بررسی سیستم‌های پردازش داده large scale

زهرا گودرزی، دانشجوی کارشناسی ارشد، دانشگاه علم و فرهنگ تهران، zr_goudarzi@yahoo.com
علیرضا یاری، استادیار دانشگاه علم و فرهنگ تهران، a_yari@csri.ac.ir

چکیده

با حجم بی سابقه ای از داده‌ها و شتابی که در سرعت تولید داده وجود دارد و انواع مختلف ساختارهای داده که روبه‌رو شده‌ایم، پشتیبانی از تحلیل داده‌های بزرگ یک کار چالش برانگیز است. در پی آن یک تغییر مدل در معماری کامپیوترهای پردازش داده‌های بزرگ نیز بوجود آمد. رایانش ابری نیز یک مدل جدید برای فراهم آوردن زیرساخت محاسباتی لازم است که محل زیرساخت یا زیر ساخت را به شبکه، جهت کاهش هزینه‌های نرم‌افزاری و سخت‌افزاری تغییر می‌دهد[۱]. پیش از پیدایش مفهوم داده‌های بزرگ، از سیستم‌های مدیریت پایگاه داده موازی برای پردازش موازی استفاده می‌شد. با ظهور داده بزرگ و عدم امکان استفاده از پایگاه داده‌ای رابطه‌ای بعلت حجم بالای داده بزرگ، مدل مپ_ردیوس برای پردازش taskها معرفی شد. اما مپ_ردیوس مشکلات و محدودیت‌هایی داشت که محققان به این نیجه رسیدند که یکسری از قابلیت های SQL را در مپ_ردیوس جهت برطرف کردن یا کم‌رنگ‌کردن این محدودیت‌ها، ایجاد کنند و در نتیجه زبان های sql-like پدید آمدند. همچنین سیستم های مبتنی بر مپ_ردیوس در مقایسه با سیستم های مدیریت پایگاه داده موازی از کارایی کمتری برخوردار بودند درنتیجه سیستم‌های ترکیبی برای مرتفع کردن این مشکل به روی کارآمدند. در این تحقیق به بررسی مدلهای پردازش داده موازی ذکرشده و مزایا و معایب آنها می‌پردازیم.

کلمات کلیدی

داده بزرگ، پردازش داده‌های بزرگ، مپ_ردیوس، SQL_like

۱- مقدمه

با ظهور وب۲ نقش کاربران و برنامه های کاربردی تحت وب به سمت یک انقلاب رفت. کاربران که تنها نظاره‌گر بودند به تولیدکننده محتوا تبدیل شدند و منجر به تولید توده‌ای از داده‌ها بر روی سرورها شده است. در معماری کامپیوترهای امروزی بصورت نامتوازن شکافی بین پردازنده‌های چند هسته‌ای و دیسک‌های مکانیکی وجود دارد. گری استدلال کرد که در روند جدید به جای تمرکز بر روی یک ماشین بزرگ سریع بر روی کلاسترهایی از ماشین های کوچک و ارزان که بصورت موازی کار میکنند تمرکز داشته باشیم. در حال حاضر داده‌ها با سرعت باورنکردنی در حال رشد هستند و منابع انسانی و مادی قابل توجهی به پشتیبانی از عملیات فشرده‌سازی داده‌ها که منجر به هزینه های مدیریت و ذخیره سازی بالایی شده اند، اختصاص یافته است[۱].

این تغییر نیاز به روش های جدید برای ذخیره سازی حجم بالای داده ها و پشتیبانی از پرس و جو های کارا و موثر بر روی آن دارد. تحلیل این داده های خام با حجم بسیار بالا و به دست آوردن اطلاعات سودمند، افق تازه ای را برای الگوریتم های جدید و ابزارها و سرویس هایی برای پردازش این داده ها با حجم بالا در بازه زمانی معقول را باز کرد. در این راستا یک دسته از اصطلاحات در ادبیات مربوطه ظاهر شد. تمایز و رابطه داده بزگ، رایانش ابری و نواسکیوال را از نقطه نظر فنی روشن مینماییم[۲].

