

به نام خداوند بخشاینده مهربان

# ابزارهای کلیدی در تحلیل Big Data

محمد حیدری

علاقه مند به دیتا ساینس و بیگ دیتا آنالیتیکس

BigDataWorld.ir

0.0.1





سہ لائپھے اصلیٰ معماری

# سه لایه اصلی : لایه مدیریت منابع

Data Processing Layer

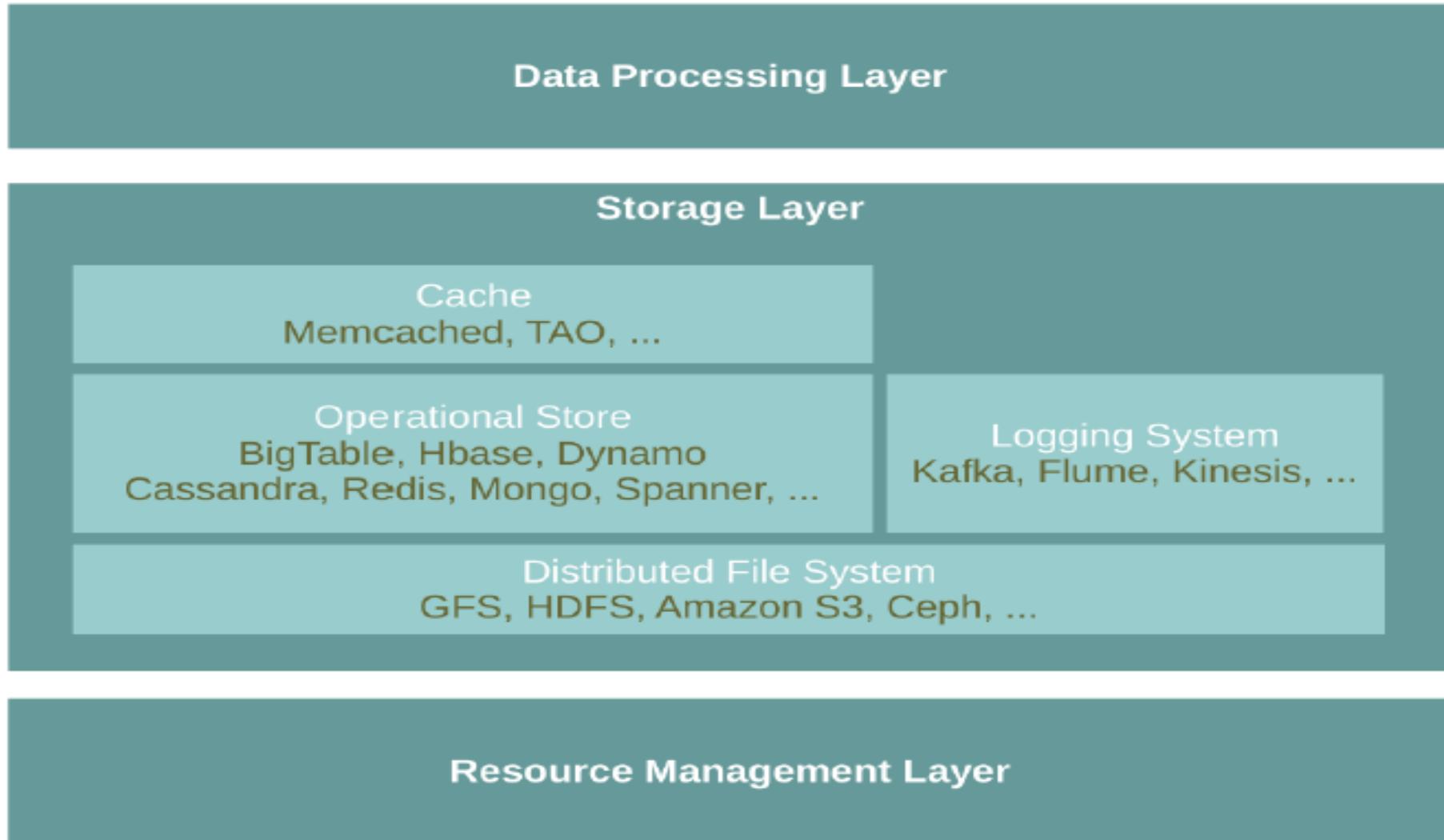
Storage Layer

Resource Management Layer

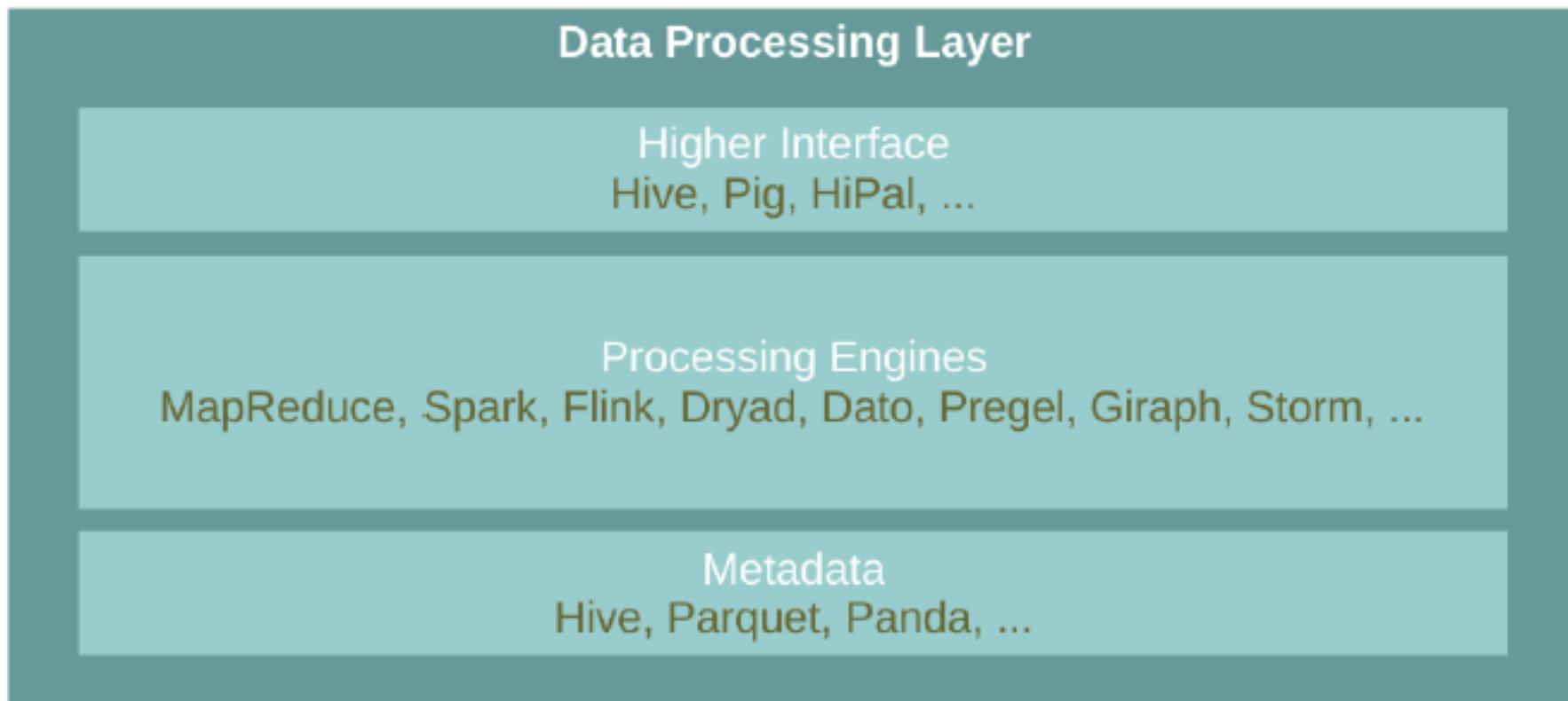
Resource Management Tools

Mesos, YARN, Borg, Kubernetes, EC2, OpenStack, ...

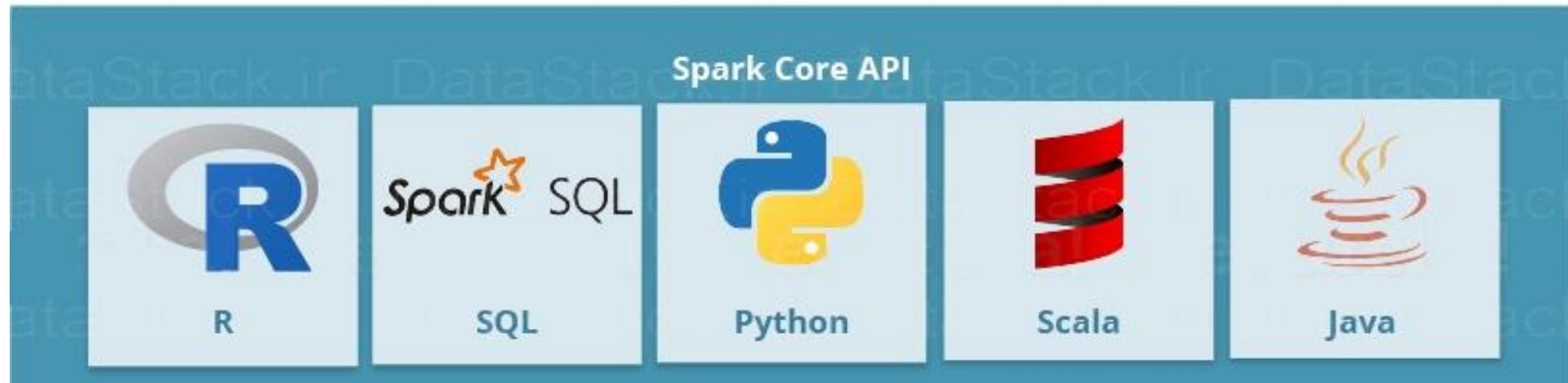
# سه لایه اصلی : لایه ذخیره سازی



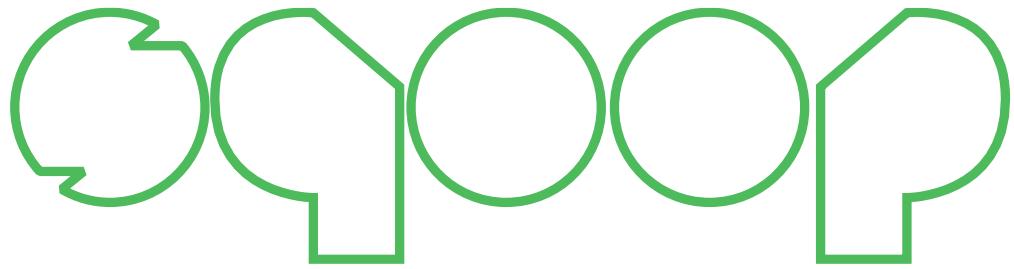
# سہ لایہ اصلی : لایہ پردازش



# موتور پردازشی اسپارک



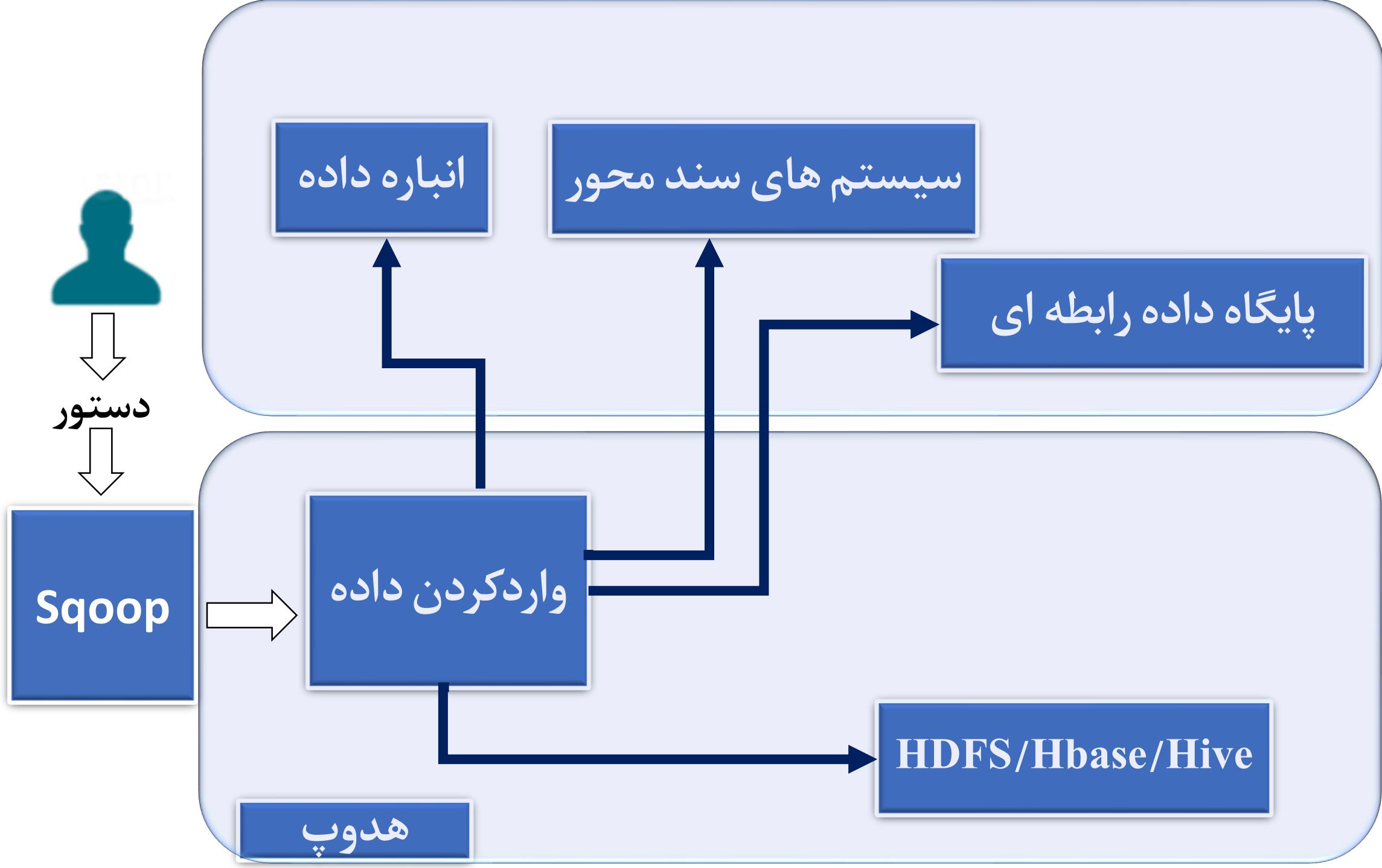
ابزارهای پکارچه سازی داده



# آپاچی اسکوپ - Apache Sqoop

- برنامه‌ی کاربردی مبتنی بر کنسول و مبتنی بر جاوا است.
- برای انتقال اطلاعات بین هدوب و غیر هدوب مانند Data Warehouse و NoSQL و RDBM طراحی شده است.
- DW: پایگاه داده‌ای است که برای گزارش‌گیری و تحلیل داده به کار می‌رود و هسته اصلی یک سیستم BI است.
- DW: از منابع مختلف اطلاعاتی سازمان جمع‌آوری، دسته‌بندی و ذخیره می‌شود.
- داده‌ها را از پایگاه داده‌های Relational به HDFS وارد کرده، و داده‌ها را از HDFS به پایگاه داده‌های ارتباطی صادر می‌کند
- برای وارد کردن داده‌ها از انبارداده‌های خارجی به Hive و HBase استفاده می‌شود.
- داده‌های وارد شده به هدوب می‌توانند قبل از صادر شدن به RDBMS به MapReduce تبدیل شوند.
- همچنین می‌تواند کلاس‌های Java را برای ارتباطات برنامه نویسی با داده‌های وارد شده ایجاد کند.
- از YARN برای وارد و صادر کردن داده‌ها استفاده می‌کند که تحمل خطا را در سطوح بالای موازی سازی فراهم می‌سازد.

```
[hope@heydari]# sqoop import -connect jdbc:mysql://localhost/employees --table employees -mohammad root
```



# آپاچی فلوم - Apache Flume

- یک سرویس توزیع یافته و قابل اطمینان برای جمع‌آوری و تجمعیح حجم انبوهی از لاغهای جریانی به محیط هدوف است.
- استفاده از فلوم به دلیل اینکه یک سرویس توزیع یافته است دشوار می‌باشد و نیازمند کانفیگ و تنظیمات بخصوص است.
- نیاز به عامل‌هایی دارد که به منابع داده‌ای متصل شوند.
- هر عامل در واقع از سه بخش سورس، کانال و سینک تشکیل شده است که قبل از عملیات انتقال باید کانفیگ آنها انجام شود.
- مثال
  - توییت‌هایی را که شامل نام دانشگاه تربیت مدرس باشد را از طریق Twitter API و Twitter Agent
  - به HDFS منتقل کنیم

```
//Start Flume after configuration
nohup flume-ng agent -conf-file /etc/flume/conf/flume.conf -name TwitterAgent
//Monitoring Flume in Logs collection
```

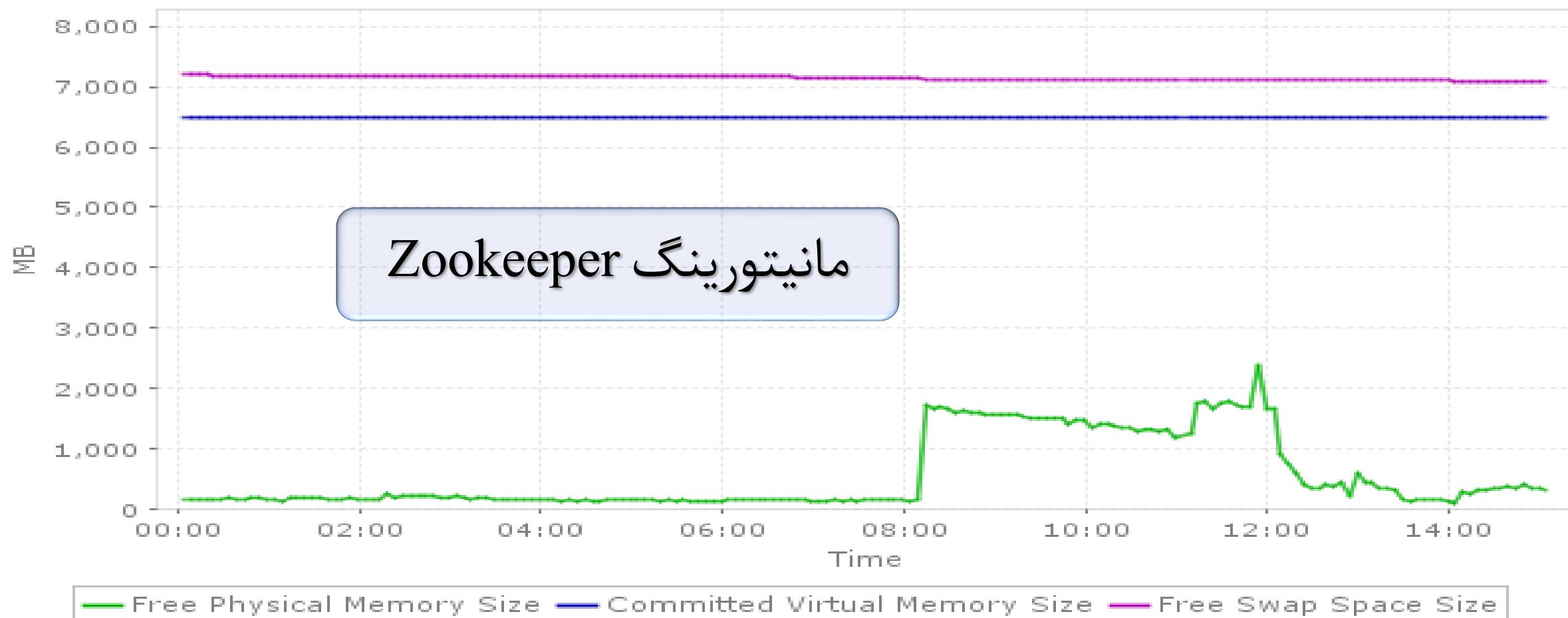
هماهنگی و مدیریت کلاسترها

# Apache ZooKeeper

- یک API متن باز است که اجازه می دهد تا پردازش های توزیع شده در سیستم های بزرگ با یکدیگر همگام سازی شوند که در نتیجه تمام مشتریانی که درخواستی می کنند اطلاعات سازگار دریافت کنند.
- مشخص می کند در یک زمان خاص کدام گره در سرویس ZooKeeper رهبر است.
- در صورت failشدن zookeeper leader محلی clientها می توانند از دیگری سرویس بخواهند.
- در واقع Zookeeper اطلاعات و متا دیتای کل سیستم را نگهداری می کند و می تواند بصورت یک مسیریاب درخواستها را به سرور مورد نظر ارسال کند.  
**Zookeeper as a Router** •

[Configure Alarms](#)

## MEMORY DETAILS



Total Physical Memory Size  
15,954.1 MB

Free Physical Memory Size  
328.9 MB

Committed Virtual Memory ...  
6,496.13 MB

Total Swap Space Size  
7,628 MB

Free Swap Space Size  
7,090.4 MB

# Apache Ambari

- پروژه ای است برای ساده تر کردن مدیریت هدوپ ، با مانیتورینگ فعالیتهای کلاسترها و پیکره بندی سرویسها.
- با فراهم آوردن محیطی بصری، مدیریت هدوپ را بسیار آسان می کند.
- قابلیت ها
- فراهم آوردن محیطی برای نصب گام به گام سرویسهای هدوپ برای تمام کلاسترها
- کنترل پیکره بندی سرویسهای هدوپ برای تمام کلاسترها
- فراهم آوردن مدیریت مرکزی برای به اجرا در آوردن ، متوقف کردن و دوباره پیکره بندی سرویسهای هدوپ برای تمامی کلاسترهای موجود
- فراهم آوردن محیطی برای مانیتورینگ وضعیت کاری کلاسترها و حجم کاری بر روی هر نود
- Ambari Metrics System برای جمع آوری معیارها
- Ambari Alert Framework ابزاری برای اطلاع رسانی مدیریت، هنگامی که توجه به سیستم ضروری می باشد(یکی از نودها متوقف شده ، فضای ذخیره سازی اندک می باشد و ...)

