پایگاه داده های NoSQL

دیتابیس های NoSQL، که گاهی تحت عنوان Not Only SQL هم شناخته می شوند، روشی برای مدیریت داده ها و طراحی پایگاه داده است که برای مجموعه زیادی از داده های نامتمرکز به کار گرفته می شوند و این در حالی است که NoSQL شامل طیف وسیعی از تکنولوژی ها و معماری ها می شود که این گوناگونی به خاطر نیاز به روشهایی برای حل مشکلات مقیاس پذیری و بازدهی BIG Data است که در طراحی دیتابیس های به اصطلاح Relational (رابطه ای) همچون MySQL این موارد در نظر گرفته نشده است.

اساساً معماری NoSQL بیشتر برای کسب و کارهایی مناسب است که نیاز به دسترسی و آنالیز حجم زیادی از داده های به اصطلاح Unstructured (بدون ساختار) دارند و یا داده های آن ها به صورت ریموت و روی چندین سرور مجازی بر بستر سرور های ابری قرار دارد. در واقع، ایده طراحی این تکنولوژی ابتدا به ساکن توسط شرکت های پیشرو در حوزه اینترنت نظیر فیسبوک، گوگل، آمازون و ... مطرح شد که نیاز داشتند تا داده های خود را در هر جایی از دنیا فراخوانی کنند و در عین حال مشکلی با مقیاس پذیری و رشد حجم داده ها نداشته باشند و بازدهی مناسبی میان میلیون ها مجموعه داده و نیازهای متنوع کاربران خود داشته باشند.

شاخص ترین مزیت های دیتابیس های NoSQL

- **ساختار نیافتگی**: در دیتابیس های NoSQL نیازی ندارید تا ابتدا schema پایگاه داده خود را طراحی کنید تا بتوانید داده های مورد نظرتان را در آن ذخیره کنید بلکه می توانید شروع به ذخیره سازی داده ها به صورت دلخواه خود کرده سپس داده مورد نیاز خود را فراخوانی کنید بدون آنکه نیازی باشد تا بدانید دیتابیس چگونه داده ها را ذخیره می کند و سازوکار آن به چه شکلی است (به طور کلی، به نظر می رسد که ویژگی ساختار نیافتگی مهمترین تفاوت بین دیتابیس های NoSQL و دیگر دیتابیس های رابطه ای باشد.)
- **مقیاس پذیری**: دیتابیس های NoSQL از روش مقیاس‌پذیری افقی پشتیبانی می کنند که این ویژگی باعث می شود تا بتوان به راحتی و بدون نیاز به ارتقاء سخت افزاری، ظرفیت دیتابیس را کم و زیاد کرد. به عبارتی، این کار باعث حذف میزان زیادی از هزینه ها و پیچیدگی های طراحی schema دیتابیس های به اصطلاح RDBMS می شود که هنگام کم و زیاد کردن ظرفیت شان، انجام چنین کاری ضروری است (اصطلاح RDBMS مخفف واژگان Relational Database Management System به معنی «سیستم مدیریت دیتابیس رابطه ای» است.)
- **عملکرد**: بعضی از دیتابیس ها طوری طراحی شده است که بهترین performance (عملکرد) خود را با مدل های خاصی از سخت افزارهای ذخیره سازی و پردازشی داشته باشند و گاهی تنها با این سخت افزارها سازگاری کامل دارند اما در مورد دیتابیس های NoSQL خواهید توانست به راحتی و با اضافه کردن سرورهای ارزان قیمتی، که اصطلاحاً Commodity Servers نامیده می شوند، بازدهی را افزایش دهید که این ویژگی باعث می شود تا کسب و کار های نوپا بتوانند با استفاده از این نوع دیتابیس ها، سرعت بالایی در ثبت و فراخوانی داده ها را رقم زده که مسلماً منجر به ایجاد تجربه کاربری خوبی بدون صرف هزینه های کلان مرتبط با دیتابیس های رابطه ای می گردد.
- **دسترسی بالا**: به طور کلی، دیتابیس های NoSQL به منظور ایجاد میزان دسترسی بالا و کاهش پیچیدگی هایی که به صورت پیش فرض در معماری دیتابیس های رابطه ای وجود دارند، طراحی شده‌اند. بعضی از دیتابیس های غیر متمرکز NoSQL (Distributed NoSQL) از نوعی معماری تحت عنوان Masterless استفاده می کنند که به

صورت خودکار داده ها را میان منابع مختلفی تقسیم می کند تا عملیات ثبت و فراخوانی، حتی در صورت از کار افتادن یکی از منابع، همچنان امکان پذیر باشد.

