انواع پایگاه داده های غیر رابطه ای

www.bigdata-ir.com

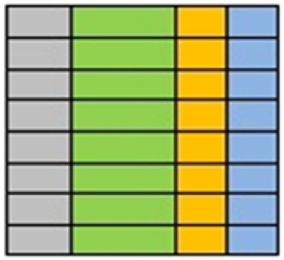
Telegram: @BigData_Channel

مدرس:
علیرضا حبیبی

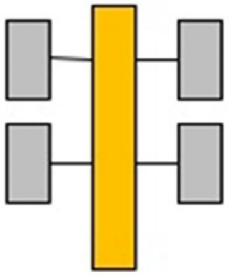
انواع الگوها در عرصه بانک های اطلاعاتی

NoSQL Database Patterns

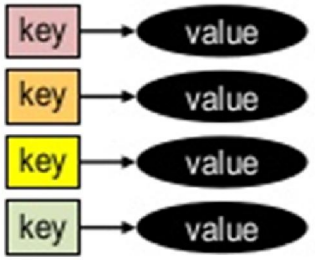
Relational



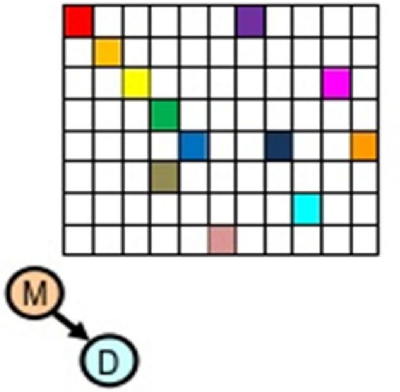
Analytical (OLAP)



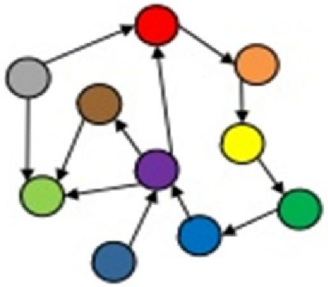
Key-Value



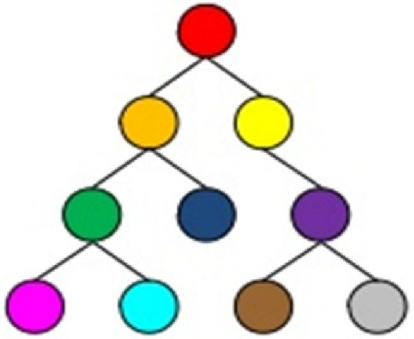
Column-Family



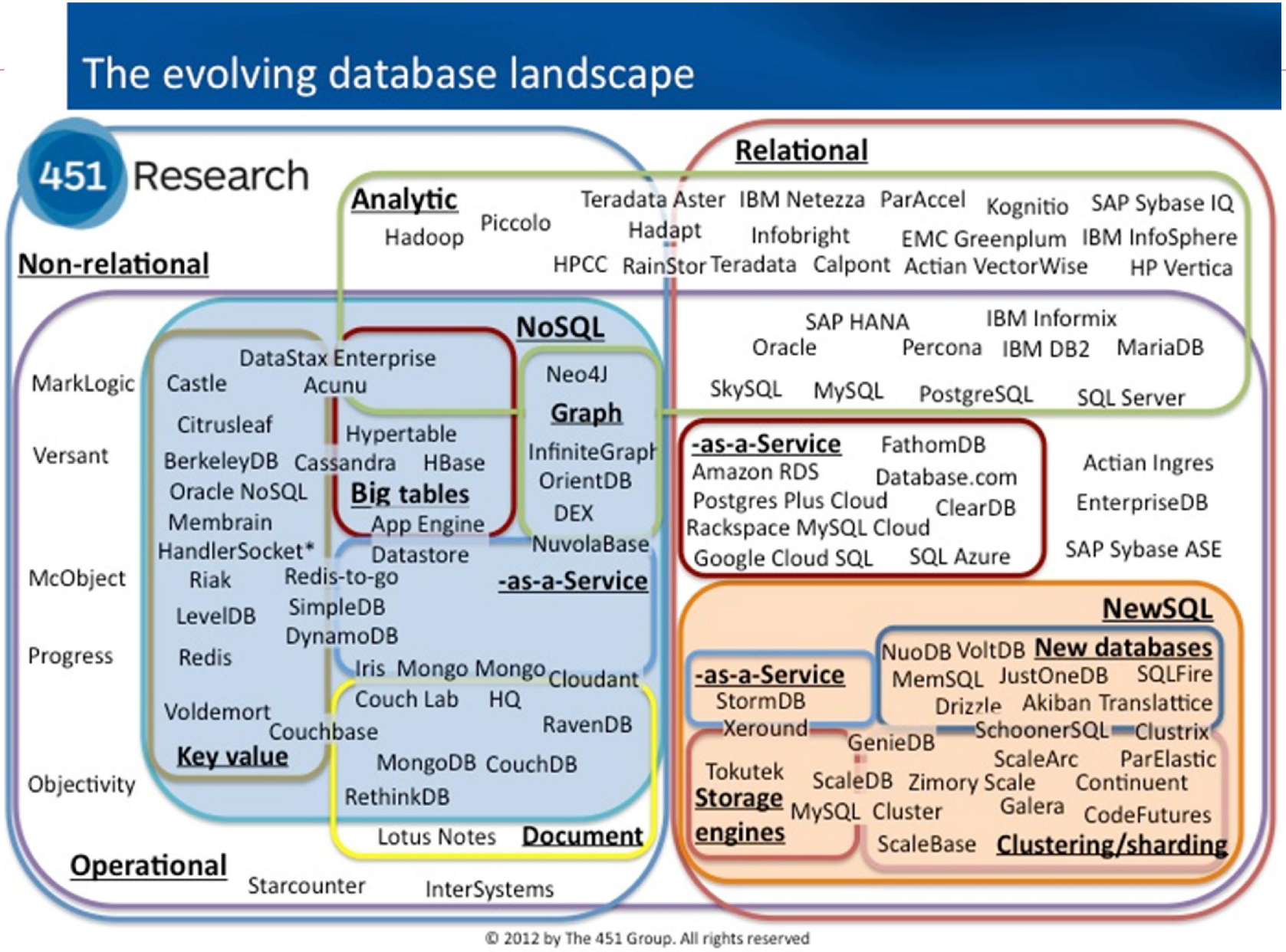
Graph



Document



انواع پایگاه داده های غیر رابطه ای



Key-value store چیست ؟

- Key-value store چیست ؟
- مزایای استفاده از key-value store
- نحوه استفاده از key-value store
- بررسی یک نمونه مثال کاربردی موتور جستجو

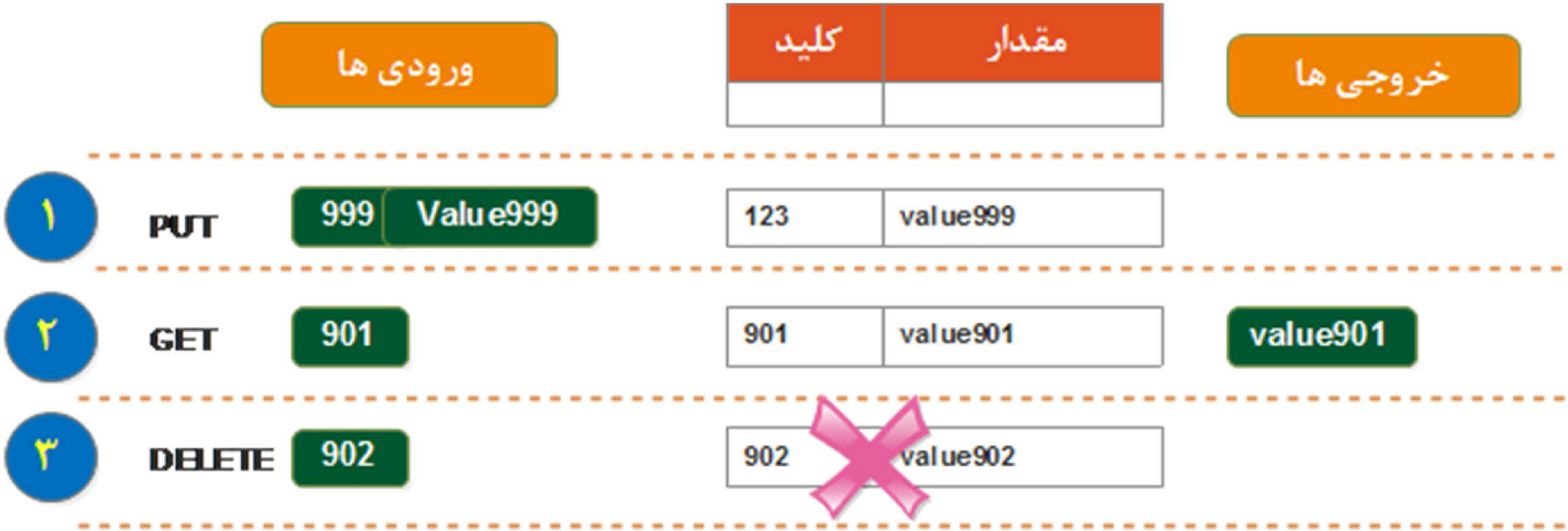
کلید	مقدار
Fabak_LogoImage.jpg	یک فایل باینری تصویری
http://www.fabak.ir/resume.html	یک صفحه اچ تی ام ال
C:/Backup/FabakDoc.pdf	یک فایل پی دی اف
c1936d9fc9677bc3398b06b4542632a9	رشته هش شده سایت فابک
show-FabakArticle?Article-id=999&format=xml	<Article><id>999</id></Article>

