پژوهشکده فناوری اطلاعات و ارتباطات

عنوان طرح

طراحی و پیاده‌سازی روشی برای حل مساله انباره داده در کلان داده

کد طرح: 20-5010

مجری طرح:

محمد تقی پور

شهریور 97

Archive of SID

www.SID.ir

بسم الله الرحمن الرحیم

Archive of SID

www.SID.ir

چکیده

بنابر اهمیت انباره‌های داده و وضعیت آنها در پاسخ‌دهی به تقاضاهای رو به گسترش کلان‌داده‌ها در صنعت IT لازم است روشهای مناسبی برای حل مسایل مربوط به این بخش ارائه گردد. امروزه انباره های داده با لحاظ درخواست‌های مربوط به انواع پردازش‌های خاص منظوره باید بتوانند مورد پایش داده-ای، گزارش‌گیری‌های متنوع، و تحلیل‌های درخواست شده قرار گیرند. در این طرح با هدف ایجاد بستری مبتنی بر کلان داده و در عین حال پاسخ‌دهی به نیازهای برخط راهکاری ارائه گردیده است. در این پژوهش تلاش گردید تا امکان اجرای همزمان پرس و جوهای برخط و تحلیلی (تاریخی) فراهم گردد. لذا در معماری پردازشی پیشنهاد شده اطلاعات هر سطر به‌نحوی قرار می‌گیرند که هر گره از لحاظ داده‌ای مستقل باشد. در دستیابی به اهداف طرح و برای ایجاد یک مزیت پردازشی قابل قبول در بستر کلان داده، با استفاده از ابزارهای موجود در اسپارک و به کمک زبان برنامه نویسی پایتون و اسکریپت‌های جاوا، یک راهکار گزارش‌گیری ایجاد گردید. این راهکار از طریق *API* خود و به صورت *REST Full* روی داده‌های موجود در انباره‌ها نسبت به تحلیل و تهیه گزارش‌های اولیه اقدام می‌نماید. مزیت اصلی این روش هزینه تمام شده پایین برای زیرساخت مورد نیاز و انعطاف در برقرار ارتباط با انواع انباره‌ها موجود و نرخ نسبی مناسب در زمان مورد نیاز برای گزارش‌های پایه‌ای در کلان داده است. با توجه به نتایج حاصل شده و در جهت پاسخ به نیاز صنعت *IT*، راهکار پیشنهادی می‌تواند روی انباره داده‌های سنتی موجود عملکرد بسیار مطلوب و سریعتری داشته باشد. به طوریکه در گزاش‌های پایه‌ای مربوط به کلان داده ممکن است زمان اجرای تهیه گزارشات در مواردی از چند ساعت به چندین دقیقه کاهش یابد. نتایج تجربی حاصل از پیاده‌سازی اولیه این روش در نمونه‌ای از صنعت اجرا و خروجی‌های فنی و مشاهدات مربوط به پاسخ‌دهی راهکار پیادسازی شده، ارائه گردیده است. عملکرد روش پیشنهادی می-تواند با بررسی‌های بیشتر در گزارش‌گیری های مربوط به داده‌کاوی روی کلان داده‌ها بهبود یابد.

کلید واژگان: کلان داده، نگاشت-کاهش، انباره داده، محلی‌سازی داده

Archive of SID

www.SID.ir

فهرست مطالب

Archive of SID

www.SID.ir

Archive of SID

www.SID.ir

فهرست شکل‌ها

Archive of SID

www.SID.ir

Archive of SID

www.SID.ir

فهرست جدول‌ها

Archive of SID

www.SID.ir

فصل 1:کلیات طرح

Archive of SID

www.SID.ir

1-1- مقدمه

حجم اطلاعات در جهان با سرعت نمایی رشد می‌کند. نرخ رشد اطلاعات نیز روز به روز سریعتر می‌گردد. مهمترین علت این موضوع تولید اطلاعات از طریق منابع مختلف اطلاعاتی جدید می‌باشد. در گذشته منابع تولید اطلاعات سیستم‌های پردازش تراکنش[1] (TPS) بودند. سپس این سیستم‌ها به سیستم‌های مدیریت اطلاعات[2] (MIS) تبدیل شدند و حجم بیشتری از اطلاعات تولید نمودند. با ظهور سیستم‌های هوش تجاری[3] (BI) و نیاز به نگهداری اطلاعات بر مبنای زمان، باز هم نیاز به حجم بیشتر فضای نگهداری اطلاعات وجود داشت. این حجم اطلاعات رفته رفته به حدی رسید که بیشتر از توانایی‌های پردازشی یک گره[4] گردید.

تولید روز افزون اطلاعات تنها مربوط سیستم‌های پردازش تراکنش (TPS)، سیستم‌های مدیریت اطلاعات (MIS) و سیستم‌های هوش تجاری (BI)، نمی‌باشد. حجم روزافزون اطلاعات دلایل دیگری نیز دارد. دلیل اول ظهور ابزارهایی[5] است که با سرعت زیادی تولید اطلاعات می‌نمایند. از مهمترین این ابزارها می‌توان به حسگر[6]ها، ابزارهای علمی مانند میکروسکوپ‌ها، ابزارهای هواشناسی نوین و ... اشاره نمود. این ابزارها داده‌هایی با ابعاد وسیع تولید می‌نمایند و با سرعت زیاد ارسال می‌کنند. حجم زیاد اطلاعات ارسالی توسط این ابزارها که تعداد آنها معمولاً زیاد است باعث ایجاد گلوگاه در ذخیره‌سازی این اطلاعات می‌گردد. همچنین بازیابی این اطلاعات خود می‌تواند معضل بزرگی را ایجاد نماید.

اخیرا نیز رشد شبکه‌های تبادل و اشتراک‌گذاری دیتا و محتوا مانند شبکه‌های اجتماعی[7] (SN) بر ابعاد و پیچیدگی پشتیبانی از پردازشی اطلاعات افزود. لذا علاوه بر ابزارهایی که با حجم زیاد اطلاعات تولید می‌کنند، سیستم‌هایی مانند شبکه‌های اجتماعی با امکان تولید و تبادل اطلاعات توسط تمام اعضای شبکه، گردش انبوهی از اطلاعات را شکل داده است. مدل تولید اطلاعات در گذشته به این نحو بود که یک منبع داده، مثلاً سایت خبری، اطلاعاتی را منتشر می‌نمود و بقیه کاربران از این اطلاعات استفاده میکردند. اما در مدل‌های شبکه اجتماعی تمامی کاربران محتوا ایجاد و تمامی آنها نیز از اطلاعات استفاده

---

[1] Transaction Processing System
[2] Management Information System
[3] Business Intelligence
[4] Node
[5] Device
[6] Sensor
[7] Social Networks

Archive of SID
www.SID.ir

می‌نمایند، لذا سرعت تولید اطلاعات بسیار بالاتر از گذشته شد. بنابر توضیحات فوق جهان با انفجار اطلاعاتی مواجه گردیده است به نحوی که سیستم‌های قبلی ذخیره‌سازی و بازیابی اطلاعات[8] توان مدیریت این سطح از اطلاعات و داده را ندارد. به عبارت دیگر سمپاد[9] (DBMS) موجود پاسخگوی نیاز ذخیره‌سازی و بازیابی اطلاعات نبودند.

به این نوع از داده‌ها که توسط سمپادهای موجود قابل مدیریت نبودند، کلان داده[10] اطلاق می‌گردد. تعریف دیگری که در مورد کلان داده وجود دارد عبارت است از داده ای که دارای سه خصوصیت حجم[11] ، سرعت[12] و تنوع[13] در داده‌ها باشد. البته بعضاً خصوصیات دیگری مانند ابهام[14] در داده‌ها نیز به خصوصیات قبلی اضافه شده است که لزوماً قابل اعتماد و اتکا نیستند و از نظر مفهومی ممکن است صحیح نباشد. معنای هر کدام از خصوصیات در ادامه بیان شده است.

- **حجم**: منظور از این خصوصیت آن است که حجم داده‌ها به قدری زیاد باشد که با سیستم‌های مدیریت داده کنونی قابل مدیریت نباشد.
- **سرعت**: منظور از این خصوصیت آن است که اطلاعات با سرعت زیادی تولید شده و اگر در زمان مقتضی مورد پردازش قرار نگیرند ارزش خود را از دست می‌دهند.
- **تنوع**: منظور از این خصوصیت کلان داده آن است که داده‌ها با قالب‌های گوناگون وجود دارد. (عدد، رشته، فایل و ...) و سیستم مدیریت کلان داده باید بتواند این گونه اطلاعات را مدیریت نماید.

برای مواجهه با کلان‌داده دو راهکار پیش روی توسعه دهندگان سیستم‌های مدیریت اطلاعات قرار دارد: راه حل اول تقویت خدمت دهنده‌های فعلی[15] است و راه حل دوم به استفاده همزمان از چندین خدمت دهنده اشاره دارد.

---

[8] data storage and reterival
[9] Database management system(DBMS)
[10] Big Data
[11] Volume
[12] Velocity
[13] Variety
[14] Veracity
[15] Scale Up

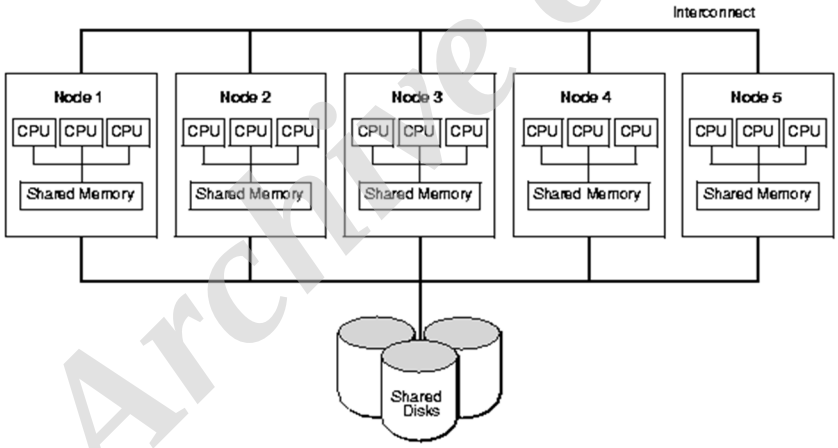
Archive of SID

www.SID.ir

راه حل اول بنا بر دلایل متعدد مناسب نیست. اول اینکه لزوماً هر خدمت دهنده‌ای را نمی‌توان ارتقاء داد. دوم اینکه در بسیاری از موارد هزینه ارتقاء بسیار بالاست. سوم اینکه استفاده از این روش با توجه به روند رو به رشد اطلاعات در نهایت منجر به شکست است و این روش تنها راه حلی موقتی برای حل این مشکل خواهد بود.

روش دیگر حل مساله این است که اطلاعات بر روی چندین گره[16] توزیع شود؛ این روش عملی‌تر به نظر می‌رسد. به منظور اجرای این روش دو معماری متداول است. معماری اول با عنوان حافظه مشترک[17] به این صورت است که در آن از گره‌های مختلف استفاده می‌شود. هر گره دارای پردازشگر مرکزی[18] و حافظه موقت[19] مخصوص خود می‌باشد؛ ولی دارای حافظه دائم مشترک[20] بین گره‌ها است.

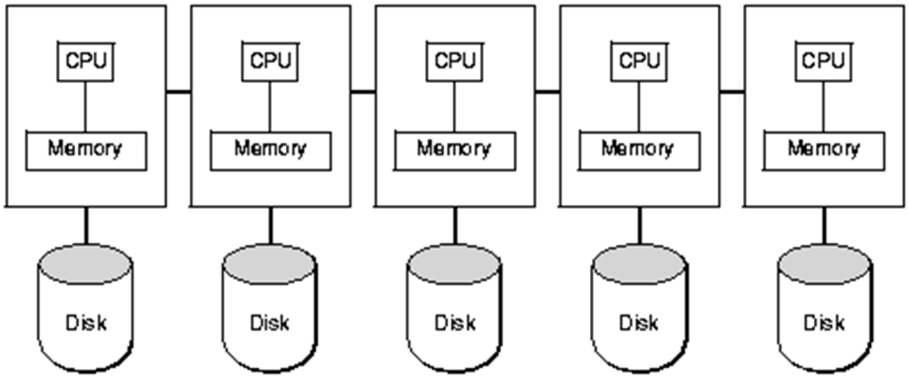
معماری دیگری که برای توزیع اطلاعات بین چندین گره مورد استفاده قرار می‌گیرد، معماری بدون اشتراک[21] می‌باشد؛ در این معماری هر گره خدمت دهنده دارای پردازشگر مرکزی، حافظه موقت و حافظه دائم جداگانه است.



شکل 1 :معماری اشتراکی

[16] Node
[17] Shared Memory
[18] CPU
[19] RAM
[20] HDD
[21] Shared Nothing

Archive of SID
www.SID.ir



شکل 2 :معماری بدون اشتراک

در مقایسه استفاده از معماری‌های حافظه مشترک و بدون اشتراک برای کلان داده‌ها نکاتی وجود دارد که عملاً استفاده از معماری حافظه مشترک را برای کلان داده غیر ممکن می‌سازد.

- اولین نکته‌ای که در استفاده از معماری حافظه مشترک وجود دارد این است که به دلیل اینکه گره‌های متعدد از یک حافظه دائم مشترک استفاده می‌نمایند، پیکربندی پیچیده‌ای برای گره‌ها باید صورت پذیرد.
- نکته دوم و مهم‌تر محدودیت تعداد گره‌ها در معماری حافظه مشترک می‌باشد. در واقع به دلیل همین نکته عملاً امکان استفاده از معماری حافظه مشترک برای کلان داده وجود ندارد، چون در کلان داده نباید محدودیت در تعداد گره‌ها وجود داشته باشد.

به منظور پردازش کلان داده بر روی معماری بدون اشتراک، روش‌های مختلفی مورد استفاده قرار گرفته است. یکی از این روش‌ها، نگاشت-کاهش می‌باشد. نگاشت-کاهش یک روش برنامه‌نویسی و پیاده‌سازی‌های مرتبط با آن به منظور ایجاد و پردازش مجموعه کلان‌داده‌ها می‌باشد [4]. وظیفه تابع نگاشت پردازش جفت‌های <کلید، مقدار> ورودی و ایجاد جفت‌های <کلید، مقدار> میانی می‌باشد. وظیفه تابع کاهنده ترکیب تمام مقادیر میانی تولیدشده به وسیله نگاشتگر بر اساس کلیدها و ایجاد جفت‌های <کلید، مقدار> نهایی می‌باشد.

اما معماری بدون اشتراک نیز خالی از اشکال نمی‌باشد. هنگامی که اطلاعات بر روی گره‌های مختلف تقسیم می‌گردد. هر گره به منظور پاسخگویی به پرس‌وجو باید با گره‌های دیگر ارتباط برقرار کند. تبادل

Archive of SID
www.SID.ir

اطلاعات بین گره‌ها باعث کندی اجرای پرس‌وجو می‌گردد. همچنین این تبادل باعث ایجاد گلوگاه در شبکه می‌شود. هنگامیکه گلوگاه در شبکه بوجود می‌آید و یا به هر دلیلی یک گره منتظر پاسخ پرس‌وجوی خود از گره‌های دیگر می‌شود، اشکال دیگری بوجود می‌آید و آن عدم استفاده بهینه از سخت‌افزار موجود هر گره می‌باشد. به عبارت دیگر هنگامیکه یک گره منتظر پاسخ از گره‌های دیگر می‌باشد منابع پردازشی و حافظه‌ای بیکار می‌مانند لذا از سخت‌افزار موجود در گره استفاده بهینه نمی‌شود.

اشکالات ذکر شده در مورد تمام داده‌هایی که به صورت تقسیم شده بر روی گره‌های مختلف نیاز به پیوند[22] داشته باشند صادق است. در واقع منشا تمام مشکلات ذکر شده وابستگی اطلاعاتی گره‌ها به یکدیگر می‌باشد.

همانطور که می‌دانیم به مجموعه داده‌های مرتبط با هم یک سازمان که پروسه تقسیم‌سازی مدیریت سازمان را پشتیبانی می‌نماید انباره داده[23] می‌گویند، که دارای خصوصیات زیر می‌باشد:

- موضوع محور
- پیوسته
- متغیر با زمان
- غیر فرار

اجزای مهم و اصلی انباره‌داده عبارتند از بعد[24]، اندازه[25]، اطلاعات مربوط به ابعاد و اندازه‌ها که در جدول وقایع[26] نگهداری می‌گردد. یکی از مدل‌هایی که برای نگهداری اطلاعات در انباره‌داده استفاده می‌شود، مدل ستاره‌ای[27] می‌باشد . در این مدل یک جدول وقایع در مرکز مدل و هر بعد با جدول وقایع ارتباط یک به چند دارد. البته مدل‌های دیگری برای انباره‌داده وجود دارد، مانند مدل دانه برفی[28] و مدل کهکشانی. مدل دانه برفی ، مانند مدل ستاره‌ای می‌باشد با این تفاوت که ابعاد می‌توانند با یکدیگر رابطه پدر-فرزندی داشته باشند. مدل کهکشانی مانند مدل دانه برفی می‌باشد با این تفاوت که در این مدل می‌توانیم چندین جدول وقایع داشته باشیم و ارتباطات جداول وقایع از طریق ابعاد می‌باشد.

---

[۲۲] *Join*
[۲۳] *Data warehouse*
[۲٤] *Dimension*
[۲٥] *Measure*
[۲٦] *Fact*
[۲۷] *Star schema*
[۲۸] *Snow flake*

Archive of SID

www.SID.ir

اندازه در انباره‌داده خصوصیتی (فیلد) می‌باشد که محاسبات (مجموع، میانگین،حداقل، حداکثر و ...) بر روی آن صورت می‌پذیرد. بعد یک خصوصیت است که جدول وقایع و اندازه‌ها بر اساس آن دسته بندی می‌شوند.

در این طرح سیستمی طراحی و پیاده‌سازی می‌گردد که قابلیت اجرای پرس‌وجو را به صورت برخط بر روی حجم اطلاعات بسیار زیاد(میلیارد رکورد) دارد. این سیستم بسیاری از مشکلات انباره داده سنتی را حل می‌نماید و با سخت افزار ارزان و موجود سرعت اجرای بسیار بالایی را ایجاد می‌نماید. این سیستم شامل یک سازنده پرس‌وجو می باشد که بر روی بستر انتخاب شده در این گزارش اقدام به ایجاد گزارشات مد نظر کاربران می نماید. بستر اجرای گزارش از بین سه محصول Spark، Hive و Elastic search انتخاب می‌گردد.

Archive of SID

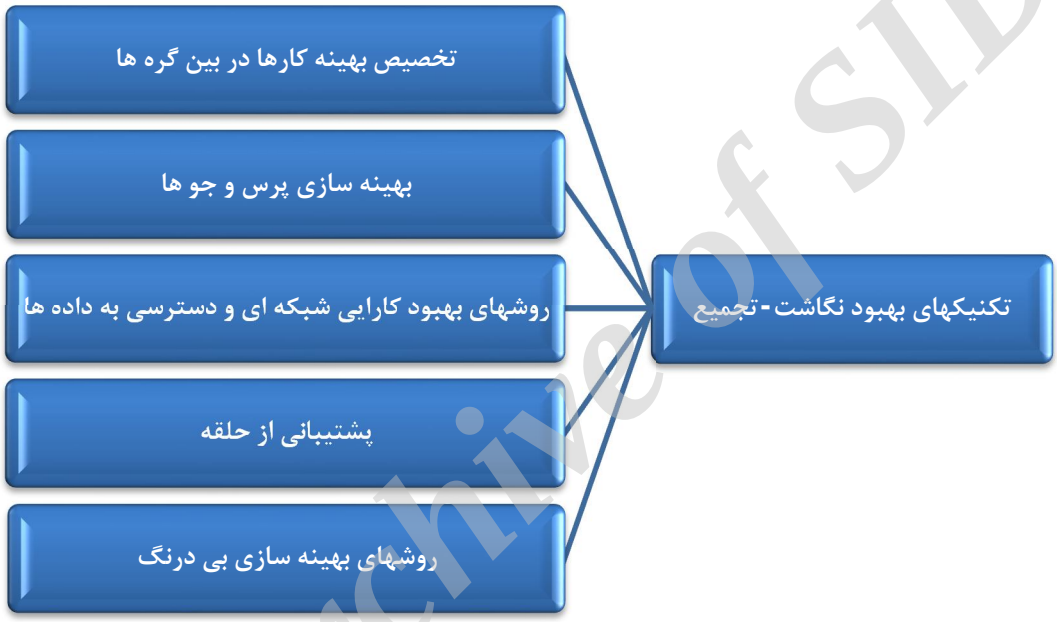
www.SID.ir

2-1- پیشینه تحقیق و پژوهش‌های مرتبط

دراین قسمت کارهای مرتبط با روش‌های ارائه شده در حوزه‌های کاراسازی نگاشت-کاهش، ابزارهای کلان‌داده و استفاده از نگاشت-کاهش برای حل مساله انباره‌داده‌ها در حوزه کلان‌داده‌ها مورد بررسی قرار می‌گیرد.

1-2-1- کارهای مرتبط با کاراسازی نگاشت-کاهش

در [5] کاراسازی‌های انجام شده بر روی ساختار نگاشت-کاهش به گروه‌های شکل 3 تقسیم شده است.



شکل 3: برخی تکنیکهای بهبود نگاشت-کاهش

یک گروه از بهینه‌سازی‌ها به تخصیص بهینه کارها بین گره‌ها می‌پردازند. در [6]MapReduce++ از روش تخمین مدت زمان انجام هر وظیفه[1] استفاده می‌نماید. سپس از هر کار[2] کوچک‌ترین وظیفه انتخاب انتخاب و اجرا می‌گردد و به این ترتیب زمان پاسخ کلی بهبود می‌یابد. در [7] از روش MapReduce++ استفاده می‌گردد با این تفاوت که زمانبندی بر اساس تخصیص پویای منابع بوسیله پارامترهایی که کاربر تعیین می‌نماید صورت می‌پذیرد. در روش Starfish[۸،۹] از بهینه‌سازی تخصیص منابع به صورت

[1] Task
[2] Job

Archive of SID
www.SID.ir

بی‌درنگ استفاده می‌شود. دراین روش نیازی به انجام تنظیمات به وسیله کاربر نیست. در این روش از بهینه‌سازی اجرای کارها بوسیله پایش نمودن وظیفه‌های مربوط به هر کار و ایجاد یک پروفایل به ازای هر وظیفه مربوط به یک کار استفاده می‌شود. همچنین دراین روش از زمانبندی آگاه از گردش‌کاری[1]، استفاده می‌شود. به وسیله این روش پردازش‌ها به گره‌ای منتقل می‌گردد که داده‌ها در آن قرار دارند لذا باعث عدم انتقال داده‌ها در سطح شبکه می‌شود.

گروه دیگر از روش‌های بهبود فرض را بر این گذاشتند که اطلاعات معمولاً به صورت ساخت‌یافته و تجمیع‌شده وجود دارند لذا از ساختارهای مانند SQL[2] می‌توان برای مدیریت آن‌ها استفاده نمود.به این گروه از ابزارها NOSQL[3] گفته می‌شود. تکنولوژی‌های NOSQL دارای مدل‌های داده منعطف،مقیاس پذیری عمودی و مدل داده‌های بدون طرح[4] هستند [42]. این پایگاه‌های داده به منظور تسهیل در کار با کلان‌داده ایجاد شده‌اند[43]. از این گروه می‌توان به Hive[10]، Pig[11]، Hbase[12]، Cassandra[13]، BigTable[14] اشاره نمود.

