یازدهمین کنگره ملی سراسری
فناوریهای نوین در حوزه توسعه پایدار ایران
11th National Congress of
the New Technologies in Sustainable Development of Iran
senaconf.ir

مروری بر تکنیک های بهینه سازی عملکرد دیتابیس با استفاده از شاردینگ

فرهنگ پدیداران مقدم[1]، رجبعلی سجادیان فر[2]

[1] استادیار گروه کامپیوتر مجتمع آموزش عالی فنی و مهندسی، اسفراین padidaran@esfarayen.ac.ir

[2] دانشجوی کارشناسی ارشد مهندس نرم افزار موسسه آموزش عالی اشراق، بجنورد rajabali.sajadianfar.uni@gmail.com

چکیده

هدف از این مقاله بررسی تکنیک شاردینگ در دیتابیس های سایز بزرگ و نحوه عملکرد تکنیک مذکور در افزایش کارایی، بهبود منابع و به طبع آن افزایش سرعت پاسخ گویی دیتابیس بوده است. نتایج به دست آمده از این تحقیق نشان از بهبود عملکرد دیتابیس ها پس از اجرای شاردینگ و وابستگی عملکرد حاصله به منابع سخت افزاری و نوع دیتابیس های مورد استفاده بوده است. افزایش سرعت پاسخ گویی به تنهایی با استفاده از تکنیک شاردینگ امکان پذیر نبوده و می بایست از تکنیک های نرم افزاری مانند خوشه بندی و مرتب سازی و حذف دیتای ورودی نامناسب و همچنین بایستی از سخت افزار با سرعت بالا نیز بهره برد. بررسی های انجام شده نشان از محدودیت های ناشی از سخت افزار و همچنین حجم داده ورودی در زمان مشخص به دیتابیس و نوع دیتابیس مورد استفاده دارد. با توجه به بررسی های انجام گرفته امکان ارائه راه کاری خاص، دقیق و جامع و مورد تایید برای استفاده در تمامی دیتابیس ها وجود ندارد تنها تاییدیه انجام شده در اکثر بررسی ها بهبود عملکرد دیتابیس ها با استفاده از تکنیک شاردینگ را نشان می دهد.

واژه‌های کلیدی

خوشه بندی پایگاه داده، تقسیم بندی پایگاه داده، پارتیشن بندیافقی داده ها ، شاردینگ

۱. متن مقاله

با توسعه سرسام آور برنامه های تحت وب و هچنین توسعه روز افزون شبکه های اجتماعی نیاز به ذخیره سازی اطلاعات به طور تصادعدی افزایش یافته است[5] در محیط کسب وکار امروزی نیاز روزافزونی به پایگاه هایاطلاعاتی توزیع شده و برنامه های کاربردی مشتری/سرور وجوددارد زیرا تقاضا برای اطلاعات قابل اعتماد، مقیاس پذیر و قابل دسترس همچنان در حال رشد است. سیستم های پایگاه داده توزیع شده به دلیل توزیع داده های آن در سایت های مختلف شبکه، بهبودهایی را در ارتباطات و پردازش داده ها ارائه میکنند. دسترسی به داده ها نه تنها سریع تر شده است بلکه احتمال خرابی آن نیز کمتر است وبه کاربران امکان کنترل محلی روی داده ها را می دهد [1]

دپارتمان فناوری اطلاعات (IT) هر سازمانی مسئولیت ارائه محاسبات و ذخیره سازی و پشتیبان گیری و امکانات شبکه قابل اعتماد را با کمترین هزینه ممکن سازد. [5] به دلیل تولید روز افزون داده در سطح جهان کار متخصصین این حوزه در تامین و توسعه زیر ساخت های مورد نیاز برای رفع مشکلات حاصل از این فرایند پرسرعت بسیار سخت خواهد بود. RDBMS ها یا همان دیتابیس های سنتی در مدیریت حجم عظیمی از داده های بدون ساختار ناتوان می باشد که این خود باعث ایجاد دیتا بیس های NoSQL گردیده است[3] بر تری این نوع دیتابیس های نسبت به دیتابیس های رابطه ای امکان ورود و نگهداری انواع مختلف داده و قابلیت Scability می باشد.

دیتابیس های NoSQL ایجاد گردیده اند تا خارج از ویژگی های ACID در دیتابیس های رابطه ای سرویس دهی انجام می دهند [2]. به علت انعطاف و توانمندی آن ها در کنترل و مدیریت دیتابیس ها بزرگ و همچنین امکان مقیاس پذیری آن ها اشتیاق توسعه دهندگان برای استفاده از این نوع دیتابیس ها روبه افزایش می باشد.