Key-value store چیست ؟

- یکی از key-value store، امکان ذخیره سازی هر نوع داده در value است
- اطلاعات به عنوان یک BLOB ذخیره می شوند و از دستور GET جهت بازیابی استفاده می گردد.BLOB
  - از کلمات Binary Large Object اقتباس شده است شامل متن، تصاویر ، صدا و سایر اشیاء چند رسانه ای است.
- کلید (Key)، انعطاف پذیر بوده و می تواند با قالب های مختلفی همچون اسامی مسیر منطقی به تصاویر و یا فایل ها ، رشته های تولید شده هوشمند از مقدار هش شده ، فراخوانی سرویس وب REST و یا SQL queries ارایه شود
- مقادیر(Value) نیز نظیر کلیدها ، دارای انعطاف لازم بوده و می توانند شامل هر نوع BLOB داده نظیر تصاویر ، صفحات وب ، مستندات و ویدیوها باشد.

نحوه استفاده از Key-value store

- در نظر گرفتن جدولی با دو ستون است . اولین ستون Key و دومین ستون value است.
- سه عملیات را می توان در ارتباط با key-value store تعریف کرد : put, get و delete دستورات فوق ، اینترفیس مورد نیاز برنامه نویسان جهت کار با Key-value store را ارایه می نماید

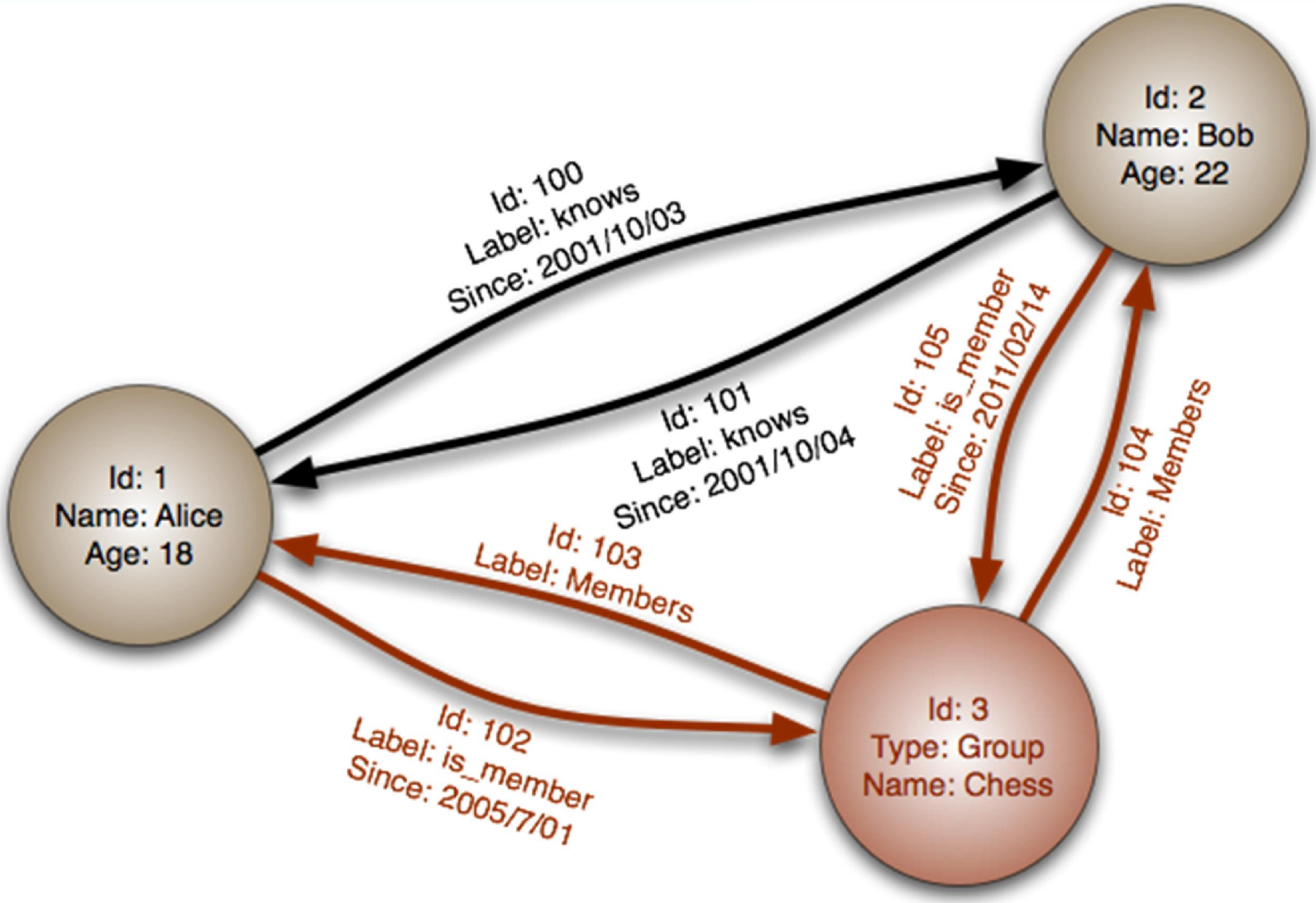


key-value store دارای دو قانون کلی زیر است:

- **Distinct keys**: نمی توان دارای دو سطر با key-value مشابه بود.
  - این بدان معنی است که تمامی کلیدها در key- value storeمنحصربفرد می باشند. در صورتی که به صورت منحصربفرد یک زوج key-value را مشخص و شناسائی نکنیم ، نمی توان یک مقدار را به عنوان نتیجه کار برگرداند .

- **No queries on values**: نمی توان بر روی مقادیر موجود در جدول query اجراء کرد.
  - در یک بانک اطلاعاتی رابطه ای ، می توان با استفاده از where clauseمحدودیت هایی را در خصوص برگرداندن نتایج حاصل از اجرای یک queryاعمال کرد . در صورتی که در یک key-value store ماحصل اجرای query،برگرداندن یک آیتم خواهدبود.