در گروه سوم از بهینه‌سازی‌ها پشتیبانی از حلقه و تکرار به ساختار نگاشت-کاهش افزوده شده است. در برخی از موارد مانند HaLoop[16] و Twister[15] حلقه به صورت وظیفه‌هایی از یک کار به تعداد دفعات مشخص یا به وقوع پیوستن یک شرط خاص تکرار می‌گردد. اطلاعات میانی مربوط به تکرارها در حافظه مشترک بین گره‌ها یا به صورت محلی برای بالا بردن کارایی نگهداری می‌گردند. در Nova[17] روشی ارائه شده است که تلاش می‌کند الگوریتم‌های افزایشی[5]/غیر افزایشی و تکرارشونده[6] را پشتیبانی نماید. این روش بر مبنای Hadoop و Pig طراحی و پیاده‌سازی شده است. دراین روش هنگامی‌که داده جدیدی وارد فرآیند پردازش می‌گردد، نیاز به پردازش اطلاعات قبلی نیست و تنها اطلاعات جدید باید پردازش شوند. در Spark[18] از روش RDD[7] به جای روش حافظه مشترک استفاده شده است که میزان زیادی از زمان را به خاطر کاهش نوشتن و خواندن از روی دیسک کاهش داده است. دراین روش تکه‌هایی از اطلاعات که در حافظه قرار دارند در صورت بروز اشکال قابل بازیابی هستند. این روش برای

---

[1] Workflow aware scheduling
[2] Structured query language
[3] Not Only SQL
[4] Schema less
[5] Incremental
[6] Iterative
[7] Resilient distributed datasets

Archive of SID

www.SID.ir

الگوریتم‌های تکرارشونده پیشنهاد می‌شود. در صورتیکه حافظه تعریف شده پر شود اطلاعات بر روی هارد دیسک‌ها براساس تنظیمات کاربران انتقال می‌یابد.

گروه بعدی روش‌هایی هستند که از طریق بهبود کارایی شبکه و دسترسی به داده‌ها اقدام به بهینه‌سازی روش نگاشت-کاهش نموده‌اند. روش‌هایی مانند ComMapReduce [19] و MRO-MPI [20] از این دسته هستند. در روشComMapReduce اطلاعات از نگاشتگرها به وسیله روش‌هایی که از فراداده‌ها استفاده می‌نمایند غربالگری می‌شوند و اطلاعات در صورت لزوم به کاهشگر انتقال می‌یابند. در روش MRO-MPI از روش‌های ارسال پیام بر روی ساختار نگاشت-کاهش استفاده شده است. در روش MRO-MPI نتایج میانی از نگاشتگر به سمت کاهشگر ارسال می‌شود، که این روش باعث می‌گردد نگاشتگر و کاهشگر به صورت موازی در حال اجرای الگوریتم باشند.

گروه دیگر از روش‌ها سعی در پیاده‌سازی مسائل بی‌درنگ با استفاده از الگوریتم نگاشت-کاهش دارند. به صورت کلی به دلیل اینکه داده‌ها باید به صورت دسته‌ای و در یک مرحله باید به ساختار نگاشت-کاهش وارد شوند، این ساختار برای حل مسائل بی‌درنگ مناسب نیستند. اما Chukwa[21] یک روش برای پردازش بی‌درنگ و تجمیع فایلهای لاگ ارائه شده است. دراین روش هر نگاشتگر به عنوان یک تولید کننده داده و هر کاهنده به عنوان یک جمع‌آوری کننده اطلاعات در نظر گرفته می‌شود. برای دریافت داده‌ها از نگاشتگرها دو روش در نظر گرفته شده است. اولی روش که مطمئن‌تر است نتایج بر روی یک حافظه واسط قرار می‌گیرد، و در روش دوم که سریع‌تر ولی نامطمئن می‌باشد استفاده از حافظه هر گره را دارد. همچنین روش‌هایی مانند Yahoo S4[22]، Twitter Storm[23] با ایجاد تغییراتی در اجزای ساختار نگاشت-کاهش سعی در حل مسائل نگاشت-تجمیع نموده‌اند. دراینجا مفهومی به نام گره‌های پردازشی تعریف می‌گردد که می‌توانند اطلاعات را پردازش کنند، اطلاعات جدیدی تولید کنند یا اطلاعات را تجمیع نمایند.

با توجه به پشتیبانی برخی از روش‌های ارائه شده از ماهیت تکرارشونده الگوریتم‌ها، ازاین روش‌ها می‌توان برای حل مسائل داده‌کاوی نیز استفاده نمود. در روش Mahout[24] تعدادی از الگوریتم‌های داده‌کاوی را جهت اجرا بر روی محیط‌های توزیع‌شده پیاده‌سازی نموده است، اما این الگوریتم‌ها با دید پردازش دسته‌ای و بدون پشتیبانی از تکرار در الگوریتم‌ها پیاده‌سازی شده اند. در روش Presto [25] با

Archive of SID

www.SID.ir

استفاده از زبان R نقائص Mahout برطرف شده است و با استفاده از پشتیبانی از تکرار زمان‌های بهتری نسبت به Mahout به دست آورده است.

روش‌هایی مانند بهینه‌سازی بی‌درنگ و تخصیص کارها به صورت بهینه، نیاز به شناخت از مساله دارد و بطور خاص منظوره مساله را حل می‌نماید. در صورت عدم شناخت از مساله ممکن است بار محاسباتی زمانبر و سنگینی برای حل مساله نیاز باشد. روش‌های پشتیبانی از حلقه تنها باعث اضافه شدن خاصیت تکرار به ساختار نگاشت-کاهش می‌شوند و این لزوماً به معنای حل کارای مساله نمی‌باشد. روش‌های بهبود کارایی شبکه و دسترسی به داده‌ها تا حدود زیادی بستگی به نوع مساله دارد و روش‌های ارائه شده در این زمینه معمولا به صورت موردی و بر اساس شرایط خاص عمل می‌نمایند. دراین پیشنهاد، روشی ارائه می‌گردد که داده مورد نیاز هر گره[1] را در همان گره و به صورت محلی قرار می‌دهد، لذا هزینه شبکه برای دسترسی به داده‌های مورد نیاز به صفر می‌رسد.

جدول 1 :روشهای بهینه سازی نگاشت کاهش

نام روش/ محصول	راهکار محوری	زمینه و کاربری روش	مزیت اصلی (با توضیح روشن از نحوه ایجاد مزیت در عمل)	نوع ارتباط با روش نگاشت-کاهش
MapReduce++	تخصیص بهینه کارها در بین گره ها	ایجاد بهینه سازی در زمان انجام وظایف	بهبود زمان پاسخ با ایجاد اولویت بین وظایف	بر پایه نگاشت کاهش و با ایجاد تغییر در روش اصلی نگاشت-کاهش
Starfish	از بهینه‌سازی تخصیص منابع به صورت بی‌درنگ	انتقال وظایف به گره دارای داده	استفاده از زمانبندی آگاه به گردش‌کاری و ایجاد پروفایل برای کاربر	بر پایه نگاشت کاهش و با ایجاد تغییر در روش اصلی نگاشت-کاهش
Hive	ایجاد واسط پرس و جو	انباره داده	تسهیل کارکرد با نگاشت -کاهش	بر پایه نگاشت کاهش و با ایجاد واسط کاربری مبتنی بر پرس وجو
Pig	ایجاد واسط	گردش کار با	تسهیل کارکرد با	بر پایه نگاشت کاهش و

[1] *Hardware node*

Archive of SID

www.SID.ir

	گردش کار	محوریت داده	نگاشت -کاهش	با ایجاد واسط کاربری مبتنی بر گردش کار
Hbase	ایجاد واسط پرس و جو	پایگاه داده	تسهیل کارکرد با نگاشت -کاهش	بر پایه نگاشت کاهش و با ایجاد واسط کاربری مبتنی بر پرس وجو
Cassandra	ایجاد واسط پرس و جو	پایگاه داده	تسهیل کارکرد با نگاشت -کاهش	بر پایه نگاشت کاهش و با ایجاد واسط کاربری مبتنی بر پرس وجو و تعداد نامحدود ستونهای داده
BigTable	ایجاد واسط پرس و جو	پایگاه داده	تسهیل کارکرد با نگاشت -کاهش	بر پایه نگاشت کاهش و با ایجاد واسط کاربری مبتنی بر پرس وجو
HaLoop	پیاده سازی حلقه	حل مسایل تکرار شونده	پیاده سازی حلقه و استفاده از حافظه میانی به منظور بالا بردن کارایی	بر پایه نگاشت کاهش و با اضافه نمودن ماهیت تکرار
Twister	پیاده سازی حلقه	حل مسایل تکرار شونده	پیاده سازی حلقه و استفاده از حافظه میانی به منظور بالا بردن کارایی	بر پایه نگاشت کاهش و با اضافه نمودن ماهیت تکرار
Nova	پیاده سازی حلقه	حل مسایل تکرار شونده	پیاده سازی حلقه و استفاده از حافظه میانی به منظور بالا بردن کارایی و استفاده از گردش کاری	بر پایه نگاشت کاهش و با اضافه نمودن ماهیت تکرار
Spark	استفاده از حافظه موقت	برای حل مسایل انباره	ایجاد پرفورمنس بالا با استفاده از حافظه	بر پایه نگاشت کاهش و با اضافه نمودن ماهیت

Archive of SID

www.SID.ir

		داده، گراف و داده کاوی	موقت	تکرار و استفاده از حافظه موقت
ComMapReduce	کارایی شبکه و دسترسی به داده‌ها	انتقال هوشمند داده ها از نگاشتگر به کاهنده	استفاده از ساختار پیام رسان به منظور اجرای بهتر وظایف	بر پایه نگاشت کاهش و اضافه نمودن ساختار پیام رسان
MRO-MPI	کارایی شبکه و دسترسی به داده‌ها	انتقال هوشمند داده ها از نگاشتگر به کاهنده	استفاده از ساختار پیام رسان به منظور اجرای بهتر وظایف	بر پایه نگاشت کاهش و اضافه نمودن ساختار پیام رسان
Chukwa	پردازش بی‌درنگ	تجمیع فایلهای لاگ	تجمیع نتایج بر روی یک حافظه واسط	بر پایه نگاشت کاهش و اضافه نمودن ساختاری برای حل مسایل بی درنگ
Yahoo S4	پردازش بی‌درنگ	پردازش توزیع شده استریم	یک پلاتفرم قابل توزیع، مقیاس پذیر و قابل برنامه ریزی است که به برنامه نویسان اجازه می دهد تا به راحتی برنامه های کاربردی برای پردازش جریان های دائمی و بدون محدودیت داده ها را توسعه دهند	بر پایه نگاشت کاهش و اضافه نمودن ساختاری برای حل مسایل بی درنگ
Twitter Storm	پردازش بی‌درنگ	پردازش توزیع شده استریم	یک پلاتفرم قابل توزیع، مقیاس پذیر و قابل برنامه ریزی است که به برنامه	بر پایه نگاشت کاهش و اضافه نمودن ساختاری برای حل مسایل بی درنگ

Archive of SID

www.SID.ir

			نویسان اجازه می دهد تا به راحتی برنامه های کاربردی برای پردازش جریان های دائمی و بدون محدودیت داده ها را توسعه دهند	
Mahout	داده کاوی	حل مسایل داده کاوی	پیاده سازی الگوریتمهای داده کاوی بر روی محیط توزیع شده و استفاده از این محیط جهت اجرای سریع این الگوریتمها	بر پایه نگاشت کاهش و اضافه نمودن ساختاری برای حل مسایل داده کاوی
Presto	داده کاوی	حل مسایل داده کاوی	پیاده سازی الگوریتمهای داده کاوی بر روی محیط توزیع شده و استفاده از این محیط جهت اجرای سریع این الگوریتمها	بر پایه نگاشت کاهش و اضافه نمودن ساختاری برای حل مسایل داده کاوی

Archive of SID

www.SID.ir

1-2-2- دسته بندی ابزارهای کلان‌داده

به صورت کلی مسائل مطرح شده و قابل حل در حوزه کلان‌داده‌ها را می‌توان به سه دسته شکل 4 تقسیم نمود [26].

شکل 4: دسته بندی مسائل حوزه کلان‌داده‌ها

الگوریتم‌های یادگیری و الگوریتم‌های تکرارپذیر، الگوریتم‌هایی هستند که برای تولید نتایج نهایی عملیات مربوط به الگوریتم چندین بار باید روی نتایج میانی اعمال گردد. پرس‌وجوها دسته دیگری از مسائل هستند که نتایج یک جستجو یا محاسبه را از منابع داده مختلف استخراج می‌نمایند و معمولاً تکرار در آن‌ها مشاهده نمی‌شود. دسته سوم مسائل مربوط به استریم[2]ها می‌باشد که معمولا با سرعت، حجم و نوع داده خاصی تولید شده و در بسیاری از موارد نیازمند محاسبات بی‌درنگ می‌باشند.

برای حل سه دسته مساله ذکر شده معمولاً از چهار روش حل مساله استفاده می‌شود که در شکل 5 آمده است[26].

[2] *Stream*

Archive of SID
www.SID.ir

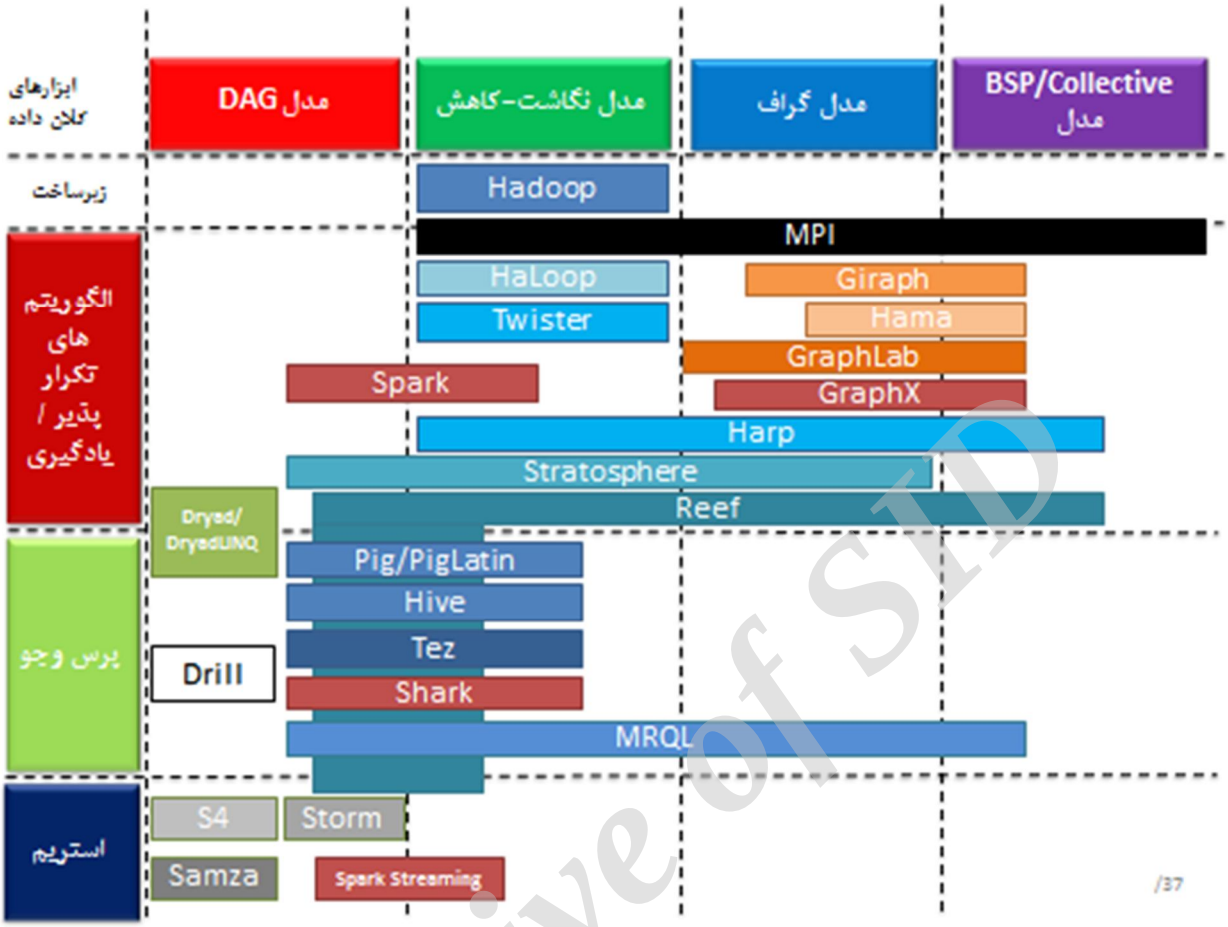
شکل 5: روش‌های حل مسائل حوزه کلان‌داده‌ها

روش گراف جهت‌دار بدون دور (DAG[3]) به این شکل است که در آن از ساختار گرافهای جهت‌دار و بدون دور و حلقه، جهت سامان‌دهی و نظم‌دهی به مسیرهای پردازش اصلی استفاده می‌گردد. گروه دیگر از راه‌حل‌ها در این رابطه از ساختار خود گراف برای حل مساله استفاده می‌نمایند. گروهی نیز روش‌های موازی و همزمان انبوه (BSP[4]) که از روش‌های موازی حل مساله می‌باشد را مورد اجرا قرار می‌دهند. در BSP الگوریتم روی گره‌های مختلف اجرا می‌شود و نتایج از طریق پیام‌هایی بین گره‌ها منتقل می‌گردد. روش‌های اشتراکی[5] مانند روش نگاشت-کاهش عمل می‌کنند با این تفاوت که گره‌ها اجرای همزمان الگوریتم را دارند و گره‌ای منتظر نتیجه اجرا توسط گره‌های دیگر نمی‌شود.

با توجه به توضیحات داده شده می‌توانیم نموداری از محصولات کلان‌داده‌ها به صورت شکل 6 داشته باشیم [26].

[3] *Directed acyclic graph*
[4] *Bulk synchronize process*
[5] *Collective*

Archive of SID
www.SID.ir



شکل 6: دسته بندی محصولات کلان‌داده‌ها[26]

همانطورکه در شکل 6 مشخص است، هدوپ به عنوان زیرساخت نرم‌افزاری برای بسیاری از محصولات تولید شده مورد استفاده می‌باشد. از آنجا که نگاشت-کاهش به صورت ذاتی از تکرار پشتیبانی نمی‌نماید، بسیاری از تلاش‌ها برای حل این مشکل و پشتیبانی از آن بوده است که Haloop، Twisterو Spark از معروف‌ترین محصولات در این زمینه هستند.

برخی دیگر از محصولات از زیرساخت هدوپ استفاده نمی‌نمایند. ساختار این محصولات غالبا بر اساس پیام می‌باشد. یکی از محصولات متداول در زمینه حل مسائل بر پایه پیام MPI است. با توجه به کارایی که MPI دارد در مدل‌های متنوعی به کار گرفته شده است.

Archive of SID
www.SID.ir

یک گروه از مسائل تکرارشونده مسائل مربوط به گراف‌ها هستند. در روش‌های پیاده‌سازی شده بر روی DAG و نگاشت-کاهش با محدودیت در زمینه تکرار داده‌ها مواجه هستیم و در بسیاری از موارد امکان جابه‌جایی داده‌ها به دلیل حجم بالا در بین گره‌ها وجود ندارد. در مسائل مربوط به گراف این موضوع بسیار برجسته‌تر می‌باشد. لذا همانطور که در شکل 8 مشاهده می‌شود، محصولات موفق در زمینه گراف از زیرساخت هدوپ و روش نگاشت-کاهش استفاده نکرده‌اند. برخی از محصولات در زمینه حل مسائل گراف عبارتند از: Giraph، Hama، GraphLab و GraphX می‌باشند.

برخی از محصولات مانند Harp، Reef و غیره از چندین روش برای پشتیبانی تکرار استفاده می‌نمایند. برخی از محصولات بر روی مساله پرس وجو تمرکز کرده‌اند. برخی از آن‌ها مانند: Dryad-LINQ و Drill از روش DAG برای حل مساله استفاده کرده‌اند، که این روش‌ها بر مبنای هدوپ نیز نمی‌باشند. البته روش‌های نگاشت-تجمیع در این حوزه به دلیل عدم وجود تکرار موفقیت بیشتری داشته اند. این روش‌ها بر مبنای هدوپ بوده و برخی از آن‌ها به صورت ترکیبی از نگاشت-تجمیع و DAG استفاده می‌نمایند. از مهم‌ترین محصولات در این زمینه می‌توان به Pig ، Hive، Tez و غیره اشاره کرد.

برخی از محصولات در حوزه پردازش استریم‌ها که اکثراً بر پایه DAG کاربردهای بلادرنگ دارند مانندS4، Storm برخی دیگر از محصولات مانند Spark streaming بر پایه DAG و نگاشت-کاهش کار می‌کنند.

روش‌های حل مساله ذکر شده دارای اشکالاتی می‌باشند که باعث افزایش زمان اجرا، عدم استفاده بهینه از سخت‌افزار و ایجاد ترافیک شبکه می‌شود. در ادامه به این اشکالات می‌پردازیم.

- **نگاشت-کاهش**: این روش به صورت ذاتی برای محاسبات دارای تکرار مناسب نمی‌باشد و روش‌هایی که سعی در شبیه‌سازی تکرار دارند برای مسائلی با تکرار نامحدود یا نامشخص کارایی مناسبی ندارند. به همین دلیل است که مسائلی مانند پیدا کردن کوتاه‌ترین مسیر در گراف‌ها با روش نگاشت-کاهش کارایی بسیار پایین و زمان اجرای زیادی دارد و در برخی از موارد قابل حل نمی‌باشد. همچنین مشکل محلی‌سازی داده‌ها در این روش مشکلات زیادی از قبیل ترافیک شبکه ایجاد می‌نماید.

Archive of SID

www.SID.ir

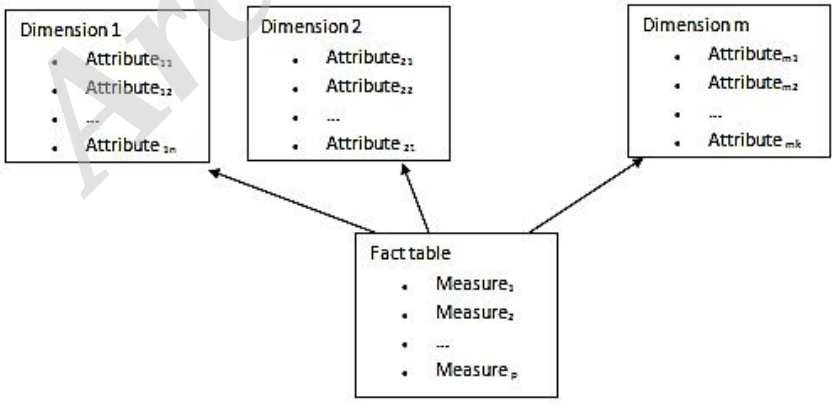
- **روش‌های گراف و BSP:** اینگونه روش‌ها برای مسائل خاصی کاربرد دارند به علاوه اینکه کاربرد این روش‌ها در این مسائل به معنی کارایی آن‌ها نمی‌باشد. مشکل عدم رعایت محلی‌سازی داده‌ها در این روش‌ها نیز دیده می‌شود و ساختار ارسال پیام می‌تواند منجر به ایجاد گلوگاه در شبکه شود.
- **روش DAG:** این روش نیز مشکلات روش نگاشت-کاهش از قبیل عدم رعایت محلی‌سازی داده‌ها را دارد. ضمن اینکه به دلیل ماهیت روش حل مساله امکان پشتیبانی از حلقه در آن وجود ندارد و در صورت پیاده‌سازی تکرار دارای مشکلات روش نگاشت-کاهش می‌باشد.