- **دسترسی جهانی**: دیتابیس های غیر متمرکز NoSQL با تکثیر خودکار داده ها در میان سرورهای متعدد، دیتاسنترها و یا فضاهای ابری می‌توانند تاخیر در دسترسی را به حداقل برسانند و صرف نظر از مکان جغرافیایی کاربران، تجربه کاربری یکسانی برای تمامی ایشان رقم بزنند. یکی دیگر از مزیت های این نوع دیتابیس ها، کاهش میزان قابل توجهی از کار مدیریت دستی دیتابیس است که در دیتابیس های رابطه ای وجود دارد و حذف چنین حجمی از کار، باعث می‌شود که تیم های توسعه نرم افزار توجه خود را به اولویت های کاری دیگر متمرکز کنند.

تاکنون مدل های مختلفی از دیتابیس های NoSQL توسط سازمان ها و شرکت های مختلفی طراحی شده‌اند تا پاسخگوی نیازها و کاربری های مختلف باشند. به طور کلی، این دیتابیس ها به چهار دسته تقسیم بندی می شوند که عبارتند از:

۱) دیتابیس های NoSQL مبتنی بر Key-Value

به نوعی می توان گفت که مدل ذخیره سازی Key-Value، ساده ترین نوع دیتابیس های NoSQL برای استفاده است به طوری که کاربر می تواند یک مقدار را بر اساس یک کلید خاص فراخوانی کند، یک مقدار را به یک کلید اختصاص دهد و یا یک کلید را از مجموعه ای از داده ها حذف کند. مقادیر ذخیره شده همگی از نوع آبجکت های باینری (Blob) هستند و مجموعه دیتابیس بدون در نظر گرفتن اینکه این مقدار چیست و چه مفهومی دارد، آن را ذخیره می سازد (در واقع، این مسئولیت اپلیکیشن است که بفهمد چه چیزی با چه نوعی در دیتابیس ذخیره شده است.)

از آنجایی که مدل های مبتنی بر Key-Value همیشه از روش دسترسی Primary Key استفاده می کنند، به طور کلی بازدهی بالایی داشته و به راحتی قابلیت مقیاس پذیری دارند. همچنین دیتابیس های مبتنی بر ذخیره سازی داده به صورت Key-Value از یک اصطلاحاً Hash Table برای ذخیره کردن Unique Key و Pointer، که در بعضی دیتابیس ها اصطلاحاً Inverted Index هم نامیده می شوند، استفاده می کنند.

در این نوع دیتابیس ها، هیچ رابطه ای از نوع ستون وجود ندارد؛ بنابراین همین مسئله پیاده سازی پروژه را بسیار آسان تر می کند. همچنین به یاد داشته باشیم که دیتابیس هایی از جنس Key-Value (کلید-مقدار) از میزان performance بالایی برخوردار بوده و به راحتی و بر اساس نیازهای کسب و کارهای مختلف مقیاس پذیرند که از جمله این نوع دیتابیس ها می‌توان به Redis، MemcacheDB و Riak اشاره کرد. به طور کلی، موارد استفاده دیتابیس های مبتنی بر Key-Value عبارتند از:

- برای ذخیره داده های مرتبط با session کاربران وارد شده در سیستم
- مدیریت پروفایل های بدون ساختار کاربران
- ذخیره تنظیمات حساب کاربری
- ذخیره داده های سبد خرید در فروشگاه های آنلاین

البته دیتابیس های Key-Value برای همه موارد استفاده هم ایده‌آل نیستند؛ برای مثال، در موارد زیر بهتر است که از این نوع دیتابیس ها استفاده نکنیم:

- وقتی باید یک کوئری بر اساس مقداری مشخص به دیتابیس بزنیم
- نیاز به وجود رابطه ای معنادار میان مقادیر ذخیره شده داریم
- باید عملیاتی را روی چندین کلید منحصر به فرد انجام دهیم
- کسب و کار ما نیاز دارد تا مرتباً بخشی از داده ها را به روز رسانی کنیم

۲) دیتابیس های NoSQL مبتنی برDocument

این نوع دیتابیس ها شبیه مدل قبلی (Key-Value) هستند از این لحاظ که اینجا هم یک جفت کلید-مقدار وجود دارد و داده به عنوان یک مقدار ذخیره می‌شود و کلید مربوط به آن هم نشان گری منحصر به فرد می باشد اما تفاوت کار در اینجا است که در مدلDocument ، مقدار ذخیره شده به نوعی کاملاً ساختار یافته (یا حدوداً ساختار یافته) است به طوری که این داده های ساختار یافته به عنوان یک داکیومنت شناخته می شوند و می‌توانند یکی از فرمت های XML ،JSON، یا BSON را داشته باشند و از جمله دیتابیس های NoSQL از جنس داکیومنت می‌توان MongoDB، Apache CouchDBو Elasticsearch را نام برد که موارد استفاده آنها به شرح زیر است:

- فروشگاه های آنلاین
- سیستم های مدیریت محتوا
- پلتفرم های تجزیه و تحلیل دیتا
- پلتفرم های وبلاگ

به یاد هم داشته باشیم که دیتابیس های مبتنی بر داکیومنت (سند) انتخاب مناسبی برای مواردی که می خواهید کوئری های پیچیده ای به پایگاه داده بزنید و یا اپلیکیشن شما نیاز به محاسبات پیچیده ای روی دیتا دارد نیست.

