جزوه درس:

تحلیل ها وسیستم هاي داده هاي حجیم

استاد مربوطه: آقای دکتر مهدی اسماعیلی

منبع تدریس:

Big Data Fundamentals

Concepts, Drivers & Techniques

Thomas Erl,
Wajid Khattak,
and
Paul Buhler

2016

#جلسه اول، پنجشنبه ۱۳۹۵/۷/۸

مقدمه

در دنیای امروز، در کسری از ثانیه تعداد نامحدودی data تولید می‌شود. در تصویر زیر برای تعدادی از شرکتهای معروف نشان می‌دهد که در هر ۶۰ثانیه، چه مقدار داده تولید می‌شود. البته این میزان داده همچنان در حال افزایش است.

برای مثال:

- هر ۶۰ ثانیه در Google نزدیک به ۷۰۰هزار جستجو انجام می‌شود.(حدودا ثانیه‌ای ۱۰هزارتا)
- در ۶۰ ثانیه در Skype، ۳۷۰هزار دقیقه صدا ثبت می‌شود.

این داده‌ها جزو ارزشمندترین داده‌ها میباشند، چرا که با استفاده از داده‌کاوی می‌توان نتایج ارزشمندی از آنها استخراج کرد.

در جایی عنوان شده که گوگل به دلیل تحلیل تعداد جستجوهایی که انجام می‌شود، سریع‌تر از پزشکان یک منطقه متوجه می‌شود که در آن منطقه چه بیماری‌ای رایج شده است. مثلا از منطقه‌ی X مدام جستجو می‌شود: "داروی سرماخوردگی؟" پس گوگل متوجه می‌شود در آنجا سرماخوردگی رایج شده است.

در نتیجه شرکتهای تجاری بزرگ دنیا می‌توانند با خریداری این داده‌ها از امثال گوگل به سودهای خوبی برسند. در مثال بالا می‌توان نتیجه گرفت که درحال حاضر در منطقه‌ی X داروی سرماخوردگی مشتری زیادی دارد.

- در ۶۰ ثانیه ۱۶۸میلیون ایمیل ارسال می‌شود.(حدودا ثانیه‌ای ۳میلیون)

تااینجا با big data تا حدودی آشنا شدید. حداقل متوجه شدید که چه حجم وحشتناکی از داده در هر ثانیه درحال تولید است. در بعضی منابع این حجم را به صورت ملموس‌تر با مقایسه‌های جالب بیان میکنند. مثلا اطلاعاتی که در یک ساعت توسط کارمندان یاهو تولید می‌شود(log fileها) مساوی است با ۵برابر تعداد کلماتی که انسانها از زمان تولد حضرت آدم تاکنون به زبان آورده‌اند.

- در لندن حدود ۶میلیون دوربین نظارتی وجود دارد، داده‌هایی که توسط این تعداد دوربین ضبط می‌شود بسیار زیاد هستند. یعنی حدود ساعتی ۶۰۰میلیون مگابایت.

تا اینجا با چالش ذخیره‌سازی داده‌ها آشنا شدیم؛ اما بعد از آن نحوه پردازش و استفاده ازاین حجم عظیمی از اطلاعات باارزش است.

- در ۶۰ ثانیه در You Tube، ۶۰۰ ویدئوی جدید آپلود می‌شود.
- در ۶۰ ثانیه در Tweets، ۱۰هزار پیام ارسال می‌شود. (حدودا ثانیه‌ای ۸۰۰تا)

نکته جالب اینجاست که اینها ترکیبی هستند. یعنی یک نفر میتواند از چند شبکه با هم حتی بصورت همزمان استفاده کند. به همین ترتیب مکانهایی که پهنای باند بهتری دارند این استفاده بیشتر است. همچنان که پیش می‌رویم این تولید عظیم داده‌ها تشدید می‌شود.

کاربردهای اصلی big data

حال به این نکته می‌رسیم که کاربردهای اصلی big data کجاست؟

کاربردهای اصلی big data که در تصویر زیر ذکر شده است شامل موارد زیر است:

- Smarter Healthcare ؛ لباسهای هوشمند
- Multi-channel sales ؛ فروشگاههای زنجیره‌ای
- Finance ؛ امور مالی
- Log Analysis ؛ تجزیه و تحلیل log fileها
- Homeland Security ؛ امنیت داخلی
- Traffic Control ؛ کنترل ترافیک
- Telecom ؛ مخابرات(ارتباطات)
- Search Quality ؛ کیفیت جستجو
- Manufacturing ؛ تولید

- Trading Analytics ؛ تجزیه و تحلیل تجارت
- Fraud and Risk ؛ شناسایی تقلب و ریسک
- Retail: Churn, NBO ؛ خرده فروشی(کسب و کار تازه)

یکی از مهم‌ترین اینها Fraud and Risk است. از ترکیب این مورد و big data پروژه‌های خوبی حاصل می‌شود. وقتی داده‌ها زیاد می‌شوند بدون تردید تقلب در آن یک بحث بحرانی می‌شود. همچنین با افزایش حجم داده‌ها و بالا رفتن احتمال تقلب در آن ریسک استفاده از آن نیز زیاد می‌شود.

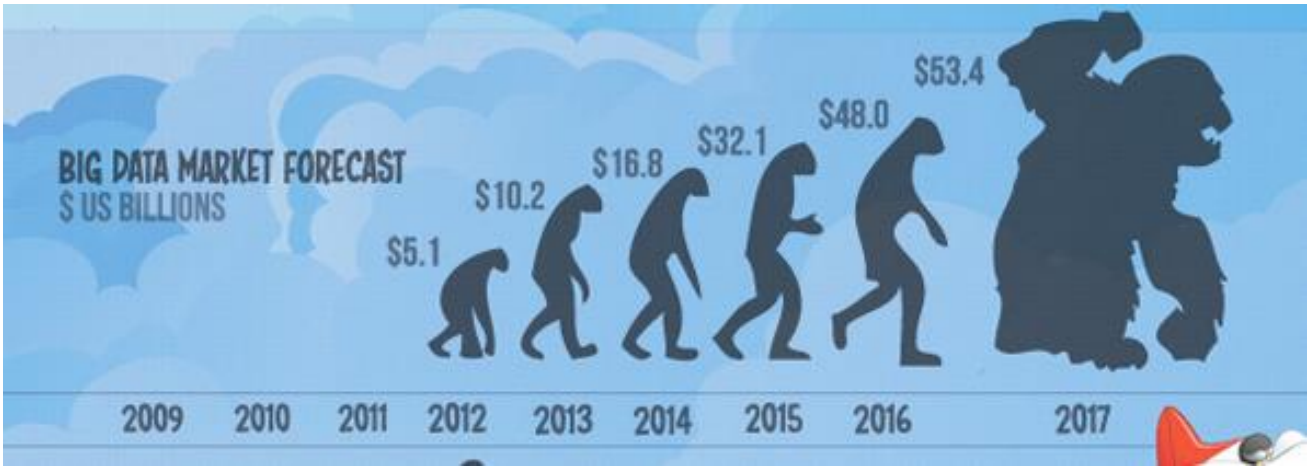
امروزه افراد مسنی که در خانه هستند لباسهای خیلی سبکی به آنها وصل می‌شود تا بطور خودکار علائم حیاتی آنها را چک کرده و در حد رساندن یک درمان سرپایی برایشان عمل می‌کنند تا اورژانس برسد. مثلا اگر سنسورهای لباس قندخون بالا را تشخیص دهند، سریعا با تزریق انسولین آن را به حالت نرمال می‌رساند. این لباسها با این افزایش حجم استفاده همچنان در حال تولید دیتا هستند.

در صنعت شهرهای هوشمند کنترل ترافیک از مسائل اصلی آن شهر می‌باشد. هر لحظه با داده‌هایی که از سنسورهای اتومبیلها جمع می‌شوند وضعیت ترافیکی یک مکان خاص بررسی می‌شود؛ حتی اگر حرکت خودرویی غیرعادی باشد مثلا با سرعت غیرعادی حرکت کند به خودروهای اطراف اطلاع داده می‌شود تا حواسشان به این خودرو باشد و از آن فاصله بگیرند تا از بروز تصادف جلوگیری شود. در امثال این کاربردها سرعت پردازش بسیار مهم است. سیستم باید در آن واحد غیرعادی بودن را تشخیص داده و به خودروهای اطراف اطلاع رسانی کند.

Logها همان ثبت و ضبط اطلاعات کاربران می‌باشد. مثلا اگر شما وارد سایت amazon شدید هرگونه جستجو و حتی کلیک روی هر web page ثبت می‌شود تا در موارد بعدی ورود شما به این سایت بتواند علاقمندی‌های شما را شناسایی کرده و به انجام امور شما سرعت بخشد.

مثلا یک فروشگاه زنجیره‌ای مانند تسکو در اروپا با فعالیت بسیار گسترده و تعداد مشتریان بالا داده‌های فراوانی تولید می‌کنند. این داده‌ها باید تحلیل شوند تا فعالیت فروشگاه بتواند هدفمند باشد. مثلا غرفه‌ها را براساس دم دست بودن اجناس پرفروش جابجا می‌کنند.

بازار big data



تصویر بالا بازار big data را در سالهای متمادی در آمریکا نشان می‌دهد. مثلا در سال ۲۰۱۶، ۴۸بیلیون دلار صرف این بازار و پروژه‌های آن می‌شود. برای مثال آمریکا گندم بیش از ۵۰درصد از کشورهای جهان را تامین می‌کند؛ که این خود یعنی یک عالمه دیتا. اگر دولت آمریکا نتواند این داده‌ها را تحلیل کند شاید نتواند برای سال آینده تولید و صادرات گندمش را مدیریت کند.

با دقت در تصویر بالا متوجه می‌شوید که رشد این بازار نه تنها در طول سالها یه یک نسبت ثابت نبوده بلکه هرسال این رشد بیشتر و بیشتر شده است.

تعریف big data

در big data روی این بحث میشود که چگونه حجم بسیار بزرگی از داده‌ها را که اغلب از منابع مختلفی سرچشمه میگیرند تحلیل، پردازش و ذخیره کنیم.(پس چالشهای big data سه مورد است: تجزیه و تحلیل، پردازش و ذخیره)

راهکارهای big data و روشهای مربوط به آن موقعی لازمند که روشهای سنتی کافی نباشند یا مشکل داشته باشند. در غیر اینصورت استفاده از big data اشتباه محض است.

مفاهیم و اصطلاحات فنی

data و dataset؛ دراینجا منظور ما از این اصطلاحات صرفا جدول نیست. بلکه داده‌های ما می‌توانند جدول، تصویر، صوت و... باشند؛

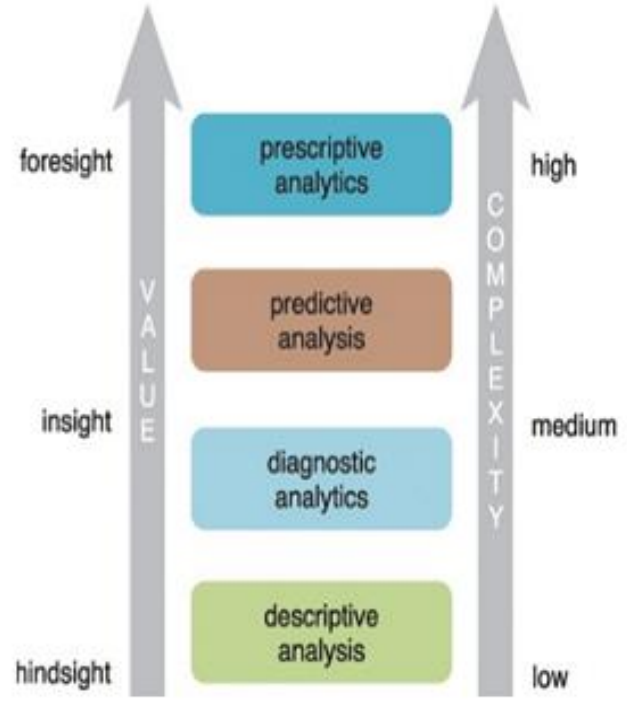
Data analysis؛ به معنی تحلیل داده‌هاست. تحلیل به پیدا کردن رابطه‌ی بین داده‌ها، یا بدست آوردن یک بینش، الگو یا حقیقتی از داده‌ها گویند. مثلا سن به تحصیلات رابطه دارد. تحلیل کردن با این منظور صورت می‌گیرد که بعدا تصمیم‌گیری درست انجام دهیم.

Data analytics؛ علم تجزیه و تحلیل. کل فرآیند جمع‌آوری، پالایش، ذخیره‌سازی، مدیریت، تحلیل، پردازش و نتیجه‌ی حاصل از تحلیل داده‌ها را گویند. به عبارتی به معنی چرخه‌ی حیات داده‌هاست. که Data analysis بخشی از آن است.(Data analysis زیرمجموعه‌ی Data analytics است)

انواع تحلیل

تحلیل ۴نوع است. انواع تحلیل به صورت زیر می‌باشد:

۱) Descriptive analytics ؛ تحلیل توصیفی
۲) Diagnostic analytics ؛ تحلیل تشخیصی
۳) Predictive analytics ؛ تحلیل پیشگویانه
۴) Prescriptive analytics ؛ تحلیل تجویزی



Value and complexity increase from descriptive to prescriptive analytics

از پایین به بالا پیچیدگی بیشتر می‌شود.

از پایین به بالا *ارزش* بیشتر می‌شود. (بینش بهتری به ما می‌دهند.)

برای مثال داریم:

داده ← اطلاعات ← دانش ← خرد

باافزایش ارزش در جهت فلش پیش می‌رویم. درواقع هرکدام از عبارات بالا پخته‌تر می‌شوند به عبارتی بعدی تبدیل می‌شوند.

همچنین برای تحلیل نیز داریم:

توصیفی ← تشخیصی ← پیشگویانه ← تجویزی

وقتی صحبت از تحلیل می‌شود یعنی ما باید عمل data mining انجام دهیم. با انجام این عمل data را به دانش تبدیل می‌کنیم. هر چه این دانش بدست آمده کاملتر باشد تحلیل ما باارزش تر است. یعنی سازمان نیز سود بیشتری بدست می‌آورد.

مثال: در ابتدا ما می‌گوییم "پنج" . این عدد به تنهایی دانشی به ما نمی‌دهد؛ اما در ادامه می‌گوییم "پنج مشتری". و در آخر میگوییم "در این ماه فروشگاه ما پنج مشتری ریزش داشته است". این می‌شود دانش کاملتر که سبب تصمیمات در جهت رفع کاستی ها یا بهبود کسب و کار می‌شود.

همینطور هر چه تحلیل سطح بالاتری انجام شود مسلما اطلاعات بیشتر و مفیدتری به ما می‌دهد.

نکته: تحلیل و انواع آن نیز جزومفاهیم ابتدایی در big data می‌باشند.

- ✓ **تحلیل توصیفی**

این نوع تحلیل کمترین پیچیدگی را دارد و به این معناست که در مورد چیزهایی که در قبل اتفاق افتاده حرف می‌زند و ما کاری به حال یا آینده نداریم. در این نوع تحلیل ما جواب سوالهایی را می‌دهیم که مربوط به گذشته است. پس در این نوع تحلیل گذشته بسیار مهم است.

نمونه سوالاتی که در این نوع تحلیل بیان می‌شود بصورت زیر است:

- در ۱۲ماه گذشته فروش شما چطور بوده است؟
- چه مقدار افرادی که با بخش پشتیبانی تماس گرفته‌اند در زمان مناسبی جواب گرفته‌اند؟
- برای هر کدام از این آژانس‌های طرف قرارداد ما چه مقدار کمیسیون دریافت کرده‌ایم؟

نکته: دقت و صحت جواب این سوالات کاملا تضمین شده است؛ چون برای گذشته است.

نکته: سوالها اغلب در این بخش با What شروع می‌شوند.

نکته: ۸۰٪ تحلیل‌های سازمانها توصیفی است.

تحلیل گذشته خوب است، اما مهمتر آنست که بتوانید به این برسید که چکار کنید که در آینده موفق باشید. و این میشود سطوح بالاتر.

مثال: وقتی که شما بر اثر بیماری به پزشک مراجعه میکنید مهم است که او از احوال روز گذشته(توصیف) شما آگاه شود، سپس با تشخیص بیماری و تجویز درست به بهبود شما کمک می‌کند.