مدل داده ای مبتنی بر گراف

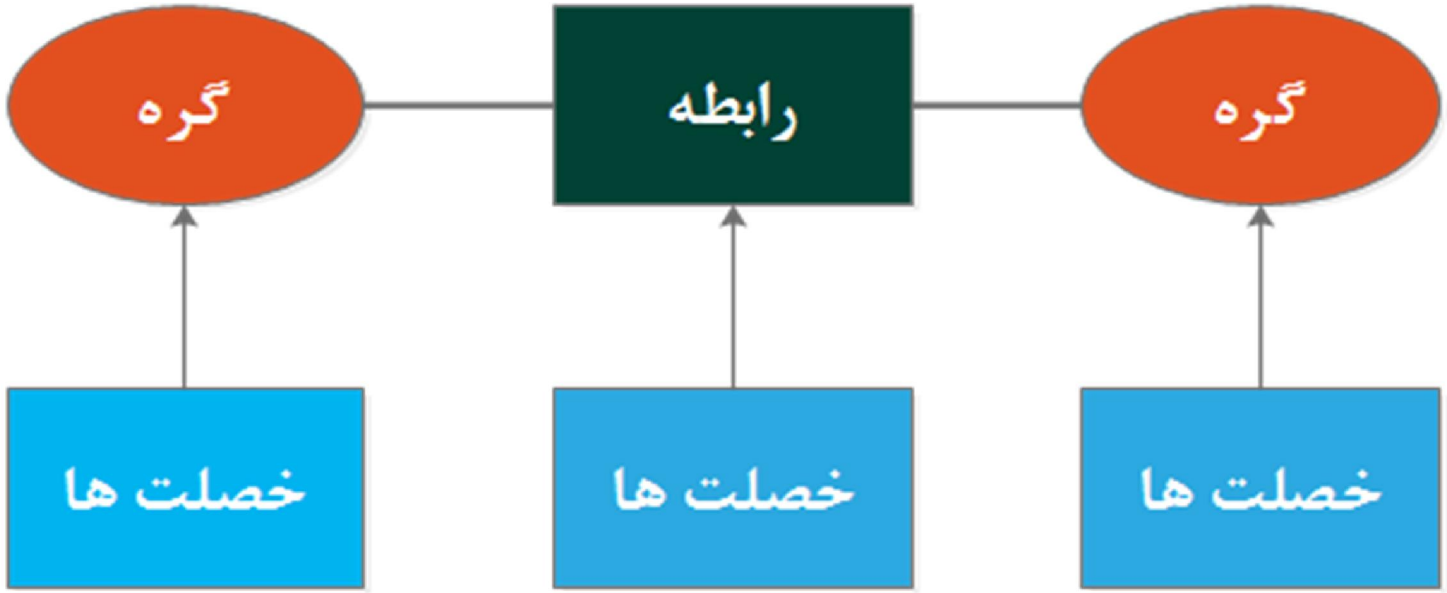


مروری بر Graph store

- بانک های اطلاعاتی گراف از مدل گراف جهت ذخیره داده استفاده می نمایند.
  - در این مدل ، ساختار گراف شامل رئوس و یال ها (به عنوان دو موجودیت مهم مدل سازی هر نوع گراف) مدل سازی می گردد.
- می توان از تمامی الگوریتم های موجود تئوری گراف (با توجه به قدمت طولانی آن) برای حل مسائل گراف با کارآیی بیشتر و در زمان کمتر استفاده کرد.
- معماری داده key-value storeشامل دو فیلد key و value است . در مقابل، در یک Graph store از سه فیلد اساسی داده با نام گره ، روابط و خصلت استفاده می شود.
  - به برخی از گراف ها به دلیل ساختار node-relationship-node، اصطلاحا triple storesگفته می شود

مروری بر Graph store

- شبکه های اجتماعی ، موتورهای مبتنـی بـر قواعـد و تحلیـل سـاختارهای پیچیده شبکه نمونه هایی از کاربرد الگوی معمـاری داده Graph store می باشند .

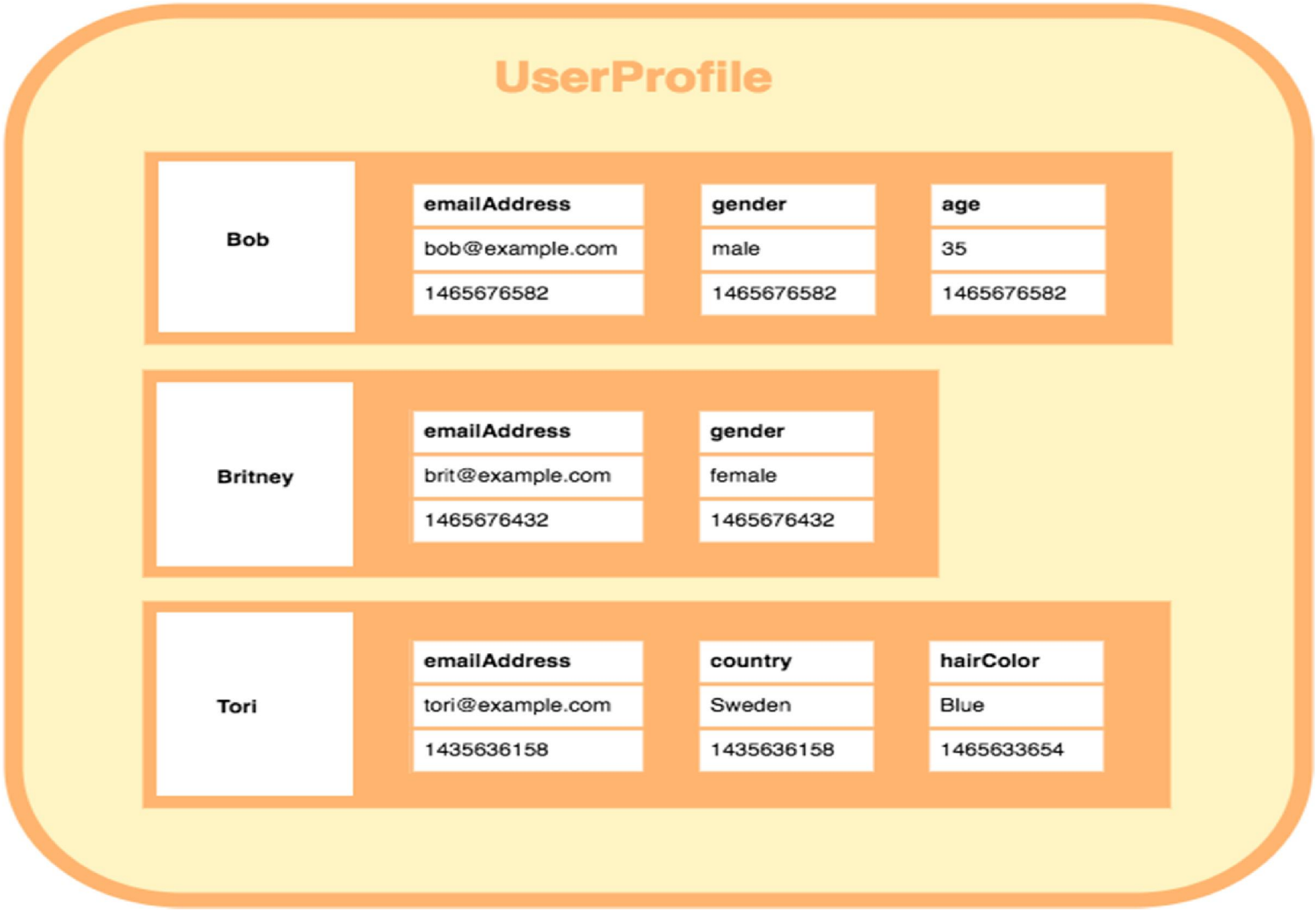


یک بانک اطلاعاتی گراف شامل تعداد زیادی ساختار گره - رابطه - گره است که برای تشریح گره ها و روابط از خصلت ها استفاده می شود

مروری بر Graph store

- گره های گراف معمولا بیانگر اشیاء دنیای واقعی هستند.
  - نظیر افراد ، سازمان ها ، شماره تلفن ، صفحات وب ، کامپیوترهای موجود بر روی یک شبکه و یا حتی سلول های بیولوژیکی در یک ارگانیزم زنده
- ارتباطات را می توان به منزله اتصالات بین اشیاء درنظر گرفت که معمولا به صورت کمان در دیاگرام ها نشان داده می شوند. اجرای یک query بر روی یک گراف ، مشابه حرکت کردن بین گره ها در یک گراف است. مثال:
  - کوتاهترین مسیر بین دو گره موجود در گراف چیست ؟
  - چه گره ای دارای همسایگانی با خصلت های خاصی است ؟
  - با دادن دو گره در گراف ، تا چه میزان همسایگان آن به هم شبیه هستند ؟
  - متوسط ارتباطات بین نقاط مختلف گراف با یکدیگر چه میزان است ؟
- در Graph stores عملیات Join سبک تراست. علت این کار به ماهیت کوچک هر گره و قابلیت نگهداری گراف در RAM برمی گردد

مدل داده ای ستون گسترده



مدل داده ای ستون گسترده

- ستون گسترده بگونه ای پیاده سازی شده است تا بتواند با سیستم فایل های توزیع شده نظیر Hadoop و MapReduce کار کند.
- سیستم هایی با میلیاردها سطر و صدها یا هزاران ستون!
  - به عنوان نمونه یک برنامه GIS نظیر GoogleEarth ممکن است از یک شناسنه سطر Row ID به عنوان طول جغرافیایی یک نقشه و از نام ستون به عنوان عرض جغرافیایی استفاده نماید .
- ماتریسی sparse است که صرفا تعداد محدودی از ستون های آن دارای مقدار می باشند. متاسفانه ، بانک های اطلاعاتی رابطه ای برای ذخیره داده sparse مناسب نمی باشند

مثال: ذخیره اطلاعات تحلیلی توسط گوگل در BigTable

- از BigTable برای ذخیره اطلاعات یک وب سایت در Google Analytics استفاده کرد .
- **سرویس Google Analytics** این امکان را فراهم می نماید که بتوان ملاقات کنندگان یک وب سایت را پیگیری کرد . هر زمان که کاربری بر روی یک صفحه وب کلیک می نماید، اطلاعات آن در یک row-column ذخیره می گردد که دارای یک URL و یک timestamp به عنوان row ID است.
- یک نمونه دیگر استفاده از BigTable برای ذخیره حجم بالایی از اطلاعات GIS است. سیستم های GIS نظیر Google map اطلاعات نقاط جغرافیایی زمین را با شناسایی هر مکان با استفاده از مختصات طول و عرض جغرافیایی انجام می دهند. سیستم به کاربران اجازه می دهد بر روی زمین حرکت کرده و بر روی آن زوم نمایند.