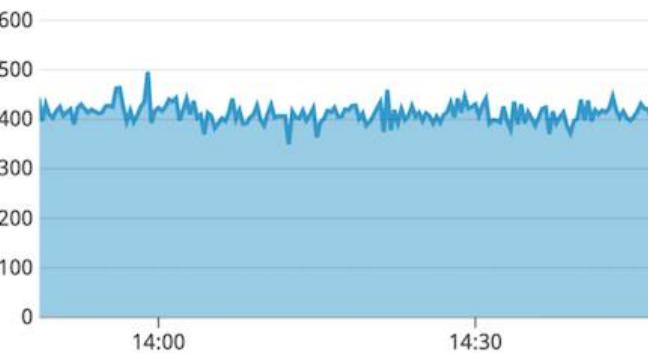


## Apache Ambari

## Agent Up

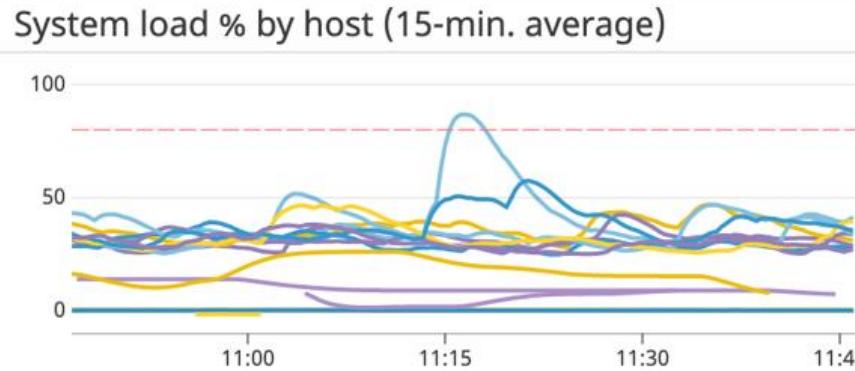
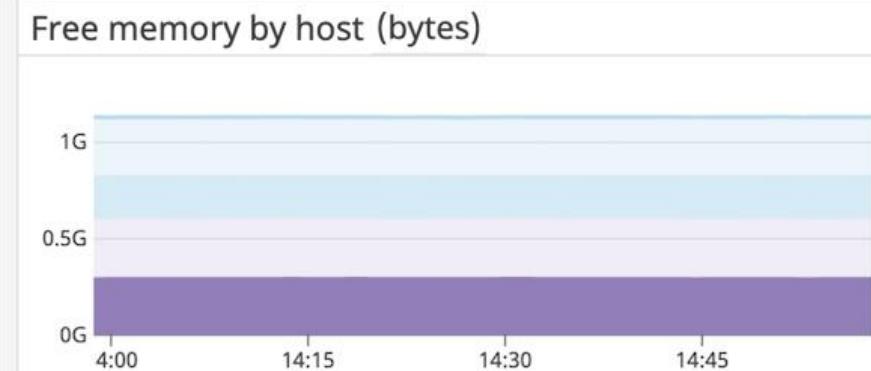
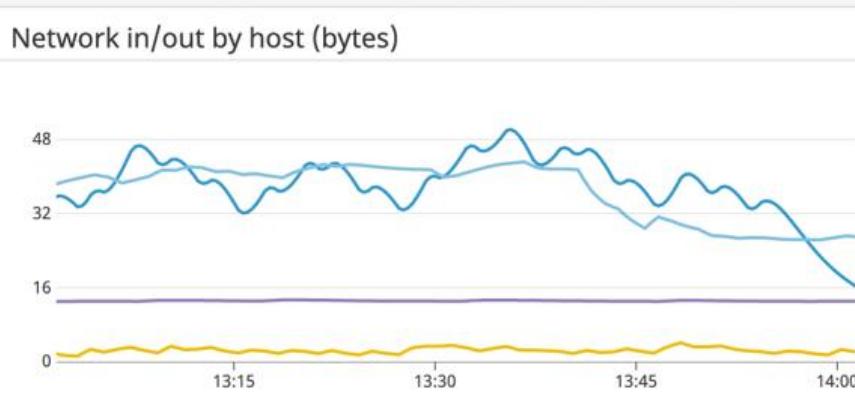
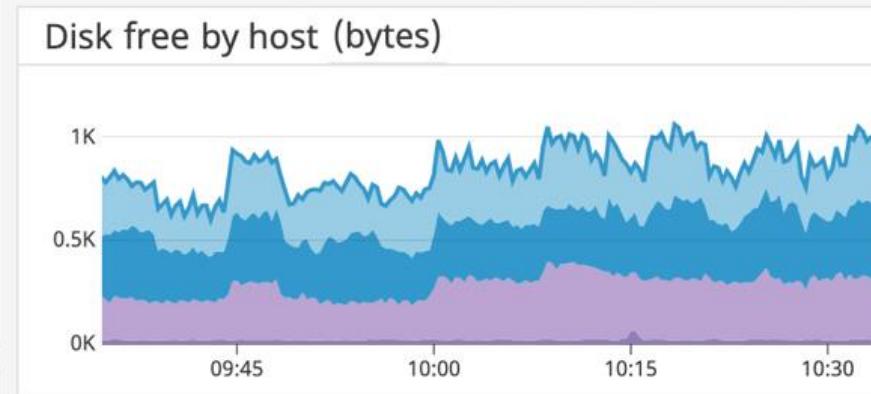
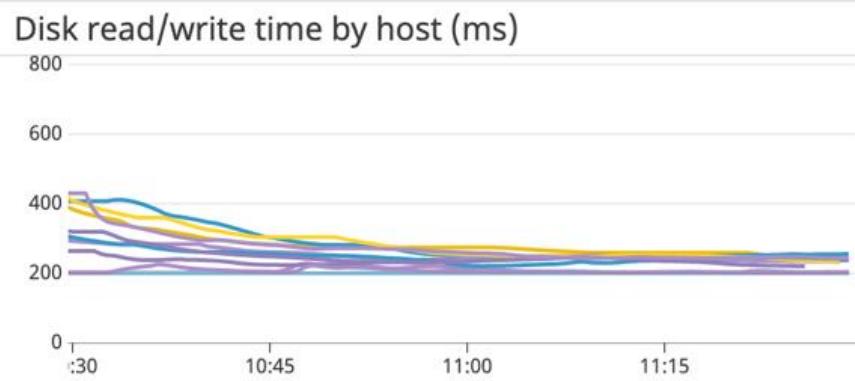
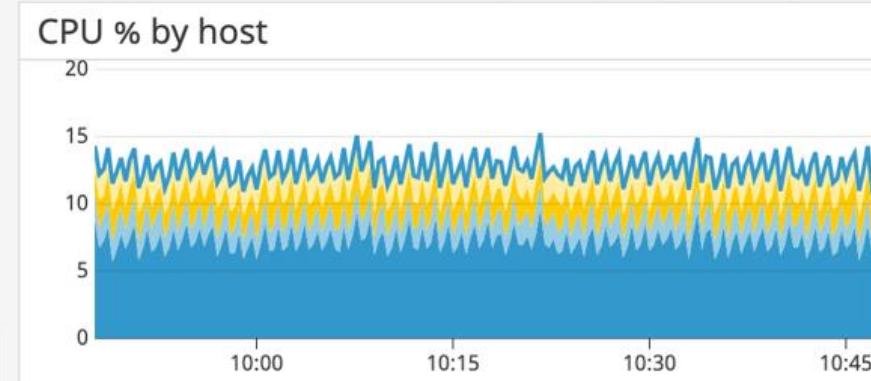
3

## Cluster process by cluster



# مانیپولینگ Ambari

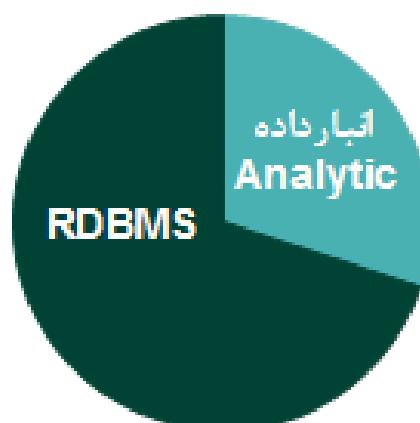
1h The Past Hour



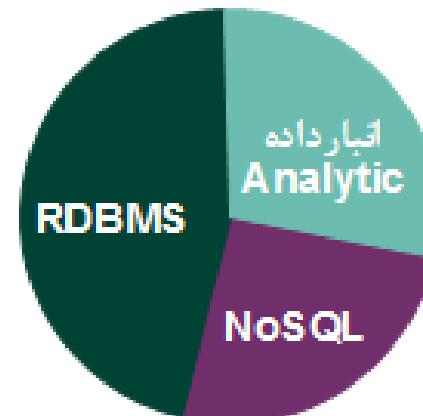
# مقایسه NoSQL و SQL



۱۹۸۵ - ۱۹۹۵



۱۹۹۵ - ۲۰۱۰



تاکنون - ۲۰۱۰

# چه زمانی از پایگاه داده SQL استفاده کنیم؟



- نیاز به جدول داریم.
- داده های ما ساده و روشن هستند.
- فیلدهای جداول تک مقداری هستند.
- داده ها ساختاری یکتا و یکپارچه دارند.

# چه زمانی از پایگاه داده NoSQL استفاده کنیم؟



- داده ها در کسری از ثانیه به مقیاس بزرگی می رسند.
- داده ها فاقد ساختار و پراکنده هستند.
- هر فیلد داده، آرایه ای از مقادیر متنوع است.
- نگران پایداری و تداوم داده ها هستیم.
- نگران از دسترس خارج شدن سرور هستیم.
- داده ها در محیط Cloud قرار دارند و می خواهیم از دیتابیس های توزیع شده استفاده کنیم.

# مزایا و معایب NoSQL



## • مزایا

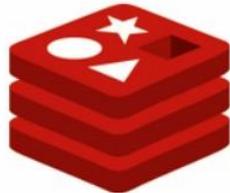
- سرعت بیشتر درج اطلاعات
- مقیاس پذیر و امکان گسترش آسان (Scale Out)
- مناسب و بهینه شده برای قالب های مشخص داده
- دسترسی پذیری بالاتر
- زمان پاسخ کوتاه
- قابلیت Replication (تکثیرپذیری داده ها)
- طراحی بدون Schema
- قابلیت Map Reduce

## • معایب

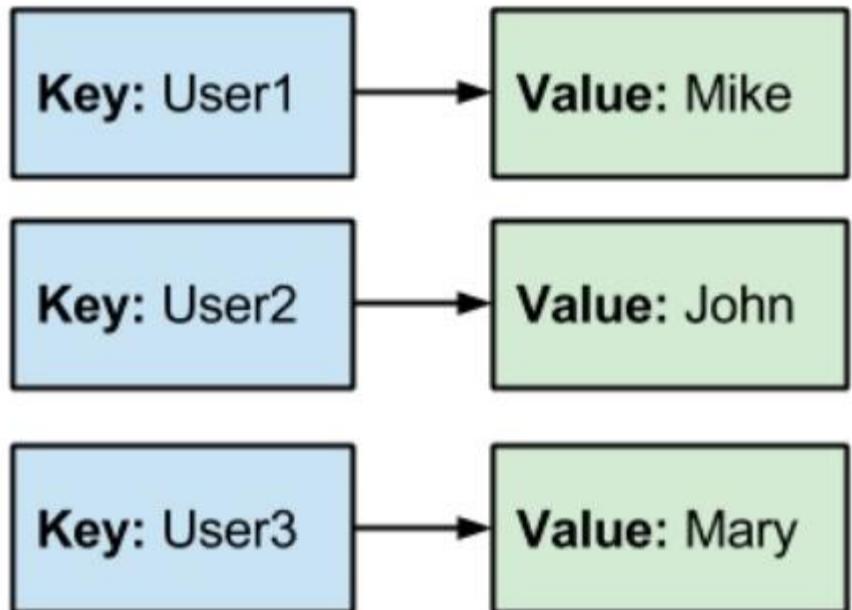
- نامناسب برای Join های پیچیده
- سرعت بازیابی داده های نرمال و ایندکس شده پایین تر است.



The logo for riak, consisting of the word "riak" in a bold, lowercase sans-serif font with a small orange dot above the letter "i".



The logo for redis, consisting of the word "redis" in a bold, lowercase sans-serif font.



## دیتابیس های Key-Value

- بر اساس مقاله **Amazon Dynamo**

- مدل داده ای: مجموعه ای از زوج های کلید-مقدار

- عملیات:

- `Insert(k,v)`, `fetch(k)`, `update(k)`, `delete(k)`

- مزایا

- بسیار سریع

- بسیار مقیاس پذیر

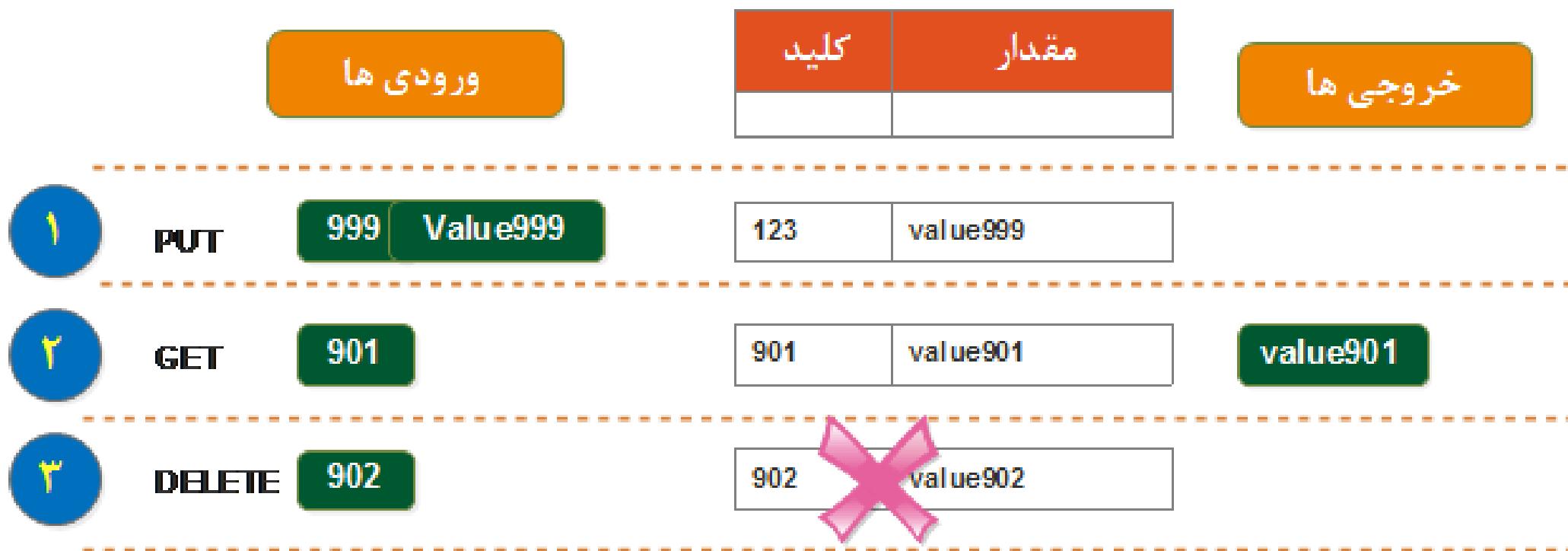
- مدل ساده

- قابلیت توسعه افقی (**Scale Out**)

- چالش

- بسیاری از ساختارهای داده قابلیت مدل شدن به صورت کلید - مقدار را ندارند.

# سه دستور key-value store را بروگه نویسی و Delete , Put , Get





## Column DB

- ذخیره سازی ستون محور

- محتوایش را بجای سطر در قالب ستون ذخیره می کند.

- می توان به هر سطر از داده ها، ستون خاص خود را نسبت داد.