1-2-3- پژوهش‌های مرتبط با انباره‌داده‌ها در کلان داده

در این قسمت ابتدا مدل‌های متداول انباره داده‌ها مورد بررسی قرار می گیرد و سپس به بررسی کارهای مرتبط در زمینه ایجاد انباره‌داده در کلان‌داده می‌پردازیم. به طور کلی سه نوع طراحی برای انباره داده ها متداول مطرح می باشدکه عبارتند از [45]:

- **مدل ستاره ای**

در این روش یک جدول وقایع در وسط دیاگرام قرار می‌گیرد و ابعاد در کنار آن واقع می شوند. شکل 7 این مدل را نمایش می‌دهد.

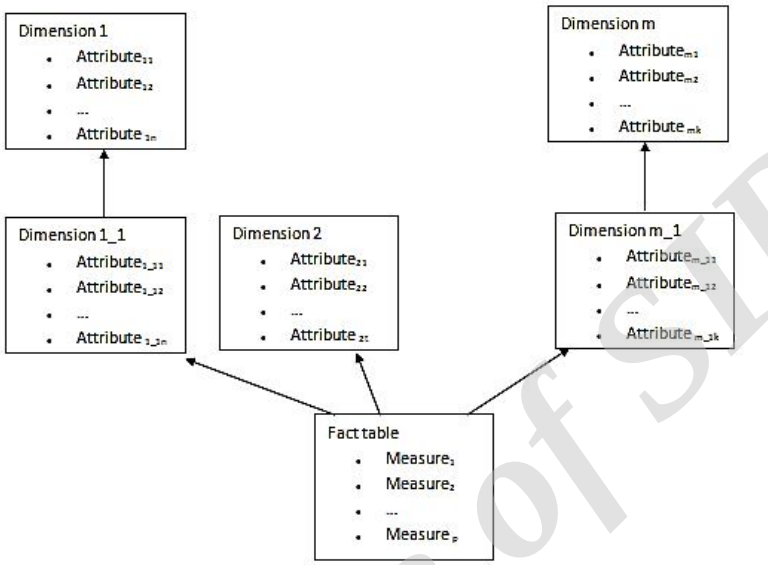


شکل 7: مدل ستاره‌ای [45]

Archive of SID
www.SID.ir

- **مدل دانه برفی**

این مدل مانند مدل ستاره‌ای است با این تفاوت که هر بعد خودش می تواند سلسله مراتب داشته باشد. شکل 8 این مدل را نمایش می‌دهد.



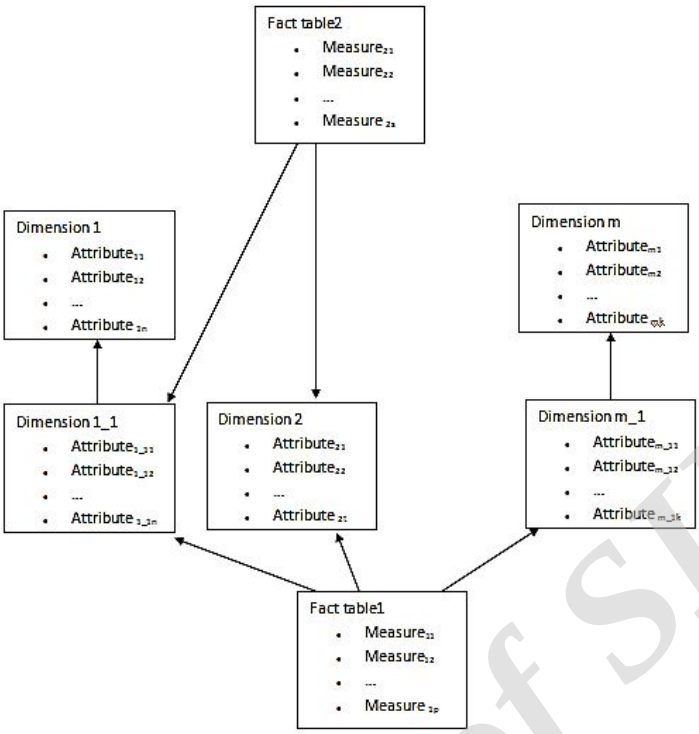
شکل 8 : مدل دانه برفی [45]

- مدل کهکشانی

دراین مدل علاوه بر موارد ذکر شده در دو مدل قبلی، چندین جدول وقایع وجود دارد. شکل 9 این مدل را نمایش می دهد.

Archive of SID

www.SID.ir

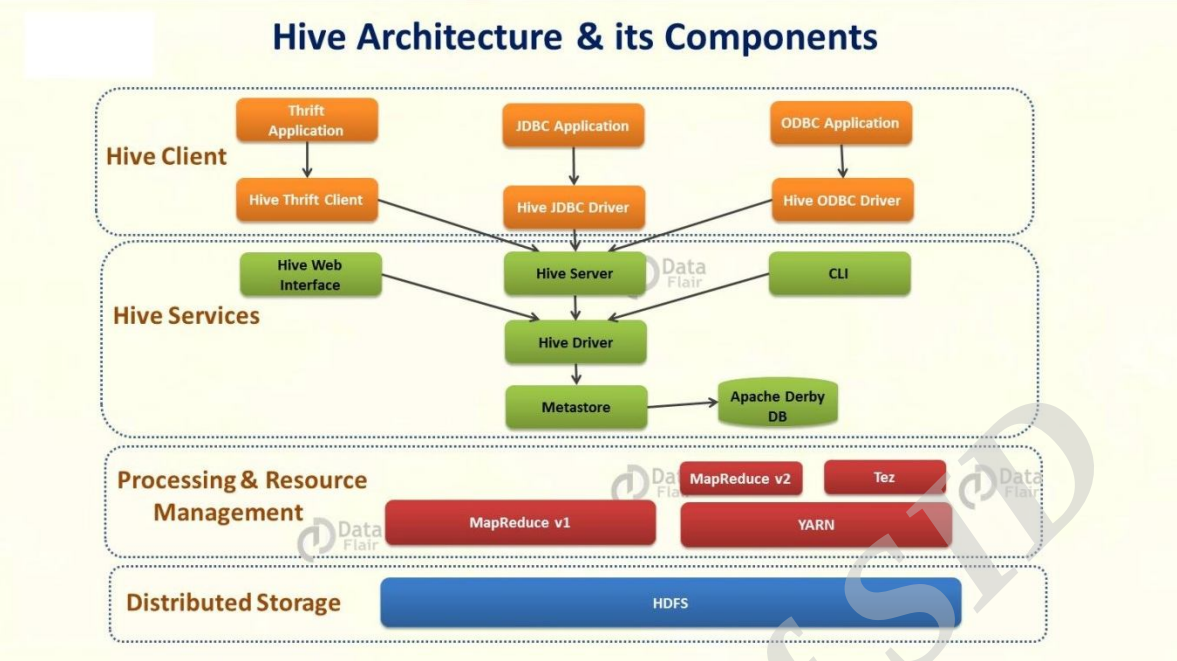


شکل 9: مدل کهکشانی [45]

با معرفی مدل‌های مختلف انباره‌داده اکنون به موضوعات مرتبط در زمینه انباره‌داده با مباحث کلان‌داده می‌پردازیم. روشن است انباره‌های داده در وضعیت کنونی و نیاز به پشتیبانی از کلان وضعیت خاصی پیدا کرده اند.

در زمینه کارهای مرتبط با انباره داده در کلان داده‌ها یکی از روش‌های معروف Hive[10] می‌باشد. در این روش یک واسط برای کاربر در نظر گرفته شده است که شبیه SQL است و نام آن HiveQL می‌باشد. کاربر پرس‌وجوهای خود را با زبان HiveQL به Hive می‌دهد و Hive این پرس وجو را به task های نگاشت-کاهش تبدیل می‌نماید. به دلیل اینکه پرس‌وجو ها توسط Hive به کارهای نگاشت-کاهش تبدیل می‌گردد کارکردن با این سیستم برای کاربران سهولت بیشتری دارد. شکل زیر معماری Hive را نمایش می‌دهد.

Archive of SID
www.SID.ir

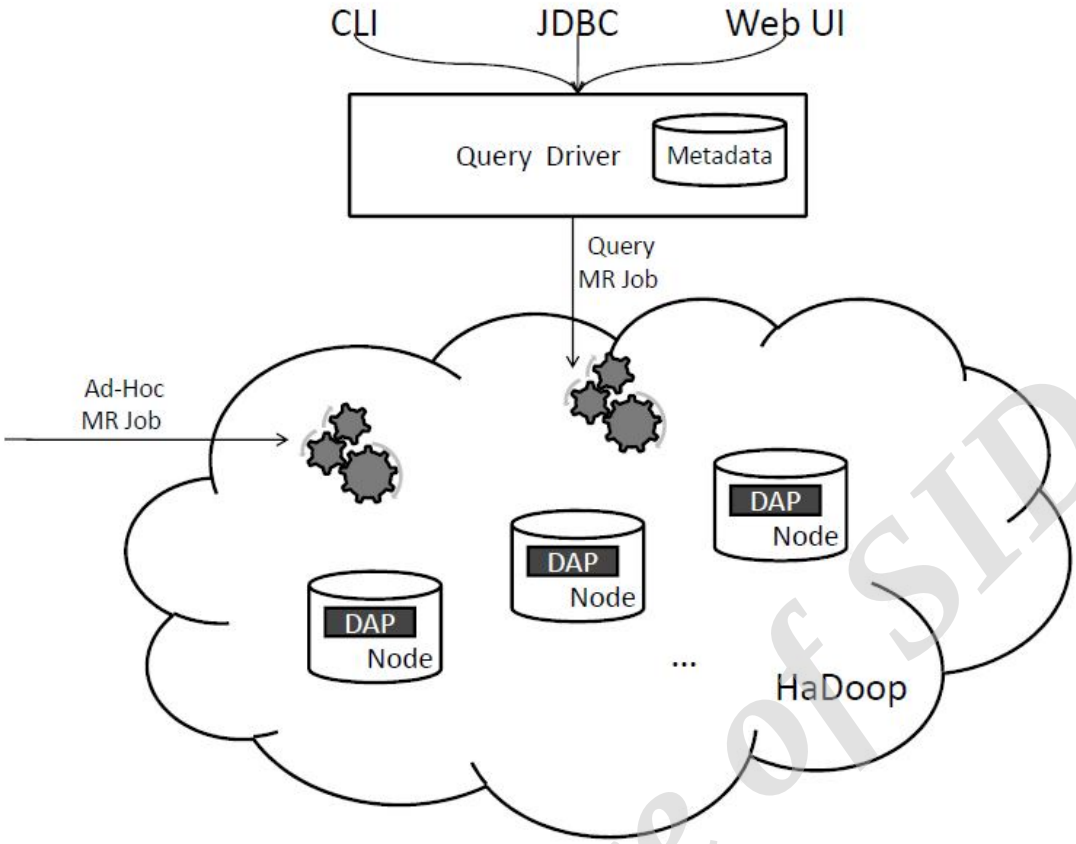


شکل 10 : معماری ***Hive***

به طور مشابه YSmart[28] نیز یک روش برای تبدیل SQL به نگاشت-کاهش ارائه داده است که ارتباط بین پرس‌وجوهای اجرا شده را نظر می‌گیرد. این امر باعث اجرای بهینه پرس‌وجوها می‌گردد. در Cheetah[29] یک انباره‌داده مبتنی‌بر نگاشت-کاهش معرفی شده است. این روش از دیدهای[6] مجازی بر روی مدل‌های موجود که بر پایه روش‌های ستاره‌ای و دانه‌برفی طراحی شده‌اند کار می‌کند.

[6] *View*

Archive of SID
www.SID.ir

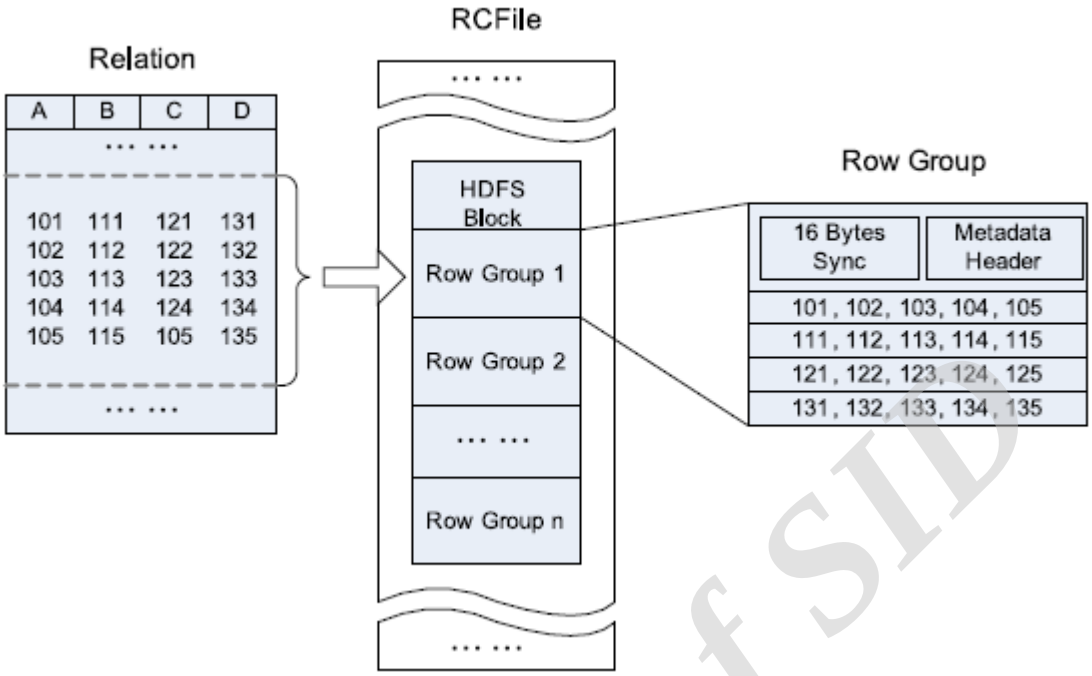


شکل 11: معماری **Cheetah**

در RCFile[30] روشی را برای جانمایی داده‌ها بر روی گره‌ها ارائه شده است. در این روش داده‌ها ابتدا به صورت افقی تقسیم می‌گردد (بر مبنای سطر) و سپس در فاز دوم به صورت عمودی (بر اساس ستون) تقسیم می‌شوند. به این ترتیب هم از مزایای روش تقسیم افقی استفاده می‌گردد و هم از مزایای روش عمودی. شکل زیر روش تقسیم داده ها را نمایش می‌دهد.

Archive of SID

www.SID.ir

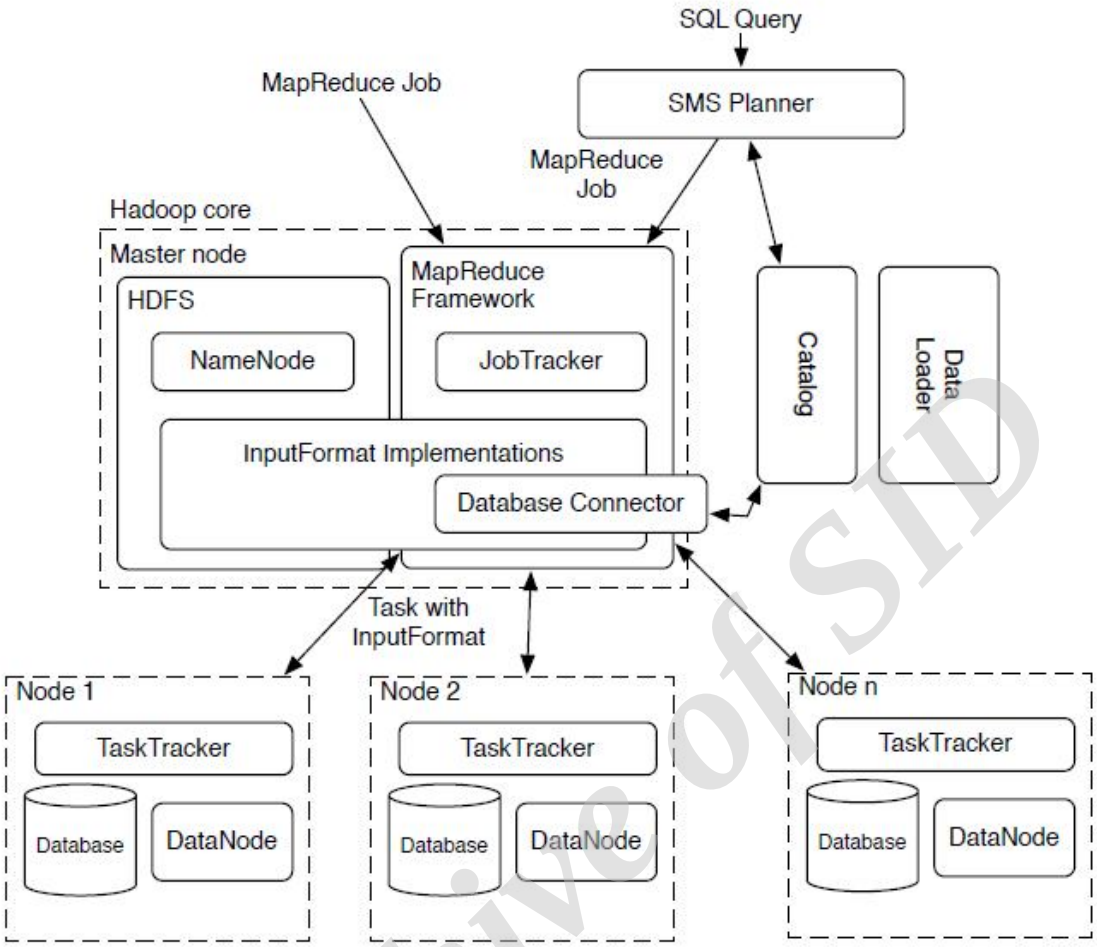


شکل 12: نحوه تقسیم داده ها در روش ***RCFile***

در [31] مقداری داده‌های آماری همراه با داده‌های نگاشت-کاهش ذخیره می‌شود که کارایی انباره‌داده‌ها را بهبود می‌دهد. در روش HadoopDB[32] از زیرساخت هدوپ برای مدیریت انباره داده‌ها استفاده شده است. بطوریکه بر روی هر گره داده یک پایگاه‌داده قرار گرفته است. بدین ترتیب از مزایای پایگاه‌های داده سنتی به صورت محلی و از مزایای مقیاس‌پذیر و توزیع‌پذیر هدوپ به صورت بین گره‌ای استفاده می‌شود. معماری روش HadoopDB در شکل زیر آمده است.

Archive of SID

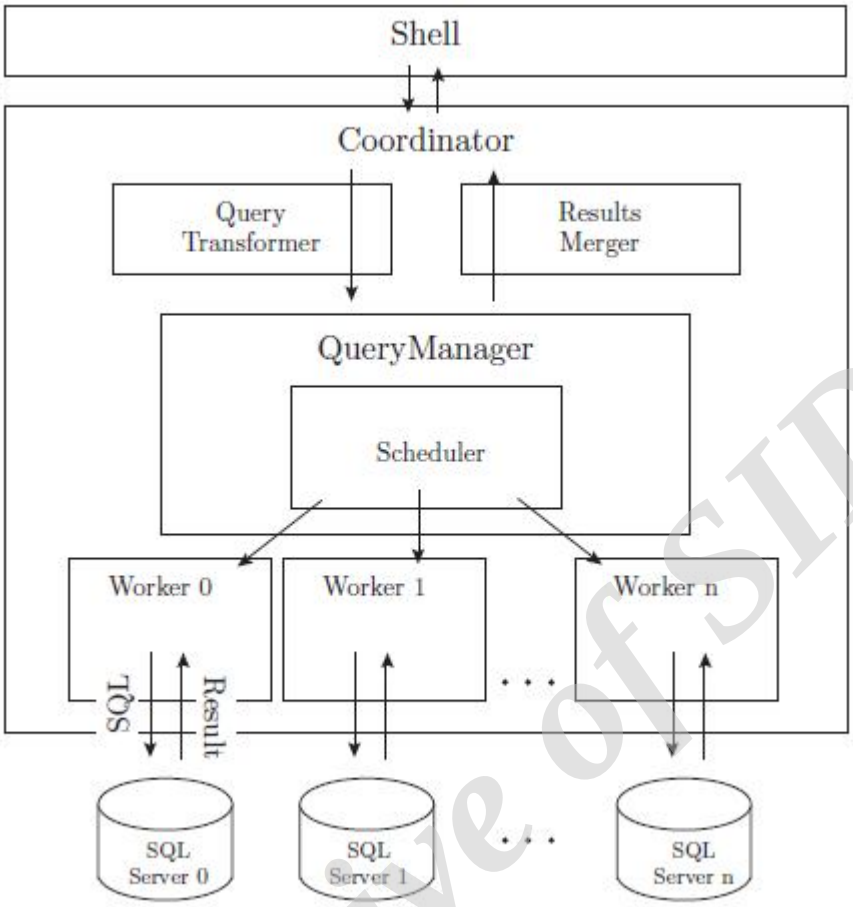
www.SID.ir



شکل 13: معماری روش HadoopDB

در روش Osprey[33] اطلاعات مربوط به جداول در بین گره‌ها تقسیم می‌گردد و هر قسمت از اطلاعات بر روی چند گره قرار گرفته است. همچنین هر پرس‌وجو به پرس‌وجوهای کوچکتری تقسیم می‌گردد که بر روی گره‌ها به صورت همزمان اجرا می‌گردند. معماری روش Osprey در شکل زیر آمده است.

Archive of SID
www.SID.ir

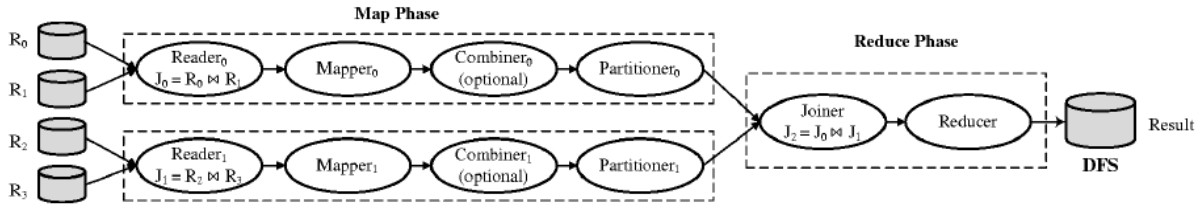


شکل 14: معماری روش Cyprey

در Clydesdale[34] اطلاعات مربوط به ابعاد بر روی دیسک محلی هر گره قرار می‌گیرد و جدول Fact بر روی HDFS قرار دارد. تابع نگاشت وظیفه پیوند بین جدول وقایع و ابعاد را دارد و تابع کاهش وظیفه گروه بندی و تجمیع را دارد. اما مشکل اصلی Clydesdale این است که فقط انباره‌داده با مدل ستاره‌ای را پشتیبانی می‌نماید و در صورتی‌که فضای مورد نیاز برای ذخیره‌سازی یک بعد از یک گره بیشتر باشد روش ارائه شده با مشکل مواجه می‌شود. در Llama[35] یک روشی است که در آن با ایجاد نوعی فایل به نام CFile بر روی هدوپ سعی شده تا سرعت بازیابی سیستم بهتر شود. در این روش نیز

Archive of SID
www.SID.ir

همچنان مشکل محلی‌سازی داده‌ها باقی است و تنها بهبودهایی در انجام عملیات join صورت پذیرفته است. شکل زیر روال بهبود عملیات join را نشان می‌دهد.