داده بزرگ

ابزار داده بزرگ به معنی حذف یا جایگزینی تکنولوژی پایگاه داده های سنتی نیست. در واقع تکنولوژی DBMS بخشی حیاتی برای حل مسئله در مدیریت داده بزرگ میباشد. داده بزرگ شامل سه بعد است که عبارتند از حجم بالای داده ها[۱]، شتاب در تولید داده[۲]، تعدد ساختارهای داده[۳].

رایانش ابری

مطابق تعریف NIST[۴] ، رایانش ابری شامل ۵ ویژگی ضروری میباشد:

۱- On-demand/self service
۲- Broad access network
۳- Resource pooling
۴- Rapid elasticity

۵- Measured service
علاوه برIaaS وPaaS وSaaS، cloud میتواند میزبان داده بزرگ باشد تحت عنوان database as a service یا DaaS[۲].

نواسکیوال

مفهوم نواسکیوال از سال ۲۰۰۹ پدید آمد. واژه نواسکیوال در مقابل پایگاه داده رابطه ای استفاده میشود برای اشاره به سیستم های پایگاه داده ای که توزیع شده اند و ممکن است که به اسکیمای ثابتی نیاز نداشته باشند و معمولا از عملیات join اجتناب میکنند و بطور افقی مقیاس پذیرند و ممکن است که رابط sql ی نداشته باشند و open source باشند.
در پایگاه داده های سنتی معمولا داده ها به آرامی وارد یک نمایش محلی میشوند قبل از اینکه برای پرس و جو آماده شوند، این باعث میشود که برای مدیریت و پردازش داده streamی مناسب نباشد. بعلاوه پایگاه داده های سنتی اغلب مقیاس پذیری برای داده بزرگ که حجم داده ها حداقل پتا بایت ها میباشد، ندارند. نواسکیوال وعده کارایی و دسترس پذیری بالا را در وضعیت غیر سنتی ذکرشده، میدهد. [۲]
ویژگی های مدل داده نواسکیوال مطابق موارد زیر است:

- **Key-value**: ساده ترین مدل داده، که در آن مقادیر عناصر داده ها، با استفاده از یک کلید منحصربفرد بازیابی می شوند.
- **Column family**: از row-store معمول استفاده نمیکند. در عوض، تعداد دلخواهی از زوج های key-value در سطرها ذخیره میشوند. Column family داده هایش را در جدول ذخیره میکند، اما ورودی ها به جای سطری، ستونی اند. Column family وابستگی جداول را اجازه نمیدهد. یک مثال معروف از Column family ، Hbase است.
- **Document**: یک Document ساختار key-value دارد با معنای value part، که به این معنی است که value part یک object معنی دار در سیستم است. مولفه value در JSON یا XML ذخیره میشود و میتواند انواع داده و ساختارهای داده پیچیده را ذخیره کند و همچنین میتواند به خوبی مورد پرس و جو قرار گیرد.

مدلهای برنامه نویسی که برای پیاده سازی application ها در سیستم های ذخیره سازی cloud پیشنهاد شده اند[۱]:

- MapReduce
- Sql-like
- Hybrid systems(PDBMS+MapReduce)

در ادامه این تحقیق در بخش ۲ به بررسی مدل برنامه نویسی mapreduce و معرفی و بررسی زبان سازال به عنوان یک زبان mapreduce-based میپردازیم. در بخش ۳ مشکلات و محدودیت های mapreduce بیان میشوند. در بخش ۴ زبان های sql-like که برای مرتفع ساختن برخی از محدودیت های mapreduce ایجاد شده اند با نام های pig-latin و hive وscope معرفی میشوند و ویژگی ها و مدل داده هریک از سیستم ها نیز بیان میشود. در بخش ۵، مقایسه ای بین PDBMSها وسیستم های mapreduce based و معرفی hybrid systemها که برای مرتفع کردن برخی از مشکلات و محدودیت های دیگر mapreduce به روی کار آمده اند معرفی میشود و hadoopDB بعنوان یک نمونه از این سیستم ها شرح داده میشود. در بخش ۶ خلاصه و جمع بندی سیستم ها در قالب جدول به نمایش آمده است. در بخش ۷ مراجع مورد استفاده و در بخش ۸ نیز زیرنویس ها ذکر شده است.

