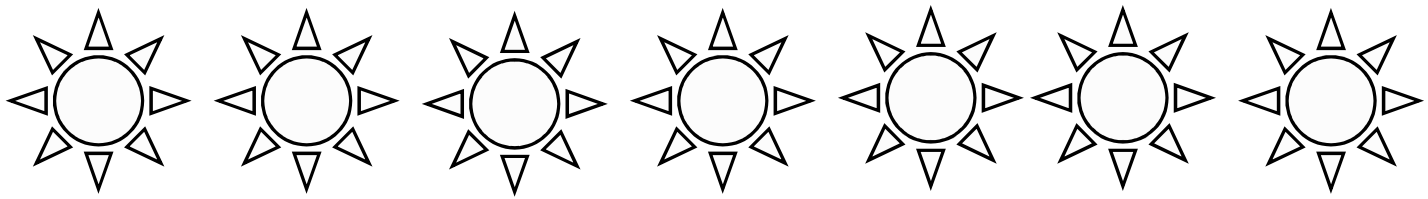
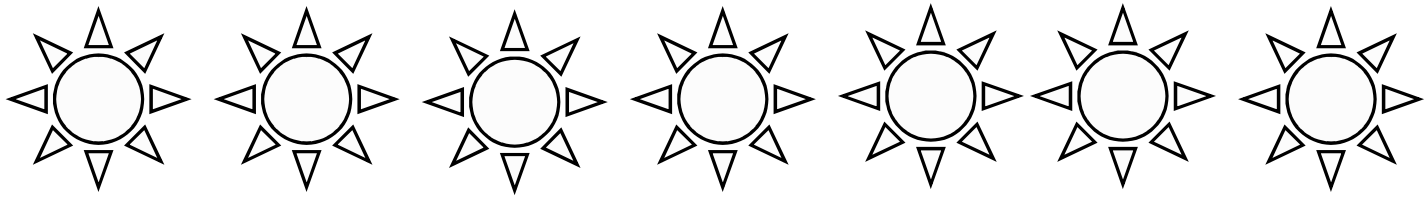


تحلیل ها و سیستم های داده های حجیم

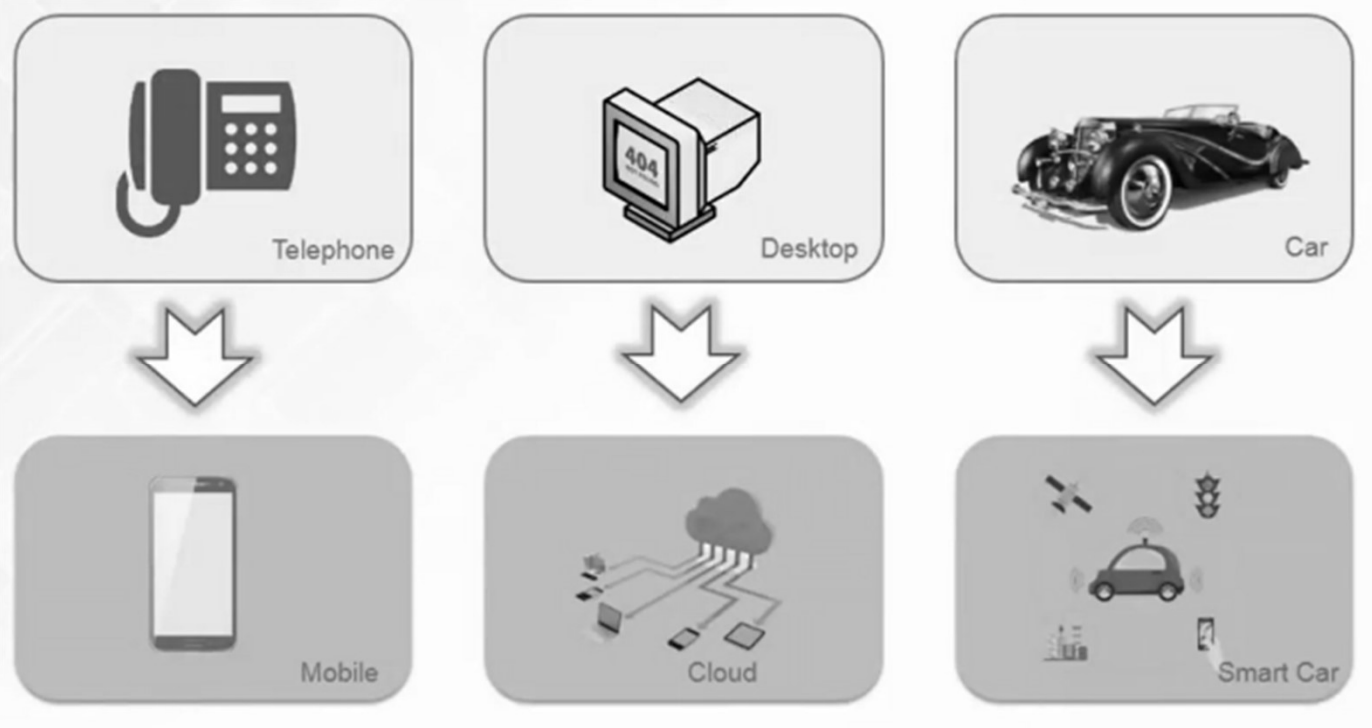


2019

M. Damrudi

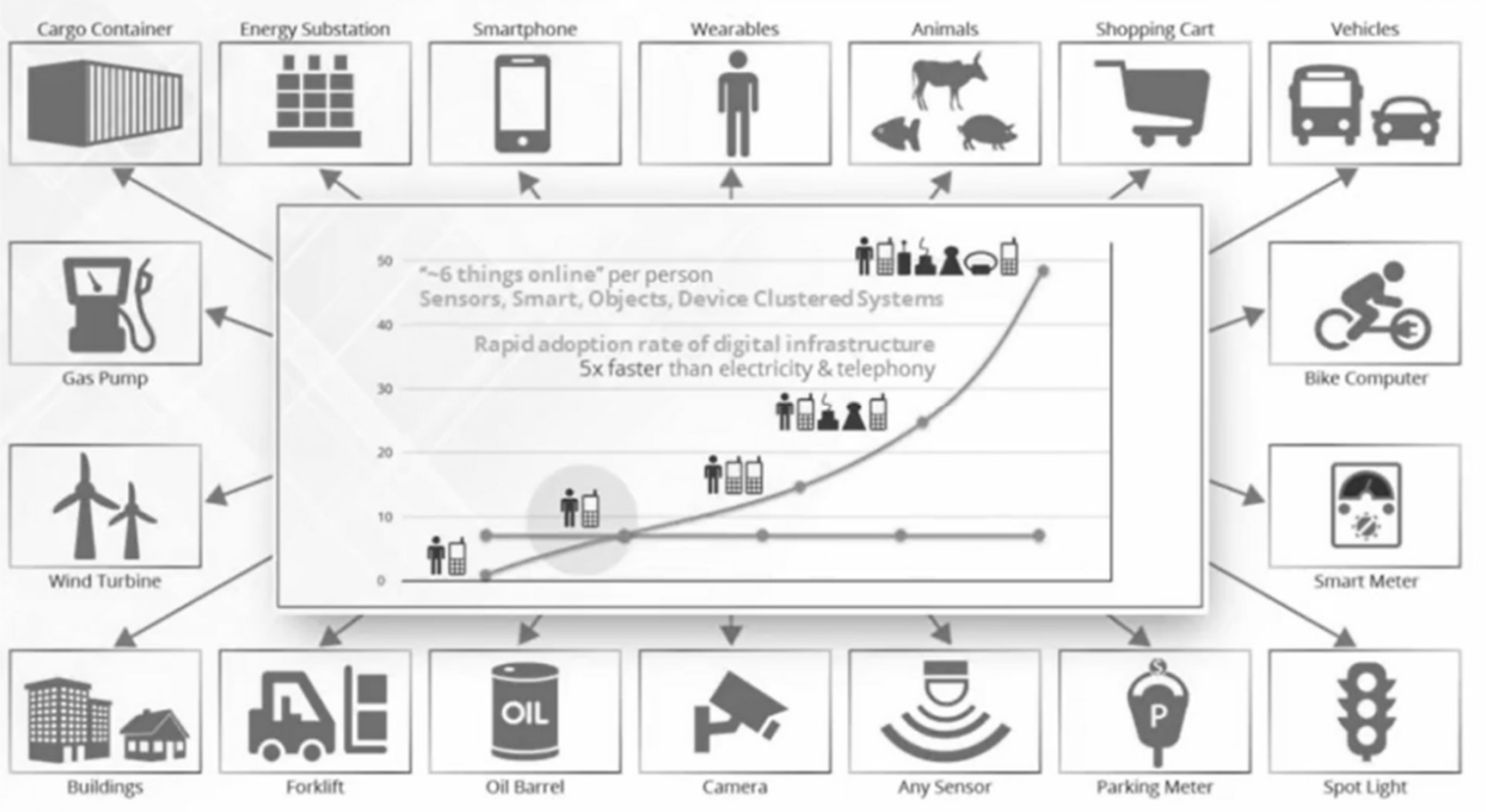
Big Data

❖ سیر تکاملی تکنولوژی



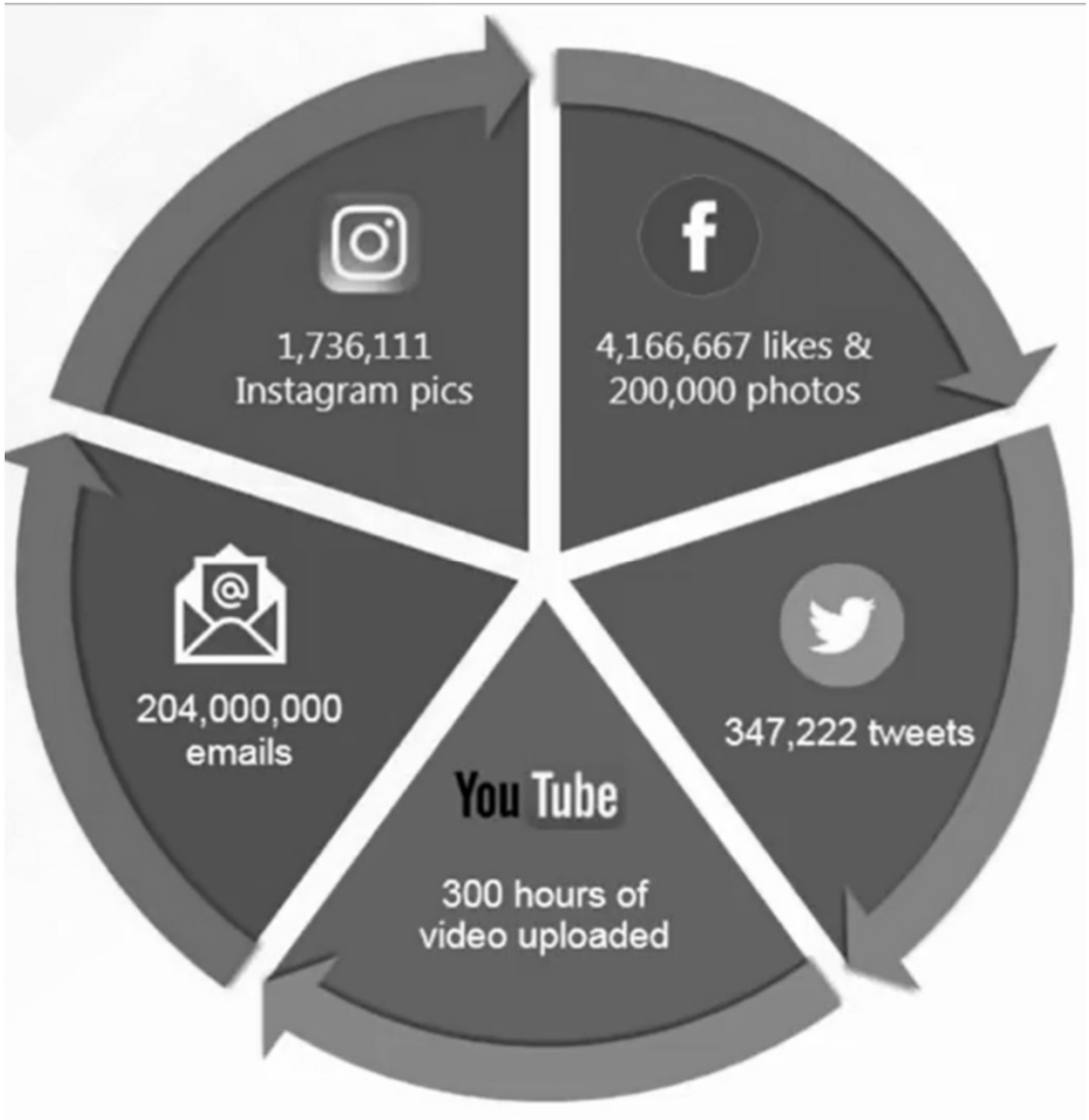
Big Data

- اینترنت اشیا (IOT): ۵۰ بیلیون دستگاه تا سال ۲۰۲۰



Big Data

- شبکه های اجتماعی در هر دقیقه



Big Data

- فاکتورهای دیگر تولید داده

Big Data

- داده های عظیم به مجموعه داده های بسیار بزرگ و پیچیده اشاره دارد که با استفاده از روش های سنتی فناوری اطلاعات و ابزارهای سخت افزاری و نرم افزاری موجود در آن نمی توانند در زمان معقولی درک، گردآوری، مدیریت و پردازش شوند.

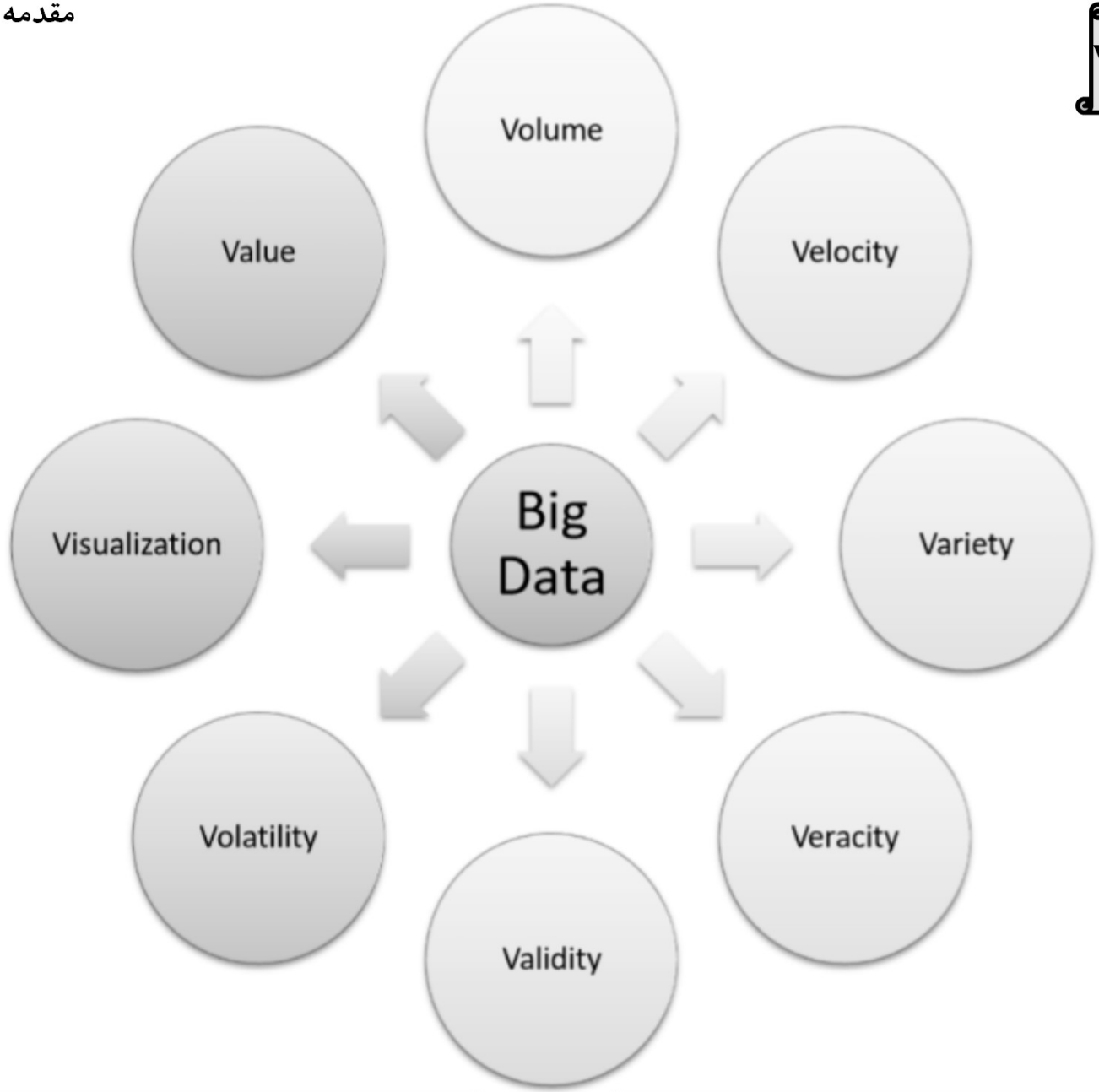
- داده های عظیم روش ها و فناوری های نوینی را جهت جمع آوری، ذخیره و آنالیز داده های غیر ساخت یافته به صورت مقیاس پذیر معرفی می کند.

Big Data

- مدیریت داده ها در Big Data
- به منظور استخراج اطلاعات، کشف دانش و در نهایت تصمیم گیری در خصوص مسائل مختلف کاربردی نیاز است که داده ها به صورت صحیح مدیریت شوند. مدیریت داده ها عموما شامل ۵ فعالیت اصلی میباشد.
  - ✓ جمع آوری
  - ✓ ذخیره سازی
  - ✓ جستجو
  - ✓ به اشتراک گذاری
  - ✓ تحلیل