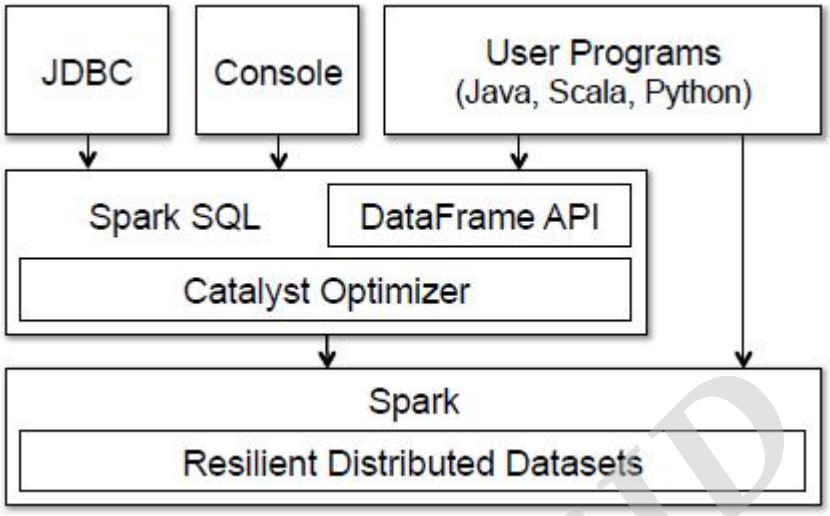


شکل 15: روال بهبود عملیات روش *Llama*

روش‌هایی مانند Shark[36] و SparkSQL[37] مبتنی‌بر Spark[38] می‌باشند. مزیت روش Spark این است که با استفاده از RDD سرعت اجرای عملیات بر روی Hadoop را تا 100 برابر افزایش داده است. که RDD مجموعه داده‌هایی می‌باشد که غیرقابل تغییر و به‌روزرسانی هستند و بر روی گره‌های مختلف قرار دارند. مجموعه RDDها به صورت موازی ساخته می‌شوند و در صورت بروز اشکال می‌توانند خود را بازسازی نمایند. این مجموعه داده‌ها قابلیت نگهداری بر روی هارد دیسک یا RAM را دارند. بخش Shark به وسیله Hive امکان مدیریت انباره‌داده‌ها را بر روی Spark فراهم آورده است. و SparkSQL نیز تلاش می‌کند تا قابلیتهای SQL را بر روی Spark برای کاربران فراهم باشد. شکل زیر معماری روش Spark را نمایش می‌دهد.

Archive of SID

www.SID.ir

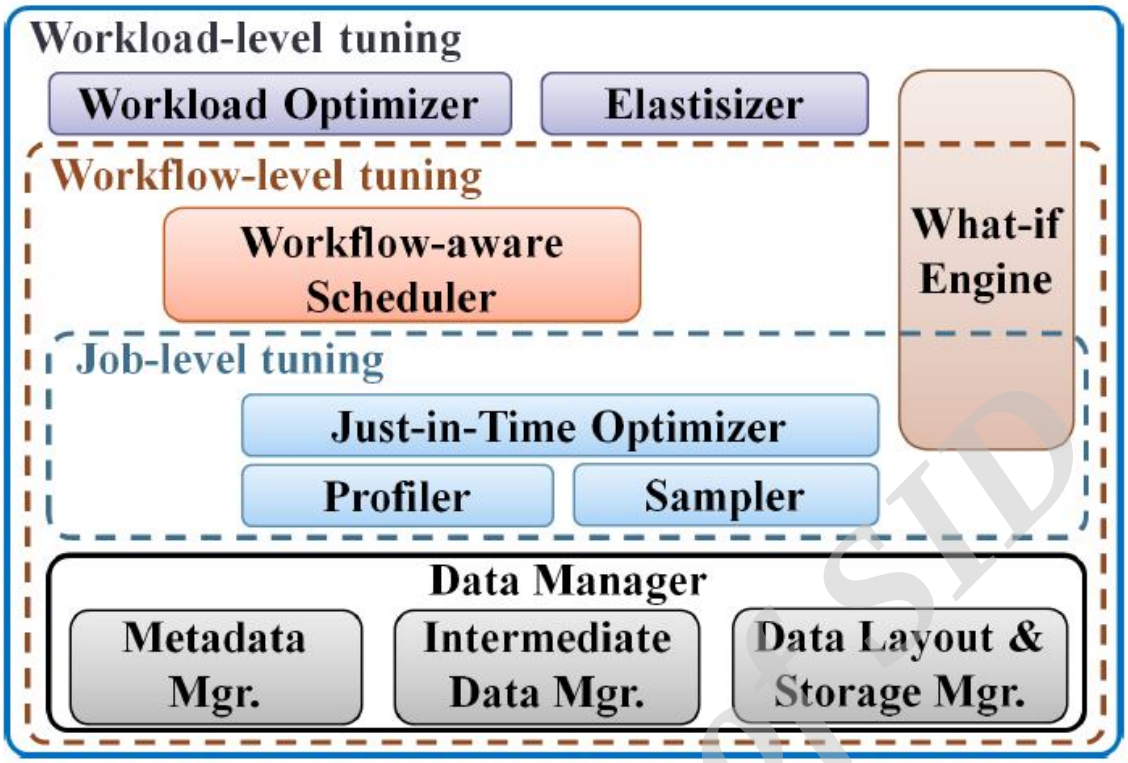


شکل 16: معماری روش ***Spark***

در روش Starfish[۸،۹] از بهینه‌سازی تخصیص منابع به صورت بی‌درنگ استفاده می‌شود. در این روش نیازی به انجام تنظیمات بوسیله کاربر نیست. در این روش از بهینه‌سازی اجرای کارها بوسیله پایش نمودن وظیفه‌های مربوط به هر کار وایجاد یک پروفایل به ازای هر وظیفه مربوط به یک کار استفاده می‌شود. همچنین دراین روش از زمانبندی آگاه از گردش‌کاری[7] استفاده می‌شود. بوسیله این روش پردازش‌ها به گره‌ای منتقل می‌گردد که داده‌ها در آن قرار دارند لذا باعث عدم انتقال داده‌ها در سطح شبکه می‌شود. شکل زیر معماری روش Starfish را نمایش می‌دهد.

[۷] Workflow aware scheduling

Archive of SID
www.SID.ir



شکل 17: معماری روش ***Starfish***

در روش CoHadoop[39] به برنامه‌کاربردی اجازه داده می‌شود که بتواند داده‌های مورد نظر خود را بر روی یک گره قرار دهد البته استفاده از این امکان در تمام مسایل امکان پذیر نمی‌باشد. در GridBatch[40] روش مانند CoHadoop است با این تفاوت که محلی‌سازی داده‌ها در لایه فایل سیستم انجام می‌پذیرد.

در روش Hadoop++[41] قراردادن فایل‌های مربوط به هم بر روی گره به صورت خودکار انجام می‌شود ولی این روش تنها می‌تواند خروجی‌های مربوط به یک کار را در کنار هم بر روی یک گره قرار دهد.

در EA2S2[44] یک لایه ذخیره سازی برای سیستم‌های نامتجانس طراحی شده است و با دسته بندی کارها سعی در بهبود محلی‌سازی داده‌ها نموده است.

روش SwiftAnalytics[45] یک روش ذخیره‌سازی برای اشیا را معرفی می نماید این روش از نوشتن آگاه از محلی سازی داده استفاده می نماید و بوسیله این روش ذخیره و بازیابی های اضافه حذف

Archive of SID
www.SID.ir

می‌گردند. در [46] از یک گردش کار بهینه‌سازی شده نگاشت-کاهش جهت الگوریتم زمان‌بندی استفاده می‌شود. کارها در مرحله اول به صورت کارهایی که نیاز به پردازش دارند یا کارهایی که نیاز به دیسک دارند تقسیم بندی می‌شوند و سپس بر اساس محلی‌سازی داده‌ها به هر بلاک اطلاعاتی زمان‌بندی تخصیص داده می‌شود. در[47] با استفاده از اطلاعات حجم کاری کارهای تکمیل شده قبلی منابع به نگاشتگر یا کاهنده تخصیص داده می شود. در[48] یک معماری جدید برای صف و یک الگوریتم زمانبندی که از دو سیاست کوتاهترین صف و بالاترین وزن تشکیل شده است جهت ایجاد تعادل بین محلی سازی داده ها و تقسیم بار بین گره‌ها استفاده شده است.

در روش Dawn[49] زمان بندی کننده تطبیق پذیر شبکه معرفی شده است که شامل یک طرح برخط و یک زمانبدی کننده تطبیق پذیر برای وظایف می باشد. زمان‌بندی کننده برخط وظیفه لحاظ کردن وابستگی کارها را داشته و مکانهای مناسب برای اجرای کارها را مشخص می نماید.

در روش AdPart[50] یک سیستم توزیع شده RDF[8] ارایه شده است که یک دسته بندی اولیه بسیار سبک روی داده‌های اولیه انجام می‌دهد به صورت همزمان بهینه ساز پرس و جو با استفاده از Hash ارتباط داده ای بین نتایج میانی را کمینه می نماید و با پایش دسترسی به داده‌ها به صورت مکانیزه قطعات داده پرکاربرد را بین گره‌های مختلف تکرار می نماید. جدول زیر روشهای مختلف و گرایش و مزیت هر روش را در حوزه نگاشت کاهش بیان می نماید

جدول2 : روشهای پیشنهاد شده برای انباره داده در کلان داده

روش بهبود نگاشت-کاهش	دسترسی به داده ها	پیشگیری از محاسبات تکراری	به انتها رسانی زود هنگام الگوریتم	پردازشهای تکرار شونده	بهینه سازی اجرای پرس‌وجو	تخصیص عادلانه وطایف	پردازش تعاملی یا بی درنگ
Hadoop++							
HAIL							
CoHadoop							
Llama							

[8] *Resource Description Framework*

Archive of SID

www.SID.ir

						■	*Cheetah*
						■	*RCFile*
						■	*CIF*
						■	*Trojan layouts*
					■		*MRShare*
					■		*ReStore*
					■		*Sharing scans*
					■		*Incoop*
				■			*EARL*
				■			*Top-k queries*
				■			*RanKloud*
			■				*HaLoop*
			■				*MapReduce online*
			■				*NOVA*
			■				*Twister*
			■				CBP
			■				Pregel
			■				PrIter
			■				PACMan
			■				REX
			■				Di erential data ow
		■					HadoopDB

Archive of SID

www.SID.ir

							SAMs
							Clydesdale
							Starsh
							AQUA
							YSmart
							RoPE
							SUDO
							Manimal
							HadoopToSQL
							Stubby
							SkewReduce
							SkewTune
							Sailsh
							Themis
							Dremel
							Hyracks
							Tenzing
							PowerDrill
							Shark
							M3R
							BlinkDB

Archive of SID

www.SID.ir

1-3- هدف طرح انباره داده در کلان داده

با اجرای این طرح ، امکان ایجاد انباره داده و آرشیو اطلاعات بر روی محیط های توزیع شده با سرعت بازیابی بالا و امکان گزارش‌گیری برای اهداف مختلف، فراهم می‌گردد. همچنین استفاده از سخت افزار در دسترس و عدم نیاز به تامین سخت افزارهای خاص از خصوصیات طرح اجرا شده می‌باشد. اجرای راهکار مورد نظر این طرح برای کلیه مواردی که دارای اطلاعات با حجم زیاد بوده و قابلیت نگهداری اطلاعات بر روی یک گره اطلاعاتی وجود ندارد، به عنوان یک راهکار کم هزینه دنبال می‌گردد.

1-3-1- سوال اصلی طرح

آیا امکان ایجا انباره داده ای که همزمان داده های آرشیو شده و داده های فعلی سیستم را با سرعت مناسب (در حد چند دقیقه) تحلیل و آنالیز نماید وجود دارد؟

1-4- روش پژوهش

روش اجرای طرح در هر یک از فازهای اصلی به شکل زیر است.

فاز اول: شناخت مساله و استخراج اشکالات و محدودیت‌های موجود در ابزارهای فعلی

در این فاز سوال اصلی طرح به صورت دقیق بررسی و چارچوب هدف پژوهش مشخص می‌گردد. ابزارها و روشهای موجود از لحاظ تئوری و شرایطی که برای استفاده کنندگان ابزارها و راهکارهای مورد بررسی وجود دارد، جمع آوری و در نهایت با استفاده از این اطلاعات به جمع بندی در مورد انتخاب ابزار مناسب در حال حاضر برای انباره داده در کلان داده می‌رسیم.

فاز دوم: ارائه و تحلیل راهکار پیشنهادی

به منظور پیاده‌سازی طرح پیشنهادی در این مرحله تحلیل نرم افزار صورت می‌پذیرد. خروجی این فاز *Use case* و *Use case specification* خواهد بود. در این قسمت خصوصیات کارکردی سیستم به صورت کامل شرح داده می شود. سپس نرم افزار پیشنهادی به صورت کامل طراحی می گردد. که فرآیند آن شامل تعیین موارد زیر است:

Archive of SID

www.SID.ir

- تشکیل نمودار کلاس، نمودار فعالیت، پروتوتایپ محصول. بعد از این مرحله تمام موارد لازم برای پیاده سازی محصول آماده می باشد
- پیاده سازی روش پیشنهادی شامل: طراحی *API*، و مشخصه‌های بخش گزارش‌گیری و تحلیل فرایند مدیریت انباره داده توسط محصول پیشنهادی. محصول (یا همان ابزار تولیدی) با توجه به تعاریف اولیه از سه قسمت زیر تشکیل شده است: 1- واسط ساخت پرس وجو، 2- ارسال پرس و جو مربوط به نگاشتگر، 3- تجمیع نتایج از نگاشتگرها توسط کاهنده، 4- نمایش نتایج نهایی به کاربر، 5- ارایه خروجی به صورت اکسل.

فاز سوم طرح: در این مرحله پیاده سازی در محیط کاربردی صورت می‌گیرد.

انتقال اطلاعات از منابع اطلاعاتی به محیط پیشنهادی و سپس آزمون روش پیشنهادی. به منظور آزمون روش پیشنهادی سرعت بازیابی اطلاعات در روش پیشنهادی با ابزارهای متداول و استخراج شده در روش قبل به صورت تجربی مقایسه می گردد. برای ساخت محیط آزمون نیاز به اطلاعاتی داریم که روش پیشنهادی و سایر ابزارهای انتخاب شده پرس و جوهای خود را بر روی آن اجرا نمایند. برای تامین داده از اطلاعات سوییچ یکی از بانکها استفاده می نماییم. حجم داده انتخاب برای آزمون 1,2 میلیارد رکورد است که اطلاعات مربوط به یکسال سوییچ می باشد. این اطلاعات حدود 80 فیلد می باشند. مراحل مختلفی برای انتقال اطلاعات باید صورت پذیرد که مهمترین آنها عبارتند از: 1- بررسی منابع داده، 2- بررسی فیلدهای مورد نیاز، 3- بررسی کیفیت داده و تصمیم گیری در مورد مقادیر غیر موجود و اشکال دار، 4- طراحی *ETL* جهت انتقال داده با حداکثر *Performance*، 5- *Tuning* داده ها در محیط مقصد پیشنهادی، 6- انتقال داده ها به محیط های *Hive*، *Spark* و *Elastic search*.

فاز چهارم: ارزیابی و مقایسه راهکار

گزارش مقایسه روش پیشنهادی با سیستم های مطرح و متن باز انباره داده *Hive* و *SparkSQL* و روش‌هایی نظیر *elasticsearch*.

در این قسمت پرس و جوهای تجمیعی مربوط به بانک بر روی اطلاعات *ETL* شده اجرا می گردد. روشها از دو جنبه سرعت بازیابی پرس وجو و حجم مورد نیاز برای ذخیره سازی داده ها مورد ارزیابی قرار می گیرند. برخی از پرس وجوها مورد نیاز بانک عبارت است از

Archive of SID

www.SID.ir

- میزان کارکرد کانالهای پرداخت به ازای ماههای مختلف
- میزان کارکرد کانالهای پرداخت به ازای استانهای مختلف
- میزان کارمزد پرداختی به ازای استانهای مختلف
- تعداد خرید شارژ تلفن همراه به تفکیک اپراتور
- میزان پرداخت قبض به ازای صادرکننده
- تعداد کد پاسخ رمز نامعتبر است به ازای شهرهای مختلف در ماههای گوناگون
- میزان کارکرد کیوسک به ازای انواع تراکش
- حجم تراکنشهای انجام شده در فاصله ساعات 12 شب تا 6 صبح به ازای استانهای مختلف
- حجم تراکنشهای انجام شده در فاصله ساعات 12 شب تا 6 صبح به ازای مناطق پرخطر
- حجم تراکنشهای انجام شده در فاصله ساعات 12 شب تا 6 صبح به ازای مناطق آزاد
- حجم ریالی و تعدادی تراکنشهای خرید شارژ به ازای کانالهای پرداخت
- حجم ریالی تراکنش دریافتی از سایر بانکها به تفکیک بانک
- میزان کارمزد دریافتی از سایر بانکها به تفکیک بانک
- ...

Archive of SID

www.SID.ir

فصل 2

راهکارها و ابزارهای مهم مدیریت انبار داده در کلان داده

Archive of SID

www.SID.ir

2-1- مقدمه

در این فصل به بررسی سه ابزار Spark، Elastic Search و Hive و امکاناتی که برای برای پیاده‌سازی انباره داده برخط ارائه می‌کنند، می‌پردازیم. در پایان این فصل بنابر چارچوب مورد نظر برای اهداف مورد نظر این طرح یکی از ابزارها که امکان و قابلیت‌های بیشتری ارائه می‌دهد را به عنوان بستر پیاده‌سازی تعیین نموده‌ایم.

2-2- اسپارک

اسپارک [51]محاسبات خوشه ای سبک و سریع برای محاسبات سریع (HPC) طراحی شده است. اسپارک در لایه بالایی Hadoop MapReduce می‌باشد و مدل MapReduce را برای موثر بودن انواع بیشتری از محاسبات که شامل کوئری های تعاملی (Interactive Queries) و جریان پردازش (Stream Processing)می باشد ، گسترش می دهد. اسپارک یکی از زیر پروژه‌های Hadoop است که در سال 2009 در آزمایشگاه AMPlab برکلی توسط Matei Zaharia ساخته شد. در سال 2010 تحت لیسانس BSDعضو متن باز (Open Source) شد. در سال 2013 تحت حمایت بنیاد نرم‌افزاری آپاچی قرار گرفت و حالا پروژه اسپارک در فوریه 2014 در بالاترین سطح پروژه های آپاچی قرار گرفت. از ویژگی های اسپارک می‌توان به نکات زیر اشاره کرد:

- **سرعت**

اسپارک کمک می‌کند تا برنامه‌ها در خوشه ی Hadoop اجراشوند، این یعنی بیش از 100 برابر سرعت اجرا در حافظه و بیش از 10 برابر سرعت اجرا روی disk را پشتیبانی می‌کند. این اتفاق به وسیله کم کردن تعداد عملیات خواندن/نوشتن روی دیسک میسر می‌شود. همچنین پردازش داده های متوسط (Intermediate) را در حافظه ذخیره می‌کند.

- **پشتیبانی از چندین زبان**

اسپارک API های از پیش تعیین شده‌ای برایjava ، Scala یا Python دارد. بنابراین شما می‌توانید به زبان‌های مختلف برنامه تان را بنویسید، اسپارک دارای 80 دستور سطح بالا (high-level) برای کوئری های تعاملی می باشد.

Archive of SID

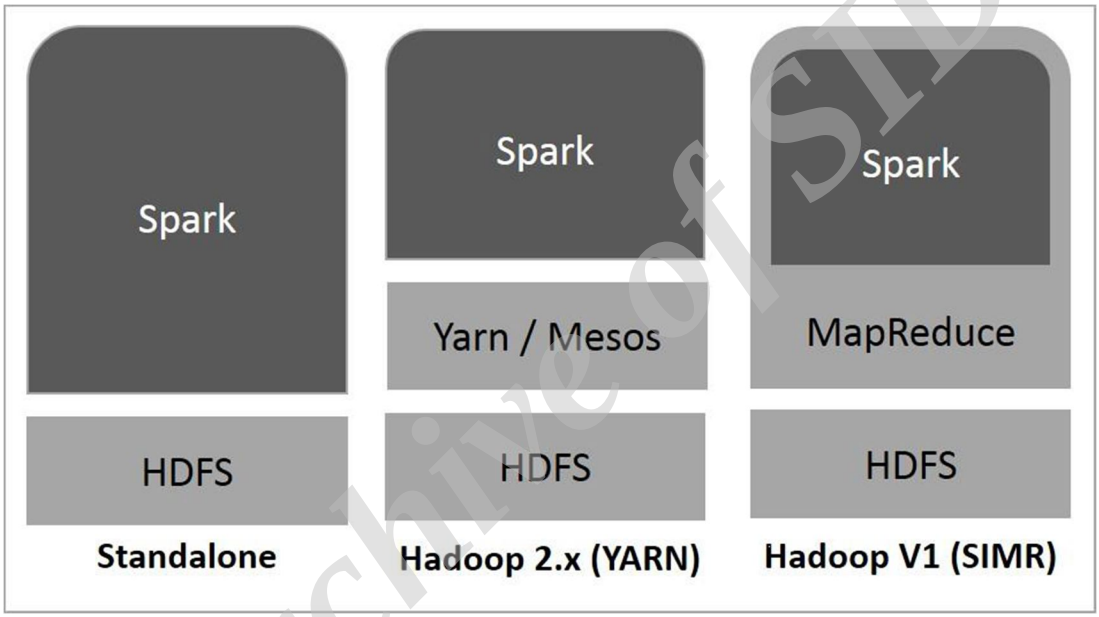
www.SID.ir

- **آنالیزهای حرفه ای**

اسپارک نه تنها از "MAP" و "Reduce" پشتیبانی می‌کند بلکه از کوئری‌های *SQL*، جریان‌های داده‌ای، ماشین یادگیرنده (Machine learning ML) و الگوریتم‌های گراف هم پشتیبانی می-کند.

- **روش‌های ساخت اسپارک روی Hadoop**

نمودار زیر سه روشی که اسپارک می‌تواند به قسمت های Hadoop وصل شود را نشان می‌دهد.