- قابلیت ذخیره سازی کلید مقدار و بازیابی آن با کلید در یک سیستم عظیم موازی

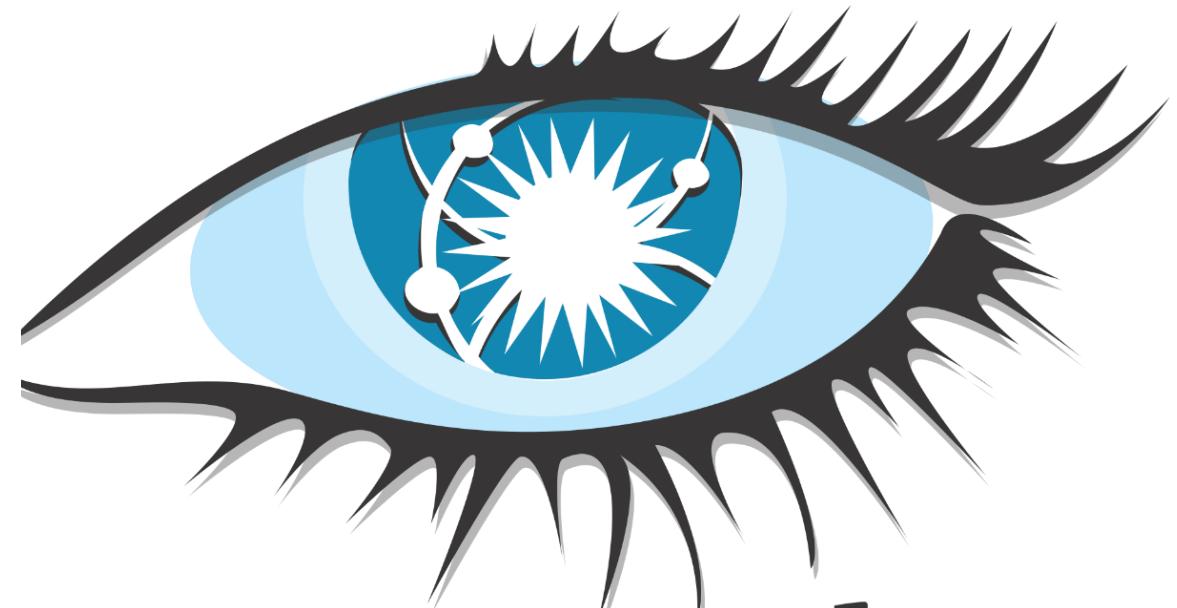
- قابلیت پارتیشن بندی بصورت عمودی و افقی

- بر اساس مقاله Google Big Table

- مدل داده ای : Big Table, Columns Family

- Cassandra, HBase, HyperTable

تّقابل دو رقیب هم نوع (ترکیبی)



# آپاچی کاساندرا - Apache Cassandra

- این دیتابیس یکی از خارق العاده ترین ها در دنیاست. (یک دیتابیس ترکیبی (Key-Value + Columnar) (Relational و Transactional هم از آن استفاده کرد.
- مفهومی به نام **Column Family** برای تقسیم‌بندی گروهی از ستون‌های مرتبط با یکدیگر
- مثال: یک **Column Family** برای مشخصات اشخاص
- توسعه پذیری آن بصورت افقی است و می‌توان به تعداد دلخواه به آن **Node** اضافه کرد!!!
- زبان کوئری متخصص خود را دارد.
- (CQL - Cassandra Query Language)، نکته: با زبان کوئری Cypher اشتباه نشود.
- می‌تواند به صورت اتوماتیک، رپلیکیشن را در سطح Node و Rack در دیتاستر انجام داد.
- می‌تواند Node های خراب را بدون مشکل **Down Time** جایگزین کند. (خاموشی سرور)
- نیاز نیست تا برای ذخیره هر رکورد جدید، تمام مقادیر مرتبط با ستون‌ها را دانسته یا از null برای پرکردن مقادیر ناآشنای آن‌ها استفاده کنیم.
- به سادگی می‌توانیم مقادیری را که نمی‌خواهیم، ذخیره نکنیم. ( فقط مقادیر ضروری را ذخیره می‌کنیم)

# آپاچی کاساندرا - Apache Cassandra

- کاساندرا **network bottlenecks** و **single points of failure** ندارد.
- برای نرم افزارهایی که **نمیخواهند** حتی یک **bit** داده از بین برود بسیار مناسب هست
- **حتی زمانی** که دیتاستر **down** میشود و در مقابل تغییرات اجزای زیرساختی بسیار مقاوم است و می تواند آنلاین باقی بماند.
- **در قلب صنعت**
  - Netflix
  - با 2500 node و حجم داده 420TB و بیش از 1 هزار میلیارد (1 تریلیون) درخواست در روز
  - Apple
  - با 75000 node و حجم داده 10 پتابایت

# آپاچی کاساندرا - Apache Cassandra

## ■ نقاط قوت

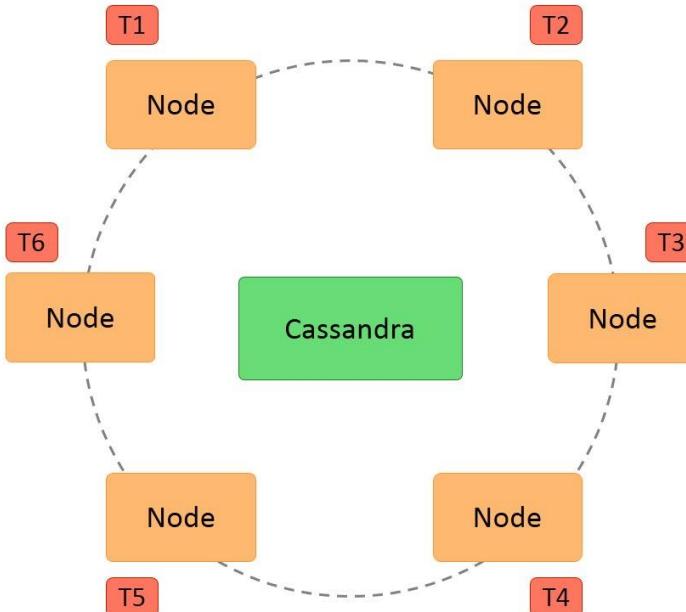
- سرعت بسیار زیاد در نوشتن داده ها
- سرعت مطلب در خواندن داده ها
- دسترسی پذیری بالا بدون **single points of failure**
- پشتیبانی از Map Reduce
- پشتیبانی از Replication
- قابلیت Multi Data Center Replication !!!!!!!!!!!!!!!
- قابلیت تنظیم سازگاری بصورت دلخواه
- نقاط ضعف
- از SQL و Join در Subquery پشتیبانی نمی کند

# آپاچی کاساندرا - Apache Cassandra

- در Cassandra از معماری **Master/Slave** استفاده نشده است.
- زیرا در این معماری معمولاً گره Master به دلیل **کارکرد زیاد** به گلوگاه سیستم تبدیل می شود.
- یعنی در صورتی که master از کار بیفتد یا کند شود، **کل سیستم از کار می افتد یا کند می شود.**

**No Fail Over at all**

- تمامی گره ها (رایانه ها)ی متصل به هم، مانند یکدیگر رفتار می کنند. داده ها در گره های مختلف Replicate می شوند.



# نصب در لینوکس

- حتما Java نصب شده باشد.
- دانلود نسخه باینری Build شده (عدم نیاز به کانفیگ های زمان بر) از Apache
- Linux Environment Path به Cassandra Root Directory
- اضافه کردن `bin/Cassandra -f`
- دستور `Background` کاساندرا در

```
wget http://archive.apache.org/dist/cassandra/3.10/apache-cassandra-3.10-bin.tar.gz
```

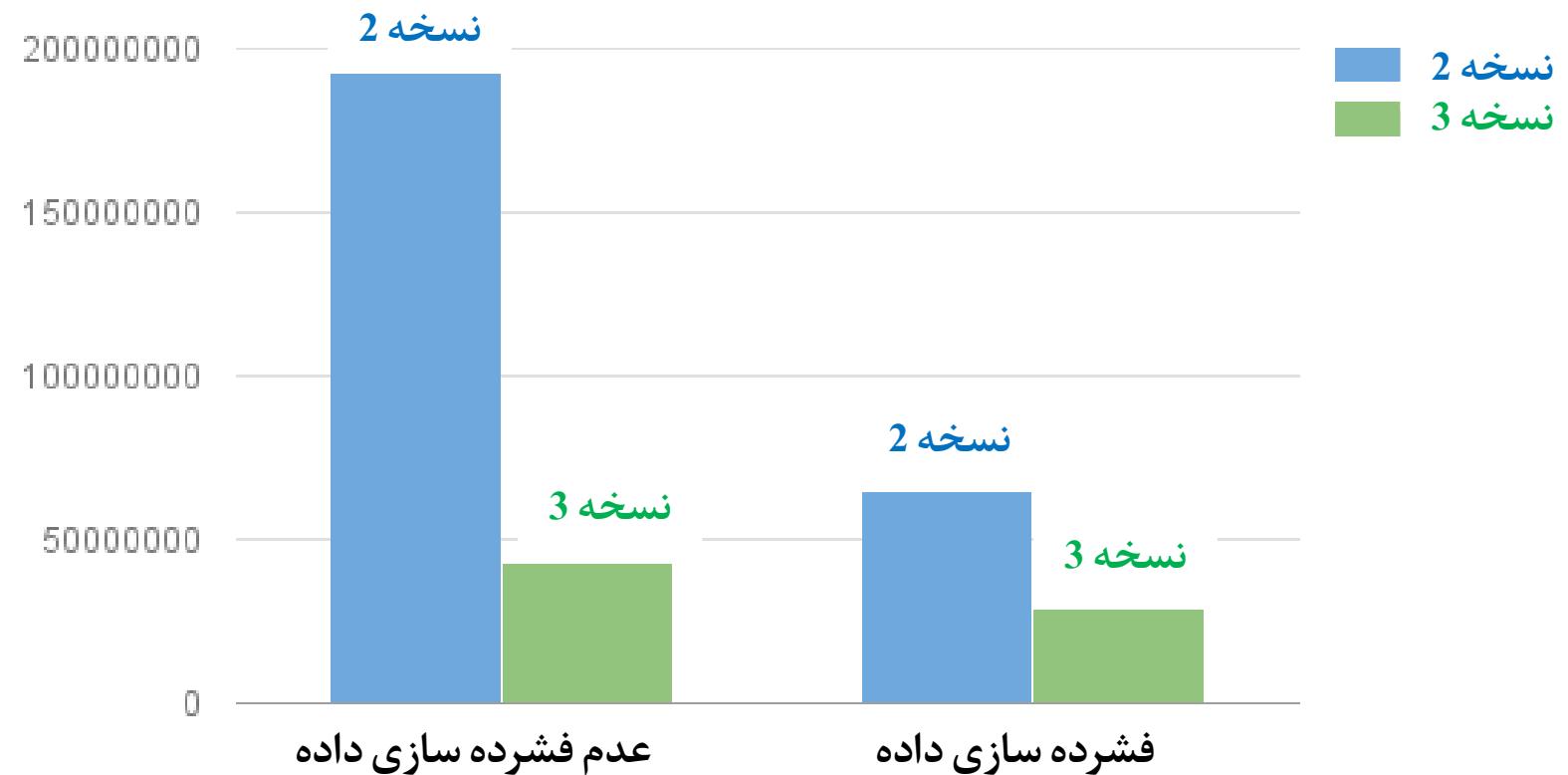
```
tar -xvzf apache-cassandra-3.10-bin.tar.gz
```

```
export CASSANDRA_HOME=/YOUR ROOT CASSANDRA DIRECTORY
```

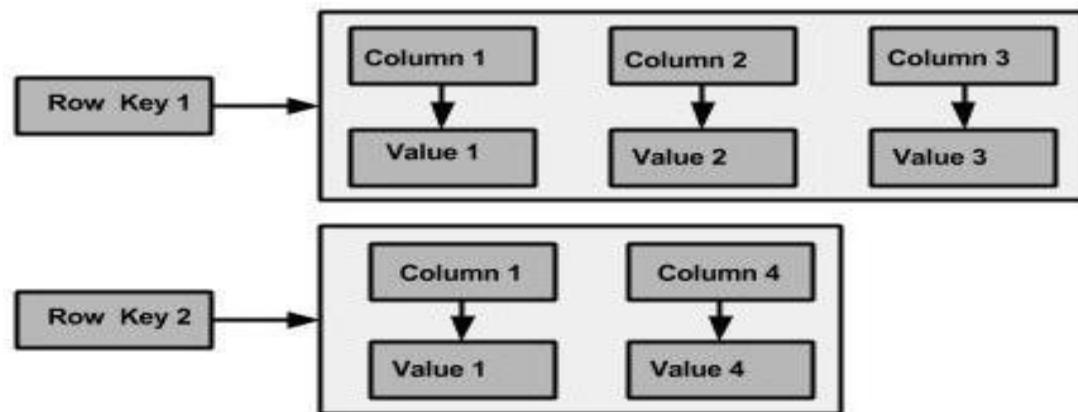
```
PATH=$PATH:$CASSANDRA_HOME/bin:
```

```
cassandra -f
```

## بهمبود فشرده سازی داده در کاساندرا در نسخه 3 در مقایسه با نسخه 2

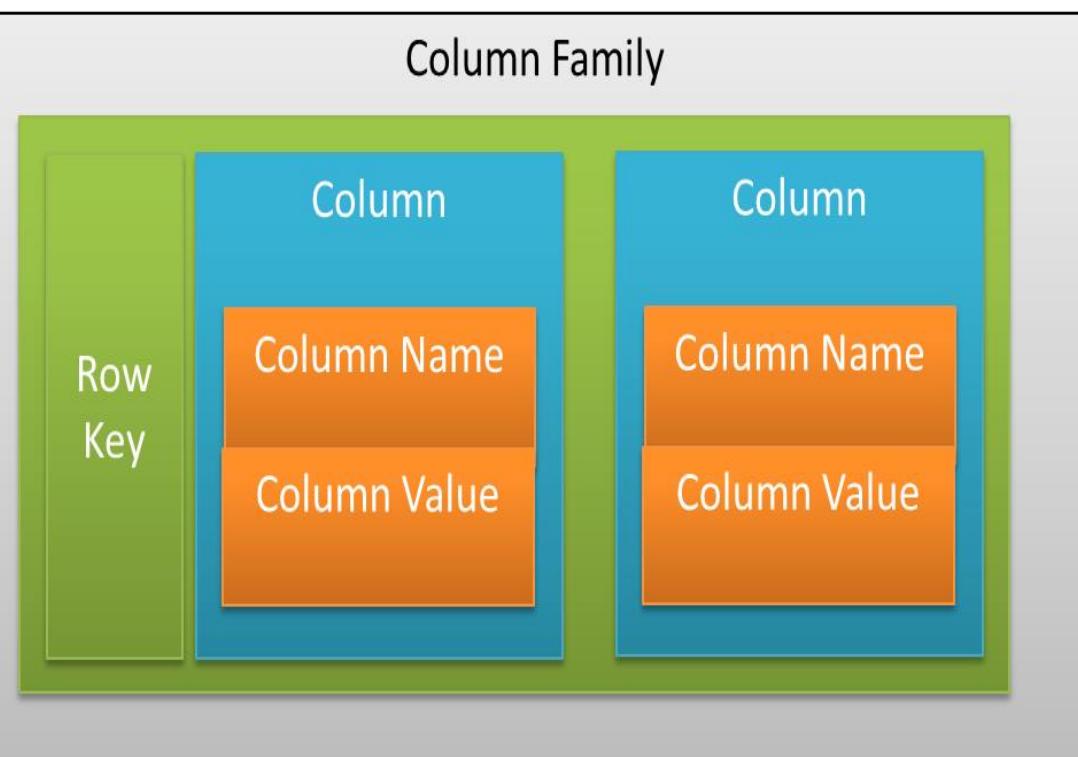


# طراحی داده و آشنایی با مدل داده ای کاساندرا



جدول کاربران		
ایمیل	نام	شناسه کاربر
hi@bigdataworld.ir	محمداحسان	101

Column Family



جدول توییت ها

پیام	نام	شناسه نویسنده	شناسه توییت
سلام!	محمداحسان	101	9990

Followed

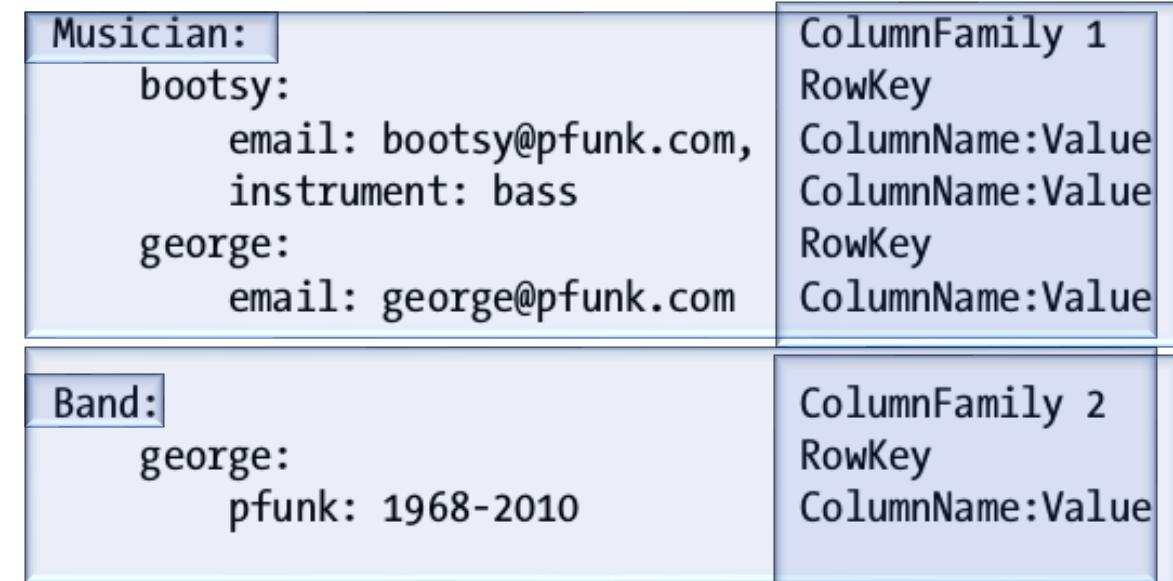
لیست فالورها	شناسه کاربر
[104,109]	101

Follows

لیست فالورها	شناسه کاربر
[101,117]	104

# ذخیره سازی داده در Cassandra با MySQL

	emp_id	emp_name	performance	salary
▶	1	Mary Doe	1	50000
	2	Cindy Smith	3	66950
	3	Sue Greenspan	4	78750
	4	Grace Dell	5	135000
	5	Nancy Johnson	3	87550
	6	John Doe	2	45450
	7	Lily Bush	3	56650
	8	Jack William	NULL	43645
	9	Ricky Bond	NULL	52780



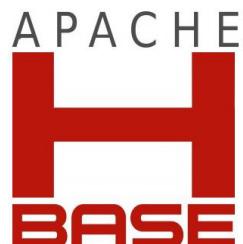
# ساخت یک جدول ساده با یک مقدار در Cassandra

cqlsh>

- **CREATE KEYSPACE** "Bigdata"
- **WITH replication = {'class': 'Strategy name', 'replication\_factor' : 'No.Of replicas'};**
- **CREATE TABLE** BigdataTools(  
    toolID int **PRIMARYKEY**,  
    toolName string,  
    toolType string.  
)
- **INSERT INTO** BigdataTools  
(toolID toolName, toolType)  
**VALUES** (1, Kafka, StreamProcessing)

## ((Key-Value + Columnar) (دیگر دیتابیس ترکیبی Apache HBase

- یک محل ذخیره داده و با نوع key/value و بر مبنای ستون است.
- برای پشتیبانی از نرخ بالای به روز رسانی جدول، و برای مقیاس گذاری به صورت افقی در خوشة های محاسباتی توزیع شده طراحی شده است.
- برای مثال، جداولی که حاوی میلیاردها ردیف و میلیون ها ستون هستند.
- یکی از برجسته ترین موارد استفاده از HBase به عنوان رسیدگی کننده داده های ساخت یافته برای زیرساخت های پایه پیام رسانی فیس بوک است.
- مانند هدوپ، HBase نیز معمولاً با استفاده از JAVA و نه SQL، برنامه ریزی می شود
- هر جدولی باید یک ستون به عنوان کلید اولیه داشته باشد و تمامی اتصالات به جداول HBase باید با استفاده از این کلیدها صورت گیرد.





## Apache HBase

- از لحاظ فنی، HBase در واقع بیشتر یک محل ذخیره سازی داده هاست
- تا اینکه بخواهد یک پایگاه داده باشد.
- زیرا بسیاری از ویژگی های یک سیستم RDBMS از جمله ستون های تایپ شده، ایندکس های ثانویه، راه انداز و زبان های پرس و جوی پیشرفته را ندارد.
- خوش بندی HBase با افزودن RegionServers ها که توسط سرورهای نسبتاً ارزان قیمت میزبانی می شوند، گسترش می یابد.
- برای مثال، اگر یک خوش RegionServers از 10 تا 20 گسترش یابد، از نظر ذخیره سازی و همچنین ظرفیت پردازش، دو برابر می شود
- در حالیکه HBase را می توان با استفاده از سخت افزار های ارزان گسترش داد.
- امکاناتی مانند Big Table را برای هadoop فراهم می آورد.
- راهی با تحمل پذیری خطأ، برای ذخیره سازی تعداد زیادی از داده های اسپارس را فراهم می آورد.