۲- MapReduce

Mapreduce یک مدل برنامه نویسی برای پردازش و تولید large data setها است. کاربر یک map function تعریف میکند که یک جفت key/value را برای تولید جفت key/value میانی پردازش میکند و یک تابع reduce را تعریف میکند که نتایج میانی با کلید یکسان را ادغام میکند. برنامه های نوشته شده با این مدل بصورت اتوماتیک موازی سازی را پشتیبانی میکنند و بر روی کلاستر بزرگی از ماشین ها اجرا میشود. سیستم در زمان اجرا مراقب جزئیات پارتیشن بندی داده های ورودی و زمانبندی اجرای برنامه بر روی کلاستری از ماشین ها و کنترل خطا و مدیریت ارتباطات بین ماشین ها میباشد. پیاده سازی مپ_ردیوس بسیار مقیاس پذیر است و روی کلاستری از ماشین ها اجرا میشود درنیجه کارایی بسیار بالایی نیز دارد و برنامه نویس را از درگیر شدن با جزئیات رها میکند[۳].چارچوب mapreduce، به زبان‌های مختلف پیاده‌سازی شده است که در ادامه به آنها میپردازیم.
در اصل مپ_ردیوس اصول زیر را درنظر گرفته است[۱]:
هزینه پایین سخت افزاری، مقیاس پذیری بالا، تحمل پذیری خطا و موازی سازی

۲-۱- زبان سازال

مرجع ۴، سازال را معرفی کرده است. در برنامه سازال عملیات روی یک تک رکورد از داده‌ها انجام می‌شود. در این زبان امکانی برای بررسی ورود چندین رکورد بطور همزمان وجود ندارد. تنها خروجی اولیه در این زبان emit statement است که data را به یک aggregator خارجی مانند sum و average و min و max میفرستد، که نتایج را از هر رکورد جمع میکنند و نتایج را پردازش میکنند. زبان سازال بعنوان یک کامپایلر متعارف یا مناسب پیاده سازی شده است، که با c++ نوشته شده و زبان مقصد یک interpreted instruction set یا مجموعه دستورات تفسیرشده است. درنتیجه کاربر source code راتوسط یک interface خارجی به سازال ارائه میدهد و سیستم آن را همراه با aggregatorهای فراهم شده خارجی کامپایل و اجرا میکند (ساختار سازال مانند library با یک interfaceخارجی است). data set در سازال معمولا در GFS ذخیره میشود. job Scheduling برای اجرای taskها روی خوشه ای از ماشینها توسط نرم افزار workqueu صورت میگیرد[۱].

data set های خیلی بزرگ اغلب ساختار flat اما regulat ی دارند و بین دیسک های و ماشین های متعدد توزیع شده اند. تنها به این دلیل که بسیار بزرگتر از آن هستند که در پایگاه داده رابطه ای مستقر شوند باید از پایگاه داده ای نواسکیوال استفاده شود.

سازال شامل ۲ فاز است. یک فاز یا مرحله، فاز تحلیل که در آن یک پرس وجو با استفاده از یک زبان برنامه سازی رویه ای بیان میشود، و در ادامه، داده را به فاز تجمیع ساطع میکند. هر دوی این فازها بر روی صدها یا هزاران کامپیوتر توزیع شده اند. در نهایت نتایج جمع آوری میشوند و در یک فایل ذخیره میشوند[۴].