شکل 18:روش‌های ساخت اسپارک بر روی هدوپ

تفسیر سه روش گسترش اسپارک مطابق شکل 18 به این صورت است:

- **Stand alone**

Archive of SID
www.SID.ir

گسترش تنهای اسپارک به این معنی است که اسپارک جایی در بالای HDFS(Hadoop Distribued File System) مستقر می شود و به وضوح فضا برای HDFS اختصاص داده شده است.

- **Hadoop YARN**

گسترش Hadoop Yarn ساده است، اسپارک بدون هیچ نصب قبلی یا دسترسی کامل (Root) روی Yarn اجرا می شود. این حالت این امکان را می دهد که بقیه قسمت ها بالای پشته (Stackاجرا شوند.

- **Spark In MapReduce**

اسپارک در MapReduce برای اجرای کارها (Job) استفاده می‌شود. با استفاده از SIMR، کاربر می‌تواند اسپارک را شروع (Start) نموده و از پوسته (Shell) بدون هیچ دسترسی خاصی (Administrative Access) استفاده کند.

اکنون به معرفی بیشتر بخش‌های اسپارک می‌پردازیم.

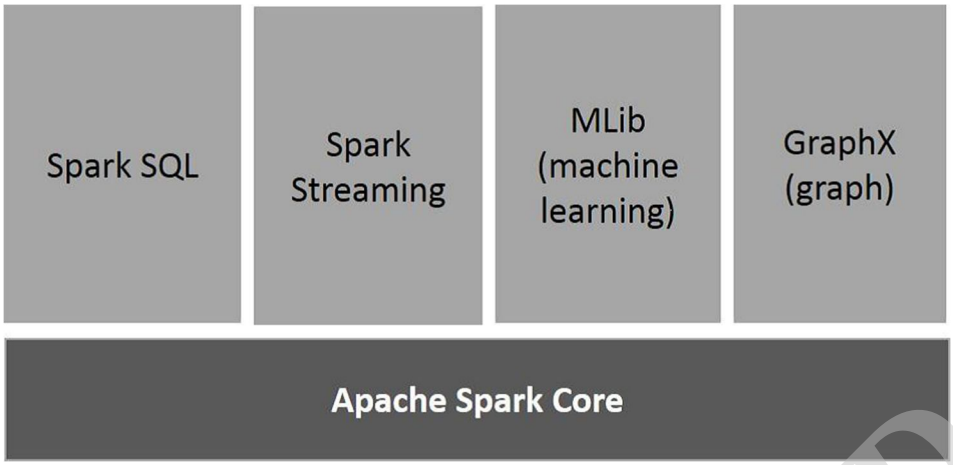
2-2-1- بخش‌های اسپارک

درتصویر زیر بخش های مختلف اسپارک را مشاهده می کنید.

- **Apache spark core**

هسته اسپارک یک موتور اجرایی عمومی اساسی برای پلتفرم اسپارک است که تمام امکانات دیگر روی آن ساخته شده است. بدین وسیله محاسبات داخل حافظه (In-Memory) فراهم می‌شود و مجموعه داده‌ها به سیستم‌های ذخیره خارجی ارجاع داده می‌شوند.

Archive of SID
www.SID.ir



شکل 19: پلتفرم اسپارک

- **Spark SQL**

بخش Spark SQL روی هسته اسپارک است که داده‌های انتزاعی جدید (New Data Abstraction) که Schema RDD نامیده می‌شوند، را معرفی می‌کند. این بخش پشتیبانی از داده‌های ساخت یافته (Structured-Data) و شبه ساخت یافته را فراهم می‌کند.

- **Spark Streaming**

در Spark Streaming از قابلیت زمان‌بندی سریع هسته های اسپارک برای فراهم کردن آنالیزهای جریانی (Streaming Analytics) استفاده می‌شود. در این بخش Ingestها داده را به گروههای کوچک (Mini-Batches) و انتقالات RDD روی آن را بصورت گروه‌های کوچک داده فراهم می‌کند.

- **MLib (Machine Learning Library)**

بخش MLib یک فریم‌ورک توزیع شده ماشین یادگیرنده روی اسپارک است زیرا معماری اسپارک بر پایه حافظه (Memory Bas) و توزیع شده است. بطوریکه MLib در اسپارک، به میزان 9 برابر سریع تر از ورژن Disk-Base را Apache Mahout می‌باشد.

- **GraphX**

بخش Graphx یک فریمورک پردازش گراف توزیع شده روی اسپارک است. که Graphx یک APIبرای محاسبه گراف Expressing فراهم می‌کند. بطوریکه توانایی مدل کردن گرافی که کابر

Archive of SID
www.SID.ir

تعریف می کند را با استفاده از Pregel Abstraction API دارد. علاوه بر این زمان اجرا را برای سطوح انتزاع بهینه می کند.

در ادامه شرایط داده‌ها در اسپارک تشریح می‌گردد.

2-2-2- دیتاست های توزیع شده ارتجاعی

مجموعه داده‌های توزیع شده ارتجاعی (RDD) از پایه‌ای‌ترین ساختار داده های اسپارک هستند. داده-های RDD کالکشن توزیع شده و غیرقابل تغییر (Immutable) اشیاء هستند. هر دیتاست (DataSet) در RDD به پارتیشن‌های منطقی تقسیم می شوند، که می‌توانند در گره‌های مختلف خوشه محاسبه شوند. هر RDD می‌تواند شامل هر شی از جاوا، پایتون، اسکالا و کلاس‌هایی که کاربر تعریف می‌کند، شوند. معمولا RDD، فقط خواندنی (Read-Only) بوده و مجموعه‌ای پارتیشن‌بندی شده از رکوردهاست. همچنین RDDها را می‌توان از طریق عملیات قطعی داده‌ها روی حافظه ثابت یا دیگر RDD ها ساخت. از طرفی RDDها کالکشن‌هایی از المان‌هایی هستند که توانایی تحمل خطا (Fault-Tolerant) را داشته باشند و به صورت موازی عمل کنند. دو راه برای ساخت RDD ها وجود دارد:

- موازی سازی کالکشن موجود دردرایو برنامه شما
- ارجاع دادن یک دیتاست به سیستم حافظه خارجی مثل سیستم اشتراک فایل، HDFS، HBaseیا هر منبع داده ای با فرمت ورودی HADoop .

اسپارک عملیات MapReduce را با استفاده از الگوی RDD (Concept)ها سریع‌تر و موثرتر می سازد. در ادامه به مکانیزم MapReduce ها و اینکه چطور کار می کنند و دلایل اینکه چرا چندان موثر و مفید نیستند، می‌پردازیم.

2-2-3- سرعت کم اشتراک داده در *MapReduce*

از MapReduce به صورت گسترده برای پردازش و تولید دیتاست‌های بزرگ با موازی‌سازی و الگوریتم های توزیع شده روی یک خوشه استفاده می شود. این امکان به کاربر اجازه می‌دهد با استفاده از آن مجموعه سطح بالای عملیات محاسبات موازی را بنویسند، بدون داشتن نگرانی درباره توزیع‌پذیری و تحمل‌پذیری خطا که احتمالا در کار وجود دارد. متاسفانه، در بیشتر فریم‌ورک‌های موجود، تنها راه برای

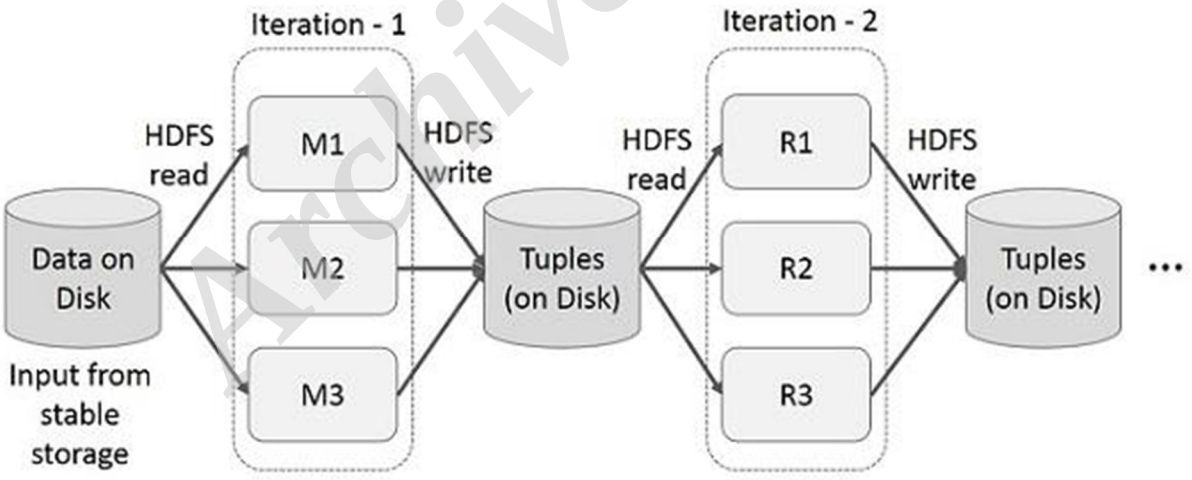
Archive of SID

www.SID.ir

استفاده مجدد از داده بین محاسبه کننده ها همین است. به عنوان مثال: بین دو MapReduce Jobs نوشتن آن داده روی یک سیستم حافظه خارجی ثابت مثلHDFS بود، با اینکه این فریم ورک تعداد زیادی انتزاع (Abstractions) برای دسترسی به منابع خوشه های محاسبه کننده فراهم کرده است، کاربران هنوز چیزهای بیشتری می خواهند. هم برنامه های تکراری و هم برنامه های تعاملی نیاز دارند تا اشتراک داده بین کارها موازی را سریع‌تر انجام دهند. اشتراک داده ها در MapReduce در موارد تکثیر (Replication)، موازی سازی (Serialization) و ورودی و خروجی دیسک (Disk IO) کند می‌باشد. درباره سیستم حافظه، اغلب برنامه های Hadoop، بالای 90% زمان را صرف عملیات خواندن / نوشتن HDFS می کنند.

2-2-4- عملیات تکراری روی *MapReduce*

استفاده مجدد از نتایج میانی بین چند محاسبه کننده در برنامه های چند طبقه (Multi-Stage) را گویند. تصویر زیر نشان می دهد که فریم ورک فعلی چگونه کارمی کند وقتی از عملیات تکراری روی MapReduce استفاده می‌کند. این خسارت قابل توجه هزینه‌هایی دارد که درحین تکثیر، ورودی و خروجی دیسک و موازی سازی سیستم را کند می‌سازد.

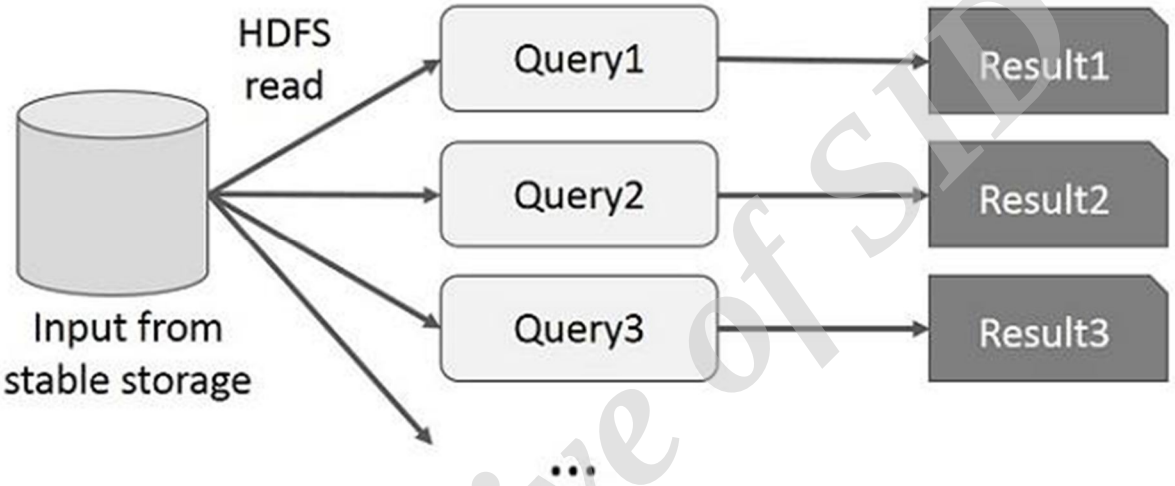


شکل 20: عملیات تکراری روی *MapReduce*

Archive of SID
www.SID.ir

2-2-5- عملیات تعاملی روی *MapReduce*

کاربران کوئری‌های موردی اختصاصی، (Ad-Hoc) را روی زیر مجموعه‌های مساوی از داده می زنند. هر کوئری باید عملیات ورودی و خروجی روی دیسک را روی حافظه ثابت ثبت کند، که همین می‌تواند زمان اجرای برنامه را افزایش دهد. تصویر زیر نحوه کار کوئری های تعاملی در فریم ورک فعلی را نشان می دهد.



شکل 21: عملیات تعاملی روی *MapReduce*

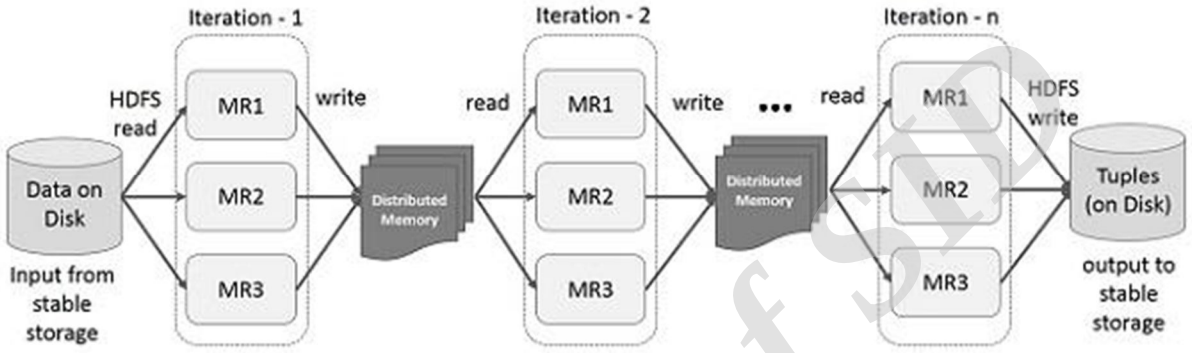
2-2-6- اشتراک داده *MapReduce* با اسپارک *RDD*

اشتراک داده هنگام تکثیر، موازی سازی و ورودی و خروجی دیسک در MapReduce کند می باشد. اغلب برنامه های Hadoop بالای 90% زمان خود را صرف عملیات خواندن / نوشتن HDFS می‌کنند. با شناسایی این مشکل، محققان یک فریم ورک ویژه به نام "اسپارک آپاچی" را گسترش دادند. ایده کلیدی اسپارک دیتاست‌های توزیع شده ارتجاعی (RDD) می‌باشد، که محاسبات پردازشی داخل حافظه را فراهم می کند. به این معنی که موقعیت حافظه را به عنوان یک شی بین کارها (Jobs) ذخیره می‌کند و این شی بین آن کارها به اشتراک گذاشته می شود. اشتراک داده داخل حافظه بین 10 تا 100 برابر سریع‌تر از اشتراک داده در شبکه و دیسک می‌باشد.

Archive of SID
www.SID.ir

2-2-7- عملیات تکراری روی اسپارک *RDD*

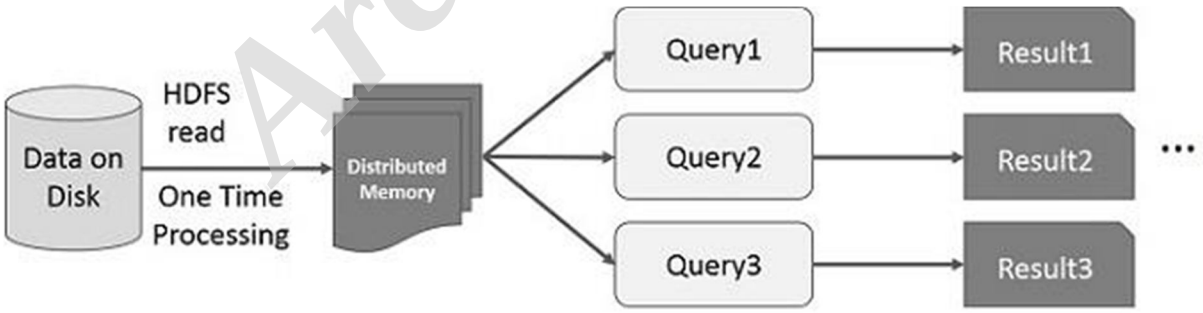
شکل زیر نحوه اجرا شدن عملیات تکراری روی اسپارک RDD را نشان می‌دهد. در این حالت نتایج میانی به جای حافظه ثابت (دیسک)، روی حافظه‌های توزیع شده (Memory) ذخیره می‌شوند که این اتفاق سیستم را سریع تر می‌کند. اگرحافظه توزیع شده (RAM) برای ذخیره نتایج میانی (حالت Job ها) کافی باشد، در آن صورت نتایج را روی دیسک ذخیره می‌کنند.



شکل 22: عملیات تکراری روی اسپارک *RDD*

2-2-8- عملیات تعاملی روی اسپارک*RDD*

شکل زیر نحوه اجرا شدن عملیات تعاملی روی اسپارک RDD را نشان می دهد. اگر کوئری های مختلف به صورت پشت سرهم روی یک مجموعه از داده اجرا شوند، این مجموعه داده برای بهتر شدن زمان اجرا می تواند در حافظه نگهداری شود.



شکل 23: عملیات تعاملی روی اسپارک *RDD*

به صورت پیش فرض ، هربار که اتفاقی روی هر RDD تغییر کرد و رخ دهد، RDDمی تواند دوباره محاسبه کند. بهرحال شاید شما RDD ثابت (Persist) در حافظه بخواهید، در این مورد اسپارک می تواند

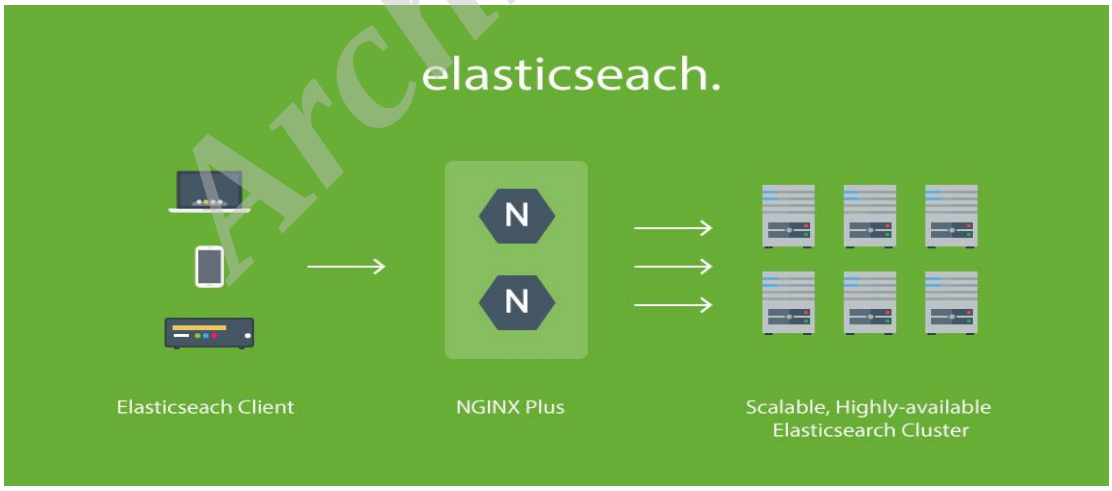
Archive of SID
www.SID.ir

برای دسترسی سریع تر المان ها را اطراف خوشه ای نگه دارد که شما دفعه بعدی روی آن کوئری می زنید. همچنین این حالت، RDDهای ثابت روی دیسک یا تکثیر شدن بین چندین گره (Node) را پشتیبانی می‌کند.

2-3- الاستیک سرچ

محصول Elasticsearch [53] یک موتور جستجو متن باز و قابل توزیع می‌باشد که با رابط کاربری وب (HTTP) و الگوی استاندارد JSON، برای انتقال داده‌ها کار می‌کند. این محصول به وسیله جاوا توسعه داده شده و به صورت گسترده و دقیق از طریق API قابل دسترسی است. این تکنولوژی می‌تواند جستجو جوی بسیار سریع و قدرتمندی را در میان داده ها انجام دهد.

این تکنولوژی برای جستجو، امکان ترکیب و استفاده از انواع مختلف داده‌های ساخت یافته، غیر ساخت یافته و ... را دارد. از طرفی همه چیز را داخل خودش index می‌کند و در کمترین زمان ممکن به درخواست‌ها پاسخ می‌دهد. مزیت مهم Elasticsearch قابلیت اجرا روی صدها سرور با داده‌های بسیار زیاد است. همچنین پایداری و انعطاف‌پذیری بالایی داشته و در هنگام بروز مشکلات سخت‌افزاری یا شبکه، خرابی را تشخیص می‌دهد و کلاستر داده‌ی خودش را حفظ می‌کند، تا قابل استفاده‌تر باشد. در این محصول برای مسائل امنیتی نیز امکان تعریف نام کاربری و رمز عبور و همچنین اعمال مجوز و ایجاد نقش در کلاستر وجود دارد.



شکل 24: معماری الاستیک سرچ

Archive of SID
www.SID.ir

2-3-1- جست و جوی سریع و دقیق در حجم زیاد داده ها

در حقیقت سیستم های مدیریت پایگاه داده برای جست و جوی تمام متن طراحی نشده اند. و در مقابل داده‌های ساخت یافته که در بیرون از دیتابیس قرار دارند به خوبی عمل نمی کنند. در مقایسه با یک سخت افزار مشابه، کوئری‌هایی که در 10 ثانیه به شما نتیجه را برمی گردانند، Elasticsearch در زمانی کمتر از 10 میلی ثانیه جواب را ارائه می‌کند.

2-3-2- ایندکس کردن مستندات در داخل ریپوزیتوری

در هنگام عملیات ایندکس کردن، Elasticsearch داده‌های خام مثل فایل‌های لاگ یا فایل‌های حاوی پیام را به مستندات داخلی تبدیل می‌کند و آنها را در داده‌های ساخت یافته اساسی شبیه شی JSON ذخیره می کند. هر سند یک مجموعه ساده از کلیدها و مقادیر وابستگی می باشد: کلید ها رشته هستند و مقادیر می‌توانند یکی از انواع داده‌های مختلفی مثل رشته، عدد، تاریخ و یا لیست باشند. اضافه کردن سند به Elasticsearch و خودکار سازی آن بسیار ساده است. کوئری را در قالب HTTP GET با یک JSON ارسال می‌شود Restfull API به سادگی آن را بازیابی و ثبت می‌کند .