# مفهوم Column Family

میزان فروش از تولیدات دانش بنیان		دانشجو		کلید سطر
مقدار	فیلد	استان	نام	شناسه کاربر
100,000,000	مهندسی IT	اصفهان	محمد	101
2,000,000	مهندسی برق	یزد	احسان	102
5,000,000	مهندسی مکانیک	Zahedan	آرش	103
8,000,000	مهندسی هوافضا	بوشهر	کورش	104
222,000,000	مهندسی شیمی	خراسان	کامیار	105
2,000,000,000	مهندسی IT	خوزستان	کامبیز	106

# Google Big Table

- سرویس ذخیره سازی اختصاصی گوگل است که سرعت و خواندن و نوشتن بسیار سریع را ارائه می دهد.
- از یک معماری داخلی پیشرفته ای که الگوهای دسترسی را یاد می گیرد و داده های شما را منتقل می کند، استفاده می کند.
- اگر که واقعا با **حجم زیاد داده** و یا **داده های فاقد ساختار سروکار دارید** و **کارایی بالایی** نیاز دارید که پایگاه داده های رابطه ای **نمی توانند جوابگو باشند**، این سرویس مناسب کارتان خواهد بود.
- این سرویس برای کار با Data Big در نظر گرفته شده است بخصوص زمانی که که شرکت های کوچک تری که توانایی فراهم کردن زیرساخت لازم را ندارند از آن استفاده کنند.
- در ازای دریافتی مبلغی به صورت ماهانه
- در بیش از ۶۰ محصول و پروژه مربوط به شرکت گوگل استفاده می شود

# مدل داده ای ساده (یک کلید سطری، یک کلید ستونی و یک برچسب زمانی)

“follows” column family

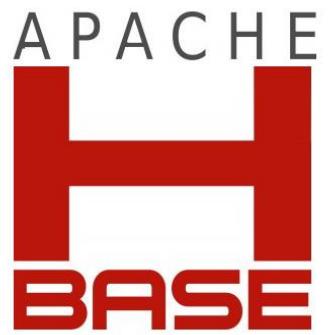
		Follows			
Row Key	gwashington	jadams	tjefferson	wmckinley	
gwashington		1			
jadams	1			1	
tjefferson	1	1	1		1
wmckinley				1	

Multiple versions

The diagram illustrates a simple data model using a "Follows" column family. It consists of a row key and a column key. The row key is "Follows" and the column key is "Follows". The data is organized into a grid where rows represent the row key and columns represent the column key. The grid contains the following values:

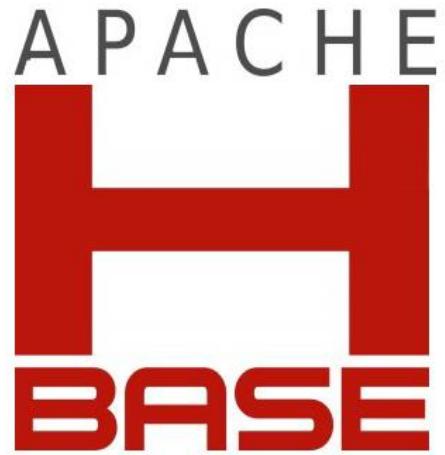
- Row "gwashington": Column "gwashington" is empty, Column "jadams" is "1", Column "tjefferson" is empty, Column "wmckinley" is empty.
- Row "jadams": Column "gwashington" is empty, Column "jadams" is empty, Column "tjefferson" is empty, Column "wmckinley" is empty.
- Row "tjefferson": Column "gwashington" is empty, Column "jadams" is "1", Column "tjefferson" is "1", Column "wmckinley" is empty.
- Row "wmckinley": Column "gwashington" is empty, Column "jadams" is empty, Column "tjefferson" is empty, Column "wmckinley" is empty.

A blue box highlights the value "1" in the cell at the intersection of the "tjefferson" row and column. A black arrow points from the text "Multiple versions" to this highlighted cell, indicating that there are multiple versions of the data stored in this column family.



## Apache HBase

- **ویژگی ها**
  - خواندن و نوشتن بسیار پایدار داده ها
  - تقسیم خودکار داده ها:
  - جداول HBase بر روی خوشه ها توزیع شده اند و این جداول بصورت خودکار با رشد مقدار داده ها بر روی خوشه ها مجددا توزیع می شوند.
  - **یکپارچگی** Hbase و Hadoop
  - HBase: از فایل سیستم هدوف به عنوان فایل سیستم توزیعی خود پشتیبانی می کند.
  - **پشتیبانی از MapReduce** Java API
  - جداول در HBase می توانند به عنوان ورودی یا خروجی برای کارهای Map Reduce که بر روی Hadoop اجرا می شوند عمل نمایند.



## چه زمانی از Apache Hbase استفاده کنیم؟

- به اندازه کافی داده در اختیار دارید
- صدها میلیون یا میلیارد رکورد، گزینه خوبی است
- تنها چندین هزار یا میلیون ردیف، RDBMS بهتر است.
- صورتی که یک رابط برای اتصال به فیسبوک و جمع آوری حجم بسیار زیادی داده و با سرعت بالا در اختیار دارید HBase می تواند یکی از گزینه های شما باشد.
- می توانید بدون بکارگیری ویژگیهای RDBMS مانند ستون های نامگذاری شده، انواع ایندکس ها بر روی جداول، تراکنش ها، و زبانهای پرس و جوی پیشرفته به کار خود ادامه دهید
- باید طمئن شوید سخت افزار کافی در اختیار دارید. (تعداد)
- حتی فایل سیستم هدوپ نیز با تعداد نودهای کمتر از 5 معمولاً نمی تواند بخوبی کار کند.
- می توان بخوبی و براحتی بر روی یک لپ تاپ نیز بکار گرفت ( فقط تستی نه تجاری )

# دیتابیس های گرافی



TITAN

 ArangoDB

 neo4j

- مُلهم از اویلر و تئوری گراف
- مدل داده ای: گره ها، یال ها
- قابلیت مناسب برای طراحی یک سیستم جست و جو
- گراف جست و جوی فیس بوک مبتنی بر Neo4j
- یک نمونه کوئری:

▪ اساتیدی که در حال حاضر مشغول به پژوهش در حوزه کلان داده در دانشگاه استنفورد هستند، زیر چهل سال سن دارند، در ده سال گذشته ، در یکی از دانشگاه های اروپا تحصیل کردند و استاد راهنمایی که داشتند اصالتا هلندی بوده است و تابحال دست کم سه پژوهش در یکی از ژورنال های معتبر بیگ دیتا با ضریب تاثیر بالای سه، منتشر کرده اند و حداقل سه سال سابقه کار در صنعت در یکی از کمپانی های آمریکایی را دارند .

- یک کوئری مشکل یا غیرممکن با SQL



## دیتابیس Neo4j

- در اصل بر مبنای JAVA می باشد
- برای کاربردهایی که میزان نوشتن کم و میزان خواندن بالاست مناسب است.
- برای کاربردهایی که گراف را بصورت دوره ای بروزرسانی می کند اما بار خواندن آنها بسیار بالاست مناسب است.
- در مقایسه با دیتابیس های رابطه ای دیگر نیازی به نوشتن join های پیچیده نیست
- براحتی با پیمایش و حرکت در گراف میتوان انواع Query های متنوع را انجام داد.
- از تکنیک Sharding (بخش بندی داده) نمی توان استفاده کرد
- چون مساله ریاضی تقسیم بهینه گراف بزرگ در بین مجموعه از سرویس دهنده ها NP Complete (پرهزینه) است.



ArangoDB

TITAN



## دیتابیس Neo4j

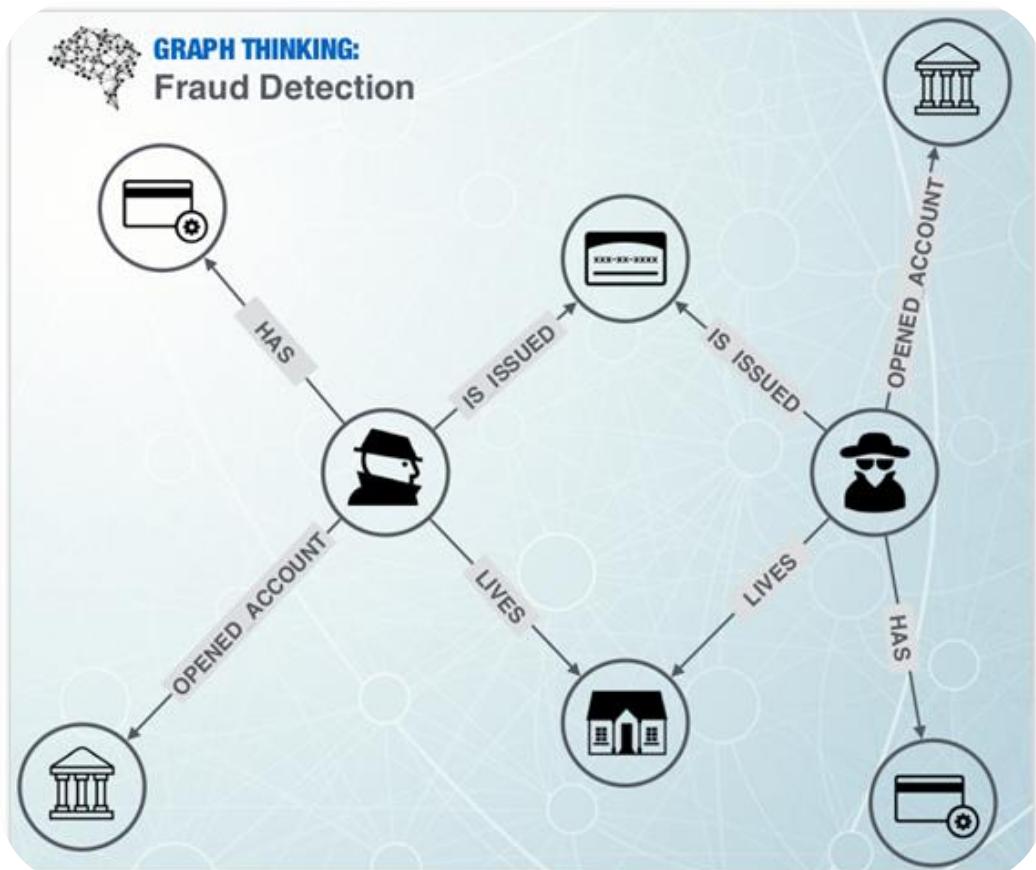
### ■ مزایا

- رتبه اول دیتابیس های گرافی دنیا بر اساس وب سایت معتبر db-engines.com
- یادگیری آسان و سهولت در استفاده
- کاهش استفاده از حافظه
- - بکاپ گیری حرفه ای
- انعطاف پذیر Schema
- زبان پرس و جوی Cypher
- 10 تا 100 برابر کدنویسی کمتر نسبت به SQL
- سرعت بالا
- درایور برای زبان های جاوا، سی شارپ، پایتون، جاواسکریپت، رو بی، پی اچ پی، آر، ...
- پشتیبانی از فریمورک های اسپرینگ، جنگو، لاراول و ...



ArangoDB

TITAN



## دیتابیس Neo4j

### ▪ کاربردهای صنعتی

- سیستم های توصیه گر Realtime

- تشخیص تقلب و پولشویی در صنعت بانک و بیمه

- جستجوی گراف محور

- شبکه های کامپیوترا

- پیاده سازی الگوریتم های Routing در Huawei

# Typical Complex SQL Join

```
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
FROM (
SELECT manager.pid AS directReportees, 0 AS count
  FROM person_reportee manager
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
UNION
  SELECT manager.pid AS directReportees, count(manager.directly_manages) AS count
  FROM person_reportee manager
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
UNION
SELECT manager.pid AS directReportees, count(reportee.directly_manages) AS count
  FROM person_reportee manager
 JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
UNION
SELECT manager.pid AS directReportees, count(L2Reportees.directly_manages) AS count
  FROM person_reportee manager
 JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
FROM (
SELECT manager.directly_manages AS directReportees, 0 AS count
  FROM person_reportee manager
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
UNION
  SELECT reportee.pid AS directReportees, count(reportee.directly_manages) AS count
  FROM person_reportee manager
 JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
UNION
SELECT L2Reportees.pid AS directReportees, count(L2Reportees.directly_manages) AS count
  FROM person_reportee manager
 JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
) AS T
GROUP BY directReportees)
```

```
SELECT depth1Reportees.pid AS directReportees,
count(depth2Reportees.directly_manages) AS count
  FROM person_reportee manager
 JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
) AS T
GROUP BY directReportees)
UNION
(SELECT T.directReportees AS directReportees, sum(T.count) AS count
FROM (
SELECT reportee.directly_manages AS directReportees, 0 AS count
  FROM person_reportee manager
 JOIN person_reportee reportee
ON manager.directly_manages = reportee.pid
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
UNION
SELECT L2Reportees.pid AS directReportees, count(L2Reportees.directly_manages) AS count
  FROM person_reportee manager
 JOIN person_reportee L1Reportees
ON manager.directly_manages = L1Reportees.pid
JOIN person_reportee L2Reportees
ON L1Reportees.directly_manages = L2Reportees.pid
 WHERE manager.pid = (SELECT id FROM person WHERE name = "fName lName")
 GROUP BY directReportees
) AS T
GROUP BY directReportees)
```

# The Same Query using Cypher

```
MATCH (boss)-[:MANAGES*0..3]->(sub),
      (sub)-[:MANAGES*1..3]->(report)
WHERE boss.name = "John Doe"
RETURN sub.name AS Subordinate,
       count(report) AS Total
```

## Project Impact

### Less time writing queries

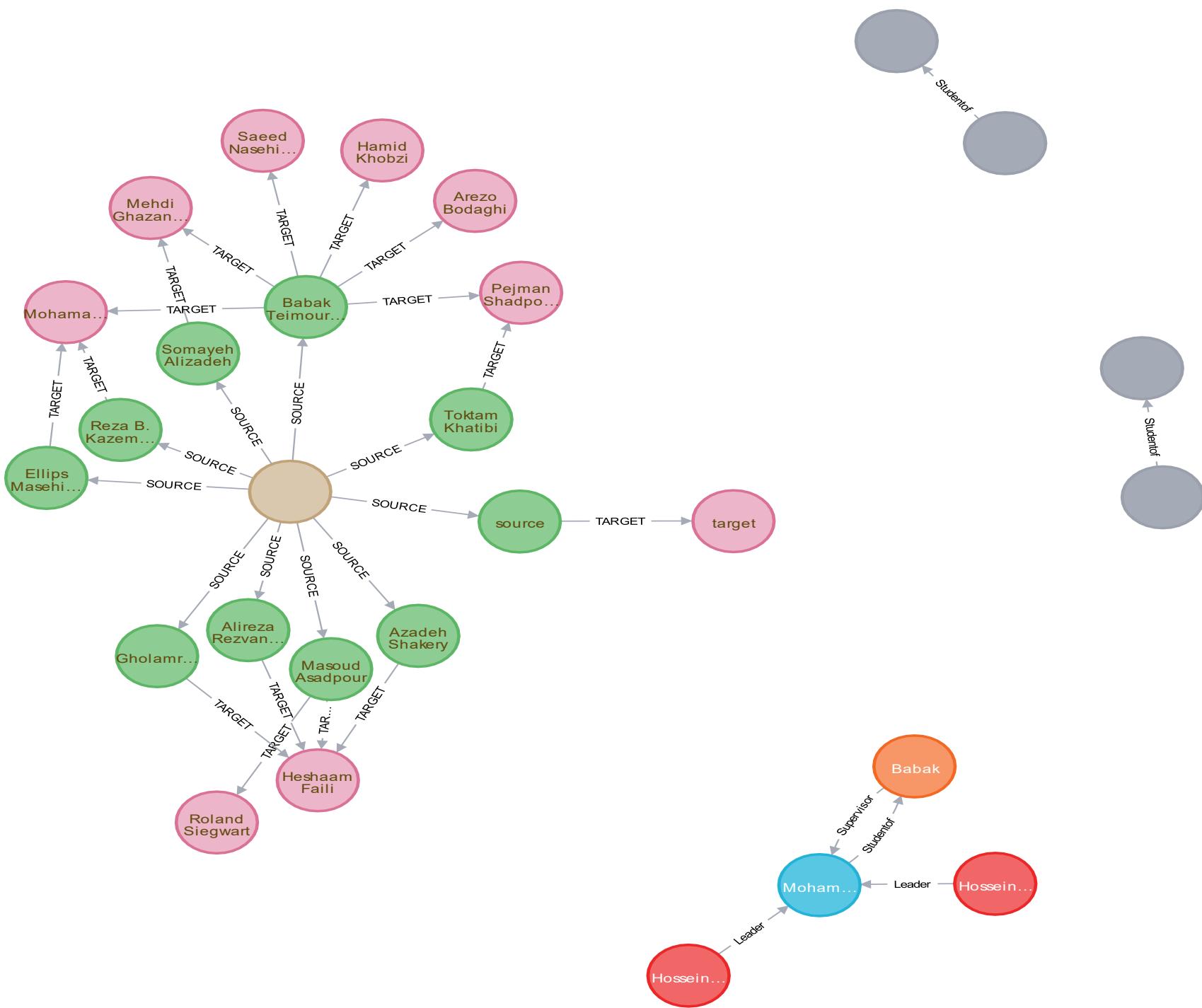
- More time understanding the answers
- Leaving time to ask the next question

### Less time debugging queries:

- More time writing the next piece of code
- Improved quality of overall code base

### Code that's easier to read:

- Faster ramp-up for new project members
- Improved maintainability & troubleshooting





ArangoDB

TITAN



## سایر دیتابیس های گرافی

FlockDB ■

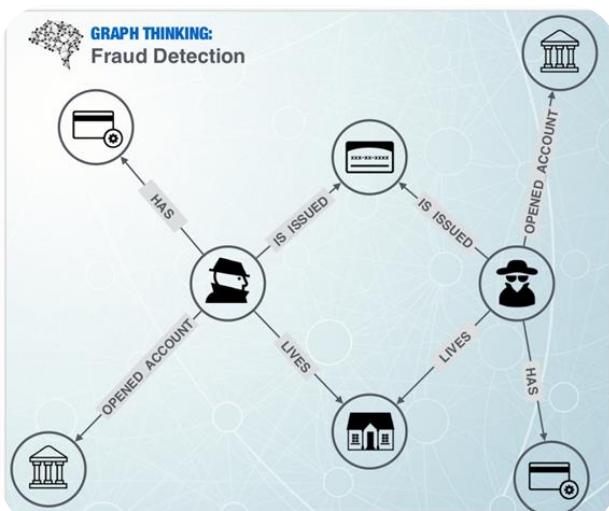
▪ توسط توئیتر و برای تحلیل ارتباطات طراحی و ساخته شده است.

AllegroGraph ■

▪ طراحی شده است تا با Semantic Web و Linked Data کار کند.