البته هنوز هم تعدادی زیر مسئله برای حل کردن، باقی می‌ماند مانند اینکه محاسبه باید به بخش هایی تقسیم شود و با توجه به محلیت در سراسر ماشین ها توزیع شود. و زمانیکه تعدادی ماشین وجود دارد، احتمال fail شدن برخی از آنها در طول تحلیل وجود دارد، درنتیجه سیستم باید fault tolerance داشته باشد. مشکلات متعددی وجود دارند، اما آنها باید بدون مداخله کاربر از سیستم حذف شوند[۴].

ویژگی های سازال

- فایلها درجا پردازش میشوند به جای اینکه وارد database server شوند.
- هیچ جدول یا index از پیش پردازش شده ای وجود ندارد. درعوض هدف سیستم ساختن ad hoc table ها و ad index hocهای مناسب برای محاسبه است.
- از جهت دیگر سازال میتواند روی هزاران یا بیشتر از هزاران ماشین در پردازش موازی data setهای عظیم اجرا شود[۴].

۳-مشکلات و محدودیت های mapreduce

گفتیم MR یک چاچوب پردازش داده مقیاس پذیر است، hadoop که یک بستر برای ذخیره سازی داده بزرگ است و میتواند از الگوریتم قدرتمند مپ_ردیوس برای پردازش داده بزرگ استفاده کند[۲]. نیز پیاده سازی متن باز آن میباشد، که یک مدل برنامه نویسی با data flow ساده را پیشنهاد میدهد که برای بسیاری از کاربران جذاب است. هرچند درعمل سادگی بیش از حد MR مشکلات و مجدودیت های متعددی را بوجود میاورد از قبیل:

- یک ورودی و ۲ مرحله data flow دردسرساز است. taskهای با data flowهای مختلف (بعنوان مثال joinها یا n stages) را بطور مستقیم پشتیبانی نمیکند.
- کاربر باید بارها و بارها کد های مربوط به join و filtering و aggregation را بطور دستی تکرار کند که این باعث از دست دادن زمان، ایجاد bug در برنامه، کاهش خوانایی برنامه و مانع بهینه سازی است.
- حتی برای عملیات های معمول مانند filtering وprojection کدهای سفارشی نوشته میشود و این نشان دهنده این واقعیت است که استفاده مجدد و نگهداری سخت خواهد بود.
- طبیعت مبهم توابع map و reduce مانع توانایی سیستم برای بهینه سازی میشود. بسیاری از برنامه نویسان با MR ناآشنا هستند و این درحالی است که در SQL ماهر و متخصص اند و ترجیح میدهند از SQL بعنوان یک زبان اعلانی برای بیان taskها استفاده کنند، در حالیکه تمام جزئیات بهینه سازی به backend engin سپرده میشود.

در بخشهای زیر به بیان تحقیقات صورت گرفته و روشهای پیشنهاد شده برای مقابله با این مشکلات و اضافه کردن sql در بالای چارچوب MR میپردازیم[۱].