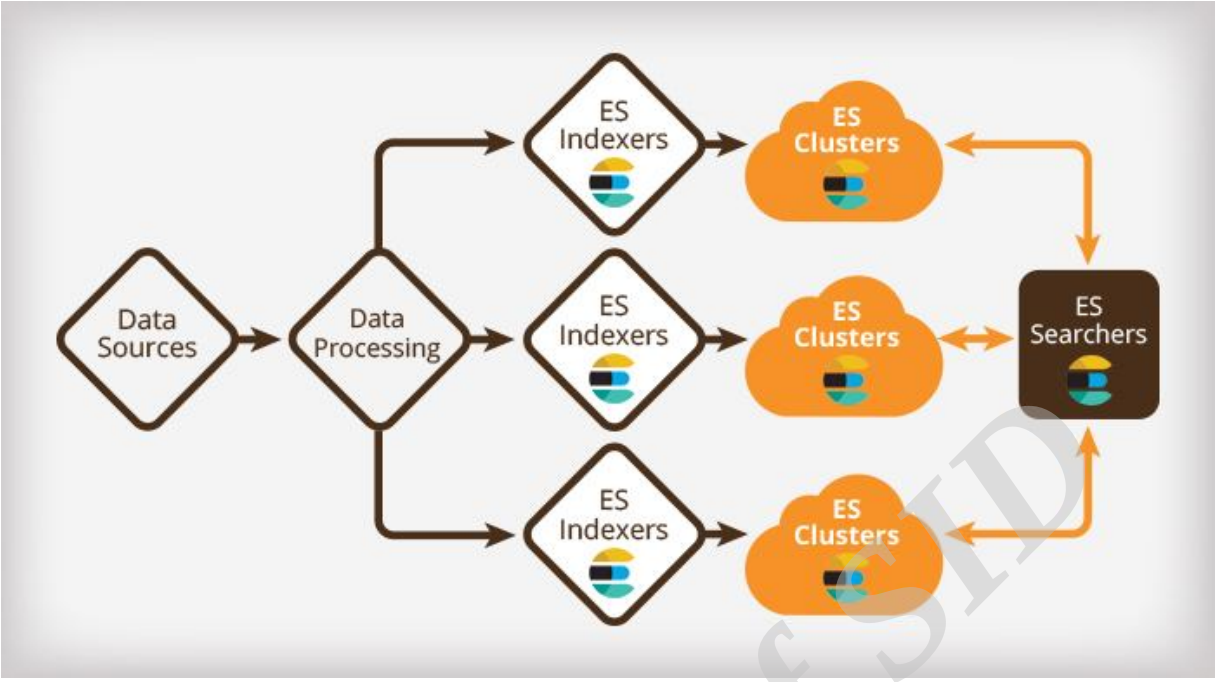
2-3-3- ذخیره سازی اسناد انحصاری

محصول Elasticsearch یا به طور مخفف ES یک پایگاه داده رابطه‌ای نیست، بنابراین مفاهیم DBMS معمولا روی آن اعمال نمی‌شوند. در واقع مهمترین مفهومی که باید از پایگاه داده‌های متداول کنار بگذارید، مبحث نرمال‌سازی می‌باشد، که ES اجازه دسترسی به زیر ساخت ها یا زیر شاخه ها را نمی دهد. بنابراین غیر نرمال سازی داده ها برای آن تا حدی ضروری است.

در محصول ES معمولا هر سند را یکبار برای هر ریپوزیتوری که در آن ساکن هست ذخیره می کند. اگر چه این دیدگاه از منظر DBMS نامتعارف است با اینحال برای Elasticsearch بهینه می باشد. جست و جوی کامل متن بسیار سریع می باشد برای اینکه این اسناد در نزدیکی متادیتاهای مرتبط با فهرست ذخیره می شوند. که این طرح به میزان قابل توجهی تعداد خواندن داده را کاهش می‌دهد.

Archive of SID

www.SID.ir



شکل 25: معماری الاستیک سرچ

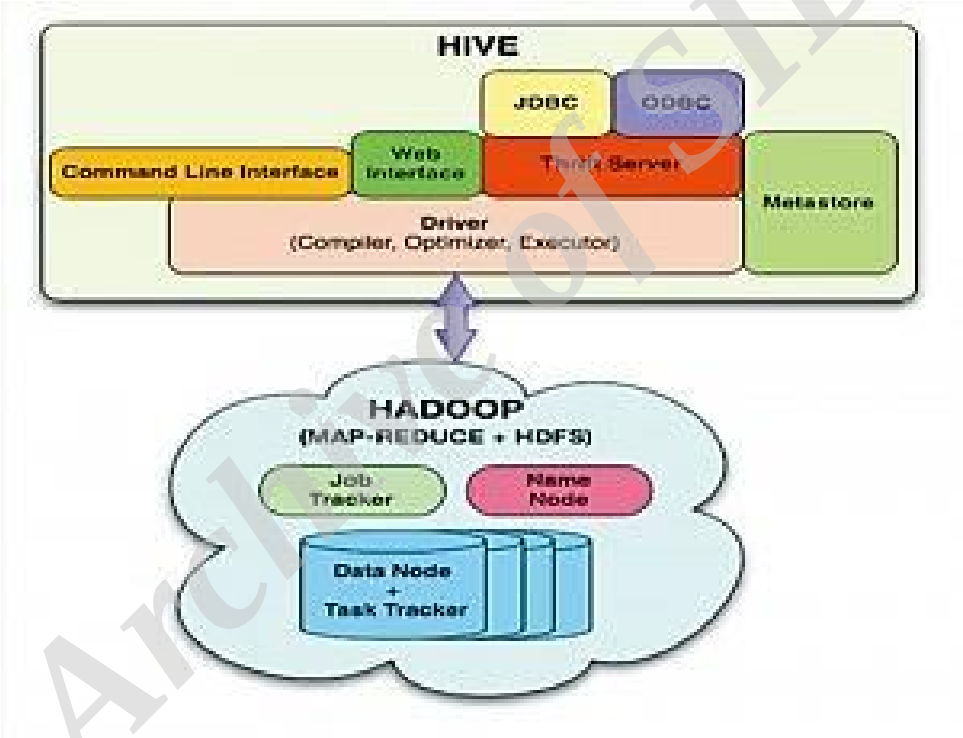
محصول Elasticsearch می‌تواند تا هزاران سرور را در بر بگیرد و داده‌ها را با هم تطبیق دهد. بسته Elasticsearch از Apache Lucene استفاده کرده و تلاش می‌کند تا تمام ویژگی‌های خود را از طریق JSON و API در دسترس قرار دهد. این موتور می‌تواند کاربر را از تعداد سندهایی که با عبارات مورد نظر آنها همخوانی دارد آگاه سازد که برای اطلاع از سند های منطبق بر کوئری‌های ثبت شده بسیار مفید است. ویژگی دیگر این موتور جستجو gateway نام دارد که وظیفه‌ی پایداری طولانی مدت ایندکس را بر عهده دارد؛ برای مثال ایندکس می‌تواند در زمان نابودی یا خرابی سرور، از gateway ریکاوری شود. بسته Elasticsearch از درخواست‌های بلادرنگ GET پشتیبانی می‌کند. که آن را به عنوان یک دیتا استور noSql مناسب می‌سازد. با این حال تراکنش های توزیع شده در آن به خوبی پاسخ داده نمی شود.

2-4- هایو

راهکار Hive[52] روش دیگری در زبان‌های سطح بالاست که در Facebook توسعه داده می‌شود. کاری که Hive انجام می‌دهد، ارائه‌ی نوعی پوسته (Shell) نزدیک به SQL است. بنابراین شما دستورالعمل‌هایی می‌نویسید که شباهت زیادی به SQL دارند و Hive نگاشت میان Schema و فایل‌ها را

Archive of SID
www.SID.ir

نگهداری می‌کند. شما 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 MapReduceباشد.



شکل 26: معماری هایو

www.SID.ir

5-2- مقایسه ابزارها و راهکارها

با توجه به اینکه هایو از حافظه موقت استفاده نمی نماید، از لحاظ سرعت اجرای پرس‌وجو نسبت به دو روش دیگر بسیار پایین‌تر می‌باشد. لذا برای پروژه تعریف شده که انباره داده برخط با امکان اجرای انواع پرس‌وجو عادی و تحلیلی می‌باشد، مناسب نیست. ضمن اینکه هایو امکان پردازش استریم را نیز ندارد.

در نتیجه مقایسه اصلی بین اسپارک و الاستیک سرچ می‌باشد. الاستیک سرچ موتور جستجویی است که به دلیل سهولت در راه اندازی و پشتیبانی انواع زبانهای برنامه نویسی بسیار محبوب و مورد استفاده می‌باشد. اما درباره پروژه مورد اجرا یک Data lake می‌باشد و لذا مناسب نیست. زیرا حجم عظیمی از داده ها به صورت لحظه ای وارد Data lake می‌شوند که باید در مدت کوتاهی مورد پردازش قرار گیرند و از طرفی اخلالی برای سرویس‌دهی به مشتریان ایجاد ننمایند. البته امکانات زیادی به این زیرساخت اضافه شده است اما تجارب عملی و مقالات علمی این ابزار را برای تشکیل Data lake مناسب نمی‌دانند. نکته دیگری که الاستیک سرچ را برای این پروژه ناماسب می‌سازد ماهیت تک منظوره الاستیک سرچ می‌باشد. به عبارت دیگر استفاده کننده از الاستیک‌سرچ باید از قبل بداند که چه نوع پرس‌وجوهایی باید اجرا گردند و براساس آن شاخص گذاری انجام دهد. اما ماهیت Data lake بنحوی است که کلیه اطلاعات برای هر گونه استفاده بعدی در آن تجمیع می‌گردد که لزوما در هنگام تشکیل اولیه Data lake مشخص نمی‌باشد.

در مجموع بررسی های انجام شده در سطح شرکتهای مختلف در هیچ کجا از این ابزار به عنوان Data lake استفاده نشده است و معمولا در لایه‌های بالاتر از دیتا از آن استفاده می‌شود. می‌توان گفت که شرکت الاستیک با ایجاد لایه هایی جهت ارتباط با هدوپ به صورت ضمنی این موضوع را تایید می‌نماید استفاده از الاستیک‌سرچ باید در لایه های بالاتر از دیتا باشد و استفاده از این ابزار به عنوان لایه دیتا مناسب نمی‌باشد.

مطالعات از منابع مختلف نشان می‌دهد که در زمینه استک مورد استفاده در شرکتهای مختلف مرجع هستند مانند https://stackshare.io نشان می‌دهد استفاده از الاستیک سرچ در لایه Utility می‌باشد و در هیچ شرکتی از این ابزار برای انباره داده یا Data lake استفاده نشده است. در شکلهای زیر استک شرکت Uber آمده است.

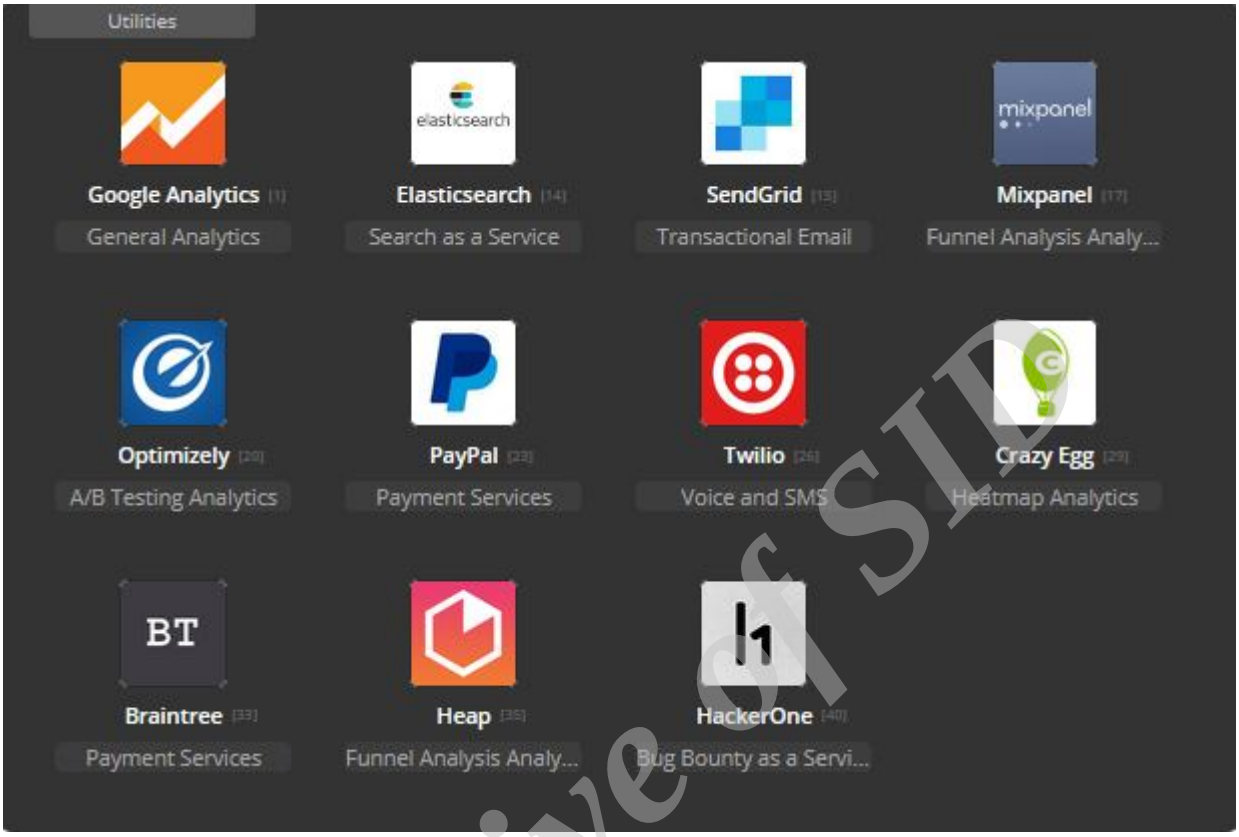
Archive of SID

www.SID.ir

Uber

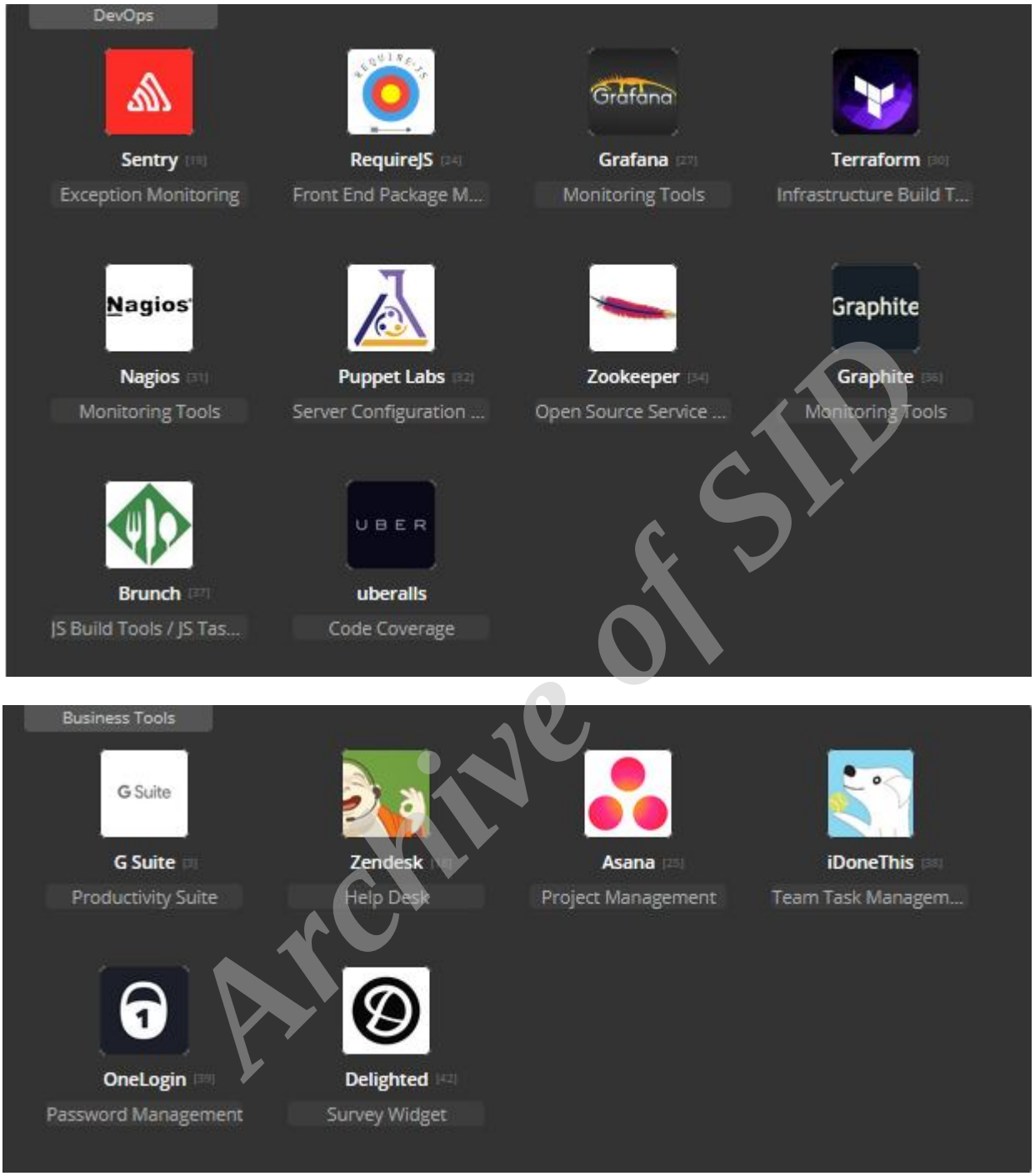
Archive of SID

www.SID.ir



Archive of SID

www.SID.ir



شکل 27: استک شرکت Uber

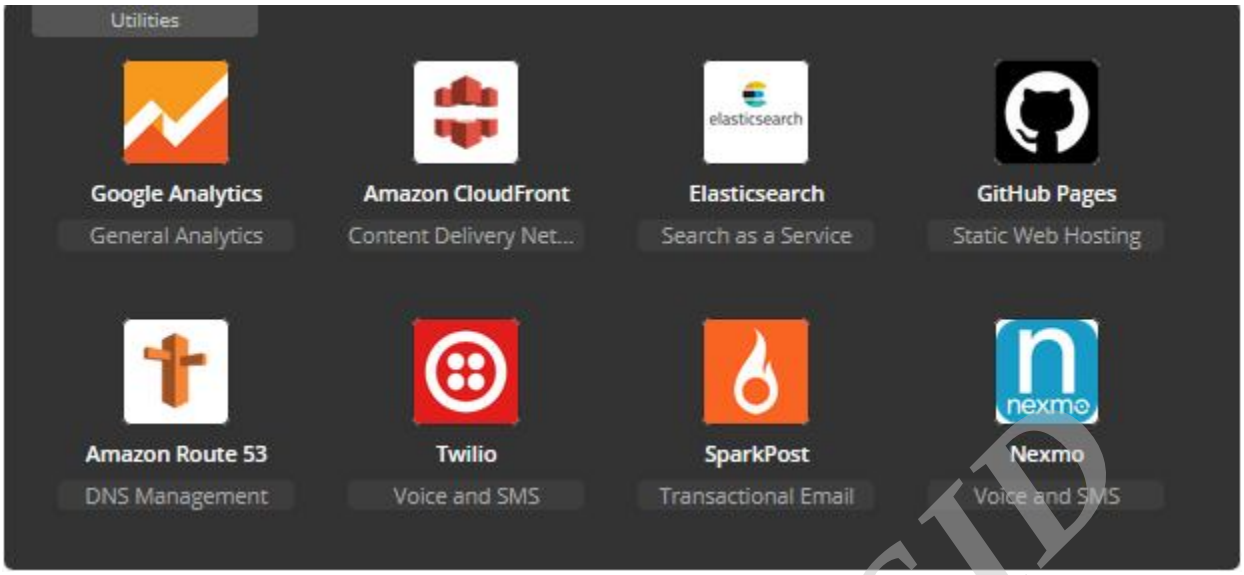
شکلهای زیر استک شرکت Dial once را نمایش می دهد

Archive of SID
www.SID.ir

Dial Once

Archive of SID

www.SID.ir

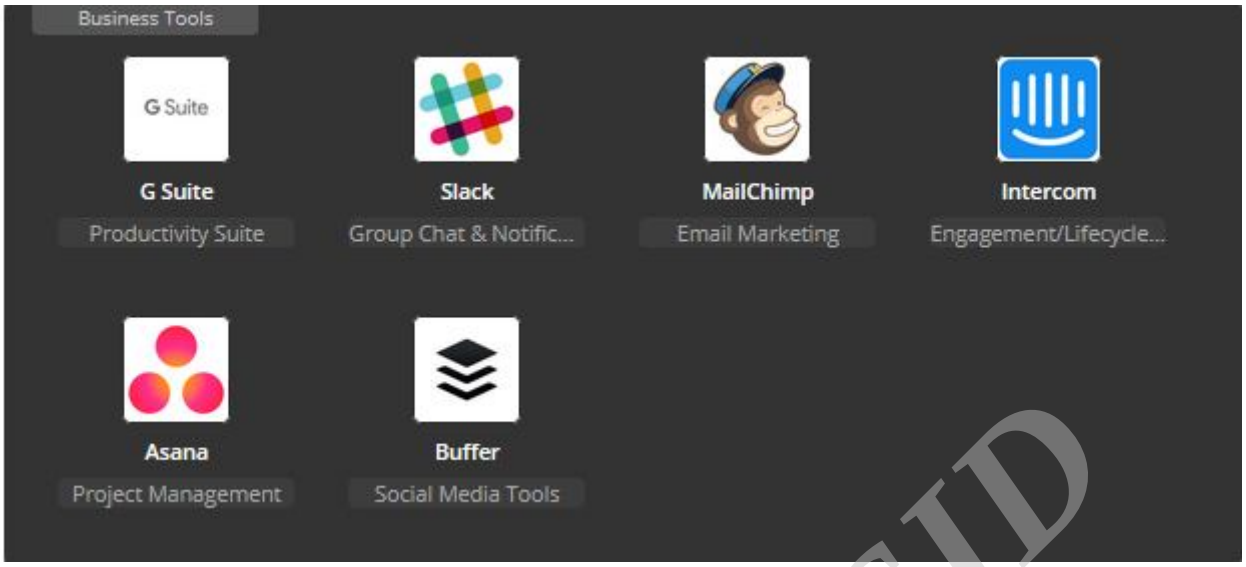


Archive of SID

www.SID.ir

Archive of SID

www.SID.ir



شکل 28: استک شرکت ***Dal once***

این موارد به صورت نمونه عنوان شد ولی در مورد حدود سه هزار شرکت بررسی شده در این سایت شرایط مشابهی وجود داشت.

همچنین با توجه به اهمیت تجاری‌سازی این محصول در حال حاضر مشتریانی که با آنها تعامل انجام شده است علاقه مند هستند که از هدوپ و اسپارک یا به طور کلی‌تر اکوسیستم هدوپ استفاده نمایند. یکی از دلایل این موضوع Community قوی‌تر اکوسیستم هدوپ می‌باشد. دلیل دیگر حجم زیاد توسعه و محصولاتی است که به صورت روزانه و در زمینه های متنوع در این اکو سیستم متولد شده یا توسعه داده می‌شود.

6-2- جمع بندی و نتیجه‌گیری

در این بخش باید اطلاعات جمع بندی شده و مقایسه ای از دلایل انتخاب اسپارک و چارچوب مد نظر برای ایجاد راهکار جدید ارائه گردد.