InfiniteGraph ■

▪ هدف ساخت یک گراف دیتابیس با مقیاس پذیری نامحدود می باشد

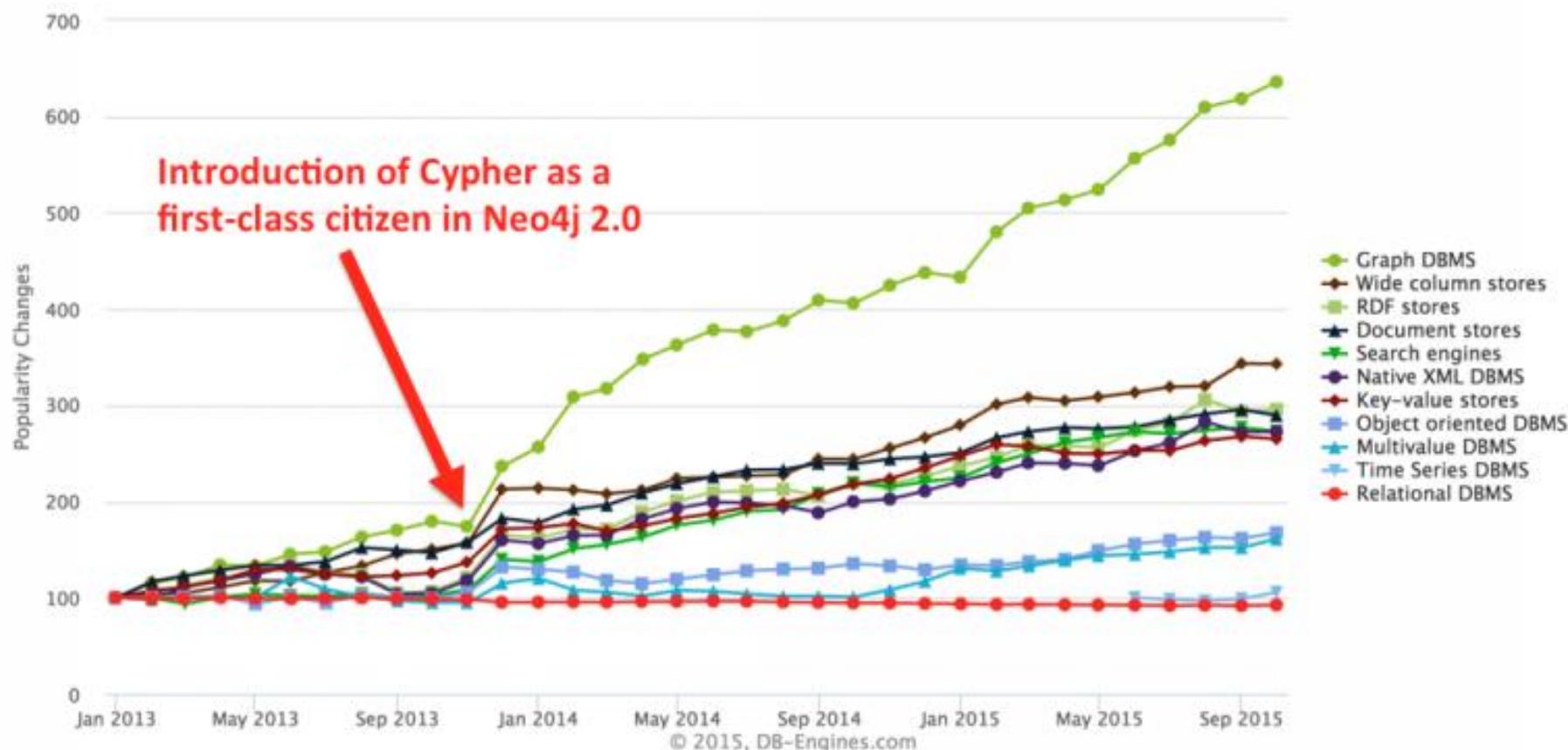


# گراف یعنی آینده

- چالش دیتابیس های رابطه ای
- تعداد Join زیاد بین جداول علاوه بر پیچیدگی بسیار کدنویسی، موجب کند شدن فراخوانی داده ها میشود.
- Neo4j، برای ذخیره داده ها و ارتباط میان آنها از مدل داده ای گراف استفاده میکند.
- در مقایسه با دیتابیس های رابطه ای دیگر نیازی به نوشتن join های پیچیده نیست.
- براحتی با پیمایش و حرکت در گراف میتوان انواع Query های متنوع را انجام داد.
- استفاده از گراف ها بجای جداول در Neo4j علاوه بر کدنویسی کمتر و ساده تر سبب میشود سیستمی کاملا چابک و منعطف برای ذخیره انواع داده ها و روابط داشته باشیم.

گراف یعنی آینده

### Popularity changes per category, October 2015



# Document DB

- هر سطر یک سند است که شامل مجموعه ای از کلید مقادیر است.
- مدل داده ای: مجموعه ای از کلید مقادیرها  
MongoDB, CouchDB ▪
- مناسب برای داده هایی که به نُدرت تغییر می کند
- مناسب برای داده هایی که ورژن بندی در آنها اهمیت دارد

```
{  
  _id:'147963658',  
  Name:'abc',  
  Contact:{  
    Phone:'8984577',  
    Email:'test@test.com'  
  },  
  Address:{  
    address:'Fanavar Street',  
    City:'Tehran'  
  }  
}
```



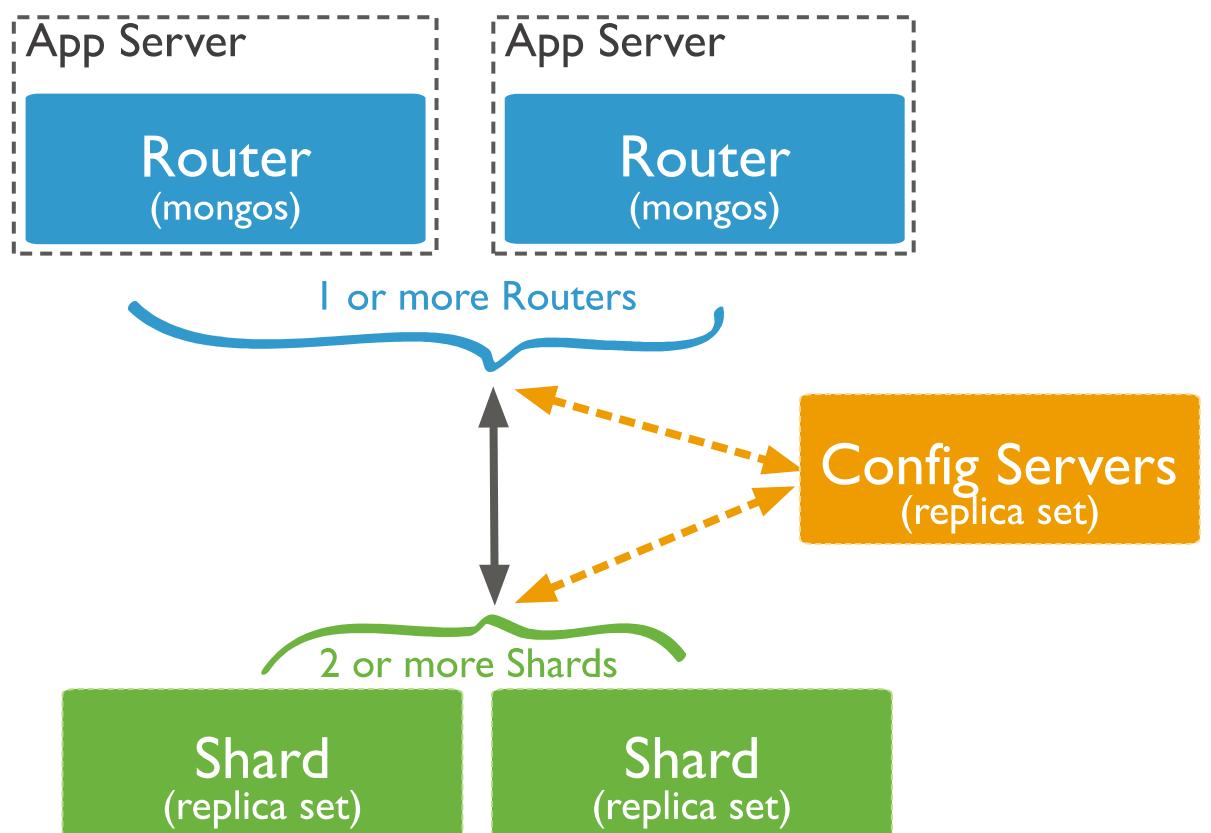


mongoDB

# مانگو دی بی - MongoDB

، یک روش برای توزیع داده‌ها بر روی نودهای مختلف توسط دیتابیس Mongo است. ■

```
{  
  _id:147963658,  
  Name:'abc',  
  Contact:{  
    Phone:'8984577',  
    Email:'test@test.com'  
  },  
  Address:{  
    address:'Fanavarjan Street',  
    City:'Tehran'  
},  
}
```





## In Memory DB

- برای نگهداری پایگاه داده، حافظه کش و واسط پیام استفاده می‌شود.
- ردیس داده‌ها را شکل **Key-value in RAM** نگهداری می‌کند پس، دسترسی و بازیابی این اطلاعات بسیار ساده‌تر خواهد شد
- سرعت و کارایی فوق العاده پاسخ دهی
- میتواند هم در حافظه اصلی و هم در حافظه جانبی ذخیره شود.
- مزایا
- ماندگاری: اطلاعات شما در اثر ریستارت سرور از بین نمی‌رود! Persistence
- پشتیبانی از رپلیکیشن: ردیس به خوبی از رپلیکیشن پشتیبانی می‌کند
- میتوان سرورهای مختلفی را به صورت Master/Slave یا سناریوهای دلخواه خود اجرا کنید.
- چالش
- حافظه **RAM** در سرورهای ابری معیار اصلی قیمت‌گذاری است. پردازنده و هارد دیسک در معیارهای بعد اند.



## In Memory DB

- سرعت و کارایی فوق العاده در پاسخ دهی
- میتواند هم در حافظه اصلی و هم در حافظه جانبی ذخیره شود.
- مزایا

■ **Persistence**: اطلاعات شما در اثر Restart سرور از بین نمی رود!

- پشتیبانی از رپلیکیشن : ردیس به خوبی از رپلیکیشن پشتیبانی میکند
- میتوان سرورهای مختلفی را به صورت Master/Slave یا سناریوهای دلخواه خود اجرا کنید.

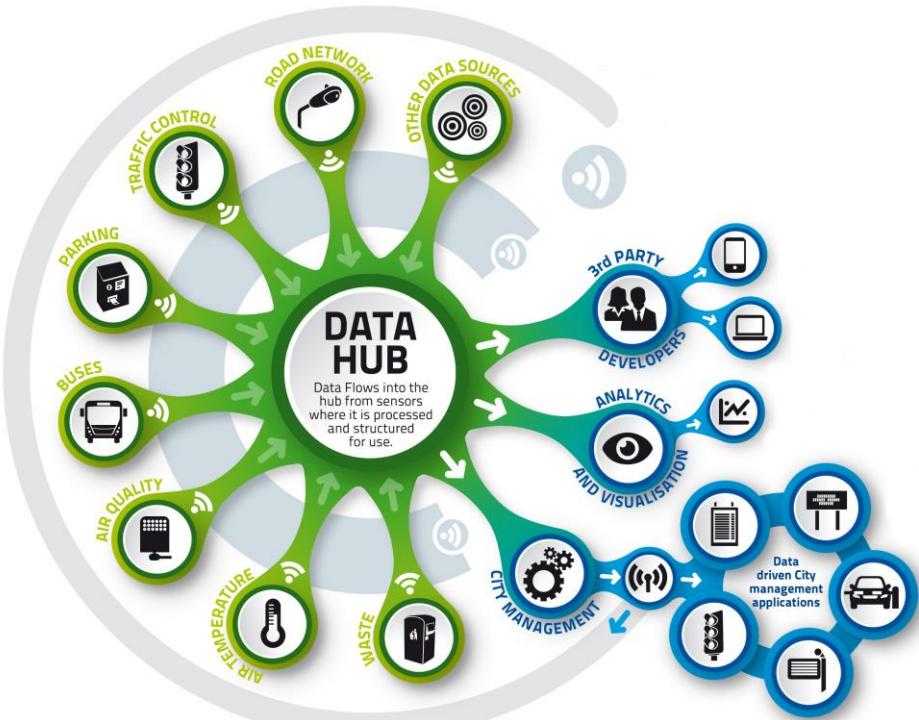


## In Memory DB

- تفاوت عمده بین Redis و دیگر سیستم‌های پایگاه ذخیره سازی داده‌ها
- نه تنها مقدار رشته را می‌پذیرد بلکه مقادیر داده‌ای زیر را نیز پشتیبانی می‌کند(نقطه قوت)
  - لیستی از رشته‌ها Lists
  - مجموعه‌ای از رشته‌ها Sets
- جداول Hash Tables که مقادیر Key,Value آنها رشته می‌باشد.
- HyperLogLogs که برای تخمین زدن و برآورد تقریبی کاردینالیتی به کار می‌رود.
- به صورت پیش فرض replication دارد دارای partitioning از طریق Redis Cluster هست.
- آیا نگه داری داده‌ها در RAM باعث از دست رفتن آنها نمی‌شود؟
- خیر، Redis برای نگه داری دائمی داده‌ها آنها را به دیسک اصلی سیستم منتقل می‌کند.  
**on-disk persistence**

# مهم است بدانیم

- مهاجرت به سمت راه کار NoSQL به دلیل محدودیت زبان SQL نبوده است.
- بلکه دلیل آن محدودیت های مدل رابطه ای در طراحی پایگاه داده بوده است.
- مدل رابطه ای، کاد، 1970



# مقایسه دیتابیس های NoSQL

کارکرد	پیچیدگی	انعطاف پذیری	مقیاس پذیری	عملکرد	مدل داده
متغیر	فاقد	بالا	بالا	بالا	کلید - مقدار
حداقل	پایین	متوسط	بالا	بالا	ستون محور
متغیر	پایین	بالا	متغیر(بالا)	بالا	سنده محور
تئوری گراف	بالا	بالا	متغیر	متغیر	گرافی
جبر رابطه ای	متوسط	پایین	متغیر	متغیر	رابطه ای

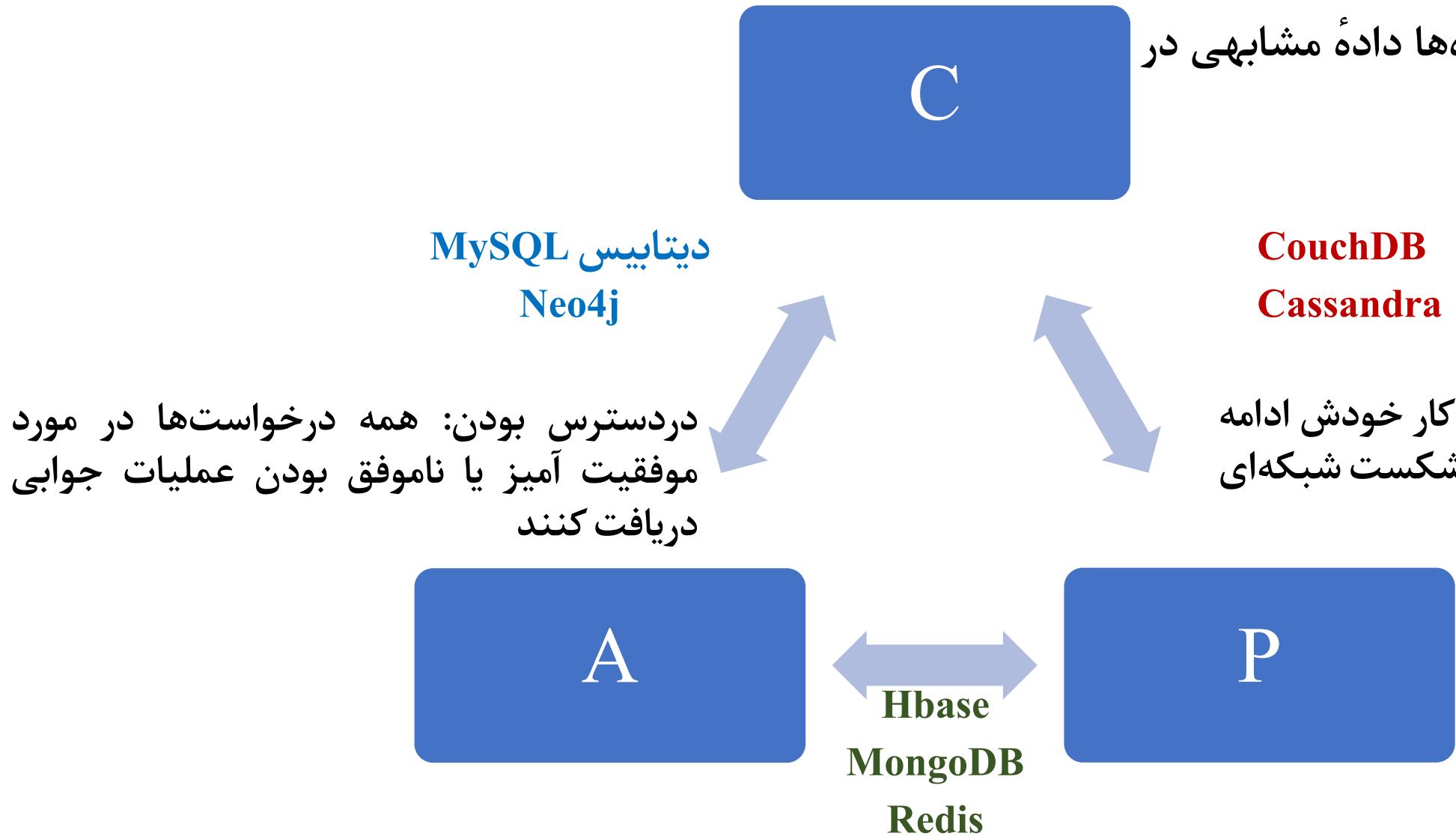
Rack Awareness	تراکنش ها	دردسترس بودن	رپلیکیت	دوام	نوع	پروژه
بله	بله	توزيع شده	بله	بله	کلید مقدار	Cassandra
خیر	بله	بله	بله	بله	کلید مقدار	Oracle No Sql
بله	بله	بله	بله	بله	کلید مقدار	HBase
خیر	Parital	Fail-Over	بله	بله	سندگرا	MongoDB
خیر	بله	بله	بله	بله	گراف	Neo4j
خیر	بله	بله	بله	بله	کلید مقدار	Redis
بله	بله	بله	بله	بله	چند مدل گرافی سندگرا کلید مقدار	OrientDB

# جدیدترین آمار رتبه بندی دیتابیس ها در دنیا (اکتبر 2019)

355 systems in ranking, October 2019

Rank			DBMS	Database Model	Score		
Oct 2019	Sep 2019	Oct 2018			Oct 2019	Sep 2019	Oct 2018
1.	1.	1.	Oracle 	Relational, Multi-model 	1355.88	+9.22	+36.61
2.	2.	2.	MySQL 	Relational, Multi-model 	1283.06	+3.99	+104.94
3.	3.	3.	Microsoft SQL Server 	Relational, Multi-model 	1094.72	+9.66	+36.39
4.	4.	4.	PostgreSQL 	Relational, Multi-model 	483.91	+1.66	+64.52
5.	5.	5.	MongoDB 	Document 	412.09	+2.03	+48.90
6.	6.	6.	IBM Db2 	Relational, Multi-model 	170.77	-0.79	-8.91
7.	7.	↑ 8.	Elasticsearch 	Search engine, Multi-model 	150.17	+0.90	+7.85
8.	8.	↓ 7.	Redis 	Key-value, Multi-model 	142.91	+1.01	-2.38
9.	9.	9.	Microsoft Access	Relational	131.18	-1.53	-5.62
10.	10.	10.	Cassandra 	Wide column 	123.22	-0.18	-0.17

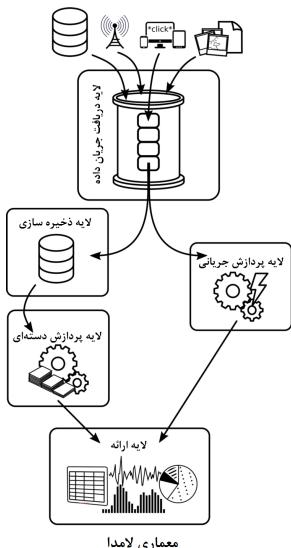
# نتئی CAP، بیان می‌کند که برای یک سامانهٔ رایانه‌ای توزیع شده امکان‌پذیر نیست که هر سه این موارد را همزمان فراهم کند:



پردازش داده های جریانی

# الگوی معماری لامدا

- در تلاش برای ترکیب کردن هر دو حالت (پردازش دسته‌ای و جریانی)، الگوی معماری با نام معماری لامدا ارائه شده است.
- برای رفع مشکل کندي پردازش‌های دسته‌ای، لایه پردازش داده‌های جریانی را به عنوان مکمل، به آن اضافه کرده است.
- درنتیجه هم بعد حجم و هم بعد سرعت در چالش‌های کلان‌داده را به صورت همزمان مورد هدف قرار داده است.