اصول اولیه الگوی معماری داده document store

- Key-value store و Bigtable values دارای فقدان یک ساختار رسمی می باشند و در عمل از ایندکسینگ و یا قابلیت های جستجو برای کل داده های مقدار استفاده نمی کنند.

- Document storesدر نقطه مقابل دو مدل فوق کار می کند. کلید معمولا جستجو نمی شود. ولی امکان بازیابی تقریبا هر آیتمی از درون یک document store وجود دارد.

- یک Key-value store قادر به ذخیره تمامی سند در بخش valueاست ولی یک document store قادر به استخراج بخش های جانبی تعداد زیادی از اسناد بدون بارگذاری هر سند درون حافظه است .

اصول اولیه الگوی معماری داده document store

JSON

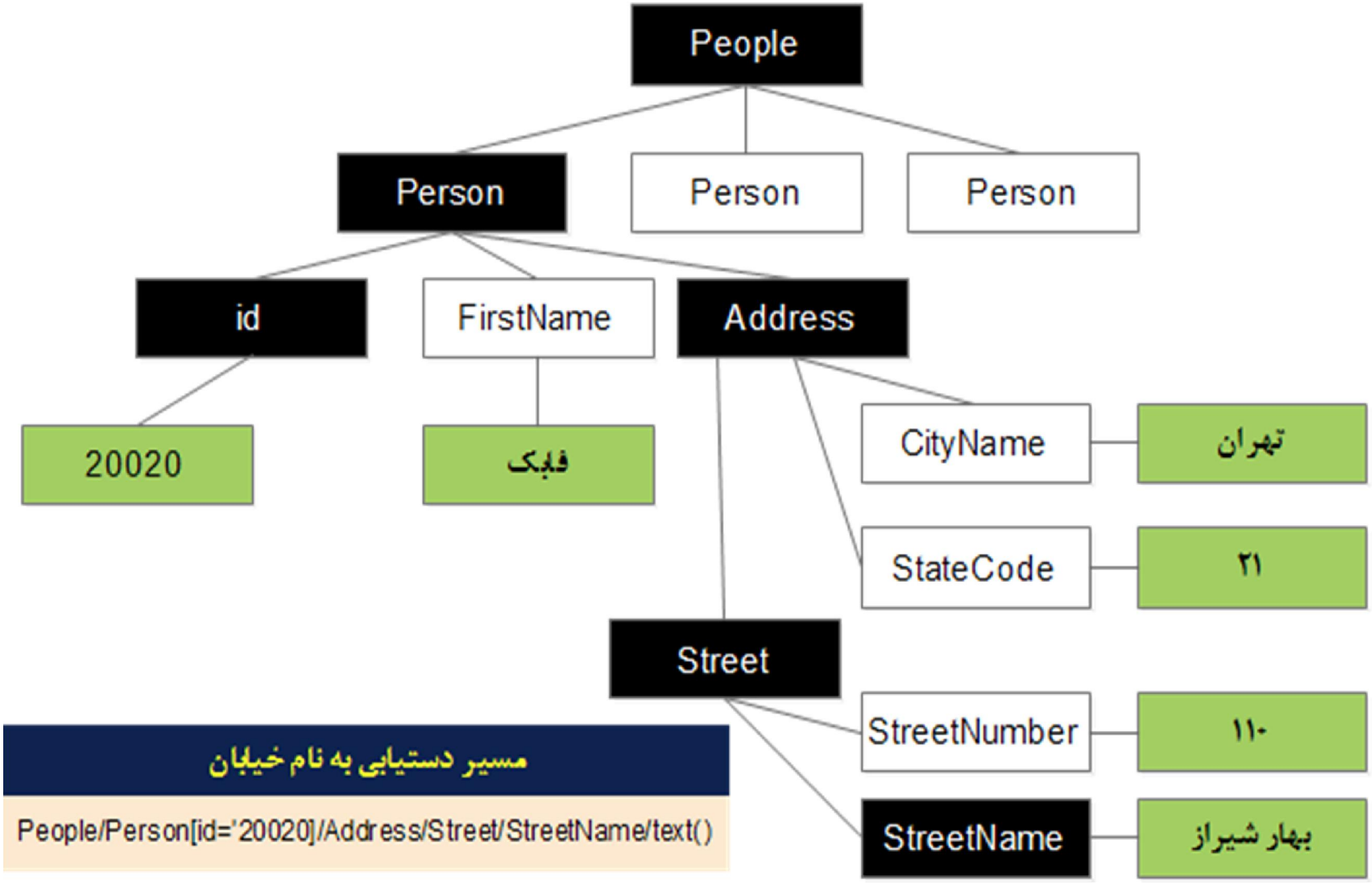
```
{
"siblings": [
{"firstName":"Anna","lastName":"Clayton"},
{"firstName":"Alex","lastName":"Clayton"}
]
}
```

XML

```
<siblings>
<sibling>
<firstName>Anna</firstName>
<lastName>Clayton</lastName>
</sibling>
<sibling>
<firstName>Alex</firstName>
<lastName>Clayton</lastName>
</sibling>
</siblings>
```

- یک document store با گونه های مختلفی ارایه می شود.
  - برخی بر اساس ساختار درختی شی سریالی ساده
  - برخی پیچیده تر بوده و شامل محتویاتی می باشند. (مثل JSON)

ساختار درختی



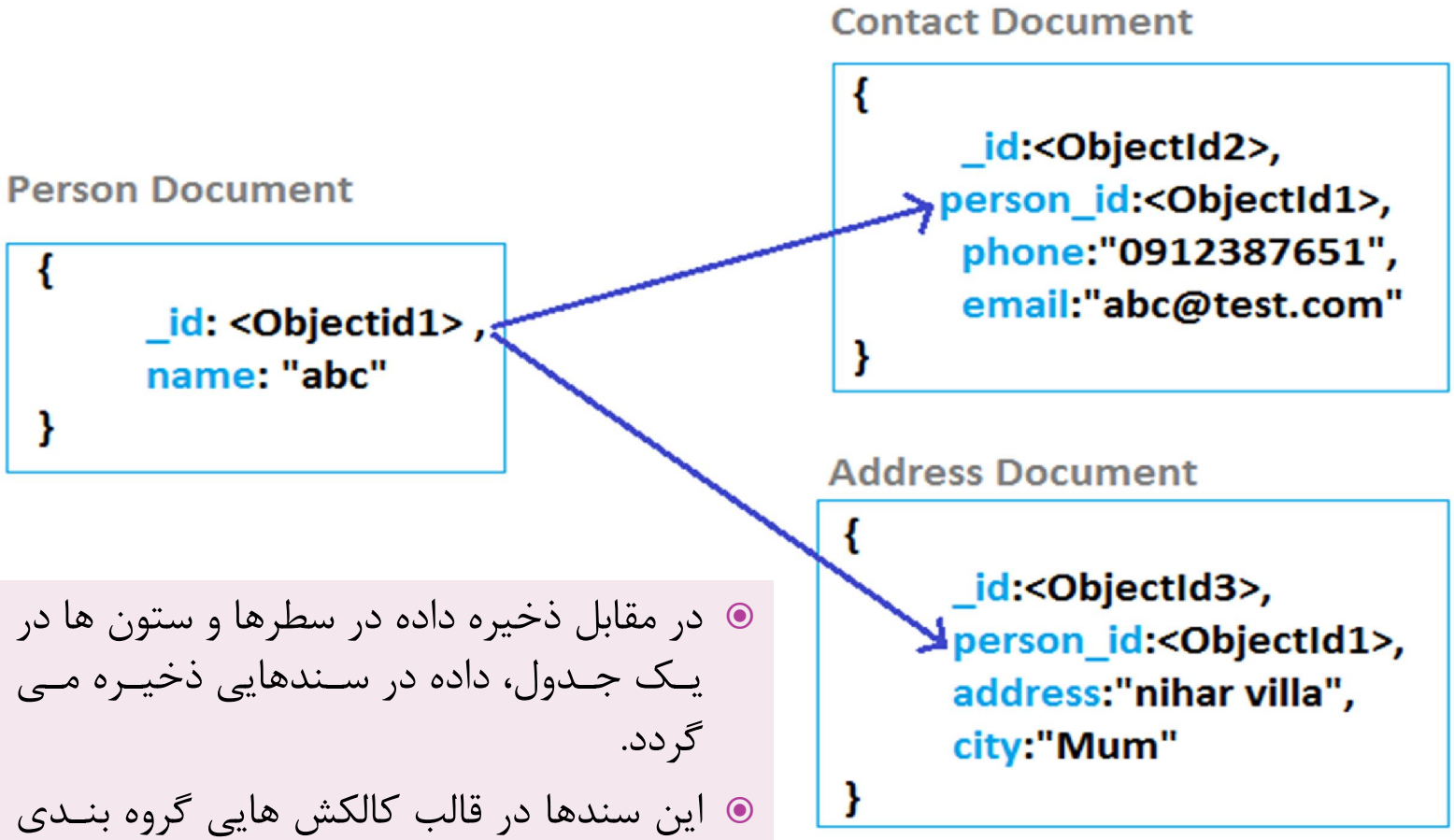
مسیر دستیابی به نام خیابان

```
People/Person[id='20020]/Address/Street/StreetName/text()
```