SQL-like -۴

pig-latin -۱-۴

در مرجع ۶ زبانی با نام pig latin ارائه شده است که یک موقعیت میانی بین بیان taskها با استفاده از gueryهای اعلانی سطح بالای SQL وبرنامه نویسی سطح پایین و رویه ای MR دارد. زبان Pig latin در پروژه pig که open source میباشد مورد استفاده قرار گرفته شده و توسط برنامه نویسان yahoo برای نوشتن task های تحلیل داده مورد استفاده قرار گرفته است[۳]. هدف pig موضعی بین MR و SQL است. Pig یک ساختار high-level و SQL-style را پیشنهاد میدهد. برنامه های pig به دنباله ای از jobهای mapreduce کامپایل میشود، و در محیط hadoop اجرا میشود. با تکیه بر hadoop برای موتور اجرایی آن، pig از ویژگی های مقیاس پذیری قابل توجه و تحمل پذیری خطا آن استفاده میکند. از جهت دیگر pig در حال حاضر قادر به بهینه سازی ساختارهای ذخیره شده مانند indexها و column groupها نیست. که البته تلاش هایی برای اضافه کردن این ویژگی ها به hadoop صورت گرفته است. در نهایت pig در yahoo بطور گسترده ای به تصویب رسیده، با هزاران job که روزانه اجرا میشود و همچنین به دلیل موارد استفاده موفق زیاد از آن، کشش هایی از خارج نیز به دست آورده است[۶]. نوشتن یک برنامه با pig latin شبیه به تعیین execution plan در پرس وجو است. که این متد برای برنامه نویسان حرفه ای جذاب تر از کدنویسی task ها بعنوان پرس وجوی sql و سپس اجبار سیستم برای انتخاب plan مطلوب از لحاظ بهینه سازی است.

ویژگی های Pig latin [۶]:

- انعطاف پذیری
- پشتیبانی از nested data model
- پشتیبانی از توابع user defined (UDF)
- قدرت عملیات بر روی فایل های ورودی plain مسطح بدون اطلاعی از schema آنها

Pig latin یک مدل داده ساده شامل ۴ نوع داده زیر را دارد:

- Atom
- Tupple
- Bag
- Map

معماری pig latin به گونه ای است که parsing یا تجزیه برنامهpig latin و ساختار logical plan آن مستقل از بستر اجراست. برنامه pig Latin قبل از اجرا یکسری transformation یا تبدیل دارد که در شکل زیر نشان داده شده است. در نهایت برای اجرا به hadoop پاس داده میشود[۶].

HIVE -۲-۴

Hive در سال ۲۰۰۹ توسط facebook توسعه پیدا کرد و هم اکنون تحت نظر apache software foundation است و بهترین گزینه برای OLAP جهت کنترل و پرس و جو بر روی حجم بالایی از داده است که بصورت توزیع شده ذخیره شده است. HDFS[۵] (در این محیط پردازش موازی و توزیع شده جدید، فایل ها ویژگی ها و رفتارهای متفاوتی دارند. DFS بطور معمول زمانیکه فایل خیلی بزرگ باشد(ترابایت) و update صورت نگیرد یا append-only باشد استفاده میشود، در غیر اینصورت اگر فایل کوچک باشد یا update های زیادی داشته باشد استفاده از DFS بی معنی خواهد بود. هر فایل به chunkهایی تقسیم میشود (معمولا ۶۴ مگا بایت) و در نودهای مختلف روی rackهای مختلف تکرار میشود. ارتباط بین chunkها و مکان آنها در یک فایل کوچک به نام master node ذخیره میشود. این فایل هم تکرار میشود. یک دایرکتوری برای کل فایل سیستم وجود دارد که مکان این کپی های داده ها را نگه میدارد. این فایل نیز تکرار میشود [2])اکوسیستمی است که HIVE با استفاده از آن داده ها را بصورت قابل اطمینان نگه میدارد و از شکست ها یا خطاهای سخت افزاری محافظت میکند. HIVE تنها روش انبار داده داده بزرگ رابطه ای است که برروی هدوپ اجرا میشود.

hive به جهت sql like بودن زبان پرس و جویی که دارد یعنی hiveql و همین طور به دلیل پشتیبانی از اکثریت عملیات های sql در RDBMS محبوب است. وقتی پرس و جو در interface مربوط به hive صادر میشود، دستخوش فازهای پردازشی متعددی میشود که شامل parsing و تحلیل معنایی و تولید logical plan وphysiacal plan و... است. در نهایت این plan به دنباله ای از عملیات mapreduce ترجمه میشود و در محیط hadoop اجرا میشود.