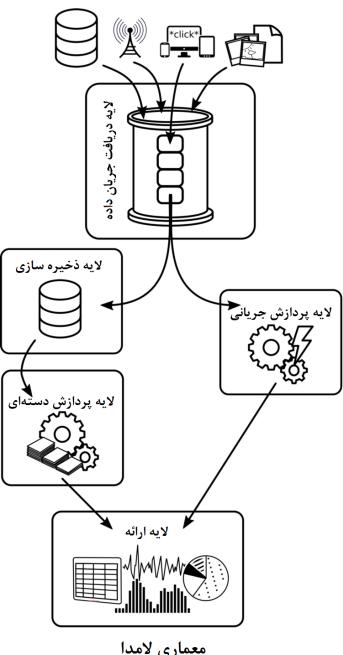


Volume ■  
Velocity ■

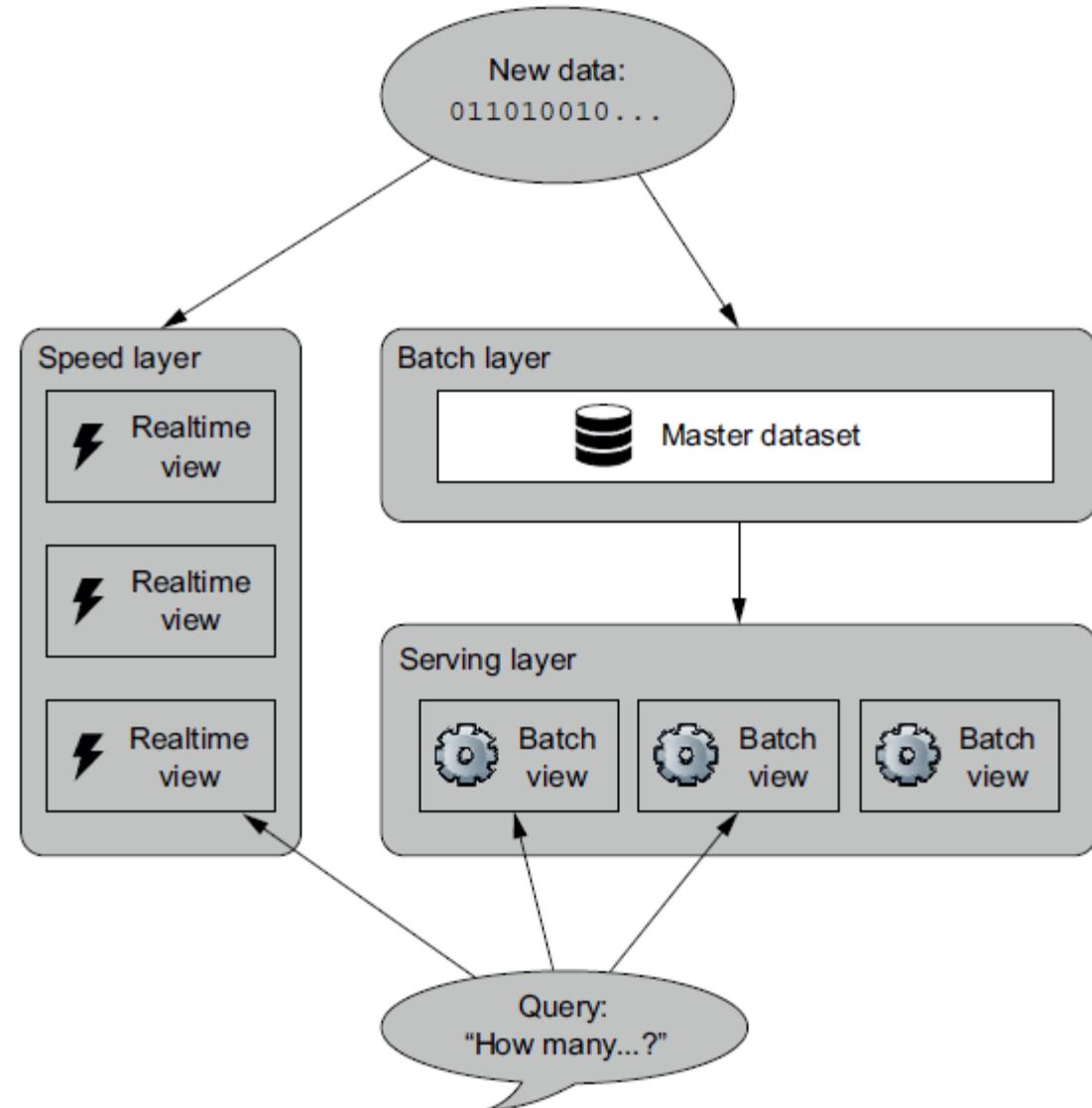
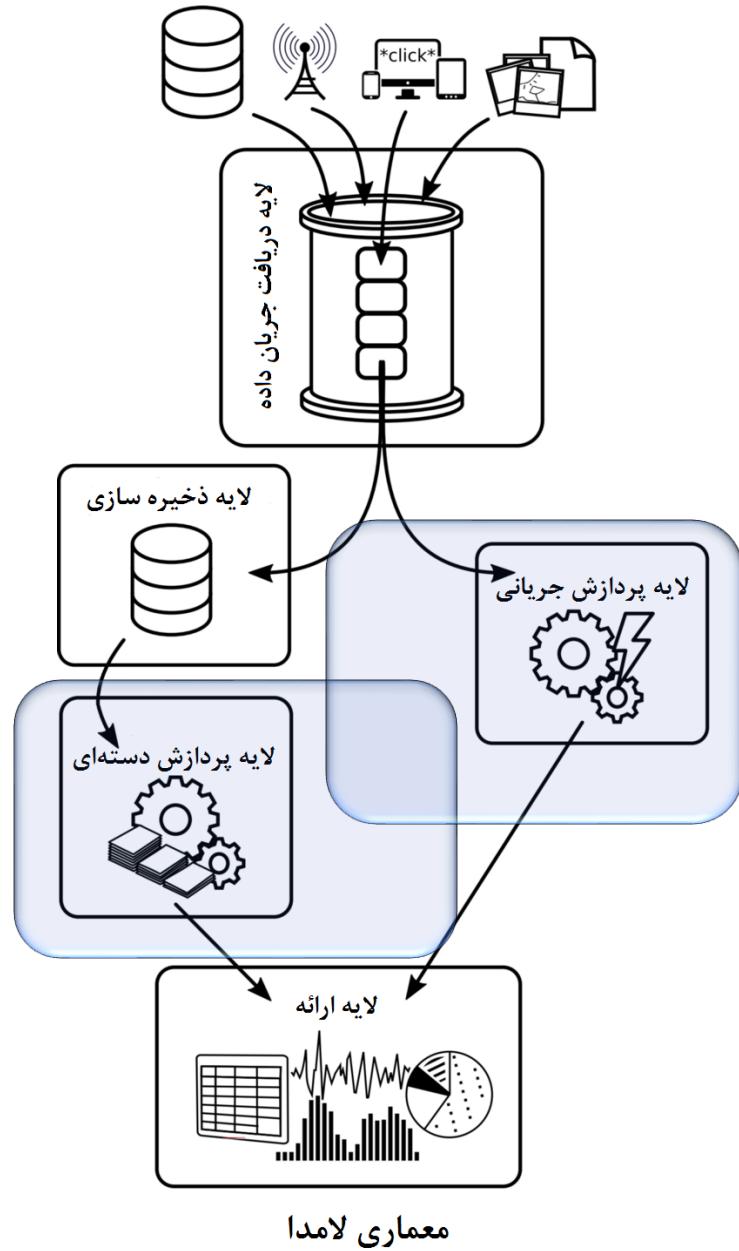
# الگوی معماری لامدا

- داده‌ها در لایه‌ای ذخیره‌سازی در ابزاری مانند HDFS ذخیره می‌شود.
- از آنجا متناوباً به لایه پردازش دسته‌ای انتقال و پردازش می‌شوند.
- لایه پردازش جریانی نیز به بخشی از داده که توسط لایه دسته‌ای پردازش نشده می‌پردازد.
- لایه ارائه خروجی‌های لایه دسته‌ای و لایه پردازش جریانی را ادغام می‌کند.

Combination two layers ▪



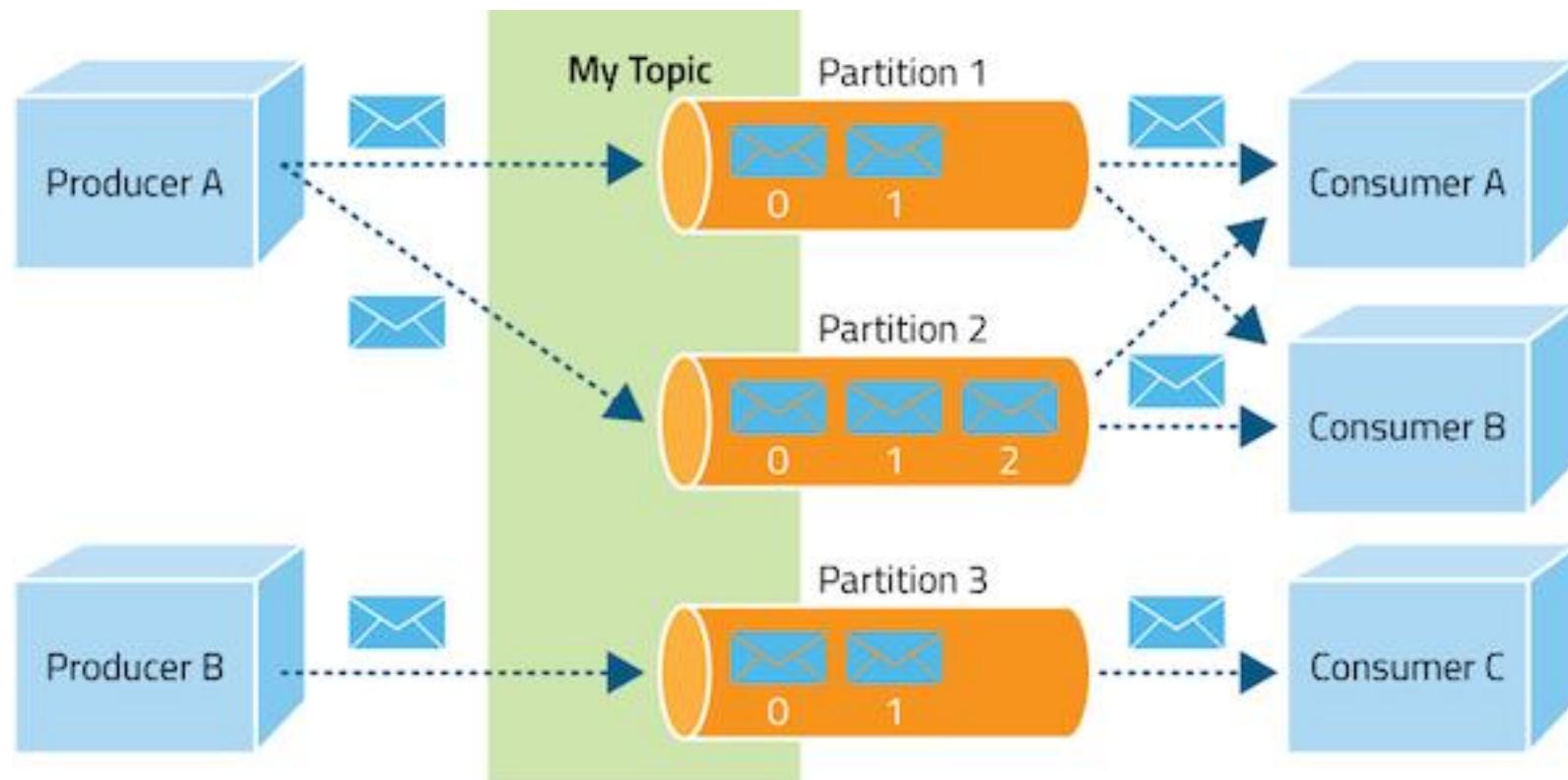
# الگوی معماری لاما



# مقایسه پردازشگرهای جریانی



# آپاچی کافکا Apache Kafka

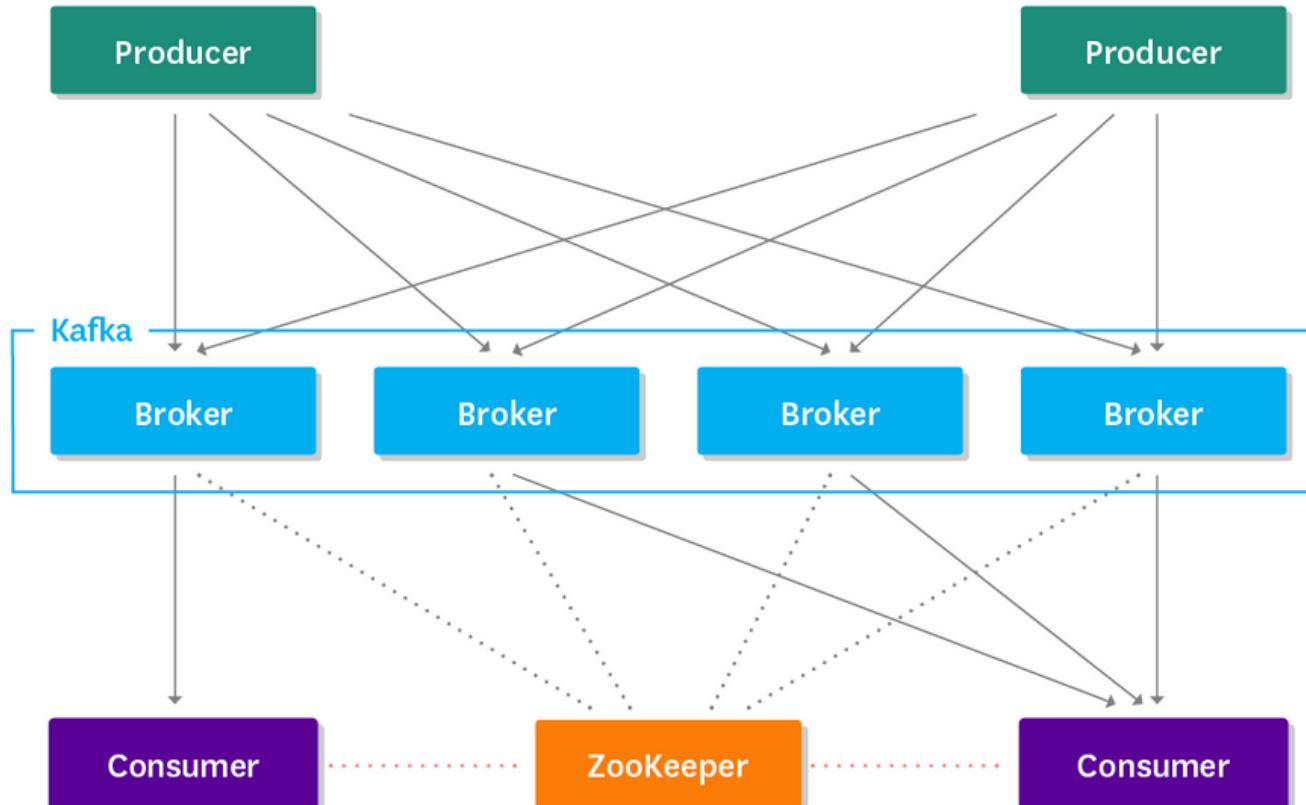


# آپاچی کافکا Apache Kafka

- کافکا با در نظر داشتن اهداف زیر ساخته شده است
- اتصال سست (حدائق وابستگی) بین مصرف کننده‌ها و تولید کنندگان پیام
- ماندگاری داده‌های پیام به منظور پشتیبانی از انواع مختلف مسائل مصرف داده و مدیریت شکست
- حداقل توان پردازشی بین دو سیستم استفاده کننده از کافکا با کمترین تأخیر در اجزا
- مدیریت انواع داده‌ها و فرمت داده‌های متنوع با استفاده از فرمت داده‌های دودویی
- مقیاس‌پذیری خطی سرورها بدون اثرگذاری بر تنظیمات کلاسترها موجود

# آپاچی کافکا Apache Kafka معماری فیزیکی (تصویری از کلاستر چند گره کافکا)

- یک کلاستر کافکا اساساً از یک یا چند سرور (گره) تشکیل شده است.



# ZooKeeper (آقای نگهبان)

- کافکا بدون زوکیپر نمی‌تواند کار کند.
- زوکیپر، جزئی مهم از کلاستر کافکا است، که حالت کارگزارها و مصرف‌کنندگان کافکا را مدیریت و هماهنگ می‌کند.
- زوکیپر دائماً افزوده شدن یا شکست و از کلاستر خارج شدن کارگزارهای موجود در کلاستر کافکا را پیگیری می‌کند.
- در نتیجه، تولیدکننده‌ها یا مصرف‌کنندگان تاپیک‌های کافکا را از حالت کلاستر مطلع می‌کند. این مسئله، به تولیدکننده‌ها و مصرف‌کنندگان اجازه می‌دهد که هماهنگ با کارگزارها فعال عمل کنند.
- زوکیپر همچنین اینکه کدام کارگزار مدیر کدام پارتیشن یک تاپیک است را نیز ثبت کرده و این اطلاعات را، هنگام نوشتن و خواندن پیام‌ها، به تولیدکننده‌ها یا مصرف‌کننده‌ها ارائه می‌دهد.

## آپاچی استورم

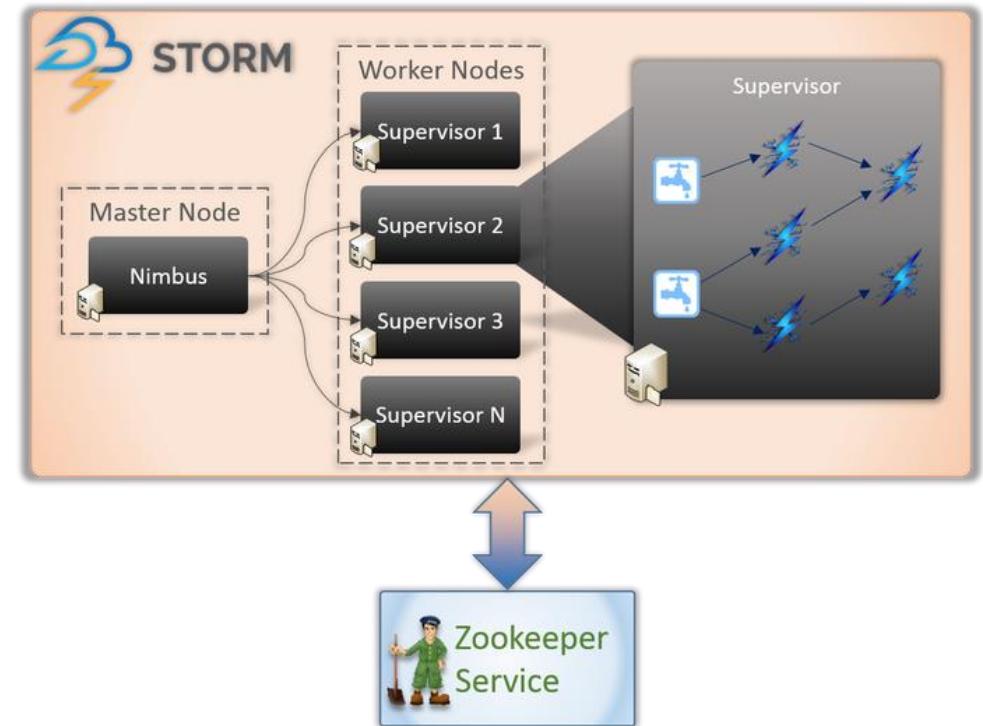
- استورم یک سیستم محاسباتی Real Time و دارای تحمل‌پذیری خطا برای پردازش جریان‌های داده است.
- این سیستم برخلاف هادوپ که برای پردازش دسته‌ای طراحی شده، به منظور انجام پردازش Real Time ساخته شده است.
- راه اندازی و اجرای آن آسان است.
- مقیاس‌پذیر و دارای تحمل‌پذیری در برابر خطا به منظور فراهم کردن کارایی رقابتی است
- کاربران خوش استورم، توپولوژی‌های گوناگونی را برای وظایف متنوع اجرا می‌کنند.



Apache Storm, in simple terms, is a distributed framework for real time processing of Big Data like Apache Hadoop is a distributed framework for batch processing. Apache Storm works on task parallelism principle where in the same code is executed on multiple nodes with different input data.