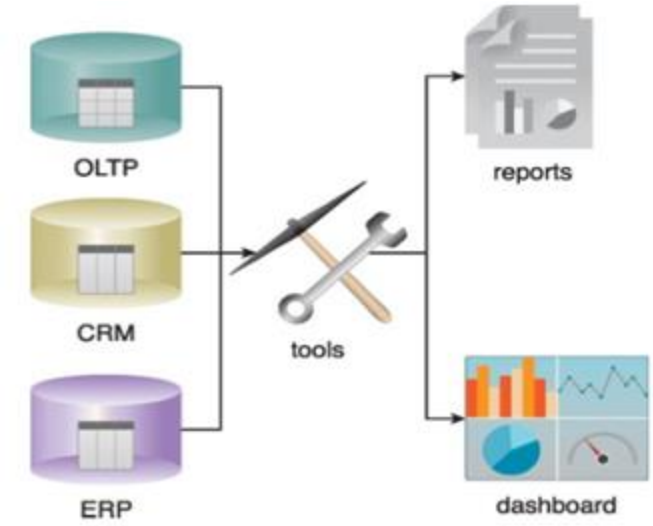
در سازمانها نیز مشابه انسانها مریضی پیش می‌آید، برای بهبود آن نیز ابتدا باید تشخیص داد منشا و دلیل بیماری چیست؛ سپس با تجویز درست بیماری(مشکل) را رفع کرد.

در تصویر زیر داریم:

OLTP(Online Transaction Processing) : سیستم سروری که داده‌های Transaction شما در آن ثبت می‌شود. تراکنش ها و کارمندان سازمان شما را مشخص می‌کند.

CRM(Customer Relationship Management) : مدیریت رابطه با مشتری. در آن ریزش و وفاداری مشتری‌ها ثبت می‌شود.

ERP(Enterprise Resource Planning) : اغلب در سازمانها یک سیستم یکپارچه است که چند سیستم مختلف (مانند فروش، انبار و ...) را به هم متصل کرده است.



اغلب در تحلیل‌های توصیفی از سیستم‌های OLTP، CRM و ERP گزارش تهیه می‌کنید. یا حتی می‌توان از داشبوردها بعنوان گزارش استفاده کرد. داشبوردها در واقع نمایش گرافیکی گزارشات هستند؛ که البته در اینجا فقط گزارشات مربوط به گذشته را به ما نشان می‌دهند واطلاعاتی از حال و آینده به ما نمی‌دهد.

نکته: در حالت کلی داشبورد میتواند گذشته، حال و آینده را نشان دهد.

✓ تحلیل تشخیصی

در این نوع تحلیل به ما دلیل اتفاقات افتاده در گذشته را به ما نشان می‌دهد.

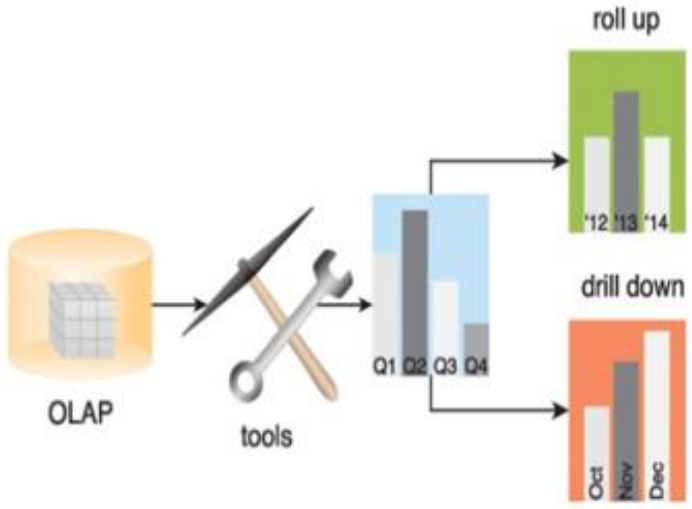
نمونه سوالاتی که در این نوع تحلیل بیان می‌شود بصورت زیر است:

- چرا فروش فصل تابستان کمتر از بهار بوده است؟
- چرا مشتری های ما کم شده‌اند؟
- چرا بیماران ما در سه ماه گذشته دوباره رجوع کرده‌اند؟

نکته: همه‌ی سوالات در این بخش با "چرا" (Why) شروع می‌شوند.

با یافتن پاسخ این پرسشها تحلیل تشخیصی ما انجام شده است.یافتن این علتها و درنتیجه این نوع تحلیل واقعا کار سختی است.

برای مثال وقتی بخاطر سرماخوردگی به پزشک مراجعه می‌کنید و پزشک علت را از شما جویا می‌شود شما خربزه خوردن را دلیل بر آن بدانید و یادتان نباشد که ممکن است جلو کولر نشستن هم مسبب این بیماری شده باشد. به همین ترتیب بعضی دلایل یا به چشم نمی‌آیند یا فراموش می‌شوند و اینگونه ممکن است به تشخیص غلط بیانجامد. البته می‌توان با تجربه احتمال قوی یک واقعه را تشخیص داد. مثلا اگر شما ۱۰بار با خوردن خربزه سرما خوردید میتوانید بگویید به احتمال قوی خوردن خربذه علت سرماخوردگی من است.(اما باز هم صددرصد نیست)



در تحلیل‌های سطح دوم از OLAP استفاده می‌شود.(که در جلسه آینده تفاوت آن با OLTP ذکر می‌شود)

✓ تحلیل پیشگویانه

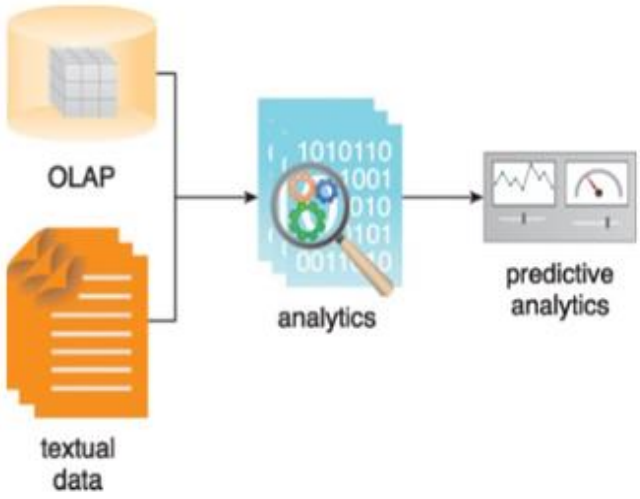
در سطح اول بررسی کردیم که چه اتفاقی افتاده است، در سطح دوم علت اتفاقات را بررسی کردیم، در سطح سوم پیش بینی می‌کنیم که اگر این راه‌حل را پیاده کنم چه اتفاقی خواهد افتاد؟ (یعنی قبل از اقدام به پیاده‌سازی راه‌حل). پس به آینده مربوط می‌شود.

نمونه سوالاتی که در این نوع تحلیل بیان می‌شود بصورت زیر است:

- چه اتفاقی می افتد اگر یکی از افرادی که وام گرفته آن را پرداخت نکند؟
- اگر داروی B را به جای داروی A استفاده کنم، وضعیت افراد سرطانی مورد آزمایش چگونه می‌شود؟

نکته: اغلب سوالات در این بخش با "چه می‌شود اگر" (What if) شروع می‌شوند.

همانطور که در سوال دوم مشخص است انجام عمل و دیدن نتیجه ممکن است کار بسیار سخت یا حتی خطرناکی باشد؛ به همین علت ابزارهایی برای انجام این پیش‌بینی‌ها وجود دارند.

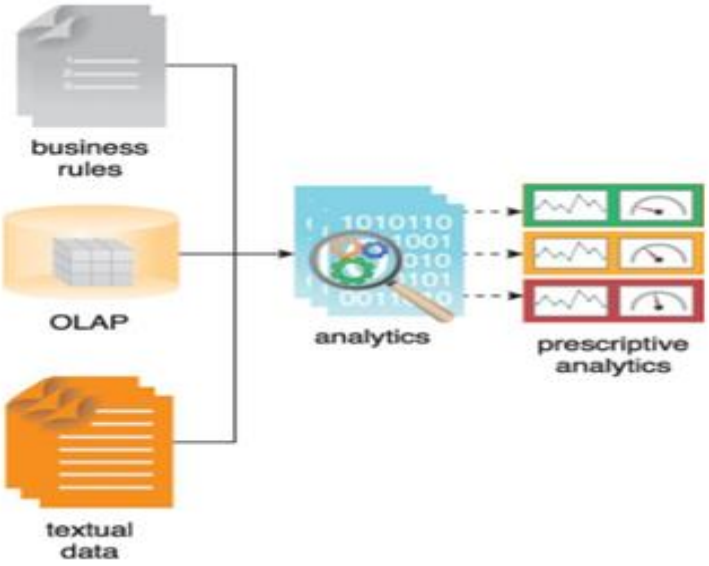


✓ تحلیل تجویزی

در این مرحله تصمیم‌گیری قطعی برای انجام راه‌حل و جامه عمل پوشاندن به آن صورت می‌گیرد.

نمونه سوالاتی که در این نوع تحلیل بیان می‌شود بصورت زیر است:

- از میان سه دارویی که پیش بینی کردیم خوب است، بالاخره کدام را استفاده کنم؟



در این مرحله نه تنها باید نوع تجویز، بلکه باید دلیل تجویز نیز مشخص باشد.

برای انجام تحلیل تجویزی علاوه بر داده‌های سازمان باید داده‌هایی نیز از بیرون جمع‌آوری کرد. مثلا اطلاعات راجع به رقبا، نظرات مردم و ...

#جلسه دوم، پنجشنبه ۱۳۹۵/۷/۱۵

تفاوت بین OLTP و OLAP :

OLTP همان تراکنش‌هایی هستند که شما پردازش می‌کنید. مثل بانکها، اتوماسیونهای اداری و ...

OLAP(Online Analytical Processing) پردازش تحلیلی برخط است.

در تحلیلهای سطح پایین اطلاعات گذشته را لازم دارید که اغلب سیستم هایی که این اطلاعات را به شما می‌دهند را OLTP می‌گویند. اما وقتی به سطوح بالاتر می‌روید به اطلاعات تروتمیزتر و دسته‌بندی شده‌تر احتیاج دارید که این اطلاعات را هم OLAP به ما می‌دهد.

تفاوت بین این دو را در یک سری مشخصه که در جدول زیر آمده است بررسی می‌کنیم:

اغلب در سیستمهای OLTP فقط داده‌های فعلی ما مهم هستند و خیلی به داده‌های گذشته فکر نمی‌کنیم، اما برای OLAP به داده‌های قبلی هم نیاز داریم؛ زیرا ما می‌خواهیم با استفاده از سوابق گذشته برای آینده تصمیم بگیریم. و به همین علت در OLAP با حجم عظیم‌تری از داده‌ها روبه‌رو هستیم. پس OLAP از داده‌های تاریخمند استفاده می‌کند.

در سیستم OLTP بازده تراکنش مهم است(بازده یعنی تعداد تراکنش‌های انجام شده در واحد زمان). هر چه بازده بیشتر باشد. (بازده یا Throughput) اما در سیستم OLAP پرسشهایی که پرسیده می‌شود و زمان پاسخ مهمتر است.

برای درک بهتر می‌توان مثالی بیان کرد؛ بصورت معمول تعداد سوالات یک کارمند معمولی در رابطه با پروژه بیشتر و جزئی‌تر است، اما تعداد سوالات مدیر همان پروژه تعداد سوالات کمتر و کلی‌تری دارد. به پیرو آن برای پاسخگویی به کارمند چون سوالات جزئی است به دیتابیس کوچکتری برای پاسخگویی نیاز داریم، اما برای پاسخگویی به مدیر به دیتابیس بزرگتری نیاز داریم، چون سوالاتش کلی است و شاید خلاصه اطلاعات کل پروژه را با یک بیان یک سوال از ما خواستار باشد. (در اینجا دیتابیس پاسخگویی به کارمند مصداق OLTP و دیتابیس پاسخگویی به مدیر مصداق OLAP می‌باشد)

مثلا کارمند می‌پرسد "نمره دانشجوی x در درس y چند بود؟" اما مدیر میپرسد "میانگین نمرات کلاسهای استاد z در ترم اول چقدر بوده است؟"

اغلب در OLTP از مدل رابطه‌ای استفاده می‌شود و تمام جزئیات را دارید، اما در OLAP از مدل خلاصه شده استفاده می‌شود.

طبیعی است که تعداد کارمندان OLTP بیشتر از OLAP است؛ چون OLAP خیلی خاص است.(تعداد کارمندان همیشه کمتر از تعداد مدیران است)

در OLTP پرس‌وجوها ساده است، اما در OLAP پرس‌وجوها پیچیده‌تر است.

اغلب طراحی پایگاه‌داده‌ها در OLTP مبتنی بر ER است، اما در OLAP موضوع‌گرا، ستاره،/برفگونه است.

کاربران OLTP کارمندان دفتری و ساده هستند، اما کاربران OLAP کارگر دانش(کسی که از دانش و تخصص خودش درآمد کسب می‌کند؛ برای مثال تحلیل‌گر ،مدیر و مدیر اجرایی) هستند.

ویژگی مورد بررسی	سیستم پردازش تراکنش برخط OLTP	سیستم پردازش تحلیلی برخط OLAP
مشخصه	پردازش عملیاتی	پردازش اطلاعاتی
گرایش	تراکنش	تحلیل
کاربر	کارمند دفتری و متخصصین پایگاه داده‌ها	کارگر دانش (برای مثال تحلیل‌گر ،مدیر و مدیر اجرایی)
کارکرد	عملیات روزانه	نیازمندی‌های اطلاعاتی بلند مدت جهت پشتیبانی از تصمیم‌گیری
طراحی پایگاه داده‌ها	مبتنی بر ER، کاربردگرا	ستاره،/برفگونه، موضوع‌گرا
داده‌ها	داده‌های جاری و بهنگام‌شده	داده‌های تاریخمند
تلخیص	ابتدایی، با جزییات زیاد	خلاصه شده و یکپارچه
دید	دارای جزییات، رابطه‌ای	خلاصه شده، چندبُعدی
واحد کاری	کوتاه، تراکنش ساده	پرس‌وجوی پیچیده
دستیابی	خواندن و نوشتن	اغلب خواندن
مرکز توجه	ورود داده‌ها	خروج اطلاعات
عملیات	دستیابی بر اساس کلید اصلی	پیمایش زیاد داده‌ها
تعداد رکوردها برای بازیابی	ده‌ها	میلیون‌ها
تعداد کاربران	هزاران	صدها
اندازه پایگاه داده‌ها	گیگابایت	بیشتر از ترابایت
اولویت	کارایی و قدرت دسترسی بالا	انعطاف‌پذیری بالا
متریک	بازده تراکنش	بازده پرسش و زمان پاسخ

ادامه بررسی انواع تحلیل

هرچه از پایین به سمت بالا حرکت می‌کنیم پیچیدگی تحلیل بیشتر شده، و درعوض دانش و خردی که از آن کسب می‌کنیم نیز بیشتر و بهتر است.

نکته: در تحلیل توصیفی فقط از OLTP استفاده شده است؛ زیرا این نوع زیاد تحلیل نیست و فقط گونه‌ای توصیف است.

نکته: در تحلیل های سطح بالاتر چون باید به چراها و راه‌حلها پاسخ گفت از OLAP استفاده می‌شود.

در تصویر مربوط به تحلیل تشخیصی در عبارت roll up و drill down داریم که در اینجا با ذکر یک مثال به مفهوم این دو عبارت می‌پردازیم. (البته این عبارات بعنوان drill up و roll down نیز می‌آیند)

مثال۱: مقدار فروش شعب ما به تفکیک ماه و شماره شعبه مشخص شده است. مدیر با نگاه به آن متوجه یک عدد غیرعادی در میزان فروش یکی از شعب در ماه خرداد می‌شود، او تمایل دارد با کلیک روی آن عدد جزئیات فروش ماه آن شعبه مشخص شود تا به علت آن پی ببرد، این می‌شود drill down. مدیر می‌خواهد فروش کلیه شعب یک استان را سرجمع در یک ماه ببیند این می‌شود roll up. و به همین ترتیب بالاتر و پایین‌تر می‌رود.

نتیجه: roll up یعنی از دید بالاتر و کلی‌تر، drill down یعنی از دید پایین‌تر و جزئی‌تر.

در ابزارهای تحلیل بیگ‌دیتا نیز این امکان (drill down و roll up) وجود دارد.

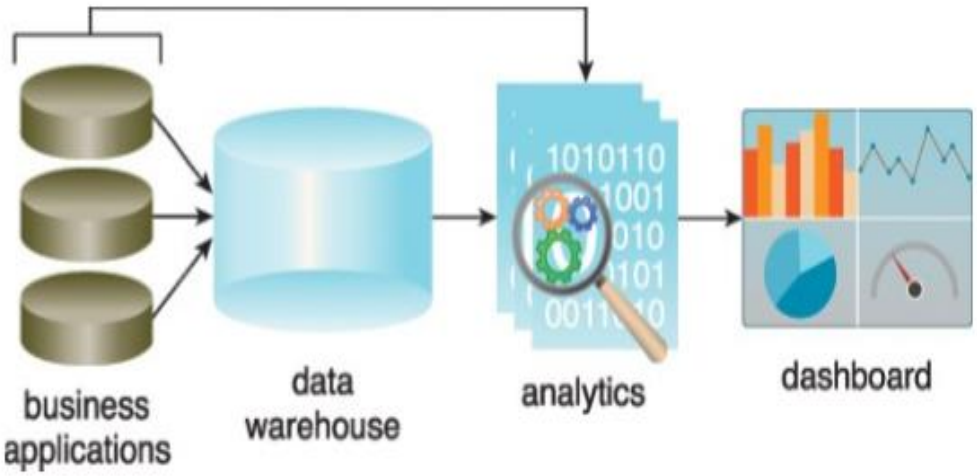
مثال۲: برای دانشگاه. نمرات دانشجویان در یک درس ← به تفکیک استاد ← به تفکیک دانشکده

همچنین حتی می‌توان در یک گزارش برای موارد غیرعادی و استثنا یک توضیح مختصر قرار داد تا با قرارگیری ماوس روی آن توضیح نمایش داده شود.