Sql زبانی مناسب برای پرس و جو برای داده های رابطه ای است. بزرگترین قطعه گمشده نواسکیوال نیز sql میباشد. در متن پایگاه داده و انبار داده، برای رسیدن به بالاترین سطح کارایی در پردازش، در ابتدا عملیات ابتدایی باید بهبود پیدا کنند. بعنوان مثال بهبود Join بعنوان پرهزینه ترین عملیات، منجر به بهبودکارایی زیادی میشود.

hiveQLیک زبان sql-like است و البته بطور کامل مطابق با آن نیست، و برای بیان پرس و جو در hive استفاده میشود. برخلاف enginهای استانداردSQL ، HIVE از update وdelete وrow-level insert وtransaction پشتیبانی نمیکند. Hive بعنوان مشتری hadoop ، فرض شده که عملکردش همراه با update های کم و ورود داده ها بصورت دسته ای باشد(batch-mode inserted). اینها ویژگی های پایه ای hive هستند. Hive یک انباره داده OLAP[۶] است و در واقع مناسب ترین گزینه برای OLAP میباشد. برای OLTP[۷] در داده بزرگ باید یک پایگاه داده نواسکیوال درنظر بگیرید و مثال خوبی برای آن Hbase است. هدف اصلی این پروژه آوردن مفاهیم آشنای relational DB (مانند جدول و ستون و پارتیشن) و زیرمجموعه ای از sql، به دنیای غیر ساخت یافته hadoop است به گونه ای که از انطاف پذیری و مقیاس پذیری hadoop برخوردار باشد. بنابراین تمام primitive type های اصلی مانند string وlist و struct را پشتیبانی کند.

Hive از عبارات پرس و جوی sql like و hive ql پشتیبانی میکند و در نتیجه برای افرادی که با sql آشنا هستند قابل فهم است. این پرس وجو ها به jobهای mapreduce کامپایل میشوند که با استفاده از hadoop اجرا میشوند.

مدل داده در hive

مفهوم پایگاه داده در hive، یک کاتالوگ یا namespace برای سازماندهی جداول است. زمانیکه یک پایگاه داده ایجاد میشود، یک directory با یک نامی که کاربر به آن میدهد با پسوند db. ایجاد میشود. تمام این دایرکتوری ها تحت یک دایرکتوری سطح بالا که بصورت پیش فرض /user/hive/warehouse است، ایجاد میشود.

انواع داده در hive اینگونه سازماندهی میشوند:

Table: که مشابه جدول در پایگاه داده رابطه ای است. هر جدول یک HDFS directory مربوطه دارد. داده ها به ترتیب در جدول قرار میگیرند و در فایلهایی در directoryاش ذخیره میشود.

partirions: هر جدول میتواند یک یا بیشتر پارتیشن داشته باشد که توزیع داده را در sub-directoryهایی از directoryهای جدول تعیین میکند. روش سنتی توزیع شدگی افقی در hive پشتیبانی میشود. پارتیشن ها ابزاری برای سازماندهی مجدد منطقی داده ها برای دسترسی سریع تر به داده ها میباشند. هر پارتیشن که برای جدول تعریف میشود وابسته به sub-directory تحت دایرکتوری است که جدول در آن قرار دارد. هر دو نوع جداول داخلی و خارجی میتوانند پارتیشن بندی شوند.

Buckets: داده در هر پارتیشن ممکن است به bucketهایی بر اساس hash ستون ها در جدول تقسیم شوند. هر Bucket بعنوان یک فایل در دایرکتوری پارتیشن ذخیره میشود. Hive از نوع داده های ابتدایی در ستون ها مانند integer وfloat وnumber وdate و boolean وgeneric strings و انواع مجموعه ای تودرتو مانند آرایه و map پشتیبانی میکند و همچنین کاربران میتوانند انواع خود را تعریف کنند[۷].

Hive نوع داده های معمول در پایگاه داده های رابطه ای را تعریف میکند و همچنین ۳ نوع مجموعه:

Struct وmap وarray .