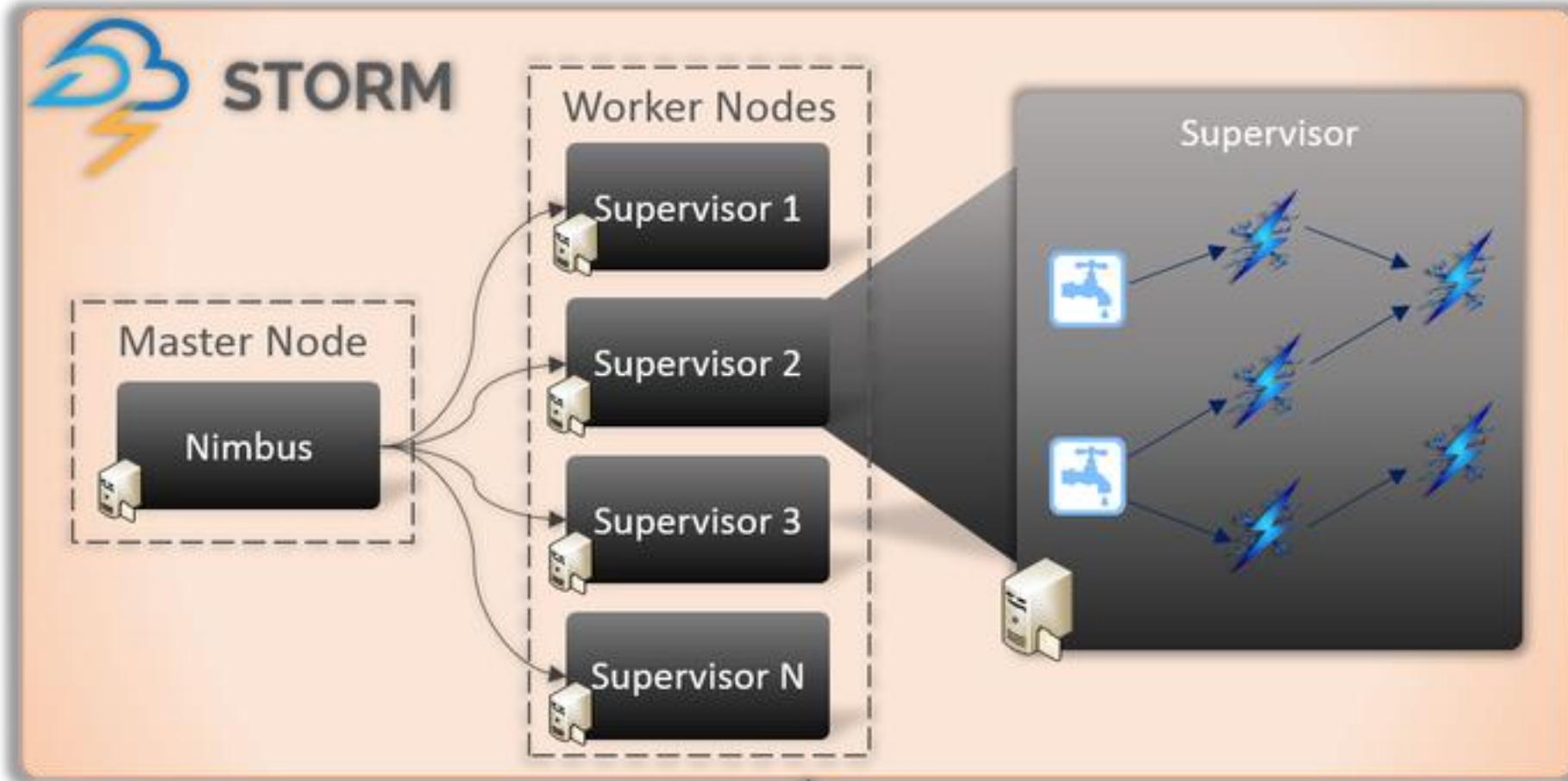
Apache Storm does not have any state managing capabilities. It instead utilizes Apache ZooKeeper to manage its cluster state such as message acknowledgements, processing status etc. This enables Storm to start right from where it left even after the restart.

Since Storm's master node (called Nimbus) is a Thrift service, one can create and submit processing logic graph (called topology) in any programming language. Moreover, It is scalable, fault-tolerant and guarantees that input data will be processed.

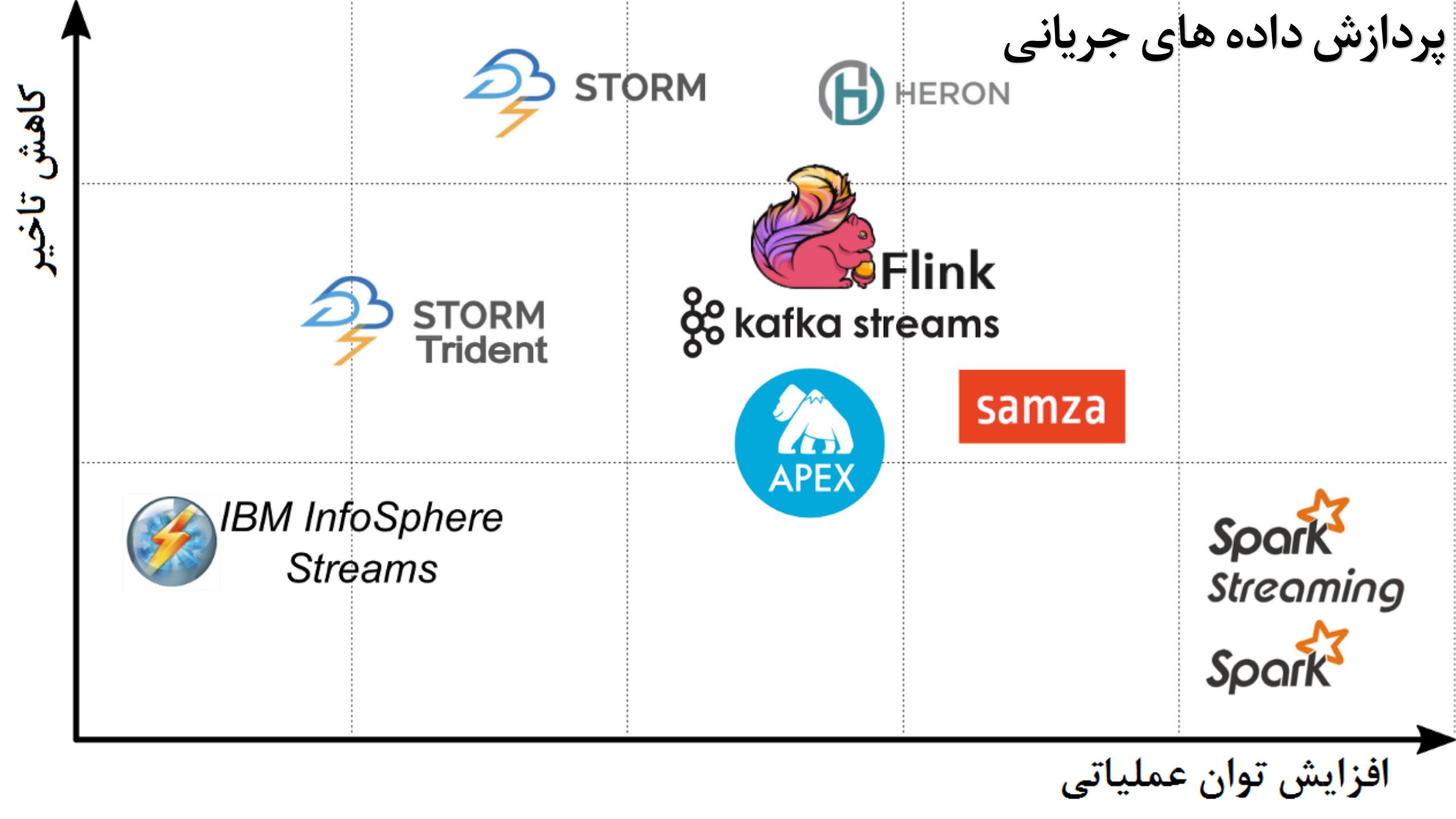




# STORM



# پردازش داده های جریانی



# مقایسهٔ مستقیم موتورهای Flink Streaminig و Spark Streaminig، Samza، Trident، Storm

*	Storm	Trident	Samza	Spark	Flink
تضمين	حداقل یک بار	دقیقاً یک بار	حداقل یک بار	دقیقاً یک بار	دقیقاً یک بار
تأخير قابل حصول	<100 ms	<100 ms	<100 ms	< 1 s	<100 ms
مدیریت حالت	بله	بله (حالت کوچک)	بله	بله	بله
مدل پردازشی	یک آیتم داده در زمان	میکرو دسته	یک آیتم داده در زمان	میکرو دسته	یک آیتم داده در زمان
backpressure	بله	بله	لزومی ندارد (استفاده از بافر)	بله	بله
تضمين ترتيب آیتم‌های داده	خیر	بین دسته‌ها	داخل پارتیشن‌های جريان	بین دسته‌ها	داخل پارتیشن‌های جريان
خاصیت کشسانی	بله	بله	خیر	بله	خیر

زبان تعریف گردش جریان داده

# آپاچی پیگ Apache Pig

- پیگ یک زبان گردش جریان داده است که به تحلیل گران داده کمک می کند بدون پرداختن به جزئیات برنامه نویسی، بروی منطق برنامه خود تمرکز کنند.
- کدهای نوشته شده به زبان پیگ در نهایت توسط کامپایلر به کدهای نگاشت/کاهش تبدیل شده و بروی کلاستر هدوف به اجرا در می آیند.
- سکویی به منظور ساخت جریان دادهها برای فرآیند ETL، پردازش و تجزیه و تحلیل کلان داده است.



# نحوه‌ی کار پیگ چگونه است؟

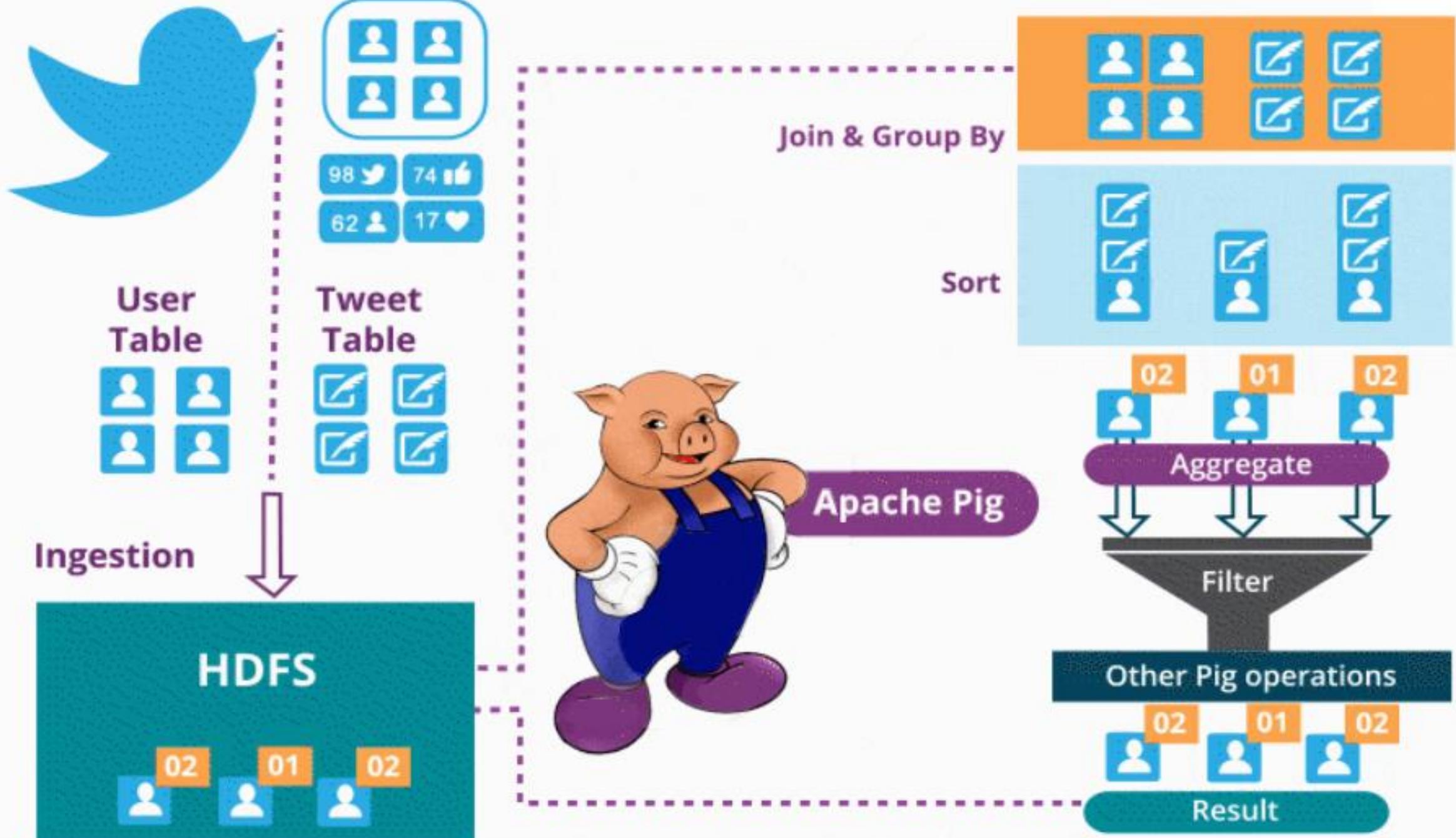
- نخست داده‌ها در PIG با دستور **load**، بارگذاری می‌شوند،
- سپس وظایف مختلفی بر روی آن از قبیل گروه‌بندی، فیلتر کردن، متصل کردن، مرتب‌سازی انجام می‌دهیم.
- در نهایت، هم می‌توانید داده‌ها را بر روی صفحه‌نمایش کپی کنید و هم می‌توانید نتایج به‌دست آمده را در HDFS ذخیره نمایید.



## کجا از آپاچی پیگ استفاده کنیم؟

- مواقعي که به پردازش مجموعه داده های کلان مانند و بلاگ ها، استریم کردن داده های آنلайн، نیاز داریم.
- در جاهایی که ما نیاز به پردازش داده ها برای جستجو داریم.
- برای مثال: یاهو برای ۴۰ درصد از فعالیت های خود از جمله اخبار و موتور جستجو از پیگ استفاده می کند.
- هنگامی که نیاز به پردازش داده های حساس به زمان را داریم.
- برخی از الگوریتم های یادگیری ماشین نیاز به داده های حساس به زمان دارند.
- برای مثال توبیتر که نیازمند استخراج داده ها از فعالیت های کاربران و تجزیه و تحلیل داده ها برای پیدا کردن الگوهای رفتاری کاربر و کاوش دانش در این زمینه نظیر توییت های ترند شده می باشد.







## Twitter Database

User Table		
Id	Name	...
1	Jay	...
2	Eilic	...
3	Sam	...

Tweet Table		
User Id	Tweet	...
1	xyz	...
2	abc	...
1	pqr	...
3	stu	...
1	lmn	...
2	vxy	...

IMPORT

①

HDFS

User Table		
Id	Name	...
1	Jay	...
2	Eilic	...
3	Sam	...

Tweet Table		
User Id	Tweet	...
1	xyz	...
2	abc	...
1	pqr	...
3	stu	...
1	lmn	...
2	vxy	...

LOAD

②

## User Table

Id	Name	...
1	Jay	...
2	Eilic	...
3	Sam	...

STORE

⑥



## Apache Pig

③

JOIN + GROUP

Tweet Table		
User Id	Tweet	...
1	xyz	...
2	abc	...
1	pqr	...
3	stu	...
1	lmn	...
2	vxy	...

user\_group=COGROUP  
tweet\_table by 'user Id',  
user\_table by 'Id'

④

COUNT /  
AGGREGATE

Id	Name	Tweets	...
1	Jay	xyz	...
1	Jay	pqr	...
1	Jay	lmn	...
2	Ellie	abc	...
2	Eilie	vxy	...
3	Sam	stu	...

BAG

Id	Count
1	3
2	2
3	1

Join Count 'Id'  
user\_table by 'Id'  
GENERATE count::Id,  
user\_table:: names  
Count :: count

⑤

Id	Name	Count
1	Jay	3
2	Eilic	2
3	Sam	1

# انبارداده ها و ابزارهای تحلیلی

## مبنی بر SQL



## آپاچی هایو - Apache Hive

- فیسبوک، Hive را برای آن دسته از افرادی طراحی کرده که می‌توانند به راحتی با SQL کار کنند.
- در نتیجه، شما با تجربه‌ای نظیر کار با دستورات SQL مشغول کار کردن در اکوسیستم هدوب هستید.
- در واقع، Hive یک انبار داده در اکوسیستم هدوب است که مسئولیت خواندن، نوشتن و مدیریت کلان داده را در یک محیط توزیع شده و با استفاده از واسطی مانند SQL، برعهده دارد.
- همه نوع دیتابایپ‌های SQL را پشتیبانی می‌کند.
- از تابع‌های نوشته شده توسط کاربر و یا توابع از پیش تعین شده، استفاده می‌توان کرد.
- زبان پرس‌وجوی Hive Query Language، HQL نامیده می‌شود، که بسیار مشابه زبان SQL است.

# آپاچی هایو - Apache Hive

- **Hive+ SQL= HQL** ■
- این زبان از دو جزء اصلی تشکیل شده است
  - **Hive Command Line** ■
  - **JDBC/ODBC Driver** ■
- رابط خط فرمان Hive، که برای اجرای دستورات HQL مورد استفاده قرار می‌گیرد.
- در حالی که درایور پایگاه داده‌ی جاوا JDBC و درایور پایگاه داده‌ی اشیاء ODBC، به منظور ایجاد اتصالی از ذخیره‌سازی داده استفاده می‌شود
- Hive به توسعه زیرساختهای انباره داده‌ها کمک می‌کند و مرکزی است برای گزارش کردن نیازهای Facebook

# در فیس بوک

- تمام مسائل تحلیلی، رابط کاربری برنامه نویسان، داده های تحقیقاتی، محصولات مدیریتی و همچنین تبلیغات برای اشخاصی که قصد دارند کسب و کار خود را در فیس بوک دنبال کنند به وسیله Hadoop، Hive و Hbase صورت می گیرد.
- به گفته مدیر زیرساخت فیس بوک Jay Parikh
- فیسبوک باید بین نیازهای خود تعادلی به وجود آورد تا با سرعت بالا بتواند به نتایج مطلوب از قبیل ابزارهای گراف و محیطی برای هرچه آسان تر کردن گزارش گیری بپردازد، پس Hive را انتخاب کرد تا بتواند به Query های خودش سرعت دهد.

# آپاچی دریل - Apache Drill

- آپاچی دریل، دیگر سیستم توزیع شده برای تحلیل‌های تعاملی بیگ دیتا است.
- این روش دارای انعطاف‌پذیری بیشتری برای پشتیبانی از بسیاری زبان‌های کوئری، فرمات‌های داده و منابع داده است.
- همچنین، این سیستم به طور ویژه برای بهره‌برداری از داده‌های تو در تو طراحی شده است و دارای این هدف است که روی ۱۰۰۰ سرور یا تعداد بیشتری مقیاس بپذیرد و به ظرفیت لازم برای پردازش پتابایت‌ها داده و تریلیون‌ها رکورد در چند ثانیه برسد.
- دریل از HDFS برای ذخیره‌سازی و نگاشت کاهش برای انجام تحلیل دسته‌ای استفاده می‌کند.