اصول اولیه الگوی معماری داده document store

- Document stores، از یک ساختار درختی که با یک گره ریشه شروع می شود و دارای شاخه هایی است که هر یک می توانند دارای شاخه های دیگری باشند، استفاده می نماید.
- مقادیر داده معمولا به عنوان برگ های یک درخت ذخیره می گردند. به موازات اضافه کردن یک سند، برای هر آیتم ذخیره شده درون آن به صورت اتوماتیک ایندکس ایجاد می گردد.
  - این بدان معنی است که با آگاهی از هر فیلد سند، می توان به سرعت تمامی سندهایی که دارای خصلت مشابه باشند را پیدا کرد.
- Document stores، می تواند نه تنها این موضوع را بگوید که آیا آیتم جستجو در document وجود دارد، بلکه مکان واقعی آیتم را با استفاده از document path در اختیار شما قرار دهد
- documet path در واقع یک نوع کلید است که به کمک آن امکان دستیابی به برگ های یک ساختار درختی فراهم می شود

مدل داده ای مبتنی بر محتوا مثل JSON



- در مقابل ذخیره داده در سطرها و ستون ها در یک جدول، داده در سندهایی ذخیره می گردد.
- این سندها در قالب کالکش هایی گروه بندی می شوند
- هر سند می تواند دارای یک ساختار کاملا متفاوت باشد

تله باز ماندگان

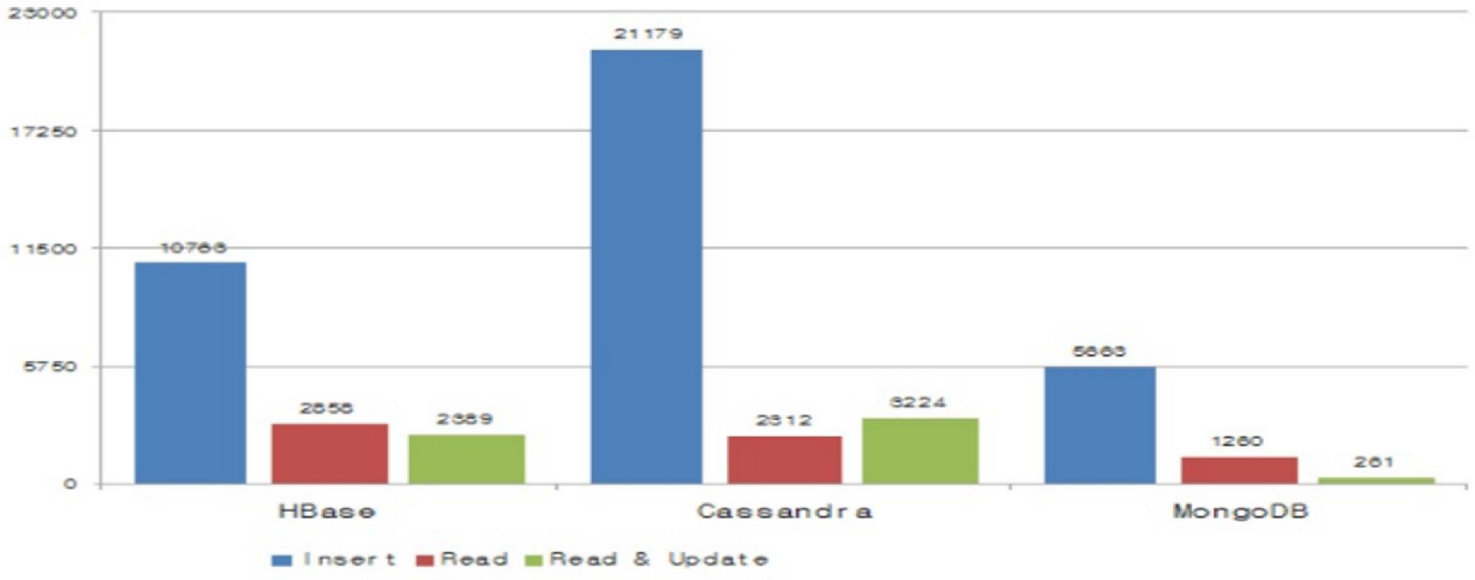
خصوصایت جزئی تر

Comparison Matrix

	Partitioning			Availability		Replication			Storage	
	Hash/sort	Dynamic	Routing	Failures handled	During failure	Sync/async	Local/geo	Consistency	Durability	Reads/ writes
PNUTS	H+S	Y	Rtr	Colo+ server	Read+ write	Async	Local+ geo	Timeline + Eventual	Double WAL	Buffer pages
MySQL	H+S	N	Cli	Colo+ server	Read	Async	Local+ nearby	ACID	WAL	Buffer pages
HDFS	Other	Y	Rtr	Colo+ server	Read+ write	Sync	Local+ nearby	N/A (no updates)	Triple replication	Files
BigTable	S	Y	Rtr	Colo+ server	Read+ write	Sync	Local+ nearby	Multi-version	Triple replication	LSM/ SSTable
Dynamo	H	Y	P2P	Colo+ server	Read+ write	Async	Local+ nearby	Eventual	WAL	Buffer pages
Cassandra	H+S	Y	P2P	Colo+ server	Read+ write	Sync+ Async	Local+ nearby	Eventual	Triple WAL	LSM/ SSTable
Megastore	S	Y	Rtr	Colo+ server	Read+ write	Sync	Local+ nearby	ACID/ other	Triple replication	LSM/ SSTable
Azure	S	N	Cli	Server	Read+ write	Sync	Local	ACID	WAL	Buffer pages

مقایسه نوشتن در کسندرا ...