مجموعه ها معمولا در پایگاه داده رابطه ای استفاده نمیشوند به دلیل اینکه منجر به شکستن فرم معمول یا طبیعی آنها میشود. عواقب شکستن فرم طبیعی آنها میتواند تکرار داده ها و هدر رفتن فضا و ناسازگاری باشد. با این حال در داده بزرگ فدا کردن فرم های طبیعی، امکان پردازش کارای دادهای پیچیده در رکورد را میسر میسازد.

scope -۳-۴

SCOPE یا Structured Computation Optimizaed For Parallel Execution یک زبان scripting است که برای تحلیل داده های large scale است. و برای انواع تحلیل دادها و appهای datamining در microsoft استفاده میشود.

scope زبان scripting به نام scope برای برای تحلیل داده های web scale بر روی کلاسترهایی از صدها یا هزاران ماشین است. Scope به عمد شباهت بسیار نزدیکی به sql دارد. این زبان زبان اعلانی و سطح بالا است و کامپایلر و بهینه ساز scope میتوانند scope

script را بهینه کند و آن را بهبود دهد(که بسیار حائز اهمیت است). تمام سخت افزار و جزئیات پیاده سازی برای کاربر transparent هستند. Scope بسیار توسعه پذیر است. این extentionها به کاربران اجازه میدهند که به شکلی موثر مسئله را حل کنند، درغیر اینصورت توسط sql این کار سخت میباشد. Execution plan های موازی تولید شده توسط کامپایلر و بهینه ساز scope از کلاسترها استفاده میکند. تجربیات و آزمایش ها تایید کرده اند که عملکرد یا کارایی پرس و جوها با حجم یا مقیاس داده و کلاسترها نسبت خطی دارد.[۸]
کاربران به سادگی میتوانند توابع شان را تعریف کنند و نسخه خودشان از عملگرها را پیاده سازی کنند. این انعطاف پذیری باعث شده این زبان گسترش پیدا کند و به کاربران اجازه دهد که مشکل هایی که به سادگی نمیتوانستند توسط traditional sql حل کنند را مرتفع سازند. Scope عملکردهایی را فراهم میکند که شبیه به sql است. این ویژگی ها modularity و قابلیت استفاده مجدد از codeها را بالا برده است. همچنین برای محدود کردن دسترسی به داده های حساس استفاده میشود.

۵- Hybrid System

۵-۱- مقایسه بین PDBMS و سیستم های mapreduce basded

طبق مقایسه و ارزیابی که توسط مرجع ۹ صورت گرفته نتایج زیر به دست آمده است:

- نتایج نشان داد که کارایی PDBMSها نسبت به hadoop MR به طور چشمگیری بالاتر بوده است.
- مصرف انرژی در PDBMSها بسیار کمتر از مصرف انرژی صدها یا هزاران نود شرکت کننده در mapreduce است.

این بالا بودن کارایی در PDBMS نتیجه تکنولوژی های متعدد است که طی ۲۵ سال انجام شده است مانند شاخص B-tree برای افزایش سرعت اجرای عملیات select، مکانیسم های ذخیره سازی جدید و تکنولوژی فشرده سازی و اجرای عملیات بطور مستقیم بر روی آنها و الگوریتم های موازی پیچیده برای اجرای پرس وجو بر روی حجم بالایی از داده های رابطه ای و ...
در نتیجه محققان به این نتیجه رسیدند که سیستم های پردازش داده large scale باید ترکیبی از محاسن روش های موجود باشد. ویژگی های قوی mapreduce به وضوح باید حفظ شوند و با کارایی و تکنیک های بهینه سازی پردازش پرس وجو سیستم های مدیریت داده سنتی همراه شوند[9].

مزایای hybrid system :[1]:

- مقیاس پذیری mapreduce
- تحمل پذیری خطا mapreduce
- کارایی parallel databases
- متن باز

مثال هایی از این سیستم ها hadoopdb و teradata هستند که به بررسی مختصر hadoopDB میپردازیم[۱].