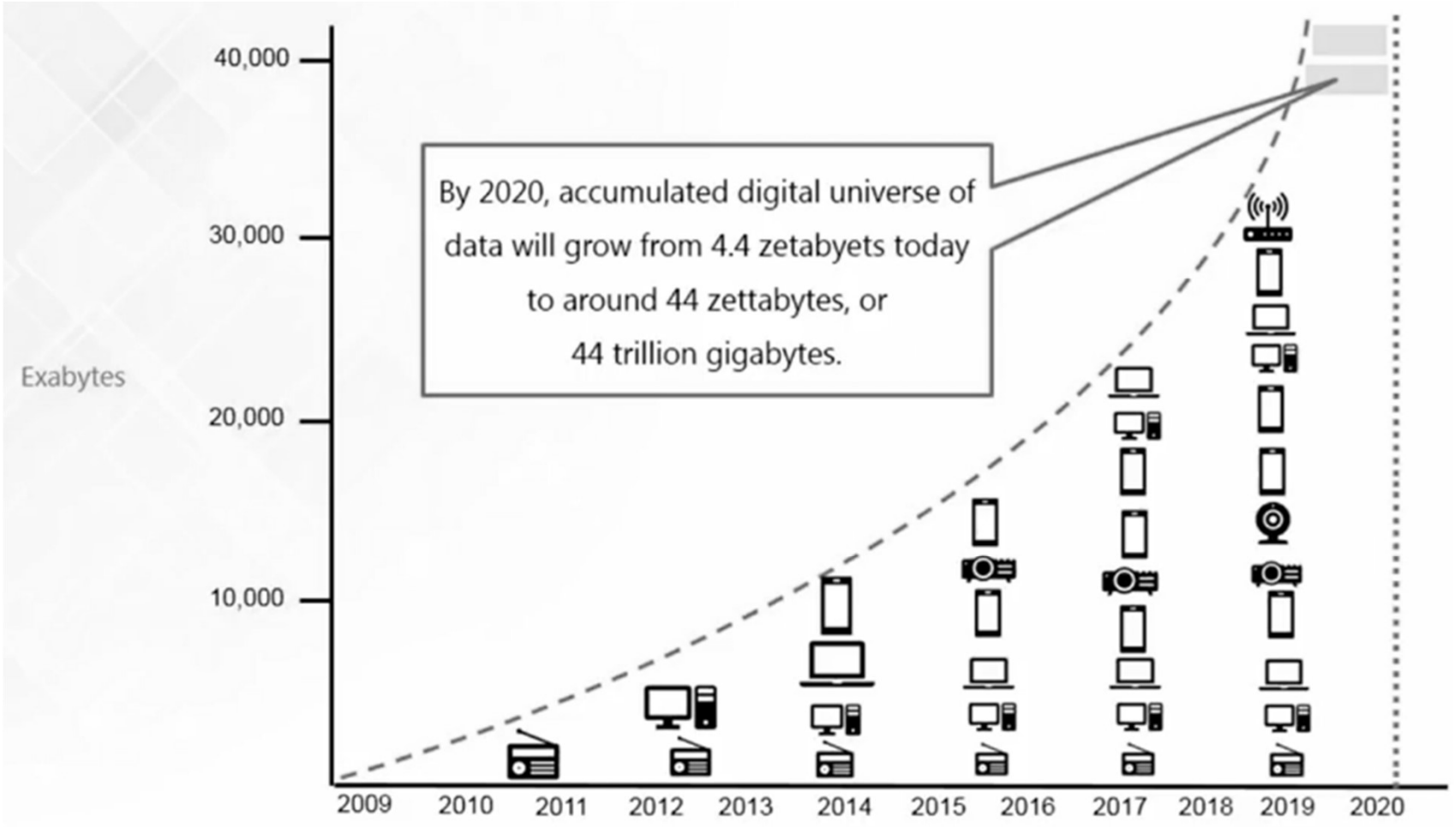
V's of Big Data

- ویژگی ها



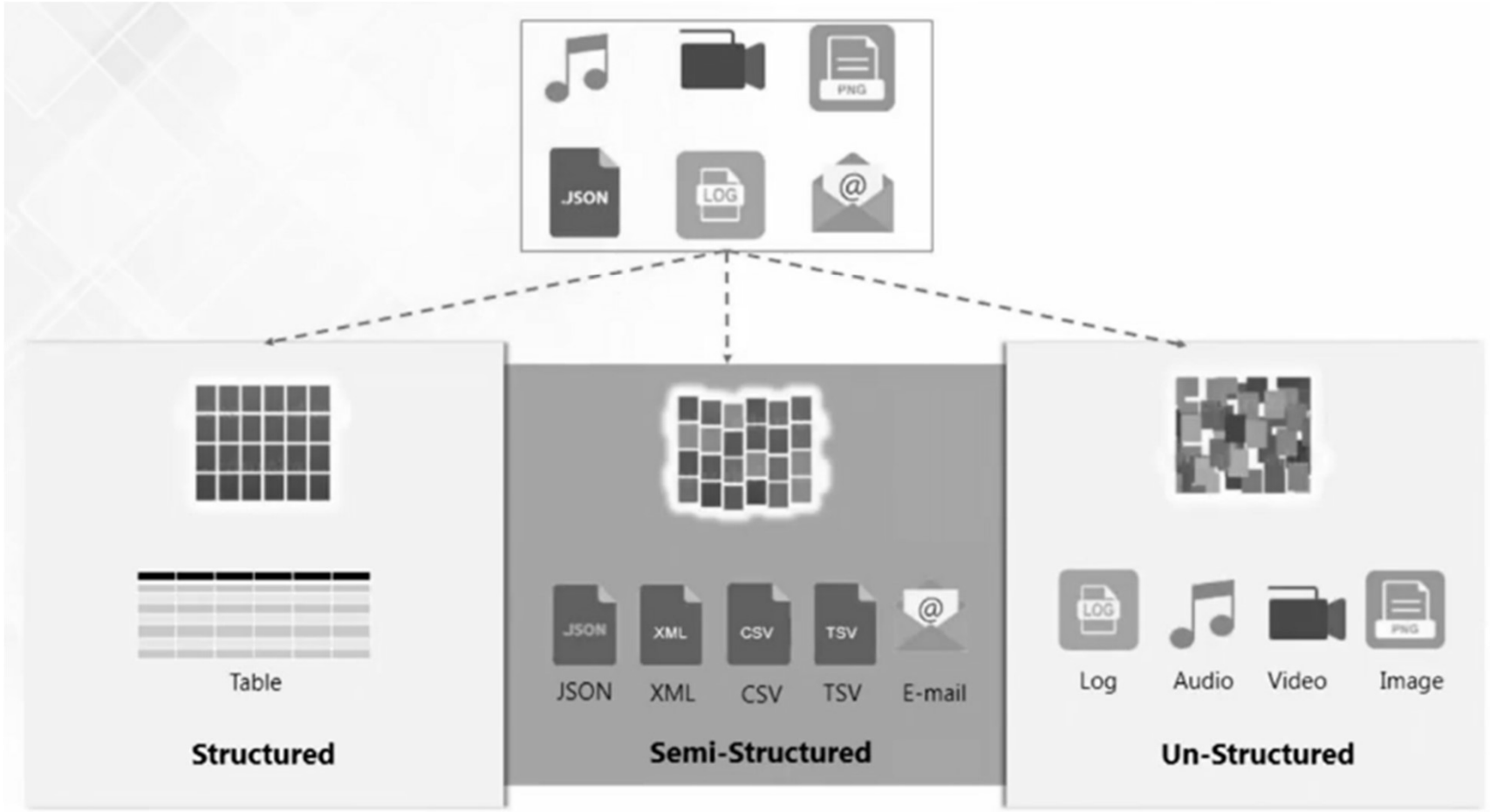
V's of Big Data

Volume ❖



V's of Big Data

❖ Variety

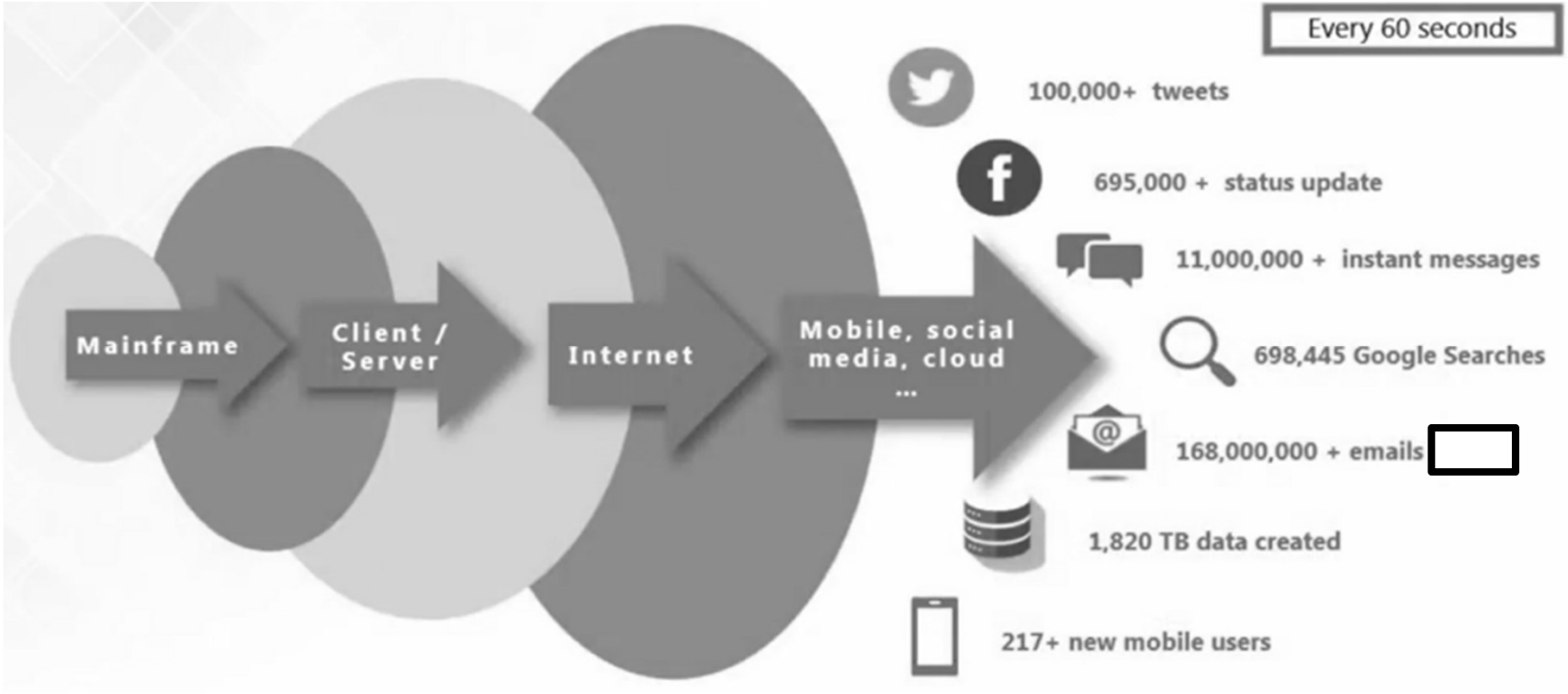


V's of Big Data

- حجم داده (Volume): حجم داده ها به صورت نمایی در حال رشد می باشد. منابع مختلفی نظیر شبکه های اجتماعی، جریان های صوتی، لاگ سرورهای وب، جریان های ترافیک، تراکنش های بانکی، تصاویر ماهواره ای، محتوای صفحات وب، اسناد دولتی و ... وجود دارد که حجم داده بسیار زیادی تولید می کنند.
- تنوع (Variety): انواع داده از منابع متفاوتی تولید می شوند. تنوع در نوع داده بسیار زیاد می باشد که در نتیجه ساختارهای داده ای بسیار زیادی وجود دارد. مثلا در وب، افراد از نرم افزارها و مرورگرهای مختلفی برای ارسال اطلاعات استفاده می کنند. بسیاری از اطلاعات مستقیما از انسان دریافت می شود و بنابراین وجود خطا اجتناب ناپذیر است. این تنوع سبب میشود جامعیت داده تحت تاثیر قرار بگیرد. زیرا هرچه تنوع بیشتری وجود داشته باشد، احتمال بروز خطای بیشتری نیز وجود خواهد داشت. داده ها می توانند structured، unstructured یا semi structured باشند.

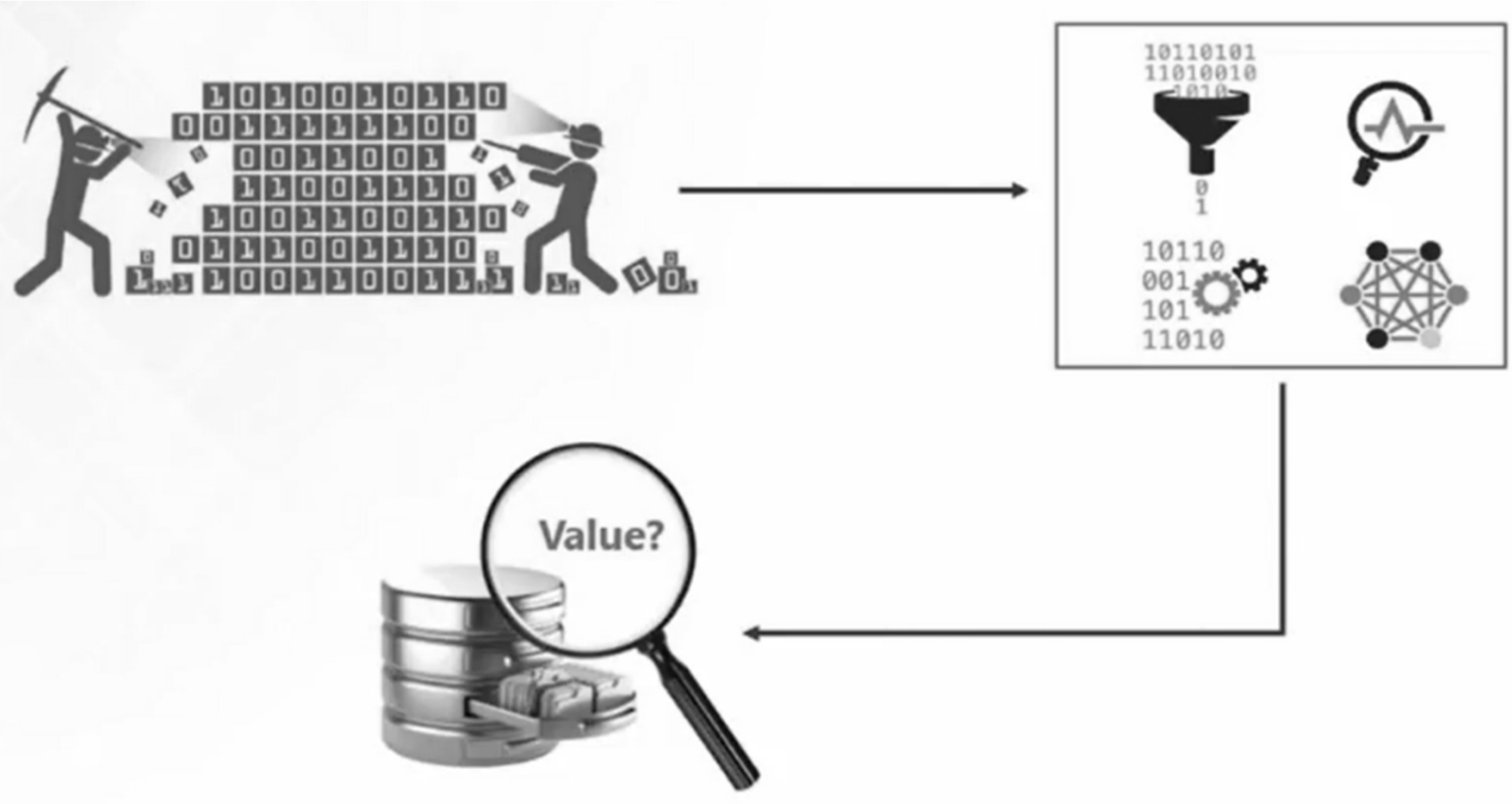
V's of Big Data

Velocity ❖



V's of Big Data

- Value



V's of Big Data

- نرخ تولید (Velocity): داده ها از طریق برنامه های کاربردی و سنسورهای بسیار زیادی که در محیط وجود دارند با سرعت بسیار زیاد و به صورت بلادرنگ تولید می شوند. بسیاری از کاربردها نیاز دارند به محض ورود داده به درخواست کاربر پاسخ دهند. ممکن است در برخی موارد نتوانیم به اندازه کافی صبر کنیم تا مثلا یک گزارش در سیستم برای مدت طولانی پردازش شود. سرعت تولید داده، سرعت پردازش داده و سرعت نمایش داده ها مطرح است.
- ارزش(Value): چگونه داده مفید را استخراج کنیم. این موضوع دلالت بر این دارد که از نظر اطلاعاتی برای تصمیم گیری چقدر داده حائز ارزش است. بعبارت دیگر آیا هزینه ای که برای نگهداری داده و پردازش آنها می شود، ارزش آن را از نظر تصمیم گیری دارد یا نه. معمولا داده ها میتوانند در لایه های مختلف جابجا شوند. لایه های بالاتر به معنای ارزش بیشتر داده می باشند. بنابراین برخی از سازمانها میتوانند هزینه بالای نگهداری مربوط به لایه های بالاتر را قبول کنند.

ابعاد Big Data

- Veracity

Min	Max	Mean	SD
4.3	?	5.84	0.83
2.0	4.4	3.05	50000000
15000	7.9	1.20	0.43
0.1	2.5	?	0.76

ابعاد Big Data

- ❖ صحت(Veracity): با توجه به اینکه داده ها از منابع مختلف دریافت می شوند، ممکن است نتوان به همه آنها اعتماد کرد.
- ✓ مثلا در یک شبکه اجتماعی ممکن است نظرهای زیادی در خصوص یک موضوع خاص ارائه شود. اما اینکه آیا همه آنها صحیح و قابل اطمینان هستند، موضوعی است که نمیتوان به سادگی از کنار آن در حجم بسیار زیادی از اطلاعات گذشت. البته بعضی از تحقیقات این چالش را به معنای حفظ همه مشخصه های داده اصلی بیان کرده اند که باید حفظ شود تا بتوان کیفیت و صحت داده را تضمین کرد. البته تعریف دوم در مولدهای کلان داده صدق میکند تا بتوان داده ای تولید کرد که نشان دهنده ویژگی های داده اصلی باشد.
- ✓ گاهی بعضی داده ها ناسازگار با بقیه داده ها هستند.
- ❖ اعتبار(Validity): با فرض اینکه دیتا صحیح باشد، ممکن است برای برخی کاربردها مناسب نباشد یا به عبارت دیگر از اعتبار کافی برای استفاده در برخی از کاربردها برخوردار نباشد.

V's of Big Data

- نوسان(Volatility): سرعت تغییر ارزش داده های مختلف در طول زمان میتواند متفاوت باشد. در یک سیستم معمولی تجارت الکترونیک، سرعت نوسان داده ها زیاد نیست و ممکن است داده های موجود مثلا برای یک سال ارزش خود را حفظ کنند، اما در کاربردهایی نظیر تحلیل ارز و بورس، داده با نوسان زیادی مواجه هستند و داده ها به سرعت ارزش خود را از دست می دهند و مقادیر جدیدی به خود می گیرند. اگرچه نگهداری اطلاعات در زمان طولانی به منظور تحلیل تغییرات و نوسان داده ها حائز اهمیت است. افزایش دوره نگهداری اطلاعات، مسلما هزینه های پیاده سازی زیادی را دربر خواهد داشت که باید در نظر گرفته شود.
- نمایش(Visualization): یکی از کارهای مشکل در حوزه کلان داده، نمایش اطلاعات است. اینکه کاری کنیم که حجم عظیم اطلاعات با ارتباطات پیچیده، به خوبی قابل فهم و قابل مطالعه باشد از طریق روش های تحلیلی و بصری سازی مناسب اطلاعات امکان پذیری است.

Big Data

❖ Big Data به عنوان یک فرصت (چرا تجزیه و تحلیل Big Data)

Big Data

❖ نیاز به تجزیه و تحلیل Big Data

۱- ایجاد سازمانهای هوشمندتر و کاراتر

Big Data

❖ نیاز به تجزیه و تحلیل Big Data

۲- بهینه کردن عملیات کسب و کار با تحلیل رفتار مشتری

Analysing all the clicks of every visitor on a website

Studying the paths leading them to buy products

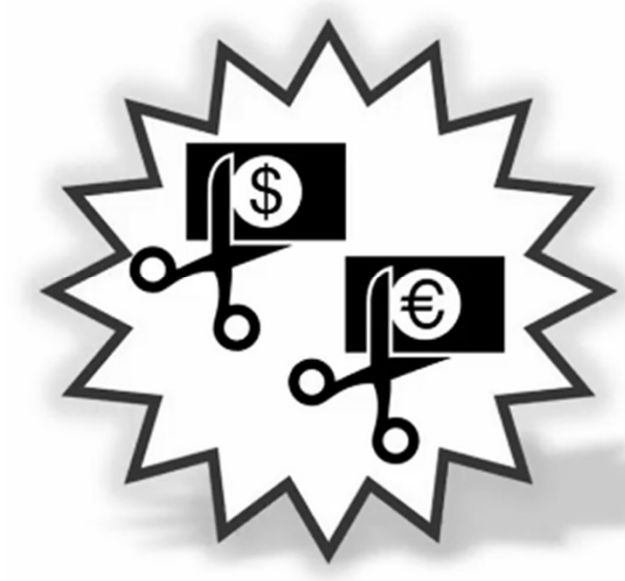
Customer Satisfaction

Amazon uses customer click-stream data and historical purchase data of more than 300 million customers and each user is shown customized results on customized web pages.

Big Data

- نیاز به تجزیه و تحلیل Big Data

۳- کاهش هزینه

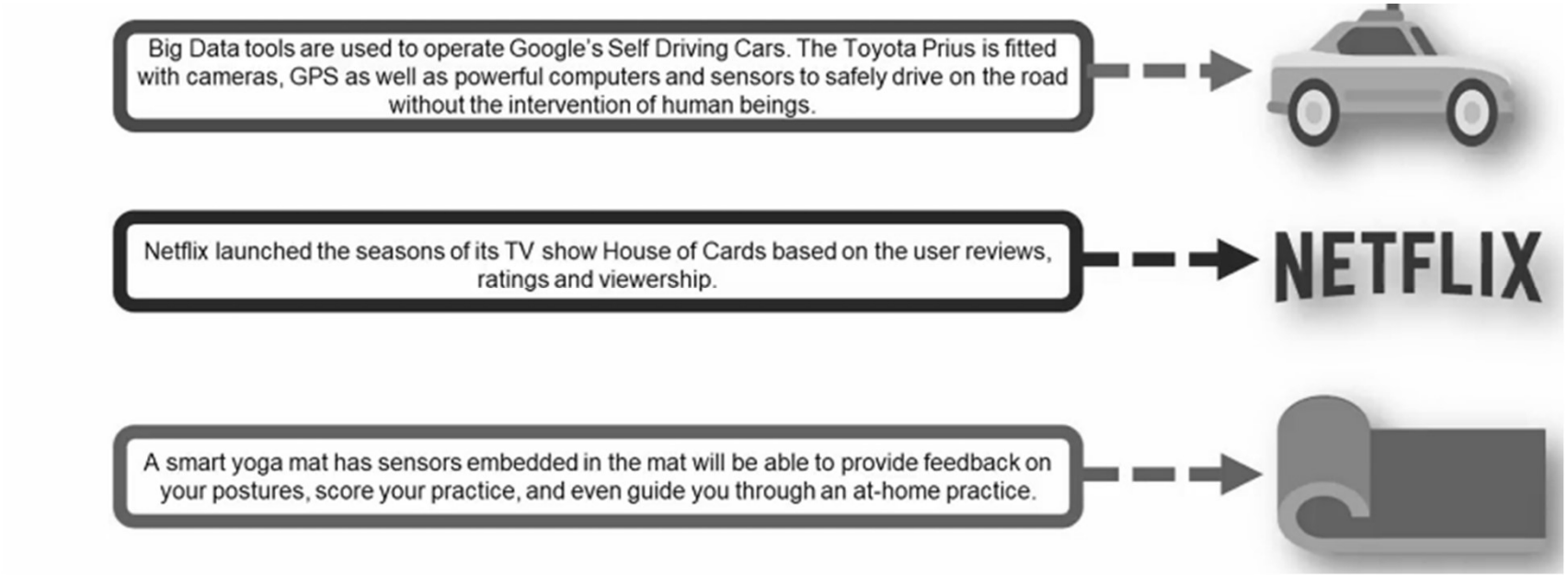


Parkland Hospital uses analytics and predictive modelling to identify high-risk patients and predict likely outcomes once patients are sent home. As a result, Parkland reduced 30-day readmissions for patients with heart failure, by 31 percent, saving $500,000 annually.