کتابخانه های پادگیری ماشین

# آپاچی ماہوت

- آپاچی ماہوت قصد فراهم کردن روش‌های یادگیری ماشین مقیاس‌پذیر و تجاری برای نرم‌افزارهای تحلیل داده هوشمند و بزرگ مقیاس را دارد.
- الگوریتم‌های اصلی ماہوت شامل خوشبندی، دسته‌بندی، کاوش الگو، رگرسیون، کاهش ابعاد، الگوریتم‌های تکاملی و فیلتر مشارکتی است.
- در هسته آن جبر خطی و عملیات آماری همراه با ساختار داده ای وجود دارد تا کار با آن در عین قدرتمندی آسانتر شود.
- دارای یک Shell تعاملی می‌باشد که قادر است عملیات توزیع شده را روی کلاستر Spark اجرا کند.
- مزیت: آزادی عمل بیشتر کاربران در اجرای الگوریتم‌ها





## Mahout Learning Applications

Clustering

Classification

Recommendations

Mahout Utilities

Apache  
Lucene

Mahout Math

Apache Spark

**H<sub>2</sub>O**

Apache  
Hadoop

Data processing

Map Reduce



Yarn

Data Storage  
Hadoop Distributed File System (HDFS)



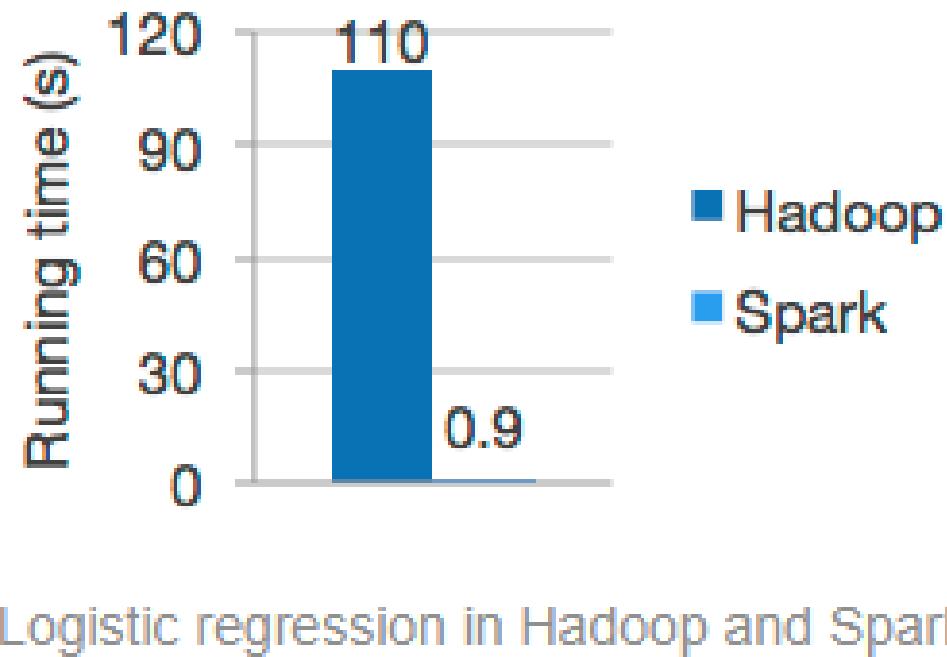
## Spark MLlib

- MLlib انواع مختلفی از الگوریتم‌های یادگیری ماشین از جمله
  - دسته‌بندی،
  - رگرسیون،
  - خوش‌بندی
  - و پالایش گروهی را ارائه می‌دهد
- همچنین از قابلیت‌های مثل ارزیابی مدل و ورود داده‌ها پشتیبانی می‌کند.
- ساختارهای سطح پایین یادگیری ماشین مثل الگوریتم بهینه‌سازی گرادیان نزولی را فراهم می‌آورد

```
data = spark.read.format("libsvm")\
    .load("hdfs://...")  
  
model = KMeans(k=10).fit(data)
```

Calling MLlib in Python

# مقایسه زمان اجرای رگرسیون لجستیک در MLlib و Mahout



# MLlib و Mahout مقایسه

	Mahout	MLlib
Regression	N/A	<b>Linear Regression , Isotonic Regression , Survival Analysis</b>
Classification	Logistic Regression, Naïve Bayes, Random Forest, <b>Hidden Markov Models</b> , Multilayer Perceptron	Logistic Regression, Naïve Bayes, <b>linear Support Vector Machine , Decision Tree</b> , Random Forest, Multilayer Perceptron
Clustering	K-Means, Spectral Clustering	K-Means, Spectral Clustering, <b>Gaussian Mixtures</b>
Dimension Reduction	Singular Value Decomposition, Principal Component Analysis, QR Decomposition	Singular Value Decomposition, Principal Component Analysis, QR Decomposition, <b>Elastic Net</b>
Text Mining	Latent Dirichlet Allocation, TF-IDF, <b>Collocations</b>	Latent Dirichlet Allocation, TF-IDF, <b>Word2Vec, Tokenization</b>
Recommendation	Alternating Least Squares	Alternating Least Squares, <b>Association Rule Mining, FP-Growth</b>

# MLlib و Mahout مقایسه

	<b>Mahout</b>	<b>MLlib</b>
<b>Input</b>	<ul style="list-style-type: none"> <li>-Text files</li> <li><b>-Lucene/Solr</b></li> <li>-Relational Databases (MySQL, SQL Server, Oracle)</li> <li>-Hadoop (HDFS, Cassandra, Hbase, <b>MongoDB</b>)</li> </ul>	<ul style="list-style-type: none"> <li>-Text files (Local, Remote); <b>JSON</b></li> <li>-Relational Databases (MySQL, SQL Server, Oracle)</li> <li>-Hadoop (<b>HDFS, Parquet, Cassandra, Hbase, Hive, Amazon S3</b>)</li> </ul>
<b>Output</b>	<ul style="list-style-type: none"> <li><b>-Trained Model in Mahout Format</b></li> <li>-Evaluation Metrics</li> <li>-Text Files</li> </ul>	<ul style="list-style-type: none"> <li><b>-Predictive Model Markup Language</b></li> <li>-Evaluation Metrics</li> <li>-Text files (Local, Remote); <b>JSON</b></li> <li><b>Relational Databases (MySQL, SQL Server, Oracle)</b></li> <li>-Hadoop (<b>HDFS, Parquet, Cassandra, Hbase, Hive, Amazon S3</b>)</li> </ul>
<b>Visualization</b>	-Only clustering results	-N/A

# MLlib و Mahout مقایسه

	Mahout	MLlib
<b>Pros</b>	-Based on Hadoop & MapReduce	-Scalability -Performance -User-friendly API's -Integration with SparkSQL, Streaming & GraphX
<b>Cons</b>	-Low efficiency on iterative algorithms -Limited coverage of algorithms	-Configurability -Reliability -High-memory consumption

زمان بند

# Apache Oozie

- نیز یکی از ارکان اساسی هدوب می باشد که بدون آن اجرا و پردازش عملیات محاسباتی و استفاده از توان پردازشی غیر ممکن خواهد بود.
- زمان بندی وظایف را در هدوب بر عهده دارد.
- اگر بخواهید عملیات MapReduce انجام دهید می بایست به Oozie مراجعه کنید.
- Oozie پس از دریافت اطلاعات مربوط به هر عملیات آنرا زمان بندی و اجرا می کند و در نهایت گزارش عملیات را تولید می کند.
- کار اصلی Oozie اجرای عملیات MapReduce و Pig است
- ولی می توان از آن برای اجرای وظایف مربوط به HTTP، SSH، Email ... هم استفاده کرد.

Oozie App

127.0.0.1:9000/list\_oozie\_workflows/

OOZIE UI Workflows Coordinators Bundles Oozie Documentation

romain

Filter: Search for username, name, etc...

Show only 1 7 15 30 days with status Succeeded Running Killed

## Running

Submission	Status	Name	Progress	Submitter	Created	Last modified	Run	Id	Action
Tue, 21 May 2013 12:00:26	RUNNING	Forks	50%	romain	Mon, 20 May 2013 15:59:38	Tue, 21 May 2013 12:00:26	2	0000000-130520143302532-oozie-oozi-W	Kill

Showing 1 to 1 of 1 entries

یک UI جدید برای Oozie بنام

← Previous 1 Next →

## Completed

Completion	Status	Name	Duration	Submitter	Created	Last modified	Run	Id
Thu, 16 May 2013 14:09:08	SUCCEEDED	pig-app-hue-script	8s	romain	Thu, 16 May 2013 14:09:00	Thu, 16 May 2013 14:09:08	0	0000016-130513182859757-oozie-oozi-W
Thu, 16 May 2013 14:02:05	SUCCEEDED	pig-app-hue-script	8s	hdfs	Thu, 16 May 2013 14:01:57	Thu, 16 May 2013 14:02:05	0	0000015-130513182859757-oozie-oozi-W
Thu, 16 May 2013 13:57:40	SUCCEEDED	sss	0s	romain	Thu, 16 May 2013 13:57:40	Thu, 16 May 2013 13:57:40	0	0000014-130513182859757-oozie-oozi-W
Thu, 16 May 2013 13:37:57	SUCCEEDED	pig-app-hue-script	21s	romain	Thu, 16 May 2013 13:37:36	Thu, 16 May 2013 13:37:57	0	0000013-130513182859757-oozie-oozi-W
Thu, 16 May 2013 13:37:26	SUCCEEDED	pig-app-hue-script	8s	romain	Thu, 16 May 2013 13:37:18	Thu, 16 May 2013 13:37:26	0	0000012-130513182859757-oozie-oozi-W
Thu, 16 May 2013 13:37:14	SUCCEEDED	pig-app-hue-script	8s	romain	Thu, 16 May 2013 13:37:06	Thu, 16 May 2013 13:37:14	0	0000011-130513182859757-oozie-oozi-W

# شاخص گذاری و جستجو

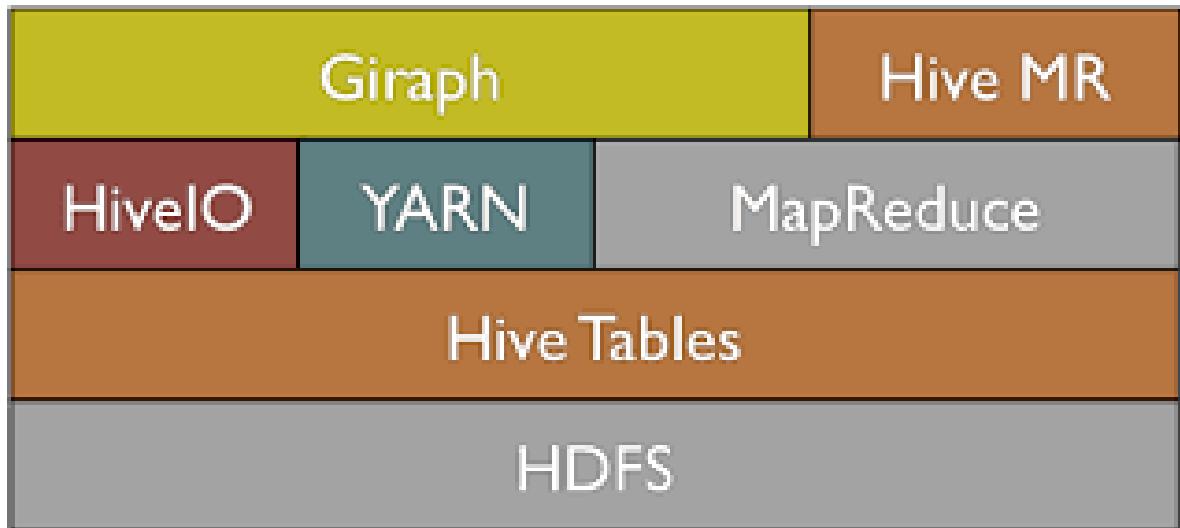
# Apache Lucene

- یک Search Library که توسط Java نوشته شده است.
- یک فراهم آورنده جستجو بر روی Document می باشد.
- داده های عددی و باینری را نیز می تواند ذخیره کند
- اما بیشتر توجه ها بر روی داده های متنی می باشد.

---

```
Document doc = new Document();
doc.add(new Field("title", "Big Data Analytics",
Store.YES, Index.ANALYZED, TermVector.NO));
doc.add(new Field("Instructor", "Dr. AkhoondZadeh",
Store.YES, Index.ANALYZED));
```

ابزارهای تحلیل گراف

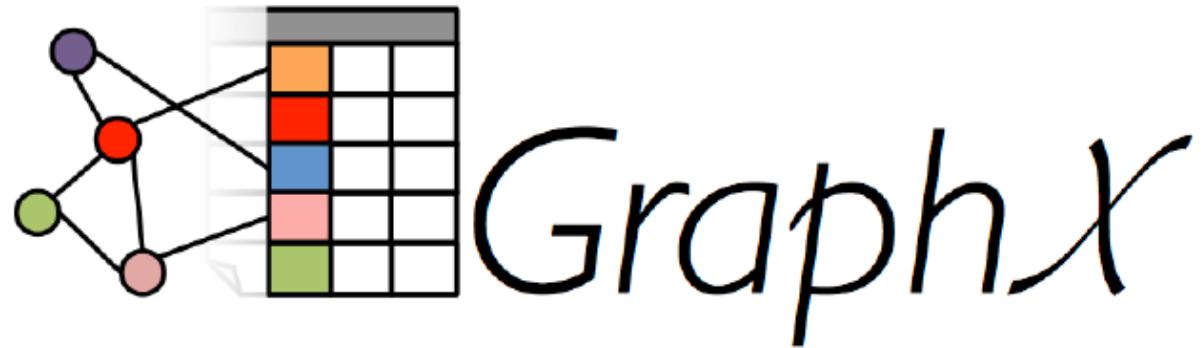


## Apache Giraph

- استفاده از تقسیم بندی مبتنی بر گره
- استفاده از HDFS برای ذخیره گراف
- مزایا
- سرعت بالای اجرای الگوریتم ها
- جامعه کاربری قوی
- برنامه سازی آسان
- معایب
- کارایی اش به الگوریتم مربوطه وابسته است.
- سرعت بالا در PageRank
- عملرد نسبتاً ضعیف در پیمایش گراف ها

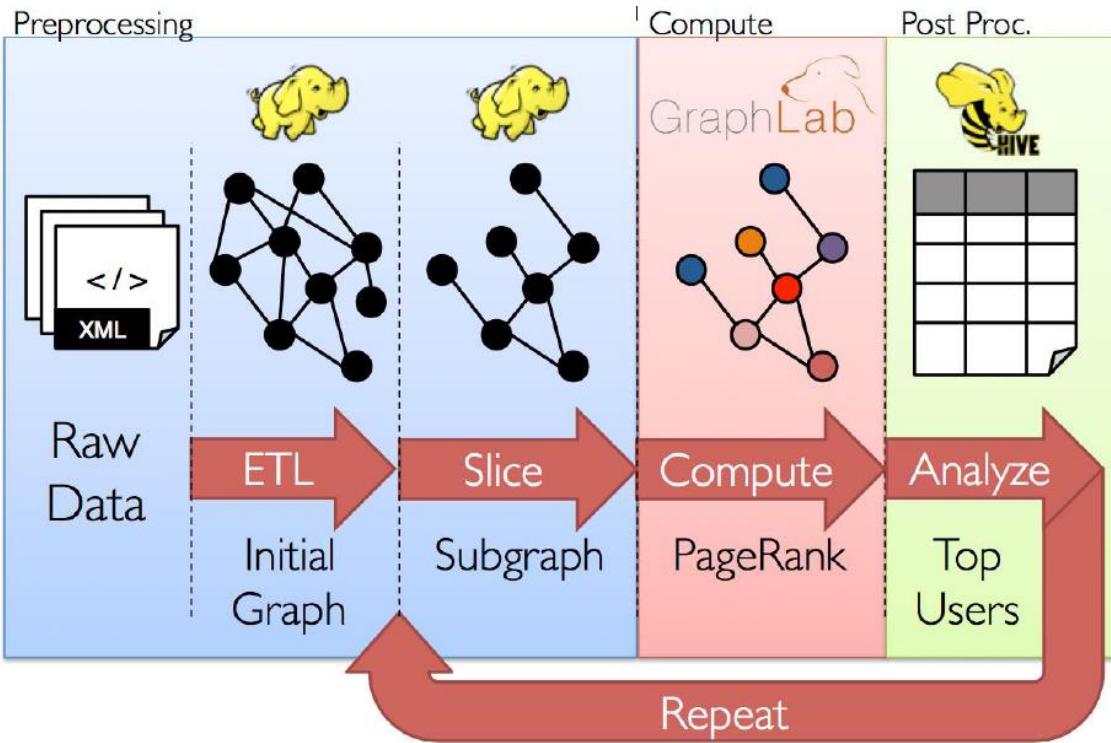
# GraphX

- ابزار اسپارک برای تعریف گراف و انجام محاسبات موازی روی آن ها
- بسیاری از توابع مرسوم و مفید برای کار با گراف ها در این ابزار پیاده سازی شده است.
- استفاده از تقسیم بندی مبتنی بر گره



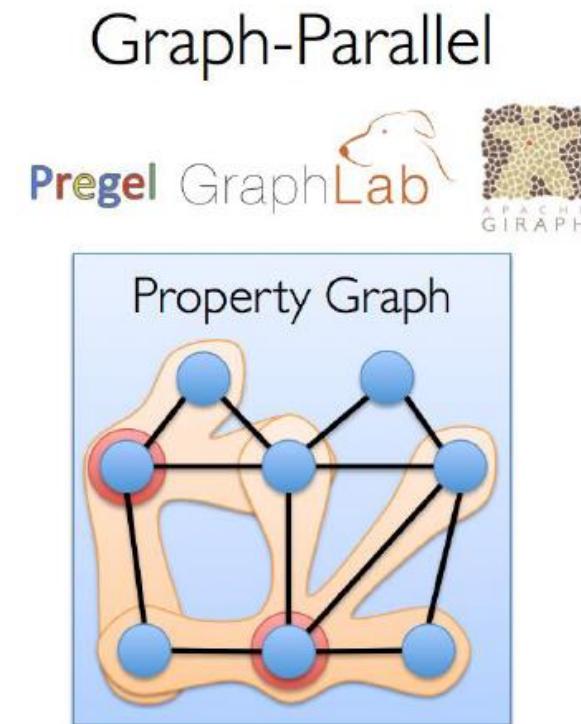
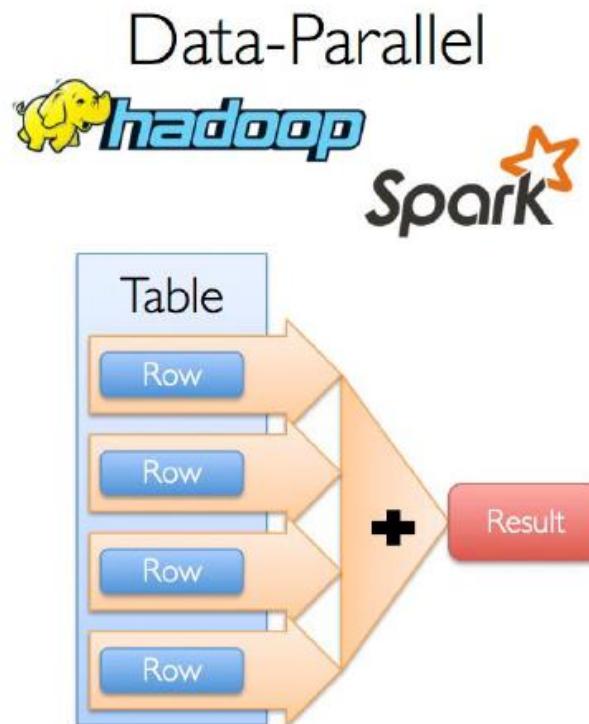
# Graph Parallel vs Data Parallel

- سیستم های Graph Parallel برای پیاده سازی الگوریتم های بازگشتی بر روی گراف، بهینه هستند.
- سیستم های Data Parallel، برای انجام محاسبات روی داده های موازی مانند عملیات Join, Filter, Map, ... مناسب هستند.



# Graph Parallel vs Data Parallel

- خاصیت های Graph Parallel و Data Parallel را با هم ترکیب کرده است.
- می تواند بدون جابجایی داده یا تکرار آن (Duplication)، شبکه را بصورت گراف یا جدول نشان دهد.



# مفهوم RDG (Resilient Distributed Property Graph)

- در RDG به RDD، GraphX گسترش پیدا کرده است.
- RDG، یک گراف جهت دار چندگانه است که هر راس و یال در آن، Property خاص خود را دارد.
- هر راس یک id منحصر به فرد دارد و هر یال با id مربوط به مبدأ و مقصد خود شناسایی می شود.
- هر راس یک Scala Object Property یا بصورت ذخیره می شوند.