Archive of SID
www.SID.ir

فصل 3

راهکار پیشنهادی

Archive of SID

www.SID.ir

3-1- مقدمه

در این قسمت، روشی پیشنهاد می‌گردد که با استفاده از آن هر گره می تواند اطلاعات مربوط به خود را به طور مستقل پردازش نموده و نیازی به گره‌های دیگر نداشته باشد. این روش مبتنی بر نگاشت-کاهش می‌باشد. ابتدا پرس‌وجو ورودی به نگاشتگر ها ارسال می‌گردد. هر نگاشتگر با توجه به اطلاعات موجود بر روی خود نتایج میانی را تولید می‌نماید و به کاهنده ارسال می‌نماید و کاهنده نتایج نهایی را تولید می‌نماید. به منظور مستقل سازی هر گره یک قالب یکنواخت داده برای هر گره در نظر گرفته می‌شود . این ساختار به گونه ای است که از حافظه موقت و دایم به منظور سرعت بازیابی بالاتر استفاده می‌نماید. این ساختار با مستقل ساختن هر گره امکان اجرای همزمان و بدون وابستگی پرس‌وجو بر روی هر گره را فراهم ساخته و لذا مشکلات مربوط به پیوند قطعات داده از گره‌های مختلف، گلوگاه شبکه و استفاده ناکارآمد از سخت‌افزار را برطرف می‌نماید. از طرف دیگر با ساختار ارایه شده تغییرات در ابعاد و اندازه‌ها به راحتی امکان‌پذیر بوده و نیاز به به‌روز رسانی داده‌های قبلی نمی‌باشد.

3-2- داده ها

3-2-1- قالب داده ها

جدول 3 :تراکنش

Transaction
Transaction_ID
Date
Time
Amount
Transaction_Type
Account_ID
Card_Number
Customer_Number
City ID

جدول 4 :شهر

City
City_ID

Archive of SID

www.SID.ir

City
Province

جدول 5 :نوع تراکنش

Transaction_Type
Transaction_Type ID
Transaction_Type_Name

جدول 6 :حساب

Account
Account Number
Open Date
Account type
Branch

جدول 7 :کارت

Card
Card_Number
Expire Date
Card_Type
Branch

جدول 8 :مشتری

Customer
Customer_Number
Name
Family
Address
Sex
Job

جدول 9 :تاریخ

Date
Date

Archive of SID

www.SID.ir

Day
Month
Year

جدول 10 :زمان

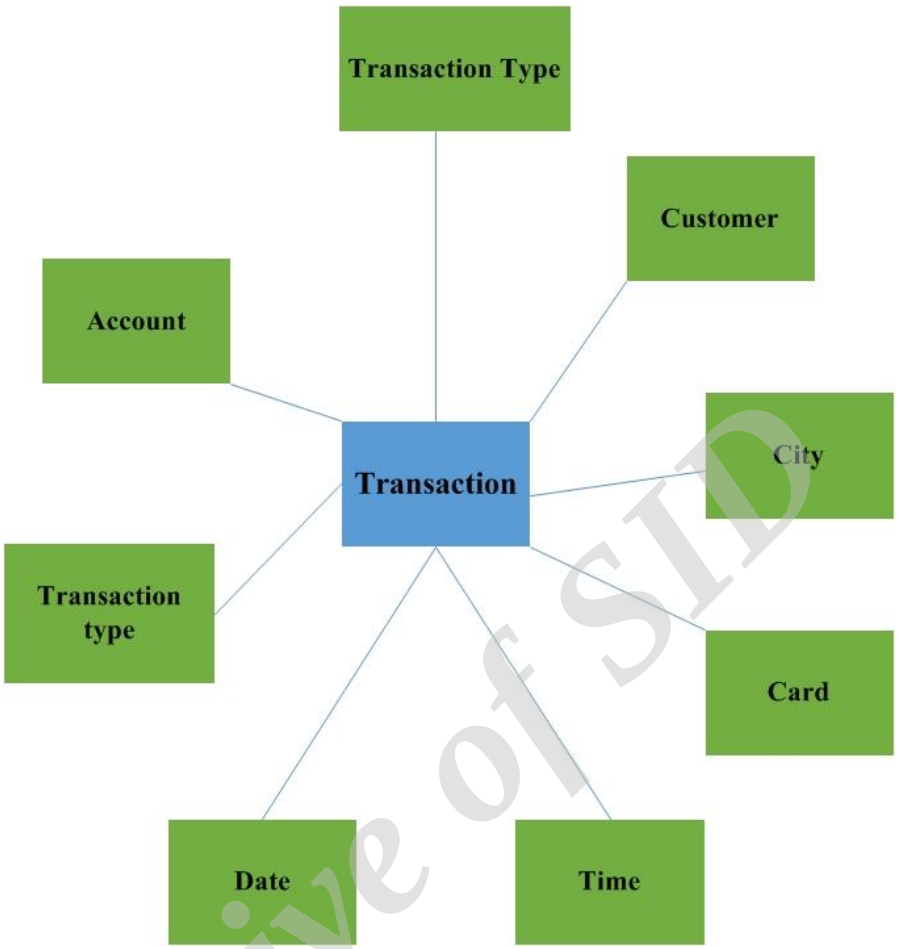
Date
Time ID
Hour
Minute
Second

3-2-2- نمودار *ER*

در ادامه نمودار *ER* مروبط نشان داده شده است.

Archive of SID

www.SID.ir



شکل 29: نمودار ER

Archive of SID

www.SID.ir

فصل 4

تحلیل و طراحی

Archive of SID

www.SID.ir

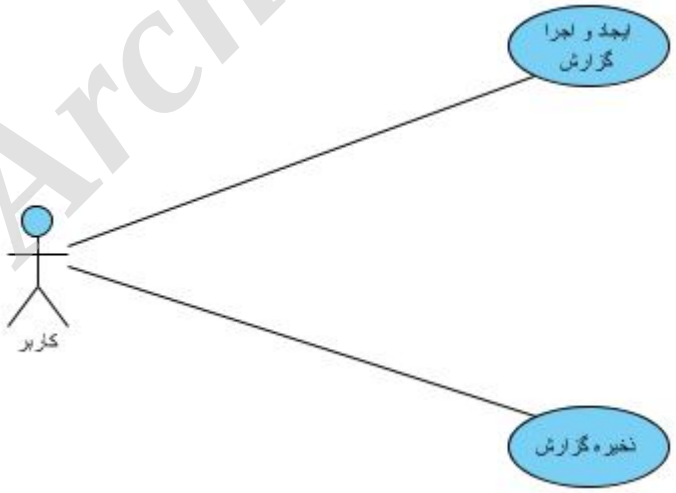
4-1- مقدمه

در این مرحله به منظور درک صحیح از نرم افزار و بیان دقیق ویژگیهای آن نمودارهای زیر آورده شده است.

- نمودار کاربرد
- نمودار کلاس
- نمودار فعالیت
- پروتوتایپ محصول

4-2- نمودار کاربرد

نمودار کاربرد مربوط به این بخش از دو قسمت ایجاد و اجرای گزارش و ذخیره گزارش تشکیل شده است.



Archive of SID

www.SID.ir

شکل 30: درخواست ایجاد و اجرا گزارش

1. نام مورد کاربرد: درخواست ایجاد و اجرا گزارش (گزارش ساز)

1-1. شرح مختصر

این مورد کاربرد به منظور ثبت ایجاد گزارش تعبیه شده است.

1-2. کنشگر اولیه

ادمین،کاربر گزارش ساز (دارای مجوز ایجاد گزارش)

تذکر: نقش یادشده بایستی در سیستم تعریف گردد.

1-3. جریان رخدادها

۱-۳-۱. جریان اصلی

1. کاربر از فرم گزارش ساز، گزینه گزارش ساز را انتخاب کرده و سپس بر روی آیکون افزودن ستون کلیک می کند.
2. سیستم فرم ساخت گزارش را نمایش می دهد.
3. کاربر پارامترهای ساخت گزارش را مقداردهی و انتخاب می نماید.
4. کاربر بر روی دکمه "اجرای گزارش " کلیک می نماید.
5. سیستم نتیجه ایجاد گزارش را در فرم *Pop-up* به کاربر نمایش می دهد.

۱-۳-۲. جریان‌های فرعی

سیستم در انجام عملیات با مشکل مواجه می شود. سیستم خطای رخ داده را به کاربر نمایش میدهد و منتظر تعامل بعدی کاربر می ماند.

کاربر بدون تغییر اطلاعات اقدام به کلیک بر روی دکمه اجرای گزارش می کند. در این صورت تغییری در سیستم رخ نمی دهد.

Archive of SID

www.SID.ir

1-4. نیازمندی های خاص

1-5. پیش شرایط

کاربر از دسترسی لازم برخوردار باشد.
دسترسی های مورد نیاز به ازای کنشگرهای اولیه می باشد.

1-6. پس شرایط

1-7. موارد کاربرد مرتبط

1-8. قوانین کاری

1- در صورت انتخاب فیلد های " تعداد" "جمع" " کمینه" " بیشینه" مقدار فیلد عنوان ، مقدار تراکنش نمایش داده می شود.
2- در صورتی که شرط در نظر گرفته نشود تنها یک رکورد نمایش داده می شود.
3- در صورتی که شرط برای تابع در نظر گرفته شود، نتیجه گزارش با توجه به شرط مورد نظر نمایش داده می شود.
4- در صورت انتخاب فیلد های " دسته بندی " " – " عنوان، شامل داده های تراکنش های سوئیچ می باشد.
5- در صورتی که شرط در نظر گرفته نشود همه رکورد های مربوط به عنوان انتخاب شده نمایش داده می‌شود.
6- در صورتی که شرط در نظر گرفته شود، نتیجه گزارش با توجه به شرط مورد نظر نمایش داده می شود.

1-9. شرایط مرتبط با واسط کاربری

1- فیلد تابع گزارش شامل موارد زیر می باشد:
- تعداد

Archive of SID

www.SID.ir

- دسته بندی با
- جمع
- کمینه
- بیشینه
- -

این فیلد به صورت انتخابی و انتخاب آن به صورت اجباری می باشد.

2- فیلد عنوان شامل موارد زیر می باشد:

- با نتخاب تابع های تعداد ، جمع ، کمینه و بیشینه در قسمت عنوان مبلغ تراکنش نمایش داده می شود.
- با انتخاب گزینه های دسته بندی با و – در قسمت عنوان تمامی مقادیر از جدول ----- نمایش داده می شود.

این فیلد به صورت انتخابی و انتخاب آن به صورت اجباری می باشد.

3- فیلد شرط شامل موارد زیر می باشد:

- مابین
- بزرگتر از
- کوچکتر از
- برابر با
- مخالف با

این فیلد به صورت انتخابی می باشد.

4- فیلد مقدار به صورت عددی و ورود داده توسط کاربر می باشد.

این فیلد به صورت اختیاری می باشد.

5- با کلیک بر روی آیکون تمامی پارامترهای ورودی تعیین شده توسط کاربر حذف و تمامی فیلد ها به روز رسانی می گردد.

Archive of SID

www.SID.ir

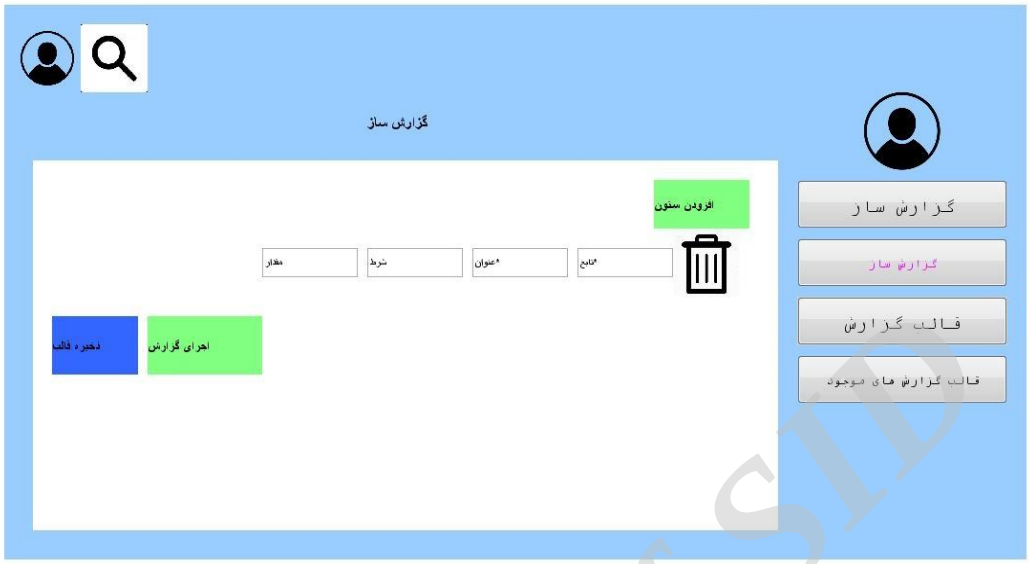
1-10. کنترل اقلام اطلاعاتی

- کاربر ایجاد کننده: برابر با عنوان کاربر ایجاد کننده درخواست
- پارامترهای ورودی: برابر با رشته ای شامل پارامترهای ورودی گزارش

1-11. واسط کاربری

Archive of SID

www.SID.ir



شکل 31: گزارش ساز

2. نام مورد کاربرد: درخواست ذخیره قالب گزارش

2-1. شرح مختصر

این مورد کاربرد به منظور ذخیره قالب گزارش تعبیه شده است.

2-2. کنشگر اولیه

ادمین،کاربر گزارش ساز

تذکر: نقش فوق به سیستم اضافه شود.

2-3. جریان رخدادها

۲-۳-۱. جریان اصلی

1. کاربر از فرم گزارش ساز، گزارش را انتخاب کرده و سپس بر روی آیکون افزودن ستون کلیک می نماید و پارامتر های ورودی را مقدار دهی می نماید .

Archive of SID
www.SID.ir

2. کاربر بر روی آیکون ذخیره قالب کلیک می نماید.
3. سیستم فرم عنوان گزارش را به کاربر نمایش می دهد.
4. کاربر عنوان گزارش را وارد و بر روی دکمه ذخیره کلیک می نماید.
5. سیستم مقادیر انتخاب شده توسط کاربر را (قالب گزارش) را ذخیره می نماید.
6. قالب ذخیره شده در زیر بخش قالب گزارش نمایش داده می شود .

۲-۳-۲. جریان‌های فرعی

سیستم در انجام عملیات با مشکل مواجه می شود. سیستم خطای رخ داده را به کاربر نمایش میدهد و منتظر تعامل بعدی کاربر می ماند.

2-4. نیازمندی های خاص

2-5. پیش شرایط

کاربر از دسترسی لازم برخوردار باشد.

دسترسی های مورد نیاز به ازای کنشگرهای اولیه می باشد.

2-6. پس شرایط

2-7. موارد کاربرد مرتبط

2-8. قوانین کاری

2-9. شرایط مرتبط با واسط کاربری

قالب ذخیره شده در زیر بخش قالب گزارش با فرمت اکسل نمایش داده می شود .

2-10. کنترل اقلام اطلاعاتی

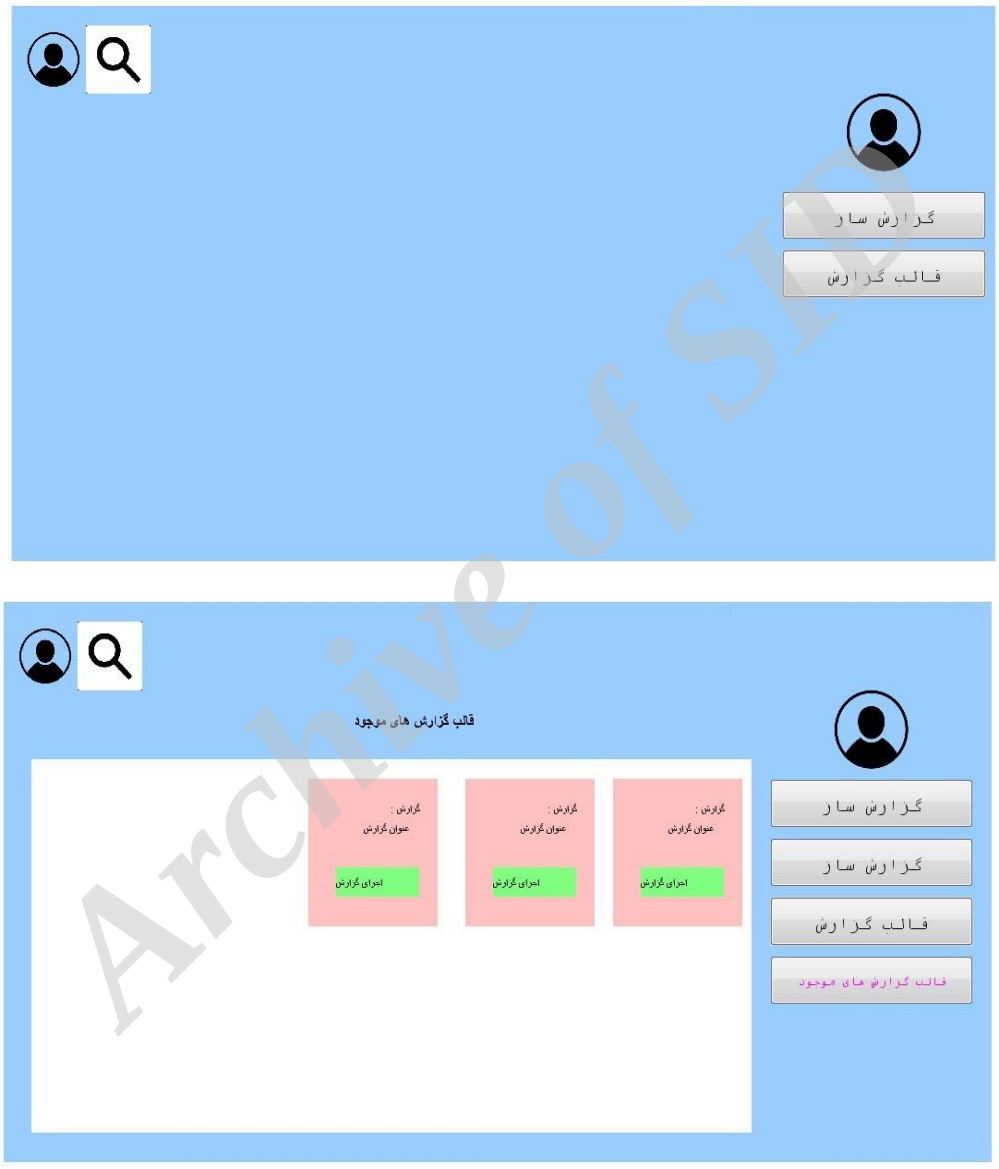
- کاربر ایجاد کننده: برابر با عنوان کاربر ایجاد کننده درخواست

Archive of SID

www.SID.ir

- پارامترهای ورودی: برابر با رشته ای شامل پارامترهای ورودی گزارش

2-11. واسط کاربری

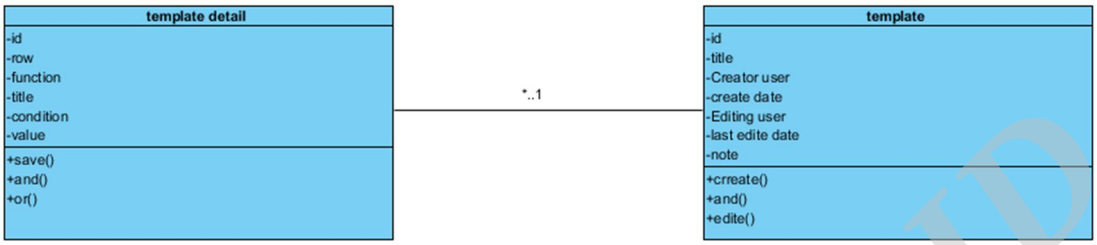


شکل 32: بارگذاری گزارشات

Archive of SID

www.SID.ir

3. نمودار کلاس

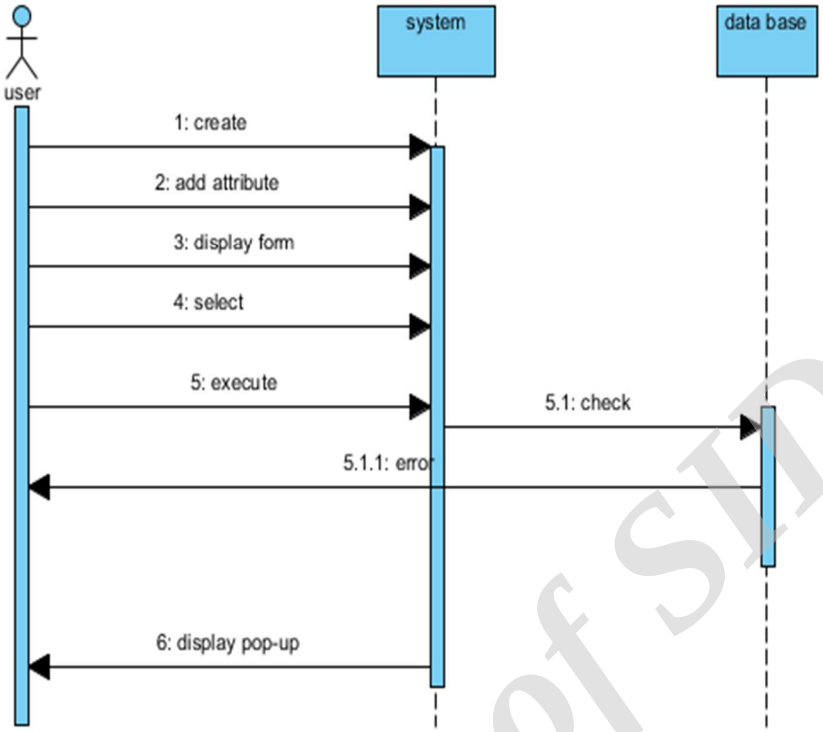


شکل 33: نمودار کلاس

4. نمودار توالی (*sequence*)