Big Data

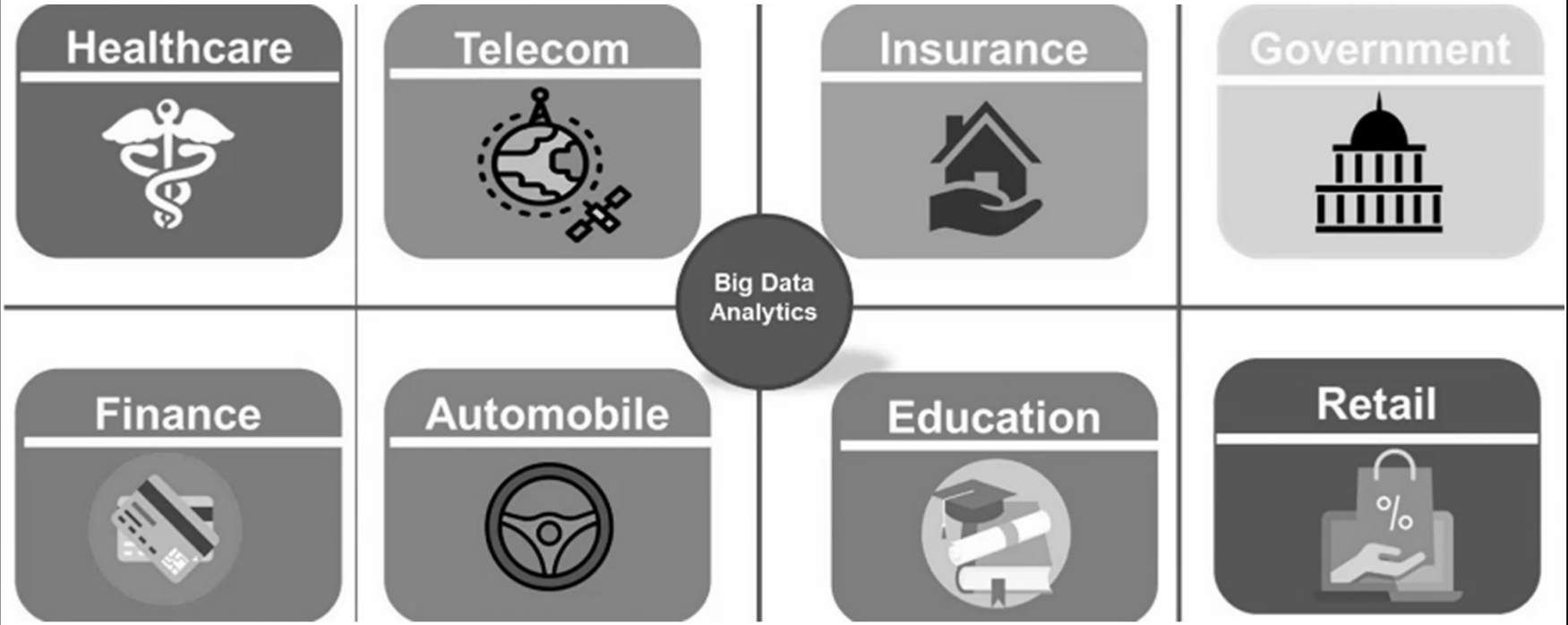
- نیاز به تجزیه و تحلیل Big Data

۴- محصولات نسل بعد



Big Data

- قلمرو تجزیه و تحلیل Big Data



Big Data

❖ مورد استفاده از Big Data

Starbucks uses behavioural analytics to cater to its customers

Starbucks gather a lot of info about their customers' coffee-buying habits from their preferred drinks to what time of day they're usually ordering

The company directs exciting offers and coupons to their customers and ensures to maintain their interest

Big Data

❖ مورد استفاده از Big Data

P&G uses Market Basket Analysis and price optimization to optimize their products

Procter&Gamble

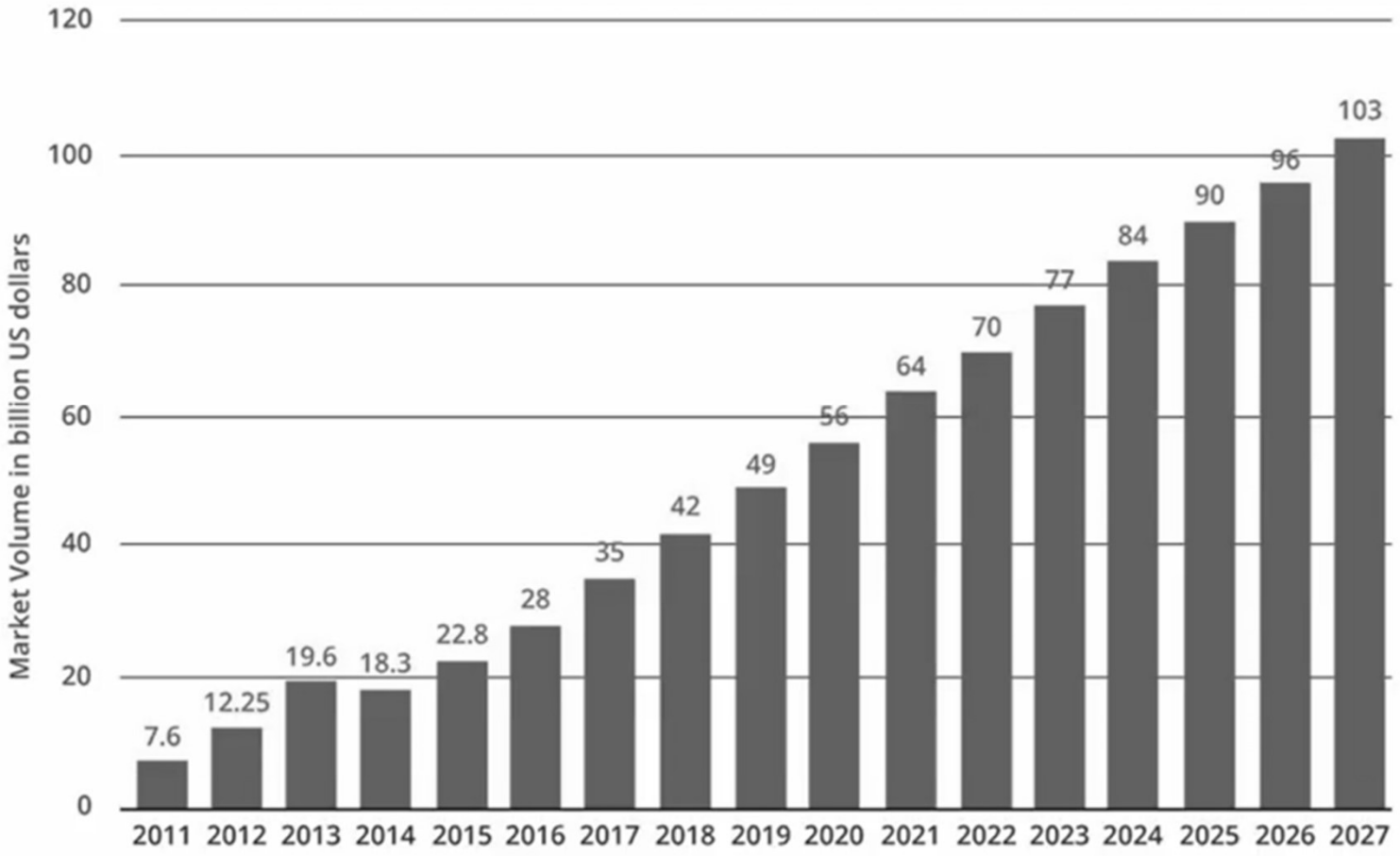
Market Basket Analysis, analyses customer buying habits by finding associations between the different items that customers place in their “shopping baskets”

P&G

The company uses simulation models and predictive analysis in order to create the best design for its products.

Big Data

❖ بازده مالی بازارBig Data

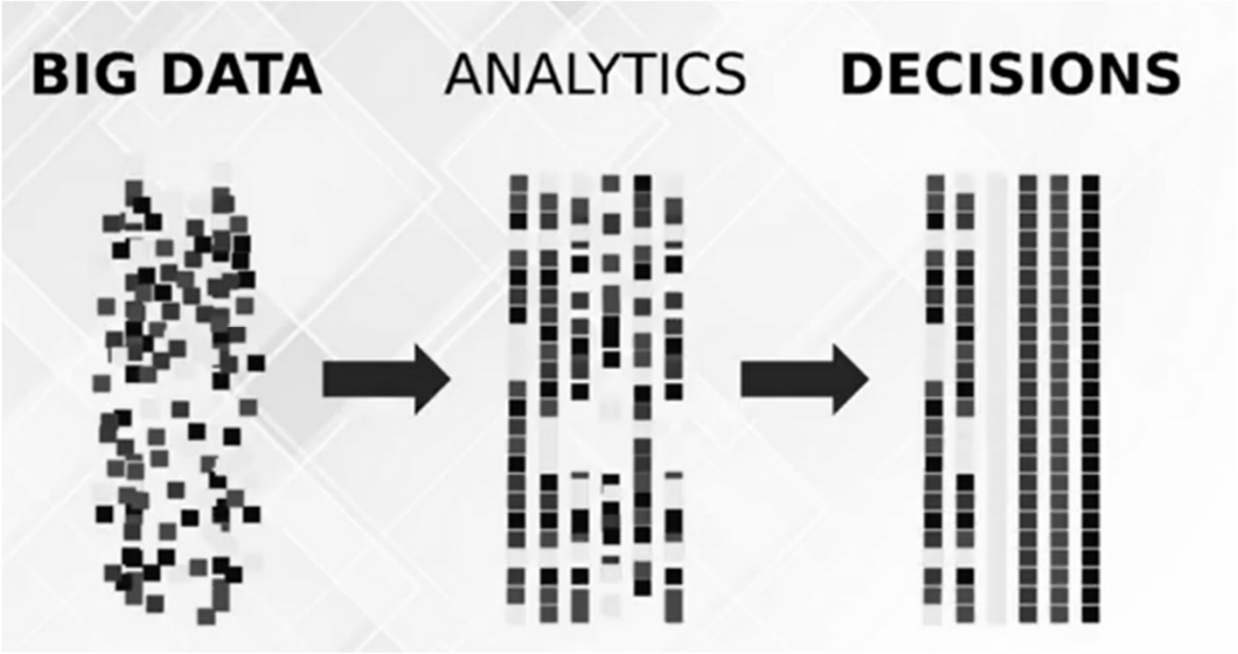


Big Data

❖ تجزیه و تحلیل Big Data چیست؟

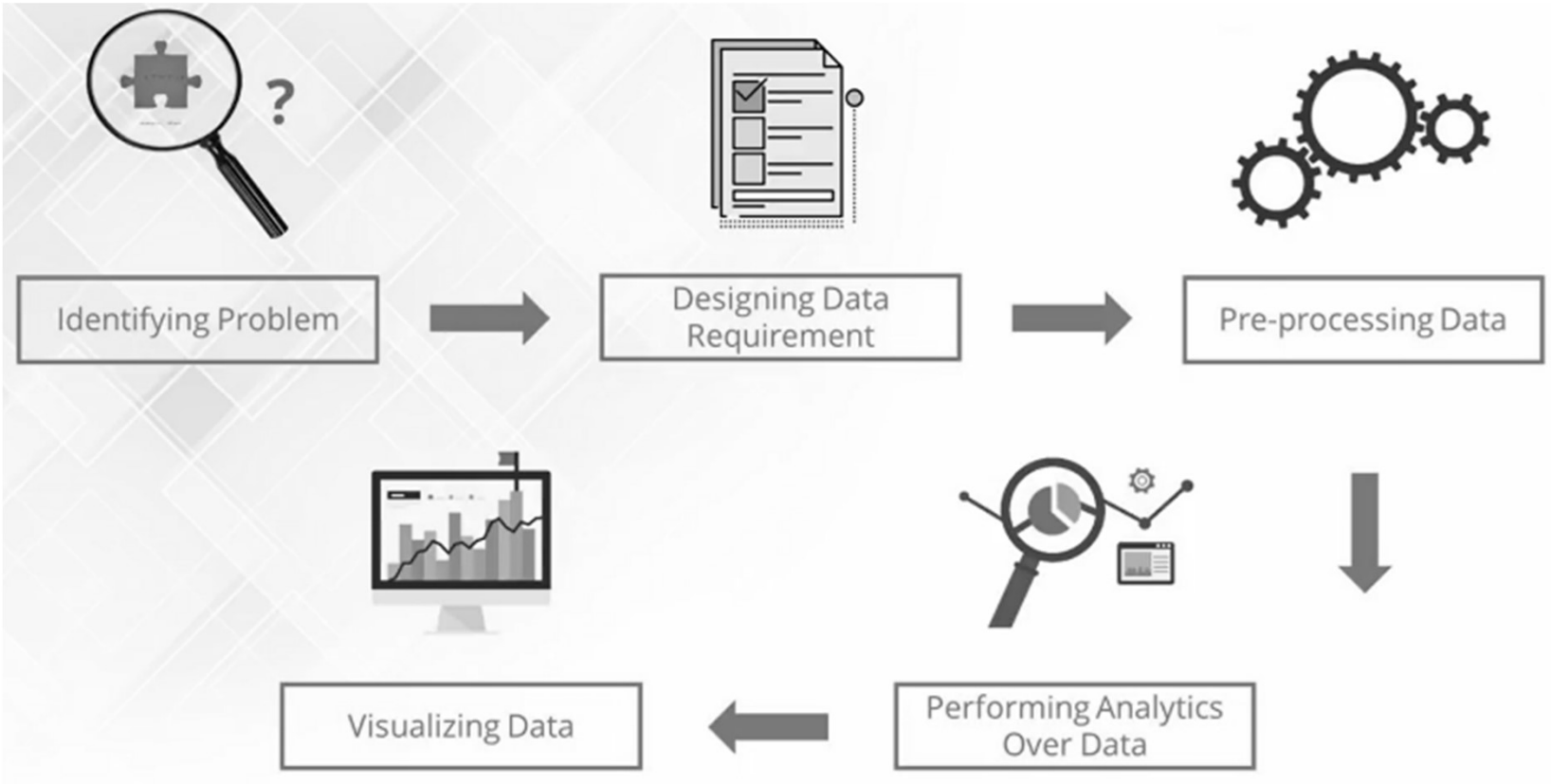
✓ تحلیل داده های حجیم، داده های بزرگ و متفاوت را بررسی می کند تا الگوهای پنهان، وابستگی ها و سایر مفاهیم را آشکار کند.

✓ تحلیل روی داده های عظیم و تصمیم گیری بر اساس آن



Big Data

❖ مراحل تجزیه و تحلیل Big Data

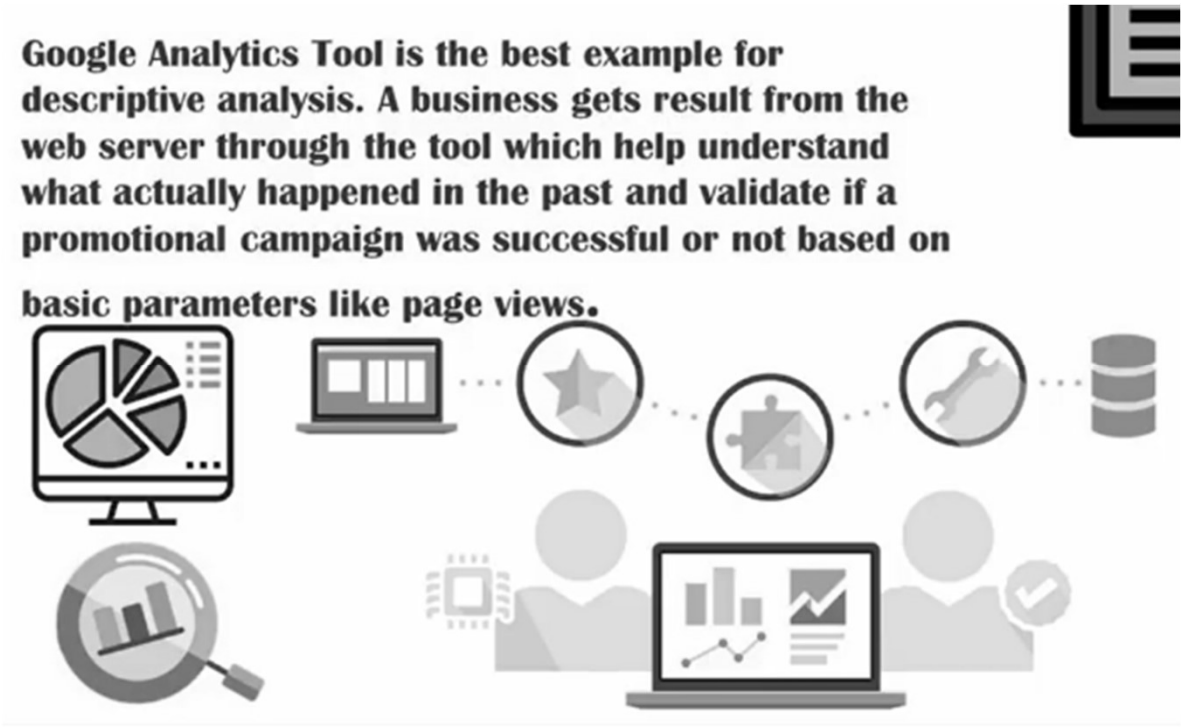


Big Data

- انواع تجزیه و تحلیل Big Data

۱- تجزیه و تحلیل توصیفی (Descriptive Analytics)

- ✓ بر اساس داده ورودی چه اتفاقی در حال رخ دادن است. (NETFLIX, Google Analytics Tool)



Big Data

- انواع تجزیه و تحلیل Big Data

۲- تجزیه و تحلیل پیشگویانه (Predictive Analytics)

- ممکن است در آینده چه اتفاقی رخ بدهد. (Southwest Airlines analyses)

For example, Southwest Airlines analyses sensor data on their planes in order to identify patterns that indicate a potential malfunction, thus allowing the airlines to the necessary repairs before its schedule.

Big Data

- انواع تجزیه و تحلیل Big Data

۳- تجزیه و تحلیل دستورالعملی (Prescriptive Analytics)

- ✓ چه عملی باید انتخاب شود. (Google self-driving car)

Google's self-driving car is a perfect example of prescriptive analytics. It analyses the environment and decides the direction to take based on data.

Big Data

- انواع تجزیه و تحلیل Big Data

۴- تجزیه و تحلیل شناختی (Diagnostic Analytics)

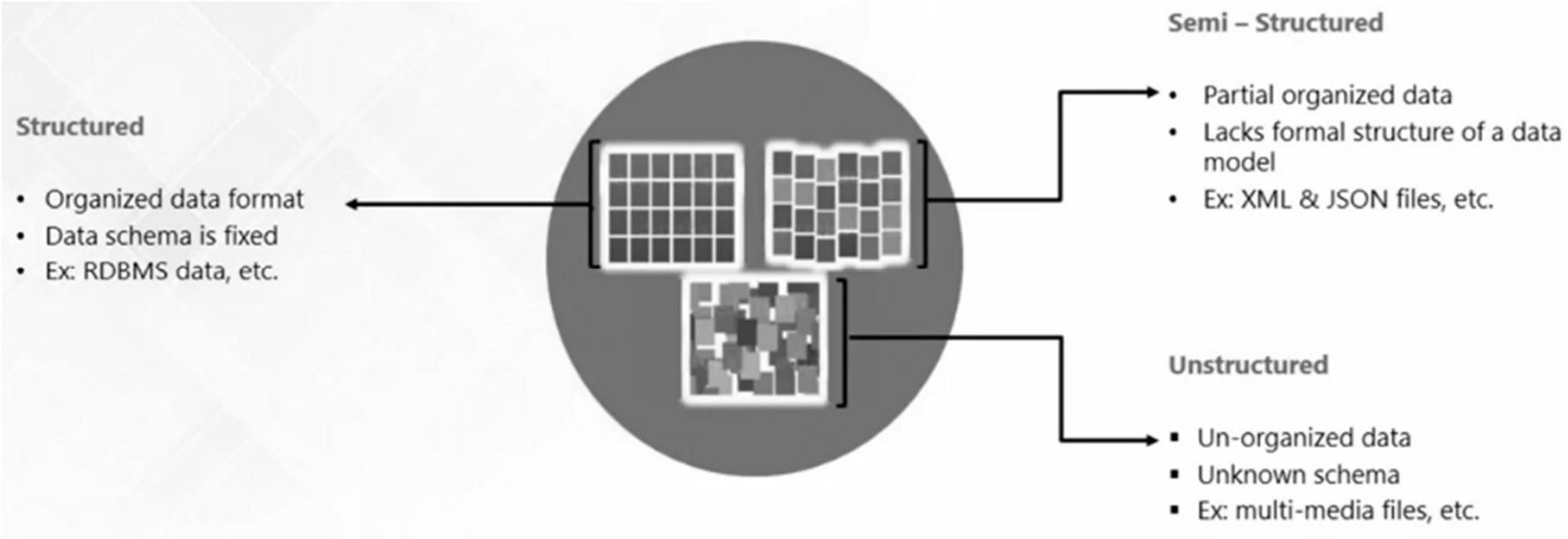
- ✓ چرا اتفاق افتاد. (Social media marketing)

Big Data

- مشکلات Big Data
- نگهداری مجموعه داده های حجیم که به صورت نمایی در حال رشد هستند.
  - ✓ داده هایی که طی دو سال قبل تولید شده اند بیشتر از کل داده هایی است که قبل از آن تولید شده اند.
  - ✓ تا سال ۲۰۲۰، کل داده های دیجیتال تقریبا ۴۴ زتابایت خواهد شد.
  - ✓ تا سال ۲۰۲۰، در هر ثانیه برای هر شخص حدود ۱.۷ مگابایت اطلاعات تولید می شود.