در تحلیل پیشگویانه اغلب بوسیله‌ی یک سری داشبوردهای خاص اطلاعات را نشان می‌دهد و OLAP هم نقش اصلی ایفا می‌کند.

نرم اغزارهایی که اغلب در روش تجویز استفاده می‌شوند، نرم افزارهای تولید داشبورد هستند. اغلب در داشبوردها یک استکیومتر داریم. در داشبورد سلیقه مهم است ولی مهم آن چیزی است که می‌خواهید نشان دهید.

نکته: یکی از محاسن اصلی داشبورد این است که با یک نگاه اجمالی اطلاعات زیادی می‌توان از آن بدست آورد.

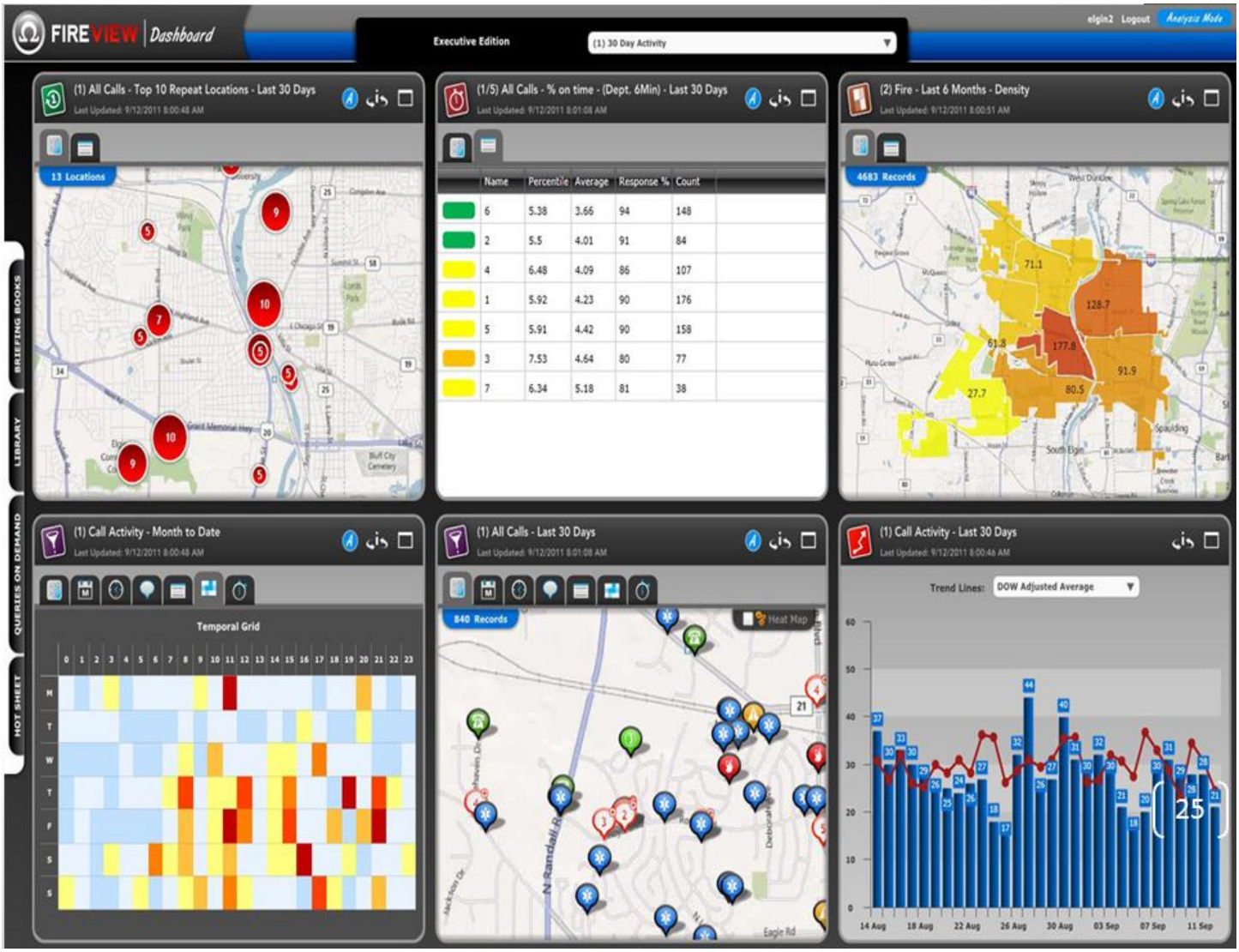


BI can be used to improve business applications, consolidate data in data warehouses and analyze queries via a dashboard.

نمونه هایی از داشبوردها در زیر آورده شده است:

KPI dashboard

A KPI dashboard acts as a central reference point for gauging business performance.



یک سوال اساسی: ما داده زیاد داریم، آیا داشبورد توانایی نشان دادن همه‌ی داده‌ها را دارد؟

واضح است که خیر. بلکه ما باید از یک سری شاخص‌ها استفاده کنیم. اهمیت شاخص‌ها به سادگی نیست، اما دسته‌ای از شاخص‌ها که زیاد هم نیستند و برای آینده مناسبند به نام شاخص‌های کلیدی عملکرد(KPI) هستند که در تولید داشبورد استفاده می‌شوند.

- فرض کنید میخواهیم سیستم حضور-غیاب کارمندان را بررسی کنیم. شاخص‌هایی که می‌توانیم بررسی کنیم از قبیل ساعت حضور در ماه. شاید نتوانیم شاخصهای مهم پیدا کنیم اما حداقل میتوان شاخصهایی که همپوشانی دارند را تعیین و حذف کرد.
- فروش سال گذشته شاخص است، اما KPI نیست، چون فقط گذشته را نشان می‌دهد و برای آینده مناسب نیست.
- در داشبورد ماشین ننوشته در صندوق عقب چه خبر است چون مهم نیست، اما "دمای موتور" و "مقدار بنزین" آورده شده چون شاخص هستند. شاید اطلاعات مهم دیگری هم وجود داشته باشند اما شاخص نیستند.
- در بخش اورژانس بیمارستان رئیس بیمارستان چه چیزی از آن بداند خوب است؟ اطلاعاتی لازم داریم که بتوان بصورت آنلاین از آن استفاده کرد. مثل "نسبت پزشک به بیمار" یک شاخص مهم است. یا "تعداد تختهای خالی". مثلا "دانستن تعداد زن و تعداد مرد" مهم است ولی شاخص نیست.

شاخص‌ها انواع مختلف دارند. مثلا شاخص‌هایی که به گذشته مربوط می‌شوند که می‌توان از آنها برای تحلیل میزان فروش استفاده کرد. و انواع مختلف دیگر.

شاخص ها در big data بسیار مهم هستند. برای کسب اطلاعات بیشتر در مورد شاخص‌ها می‌توانید به کتاب هوش تجاری دکتر اسماعیلی رجوع کنید.

ویژگی‌های Big Data

The Five Vs of Big Data

به چه چیزهایی می‌گوییم big data؟ آیا ۳۰۰گیگابایت می‌شود big data؟ ۳ترابایت چطور؟

اغلب کتابهای big data سه ویژگی اصلی برای big data درنظر می‌گیرند و آنها را با عنوان 3v میشناسند.

۱) Volume ، حجم
۲) Velocity ، سرعت
۳) Variety ، تنوع

نکته: در بعضی از کتابها حتی بین ۴ تا ۶ ویژگی دیگر نام برده‌اند، اما این سه ویژگی نام برده شده ویژگی‌های اصلی هستند.

حجم داده‌ها وقتی زیاد باشد به آن می‌گوییم big data، ولی این سوال پیش می‌آید اگر حجم زیاد نباشد یعنی داده‌ها big data نمی‌توانند باشند؟! گاهی سرعت تولید داده‌ها بسیار زیاد است آنها را هم جزو big data می‌دانیم؛ مانند داده‌های تولید شده از دوربین‌های نظارتی. اگر داده‌های تولیدی تنوع زیاد هم داشته باشد آن را جزو big data می‌دانیم؛ مانند اینکه داده‌ها از انواع مختلفی باشند، مثل متنی، رابطه‌ای، صوتی و ... باشند.

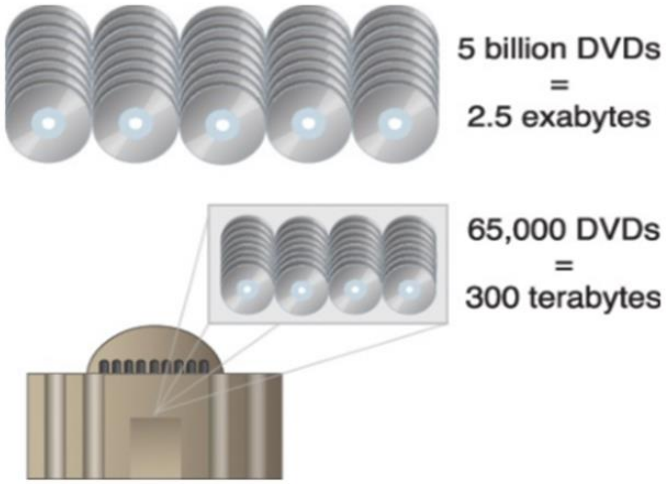
پس وجود حجم در big data الزامی نیست و کافی است داده ها یکی از این سه ویژگی بالا را داشته باشند، جزو big data به حساب می‌آیند.(البته اغلب big data هرسه ویژگی را با هم دارد)

امروزه اغلب سازمانهایی که با big data کار می‌کنند داده‌هایشان فراتر از ترابایت است.

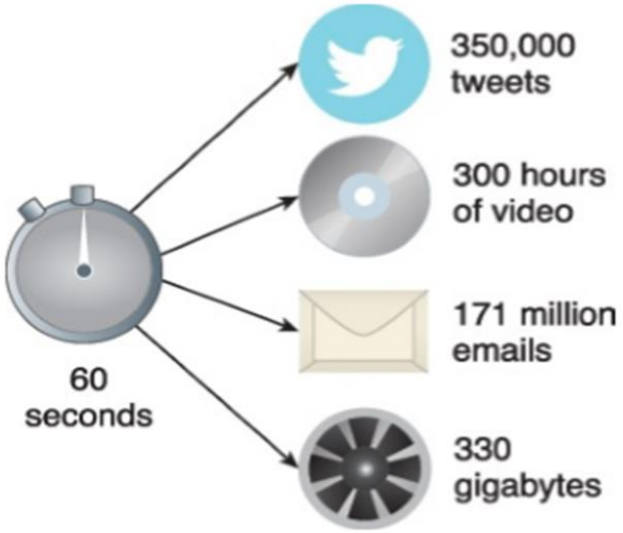
مقادیر بایت‌ها					
ده‌دهی		دودویی		نامشخص	
نام (نماد)	استاندارد بین‌المللی	نام (نماد)	مقدار	کاربرد عمومی (درکشورها)	مقدار
کیلوبایت (kB)	$۱۰^{۳}$	کیبی‌بایت (KiB)	$۲^{۱۰}$	کیلوبایت (KB)	$۲^{۱۰}$
مگابایت (MB)	$۱۰^{۶}$	مبی‌بایت (MiB)	$۲^{۲۰}$	مگابایت (MB)	$۲^{۲۰}$
گیگابایت (GB)	$۱۰^{۹}$	گیبی‌بایت (GiB)	$۲^{۳۰}$	گیگابایت (GB)	$۲^{۳۰}$
ترابایت (TB)	$۱۰^{۱۲}$	تبی‌بایت (TiB)	$۲^{۴۰}$	ترابایت (TB)	$۱۰^{۳}*۲^{۳۰}$
پتابایت (PB)	$۱۰^{۱۵}$	پبی‌بایت (PiB)	$۲^{۵۰}$	پتابایت (PB)	$۱۰^{۶}*۲^{۳۰}$
اگزابایت (EB)	$۱۰^{۱۸}$	اگزبی‌بایت (EiB)	$۲^{۶۰}$	اگزابایت (EB)	$۱۰^{۹}*۲^{۳۰}$
زتابایت (ZB)	$۱۰^{۲۱}$	زبی‌بایت (ZiB)	$۲^{۷۰}$	زتابایت (ZB)	$۱۰^{۱۲}*۲^{۳۰}$
یوتابایت (YB)	$۱۰^{۲۴}$	یوبی‌بایت (YiB)	$۲^{۸۰}$	یوتابایت (YB)	$۱۰^{۱۵}*۲^{۳۰}$
سوتابایت (SB)	$۱۰^{۲۷}$	سوبی‌بایت (SiB)	$۲^{۹۰}$	سوتابایت (SB)	$۱۰^{۱۸}*۲^{۳۰}$

برای آنکه بهتر این حجم داده را درک کنید به تصویر زیر دقت کنید:

۳۰۰ترابایت یعنی، ۶۵هزار DVD . که این حجم بسیار بالایی است. البته باید درنظر داشته باشیم کهاین حجم همچنان در حال افزایش است. پس کا بر روی این داده‌ها بسیار مهم است.



تا اینجا در مورد حجم گفتیم. اما سرعت تولید داده‌ها، با توجه به تصویر زیر در یک ثانیه مثلا حدود ۳میلیون ایمیل ارسال شده است.



و اما تنوع، تصویر زیر گویای تنوع زیاد موجود در داده‌هاست.

حال به سراغ دیگر ویژگی های نام برده شده در تصویر قبل می‌رویم تا با دیگر Vها آشنا شویم.

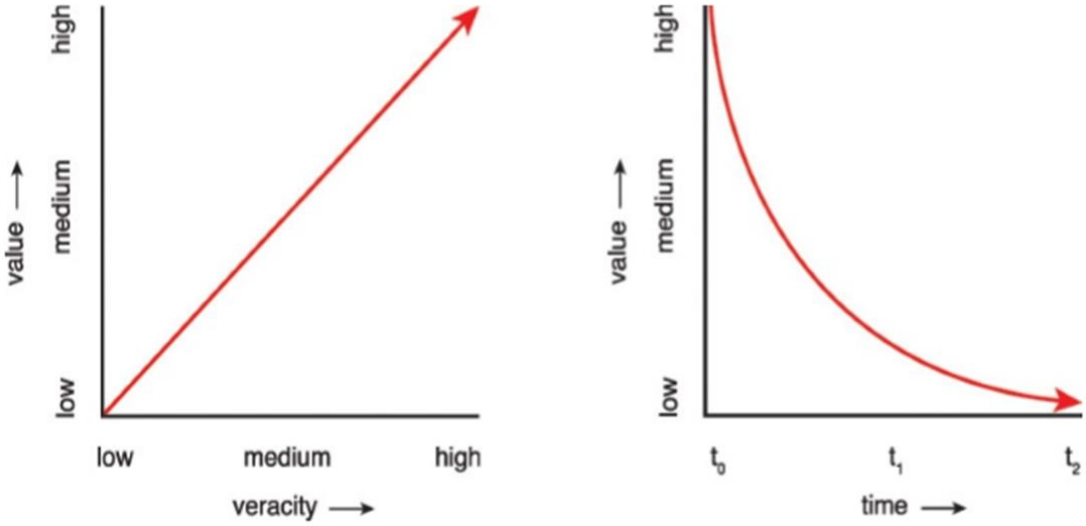
۴) Veracity ، درستی

البته به بیانی میتوان گفت که این عبارت جزو ویژگی یا یک تعریف برای big data نیست بلکه خوب است که کیفیت داده‌ها بالا باشد تا تحلیل ما درست شود. کیفیت داده‌ها یعنی داده‌های تمیز. داده‌های کثیف عبارتند از: داده‌های ناموجود(missing)، داده‌های تکراری، حجم بالا، واحدهای اندازه‌گیری متفاوت، داده‌های پرت و ... . البته خیلی از اوقات نمیتوان داده‌های کثیف را به صفر رساند، و حتی گاهی این کثیفی ها برای ما مفید است. مثل تکرار، که دسترسی پذیری را بالا می‌برد.

۵) Value ، ارزش

الزاما حجم زیاد داده ها برای تحلیل ما مفید نیست و این سبب پیدایش V پنجم می‌شود. یعنی داده باید باارزش باشد تا مفید باشد. هر چه داده‌ها تکراری تر شوند ارزش آنها پایین می‌آید ولی فرمولهایی برای تعیین ارزش داده‌ها داریم.

به نمودارهای زیر توجه کنید:



در نمودار سمت راست ارزش نسبت به زمان سنجیده می‌شود، هر چه که داده‌ها مربوط به زمانهای گذشته‌تری هستند ارزش آنها کمتر می‌شود.