۱-۱ مقیاس پذیری

Scale Up – مقیاس پذیری عمودی با توسعه سخت افزار های ماشین از قبیل سی پی یو و رم و هارد و ... انجام می پذیرد و به دلیل محدودیت های سخت افزاری تا حدی قابل اجرا بوده و پس از تکمیل ظرفیت دیگر امکان اجرا وجود ندارد[6]

Scale Out - مقایس پذیری افقی که تنها در دیتابیس های NO SQL وجود دارد به مفهوم توسعه سرور ها و منابع پردازشی در موازات یکدیگر می باشد که این روش با توجه به ایجاد موازی کاری درخواستها و افزایش منابع دیتابیس ها باعث بهبود عملکرد خواهد گردید.[6]

۱-۲ NoSQL [8]

یک رویکرد بدون طرحواره(Schema) می باشد که ادعا می کند حج بزرگی از دیتا با تنوع بالا را به طور کامل مدیریت نماید.[2] این نوع دیتابیس ها از مقیاس پذیری عمودی و افقی پشتیبانی می نماید و در چهار نوع طبقه بندی می گردند:

۱-۲-۱ مدل داده های سند گرا. این پایگاه داده ها داکیومنت محور می باشند از محبوبترین ها می توان به CouchDB , MongoDB نام برد [3]

۱-۲-۲ مدل داده های ستون گرا که داده های ورودی بر مبنای ستونهای جداول می باشند مانند Cassandra HBASe [3]

۱-۲-۳ مدل داده گراف گرا از گراف های داده ای برای نگهداری و پشتیبانی استفاده می نماید مانند FlockDB-Neo4j [3]

۱-۲-۴ مدل داده بر مبنای کلید و اشاره گر و جدول هش می باشد .مانند Amazons Dynamo , Riak [3]

۱-۳ شاردینگ

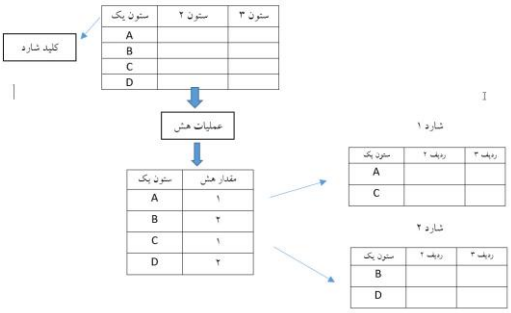
تکنیکی برای مقیاس پذیری افقی می باشد که بر مبنای آن دیتابیس های بزرگ به قسمت های کوچک تر و در سرور های بیشتر تقسیم می گردد که با این کار منابع بیشتر سخت افزاری و پهنای باند بیشتر تخصیص می یابد.. این تکنیک بخش جدایی ناپذیر دیتابیس های NoSQL می باشد.

این تکنیک اولین بار در سال ۱۹۹۰ و در یک بازی رایانه ای آنلاین و برای توزیع تعداد کاربران در سرور های متفاوت انجام گردید و سپس در فروشگاه های اینترنتی برای تقسیم اطلاعات مشتریان بر اساس مختصات جغرافیایی استفاده گردید.

این تکنیک برای مقیاس پذیری افقی مورد استفاده قرار می گیرد مثال قابل درک تر آن در فروشگاه های اینترنتی ایجاد شعب متعدد در مراکز استان ها می باشد که این خود باعث کاهش بار کاری شعبه مرکزی می گردد.

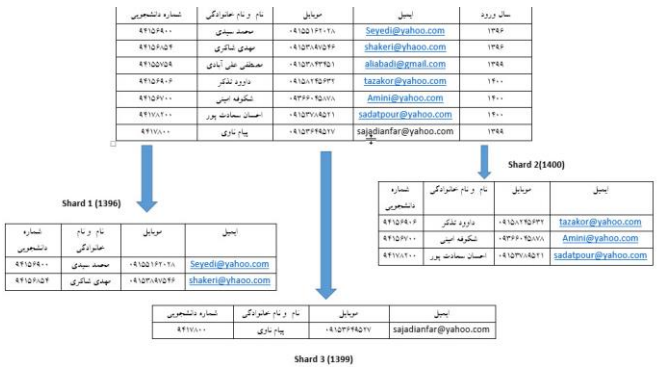
به صورت معمول شاردینگ بر دو اساس انجام می پذیرد:

۱-۳-۱ شاردینگ بر اساس کلید (خرد کردن دیتابیس بر اساس کلید های خاص انجام می پذیرد[4] (شکل یک)



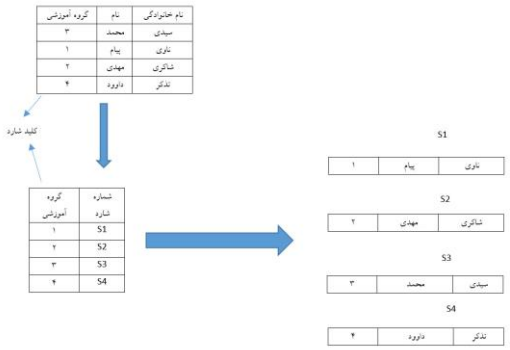
شکل یک. نمونه ساده شاردینگ بر اساس کلید

۱-۳-۲ شاردینگ بر اساس دامنه داده ها (شکل دو)



شکل دو. نمونه شاردینگ بر اساس دامنه داده

۱-۳-۳ شاردینگ بر اساس مسیر که در این نوع تکنیک از جداول Lookup برای خرد کردن و مدیریت دیتا استفاده می گردد.(شکل سه)



شکل سه. نمونه شاردینگ بر اساس دایرکتوری

زمانی از تکنیک شاردینگ استفاده می کنیم که حجم دیتا بسیار بزرگ بوده و خود دیتابیس به تنهایی توانایی مدیریت درخواست ها را نداشته باشد. از جمله مزایای استفاده از تکنیک شاردینگ می توان به توسعه پذیری و در صورت استفاده از تکنیک مناسب باعث کاهش هزینه ها و در طولانی مدت باعث افزایش بهره وری و کارای سیستم می گردد. از معایب استفاده از تکنیک شاردینگ میتوان به پیچیدگی پشتیبان گیری و راه اندازی و نگهداری اشاره نمود.

در تمامی مقالات مرتبط و تحقیقات انجام گرفته شده دیتابیس MongoDB به دلیل کاربر پسند بودن و همچنین متن باز بودن و همچنین پشتیبانی از قابلیت شاردینگ خودکار مورد استفاده و بررسی قرار گرفته است. از سوی دیگر این خود محدودیتی برای انجام بررسی دیتابیس های دیگر می باشد که گاها خرید آنها برای اشخاص عادی معقول به نظر نمی رسد.

یکی از تحقیقات مفید انجام گرفته شده در پروژه تحقیقاتی SYNAT بوده است که با استفاده از دیتابیس Postgres و با استفاده از تکنیک نرم افزاری حذف داده ها و روابط نامرتبط و ایجاد روابط معنایی مناسب مابین دیتا موجود و تقسیم کوئری ها با استفاده از شاردینگ بهره وری سیستم را در برخی از موارد تا ۲۳ ٪ افزایش داده اند و پس از اجرای شاردینگ مصرف رم کاهش چشمگیری داشته است. در این تحقیق[9] پیشنهاد گردیده است که جداول تقسیم شده و آن ها را به صورت تکه ای پردازش کنیم. هر قطعه (شارد) حاوی اطلاعاتی در مورد یک شی در داده است، به طوری که میتواند به طور مستقل از اشیاء دیگر بدون تحریف نتیجه کلی پردازش شود. با اجرای این تکنیک و تقسیم بندی (تجزیه کار) محاسبات بین چندین رشته در یک پردازنده چندهسته ای یا بین چندین ماشین در محیط شبکه توزیع می گردد و کلیدموفقیت در بهبود عملکرد ترکیب و روابط بین اشیاء است. اگر این اطلاعات (دانش) را داشته باشیم که اشیا تا حد زیادی مستقل هستند و می توانند به صورت موازی پردازش گردند می توان بهبود عملکرد پرس و جو ها را دریافت نمود. [9]

در برخی موارد با بررسی های انجام گرفته بر روی دیتابیس های متفاوت گاها نتایجی متفاوت از برداشت این تحقیق حاصل گردیده است به طور مثال در بررسی سه دیتا بیس Cassandra , Sql Server 2012 , Mongo DB بر اساس نظرات کارشناسان این امر و تحقیقات انجام گرفته بایستی دیتابیس MongoDB عملکرد بهتری نشان دهد ولی پس از انجام تحقیقات بر اساس ورود اطلاعات به میزان ۱۰۰۰۰۰۰ ردیف و ۳۱ ستون[2] و اجرای کوئری های متفاوت مانند درج و واکشی و حذف و غیره نتایج زیر حاصل گردید:

دیتابیس Sql Server و Cassandra عملکرد بهتری در زمان پاسخ گویی نسبت به MongoDB از خود نشان داده اند البته این موضوع می توان رابطه مستقیمی با داده ورودی و سخت افزار مورد استفاده داشته باشد ولی در برخی موارد این آزمایش افزایش عملکرد پس از اجرای تکنیک شاردینگ مشهود می باشد.[2]

با توجه به این موضوع که بهبود عملکرد بایستی از نظر اقتصادی نیز مورد تایید باشد تحقیقاتی در زمینه استفاده از دیتابیس های ابری و همچنین رابطه هزینه و عملکرد انجام پذیرفته است در این تحقیقات که بر مبنای تکنیک شاردینگ انجام پذیرفته است نتایجی حاصل گردیده است که در محیط های غیر ابری نیز امکان استناد خواهد داشت. تقسیم مناسب و درست دیتا و جداول و توزیع یکنواخت بار بر روی فضاها خود می تواند باعث بهبود عملکرد و سرعت پاسخ گویی گرددو در صورت انتخاب نادرست میزان شاردها باعث کاهش عملکرد و افزایش هزینه تحمیلی به شرکت ها گردد. به طور مثال یک دیتابیس با حجم ۱۰۰ گیگ می تواند به صورت یکجا استفاده گردد و یا به ۱۰ تکه ۱۰ گیگی تقسیم گردد و یا به دو تکه ۵ گیگی تقسیم شده که با استناد به استفاده زافضای ابری هزینه خرید و نگهداری ۱۰ شارد در کوتاه مدت به مراتب بیشتر از هزینه نگهداری دو شارد با حجم ۵ گیگ می باشد. این نتیجه گیری با توجه به هزینه های اعلام شده خرید سرویس های ابری متفاوت خواهد بود و لزوما تایید شده نمی باشد.[7]

انتخاب نوع تکنیک شاردینگ و استفاده از خوشه بندی در کاهش هزینه خرید سخت افزار و با منابع ابری می تواند نقش موثری در کاهش سرعت پاسخ گویی ها داشته باشد.[7] استفاده تنها از تکنیک شاردینگ در هیچ یک از مقالات ارائه شده توصیه نگردیده است . یکی از تکنیک های پیشنهادی استفاده از خوشه بندی داده و سپس تقسیم داده ها با استفاده از تکنیک شاردینگ می باشد این فرایند می توان نقش به سزایی در بهبود عملکرد دیتابیس MongoDB ایجاد نماید[5] .

دیتابیس MongoDB به دلیل دارا بودن قابلیت شاردینگ خودکار و تقسیم متوازن داده مابین شاردها و ایجاد کمترین بار کاری برای جابه جایی داده ها ،پس از استفاده از خوشه بندی پیشنهادی K-Means بهوبد عملکرد را شامل گردیده است.[5]

شاردینگ بر اساس دامنه داده پیچیدگی کمتری دارا می باشد[4] به طور مثال تقسیم دیتابیس اطلاعات دانشجویان دانشگاه بر اساس سال ورود می تواند استفاده گردد که این تقسیم بندی داده ها بسیار ساده و در فرایند خواندن بسیار کارامد می باشد ولی امکان ایجاد Hot Spot بسیار زیاد می باشد .این مفهوم به مانند گلوگاه می باشد به طور مثال تعداد تقاضا های ارسالی برای دیتابیسی که اطلاعات دانشجویان سال ۱۴۰۰ را دارا می باشد بسیار بیشتر از دیتابیسی می باشد که اطلاعات دانشجویان سال ۱۳۸۷ را شامل می شود و این خود می تواند باعث کاهش عملکرد سیستم در آن نقطه گردد راه حل جایگزین برای رفع این مشکل استفاده از تکنیک شاردینگ با استفاده از کلید می باشد که این کلید اختصاصی بوده و میتواند بر اساس الگوی هش داده های را به صورت متوازن بر روی شاردها توزیع نماید پیچیدگی این روش بیشتر از روش دامنه می باشد. تایید افزایش کارایی دیتابیس پس از اجرای تکنیک شاردینگ بر کسی پوشیده نیست ولی این موضوع مشکلات دیگری را نیز ایجاد می نماید مشکلاتی از قبیل پیچیدگی افزایش تعداد شارد ها و کندی فرایند توسعه و به روزرسانی ...