Big Data

- مشکلات Big Data
- پردازش داده هایی با ساختار پیچیده

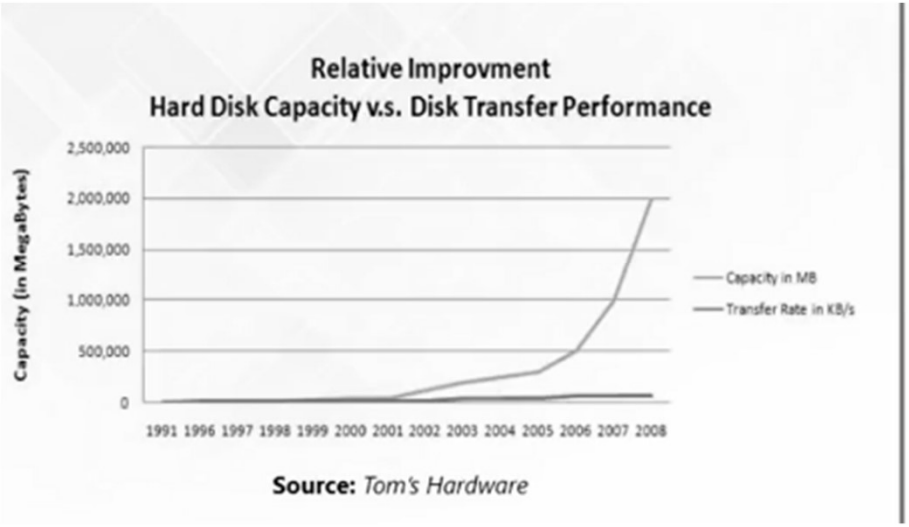


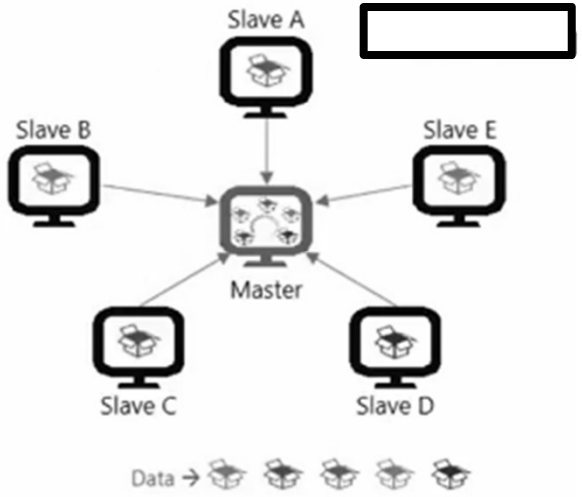
Big Data

- مشکلات Big Data
- پردازش سریعتر داده.

رساندن حجم بسیار زیادی داده به واحد محاسبه تبدیل به گلوگاه می شود

داده بسیار سریعتر از سرعت نوشتن/خواندن دیسک رشد می کند

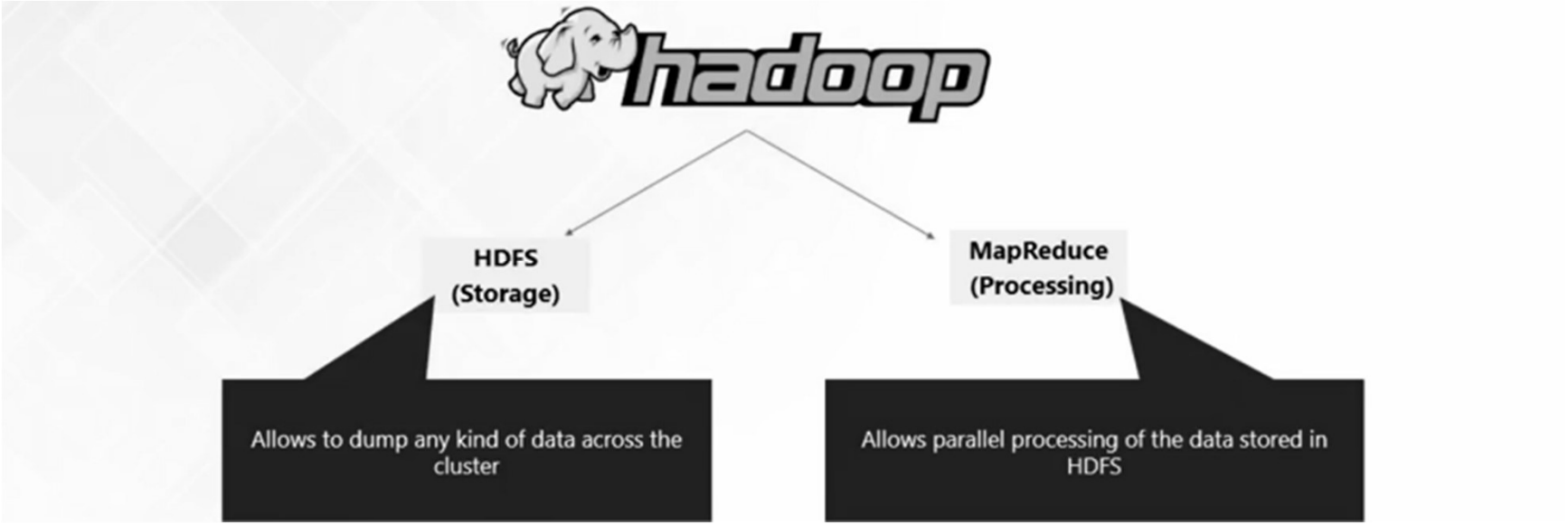




Big Data

- راه حل مشکلات Big Data
- Hadoop

هادوپ چارچوبی است که به ما جازه می دهد مجموعه داده های بزرگ را به روش موازی و توزیع شده ذخیره و پردازش کنیم.



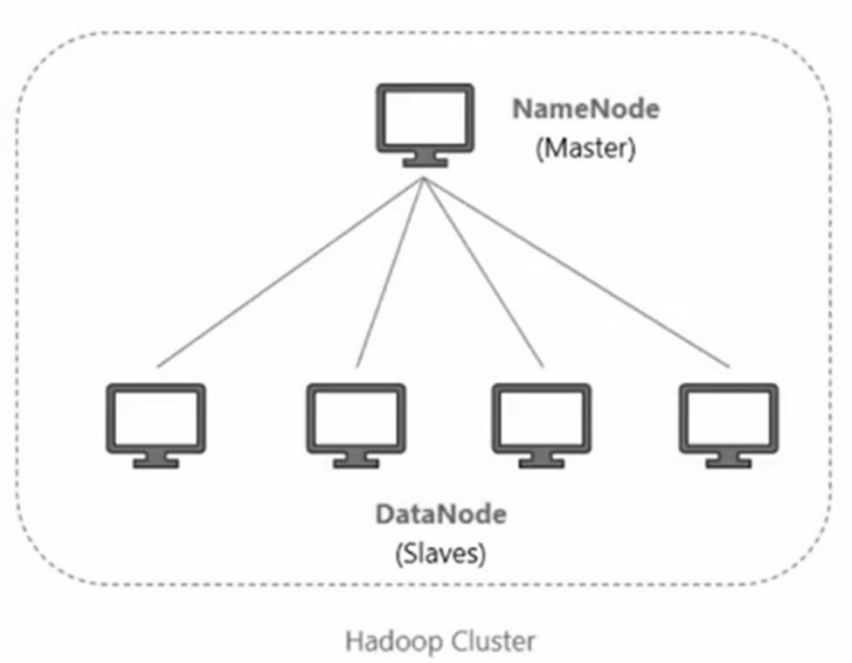
- Hadoop را می توان به یک سیستم عامل تشبیه کرد که طراحی شده تا بتواند حجم زیادی از داده ها را بر روی ماشین های مختلف پردازش و مدیریت کند.

Big Data

❖ Hadoop Distributed File System

HDFS سطحی از انتزاع را روی منابع ایجاد می کند بطوری که می توانیم تمام HDFS را به عنوان یک واحد یکپارچه ببینیم.

✓ HDFS دو جزء پایه دارد: NameNode و DataNode

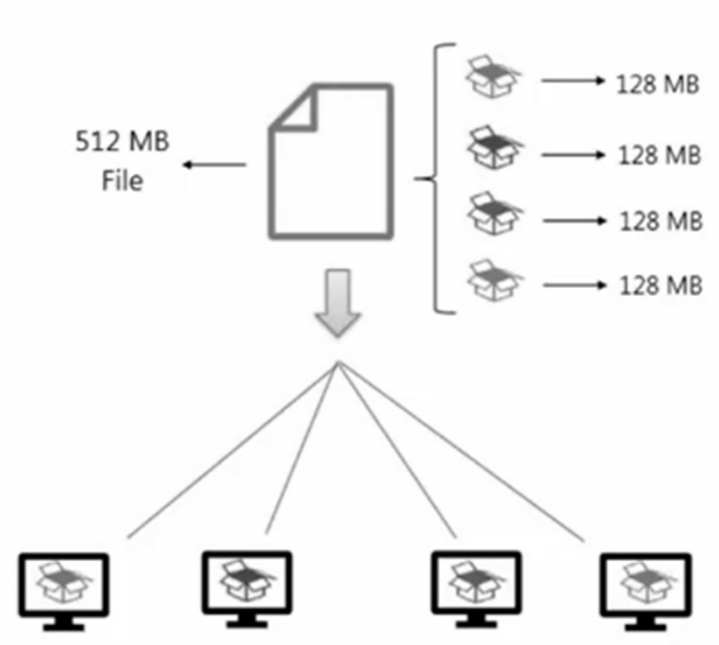


❖ NameNode نود اصلی است که شامل متا داده درباره داده ای است که ذخیره شده است.

❖ داده در DataNode ذخیره می شودکه سخت افزاری مناسب در محیط توزیع شده است.

Big Data

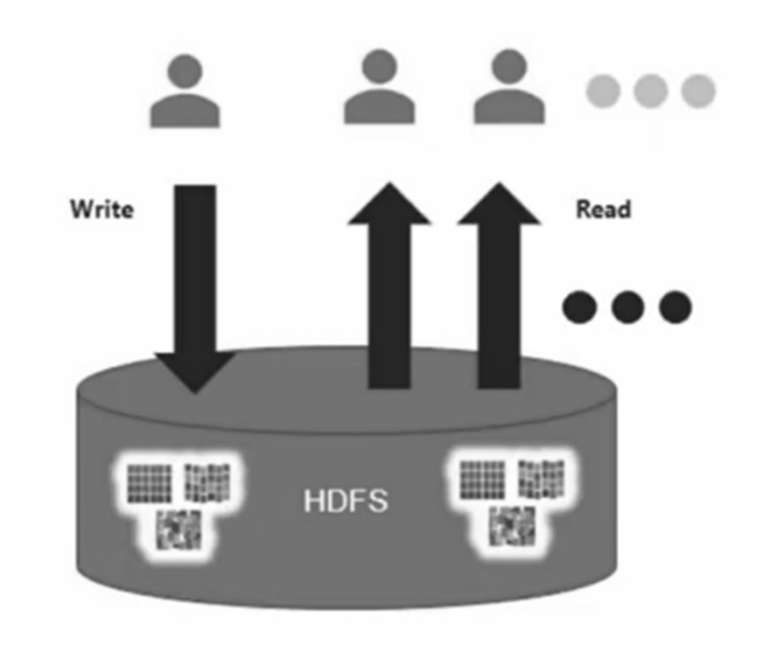
- ❖ مشکل نگهداری مجموعه داده های حجیم که به صورت نمایی در حال رشد هستند.
- ✓ راه حل: HDFS



- ❖ واحد ذخیره هادوپ
- ❖ که یک سیستم فایل توزیع شده است
- ❖ فایل ها (داده های ورودی) را به واحدهای کوچکتری تقسیم کرده و در کلاستر ذخیره می کند.
- ❖ مطابق با هر نیازی مقیاس پذیر است.

Big Data

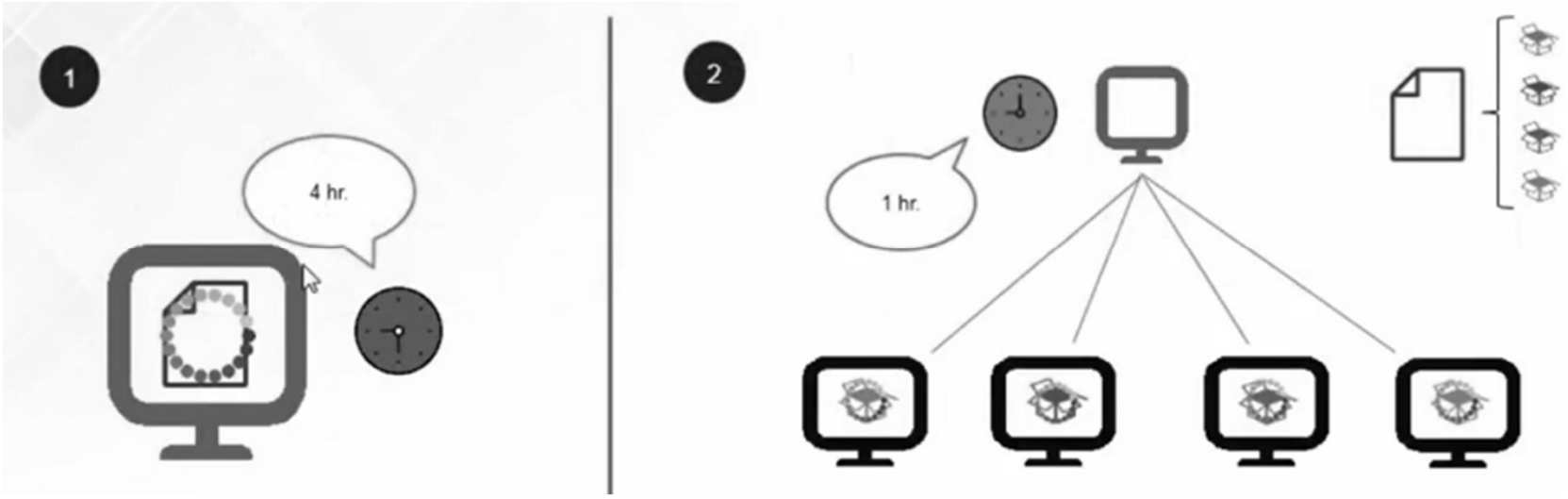
- ❖ مشکل نگهداری داده های متفاوت (داده هایی با ساختار پیچیده)
- ✓ راه حل: HDFS



- ❖ اجازه ذخیره هر نوع داده (از قبیل structured، semi ،unstructured structured) را می دهد.
- ❖ از WORM(Write Once Read Many) تبعیت می کند.
- ❖ هنگام ریختن داده ارزیابی شما (Schema) انجام نمیگیرد.

Big Data

- مشکل پردازش سریعتر داده
- ✓ راه حل: Hadoop MapReduce
- امکان پردازش موازی داده موجود در HDFS را فراهم می کند.
- اجازه پردازش محلی داده را میدهد، برای مثال هر نود با بخشی از داده که در آن ذخیره شده است کار می کند.



Big Data

- Hadoop چیست؟
- Apache Hadoop یک چارچوب نرم افزاری open source است که برای توسعه برنامه های کاربردی پردازش داده ها که در محیط محاسباتی توزیع شده اجرا می شوند، استفاده می شود.
- برنامه های کاربردی ساخته شده با استفاده از HADOOP بر روی مجموعه داده های بزرگ توزیع شده بر روی خوشه های رایانه های رایج اجرا می شود.
- این رایانه ها، رایانه های ارزان قیمت و به طور گسترده ای در دسترس هستند.
- اینها عمدتا برای دستیابی به قدرت محاسباتی بیشتر با هزینه کم مفید هستند.

Big Data

- مشابه داده هایی که در سیستم فایل محلی سیستم کامپیوتری شخصی، در Hadoop، داده ها در یک سیستم فایل توزیع شده قرار می گیرد که سیستم فایل توزیع شده Hadoop نامیده می شود.
- مدل پردازش بر اساس مفهوم Data Locality است که منطق محاسباتی به گره های خوشه ای (سرور) حاوی داده ها ارسال می شود.
- این منطق محاسباتی یک نسخه کامپایل شده از برنامه ای است که در زبانی سطح بالا مانند جاوا نوشته شده است.
- این برنامه داده های ذخیره شده در Hadoop HDFS را پردازش می کند.

Big Data

❖ Apache Hadoop شامل دو زیر پروژه است:

✓ Hadoop MapReduce: MapReduce یک مدل محاسباتی و چارچوب نرم افزاری برای نوشتن برنامه هایی است که در Hadoop اجرا می شوند. این برنامه های MapReduce قادر به پردازش داده های بیشمار به صورت موازی در خوشه های بزرگ گره های محاسباتی هستند.

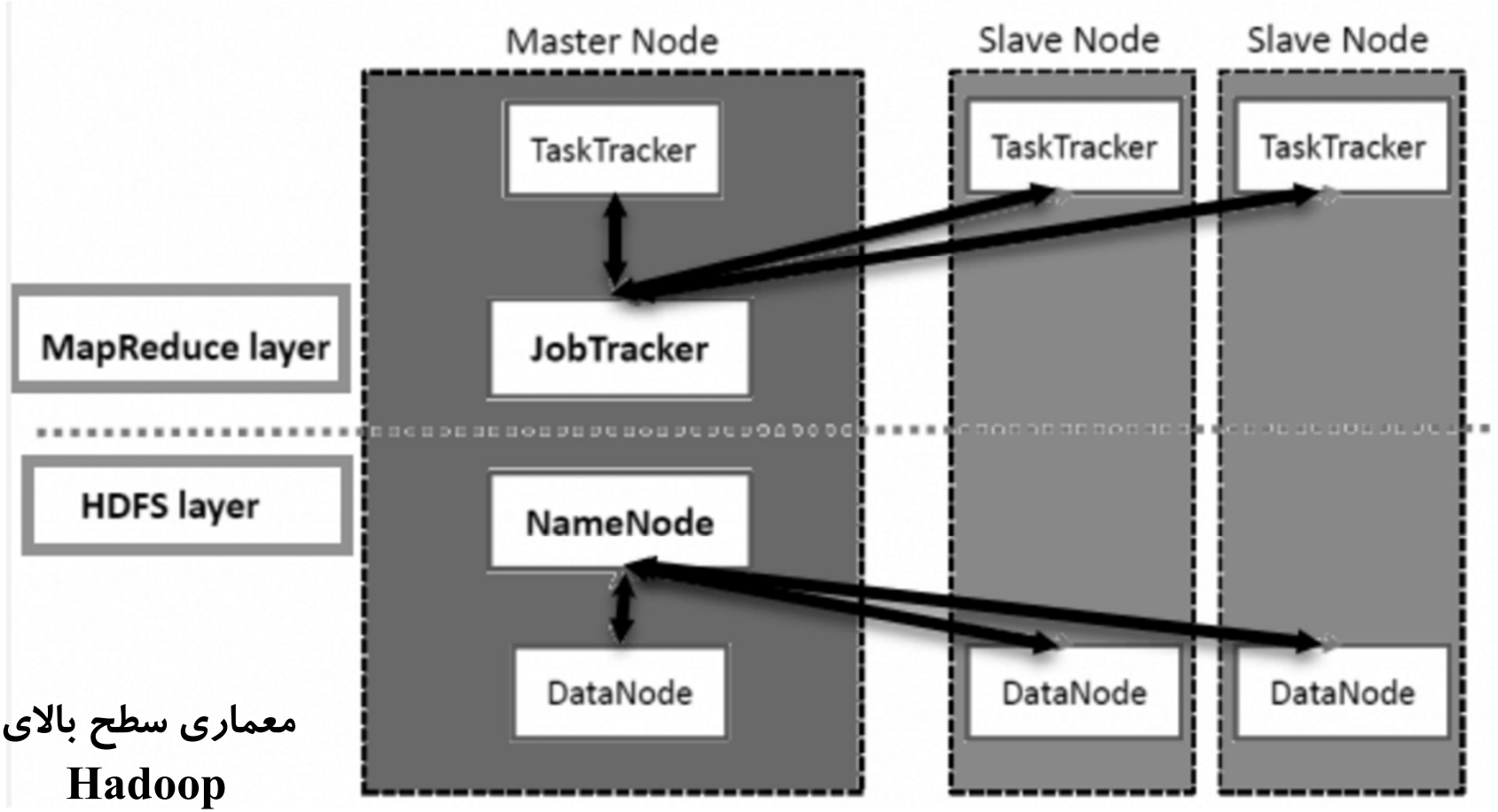
✓ HDFS (Hadoop File System Distributed): HDFS مسئول بخش ذخیره سازی برنامه های Hadoop است. برنامه های MapReduce از داده های HDFS استفاده می کنند. HDFS چندین کپی از بلوک های داده ایجاد می کند و آنها را در گره های محاسباتی یک خوشه توزیع می کند. این توزیع محاسبات قابل اعتماد و بسیار سریع را مهیا می کند.