در نمودار سمت چپ ارزش نسبت به درستی سنجیده می‌شود، هر چه که داده‌ها تمیزتر باشند ارزش آنها بیشتر می‌شود.(فاز تمیزکاری داده‌ها فاز سختی است اما چون به چشم نمی‌آید برای آن بهایی پرداخت نمیشود و اغلب نادیده گرفته میشود.)

نکته: هرچه داده‌ها حجیم‌تر شوند احتمال اینکه داده‌های بی‌ارزش در آن بیشتر شود زیادتر است.

سوال اینجاست که با چه نرم‌افزاری حجم داده‌ها را کنترل کنیم؟ با چه ابزاری داده‌ها را تحلیل کنیم؟ با چه ابزاری سرعت داده‌ها را کنترل کنیم؟ با چه ابزاری کیفیت داده‌ها را بالا ببریم؟ برای پاسخ به این سوالها باید با ابزارهای متنوع موجود در big data و کاربردهای آنها آشنا شوید.

#جلسه سوم، پنجشنبه ۱۳۹۵/۷/۲۹

چالش های big data:

- تعریف big data (به چه داده‌هایی big data می‌گویند)
- ذخیره کردن big data (چون ذخیره ی این همه داده هزینه بر است)
- بازیابی اطلاعات از این داده های حجیم
- پردازش داده‌ها
- تحلیل داده‌ها
- مدیریت این داده ها و ...

- اگر بخواهم داده های حجیم را ذخیره کنم چگونه ذخیره کنم؟ آیا پلتفرمی وجود دارد برای این کار؟ بله. Hadoop یکی از این پلتفرمهاست.

- این داده های حجیم را چگونه تحلیل کنیم؟ آیا ابزاری برای این تحلیل وجود دارد؟ و آیا این ابزار با ابزارهای غیر big data فرق دارد؟ بله. ابزاری مثل mahout این تحلیل را برای big data انجام می‌دهد.

- آیا کامنتها و توضیحات شبکه های اجتماعی را در یک جدول می‌توان ذخیره کرد؟ بله اما بسیار سخت است و فضای زیادی هدر می‌رود؛ زیرا کامنتها طول مشخصی ندارند و اینگونه اکثر رکوردها خالی می‌ماند. پس مدل جدولی برای آنها مناسب نیست و به یک پایگاه داده غیررابطه‌ای احتیاج داریم. به ابزارهایی که برای پایگاه داده‌های غیررابطه ای استفاده می شوند NOSQL می‌گویند. یعنی نه فقط SQL ؛ بلکه انواع دیگری هم هستند مثل document base ، key/value ، گراف و... . نرم افزارهایی مثل mongo db و neo4j این‌گونه داده‌ها را ذخیره می‌کنند.

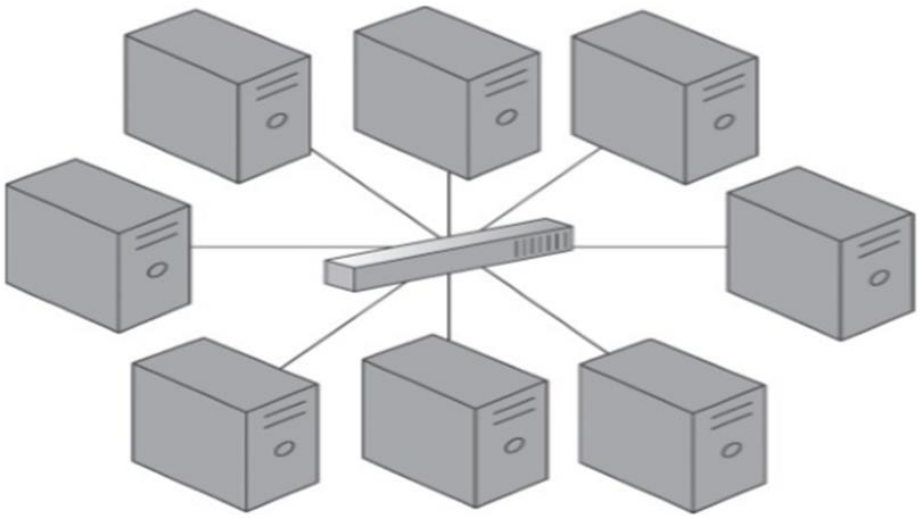
برای رفع هرکدام از چالش های big data یک نرم افزار وجود دارد. که در ارائه های این درس با این نرم‌افزارها آشنا می‌شوید.

ادامه اصطلاحات مربوط به big data:

Cluster

Cluster، مجموعه ای از گره‌های محاسباتی است. گره محاسباتی(Computing node)، یک کامپیوتر با یک سری مشخصات است که CPU، هارددیسک و... دارد و این کامپیوترها به هم متصل هستند و باهم یک cluster را تشکیل می‌دهند.

اغلب در شرکتهای بزرگ یک cluster را در یک Rack می‌بینید. گره‌های محاسباتی را در یک Rack قرار دارد و اغلب این گره ها در یک خوشه با یک شبکه با سرعت بالا به هم متصل هستند. اغلب شرکتهای بزرگ که با big data سر و کار دارند cluster دارند. {cluster یعنی مجموعه ای از سرورها. یا به بیان بهتر باید بگوییم مجموعه ای از گره های محاسباتی}



شکل بالا یک cluster را نشان می‌دهد.

Replication

Replication به معنی تکرار کردن است. یکی از تکنیکهای کتاب برای مواجه شدن با big data، تکرار است؛ که امروزه در بسیار از نرم افزارها هم انجام می‌شود.

حسن تکرار داده‌ها در big data بالا بردن دسترسی(availability) است. با ثبت کردن داده‌ها در چندجا باعث سرعت در دسترسی می‌شود چون هر درخواست به نزدیکترین مکان رجوع می‌کند.

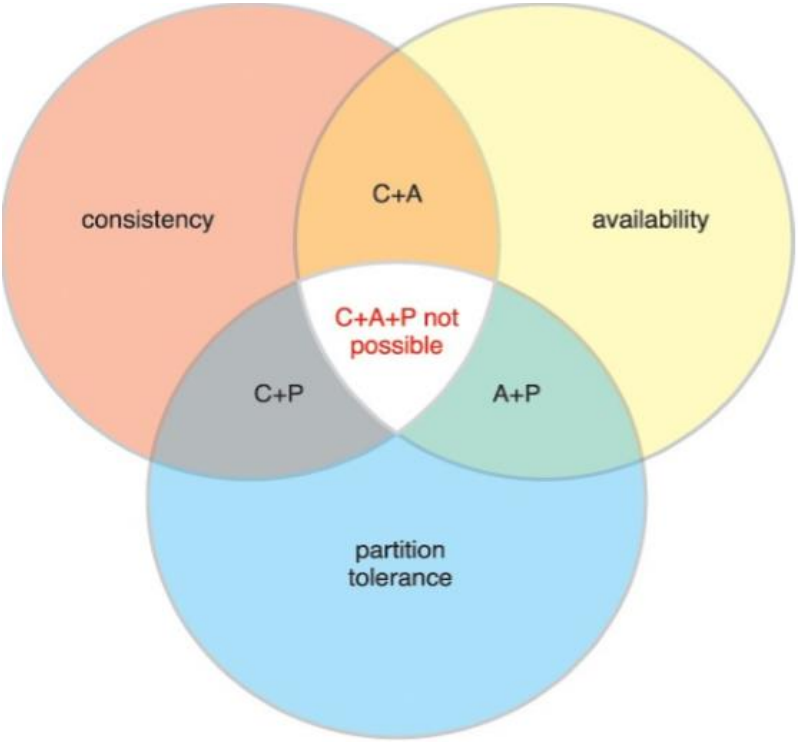
حسن دیگر قابلیت اعتماد(reliability) است. یعنی اگر یکی از مخازن خراب شود داده‌های ما از بین نرفته و در جای دیگر نیز وجود دارد.

البته با داشتن تکرار همه چیز بر وفق مراد نیست و یک مشکلی بوجود می‌آید به نام ناسازگاری(inconsistency)، که چالش آن consistency است. یعنی مقادیر باید در تکرارها یکی باشند. البته در بسیاری از جاها صددرصد consistency ضروری نیست. مثلا بانکها باید حتما consistent باشند. (بانکها بوسیله ی رعایت قاعده ی ACID این سازگاری را تضمین می‌کنند.)

نکته: قاعده ACID: خواص تراکنش است.(Atomic، consistency، Isolation، Durability)

متأسفانه ما باید بین این محاسن تکرار، انتخاب کنیم. زیرا برخی از این عاملها در کنار هم محقق نمی‌شوند و نمی‌توانیم آنها را باهم داشته باشیم (از لحاظ علمی هم نمی‌توانیم همه چیز را با هم داشته باشیم). بطور واضح‌تر باید بگوییم اگر availability و reliability بالا برود نمی‌توانیم consistency صددرصد داشته باشیم. بخاطر اینکه اصلا این سه مورد با هم جمع نمی‌شوند.

تئوری CAP



اگر در شکل cluster کانکشن ما قطع شود چه اتفاقی می افتد؟ اغلب این توپولوژی ها طوری چیده می‌شوند که tolerance باشند. یعنی قدرت تحمل داشته باشند. یعنی اگر کانکشن قطع شد و یک نفر اطلاعاتی را خواست و نزدیکترین سرور به او قطع بود بتواند اطلاعات را از سرور دیگر دریافت کند. به نظر میرسد replication این امکان را برقرار کند.

Partition tolerance یعنی تحمل تکه شدن.

همانطور که شکل بالا نشان می‌دهد، هر سه ویژگی با هم جمع نمی‌شوند و فقط دوتا دوتا با هم تجمیع می‌شوند.

این مسئله اصلا بد و خوب ندارد و شما باید طبق کاربردتان انتخاب کنید که کدام موارد برایتان مهمتر است و به آن نیاز دارید.

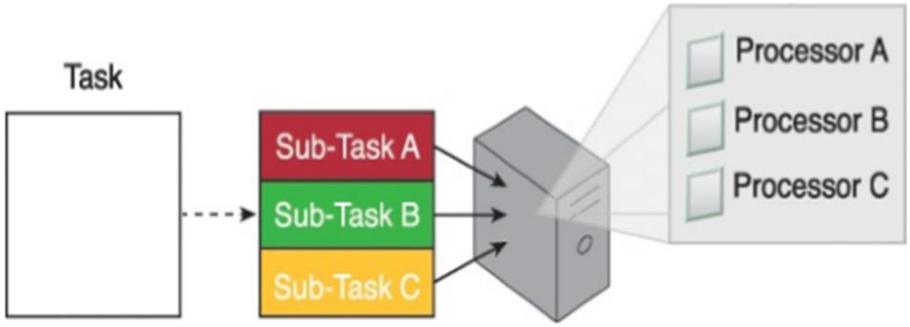
سوال امتحانی: تئوری cap را تعریف کنید؟ در این تئوری c مخفف Consistency و به معنی سازگاری، a مخفف Availability و به معنی دسترسی پذیری، p مخفف Partition tolerance و به معنی تحمل تکه شدن است. و این تئوری به ما میگوید هیچ وقت نمیتوانیم شرایطی را ایجاد کنیم که هر سه مورد را با هم داشته باشیم.

سوال امتحانی: مثالی بزنید که c و a هست و p نیست و بگویید چرا p نیست؟ با دلیل و برهان. {این سوال برای سه حالت مطرح می‌شود}

بعنوان مثال، وقتی یک disconnectی اتفاق می افتد و یک نسخه از تکرار در آنجاست پس سازگار نگه داشتن شدنی نیست و تحمل تکه شدن کمتر می‌شود.

(Multi-processor) Parallel Data Processing

پردازش داده بصورت موازی.



در این حالت یک task به چند sub-task تبدیل می شود. در اینجا پردازش مربوط به همه‌ی sub-taskها را یک سیستم انجام می‌دهد؛ که آن سیستم دارای چند processor است.

نکته: پس اگر سیتم ما یک processor(پردازشگر) داشته باشد موازی نیست، بلکه همروند است، اما آنقدر بین کارها سوئیچ می‌کند که ما تصور می‌کنیم کارها بصورت موازی انجام می‌شود. همروند taskها را توسط یک پردازشگر لابه‌لای هم انجام می‌دهد، اما در موازی taskها توسط چند پردازشگر بصورت موازی انجام می‌شوند.

پس در موازی سیستم دارای چند processor، یک حافظه مشترک، I/O مشترک و... می‌باشد. در کل همه چیز مشترک و یکی است فقط پردازشگرها متفاوت و چندتا هستند.

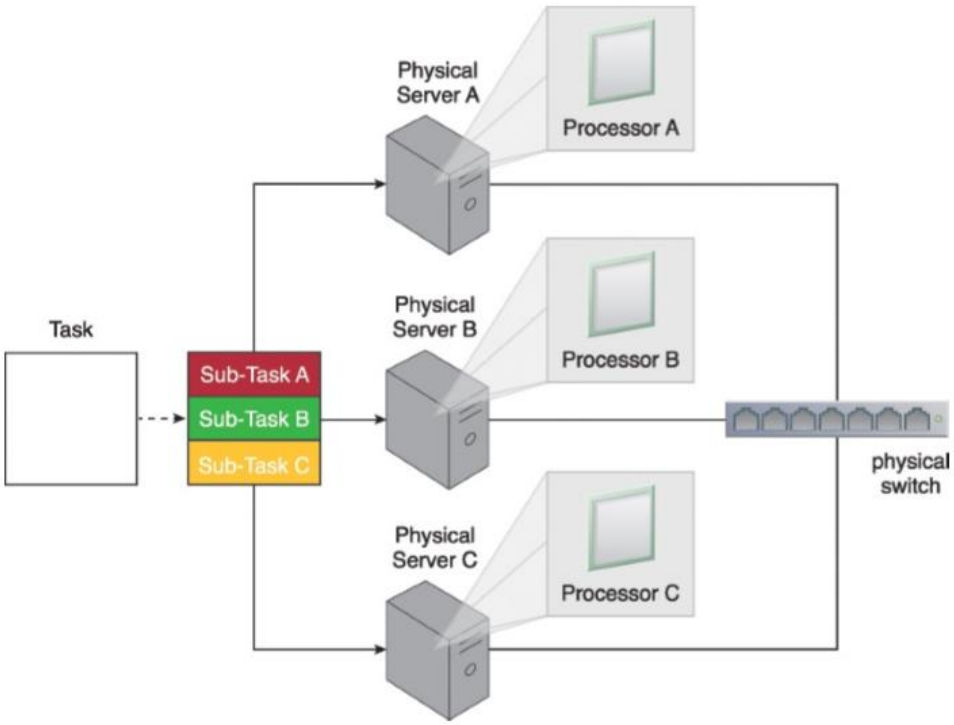
این سیستم را گاهی tightly coupled هم می‌گویند. (coupled: زوج، tightly:تنگ) زیرا همه چیز در یک کامپیوتر تنگ هم قرار دارند.

Distributed Data Processing

پردازش داده بصورت توزیع شده.

در این حالت نیز یک task به چند sub-task تبدیل می شود. پردازش مربوط به هر sub-task را یک سیستم انجام می‌دهد؛ که آن سیستم دارای یک processor مخصوص به خود است. پس در توزیع شده چند سیستم که هر کدام دارای یک processor، یک حافظه مجزا، I/O مجزا و... می‌باشند و وظایف را انجام می‌دهند. در کل هیچ چیز مشترک نیست.

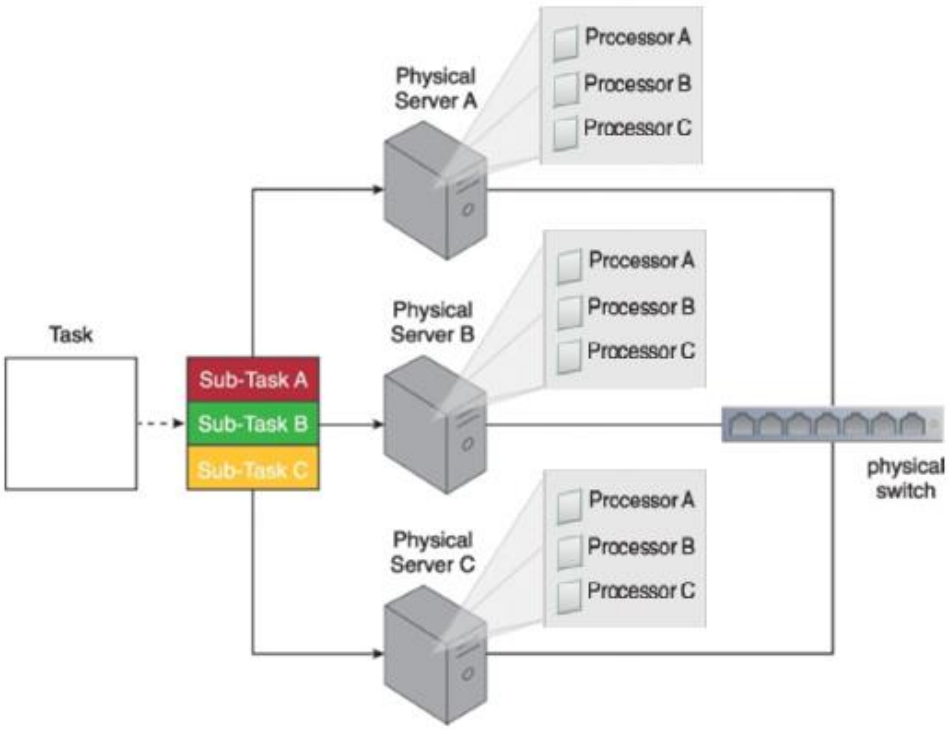
این سیستم را نیز گاهی loosely coupled هم می‌گویند. (loosely: شل، بی‌ربط) زیرا همه چیز در کامپیوترهای مجزا و جدا از هم قرار دارند.



