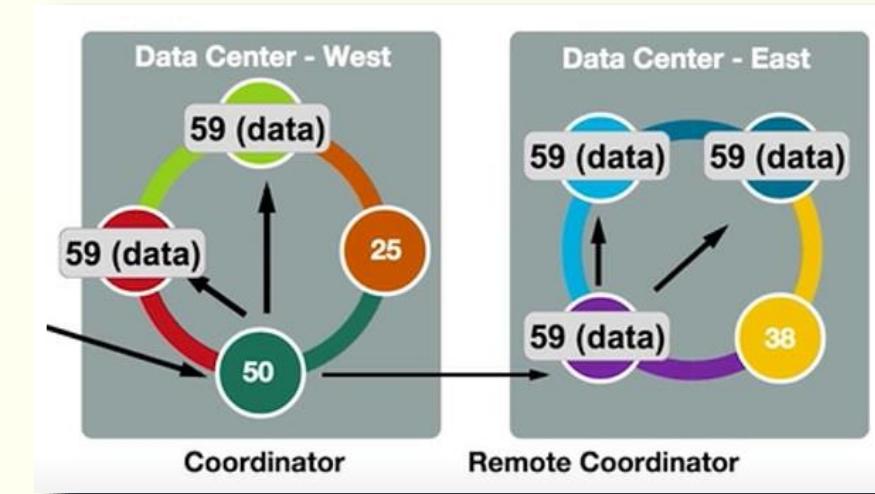


Network Topology Strategy

Create Keyspace sample2

With Replication =

('class': 'Network Topology Strategy', 'dc1':2, 'dc2':3)