Big Data

- گرچه Hadoop به خاطر MapReduce و سیستم فایل توزیع شده آن - HDFS شناخته شده است، این اصطلاح برای خانواده ای از پروژه های مرتبط استفاده می شود که زیر مجموعه محاسبات توزیع شده و پردازش داده ها در مقیاس بزرگ قرار می گیرند. سایر پروژه های اصلی مربوط به Hadoop در Apache عبارتند از:
  - ✓ Hive
  - ✓ Hbase
  - ✓ Mahout
  - ✓ Sqoop
  - ✓ Flume
  - ✓ ZooKeeper

Big Data

❖ معماری Hadoop



معماری سطح بالای Hadoop

Big Data

- معماری Hadoop
- Hadoop دارای معماری Master-Slave برای ذخیره داده ها و پردازش توزیع شده داده ها با استفاده از روش MapReduce و HDFS است.
- NameNode:
  - NameNode نمایاننده فایلها و دایرکتوریهای مورد استفاده در فضای نام است.
- DataNode:
  - DataNode به شما کمک می کند تا وضعیت یک گره HDFS را مدیریت کنید و به شما اجازه می دهد با بلاک ها ارتباط برقرار کنید.

Big Data

- معماری Hadoop
- MasterNode:

✓ گره اصلی (MasterNode) اجازه می دهد تا پردازش موازی داده ها را با استفاده از Hadoop MapReduce اجرا کنید.

- Slave node:

✓ گره های slave ماشین های دیگری در خوشه Hadoop هستند که به شما امکان می دهند داده ها را برای انجام محاسبات پیچیده ذخیره کنید. علاوه بر این، همه گره های slave با Task Trackerو DataNode همراه هستند. این به شما اجازه می دهد تا فرآیندها را به ترتیب با NameNode و Job Tracker هماهنگ کنید.

- در Hadoop، سیستم master یا slave می تواند بصورت cloud و یا on-premise تنظیم شود.

Big Data

- معماری Hadoop

- هر Master Node از دو بخش بنامهای Name Node و Job Tracker تشکیل شده است. Name Node، وظیفه نظارت و هماهنگی بر نحوه ذخیره کردن داده ها (HDFS) و Job Tracker نیز وظیفه تعیین نحوه اجرای پردازش موازی بر روی یک مجموعه داده با استفاده از دستورات MapReduce را بر عهده دارد. سیستمهایی که درنقش Slave Node در شبکه هادوپ تعریف میشوند، اکثریت کامپیوترهای شبکه هادوپ را تشکیل داده، وظیفه اجرای اصلی دستورات ذخیره سازی و پردازشهای موازی بر عهده آنها گذاشته شده است.

- هر Slave Node از دو بخش بنامهای Data Node و Task Tracker تشکیل خواهد شد که اطلاعات و دستورها را از سیستمهای گروه Master Node دریافت کرده و پردازشهای لازم را بر روی آنها انجام میدهد. در واقع Task Tracker یک زیر مجموعه از Job Tracker و Data Node نیز یک زیر مجموعه از Name Node بوده، و دستورات خود را مستقیماً از لایه های بالاتر، دریافت میکنند.

Big Data

- ویژگیهای Hadoop

- مناسب برای تجزیه و تحلیل داده های بزرگ

- ✓ از آنجا داده های بزرگ ذاتا به صورت توزیع شده و ساخت نیافته می باشند، خوشه های HADOOP بهترین تناسب را برای تجزیه و تحلیل داده های بزرگ دارند.
- ✓ از آنجایی که منطق پردازش (نه داده واقعی) است که میان گره های محاسباتی جریان دارد، پهنای باند شبکه کمتری مصرف می شود. این مفهوم به عنوان مفهوم محلی بودن داده (data locality concept) نامیده می شود که به افزایش کارایی برنامه های مبتنی بر Hadoop کمک می کند.

Big Data

❖ ویژگیهای Hadoop

❖ مقیاس پذیری (Scalability)

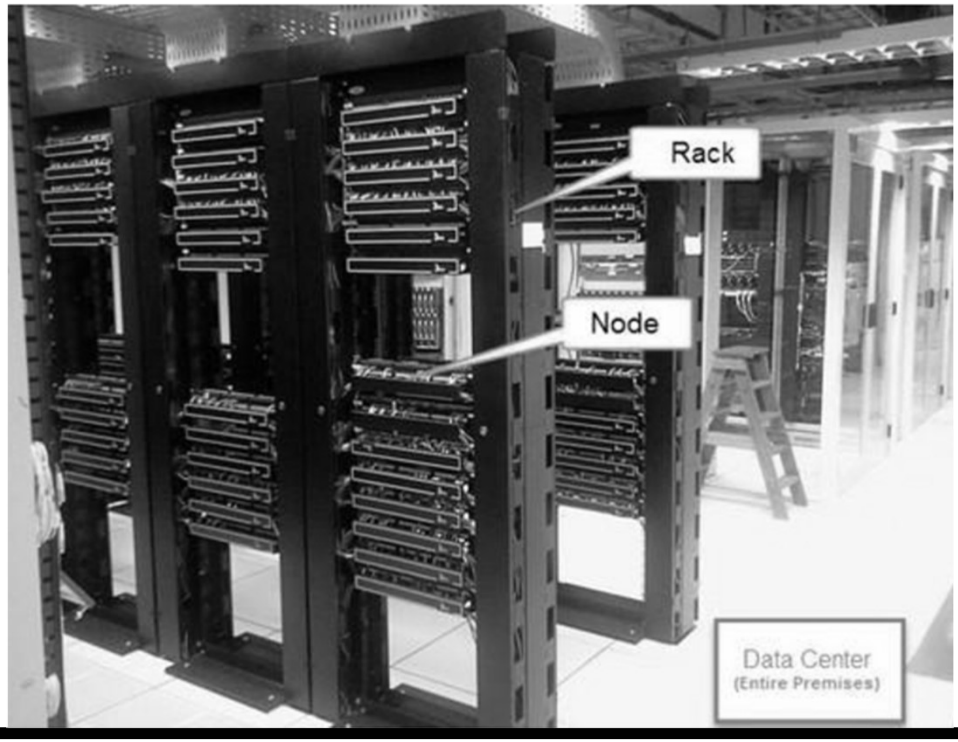
✓ خوشه های HADOOP به راحتی می توانند با اضافه کردن گره خوشه ای جدید به هر میزان مقیاس پذیر شوند که درنتیجه به داده های بزرگ اجازه رشد می دهد. همچنین، مقیاس پذیر کردن نیازی به تغییر در منطق برنامه ندارد.

❖ تحمل خطا (Fault Tolerance)

✓ اکوسیستم HADOOP یک ساز و کار تغذیه برای تکثیر داده های ورودی برروی سایر گره های خوشه دارد. به این ترتیب، در صورت خرابی یک گره خوشه، پردازش داده ها همچنان می تواند با استفاده از داده های ذخیره شده در گره خوشه دیگر ادامه یابد.

Big Data

- توپولوژی شبکه در Hadoop
- زمانی که اندازه خوشه Hadoop رشد می کند، توپولوژی (پیکربندی) شبکه کارایی خوشه Hadoop را تحت تأثیر قرار دهد. علاوه بر کارایی، توجه به نرخ بالای در دسترس بودن و رسیدگی به خرابی ها نیز مورد نیاز است. برای رسیدن به این Hadoop، تشکیل خوشه از توپولوژی شبکه استفاده می کند.

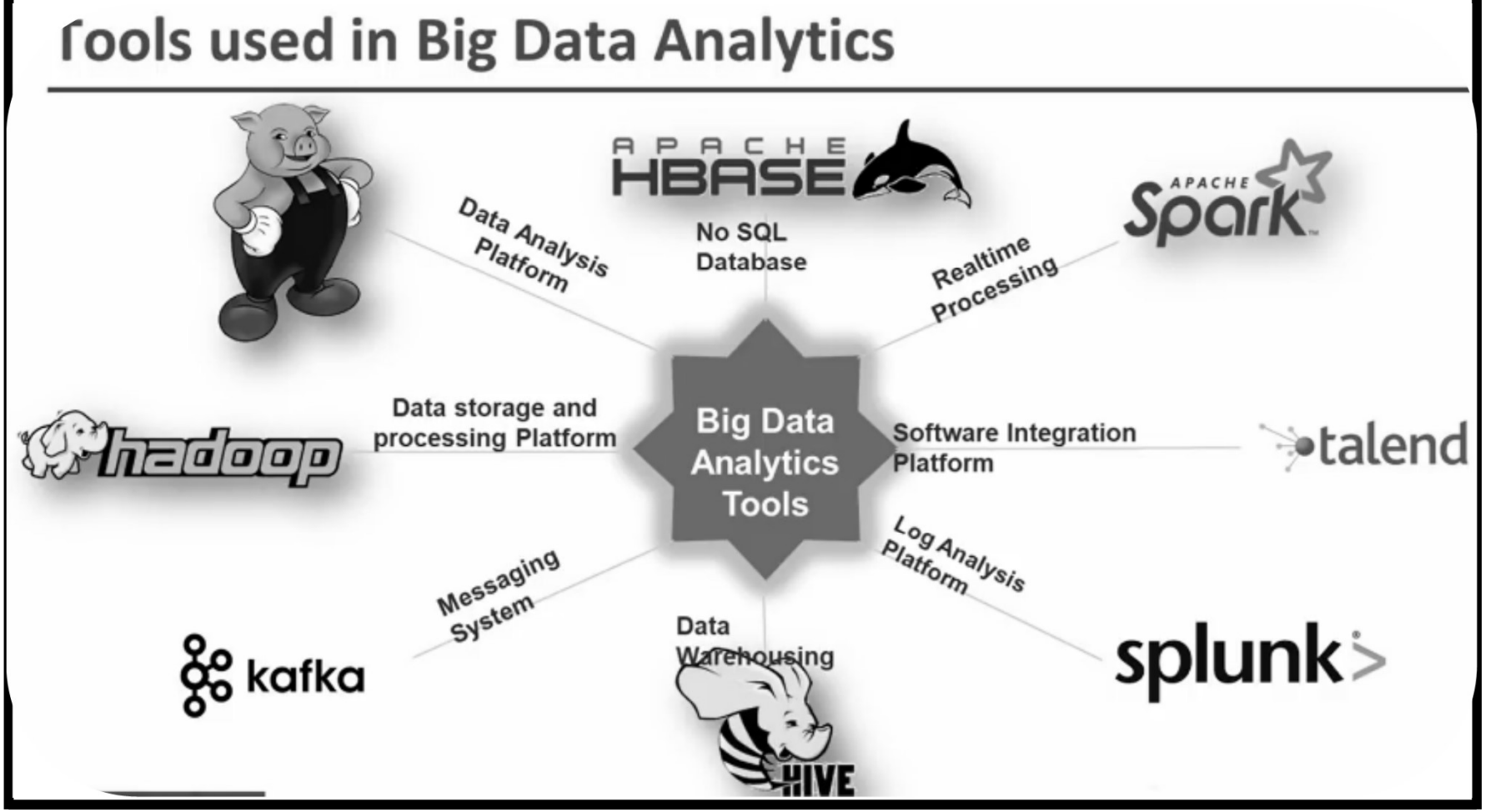


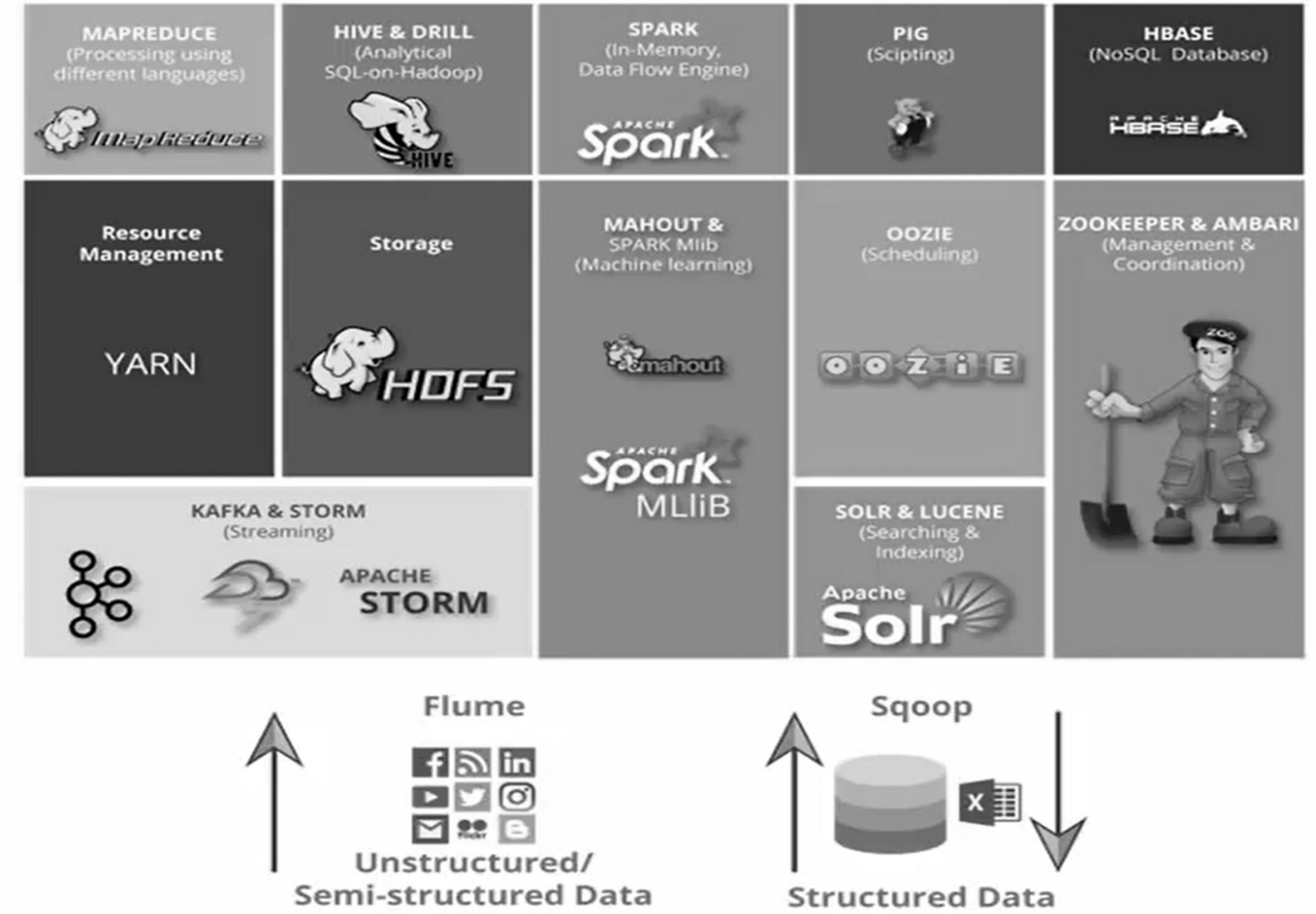
Big Data

- به طور معمول پهنای باند شبکه عاملی مهم است که باید در هنگام تشکیل هر شبکه در نظر گرفت. با این حال، چون اندازه گیری پهنای باند ممکن است دشوار باشد، در Hadoop یک شبکه به صورت یک درخت نمایش داده می شود و فاصله بین گره های این درخت (تعداد hopها) به عنوان یک عامل مهم در شکل گیری خوشه Hadoop محسوب می شود. در اینجا، فاصله بین دو گره برابر با مجموع فاصله آنها با نزدیکترین جد مشترک آنهاست.
- خوشه Hadoop شامل یک مرکز داده، rack و گرهی است که در واقع کار توسط آن اجرا می شود. در اینجا، مرکز داده شامل rackها و rack متشکل از گره ها هستند. پهنای باند شبکه در دسترس برای فرآیندها بسته به مکان فرآیندها متغیر است. به این معنا که پهنای باند در دسترس با دور شدن از عوامل زیر کمترمی شود:
  - ✓ فرآیندهای همان گره
  - ✓ گره های مختلف همان rack
  - ✓ گره های rack های مختلف یک مرکز داده
  - ✓ گره های مراکز داده های مختلف

Big Data

- بعضی ابزارهای استفاده شده در تجزیه و تحلیل Big Data




MAPREDUCE
(Processing using different languages)
MapReduce
HIVE & DRILL
(Analytical SQL-on-Hadoop)
HIVE
SPARK
(In-Memory, Data Flow Engine)
APACHE Spark
PIG
(Scipting)
HBASE
(NoSQL Database)
APACHE HBASE
Resource Management
YARN
Storage
HDFS
MAHOUT &
SPARK Mlib
(Machine learning)
mahout
APACHE Spark
MLliB
OOZIE
(Scheduling)
OOZIE
ZOOKEEPER & AMBARI
(Management & Coordination)
KAFKA & STORM
(Streaming)
APACHE STORM
SOLR & LUCENE
(Searching & Indexing)
Apache Solr
Flume
Unstructured/
Semi-structured Data
Sqoop
Structured Data

Big Data

❖ Hadoop Ecosystem

هادوپ یک راه حل مقیاس پذیر برای ذخیره و پردازش داده حجیم به صورت موازی و توزیع شده فراهم می کند.

Apache Hive یک ابزار انبار داده است که به ما جازه می دهد برای انجام تحلیل داده های بزرگ از زبان پرسشی **Hive** استفاده کنیم که بسیار شبیه **SQL** است. (**Facebook**)

Apache Pig یک **platform** است که برای تحلیل مجموعه داده های بزرگ استفاده می شود و آن ها را بصورت جریان داده نشان میدهد. (**Yahoo**)

Apache Spark یک موتور پردازش در حافظه است که به ما اجازه می دهد جریان، یادگیری ماشین یا بارکاری **SQL** را به طور کارآمد اجرا کنیم و نیاز به دسترسی مکرر به مجموعه داده ها دارد.

Apache HBase یک پایگاه داده **NoSQL** است که به ما اجازه می دهد تا داده بدون ساختار و نیمه ساخت یافته را به سادگی ذخیره کنیم و دسترسی نوشتن/ خواندن زمان واقعی را فراهم کنیم.

Big Data

❖ Hadoop Ecosystem

یک سیستم پیغام دهی است که مسئول ارسال پیغام از یک برنامه کاربردی به برنامه کاربردی دیگر است. برنامه روی داده تمرکز می کند و نیازی نیست نگران به اشتراک گذاشتن داده باشد.

یک **platform** یکپارچه نرم افزار **open source** است که اجازه میدهد داده را بدون زحمت اضافی تحلیل کنید و سپس داده را به مفاهیم تجاری تبدیل کنید واین امکان رابرای شرکت ها فراهم میکند که تصمیم گیری کنند.

splunk یک **platform** تحلیل کارنامه (**log**) است. **log** در دستگاههای محاسباتی و غیر محاسباتی تولید می شود و در دایرکتوری یا مکان مخصوص ذخیره می شوند. **log** اطلاعات مفصل در مورد تمام تراکنش را در بر دارد.

مغز **Hadoop ecosystem** است. تمام فعالیت های پردازشی را انجام می دهد و اختصاص منابع و زمانبندی **task** ها بر عهده این جزء است.

از زبانهای برنامه نویسی مثل **Java** استفاده می کند تا داده های حجیم را پردازش کند.

- HDFS چیست؟
- HDFS یک سیستم فایل توزیع شده برای ذخیره فایل های بسیار بزرگ داده است که در حال اجرا بر روی خوشه ای از سخت افزار رایج است. قابلیت تحمل خطا و مقیاس پذیری دارد و بسیار ساده گسترش می یابد. Hadoop باHDFS (سیستم فایل توزیع شده Hadoop) همراه است.
- هنگامی که داده فراتر از ظرفیت ذخیره سازی در یک دستگاه فیزیکی می شود، لازم است آن را بین تعدادی ماشین جداگانه تقسیم کنیم. یک سیستم فایل که عملیات مختص ذخیره سازی را در یک شبکه از ماشین ها مدیریت می کند یک سیستم فایل توزیع شده است. HDFS چنین نرم افزاری است.

Big Data

- خوشه بندی HDFS در اصل شامل یک NameNode است که ابرداده های سیستم فایل را مدیریت می کند و یک DataNode که اطلاعات واقعی را ذخیره می کند.