سرعت پردازش در این دو سیستم قابل مقایسه نیست، و کاملا بستگی به کاربرد دارد.

اما امروزه توزیع شده متداول‌تر است. زیرا داده‌ها آنقدر زیاد هستند که در یک کامپیوتر جا نمی‌شوند. وقتی داده‌ها در سیستمهای متفاوت قرار بگیرند نحوه‌ی پرددازش‌ها نیز متفاوت می‌شود و به این ترتیب برنامه‌نویسی توزیع‌شده هم مرغوب‌تر می‌شود.

ترکیب این دو نوع را هم می‌توان بصورت زیر استفاده کرد.



در big data نیز این مورد کاربرد دارد. در شرکت گوگل یک ساختمانی وجود دارد که در آن Rackهایی قرار دارند که در آن Rackها هر کدام یک Multi-processor است. البته این را به یاد داشته باشید که در big data پردازش حرف اول را نمی‌زند، و انتقال بار هم مهم است.

نکته: یک Task هرچه به sub-taskهای بیشتری شکسته شود همیشه خوب نیست. بلکه تا یک حد خاص این کار مفید است. (شرط اول اینست که Task ما قابل شکستن باشد) (به دست آوردن این حد خاص امکانپذیر است اما در مقوله درسی ما نمی‌گنجد)

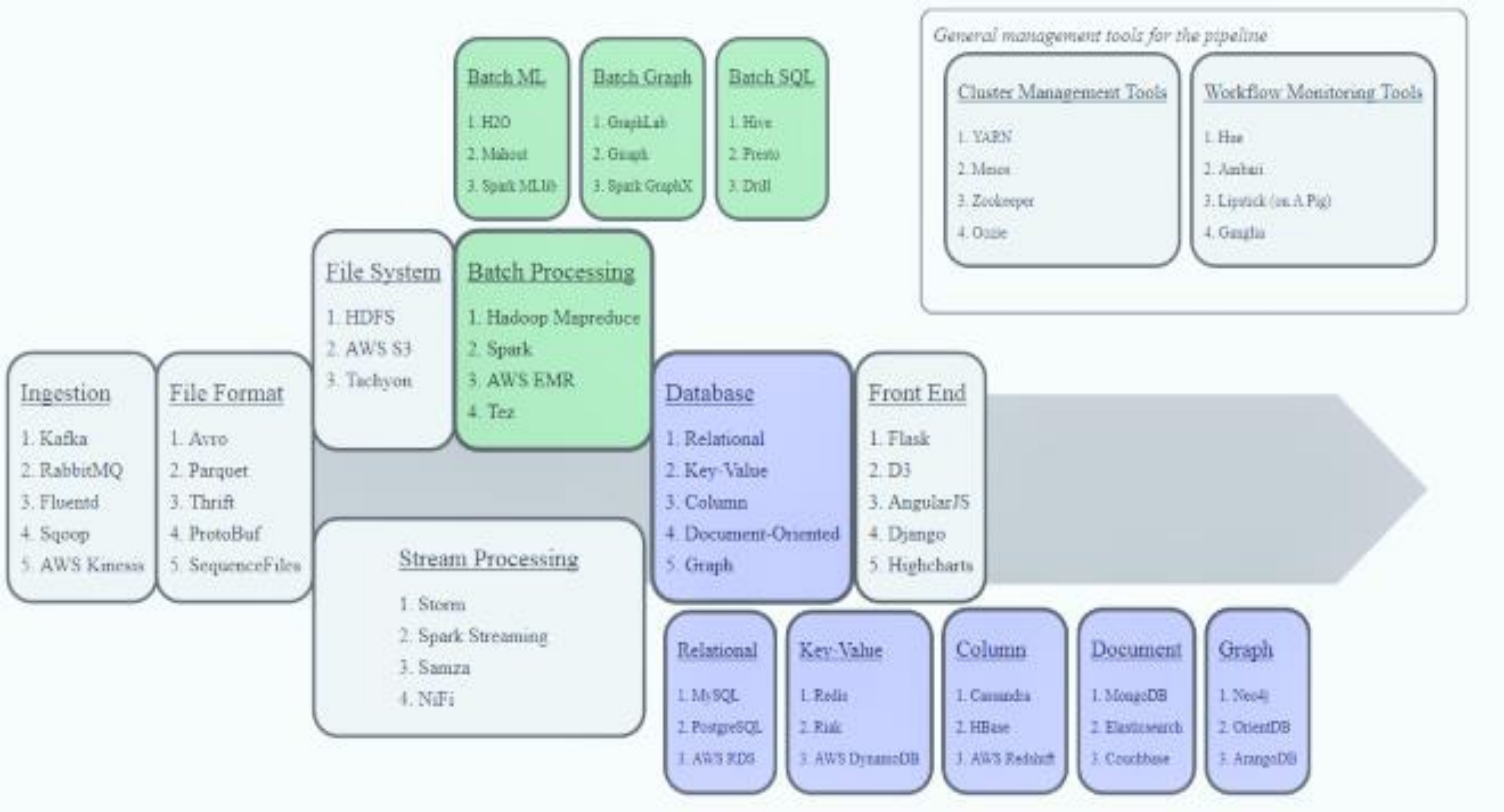
مثال: فرض کنید کاری هست که اگر استاد به تنهایی انجام دهد یک ساعت طول می‌کشد. می‌خواهیم آن کار را به دو نفر محول کنیم تا در زمان کمتری انجام شود. البته این ضمانتی ندارد که این کار با وجود دو نفر در ۳۰دقیقه انجام شود. مسئله مهم‌تر در این کار توزیع Taskها و در آخر جمع‌آوری آنهاست.

اگر این کار را به دو قسمت بشکنیم، آنها هرکدام ۳۲ دقیقه طول خواهند کشید که در مجموع میشود ۶۴دقیقه.(۴دقیقه بیشتر)

اگر این کار را به سه قسمت بشکنیم، آنها هرکدام ۲۵ دقیقه طول خواهند کشید که در مجموع میشود ۷۵دقیقه.(۱۵دقیقه بیشتر)

و به همین ترتیب این زمان زیاد می‌شود. تا اینکه به جایی می‌رسیم که دیگر این زمان زیاد شده به صرفه نیست، از آنجا به بعد شکستن کارها اصلا منطقی و صحیح نیست. {زیرا مثلا اگر استاد بخواهد کار یک ساعته‌اش را به عهده‌ی ۲۰نفر بگذارد، فقط باید ۱/۵ساعت وقت بگذارد که وظیفه‌ی هر شخص را برایش توضیح دهد. پس این اصلا به صرفه نیست.}

این تصویر ابزارها و نرم‌افزارهای آپاچی مفید برای big data را نام برده است، که برای کاربردهای خاص دسته بندی شده است.
(البته اینها تعداد این ابزارها بیشتر است که می‌توانید از منابع دیگر آنها را مطالعه کنید)



نکته: اغلب یا بهتر است بگوییم همه‌ی این نرم افزارها روی لینوکس نصب می‌شوند. که برای این کار اگر نمی‌خواهید سیستم‌عامل خود را تغییر دهید میتوانید از مجازی‌سازی و ماشینهای مجازی استفاده کنید.

نکته: می‌توانید برای انتخاب آگاهانه یکی از این ابزارها از مقایسه‌هایی که در بستر اینترنت وجود دارد استفاده کنید.

سایتهایی وجود دارند که در آنها هم فیلم آموزسی قرار دارد و هم مطالب زیادی در رابطه با این آپاچی‌ها. مانند:

http://myhadoop.ir/

http://hadoop.ir/

http://www.bigdata.ir/

https://bigdatauniversity.com/

برای گرفتن داده برای تست روی نرم افزارها از سایت زیر استفاده کنید:

Big data repository

سایتهای دانشگاههای معروف دنیا مثل هاروارد و استندفورد

در ادامه با تعدادی از ابزارهای هر یک از این دسته‌ها آشنا می‌شویم.

- **ابزار تزریق داده** : ابزار و فناوریهای که به کمک آنها می توان داده ها را وارد سامانه های کلان داده نمود. بوسیله‌ی این ابزارها میتوانید داده‌های حجیم را وارد سیستم کنید.

۱. Kafka

۲. RabbitMQ

۳. Fluentd

۴. Sqoop

۵. AWS Kinesis

- **قالب های فایلی ذخیره و بازیابی اطلاعات** : گاهی اوقات نیاز داریم بعضی اطلاعات را که در قالب های مختلف هستند درون یک فایل ذخیره کنیم. این ابزارها برای ذخیره و بازیابی قالبها و فرمتهای مختلف استفاده می‌شود.

۱. Avro

۲. Parquet

۳. Thrift

۴. ProtoBuf

۵. SequenceFiles

اگر در سازمانتان بخش‌های مختلفی دارید که در هر بخش داده‌ها را با یک متد خاص نگهداری می‌کنند، مثلا exel، word، table و... . برای کنارهم قرار دادن این داده‌ها از این ابزارها استفاده می‌کنند.

- **سیستم های فایلی ذخیره و بازیابی اطلاعات :**

۱. HDFS (Hadoop Distributed File System)

۲. AWS S3

۳. Tachyon

فایل سیستم چیست؟ روش های متفاوتی برای سازماندهی و ذخیره ی اطلاعات در هارد دیسک ها هستند.

فایل سیستم ویندوز NTFS است. شما می‌توانید تفاوت بین NTFS و HDFS را پیدا کنید و بگویید که چرا ویندوز با فایل سیستم NTFS نمیتواند داده های حجیم را نگه‌دارد و ما برای این کار نیاز به HDFS و هدوپ داریم؟

❖ **پردازش دسته ای یا Batch :**

۱. Hadoop Map/Reduce

۲. Spark

۳. AWS EMR

۴. TEZ

Map/Reduce یک روشی است برای اجرای سریعتر برنامه ها. مانند ضرب ماتریس‌های بزرگ، که با این روش انجام می‌شوند.
نکته: یکی از کاربردهای مهم در ضرب ماتریس‌ها پردازش تصویر است.

❖ **یادگیری ماشین:** الگوریتم‌های classification مانند Decision tree، Bays همه یادگیری ماشین هستند.

۱. H2O

۲. Mahout

۳. Spark MLib

سوال پیش می‌آید که آیا Rapid Miner روی داده‌های حجیم نیز پاسخگو است؟ که در جواب باید بگوییم خیر. بلکه ابزارهای بالا برای این کار مناسبند. شما می‌توانید با نصب این ابزارها و تست کردن یک سری داده در Rapid Miner و یکی از این ابزارها تفاوت در پاسخگویی آنها را مشاهده کنید.

❖ **پردازش گراف:**

۱. GraphLab

۲. Giraph

۳. Spark GraphX

امروزه خیلی داده‌ها و کسب و کارها به شکل گراف هستند که معمول‌ترین آنها هم شبکه‌های اجتماعی هستند و هر فرد گره‌ای از آن گراف است.

نکته: البته دقت داشته باشید که پردازش با تحلیل فرق دارد. مثلا می‌گوییم "در این گراف تلگرام چه کسی از همه دوستان بیشتری دارد؟" این می‌شود پردازش. اما اگر بگوییم "افرادی که دوستان زیادی دارند چه خصوصیاتی دارند؟" می‌شود تحلیل.

نکته: اگر توجه کنید Spark در چند قسمت دیده شد و شاید این نشان می‌دهد که این ابزار بسیار وسیع و پرکاربرد است، که هر قسمت آن برای مورد خاصی به کار می‌رود.

❖ **اجرای SQL:** ما در محیط‌های big data نیز میتوانیم SQL داشته باشیم. اما مسئله این است که برای پاسخگویی به عبارات SQL از چه ابزاری باید استفاده کنیم؟ سه تا از ابزارهای معروف برای این کار در زیر آورده شده است.

۱. Hive . نرم افزاری است که روی هدوپ نصب می‌شود تا شما توسط آن بتوانید پرسشها وکوئریهای sql را انجام دهید.

۲. Presto

۳. Drill

❖ **پردازش جریانهای داده:** مثلا ما چهار نرم افزار در اینجا معرفی کردیم که برای پردازش داده‌های stream هستند. داده‌های stream مانند تصاویری که از دوربینهای نظارتی یا دوربین پارکینگها ضبط می‌کنند. یعنی داده هایی که مدام در حال تولید هستند. حتی کامنتهایی که در شبکه‌های اجتماعی ردوبدل می‌شوند می‌توانند جزو داده‌های stream قرار گیرند.

۱. Storm

۲. Spark Streaming

۳. Samza

۴. NiFi

دردسر اصلی داده‌های stream اینست که نمی‌توان آنها را ابتدا ذخیره و سپس آنها را تحلیل کرد. بلکه باید این داده‌ها در آن واحد تحلیل شوند. همچنین آنقدر سرعت تولید این داده‌ها زیاد است که اصلا فرصت ذخیره همه را نداریم.

مثلا فکر کنید وقتی در یک پارکینگ دزدی می‌شود، نمی‌توانیم صبر کنیم تا فردا و تصاویر را بررسی کنیم.

در کف اقیانوسها سنسور وجود دارد. که این سنسورها میزان رطوبت، درجه و دیگر شرایط محیطی را مخابره می‌کنند. تعداد این سنسورها بسیار زیاد است. وقتی یک دفعه همه‌ی این سنسورها اطلاعات را مخابره می‌کنند آنقدر داده‌ها زیاد هستند که نمیتوان ابتدا آنها را جمع و سپس تحلیل کرد. پس باید همانطور که سرعت تولید این داده‌ها بالاست سرعت پردازش آنها نیز بالا باشد. که این نشاندهنده‌ی قدرت این ابزارهای تحلیل است.

❖ **بانکهای اطلاعاتی:**

۱. بانکهای اطلاعاتی رابطه ای

۲. بانکهای سندگرا

۳. بانکهای سطر گسترده

۴. بانکهای کلید مقدار

۵. بانکهای گراف محور

❖ **ابزارهای مدیریت شبکه و کلاستر:** این ابزارها مشخص می‌کنند که کپی‌ها، کلاسترها و ... در کجا قرار بگیرند.

۱. YARN

۲. Mesos

۳. Zookeeper

۴. Oozie

- **ابزارهای نظارت و مانیتورینگ:** همیشه نرم‌افزارهایی مثل سیستم عامل وقتی که اجرا می‌شوند در اجرا خود را با نظارت بهبود می‌دهند. مثلا فرض کنید شما ویندوز را نصب نمودید، در مدت ۱۰روز هر بار که میخواهید از آن استفاده کنید از همان ابتدا از نرم‌افزار paint استفاده می‌کنید، پس ویندوز متوجه میشود این برنامه برای شما پرکاربرد است. و با همین دید دفعه بعد قبل از اینکه شما سیستم را راه‌اندازی کنید ویندوز برنامه paint را در حافظه cash قرار می‌دهد تا به محض درخواست آن را سریعا اجرا کند. ویندوز این کار را با نظارت و مونیتورینگ اجراهای قبلی انجام می‌دهد.

زمانیکه می‌خواهیم در big data گزارش بگیریم را درنظر بگیرید. برای این گزارش‌گیری ما به همه‌ی داده‌ها نیاز نداریم. اگر با چندبار گزارش‌گیری متوجه بشوید که شما به این داده‌ها بیشتر نیاز دارید بهتر است این داده‌ها در جایی قرار بگیرند که هرزمان نیاز به گزارش‌گیری بود سریعا این داده‌ها را دم دست قرار دهد.

نظارت فقط برای خوب کار کردن نیست، بلکه برای رفع خرابی هم هست. مثلا به ما می‌گوید اگر یک گره محاسباتی خراب شد replication ان را کجا می‌توان پیدا و استفاده کرد.

۱. Hue

۲. Ambari

۳. Lipstick

۴. Ganglia

- **داشبوردهای تحلیلی و تولید برنامه های کاربر:** داشبورد یک سری گزارشاتی است که در جلسات قبل نیز در رابطه با آن صحبت کردیم و نمونه‌هایی از آن را دیدیم. و ابزاری برای تحلیل داده‌هاست.