شرکت‌های استفاده کننده از NoSQL

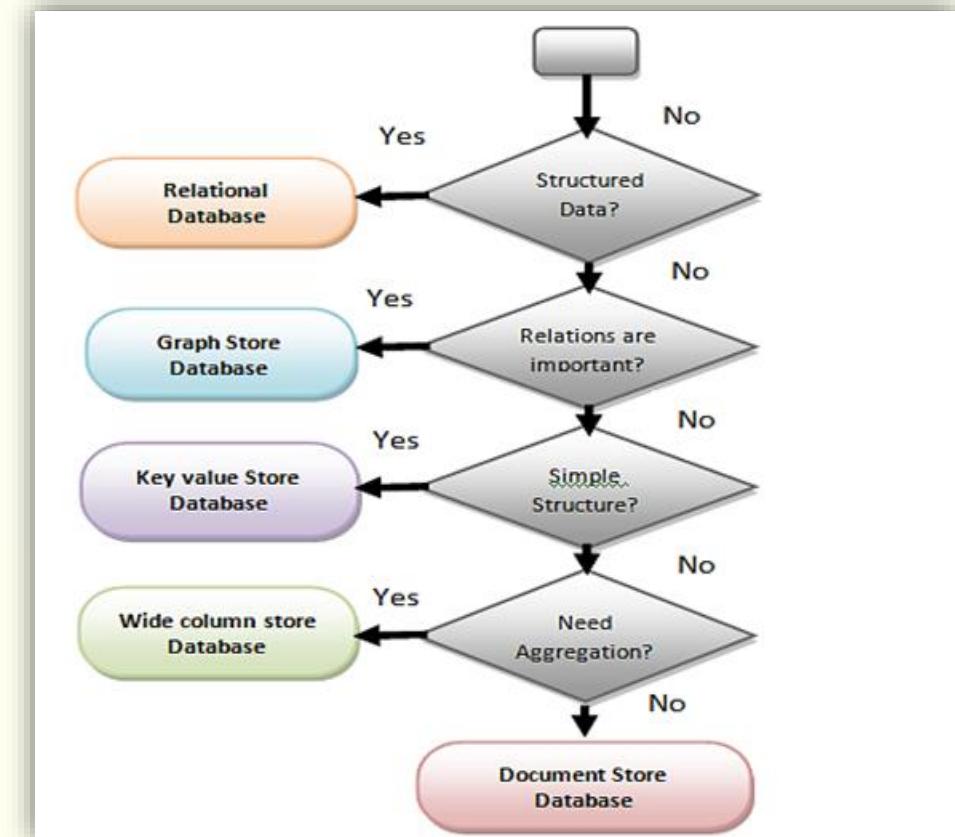
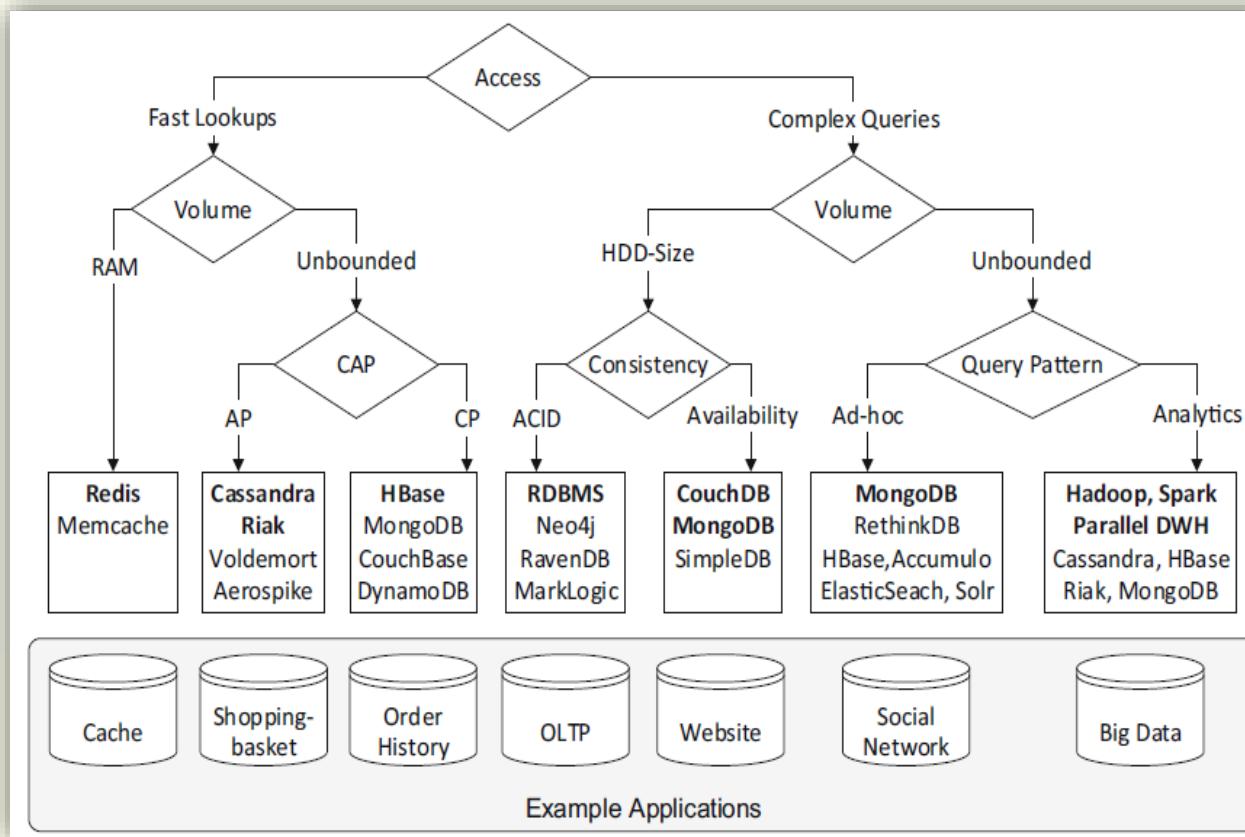
Company Name	NoSQL Name	NoSQL Storage Type
Adobe	HBase	Column
Amazon	Dynamo SimpleDB	Key-Value Document
BestBuy	Riak	Key-Value
eBay	Cassandra MongoDB	Column Document
Facebook	Cassandra Neo4j	Column Graph
Google	BigTable	Column
LinkedIn	Voldemort	Key-Value
LotsOfWords	CouchDB	Document
MongoHQ	MongoDB	Document
Mozilla	HBase Riak	Column Key-Value
Netflix	SimpleDB HBase Cassandra	Document Column Column
Twitter	Cassandra	Column



معیارهای انتخاب پایگاهداده مناسب

انتخاب پایگاه داده مناسب بستگی به نیاز برنامه کاربردی دارد و توجه به معیارهای زیر اهمیت دارد :

- ۱ - اندازه داده و میزان پیچیدگی
- ۲ - تئوری CAP
- ۳ - ساختار و ساخت یافتنگی دادهها



مقایسه پرخی از پایکاههای داده

System/tools	Data stores	Based on ACID properties	SQL-like language	Open source	Built in language	Function	Developed at	Features
Hadoop	Master/slave	No	No	Yes	Java, C	Distributed data Storage, Processing	yahoo	Flexible cost-effective
Cassandra	Column	No	Cassandra query language (CQL)	Yes	Java		Facebook	Highly scalable no single point failure
Neo4j	Graph	Yes	Cypher query language	Yes	Java	Graph processing	Neo technology	Graph storage and processing for super-fast writes and reads flexible data modeling
MongoDB	Document store	No	No	Yes	C++	Distributed data storage	MongoDB, Inc.	Schema less index on any attribute
Hypertable	Column	No	No	Yes	C++	Distributed data storage	Hypertable Inc.	Efficient fast performance
DyanamoDB	Key/value	No	Yes	No	NA	Data storage	Amazon	seamless throughput and storage scaling





پایان فصل دهم

مهدی دادبخش

mahdi.dadbakhsh@sharif.edu

۱۴۰۱ - ۱۴۰۲