- ✓ NameNode:NameNode را می توان به عنوان سرویس دهنده سیستم در نظر گرفت. NameNode درخت سیستم فایل و ابرداده تمام فایل ها و دایرکتوری های موجود در سیستم را نگهداری می کند. دو فایل Namespace 'image' و 'edit log' برای ذخیره اطلاعات ابرداده ها استفاده می شوند. Namenode اطلاعات تمام datanode های حاوی بلوک های داده برای یک فایل را دارد، با این وجود، مکان های بلوک را بصورت طولانی مدت ذخیره نمی کند. هربار که سیستم شروع به کار می کند، این اطلاعات از datanodes بازسازی می شود.

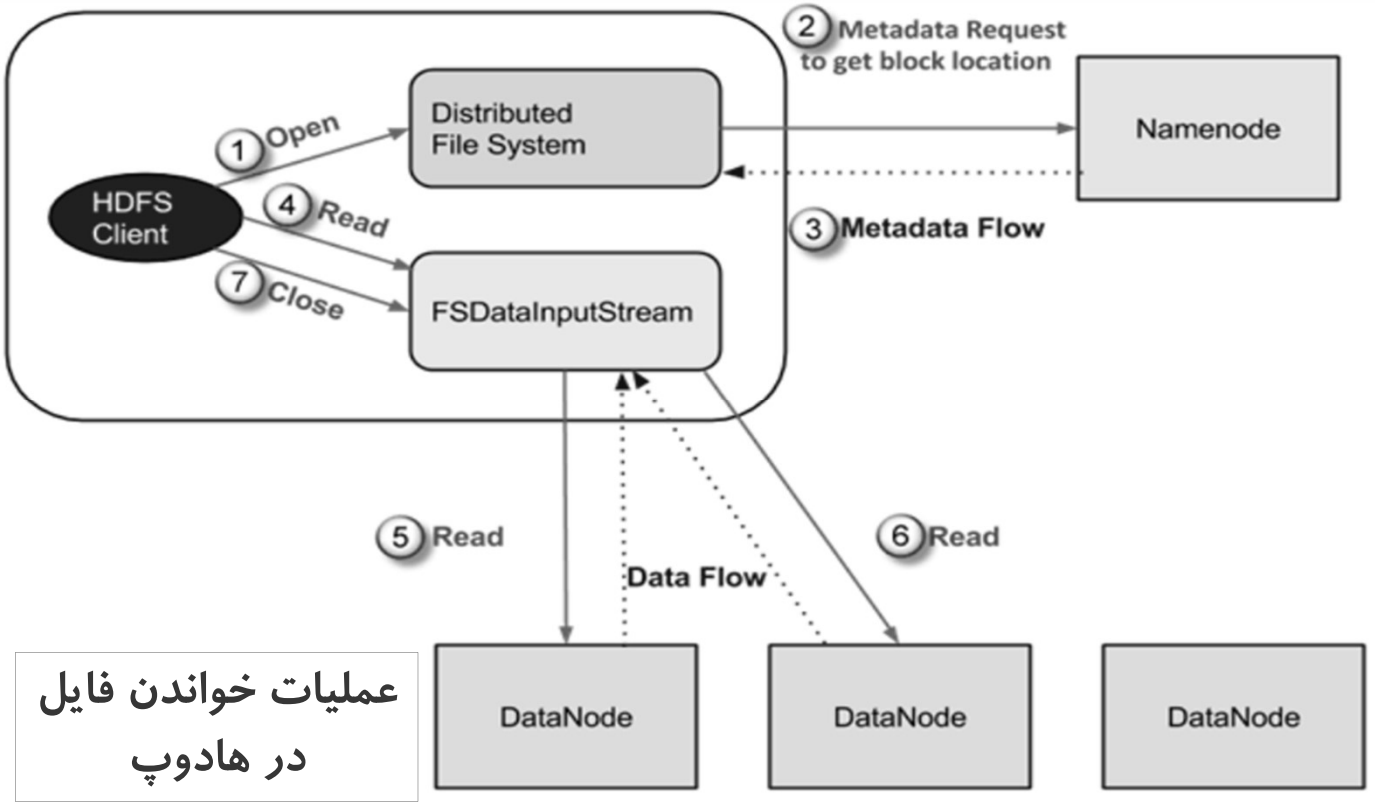
- ✓ DataNode:DataNodes ها سرویس گیرنده هایی هستند که بر هر دستگاه در یک خوشه قرار می گیرند و ذخیره واقعی را انجام می دهند و مسئول ارائه خدمات برای درخواست های خواندن و نوشتن مشتریان است.

Big Data

- عملیات خواندن / نوشتن در HDFS در سطح بلاک کار می کنند. فایل های داده در HDFS به تکه هایی با اندازه بلاک ها شکسته می شوند و به عنوان قطعات مستقل ذخیره می گردند. اندازه بلاک پیش فرض ۶۴ مگابایت است.
- HDFS بر پایه یک مفهوم از تکرار داده ها (replication) کار می کند به این شکل که چندین کپی از بلاک های داده ایجاد شده و در گره های سراسر یک خوشه توزیع شده اند تا در صورت خرابی گره امکان در دسترس بودن بالای داده فراهم باشد.
- نکته: یک فایل در HDFS، که کوچکتر از یک بلاک است، برای ذخیره سازی تمام فضای بلاک را اشغال نمی کند.

Big Data

- عملیات خواندن در HDFS
- درخواست خواندن داده توسط HDFS، NameNode و Datadata اجابت می شود. در صورتی که درخواست دهنده را مشتری (client) بنامیم، روند عملیات خواندن فایل در دیاگرام زیر مشاهده می گردد:



عملیات خواندن فایل در هادوپ

۱- یک مشتری با فراخوانی متد 'open ()' از شیء FileSystem (که از نوع DistributedFileSystem است) درخواست خواندن را آغاز می کند.
۲- این شیء با استفاده از RPC به namenode متصل می شود و اطلاعات ابرداده ای مانند مکان بلاک های فایل را دریافت می کند. توجه داشته باشید که این آدرس ها مربوط به چند بلاک اول یک فایل هستند.
۳- در پاسخ به این درخواست ابرداده، آدرسهای DataNodeهایی که یک کپی از آن بلاک را در اختیار دارند برگردانده می شود.
۴- هنگامی که آدرس DataNodeها دریافت می شود، یک شی از نوع FSDataInputStream به مشتری برگشت داده می شود که حاوی DFSInputStream است برای مراقب از تعامل بین DataNode و NameNode است. در مرحله ۴ از در نمودار، یک مشتری متد 'read()' را فراخوانی می کند که باعث میشود DFSInputStream با اولین DataNode در بردارنده اولین بلاک یک فایل ارتباط برقرار کند.

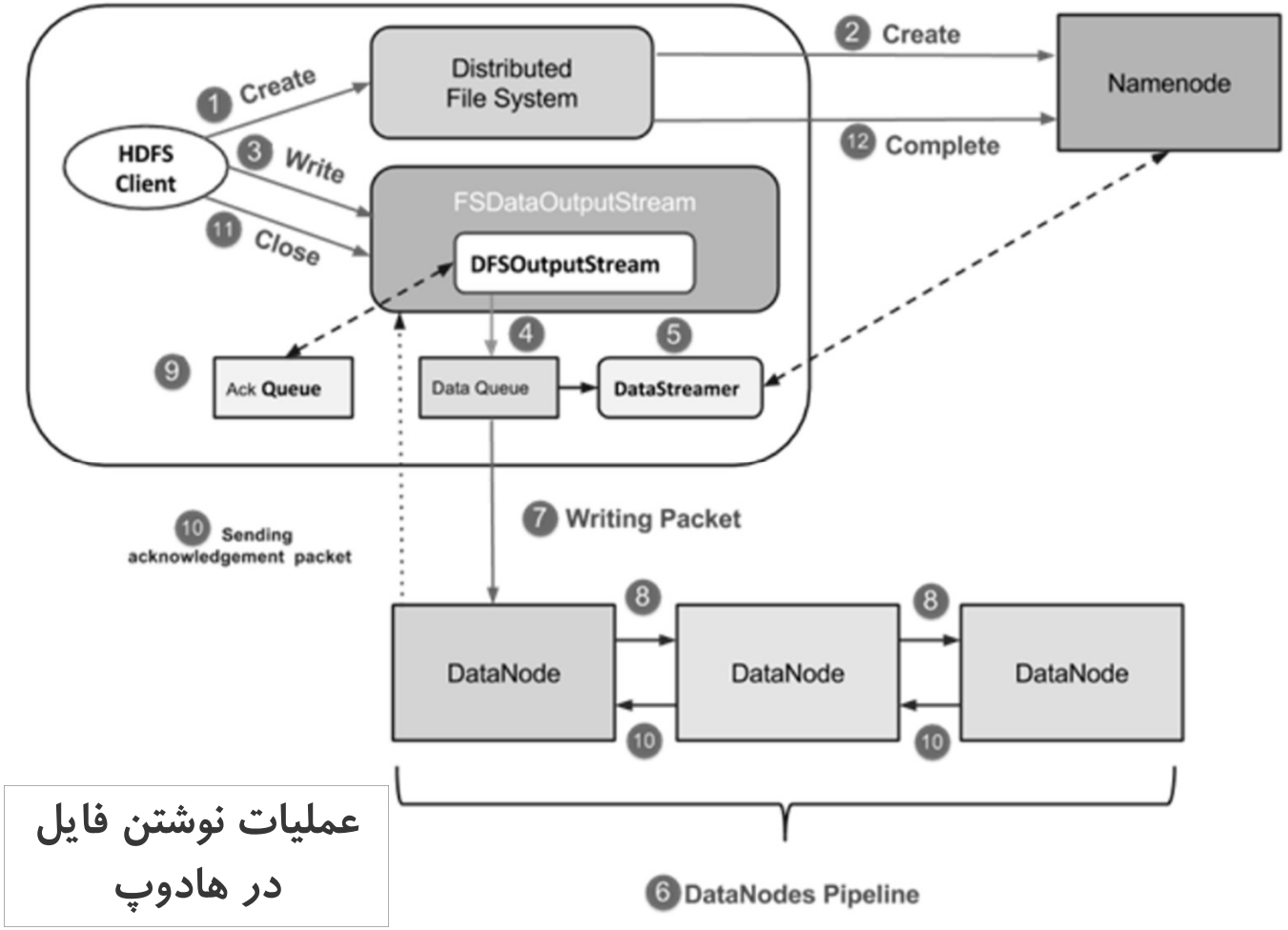
۵- داده ها در جایی که مشتری مند ' read () ' را به صورت مکرر فراخوانی می کند به صورت جریانهایی خوانده می شوند و این فرایند عملیات 'read()' تا رسیدن به انتهای بلاک ادامه می یابد.

۶- وقتی پایان بلاک فرا می رسد، DFSInputStream به اتصال خاتمه می دهد و اقدام به مکان یابی NextName بعدی برای بلوک بعدی می کند.

۷- هنگامی که عملیات خواندن یک مشتری خاتمه پیدا می کند، متد close () را فراخوانی می کند.

Big Data

- عملیات نوشتن در HDFS
- روند نوشته شدن داده در HDFS به واسطه فایل ها:



عملیات نوشتن فایل در هادوپ

۱- یک مشتری با فراخوانی متد 'create ()' از شی DistributedFileSystem عملیات نوشتن را آغاز می کند که یک فایل جدید ایجاد می کند - مرحله ۱ در دیاگرام قبل.

۲- شی DistributedFileSystem با فراخوانی RPC به NameNode متصل می شود و آغازگر ایجاد فایل جدید می شود. اگرچه، این فایل عملیات را ایجاد می کند ولی هیچ بلاکی را با فایل مرتبط نمی کند. این وظیفه NameNode است که تأیید کند که فایل (که در حال ایجاد است) فعلا وجود ندارد و مشتری مجوزهای صحیح برای ایجاد یک فایل جدید را دارد. اگر یک فایل در حال حاضر وجود داشته باشد یا مشتری مجوز کافی برای ایجاد یک فایل جدید نداشته باشد، IOException به مشتری ارسال می شود. در غیر این صورت، عملیات با موفقیت انجام می شود و یک رکورد جدید توسط NameNode برای فایل ایجاد می گردد.

۳- به محض ایجاد یک رکورد جدید در NameNode، یک شی از نوع FSDataOutputStream به مشتری برگردانده می شود و مشتری از آن برای نوشتن داده ها در HDFS استفاده می کند. متد نوشتن داده ها (مرحله ۳ در شکل) فراخوانی می شود.

۴- FSDataOutputStream شامل شیء DFSOutputStream است که مراقبت از ارتباط با DataNodes و NameNode را بر عهده دارد. در حالی که مشتری همچنان نوشتن داده ها را ادامه می دهد، DFSOutputStream ایجاد بسته ها با این داده ها را ادامه می دهد. این بسته ها در صفی که با عنوان DataQueue شناخته می شود، قرار می گیرند.

۵- جزء دیگری به نام DataStreamer وجود دارد که DataQueue را مصرف می کند. DataStreamer همچنین برای تخصیص بلاک های جدید NameNode را درخواست می کند و از این طریق DataNodeهای مناسب برای تکرار انتخاب می شوند.

۶- در این مرحله، فرایند تکرار با ایجاد یک pipeline با استفاده از DataNodeها آغاز می شود. در این مورد، سطح تکرار ۳ انتخاب شده است. به همین خاطر ۳ DataNode در pipeline وجود دارد.

۷- DataStreamer بسته ها را در اولین DataNode موجود در pipeline می ریزد.

۸- هر DataNode موجود در pipeline بسته دریافت شده را ذخیره کرده و به DataNode بعدی در pipeline ارسال می کند.

۹- صف دیگری با نام "Ack Queue" توسط DFSOutputStream نگهداری می شود تا بسته هایی که منتظر تایید از DataNodeها هستند را ذخیره کنند.

۱۰- به محض دریافت تاییدیه برای یک بسته موجود در صف از تمام DataNodeهای pipeline، آن بسته از Ack Queue حذف می شود. در صورت وقوع هر گونه خرابی DataNode، بسته های این صف آغاز دوباره عملیات مورد استفاده قرار می گیرند.

۱۱- پس از خاتمه کار مشتری نوشتن داده ها متد() close را فرا می خواند (مرحله ۹ در شکل) فراخوانی متد () close باعث می شود بسته های داده باقی مانده به pipeline سرازیر شده و سپس منتظر تایید باقی بماند.

۱۲- به محض دریافت تایید نهایی، با NameNode تماس گرفته می شود تا اعلام گردد عملیات نوشتن فایل تکمیل شده است.

Big Data

- MapReduce در هادوپ چیست؟
- MapReduce یک مدل برنامه نویسی است که برای پردازش داده های بزرگ مناسب می باشد. Hadoop قادر به اجرای برنامه های MapReduce نوشته شده در زبان های مختلف مانند Java، Ruby، Python و C ++ می باشد. برنامه های MapReduce به طور ذاتی موازی هستند، بنابراین برای تجزیه و تحلیل داده ها در مقیاس بزرگ با استفاده از چندین ماشین در خوشه بسیار کارآمد هستند.
- برنامه های MapReduce در دو مرحله کار می کنند:
  - ✓ مرحله Map
  - ✓ مرحله Reduce
- ورودی هر مرحله زوج های key-value هستند. علاوه بر این، هر برنامه نویس نیاز دارد دو تابع را مشخص کند: تابع map و تابع reduce.

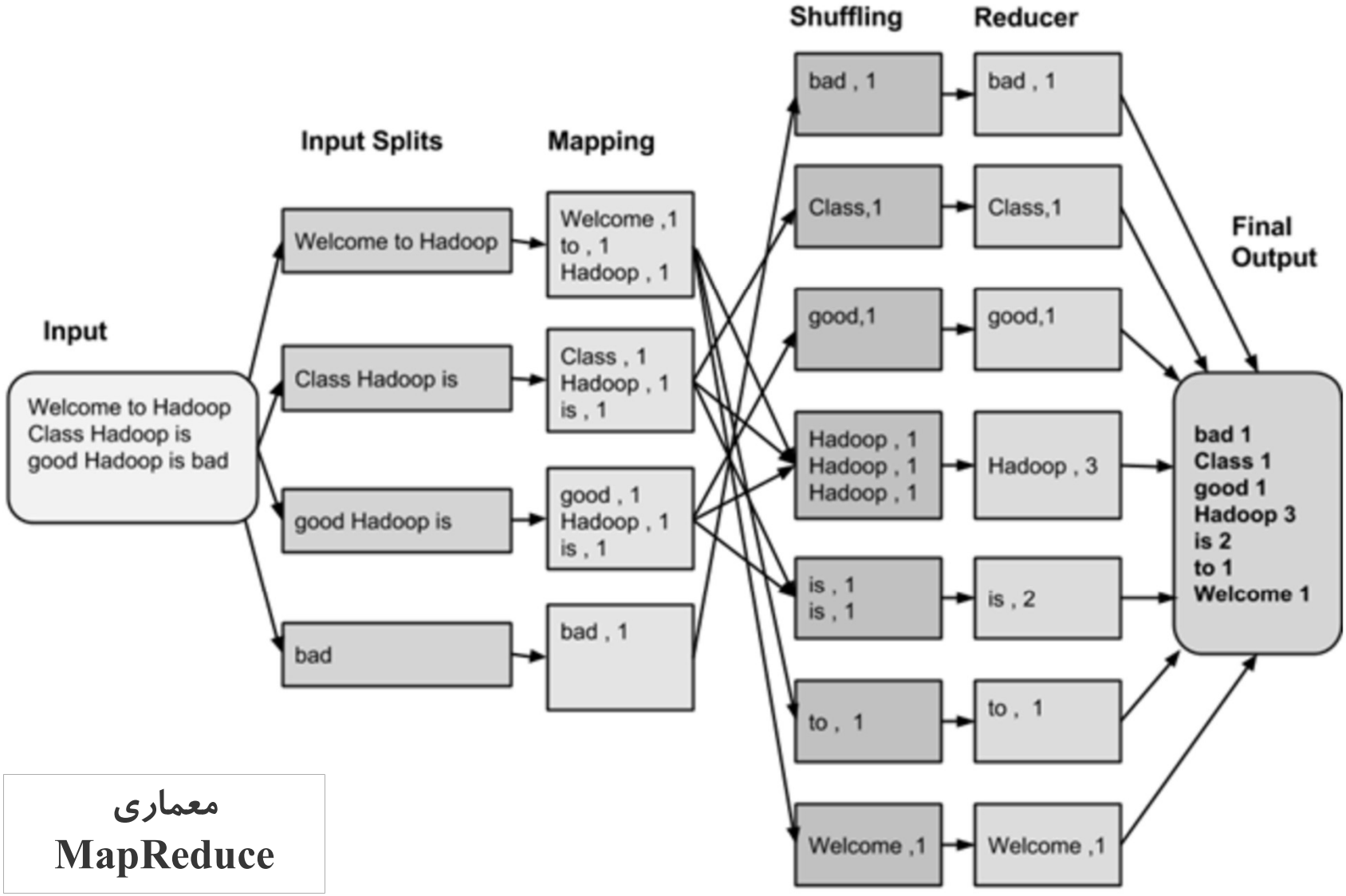
- MapReduce چگونه کار می کند؟ فرآیند کامل
- کل فرایند با اجرای چهار مرحله انجام می شود که به ترتیب splitting، mapping، shuffling و reducing می باشند.

- این مطلب با یک مثال توضیح داده می شود.
- داده های ورودی زیر را برای برنامه MapReduce خود در نظر بگیرید.

```
Welcome to Hadoop Class
Hadoop is good
Hadoop is bad
```

MapReduce

Big Data



معماری
MapReduce

❖ نتیجه نهایی کار MapReduce به صورت زیر است.

bad	1
Class	1
good	1
Hadoop	3
is	2
to	1
Welcome	1

❖ داده ها وارد مراحل زیر می شوند.

- input splits:

- ورودی به یک کار MapReduce به قطعاتی با اندازه ثابت تقسیم می شود که input splits نامیده می شود. input split، یک بخش از ورودی است که توسط یک map تک استفاده می شود.

- Mapping

- این اولین مرحله در اجرای برنامه map-reduce است. در این مرحله داده های هر بخش به یک تابع map منتقل می شود تا مقادیر خروجی تولید گردد. به عنوان مثال، یک کار در فاز map شمارش تعداد هر کلمه در input split است (جزئیات بیشتر در مورد input split در ادامه آمده است) که یک لیست را به شکل <word, frequency> آماده می کند.

- Shuffling:

- این مرحله خروجی فاز Mapping را استفاده می کند. وظیفه آن ترکیب رکوردهای مرتبط در خروجی فاز Mapping است. در مثال ذکر شده، کلمات یکسان با مقادیر تکرار مربوطه به هم می پیوندند.

- Reducing:

- در این مرحله، مقادیر خروجی از فاز Shuffling تجمیع می شوند. این فاز مقادیر حاصل از فاز Shuffling را ترکیب می کند و یک مقدار خروجی واحد تولید می کند. به طور خلاصه، این مرحله کل مجموعه داده را جمع بندی می کند.
- در مثال قبل، این فاز مقادیر فاز Shuffling را تجمیع می کند و تعداد کل هر کلمه را محاسبه می کند.