۱. Flask

۲. D3

۳. AngularJS

۴. Django

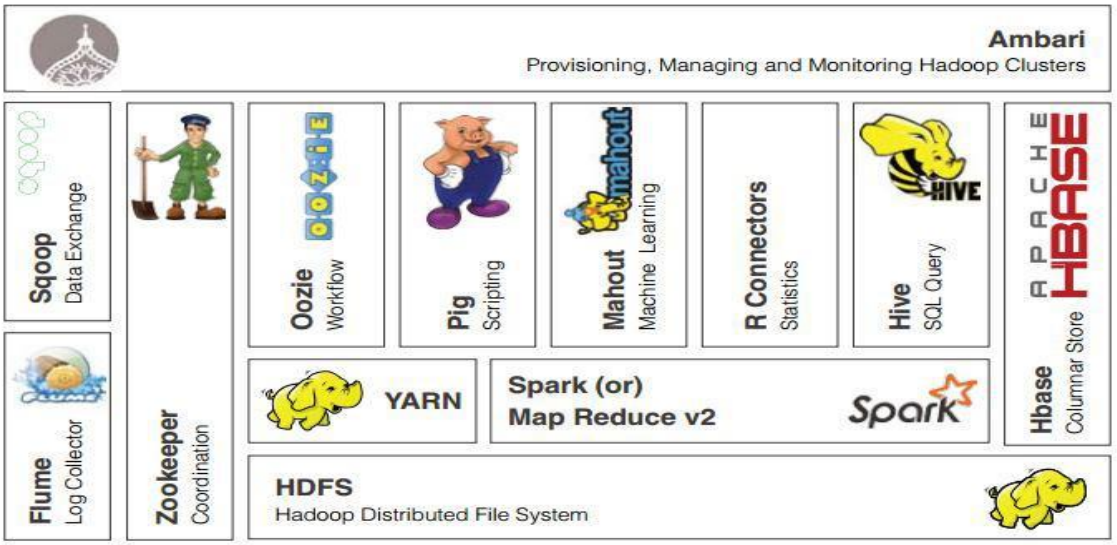
۵. Highcharts

Click view یک نمونه از این نرم افزارها در ویندوز است اما برای big data مناسب نیست، که می‌توان تفاوت آن را با ابزارهای big data بررسی کرد و ذکر کرد که چرا این نرم‌افزار برای big data مناسب نیست.

به طور خلاصه ، هدوپ یک framework یا مجموعه ای از نرم افزارها و کتابخانه هایی است که ساز و کار پردازش حجم عظیمی از داده های توزیع شده را فراهم میکند. در واقع Hadoopرا 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. فریم ورک هدوپ شامل زیر پروژه های مختلفی می‌شود که در زیر لیست کامل آنها آمده است:

- HDFS
- YARN
- Map/Reduce
- Ambari
- Avro
- Cassandra
- Chukwa
- HBase
- Hive
- Mahout
- Pig
- Spark
- Tez
- ZooKeeper

در سایت آپاچی همه‌ی این ابزارها وجود دارند و هر هفته هم آپدیت می‌شود.



#جلسه چهارم و پنجم، پنجشنبه ۶ و ۱۳۹۵/۸/۱۳

در ادامه بحث ما در رابطه با چالشها گفتیم که ذخیره، بازیابی، مدیریت و تحلیل بیگ دیتا، چالش می‌باشند.

بحث بیگ دیتا شامل مباحث ریز دیگری هم هست که ما در اینجا روی آنها صحبت نمی‌کنیم. مثلا یکی از فصل‌های درس بیگ دیتا رایانش ابری(cloud computing) است، که از زمان درس ما خارج است.

افرادی که می‌خواهند در حوزه بیگ دیتا کار کنند و پایان‌نامه یا هر پژوهش دیگری انجام دهند باید بدانند که حتما لازم نیست در آن حوزه‌ی پژوهشی عین عبارت "big data" وجود داشته باشد؛ بلکه حوزه‌هایی همچون موارد زیر نیز از زیرمجموعه‌های بیگ دیتا می‌باشند. که اتفاقا این مباحث در دنیا به‌روز و پرطرفدار می‌باشند:

رایانش ابری - الگوریتمی را بهبود دهید که بتواند با داده‌های بزرگ خوب کار کند - اینترنت اشیا (IOT) - کار کردن روی صداها و تصاویر دوربین‌ها {مثل: رسم درخت تصمیم برای فیلم‌های ضبط شده از دوربین‌های نظارتی} - شبکه‌های اجتماعی - عقیده کاوی(opinion mining) و ... .

اغلب ابزارهای بیگ دیتا روی ویندوز نصب نمی‌شوند و نیاز به پلتفرم خاصی که یکی از آنها هدوپ می‌باشد دارند. برای این کار باید ابتدا روی سیتم‌عامل لینوکس هدوپ را نصب کرده، سپس ابزار مربوطه را روی هدوپ نصب کنید.

میتوان به نحوی گفت هدوپ یک سیستم‌عامل برای بیگ دیتاست.

به جای اینکه بگوییم ویندوز بریز و بعد اکسل بریز؛ میگوییم هدوپ بریز و بعد mahout، Hive ، mongo db و ... بریز، و بعد کارهایی که می‌خواهی روی آن انجام بده.

ابزارهایی که در جلسه قبل گفتیم همه روی هدوپ نصب می‌شوند.

مشکل بزرگ ما اینست که ما روی سیستم‌هایمان ویندوز داریم. که برای حل این مشکل گفتیم از مجازی‌سازی استفاده می‌کنیم. به این صورت که ابتدا یک virtual machin(ماشین مجازی) مانند VMWare یا virtual box روی سیتم خود نصب کرده و سپس روی آن لینوکس، روی لینوکس هدوپ و روی هدوپ ابزار بیگ دیتای مورد نظر را نصب می‌کنیم.

virtual machin سیستم عامل را عوض نمی‌کند بلکه یک پوسته است که فقط روی سیستم عامل شما قرار می‌گیرد و توسط ایمیج‌های مورد نیاز یک محیط مجازی برای شما ایجاد می‌کند.

نکته: virtual box سبک‌تر از VMWare است، اما VMWare رایج‌تر است.

شما می‌توانید روی سیستمتان یک Vmware Workstation نصب کنید(نسخه Server آن برای نصب روی سرور است) این نسخه client است و به سیستم عامل وابسته است. یعنی باید روی سیستمتان یک سیستم عامل مانند ویندوز داشته باشید تا بتوانید Vmware Workstation را نصب کرده و روی آن کار کنید. Vmware منابع، هارد و Cpu کامپیوتر میزبان را تقسیم بندی کرده و به چند سیستم مختلفی که در محیطش ایجاد می‌کنیم اختصاص می‌دهد.

برای یادگیری می‌توانید روی این سیستم مجازی کار کنید و از طریق سعی و خطا کارها را یاد بگیرید. اگر هم تغییری انجام دادید که خرابی به بار آورد مشکلی پیش نمی‌آید و می‌توانید آن را حذف کرده و دوباره از اول نصب کنید.

ویژگی ای که اکثر نرم افزارهای لینوکسی دارند اینست که یک رابط وب دارند. یعنی می‌توان توسط یک Browser و وارد کردن Ip سیستم(که در اینجا یک سیستم مجازی است) به آن محیط وصل شد و دستور وارد کرد.

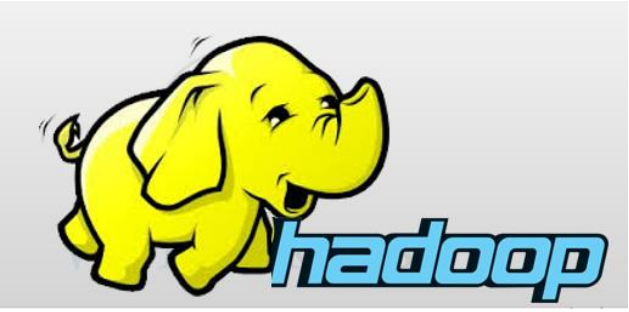
چند دستورات لینوکس:

Halt خاموش کردن سیستم

If Config آی پی پورتها را میدهد

Su پسورد Root را میگیرد و به شما بعنوان Root اجازه اموری مانند خاموش کردن سیستم را می‌دهد.

توصیه: فیلم‌های آموزشی برای معرفی این ابزارها، حتی نحوه نصب این نرم‌افزارها هم وجود دارد که می‌توانید از آنها استفاده کنید.



معرفی Hadoop

در ابتدا باید بگوییم که هدوپ رایگان نیست، اما سه شرکت توزیع‌کننده هدوپ برای امورآموزشی یا حتی بعنوان نمونه کار یک ایمیج تک پروسسوره(تک نود) از هدوپ را برای قرارگیری روی VMWare ارائه نموده‌اند. یعنی می‌توانید با دانلود این ایمیج و open کردن آن روی VMWare، بطور خودکار محیط لینوکس، بستر هدوپ و حتی تعدادی از ابزارهای بیگ دیتا را هم داشته باشید.

نکته: این نسخه چون دارای یک نود بیشتر نیست، نمی‌توان آن را برای امور تجاری به کار برد. چرا که ما زمانی نیاز به این ابزارها داریم که بیگ دیتا داشته باشیم و بیگ دیتا هم اصولا روی نودهای مختلف توزیع شده است.

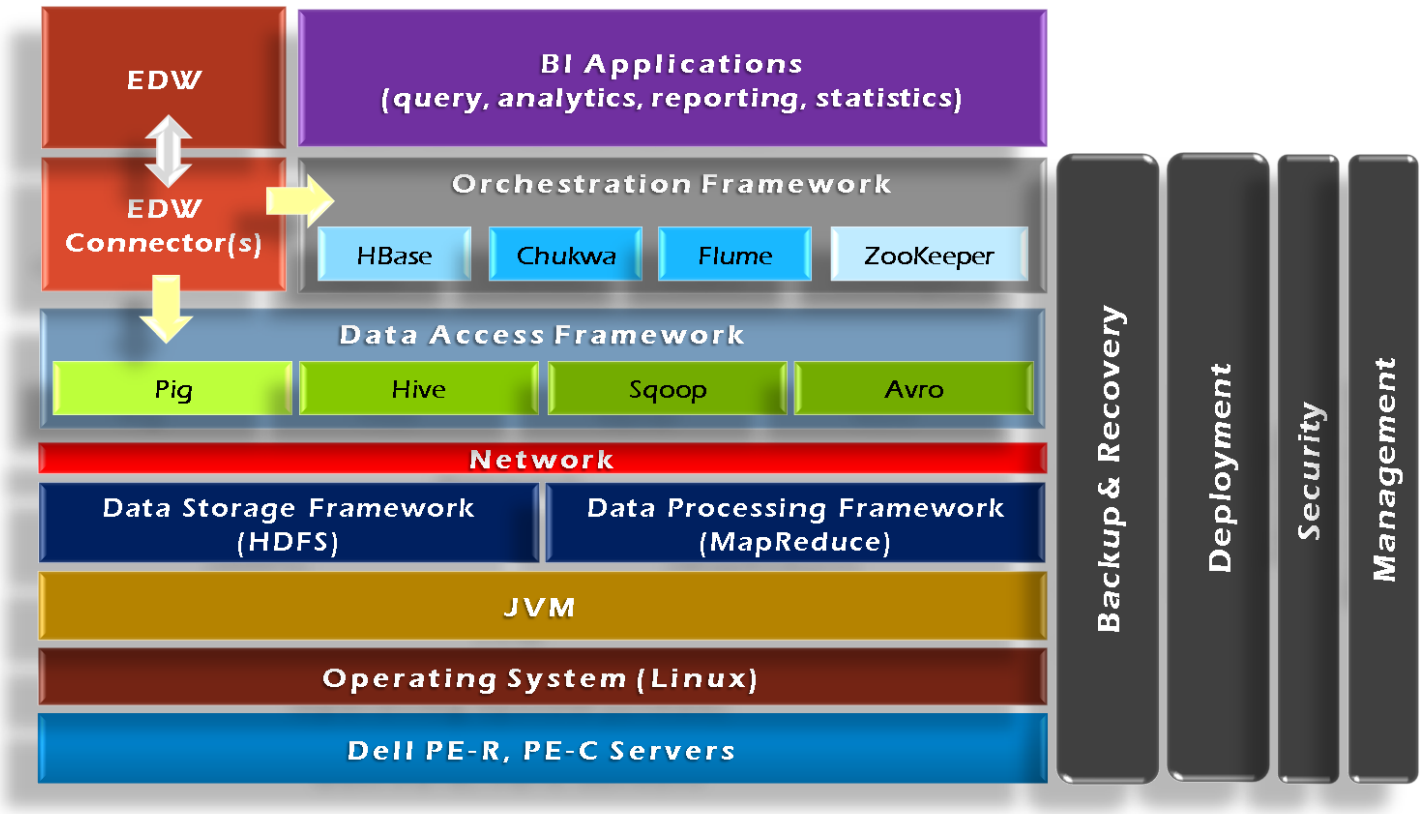
نکته: اگر ابزار مورد نظر شما روی این ایمیج قرار نداشت، میتوانید ایمیج آن ابزار را روی این ایمیج هدوپ نصب نمایید.

سه شرکت توزیع کننده هدوپ:

- Hortonworks (با حجم حدود ۱۱گیگ)
- Cloudera (۴/۵ گیگ)
- MAPR (۳/۵ گیگ)

شکل زیر معماری هدوپ را نشان می‌دهد.

در قسمت پایین سخت افزار ما را نشان می‌دهد. بعد از آن سیستم عامل لینوکس داریم، بعد از آن java machine داریم. چون هدوپ به زبان جاوا نوشته شده است برای اجرا نیاز به یک موتور جاوا دارد. در هدوپ ما دو بحث HDFS و Map-Reduce را داریم. HDFS سیتم توزیع فایل. کلا یک کامپیوتر دو نوع کار بیشتر انجام نمی‌دهد. یا داده ذخیره می‌کند، یا داده‌ها را پردازش می‌کند. هدوپ هم همین دو کار را انجام می‌دهد. توسط HDFS داده‌ها را ذخیره می‌کند و توسط Map-Reduce آنها را پردازش می‌کند. اما تفاوت آن(هدوپ) با سیستم‌های معمولی در این است که هدوپ این وظایف را بصورت توزیع‌شده انجام می‌دهد. یعنی ما یک سیستم Master به نام Name Node و یک سری سیستم‌های کاربر به نام Data Node داریم. که Name Node این پردازش‌ها را بین Data Nodeها تقسیم می‌کند.



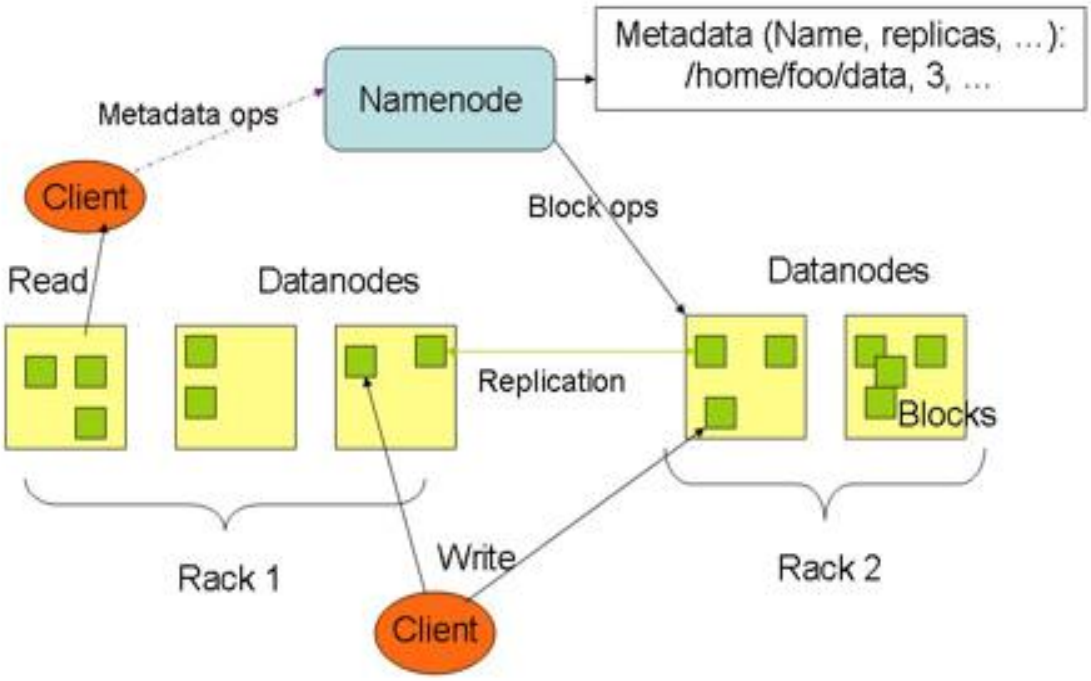
HDFS در واقع مدیریت ذخیره‌سازی فایلها را انجام می‌دهد. یعنی همان کارهایی که NTFS برای ویندوز انجام می‌دهد(کدام فایل در کدام قسمت حافظه قرار بگیرد و...) HDFS آن را برای هدوپ انجام می‌دهد. البته نکته ای که وجود دارد اینست که هدوپ برای ذخیره‌سازی فایلها و داده‌ها، آنها را تکه‌تکه می‌کند و ظرفیت هر تکه را از قبل مشخص کرده است. مدیریت این تکه‌ها برعهده‌ی Name Node است.