۵-۲- hadoopDB

سیستم های DB موازی برای نزدیک دو دهه است که از لحاظ تجاری بر روی کار آمده اند. هدف اصلی این سیستمها بهبود کارایی با استفاده از موازی سازی عملیات مختلف مانند loading data، building index، evaluatingپرس وجو است. این سیستم ها معمولا برای اجرا بر روی shared nothing architecture که داده ممکن است ذخیره شود بصورت توزیع شده و با استفاده از multi cpuها، دیسک های موازی و network likeهای با پهنای باند بالا طراحی میشوند[۱].
نتایج مقایسه در مرجع ۹ نشان داده که سیستم های parallel DB کارایی یا عملکرد چشمگیری نسبت به MR دارند. از جهت دیگر پیاده سازی MR و تنظیمات و استفاده اش بسیار ساده تر و سر راست تر از parallel DB است. همچنین نشان داده شده که MR کارایی عالی در زمانیکه یک hardware failure اتفاق میافتد در حداقل کردن مقدار کاری که شکست خورده است، دارد. ایده اصلی hadoop DB اتصال یا متصل کردن تعدادی single node با استفاده از hadoop به عنوان مدیر یا هماهنگ کننده task و ارتباطات لایه های شبکه است.پرس وجو ها با SQL تعریف میشوند اما اجرایشان در سراسر nodeها با استفاده از چارچوب MR موازی است. بنابراین hadoop DB سعی میکند fault tolerance به دست آورد و قدرت عمل کردن در محیط ناهمگن با استفاده از ارث بری scheduling و پیاده سازی job tracking از hadoop را داشته باشد. بطور موازی برای به دست آوردن کارایی parallel DB با انجام اجرای پرس وجو های بیشتر درون DB engine تلاش میکند[۹].

۶- مراجع

[۱] S. Sakr, A. Liu, D. M. Batista, and M. Alomari, "A survey of large scale data management approaches in cloud environments," *Communications Surveys & Tutorials, IEEE,* vol. 13, pp. 311-336, 2011.

[۲] M. Mofidpoor, "Index-based Join Operations in Hive ",Concordia University, 2013.

[۳] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Communications of the ACM,* vol. 51, pp. 107-113, 2008.

[٤] R. Pike, S. Dorward, R. Griesemer, and S. Quinlan, "Interpreting the data: Parallel analysis with سازال," *Scientific Programming,* vol. 13, pp. 277-298, 2005.

[٥] C. Sauer and T. Härder, "Compilation of پرس وجو Languages into MapReduce," *Datenbank-Spektrum,* pp. 1-11, 2013.

[٦] A. F. Gates, O. Natkovich, S. Chopra, P. Kamath, S. M. Narayanamurthy, C. Olston, *et al.*, "Building a high-level dataflow system on top of Map-Reduce: the Pig experience," *Proceedings of the VLDB Endowment,* vol. 2, pp. 1414-1425, 2009.

[۷] A. Thusoo, J. S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, *et al*" ,.Hive: a warehousing solution over a map-reduce framework," *Proceedings of the VLDB Endowment,* vol. 2, pp. 1626-1629, 2009.

[۸] R. Chaiken, B. Jenkins, P.-Å. Larson, B. Ramsey, D. Shakib, S. Weaver, *et al.*, "SCOPE: easy and efficient parallel processing of massive data sets," *Proceedings of the VLDB Endowment,* vol. 1, pp. 1265-1276, 2008.

[۹] A. Pavlo, E. Paulson, A. Rasin, D. J. Abadi, D. J. DeWitt, S. Madden, *et al.*, "A comparison of approaches to large-scale data analysis," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, 2009, pp. 165-178.

۸ - زیرنویس ها

[1] volume
[2] velocity
[3] variety
[4] National Institue of Standards Technology
[5] Hadoop Distributed File System
[6] Online Analytical Processing
[7] Online Transactional Processing