Archive of SID

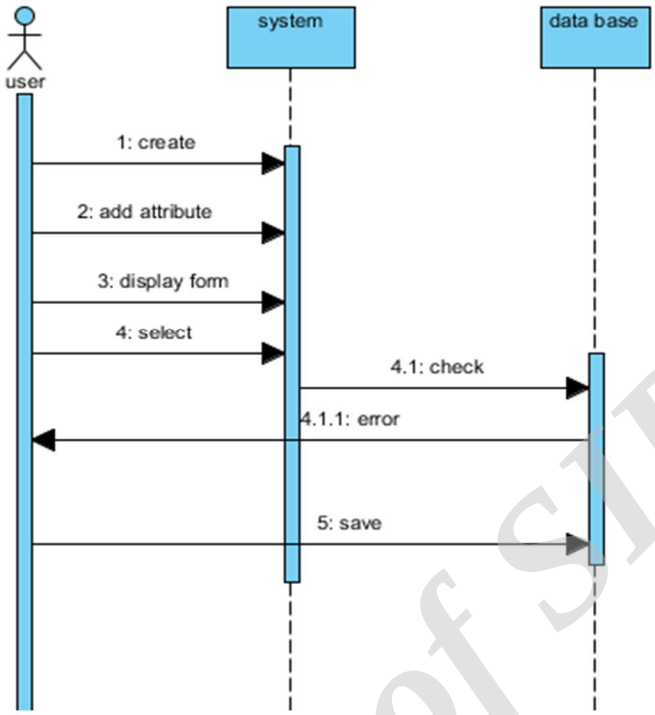
www.SID.ir



شکل 34: نمودار توالی ساخت گزارش

Archive of SID

www.SID.ir

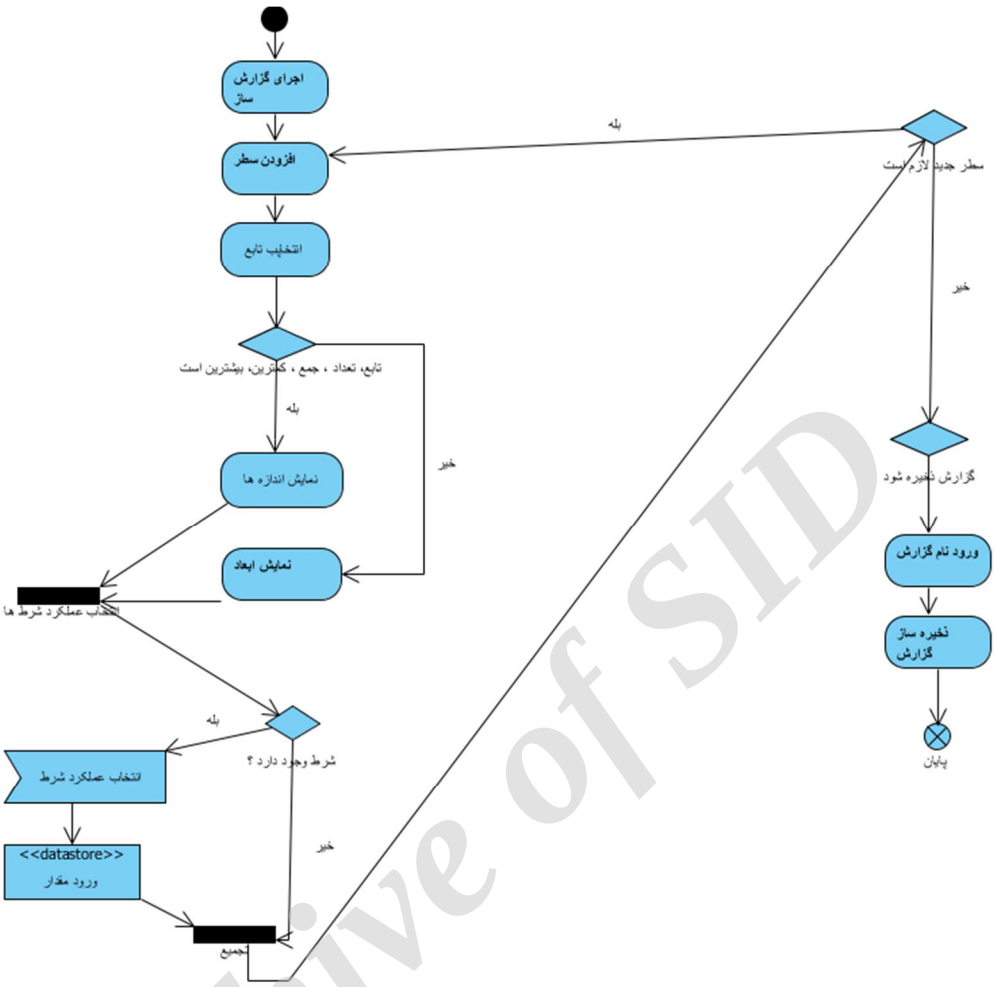


شکل 35: نمودار توالی بارگذاری گزارش

5. نمودار فعالیت

Archive of SID

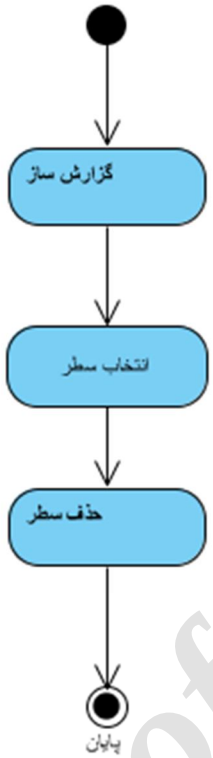
www.SID.ir



شکل 36: نمودار فعالیت ساخت گزارش

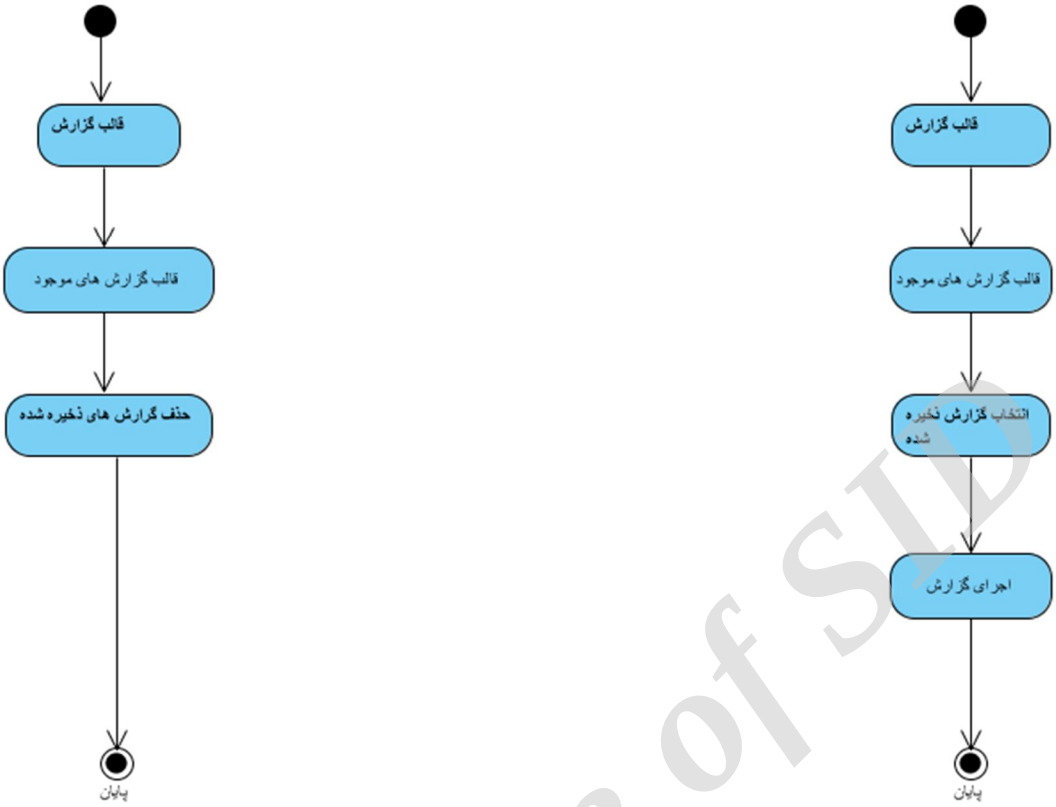
Archive of SID

www.SID.ir



Archive of SID

www.SID.ir



شکل 37: نمودار فعالیت بارگذاری گزارش

Archive of SID

www.SID.ir

فصل 5

نتایج اجرای راهکار

Archive of SID

www.SID.ir

5-1- مقدمه

در این بخش از گزارش نتایج عملی پیاده سازی به روش پیشنهادی آورده شده است

5-2- محیط عملیاتی

داده هایی که سیستم بر روی آن پیاده سازی و تست گردید اطلاعات سوییچ مربوط به یک بانک است و حجم اطلاعات روزانه 4 میلیون رکورد است. اطلاعات مربوط به یک سال و نیم بانک در انباره داده قرار گرفت و حدود 2 میلیارد رکورد مورد آزمون قرار گرفت

5-3- گزارشات مورد بررسی

- میزان کارکرد کانالهای پرداخت به ازای ماههای مختلف
- میزان کارکرد کانالهای پرداخت به ازای استانهای مختلف
- میزان کارمزد پرداختی به ازای استانهای مختلف
- تعداد خرید شارژ تلفن همراه به تفکیک اپراتور
- میزان پرداخت قبض به ازای صادرکننده
- تعداد کد پاسخ رمز نامعتبر است به ازای شهرهای مختلف در ماههای گوناگون
- میزان کارکرد کیوسک به ازای انواع تراکش
- حجم تراکنشهای انجام شده در فاصله ساعات 12 شب تا 6 صبح به ازای استانهای مختلف
- حجم تراکنشهای انجام شده در فاصله ساعات 12 شب تا 6 صبح به ازای مناطق پرخطر
- حجم تراکنشهای انجام شده در فاصله ساعات 12 شب تا 6 صبح به ازای مناطق آزاد
- حجم ریالی و تعدادی تراکنشهای خرید شارژ به ازای کانالهای پرداخت

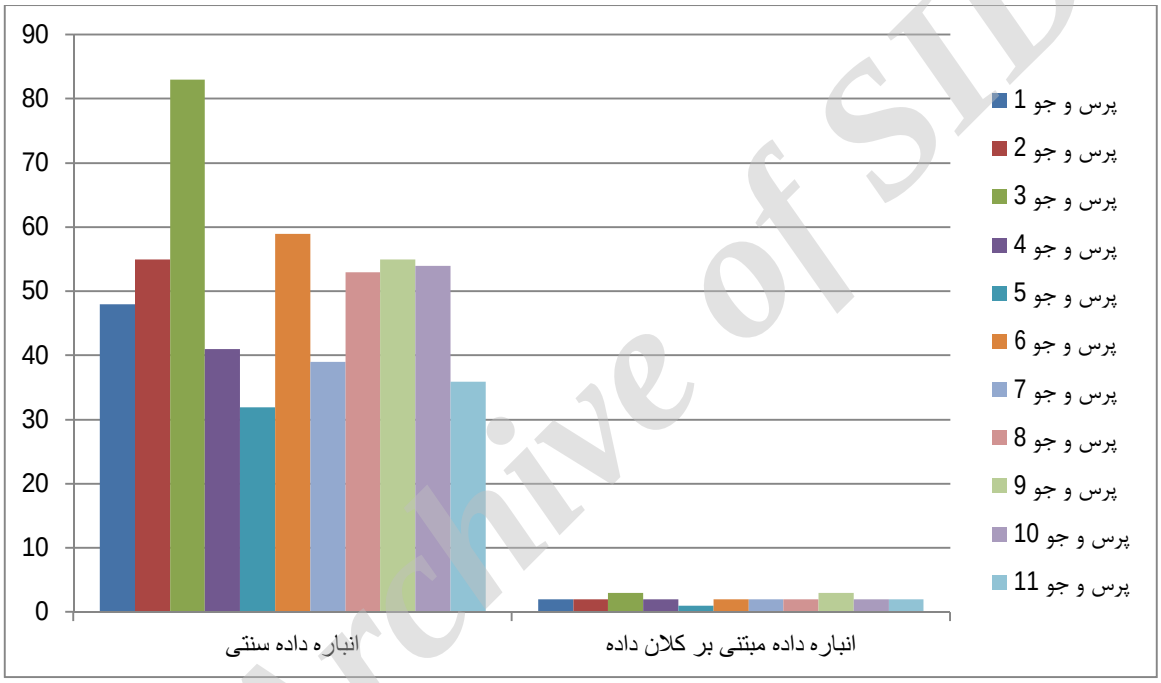
Archive of SID

www.SID.ir

- حجم ریالی تراکنش دریافتی از سایر بانکها به تفکیک بانک
- میزان کارمزد دریافتی از سایر بانکها به تفکیک بانک

4-5- نتایج

روش پیاده سازی شده در مقایسه با راهکار سنتی انباره داده که بوسیله اوراکل پیاده سازی شده بود نتایج بسیار بهتری را در محیط عملیاتی به دست آورد. به صورت کلی می توان گفت که زمان اجرای پرس وجو ها از چند ده دقیقه به کمتر از سه دقیقه کاهش پیدا کرد.



شکل 38: سرعت اجرای پرس وجو

Archive of SID

www.SID.ir

Archive of SID

www.SID.ir

مراجع

[۱] Codd, Edgar F. "A relational model of data for large shared data banks." Communications of the ACM ۱۳,٦ (۱۹۷۰): ۳۷۷-۳۸۷.

[۲] Krishnan, Krish. Data Warehousing in the age of Big Data. Newnes, ۲۰۱۳.

[۳] Berman, Jules J. Principles of big data: preparing, sharing, and analyzing complex information. Newnes, ۲۰۱۳.

[٤] Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." Communications of the ACM ٥١.(۲۰۰۸): ۱۰۷-۱۱۳.

[٥] Osman, Amr, Mohamed El-Refaey, and Ayman Elnaggar. "Towards Real-Time Analytics in the Cloud." Services (SERVICES), ۲۰۱۳ IEEE Ninth World Congress on. IEEE, ۲۰۱۳.

[٦] ZHANG, Guigang, et al. "Mapreduce++: Efficient processing of mapreduce jobs in the cloud." Journal of Computational Information Systems ۸,۱٤ (۲۰۱۲): ٥٧٥٧-٥٧٦٤.

[۷] Polo, Jorda, et al. "Performance management of mapreduce applications." (۲۰۰۹).

[۸] Herodotou, Herodotos, Fei Dong, and Shivnath Babu. "Mapreduce programming and cost-based optimization? crossing this chasm with starfish." Proceedings of the VLDB Endowment ٤,۱۲ (۲۰۱۱): ۱٤٤٦-۱٤٤۹.

[۹] Herodotou, Herodotos, et al. "Starfish: A Self-tuning System for Big Data Analytics." CIDR. Vol. ۱۱. ۲۰۱۱.

[۱۰] Thusoo, Ashish, et al. "Hive-a petabyte scale data warehouse using hadoop." Data Engineering (ICDE), ۲۰۱۰ IEEE ۲٦th International Conference on. IEEE, ۲۰۱۰.

[۱۱Olston, Christopher, et al. "Pig latin: a not-so-foreign language for data processing." Proceedings of the ۲۰۰۸ ACM SIGMOD international conference on Management of data. ACM, ۲۰۰۸.

[۱۲] Khetrapal, Ankur, and Vinay Ganesh. "HBase and Hypertable for large scale distributed storage systems." Dept. of Computer Science, Purdue University (۲۰۰٦).

[۱۳] Lakshman, Avinash, and Prashant Malik. "Cassandra: structured storage system on a p۲p network." Proceedings of the ۲۸th ACM symposium on Principles of distributed computing. ACM, ۲۰۰۹.

[۱٤] Chang, Fay, et al. "Bigtable: A distributed storage system for structured data." ACM Transactions on Computer Systems (TOCS) ۲٦,۲ (۲۰۰۸): ٤.

[۱٥] Ekanayake, Jaliya, et al. "Twister: a runtime for iterative mapreduce." Proceedings of the ۱۹th ACM International Symposium on High Performance Distributed Computing. ACM, ۲۰۱۰.

[۱٦] Bu, Yingyi, et al. "HaLoop: efficient iterative data processing on large clusters." Proceedings of the VLDB Endowment ۳,۱-۲ (۲۰۱۰): ۲۸٥-۲۹٦.

[۱۷] Olston, Christopher, et al. "Nova: continuous pig/hadoop workflows." Proceedings of the ۲۰۱۱ ACM SIGMOD International Conference on Management of data. ACM, ۲۰۱۱.

[۱۸] Zaharia, Matei, et al. "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing." Proceedings of the ۹th USENIX conference on Networked Systems Design and Implementation. USENIX Association, ۲۰۱۲.

[۱۹] Ding, Linlin, et al. "Commapreduce: an improvement of mapreduce with lightweight communication mechanisms." Data & Knowledge Engineering ۸۸ (۲۰۱۳): ۲۲٤-۲٤۷.

[۲۰] Mohamed, Hisham, and Stéphane Marchand-Maillet. "MRO-MPI: MapReduce overlapping using MPI and an optimized data exchange policy." Parallel Computing ۳۹,۱۲ (۲۰۱۳): ۸٥۱-۸٦٦.

[۲۱] J Boulon, Jerome, et al. "Chukwa, a large-scale monitoring system." Proceedings of CCA. Vol. ۸. ۲۰۰۸.

Archive of SID

www.SID.ir

[۲۲] Neumeyer, Leonardo, et al. "S٤: Distributed stream computing platform." Data Mining Workshops (ICDMW), ۲۰۱۰ IEEE International Conference on. IEEE, ۲۰۱۰.
[۲۳] Nathan, M., X. James, and J. Jason. "Storm: Distributed real-time computation system." (۲۰۱۲).
[۲٤] Chu, Cheng, et al. "Map-reduce for machine learning on multicore." Advances in neural information processing systems ۱۹ (۲۰۰۷): ۲۸۱.
[۲٥] Venkataraman, Shivaram, et al. "Presto: Complex and continuous analytics with distributed arrays." (۲۰۱۱).
[۲٦] Inmon, W. H. "WHAT IS AData WAREHOUSE?." (۲۰۰۰).
[۲۷] Han, Jiawei, and Micheline Kamber. Data Mining, Southeast Asia Edition: Concepts and Techniques. Morgan kaufmann, ۲۰۰٦, Pages ۱۱٤-۱۱۸.
[۲۸] Lee, Rubao, et al. "Ysmart: Yet another sql-to-mapreduce translator." *Distributed Computing Systems (ICDCS), ۲۰۱۱ ۳۱st International Conference on*. IEEE, ۲۰۱۱.
[۲۹] Chen, Songting. "Cheetah: a high performance, custom data warehouse on top of MapReduce." Proceedings of the VLDB Endowment ۳,۱-۲ (۲۰۱۰): ۱٤٥۹-۱٤٦۸.
[۳۰] He, Yongqiang, et al. "RCFile: A fast and space-efficient data placement structure in MapReduce-based warehouse systems." Data Engineering (ICDE), ۲۰۱۱ IEEE ۲۷th International Conference on. IEEE, ۲۰۱۱.
[۳۱] Liu, Jialin, and Yong Chen. "Improving data analysis performance for high-performance computing with integrating statistical metadata in scientific datasets." High Performance Computing, Networking, Storage and Analysis (SCC), ۲۰۱۲ SC Companion: IEEE, ۲۰۱۲.
[۳۲] Abouzeid, Azza, et al. "HadoopDB: an architectural hybrid of MapReduce and سمپاد technologies for analytical workloads." Proceedings of the VLDB Endowment ۲,۱ (۲۰۰۹): ۹۲۲-۹۳۳.
[۳۳] Yang, Christopher, et al. "Osprey: Implementing MapReduce-style fault tolerance in a shared-nothing distributed database." Data Engineering (ICDE), ۲۰۱۰ IEEE ۲٦th International Conference on. IEEE, ۲۰۱۰.
[۳٤] Kaldewey, Tim, Eugene J. Shekita, and Sandeep Tata. "Clydesdale: structured data processing on MapReduce." *Proceedings of the ۱٥th International Conference on Extending Database Technology*. ACM, ۲۰۱۲.
[۳٥] Lin, Yuting, et al. "Llama: leveraging columnar storage for scalable join processing in the mapreduce framework." *Proceedings of the ۲۰۱۱ ACM SIGMOD International Conference on Management of data*. ACM, ۲۰۱۱.
[۳٦] Engle, Cliff, et al. "Shark: fast data analysis using coarse-grained distributed memory." *Proceedings of the ۲۰۱۲ ACM SIGMOD International Conference on Management of Data*. ACM, ۲۰۱۲.
[۳۷] Armbrust, Michael, et al. "Spark SQL: Relational data processing in Spark." *Proceedings of the ۲۰۱٥ ACM SIGMOD International Conference on Management of Data*. ACM, ۲۰۱٥.
[۳۸] Zaharia, Matei, et al. "Spark: cluster computing with working sets." *Proceedings of the ۲nd USENIX conference on Hot topics in cloud computing*. Vol. ۱۰. ۲۰۱۰.
[۳۹] Eltabakh, Mohamed Y., et al. "CoHadoop: flexible data placement and its exploitation in Hadoop." *Proceedings of the VLDB Endowment* ٤,۹ (۲۰۱۱): ٥۷٥-٥۸٥.
[٤۰] Liu, Huan, and Dan Orban. "Gridbatch: Cloud computing for large-scale data-intensive batch applications." *Cluster Computing and the Grid, ۲۰۰۸. CCGRID'۰۸. ۸th IEEE International Symposium on*. IEEE, ۲۰۰۸.
[٤۱] Dittrich, Jens, et al. "Hadoop++: making a yellow elephant run like a cheetah (without it even noticing)." *Proceedings of the VLDB Endowment* ۳,۱-۲ (۲۰۱۰): ٥۱٥-٥۲۹.
[٤۲] Gorton, Ian, and John Klein. "Distribution, data, deployment: Software architecture convergence in big data systems." IEEE Software ۳۲,۳ (۲۰۱٥): ۷۸-۸٥.

Archive of SID
www.SID.ir

[۴۳] Padhye, Vinit, and Anand Tripathi. "Scalable transaction management with snapshot isolation for NoSQL data storage systems." IEEE Transactions on Services Computing ۸,۱ (۲۰۱۵): ۱۲۱-۱۳۵.
[۴۴] Wang, Teng, et al. "EA۲S۲: An Efficient Application-Aware Storage System for Big Data Processing in Heterogeneous Clusters." Computer Communication and Networks (ICCCN), ۲۰۱۷ ۲۶th International Conference on. IEEE, ۲۰۱۷.
[۴۵] Rupprecht, Lukas, et al. "SwiftAnalytics: Optimizing Object Storage for Big Data Analytics." Cloud Engineering (IC۲E), ۲۰۱۷ IEEE International Conference on. IEEE, ۲۰۱۷.
[۴۶] Tang, Zhuo, et al. "An optimized MapReduce workflow scheduling algorithm for heterogeneous computing." The Journal of Supercomputing ۷۲,۶ (۲۰۱۶): ۲۰۵۹-۲۰۷۹.
[۴۷] Yao, Yi, et al. "Self-adjusting slot configurations for homogeneous and heterogeneous hadoop clusters." IEEE Transactions on Cloud Computing ۵,۲ (۲۰۱۷): ۳۴۴-۳۵۷.
[۴۸] Wang, Weina, et al. "Maptask scheduling in mapreduce with data locality: Throughput and heavy-traffic optimality." IEEE/ACM Transactions on Networking ۲۴,۱ (۲۰۱۶): ۱۹۰-۲۰۳.
[۴۹] Wang, Shaoqi, et al. "Network-Adaptive Scheduling of Data-Intensive Parallel Jobs with Dependencies in Clusters." Autonomic Computing (ICAC), ۲۰۱۷ IEEE International Conference on. IEEE, ۲۰۱۷.
[۵۰] Harbi, Razen, et al. "Accelerating SPARQL queries by exploiting hash-based locality and adaptive partitioning." The VLDB Journal ۲۵,۳ (۲۰۱۶): ۳۵۵-۳۸۰.
[۵۱] http://spark.apache.org/
[۵۲] https://hive.apache.org/
[۵۳] https://www.elastic.co/

Archive of SID

www.SID.ir

Abstract

Creating an Online data warehouse based on big data is the main purpose of this proposal. This proposal creates situation that a query can be executed in online and analytical(historical) modes simultaneously.

This system is implemneted by Spark , Python and Java.

This proposal is according to market requirements and creates a system that executes queries more rapidly. The propsed system decreases query execution time from a few hours to a few minutus.

Keywords: *Big data , MapReduce, Data warehouse, Data localization*

Archive of SID

www.SID.ir

Information and Communication Technology Research Center

Title

A Method to Solve Data Warehouse Problem in Big Data

*Code:***4487**

By:

Mohammad Taghipour

August, **2018**

Archive of SID

www.SID.ir