Map-Reduce نوعی پردازش است که از دو جزء Map(نگاشت) و Reduce(کاهش) تشکیل شده است. مثلا اگر بخواهیم ۵۰تا عدد را باهم جمع کنیم Map-Reduce چگونه این کار را انجام می‌دهد.

وقتی شما هدوپ را نصب می‌کنید در واقع تا لایه چهارم این معماری انجام شده است در همه لایه‌ها ما بکاپ وامنیت و ... را داریم که به موازات همه در شکل آورده شده است. بعد از آن یک سری patchهای نرم‌افزاری داریم که شامل Sqoop و ... است. که مثلا Sqoop برای بارگذاری داده‌ها روی Hive است. لایه‌ی network می‌تواند باشد یا نباشد و همه‌ی اینها روی یک سیستم باشند. DW(Data Warehouse) به معنای مجموعه‌ای کل داده‌های سازمان است(انبار داده‌های سازمانی). داده‌هایی که در یک سازمان هستند وبا هم در یک جا جمع شده‌اند. DW انبار داده‌هاست و EDW واسط ارتباطی بین این همه داده با تحلیلگرهاست.

داده‌های ما آنقدر حجیم هستندکه وقتی داریم داده‌ها را توسط چیزی مانند دوربین capture می‌کنیم برای ذخیره آنها به نرم‌افزاری نیاز داریم. چون اصلا گاهی فرصت ذخیره کردن نداریم. حتی گاهی این داده‌ها را sampling می‌کنیم. یعنی نمونه هایی از آنها را ذخیره می‌کنیم.

آیکون هدوپ یک فیل زرد است. شخصی که این پیشنهاد داد بچه‌اش با یک فیل زرد بازی می‌کرد و اسم آن فیل هدوپ بود.



همانطور که در شکل مشاهده می‌نمایید ما در هدوپ یک Namenode(نود مستر یا مدیر) و یک سری Datanode داریم. قاعدتا روی همه‌ی نودهای ما هدوپ نصب است. اگر ما کل این شکل را یک کلاستر در نظر بگیریم، Namenode مدیریت کلاستر را برعهده دارد و Datanode مسئول ذخیره سازی داده هاست.

نکته: الزاما ما یک Namenode نداریم.

در بحث بیگ دیتا چون حجم داده‌ها بالاست بنابراین هر نود را برای ذخیره‌ی یک سری داده خاص در نظر می‌گیریم، که این توزیع داده روی نودها توسط نود مستر صورت می‌گیرد. و هر Datanode مسئول ذخیره سازی و پردازش داده های خودش است.

در شکل بالا دو Rack داریم که در هر Rack سه کامپیوتر است، جمعا می‌شود ۶ کامپیوتر؛ که در یک شبکه پر سرعت توسط کابلها به هم متصل شده اند. ۵تا را برای ضبط اطلاعات تخصیص داده اند و یکی را برای مدیریت. در ابتدا خودمان مشخص می‌کنیم که کدام نود مدیر باشد همچنین در ادامه نمی‌توان بین کار نود مدیر را عوض کرد.

Namenode یک ویژگی به نام پالس(ضربان) دارد ، و هر چند ثانیه یک بار از Datanodeها می پرسد چه خبر؟ هستید؟

در قسمت کنسول نیز ما یک قسمت به نام آلارم داریم. که هرگاه یک نود خراب شد و بعد از یک زمان مشخص از کار افتاد آلارم میدهد که دیگر وظیفه ای به او محول نشود، دیگر چیزی آنجا ضبط نکند یا اگر تکراری در آنجا دارد در جاهای دیگر هم داشته باشد.

توجه: البته آلارم لایه جلویی است پشت صحنه پالس است.

سوالهایی که در رابطه با این شکل پیش می‌آید می‌تواند بصورت زیر باشد:

در هدوپ آیا میتوان Namenode را در حین استفاده عوض کرد؟

Fault Tolerance. اگر Namenode ما خراب شود آیا ما دیگر به دیتانودها دسترسی نخواهیم داشت؟

اینها عیب این مسئله هستند. هدوپ برای این مشکلات چه چاره‌ای اندیشیده است؟

Namenode چه کار میکند که اگر خراب شود برای ما دردسر دارد؟

آیا اگر ما ۱۰۰تا Rack داشته باشیم، یک Namenode داریم؟

اگر چندتا Namenode داریم، آیا این Namenodeها هم خودشان یک مستر دارند؟

در معماری هدوپ ما یک Network داریم. این لایه در واقع واسط Namenode و Datanodeها هستند. در واقع ما در اینجا ابزارهایی را که داریم موتورهای اصلی شان روی Namenodeها قرار می‌گیرد و درنتیجه ابزار ما Datanodeها را نمی‌بینند و در واقع آن سیستم توزیع شده فایل مدیریت بین داده ها را انجام می‌دهد. Namenode دستورات SQL را برای اجرا به Driver ارسال می‌کند. هر دستور به سمت نود مربوطه می‌رود و دستور انجام می‌شود، سپس جوابها تجمیع شده و به موتور Hive برمی‌گردد.

در روی Namenode یک نرم افزار مدیریت باید نصب کنیم.

خود هدوپ رایگان است، ولی نرم افزارهایی که برای مدیریت آن به کار می‌رود پولی است. این شرکتها این فایل مدیریت(Cloudera Manager) را به صورت دو نسخه ارائه می‌کنند. نسخه Enterprise و Express. نسخه Express رایگان است. و یک سری قابلیتها را ندارد.

هدوپ داده‌ها را که می‌گیرد بصورت پیش‌فرض داده‌ها را تکه تکه و هر تکه را در سه جا کپی می‌کند و موقع دستیابی به داده‌ها باز با همان مدیریت هدوپ به داده‌ها دسترسی پیدا می‌کنیم.(البته می‌توانیم تعداد و محل قرارگیری این تکه‌ها را خودمان نیز مشخص کنیم)

معرفی آپاچی Hive

این ابزار یک Data Warehouse است که کار ذخیره‌سازی داده‌ها را انجام می‌دهد.

همانطور که می‌دانید Hive نیاز دارد روی بستر هدوپ نصب شود.

Hive به معنای کندو است. یک سیستمی برای مدیریت پرس و جوهای داده‌های غیرساختاریافته که ساختاریافته شده‌اند. پس می‌توان گفت Hive یک سیستمی است که داده‌های ساختار یافته را ذخیره می‌کند. در Hive ما پردازش و ذخیره سازی داریم و همانطور که برای هدوپ ذکر شد، توسط HDFS داده‌ها ذخیره می‌شود و توسط Map-Reduce داده‌ها پردازش می‌شود.

Hive اولین بار در فیسبوک ارائه شد. در مواردی ذکر شده که Hive یک دیتابیس است؛ در حالیکه این ادعا اشتباه است، بلکه Hive یک DW است. مثلا ما در دانشگاه دیتابیس ثبت نام دانشجویی داریم، یک دیتابیس رزرو غذا داریم و...؛ همه‌ی اینها که با هم جمع می‌شوند می‌شود یک DW. از این DWها استفاده تراکنشی نمی‌کنند و فقط برای تحلیل داده‌ها از آن استفاده می‌کنند.

نکته: داده‌ها برای انتقال به DW تمیز و تجمیع می‌شوند. در DW اطلاعات جزئی داده‌های ما نگهداری نمی‌شود و فقط خلاصه‌ای از داده‌ها قرار می‌گیرد. مثلا نمره هر درس در DW قرار نمی‌گیرد بلکه فقط میانگین نمره کلاس در آن قرار می‌گیرد. بعبارتی در DW داده‌های مدیریتی قرار می‌گیرند (برگردید به جلسه دوم. تفاوتهای OLAP و OLTP).

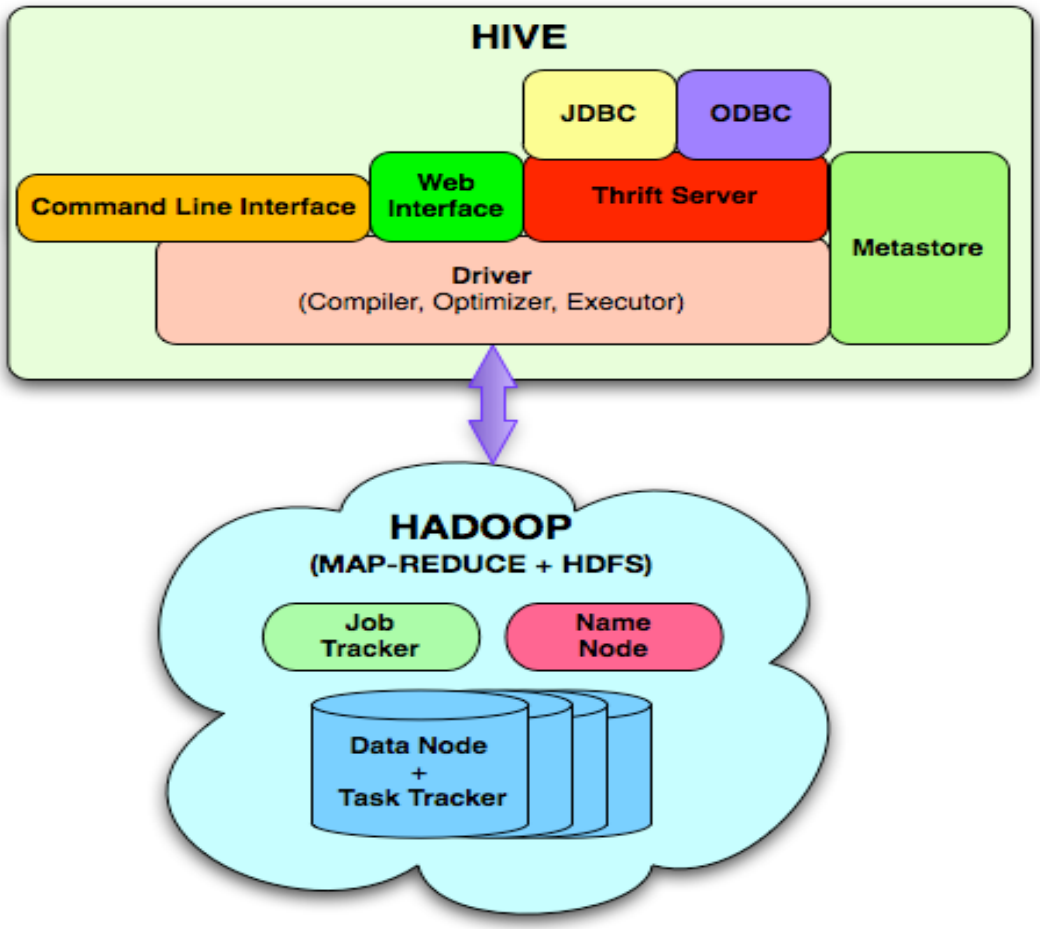
و Hive کنترل این DW را برعهده دارد، نه کنترل دیتابیس‌ها را.

حال سوال پیش می‌آید که این داده‌ها با چه مدلی ذخیره شده‌اند؟ دانستن این مدل در انتخاب ابزار مناسب برای کار روی داده‌ها به ما کمک می‌کند. دیتابیس‌ها اصولا از نوع مدل رابطه‌ای و... باشند اما اغلب مدل DW از نوع Multi dimensional Model (مدل چند بعدی) است و Hive در خواندن این نوع مدل قهار است. از همه مهم‌تر چون شما این داده‌ها را از سازمانهای مختلفی جمع‌آوری کرده‌اید حتما حجم این داده‌ها حتی در صورت تجمیع شدن کم نیست. پس می‌توان گفت که Hive یک زیرساختار برای DW است.

زبان اصلی خود Hive، HQL است ولی از دستورات SQL نیز پشتیبانی می‌کند. ضمن اینکه زبان HQL در زمان کامپایل شدن به jobهای Map-Reduce توزیع می‌شود. این jobها توسط سیستم‌های توزیع شده اجرا می‌شوند. یعنی در واقع وقتی ما یک select می‌نویسیم چندتا سیستم توزیع شده این select را برای ما اجرا می‌کنند. زیرا داده‌های ما در چند سیستم توزیع شده است.(در واقع دستوری که شما اجرا می‌کند خود Hive سریعا آنرا به چند تکه می‌شکند تا بتواند آنها را موازی انجام دهد و سریع

آن را روی بیگ دیتا جمع کند چون اگر بخواهد این کار را یک جا انجام دهد چون داده‌ها حجیم هستند زمان انجام کار فوق‌العاده بالا می‌رود.)

در شکل زیر ما جایگاه هدوپ نسبت به Hive مشخص شده است.



در معماری Hive اجزاء زیر وجود دارد:

- ✓ **Metastore** . اطلاعات ما مثل کاتالوگ سیستم، اطلاعات راجع به جدولها، پارتیشنها و ... را ذخیره می‌کند. و درواقع ذخیره‌سازی آن مانند پایگاه داده رابطه‌ای رایج است.(Metastore محل نگهداری Metadata است)

Metadata یک سری توضیحات در رابطه با داده‌هاست. هر چه داده‌ها حجیم‌تر می‌شود نقش Metadata پررنگ‌تر می‌شود. (فرض کنید می‌خواهید در یک روستا حرکت کنید، کافیست dataی شما جاده‌ها و خیابانهای روستا و مکان اتصال آنها به هم باشد و اصلا به Metadata نیاز ندارید. اما اگر بخواهید در یک شهر بزرگ سیر کنید، غیر از اطلاعات جاده‌ها به اینکه بدانید از کدام مسیر زودتر می‌رسید، در کجا رستوران بیشتری وجود دارد و... نیاز دارید؛ اینها می‌شوند Metadataها.) پس هر چه داده‌ها بیشتر می‌شوند شما به داده‌های کناری که توضیحی در رابطه با داده‌های اصلی هستند بیشتر نیاز دارید.

مثال: اگر در یک شبکه اجتماعی‌ای که میلیونها نفر کامنت گذاشته‌اند بخواهید بدانید کدامیک در رابطه با محیط زیست نظر داده‌اند، اگر بخواهید یکی یکی واژه‌ی محیط زیست را چک کنید دردسر دارید. اما اگر در جایی ثبت کرده باشیم که مثلا "افرادی که در منطقه x زندگی می‌کنند اغلب در رابطه با محیط زیست نظر می‌دهند"(این می‌شود Metadata) سپس به راحتی می‌توانید از بین آنها این جستجو را انجام دهید.

مثال: فرض کنید می‌خواهید مخاطبین دو سیم‌کارت را در یک گوشی ادغام کنید. دردسری که ممکن است وجود داشته باشد اینست که ممکن است یک شماره تلفن با دو نام متفاوت در سیم‌کارتها ذخیره شده باشد یا حتی یک فرد دو شماره متفاوت داشته باشد که در دو سیم‌کارت نیز با نامهای متفاوت ذخیره شده است. حال سوال اینجاست که ما چگونه می‌توانیم با قرار دادن این مخاطبین کنار هم کاری کنیم که گوشی به این تشخیص‌ها برسد؟ جواب Metadata است. یعنی مثلا کنار اسم دکتر اسماعیلی ذخیره شود "استاد داده‌کاوی"، کنار اسم م اسماعیلی هم نوشته شود "استاد داده‌کاوی". این می‌شود Metadata. و سیستم طبق آن این دو شماره را در یک محل قرار می‌دهد.

پس داده‌ها نیز برای قرار گرفتن در DW برای تمیز شدن از Metadata استفاده می‌کنند. مثلا اگر جایی تاریخ تولد فردی را خواستیم که نداشتیم، مشکلی وجود ندارد چون در جایی بعنوان Metadata یادداشت کرده‌ایم "۸۰٪ آدمها متولد فروردین هستند".

مثلا اگر مدیر بپرسد چند نفر نمره زیر ۱۲ داریم؟ چون در Metadata نوشته‌ایم "کمترین نمره ۱۵، بیشترین نمره ۱۹" پس بدون انجام هیچگونه جستجویی به سوال پاسخ می‌دهیم هیچی.

نکته: اما باید دقت داشته باشیم که آنقدر Metadata قرار ندهیم که بخواهیم داخل آنها دنبال مطلب بگردیم یا اینکه حجم آن از اصل dataهای ما بیشتر شود.

در پروژه‌های هوش تجاری مثل هوشمندسازی کسب و کار Metadata نقش زیادی را ایفا می‌کند.

حتی سیستم‌عامل ویندوز هم Metadata نگه داری می‌کند. مثلا در مدت ۱۰روز هر بار که میخواهید از آن استفاده کنید کارهای شما را ثبت می‌کند سپس متوجه می‌شود یک برنامه خاص برای شما پرکاربرد است. و با همین دید دفعه بعد قبل از اینکه شما سیستم را راه‌اندازی کنید ویندوز برنامه مورد نظر را در حافظه cash قرار می‌دهد تا به محض درخواست آن را سریعا اجرا کند.