❖ جزئیات معماری MapReduce

- ✓ یک وظیفه map برای هر یک از بخش ها ایجاد می شود و سپس تابع map را برای هر رکورد آن بخش اجرا می کند.
- ✓ همیشه بهتر است چندین بخش داشته باشید، زیرا زمان لازم برای پردازش یک بخش نسبت به زمان انجام شده برای پردازش کل ورودی کوچک است. به دلیل این که بخش ها بصورت موازی پردازش می شوند، هنگامی که بخش ها کوچکتر می شوند، توازن بار (load balancing) بهتر انجام می گیرد.
- ✓ با این حال، اندازه بیش از حد کوچک بخش ها مطلوب نیست. هنگامی که بخش ها بیش از حد کوچک هستند، سربار حاصل از مدیریت بخش ها و ایجاد وظیفه map بر زمان کل اجرای کار غلبه خواهد کرد.
- ✓ برای اکثر کارها، بهتر است اندازه یک بخش برابر با اندازه یک بلوک HDFS (که به طور پیش فرض ۶۴ مگابایت است) باشد.

Big Data

❖ جزئیات معماری MapReduce

- ✓ خروجی حاصل از اجرای وظایف map در دیسک محلی بر روی گره مربوطه نوشته شود و نه روی HDFS.
- ✓ دلیل انتخاب دیسک محلی نسبت به HDFS، جلوگیری از تکرار در هنگام عملیات ذخیره سازی HDFS است.
- ✓ خروجی map یک خروجی میانی است که با وظایف reduce برای تولید خروجی نهایی پردازش می شود.
- ✓ پس از تکمیل کار، می توان خروجی map را دور ریخت. بنابراین، ذخیره آن در HDFS که با تکرار همراه است، کاری بیش از حد نیاز خواهد بود.
- ✓ در صورت خرابی گره، قبل از استفاده خروجی map توسط وظیفهreduce، Hadoop وظیفه map را مجددا در گره دیگری اجرا می کند و خروجی map را دوباره تولید می کند.

❖ جزئیات معماری MapReduce

- ✓ وظیفه reduce بر روی مفهوم محلی بودن داده کار نمی کند. خروجی هر وظیفه map به وظیفه reduce داده می شود. خروجی map به ماشینی که وظیفه reduce در آن در حال اجرا است منتقل می شود.
- ✓ در این ماشین، خروجی ادغام شده و سپس به تابع reduce تعریف شده توسط کاربر منتقل می گردد.
- ✓ بر خلاف خروجی map، خروجی reduce در HDFS ذخیره می شود (اولین تکرار در گره محلی ذخیره می شود و بقیه کپی ها در گره های خارج از رک ذخیره می گردند).

Big Data

❖ MapReduce چگونه کار می کند؟

✓ Hadoop کار را به وظایف تقسیم می کند. همانطور که مطرح شد دو نوع وظیفه وجود دارد:

✓ Map tasks (Splits & Mapping)

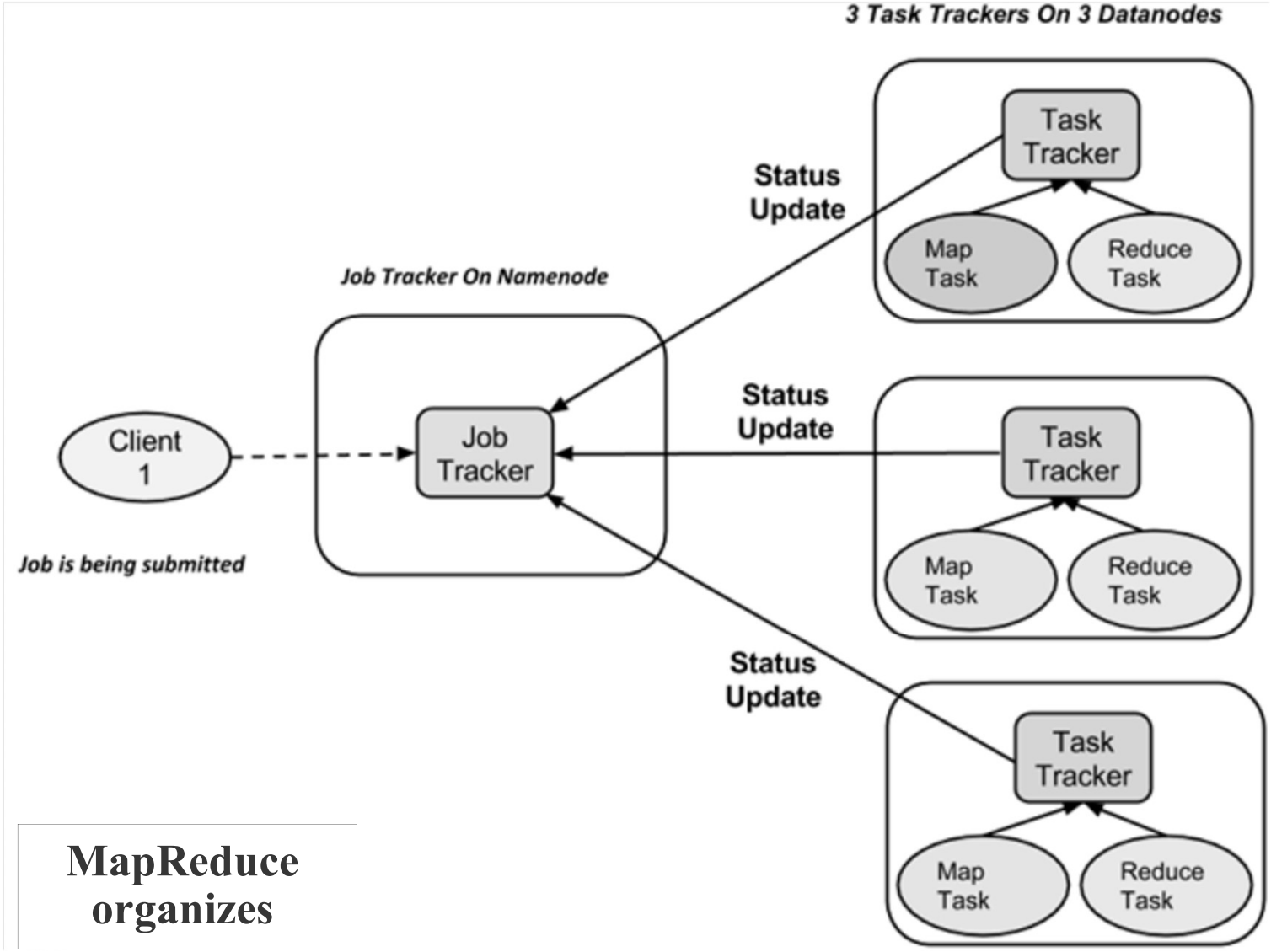
✓ Reduce tasks (Shuffling, Reducing)

✓ کل فرایند اجرا (اجرای map و reduce) توسط دو نوع نهاد کنترل می شود:

✓ Jobtracker : مانند یک سرور(master) عمل می کند(مسئول اجرای کامل کار ارسالی)

✓ چندین Task Tracker : همانند سرویس گیرنده (slave) عمل می کند و هر یک از آنها کار را انجام می دهند.

✓ به ازای هر کار ارسال شده برای اجرای در سیستم، یک Jobtracker وجود دارد که در Namenode است و چندین tasktracker که در Datanode است.



MapReduce organizes

Big Data

- MapReduce چگونه کار می کند؟
- یک کار به چندین وظیفه تقسیم می شود و سپس بر روی چندین گره داده در یک خوشه اجرا می شود.
- job tracker مسئول هماهنگی فعالیت با برنامه ریزی وظایف برای اجرا بر روی گره های داده های مختلف است.
- پس از آن، اجرای تک وظیفه، که در هر گره داده ای بخشی از کار را اجرا می کند قرار دارد، توسط task tracker دنبال می شود.
- مسئولیت task tracker ارسال گزارش پیشرفت به job tracker است.
- علاوه بر این، task tracker، به طور مداوم سیگنال "heartbeat" را به Jobtracker می فرستد تا او را از وضعیت فعلی سیستم مطلع کند.
- بنابراین job tracker پیشرفت کلی هر کار را پیگیری می کند. در صورت شکست وظیفه، job tracker می تواند آن را مجددا بر روی task tracker دیگری برنامه ریزی کند.

Big Data

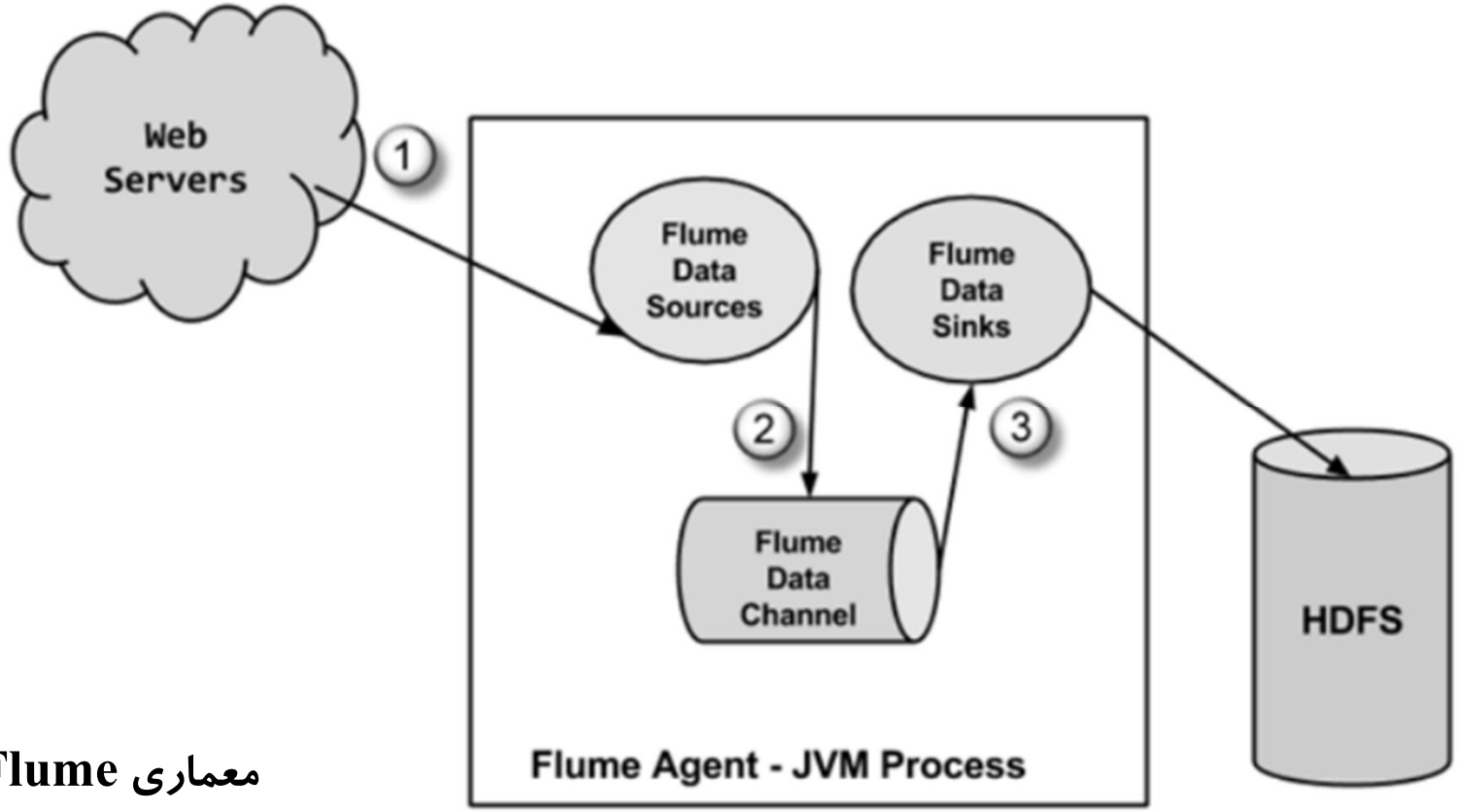
- صدها سرویس روی سرور های مختلف اجرا می شوند که تعداد زیادی log تولید می کنند و نیاز به تحلیل دارند.
- می توان از هادوپ برای پردازش این log ها استفاده کرد. نکته اینجاست که چگونه می توان تمام این log ها را برای محلی که هادوپ در حال اجراست ارسال کرد. این حقیقتی است که ما به یک راه قابل اطمینان، مقیاس پذیر، قابل مدیریت نیاز داریم تا این کار را انجام دهیم. برای حل این مساله Apache Flume معرفی شده است.
- اصلی ترین بخش داده کلان و حجیم را داده های بدون ساختار در بر دارد. برای ذخیره کردن این داده از Hadoop استفاده می شود. اولین گام در پردازش داده، بارگذاری و وارد کردن داده ها برای استفاده های بعدی می باشد.
- برای وارد کردن حجم زیادی از داده های بدون ساختار برای جریان داده های بلادرنگ، Big Data، ابزار Apache Flume معرفی شده است. Flume داده های بدون ساختار و نیم ساخت یافته را به HDFS ارسال می کند.

- Flume راه حلی قابل اعتماد و توزیع شده برای جمع‌آوری، انباشتن و انتقال دادن مجموعه داده‌های حجیم است.
- از این ابزار می‌توان در دریافت داده‌های آنلاینِ جاری از منابع مختلف، مانند ترافیک شبکه، رسانه‌های اجتماعی، پیام‌های ایمیل، فایل‌های log در سیستم و ... و ذخیره‌سازی آن روی HDFS، استفاده کرد.
- Flume منابع مختلفی را پشتیبانی می کند مانند:

  - tail (که داده را از یک فایل محلی کشیده و در HDFS از طریق Flume می نویسد. مشابه دستور tail مربوط به Unix)
  - log های سیستمی
  - Apache log4j (برنامه های جاوا را قادر می سازد تا از طریق Flume رویداد ها را در فایل های HDFS بنویسد)

Big Data

- Flume یک فرایند JVM است که سه جزء دارد:
- Flume Source، Flume Sink و Flume Channel



معماری Flume

- در شکل قبل، رویدادهای تولید شده از منبع خارجی(وب سرور) توسط Flume Data Source استفاده شده اند. منبع خارجی رویدادها را در قالبی که توسط منبع هدف شناسایی شده است برای Flume Source ارسال می کند.
- Flume Source یک رویداد را دریافت می کند و آن را در یک یا چند channel ذخیره می کند. این channel به عنوان انباری عمل می کند که رویداد را تا زمانی که توسط Flume Sink استفاده شود نگه می دارد. این channel ممکن است برای ذخیره این رویدادها از یک سیستم فایل محلی استفاده کند.
- Flume Sink رویدادها را از channel حذف کرده و آن را در مخزنی خارجی مانند HDFS ذخیره می کند. ممکن است چند عامل Flume وجود داشته باشند به این شکل که Flume Sink در روند جریان، رویداد را به سمت Flume Source عامل flume بعدی ارسال کند.

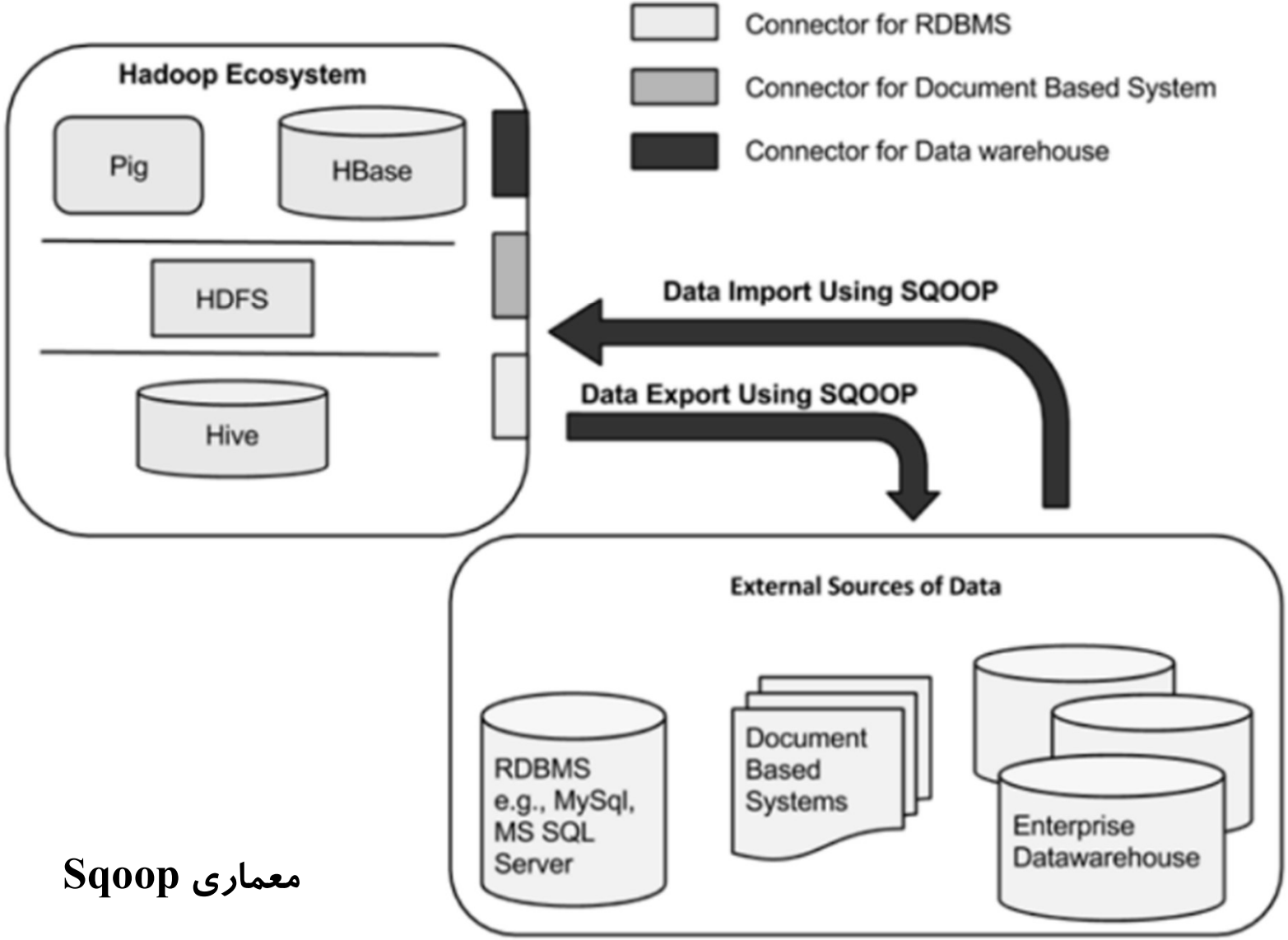
Big Data

- برخی از ویژگیهای مهم Flume
- Flume دارای طراحی انعطاف پذیر بر اساس جریان داده هاست. با چندین مکانیزم بازیابی و غلبه برخطا متحمل خطا و قدرتمند است. Flume سطوح متفاوتی از قابلیت اطمینان را پیشنهاد می دهدکه شامل best-effort delivery و end-to-end delivery است.best-effort delivery شکست هیچکدام از نودهای Flume را تحمل نمی کند درحالی که end-to-end delivery حتی در صورت بروز چندین نود شکست خورده تحویل را تضمین می کند.
- Flume اطلاعات را بین sourceها و sinkها منتقل می کند. این جمع آوری داده ها می تواند بصورت برنامه ریزی شده یا مبتنی بر رویداد باشد. Flume موتور پردازش پرس و جوی خود را دارد که تبدیل هر دسته جدید از داده ها قبل از انتقال به sink مدنظر را ساده می کند.
- Flume sink ها ممکن است شامل HDFS و Hbase باشند. Flume همچنین می تواند برای انتقال داده های رویدادها استفاده شودکه شامل داده ترافیک شبکه، داده تولید شده توسط وب سایتهای رسانه اجتماعی و پیامهای ایمیل می باشد اما محدود به این موارد نیست.

- Apache Sqoop (SQL-to-Hadoop) برای پشتیبانی از ورود حجیم داده به HDFS از داده های ساخت یافته مانند پایگاه داده های رابطه ای، انبارداده های سازمانی وسیستم های NoSQL طراحی شده است.
- Sqoop مبتنی بر معماری اتصال است که از pluginها برای اتصال به سیستم های خارجی جدید پشتیبانی می کند.
- یک مثال مورد استفاده از Sqoop شرکتی است که که یک Sqoop شبانه را اجرا می کند تا داده های روز از یک RDBMS مبادلاتی داده های تولید را به انبار داده Hive برای تحلیل های آتی بار می کند.

- تمام سیستم های مدیریت پایگاه داده موجود با توجه به استاندارد SQL طراحی شده اند. با این حال، هر DBMS تا حدودی با توجه به زبان متفاوت است. بنابراین، این تفاوت چالش هایی را در مورد انتقال داده ها در سراسر سیستم ها به وجود می آورد. اتصالات Sqoop اجزائی هستند که به غلبه بر این چالش ها کمک می کنند.
- انتقال داده بین Sqoop و سیستم ذخیره سازی خارجی با کمک اتصالات Sqoop امکان پذیر است.
- Sqoop دارای اتصالات برای کار با طیف وسیعی از پایگاه های داده های رابطه ای متداول، از جمله MySQL، PostgreSQL، اوراکل، SQL Server و DB2 است. هر یک از این اتصالات می داند چگونه با DBMS مربوط به آن ارتباط برقرار کند. همچنین یک اتصال JDBC عمومی برای اتصال به هر پایگاه داده ای که از پروتکل JDBC جاوا پشتیبانی می کند وجود دارد. علاوه بر این، Sqoop اتصالات MySQL و PostgreSQL بهینه سازی شده را فراهم می کند که از APIهای خاص پایگاه داده ای استفاده می کنند تا انتقالهای حجیم را به صورت کارا انجام دهد.