۳) دیتابیس های NoSQL مبتنی برColumn

در این دست دیتابیس های NoSQL ، داده ها به جای ذخیره سازی در Row (ردیف) در قالب یکسری سلول (Cell) ذخیره شده سپس چند ستون از داده ها با یکدیگر هم گروه می شوند و مجموعه ستون های متعلق به یک گروه هم می‌توانند به صورت مجازی بی نهایت ستون را شامل شوند که می‌توان در زمان اجرا و یا در زمان تعریف ساختار، آنها را ساخت.

جالب است بدانید که برای عملیات ثبت و فراخوانی داده ها، به جای سطر از ستون ها استفاده می‌شود (مجموعه ستون ها، گروهی از داده های یکسان هستند که معمولاً همراه با هم فراخوانی می شوند.) به عنوان مثال، ما معمولاً نام کاربری، نام خانوادگی و دیگر اطلاعات کاربران را به صورت هم زمان فراخوانی می کنیم ولی دستیابی به اطلاعات سفارش های آنها در یک فروشگاه آنلاین به همراه این دست اطلاعات فراخوانی نخواهد شد.

مزیت اصلی ذخیره سازی داده ها در ستون نسبت به دیتابیس های رابطه ای، جستجو و دستیابی سریع و تجمیع داده ها است. دیتابیس های رابطه ای، یک ردیف از داده را به عنوان یک ورودی به هم پیوسته ذخیره می کنند و ردیف های متفاوت در مکان های متفاوتی ذخیره می شوند اما این در حالی است که دیتابیس های مبتنی بر ستون از جنس NoSQL ، همه سلول های مربوط به یک ستون را به عنوان یک ورودی پیوسته ذخیره می کنند که این عمل بالطبع باعث سریع تر شدن جستجو و دستیابی به داده ها می‌شود. همچنین هر گروه از ستون ها می‌توانند با مجموعه ای از ردیف ها در دیتابیس های رابطه ای مقایسه شوند بدین شکل که Key همان Row

است و هر Row هم شامل چندین Column می‌شود اما تفاوت در اینجا است که ردیف های متفاوت نیازی ندارند که ستون های یکسانی داشته باشند و ستون ها را می‌توان در هر زمان و به هر ردیفی اضافه کرد بدون اینکه نیاز باشد آنها را به ردیف های دیگر هم اضافه نمود. به طور کلی، موارد استفاده دیتابیس های مبتنی بر Column عبارتند از:

- سیستم های مدیریت محتوا
- پلتفرم های وبلاگ
- سرویس هایی که داده هایی با تاریخ انقضاء را در خود ذخیره می کنند
- سیستم هایی که نیاز به request های بسیار سنگین ثبت داده (مانند ثبت لاگ های سیستم) دارند.

از جمله از دیتابیس های NoSQL مبتنی بر ستون می‌توان Cassandra و Apache Hadoop Hbase را نام برد. به طور کلی، چنانچه نیاز به انجام کوئری های پیچیده دارید و یا روش کوئری زدن شما به دیتابیس مرتباً تغییر می کند، نباید از دیتابیس های NoSQL مبتنی بر Column استفاده کنید. مورد دیگری که نباید از این نوع دیتابیس استفاده کنید، زمانی است که تعریف مشخصی از سازوکار دیتابیس ندارید.

۴) دیتابیس های NoSQL مبتنی برGraph

دیتابیس های مبتنی بر گراف اصولا بر اساس مدلی تحت عنوان Entity-Attribute-Value (موجودیت-ویژگی-مقدار) ساخته می شوند؛ موجودیت ها به عنوان Node (گره) هم شناخته می شوند که دارای یکسری خصوصیات مخصوص به خود هستند. این یک روش بسیار انعطاف پذیر برای توضیح اینکه چگونه داده ها با دیگر داده ها ارتباط دارند است در حالی که دیتابیس های قدیمی مثل RDBMS توضیحات مربوط به هر رابطه ممکن را به صورت یک فیلد حاوی Foreign Key (کلید خارجی) و یا جداول میانی ذخیره می کنند در حالی که دیتابیس های مبتنی بر گراف این اجازه را به شما می دهند تا به صورت مجازی هر رابطه ای را در لحظه تعریف کنید. از جمله دیتابیس های NoSQL مبتنی بر گراف می‌توان به Neo4j،ArangoDB و OrientDB اشاره کرد به طوری که موارد استفاده آنها شامل موقعیت های زیر می‌شود:

- سیستم های شناسایی کلاه برداری آنلاین
- جستجوی مبتنی بر گراف
- انجام task های مختلف در حوزه های شبکه و فناوری
- شبکه های اجتماعی و ...