- ✓ **Driver** . این قسمت دستورات اعمال شده توسط اینترفیس‌ها را کامپایل، اجرا و به هدوپ انتقال می‌دهد تا تبدیل به jobهای Map-Reduce شوند. در واقع اجرای دستورات HQL توسط این قسمت انجام می‌شود.
- ✓ **ODBC** (Open Database Connectivity) . وقتی که با بیگ دیتا کار می‌کنید ممکن است مدلهای داده‌هایتان با هم متفاوت باشد. وقتی می‌خواهید آنها را کنار هم جمع کنید به یک چیزی بعنوان میان‌افزار(middleware) نیاز دارید. یکی از میان‌افزارها پروتکل ODBC است.
- ✓ **JDBC** (Java Database Connectivity) . این پروتکل هم مانند ODBC یک میان‌افزار است که داده‌ها را به زبان Java تفسیر می‌کند و چون در اینجا ابزار ما مبتنی بر جاوا است از این پروتکل هم استفاده می‌شود.

نکته: این پروتکلها مانند ODBC file یک بسته است که در آن برای سیستم عاملهای مختلف فایل ODBC مخصوص وجود دارد و ما می‌توانیم متناسب با نیازمان آن فایل را نصب کنیم.

- ✓ **Web Interface** . برای دسترسی پیدا کردن به مخازن داده‌ها از رابط وب یا Web Interface استفاده می‌شود.
- ✓ **Command Line Interface** . مانند Web Interface یک رابط برقراری ارتباط با داده‌هاست.
- ✓ **Thrift Server** . اگر بخواهیم توسط یک نرم افزار جانبی مانند Click view یا خود نرم افزارهایی که در بحث هدوپ کارهای data mining انجام می‌دهند داده‌ها را استفاده کنیم، باید این داده‌ها را توسط دو پروتکل بالا و این سرور بخوانیم.

نکته: البته ما در سطح عملیاتی پروتکلها را می‌بینیم ولی سرور را نمی‌بینیم.

متغیرها

ما در اینجا دو نوع متغیر داریم. متغیرهای اصلی و متغیرهای پیچیده. که هر کدام به شرح زیر است.

متغیرهای اصلی:

Type	Comments
Tinyint, smallint, int, bigint	1, 2, 4 and 8-byte integers
Boolean	TRUE/FALSE
Float, double	Single and double precision real numbers
String	Character string
Timestamp	Unix-epoch offset *or* datetime string
DECIMAL	Arbitrary-precision decimal
BINARY	Opaque; ignore these bytes

متغیرهای پیچیده:

Type	Comments
Struct	A collection of elements If S is of type STRUCT {a INT, b INT}: S.a returns element a
MAP	Key-value tuple

	If M is a map from 'group' to GID: M['group'] returns value of GID
Array	Indexed list If A is an array of elements ['a','b','c']: A[0] returns 'a'

یک فیلد ما میتواند از نوع استراکچر، آرایه یا MAP باشد.

نکته: انعطاف‌پذیری نوع متغیرهای پیچیده خیلی بیشتر از ساده است. چون داده‌های ما از جنسهای مختلف هستند و ما نیاز داریم داده از جنس‌های مختلفی را روی Hive ثبت کنیم.

مدل‌های داده‌ای Hive

ما سه ساختار روی Hive داریم:

- Tables . همان چیزی است که ما در SQL داریم،که شامل سطر و ستون می‌باشد. اما نکته قابل توجه اینست که در Hive هر جدول یک شاخه در دیسک دارد. همچنین سیستم HDFS برای هر جدول یک فولدر با نام همان جدول در دیسک ایجاد می‌کند.
- Partitions . هر Table می‌تواند شکسته شود به یک یا چند پارتیشن (یک جدول را تکه‌تکه می‌کنیم می‌شود پارتیشن).

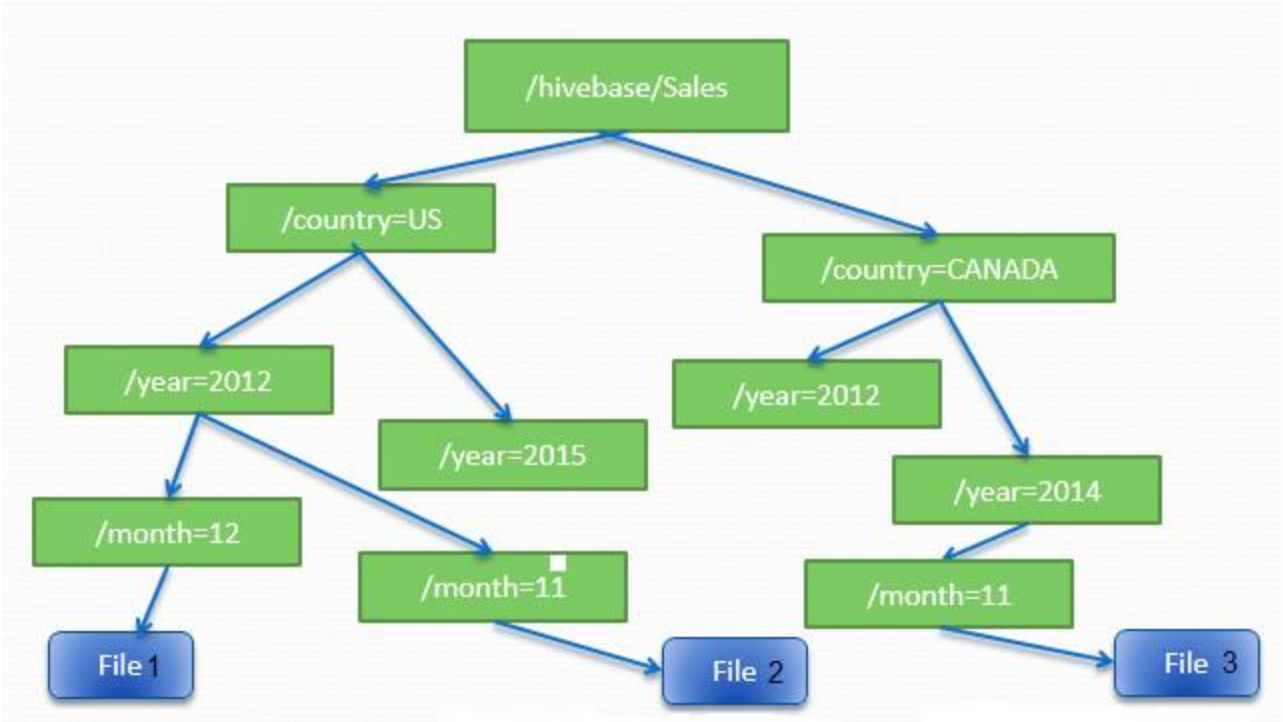
برای مثال در دستور select ساده‌ی زیر در ابتدا فقط دو فیلد sale_id و amount را داریم.

CREATE_TABLE Sales (**sale_id** INT, **amount** FLOAT)

اما در واقع این جدول پنج فیلد دارد(فیلدهای قبل + country، year و month)، که این فیلدها بعد از پارتیشن‌بندی جدول بوجود آمده‌اند.

PARTITIONED BY (**country** STRING, **year** INT, **month** INT)

در شکل زیر این سلسله مراتب مشاهده می‌شود.



همانطور که گفتیم به ازای هر table یک فولدر وجود دارد، پس در اینجا یک فولدر به نام Sales داریم. وقتی آن را پارتیشن‌بندی کنیم در فولدر اولیه دو فولدر برای دو مقدار موجود در فیلد country (US، CANADA) ایجاد می‌شود (پارتیشن‌بندی سطح اول). پارتیشن سطح دوم ما سالها هستند(year) . و سطح سوم ماه‌ها(month) . و به همین ترتیب برای هر سطح پارتیشن‌بندی به تعداد مقادیر متفاوت موجود در آن فیلد یک فولدر ایجاد می‌شود. و در نهایت در آخرین سطح فولدرهای ما فایلها قرار می‌گیرند. پس در پارتیشن ما در نهایت به یک فایل می‌رسیم.

برای مثال تمام فایلهای موجود در قسمت file1 دارای مقادیر فیلدهای country=US، year=2012 و month=12 هستند.

Sales/country=US/year=2012/month=12

پس مثلا اگر جدول ما ۱۰ رکورد داشته باشد هر رکورد در فایل متناظر به خودش قرار می‌گیرد.

اغلب پارتیشن انجام می‌شود بر روی فیلدهایی که کوئری در آن زیاد است. در اینجا به عبارتی ما یک ایندکسینگ انجام می‌دهیم. و کوئری را هدایت می‌کنیم. که البته این کار یک دردسر دارد. چون اگر تعداد رکوردهای مربوط به فایلها برابر یا نزدیک به هم نباشد این توزیع بار یکنواخت نیست.(البته اغلب این اتفاق نمی‌افتد)

این کار یک نوع فشرده‌سازی هم هست، و از افزونگی جلوگیری می‌کند. یعنی هم جستجو را راحت‌تر میکند و هم ذخیره‌سازی را.

- Buckets

Bucket به معنای سطل یا دلو است. پرتیشنها می‌توانند به باکتهای مختلف تقسیم شوند. و هر باکت میشود یک فایل.

در شکل مثال قبل به جای اینکه رکوردها در یک فایل ذخیره شوند در ۳فایل ذخیره می‌شوند.

سریع‌ترین راه دستیابی به داده‌ها در کامپیوتر چه روشی است؟ ساختاری داریم به نام direct یا مستقیم(hashing function)

برای مثال فرض کنید در یک کلاس صندلی‌ها شماره‌گذاری شده هستند. چطور به دانشجویان شماره صندلی ها اختصاص دهیم که وقتی دنبال شخص خاصی بودیم سریع بتوانم او را طبق شماره صندلی پیدا کنیم؟ برای هر نفر طبق یک سری محاسبات روی مشخصات او (تابع hash) برایش یک شماره صندلی قرار می‌دهیم. این تابع به نحوی عمل می‌کند که این شماره‌ها با هم تصادم نداشته باشند. به این صندلی ها می‌گویند Bucket.

حال هر دانشجو را بعنوان یک رکورد در نظر بگیرید. فرض کنید ما در اینجا چهار Bucket داریم، در هر Bucket تعدادی رکورد جا می‌شود. سپس با ورود هر رکورد با توجه به بخشی از داده‌ها و ضرب و تقسیم داده‌ها Bucket مناسب برای آن رکورد یافت می‌شود. به صورتی که وقتی خواستیم آن رکورد را پیدا کنیم همان عملیات را انجام می‌دهیم و محل آن رکورد را پیدا می‌کنیم.

بنابراین اغلب از تابع hash برای دسترسی سریع استفاده می‌شود. در آخر اغلب توابع hashing یک mod وجود دارد. تا اگر عدد حاصل از محاسبات تابع hash در محدوده‌ی شماره Bucketهای ما نبود با mod گرفتن، آن را به بازه مورد نظر تبدیل کنیم.

H(column) mod NumBuckets = bucket number

البته باید توجه داشت که تابع hashingای خوب است که همه‌ی رکوردها را داخل یک Bucket هدایت نکند. و رکوردهای متفاوت را در Bucketهای متفاوت قرار دهد.

نکته: توابع hash بسیار مفیدند و در بیگ دیتا نیز زیاد استفاده می‌شوند.

من وارد سایت شدم سیستم میخواهد بداند من به کدام شبیه هستم تا چیزهایی که آنها سفارش داده اند به من پیشنهاد بدهد. این کار را میتواند توسط تابع Hash و تقسیم بندی کل مشتری ها به یک تعداد محدود باکت این شباهت را پیدا کرده و مشابه ها را پیدا کرده و پیشنهاد خود را انجام دهد.

Hive Query Language

- JOIN

در مثال روبرو ما یک select و یک join(کاملا شبیه SQL) داریم؛ تنها نکته در اینجا اینست که ما در HQL همه‌ی انواع join را نداریم.

```
SELECT t1.a1 as c1, t2.b1 as c2
FROM t1 JOIN t2 ON (t1.a2 = t2.b2);
```

- INSERTION

```
INSERT OVERWRITE TABLE t1
SELECT * FROM t2;
```

نکته: Hive هم اطلاعات را ذخیره میکند و هم بازیابی.

سایتهای توزیع هدوپ مثل کلودرا یک فایل Zip به شما می‌دهد که تماما فایلهای یک کامپیوتر مجازی است. و با open کردن بصورت خودکار اجرا می‌شود. حتی رابطهایی مانند Hue نیز بصورت خودکار در آن وجود دارد.

توجه: هدوپ کلودرا رایج‌تر است چون هم patchهای نرم افزاری بیشتری دارد و هم مدیریت آن امکانپذیرتر است.

Cloudera Manager در این سیستم مجازی نصب نیست و برای مدیریت هدوپ باید آن را نصب کنیم.

نکته: یکی از راههایی که می‌توانید دیگر نرم افزارها را نصب کنید همین Cloudera Manager است.

ولی یک مشکلی وجود دارد Cloudera Manager خیلی سنگین است. و نمی‌توان با وجود آن ابزار مورد نظرمان را اجرا کنیم. پس به همین علت ما Cloudera Manager را روی یک سیستم مجازی و ابزارمان را روی یک سیستم مجازی دیگر نصب کنیم. و موقع استفاده سیستمی که Cloudera Manager روی آنست را خاموش کرده و از سیستم دوم استفاده کنید. (محیط Cloudera Manager برای مدیریت است نه کار کردن با نرم افزارها)

در این توزیع‌ها هدوپ برای یک نود ارائه شده و نمی‌توان استفاده عملیاتی از آن کرد.

نکته: حالا که یک نود داریم این نود هم مستر است و هم Datanode. پس Namenode میتواند Datanode هم باشد. البته در کاربرد اینگونه نیست. ولی برای یک نود به این صورت است.

سیستم که بالا می‌آید همان ابتدا بطور خودکار browser را باز کرده و رابط Hue را اجرا می‌کند. Hue یک نرم افزار تحت وب و یک واسط برای مونیتورینگ است مانند Cloudera Manager. در واقع در اینجا Hue یکی از سه اینترفیس ما است.

پورتی که Hue کار می‌کند 8888 است. در محیط Hue نرم افزارهایی که نصب هستند لیست شده اند که میتوان از آنها استفاده کرد. حتی مثالهایی هم در این محیط قرار دارد.

می‌توان گفت Hive یک DBMS و Hue یک User Interface است.

می‌توان از جاهای مختلف فایل Csv به داخل Import Hive کرده و از آن بعنوان داده‌هایمان استفاده کنیم.

آیا می‌توان با مثلا ویژوال C یک منو تهیه کرد و توسط Hive یک کوئری را اجرا کرد؟ بله.

در اینجا ما در جدولی اطلاعات بازدیدکنندگان از یک سایت را داریم. در این جدول یک فیلد به نام Date داریم که جدول ما طبق این فیلد به ۴ پارتیشن تقسیم بندی شده است. و به صورت ۴ فایل توزیع شده است. این پارتیشن‌بندی را خودمان در زمان تعریف جدولمان مشخص می‌کنیم.

همانطور که ذکر شد پارتیشن ها هر کدام در فولدرهای متفاوت ذخیره شده اند. و در نتیجه ما در اینجا ۴فایل داریم.

تا اینجا با رابط وب کار کردیم. اما اکنون میخواهیم با کنسول کارهایمان را روی Hive انجام دهیم(Command Line). کنسول نیز یکی از راههای ارتباط با Hive است. در اینجا همه چیز را باید توسط کدنویسی انجام دهید.

رابط دیگر پروتکل ODBC است. درایور آن را می‌توان از سایت کلودرا بصورت فایل مخصوص سیستم عامل خودتان(در اینجا ویندوز) دانلود و نصب کنید. توسط یک نرم افزار خاص مانند Click View میخواهیم از طریق ODBC با Hive ارتباط برقرار می‌کنیم؛ همانطور که قبلا گفتیم ODBC یکی از ابزارهایی است که ارتباط دو ابزار ناهمگن (Heterogeneous) را با هم برقرار میکند. از طریق ODBC می‌توان با هر نرم افزاری که این پروتکل را ساپورت کند ارتباط برقرار کرد. (پروتکل ODBC هم یک نرم افزار جانبی دارد که با نصب آن می‌توان توسط آن از نرم افزارهایی مثل visual studio به Hive وصل شد.)

JDBC هم به همین ترتیب.

نکته: در محیط Hive می‌توان دستورات SQL را وارد کرده و جواب بگیریم. در صورتی که همان دستور را در My SQL و یا Oracle اجرا کنیم سرعت اجراها واقعا متفاوت است.

برای ارائه های بعدی چون مفاهیم کلی در این ارائه بیان شد سعی کنید مقایسه کنید، چند مطلب مرتبط در جاهای مختلف روی این ابزار پیدا کنید. موارد استفاده ابزار را مشخص کنید.