Big Data



معماری Sqoop

- Sqoop دارای اتصالات third party مختلف برای انبارهای داده است، از انبار داده های سازمانی (از جمله Netezza، Teradata، و اوراکل) گرفته تا انبارهای NoSQL مانند (Couchbase). با این حال، این اتصالات با بسته نرم افزاری Sqoop نمی آیند؛ بلکه باید جداگانه دانلود شوند و می توانند به راحتی به Sqoop نصب شده موجود اضافه شوند.
- چرا به Sqoop نیازمندیم؟
- پردازش تحلیلی با استفاده از Hadoop نیازمند بارگیری مقادیر زیادی داده از منابع متنوع به خوشه های Hadoop دارد. این فرآیند انتقال داده های حجیم به Hadoop، از منابع ناهمگن و سپس پردازش آن، با مجموعه ای از چالش های خاص همراه است. حفظ و حصول اطمینان از انطباق داده ها و اطمینان از بهره برداری موثر از منابع، برخی از عواملی هستند که باید قبل از انتخاب رویکرد صحیح بارگذاری داده مورد توجه قرار گیرند.

❖ موارد اصلی:

۱- بارگیری داده ها با استفاده از اسکریپت ها

رویکرد سنتی استفاده از اسکریپت برای بارگیری داده ها، برای بارگیری داده های حجیم به Hadoop مناسب نیست این روش ناکارآمد و بسیار وقت گیر است.

۲- دسترسی مستقیم به داده های خارجی از طریق برنامه Map-Reduce

فراهم نمودن دسترسی مستقیم به داده های موجود در سیستم های خارجی (بدون بارگذاری در Hadoop) برای برنامه های کاربردی Map-Reduce، این برنامه ها را پیچیده می کند. بنابراین این رویکرد امکان پذیر نیست.

۳- Hadoop علاوه بر توانایی کار با داده های بیشمار، می تواند با داده هایی در چندین فرم مختلف کار کند. بنابراین، برای بارگذاری چنین داده های ناهمگنی در Hadoop، ابزارهای مختلفی توسعه یافته اند. Sqoop و Flume دو نمونه از این ابزارهای بارگیری اطلاعات هستند.

Sqoop

Sqoop vs Flume vs HDFS in Hadoop

Big Data

HDFS	Flume	Sqoop
HDFS یک سیستم فایل توزیع شده است که توسط اکوسیستم Hadoop برای ذخیره داده ها استفاده می شود.	Flume برای انتقال جریان داده حجیم به HDFS استفاده می شود.	Sqoop برای وارد کردن داده ها از منابع داده ساختیافته مانند RDBMS استفاده می شود.
HDFS دارای یک معماری توزیع شده است که داده ها در بین گره های مختلف داده توزیع می شوند.	Flume دارای معماری مبتنی بر عامل است. کدی نوشته شده (که به عنوان «عامل» نامیده می شود) و به مراقبت از واکشی داده ها می پردازد.	Sqoop معماری مبتنی بر اتصال دارد. اتصالات می دانند که چگونه به منبع داده مربوطه متصل شوند و داده ها را واکشی کنند.
HDFS یک مقصد نهایی برای ذخیره سازی داده است.	داده ها از طریق هیچ یا چند کانال به HDFS جریان دارند.	HDFS یک مقصد برای وارد کردن داده ها با استفاده از Sqoop است.
HDFS فقط داده ها را که به وسیله ای برایش فراهم شده اند، ذخیره می کند.	بارگذاری داده در Flume می تواند توسط یک رویداد انجام شود.	بارگذاری داده در Sqoop رویدادگرا نیست.
HDFS دارای دستورات داخلی shell برای ذخیره داده ها در آن است. HDFS نمی تواند داده های جریانی را وارد کند.	Flume باید به منظور بارگیری داده های جریان مانند توییت های ایجاد شده در توییتر یا فایل های log یک وب سرور استفاده شود. عوامل Flume برای تهیه داده های جریان داده ساخته شده اند.	برای وارد کردن داده ها از منابع داده های ساخت یافته، تنها باید از Sqoop استفاده شود، زیرا اتصال دهنده ها می دانند که چگونه با منابع داده ای ساخت یافته ارتباط برقرار کرده و داده ها را از آنها استخراج کنند.

Big Data

❖ Pig چیست؟

✓ Pig یک زبان برنامه نویسی سطح بالا است که برای تجزیه و تحلیل مجموعه داده های بزرگ مفید است. Pig نتیجه تلاش توسعه در Yahoo بود.

✓ در یک چارچوب MapReduce، برنامه ها باید به یک سری مراحل map و reduce ترجمه شوند. با این حال، این یک مدل برنامه نویسی نیست که تحلیلگران داده با آن آشنا هستند. بنابراین، برای پر کردن این شکاف، یک انتزاع به نام Pig بر بالای Hadoop ساخته شد.

✓ Apache Pig باعث می شود که بیشتر تمرکز روی تجزیه و تحلیل مجموعه داده های انبوه باشد و زمان کمتری برای نوشتن برنامه های Map-Reduction صرف شود. زبان برنامه نویسی Pig به مانند خوک ها، که هر چیزی را می خورند، طراحی شده است تا بر روی هر نوع داده ای کار کند. دلیل نام Pig هم همین است.

Big Data

❖ معماری Pig

✓ Pig شامل دو جزء است:

- Pig Latin، که یک زبان است
- یک محیط زمان اجرا، برای اجرای برنامه های PigLatin

✓ یک برنامه Pig Latin شامل یک سری عملیات یا تبدیلات است که روی داده های ورودی برای تولید خروجی اعمال می شود. این عملیات یک جریان داده را توصیف می کند که به وسیله محیط اجرایی Pig به یک نمونه قابل اجرا ترجمه شده است.

✓ در سطح زیرین، نتایج این تبدیلات مجموعه ای از کارهای MapReduce است که برنامه نویس از آن آگاه نیست. به همین ترتیب، Pig اجازه می دهد تا برنامه نویس بر روی داده ها و نه ماهیت اجرا متمرکز شود.

✓ PigLatinیک زبان نسبتا سفت و سخت است که از کلمات کلیدی آشنا از پردازش داده ها استفاده می کند، به عنوان مثال، Join، Group و Filter.

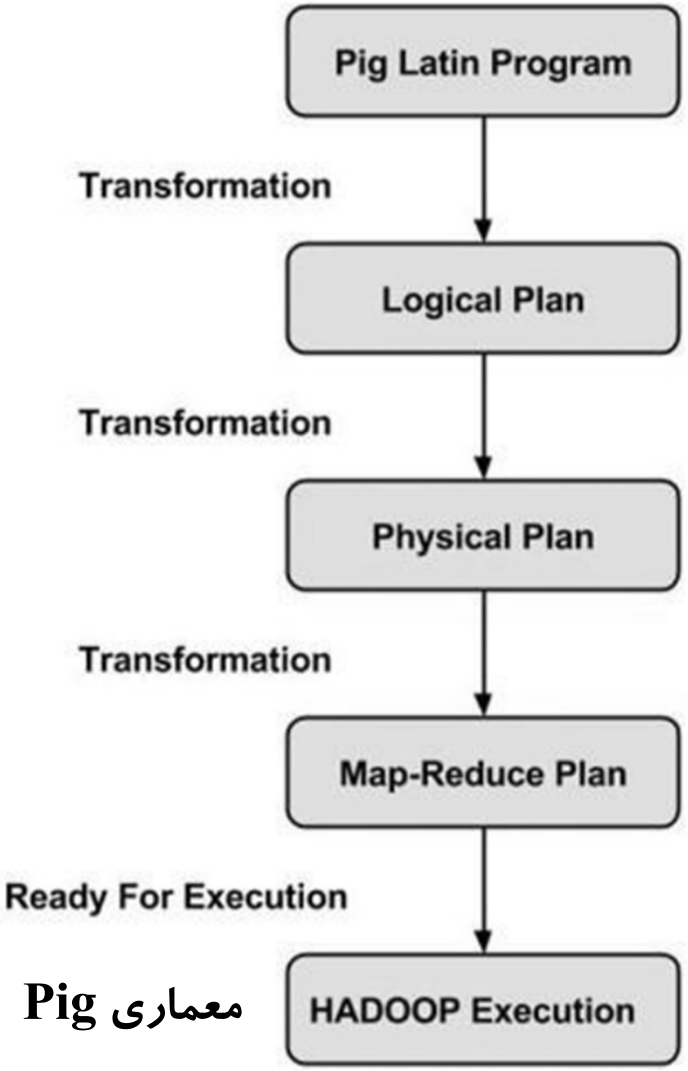
Big Data

❖ معماری Pig

✓ Pig دارای دو حالت اجرا است:

✓ حالت محلی: Pig در یک JVM تک اجرا می‌شود و از سیستم فایل محلی استفاده می کند. این حالت فقط برای تجزیه و تحلیل داده‌های کوچک با استفاده از Pig مناسب است.

✓ حالت Map Reduce: پرس و جوهای نوشته شده در Pig Latin به کارهای MapReduce ترجمه می شوند و بر روی یک خوشه Hadoop اجرا می شوند (خوشه ممکن است بصورت شبه یا کامل توزیع شود). حالت MapReduce با خوشه کاملا توزیع شده برای اجرای Pig روی مجموعه داده های بزرگ مفید است.



معماری Pig

- Pig Latin برنامه نویس را از کدنویسی Map Reduce براساس جاوا بی نیاز کرده و این قابلیتها را به صورت سیستم نگارشی سطح بالایی عرضه می کند که بسیار شبیه SQL در سیستم های RDBMS است.
- Pig Latin را می توان به کمک UDF(یا توابع تعریف شده توسط کاربر User Defined Functions) گسترش داد که کاربر می تواند این توابع را در جاوا پیاده سازی کرده و بعدتر مستقیماً از طریق Pig Latin فراخوانی کند.
- Pig در اصل در سال ۲۰۰۶ در مراکز تحقیقاتی yahoo برای محققان توسعه داده شد تا روشی ساده برای ایجاد و اجرای عملیات Map Reduce روی مجموعه های بسیار عظیم داده در اختیار داشته باشند. در سال ۲۰۰۷ این پروژه به بنیاد نرم افزاری آپاچی منتقل شد.

Big Data

❖ Oozie چیست؟

✓ Apache Oozie یک زمانبند گردش کاری برای Hadoop است. سیستمی است که گردش کار کارهای وابسته را اجرا می کند. در اینجا، کاربران مجازند از گردش کارها Directed Acyclic Graphs را ایجاد کنند که می تواند به طور موازی و به ترتیب در Hadoop اجرا شود.

❖ Apache Oozie از دو بخش تشکیل شده است:

✓ موتور گردش کار: مسئولیت یک موتور گردش کار، ذخیره و اجرای گردش کارهایی متشکل از کارهای Hadoop مانند MapReduce، Pig و Hive است.

✓ موتور هماهنگ کننده: گردش کارها را براساس زمانبندی های از پیش تعریف شده و در دسترس بودن داده ها اجرا می کند.

❖ Oozie مقیاس پذیر است و می تواند اجرای به موقع هزاران گردش کاری (که هر کدام شامل چندین کار است) را در یک خوشه Hadoop مدیریت کند.

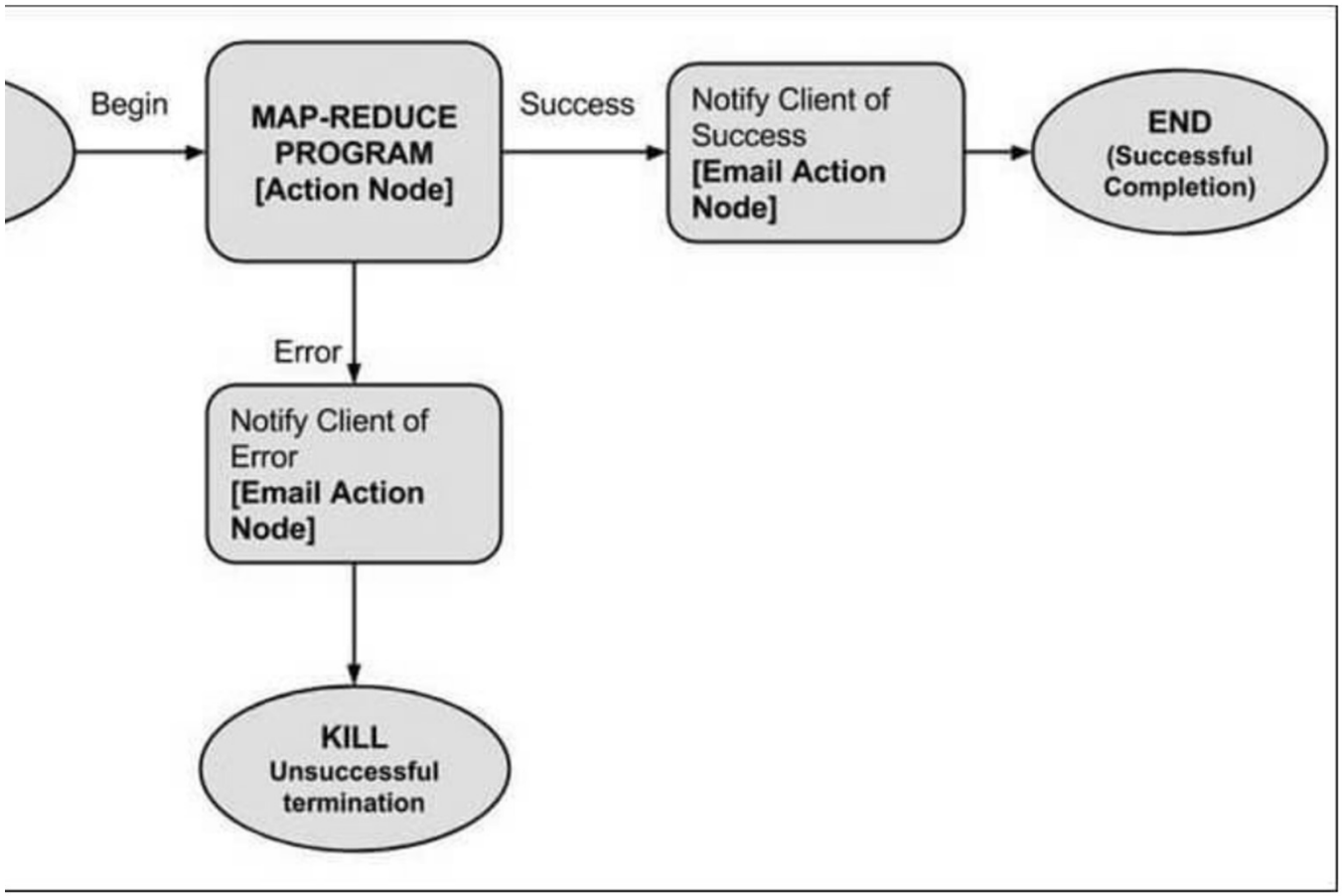
Oozie

Tools Help

Apache Oozie Tutorial: What is | Google Translate

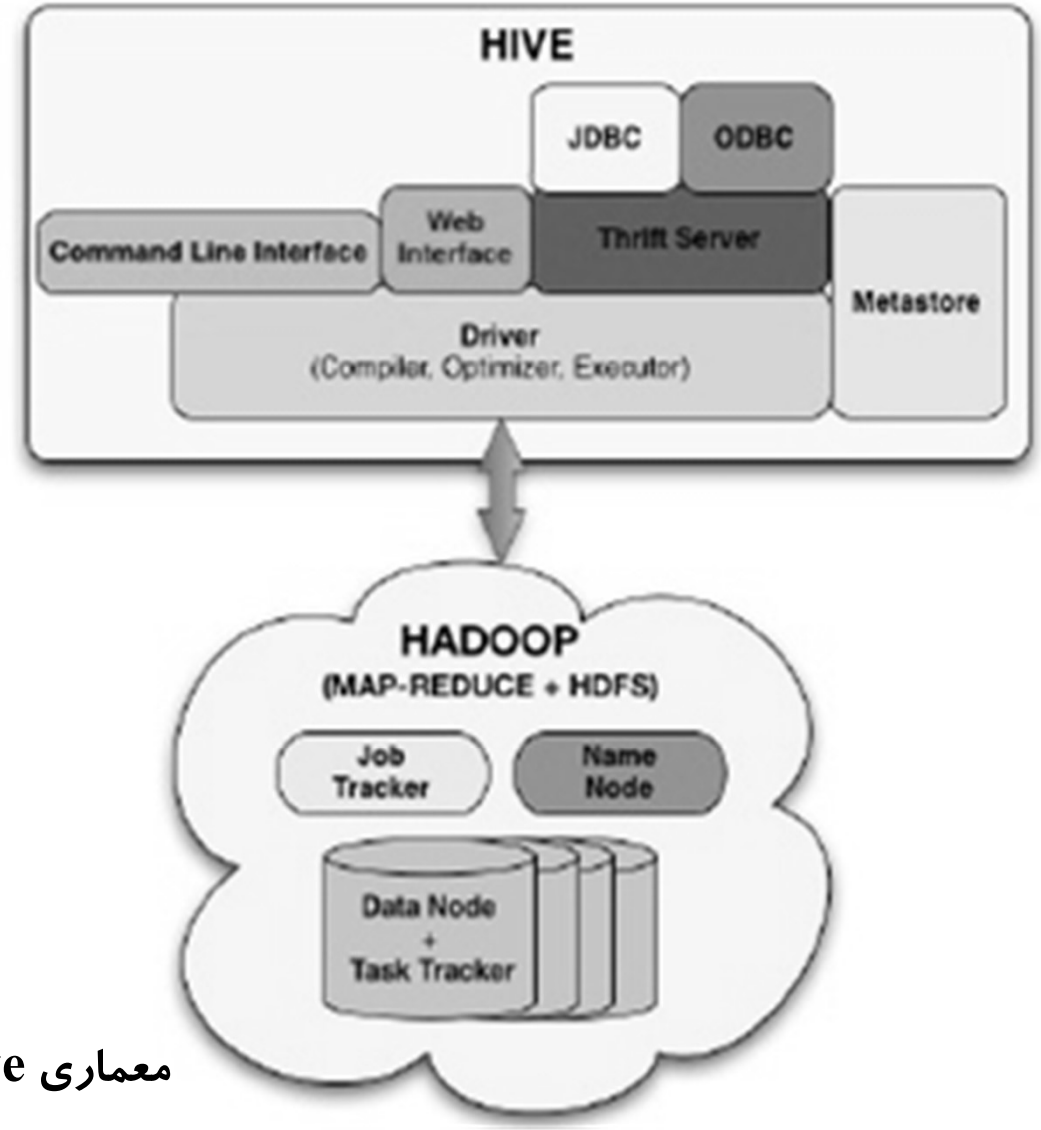
https://www.guru99.com/learn-oozie-in-5-minutes.html 110% Search

ozens of jobs) in a Hadoop cluster.



uch flexible, as well. One can easily start, stop, suspend and rerun jobs. Oozie

- hive یک روش در زبان‌های سطح بالاست که در Facebook توسعه داده شده است. کاری که Hive انجام می‌دهد ارائه‌ی نوعی پوسته (Shell) نزدیک به SQL است. بنابراین دستورالعمل‌هایی می‌نویسید که شباهت زیادی به SQL دارند و Hive نگاشت میان Schema و فایل‌ها را نگهداری می‌کند.
- اطلاعات فایل‌های درون فایل سیستم و اطلاعاتی در مورد محتوای آن‌ها را به هایو داده و هایو آن‌ها را در ستون‌هایی مرتب می‌کند. سپس پرس‌وجوها را می‌نویسید که به عنوان کارهای MapReduce اجرا می‌شوند.
- هر دوی Pig و Hive از بسیاری جنبه‌ها کارهای یکسانی را انجام می‌دهند. تلاش‌هایی برای پشتیبانی Pig از پرس و جو های شبیه به SQL هم وجود دارد. هر دو زبان دارای بهینه‌ساز (Optimizer) هستند. هر دو قادر به اجرای کارهایی هستند که ممکن است شامل چند کار MapReduce باشد.


HIVE
JDBC
ODBC
Command Line Interface
Web Interface
Thrift Server
Metastore
Driver
(Compiler, Optimizer, Executor)
HADOOP
(MAP-REDUCE + HDFS)
Job Tracker
Name Node
Data Node
+
Task Tracker

معماری Hive

منابع

References:

[1] S. G. Akl, *The Design and Analysis of Parallel Algorithms*. USA: Prentice-Hall, 1989.

[2] Guru99

[3] www.edureka.co EDUREKA HADOOP CERTIFICATION TRAINING

[4] myhadoop.ir

[5] داده پردازی سیمیگران، داده های عظیم