Writes in Cassandra vs. Other Databases

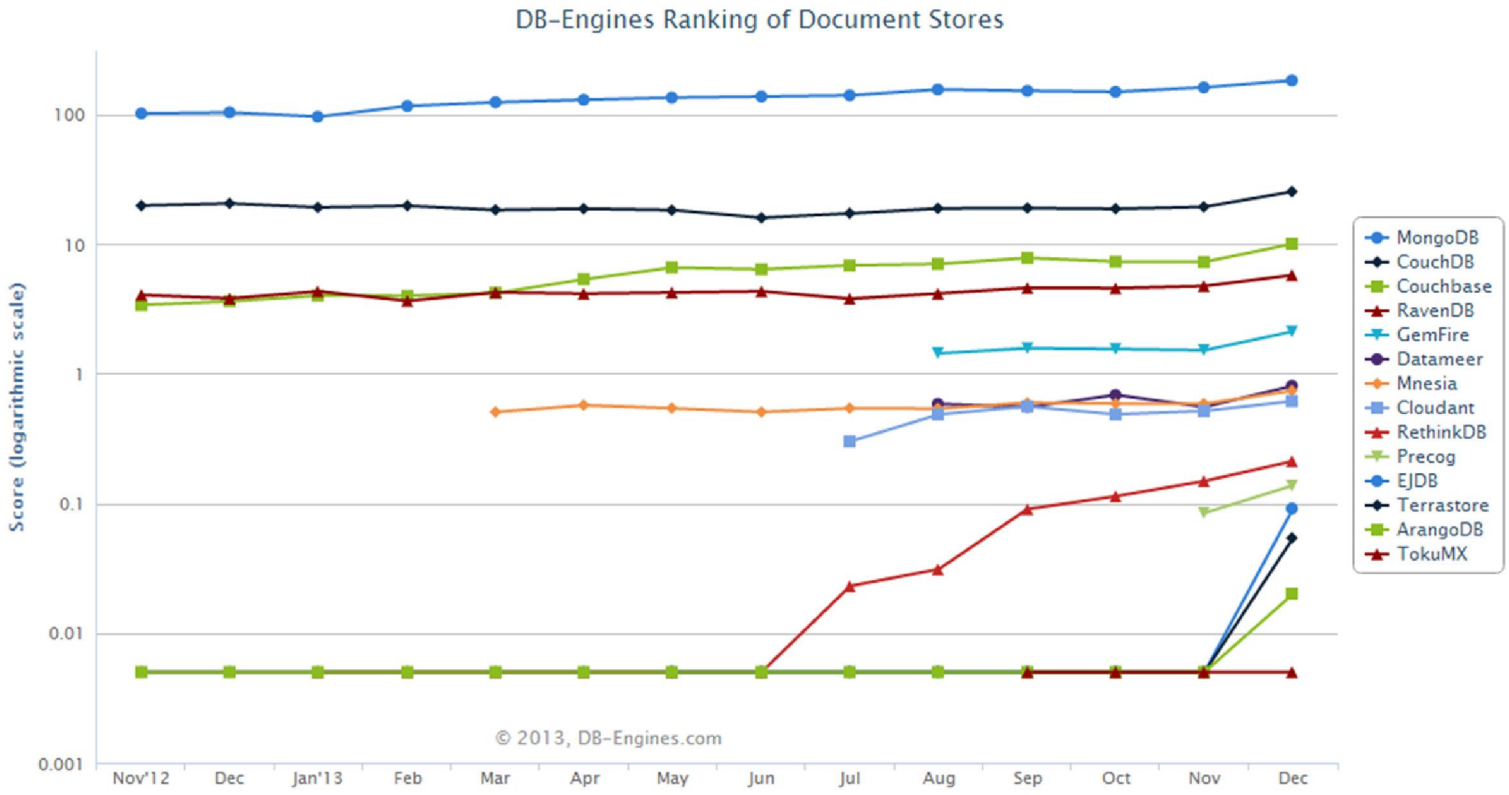


Cassandra is up to:

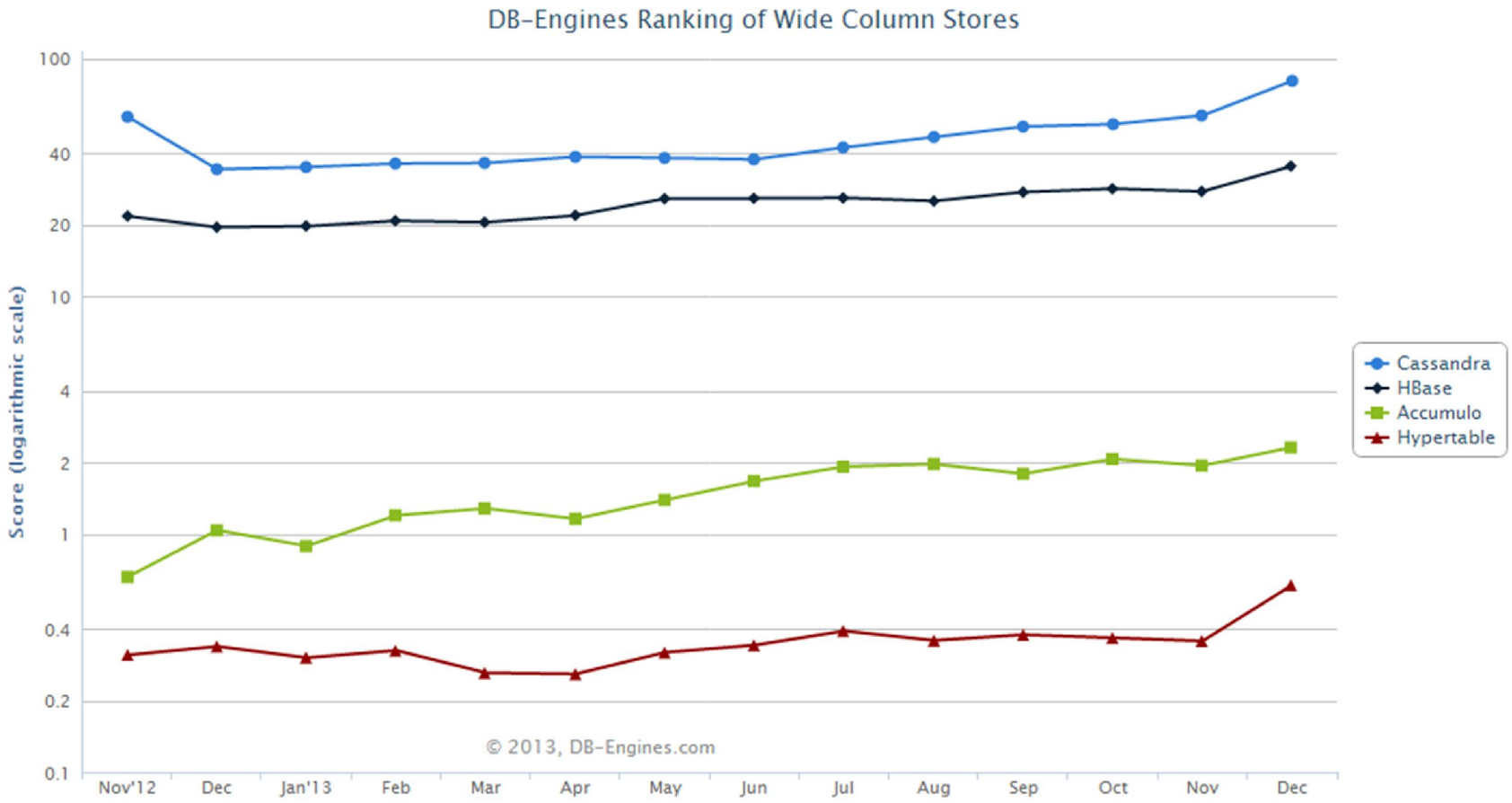
4x better in writes!
2x better in reads!
12x better in reads/updates!

Sept, 2011: http://blog.cubrid.org/dev-platform/nosql-benchmarking/

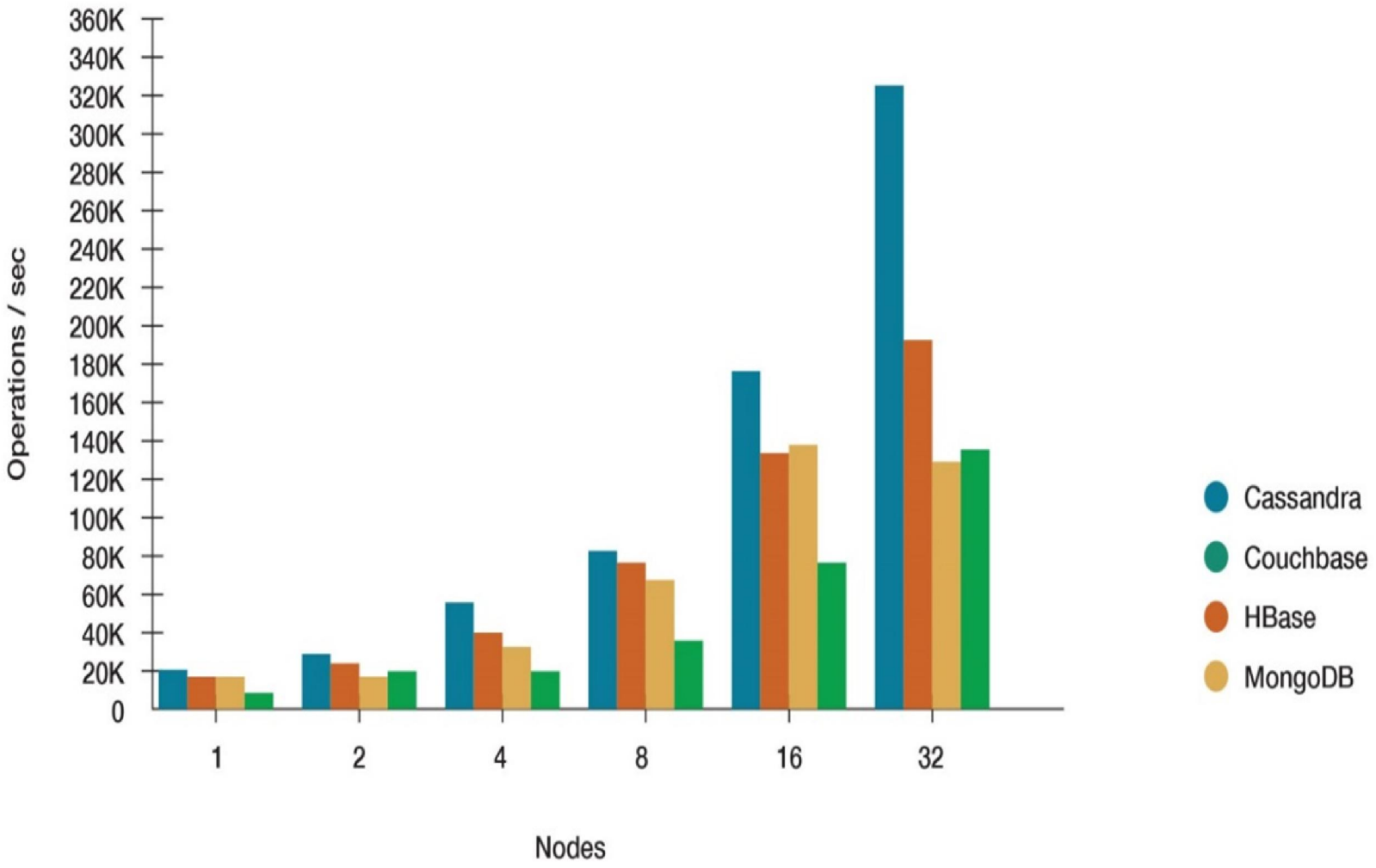
مقایسه مبتنی بر سند



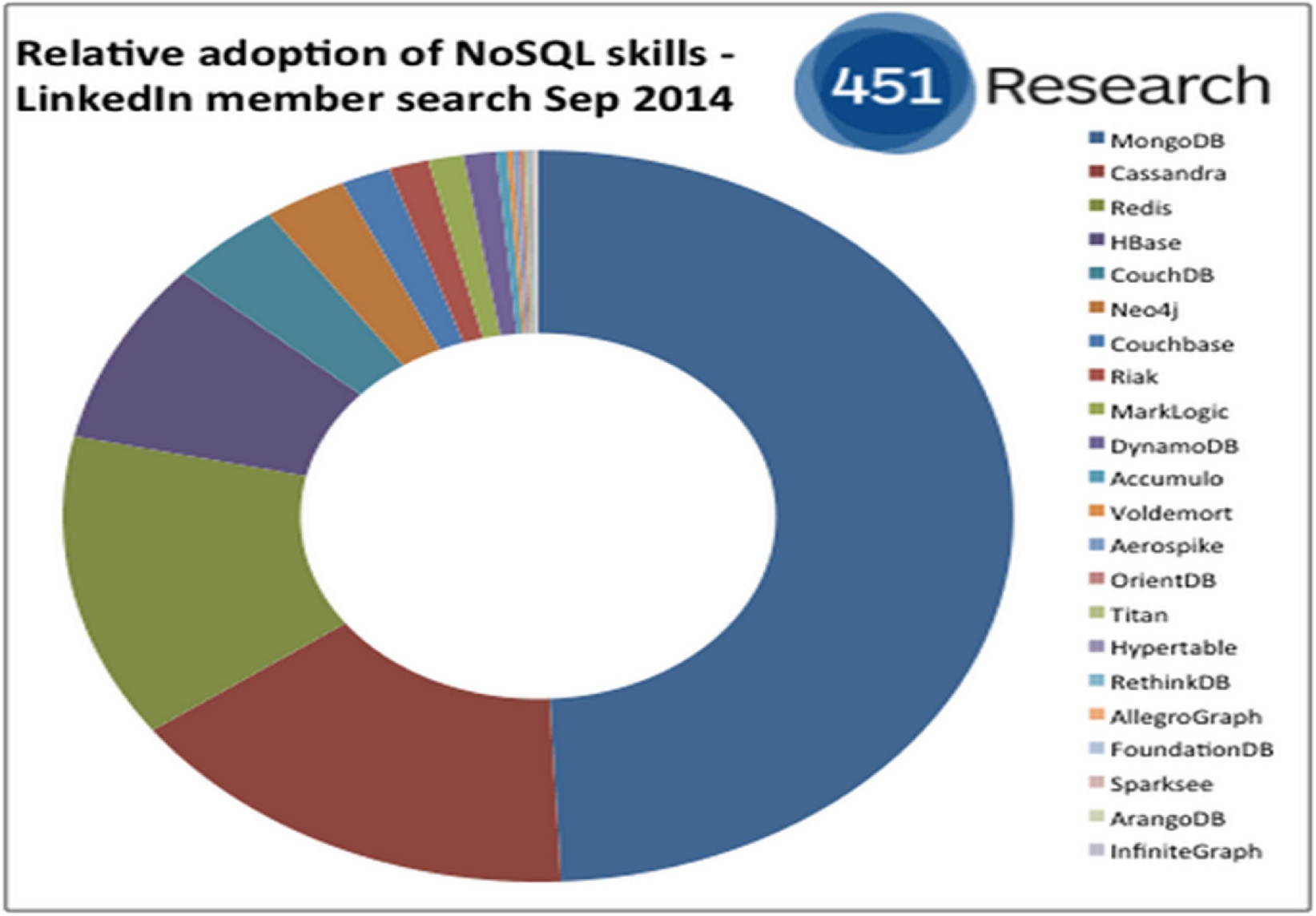
مقایسه مبتنی بر ستون



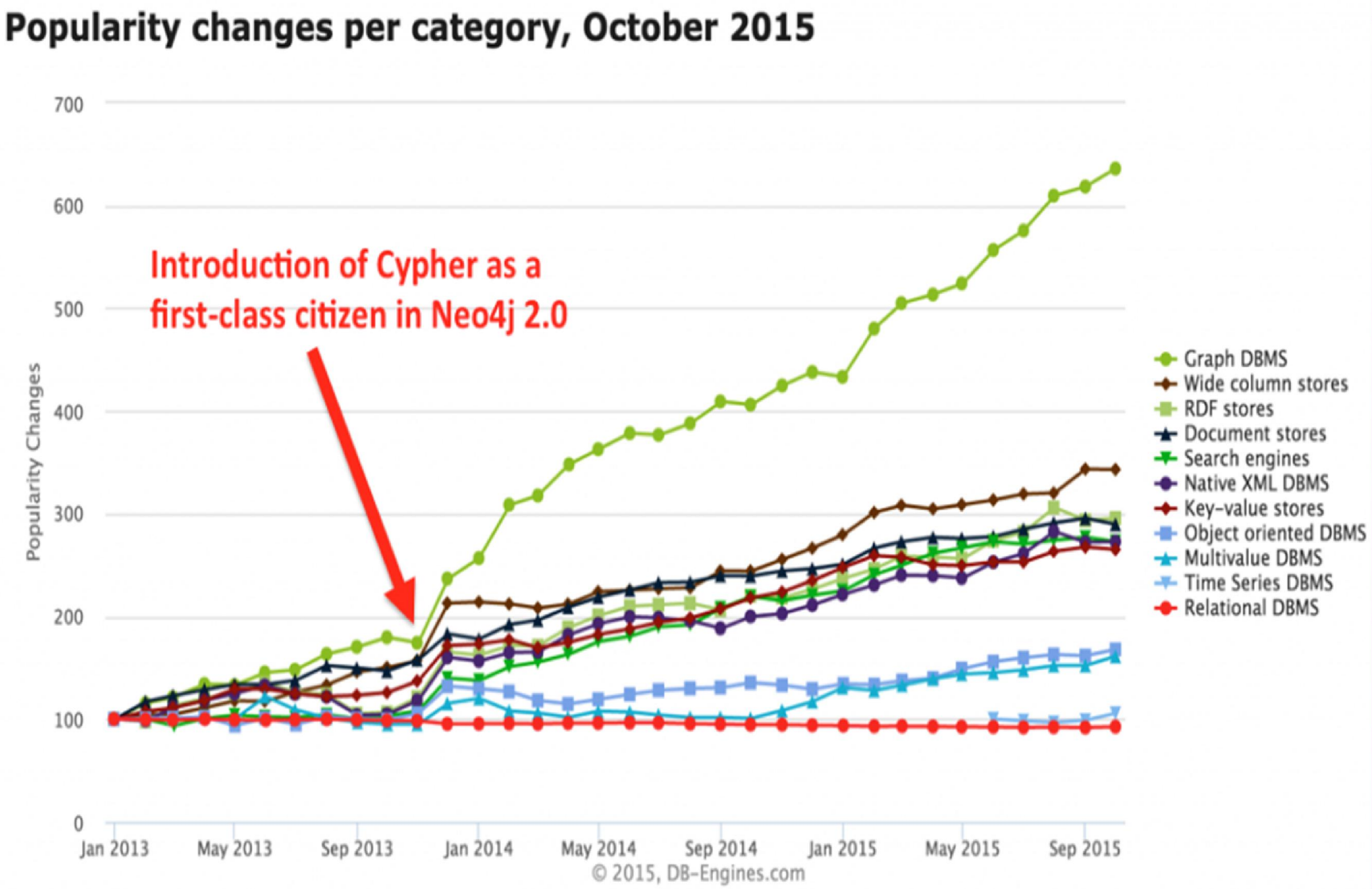
مقایسه عملکرد



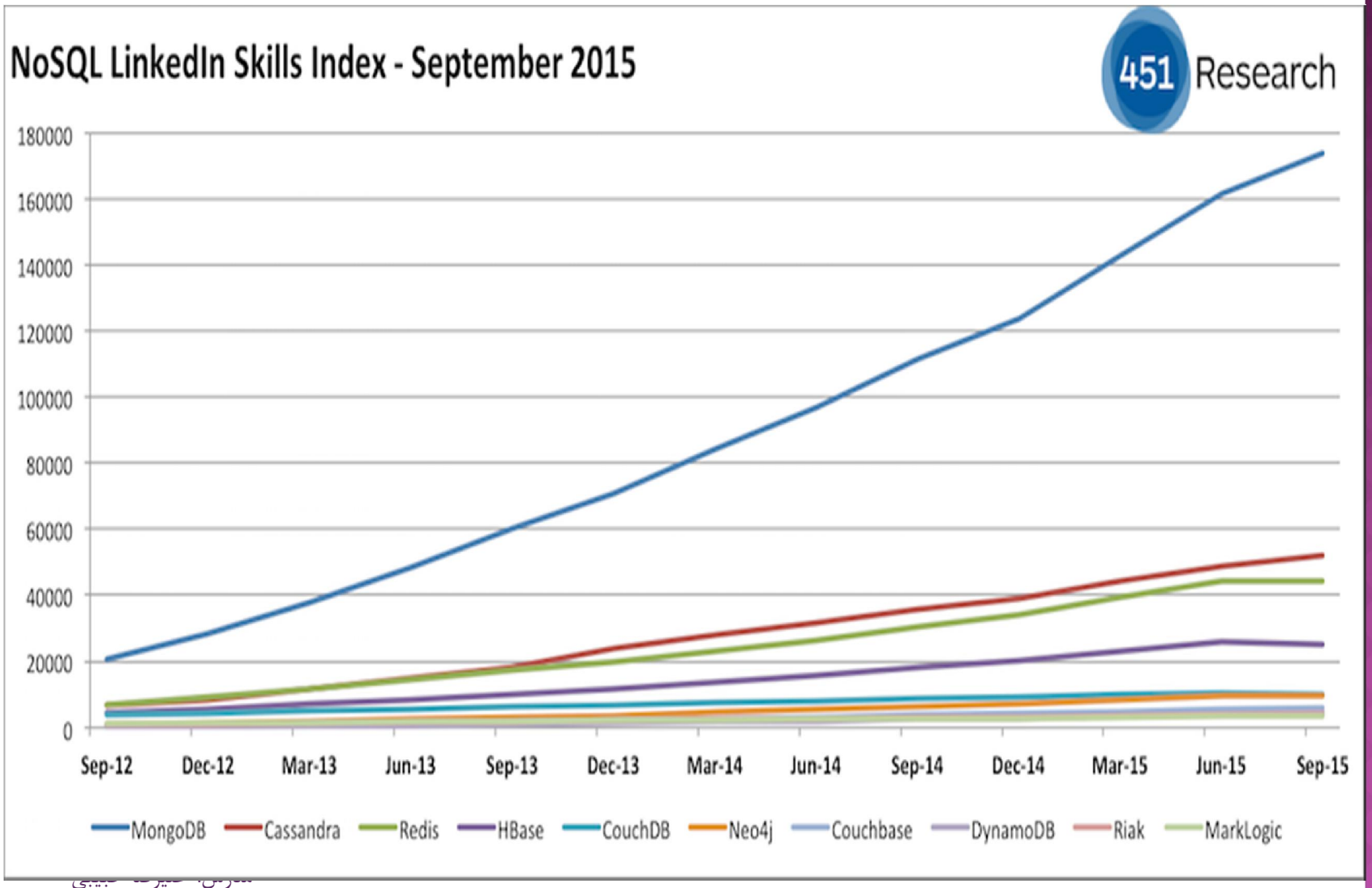
مقایسه فراوانی استفاده



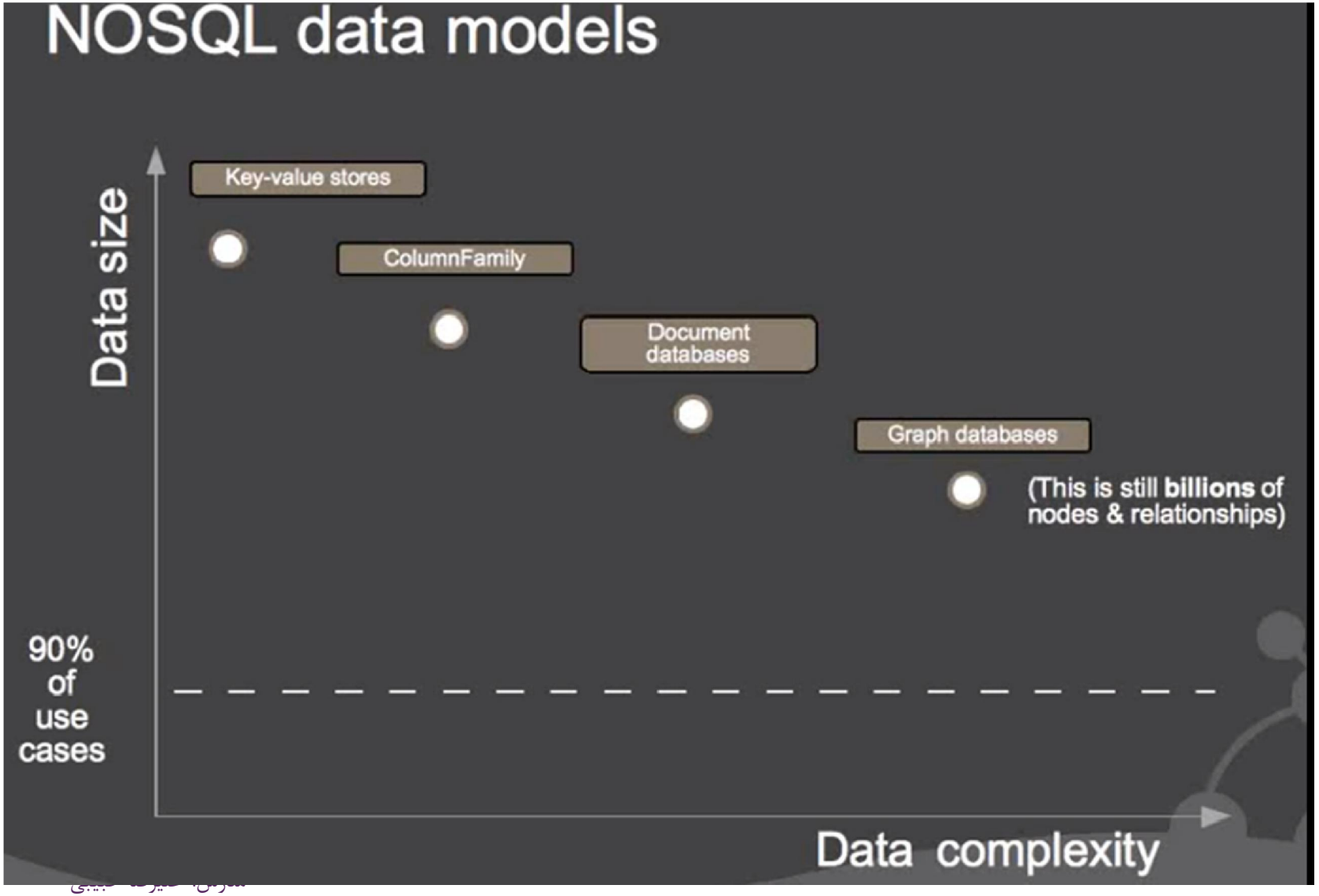
میزان شیوع



توان شاخص گذاری



پشتیبانی از حجم در برابر پیچیدگی



Cassandra VS MongoDB

Cassandra	MongoDB	
Semi starchier	Schema less (is lie)	
Support Varity, Vast, Velocity	Support Varity	
Not Single point to failure	Single point to failure	
linear-scale performance	Dependent on hardware	
Peer-to-Peer Replication	Master-Slave Replication	
Best partitioning	medium partitioning	
Not Single point to write	Single point to write	