استفاده از کلید های مناسب و جدوال هش و ..... در رفع پیچیدگی های این نوع تکنیک کمک می نماید. بر اساس تحقیقات انجام گرفته شده استفاده از تکنیک شاردینگ در دیتابیس های سایز بزرگ نتاییج متفاوت تری را نسبت به دیتابیس سایز کوچک نشان می دهد تکنیک مذکور در مقیاس های بزرگ افزایش کارایی و بهبود سرعت پاسخگویی و کاهش منابع مورد استفاده سرور ها را ایجاد می نمایدو در دیتابیس های کوچک تر عملکرد نسبتا پایین تری از خود نشان می دهد.

در تحقیقات انجام گرفته شده اولویت سخت افزار بسیار مشهود بوده است دیتابیس هایی که از حافظه های فلش استفاده میکنند بسیار پر قدرت تر از دیتابیس های معمولی عمل می نمایند و پیرو این موضوع دیتابیس هایی که از تکنیک شاردینگ استفاده می نمایند در حجم بالا بسیار پر سرعت بوده و عملکرد بهتری از خود نشان می دهند و ترکیب این عملیات با خوشه بندی و همچنین استفاده از تکنیک های حذف داده های تکراری و بهینه سازی دیتابیس ها خود عاملی برای بهبود پاسخ گویی دیتابیس ها می باشد.

ردیف	نام مقاله	تکنیک پیشنهادی	مزایا	معایب
۱	Distributed Database Using Sharding DatabaseArchitecture	توزیع داده در دیتابیس هایی که منابع سخت افزاری مشترک ندارند	افزایش کارایی و سهولت توسعه پذیری	پیچیدگی در اجرا
۲	Query Performance Analysis of NoSQL and Big Data	قیاس مابین سه دیتا بیس پیسنهادی	کاهش زمان پاسخ گویی پس از اجرای شاردینگ	عدم امکان تایید دیتابیس پیشنهادی برای هر نوع دیتابیسی و با هر حجمی
۳	A qualitative analysis of the performance of MongoDB vs MySQL Databas	استفاده از دیتابیس **MongoDB**	کاهش زمان ورود اطلاعات و افزایش کارایی در دیتابیس های سایز بزرگ	وابستگی به سخت افزار و میزان داده ورودی در بررسی ها موثر می باشد
۴	Efficient Horizontal Scaling of Databases using Data Sharding Technique	بررسی مقیاس پذیری افقی و عمودی و انواع مقیاس پذیری های افقی	کار آمدی سیستم را بالا می برد و مقیاس پذیری را تسهیل می نماید.	یکپارچگی با تهدید روبه رو می باشد/افزودن نود به دیتا بیس سخت و گران می باشد و اعمال تغییرات بسیار پر هزینه می باشد.
۵	. Performance improvement of sharding in mongoDB using k-mean clustering algorithm	استفاده از الگوریتم خوشه بندی **K-Means**	بهبود عملیات خواندن و نوشتن و متعادل سازی توزیع داده در شارد ها	در صورت انتخاب نادرست نقاط **K** مشکلاتی در توزیع ایجاد خواهد شد
۶	Sharding by Hash Partitioning A Database Scalability Pattern to Achieve Evenly Sharded Database Clusters	استفاده از مقیاس پذیر افقی بر اساس کلید هش	قابلیت خواندن و نوشتن بهبود می یابد. بهبود عملکرد را در خوشه ایجاد می نماید	قابل اجرا در تمامی دیتابیس های نمی باشد.

۷	. Economical AspectsofDatabaseSharding	بررسی تکنیک شاردینگ از نظر صرفه اقتصادی	کاهش هزینه ها با استفاده از فضای ابری در مدت زمان طولانی	در صورت انتخاب شاردبندی نادرست هزینه راه اندازی و نگهداری در کوتاه مدت بسیار سنگین خواهد بود
۸	. Knowledge Driven Query Sharding	استفاده از شاردینگ در کوئری	اجرای سریع تر پرس و جو ها و	ایجاد تعادل در سربار محاسبانی کاری بسیار پیچیده می باشد.

نتیجه گیری:

با ترکیب راه حل های دکر شده در متن مقاله می توان بهبود نسبی سرعت پاسخ گویی را ایجاد نمود. در مرحله بایستی داده ورودی اولیه بهینه سازی گردد و دیتای مازاد و کم مصرف از لیست حذف و یا در طبقه پایین تری از اهمیت قرار داده شود و در مرحله بعدی باید بر اساس ارزش داده ها و درجه بندی لازم انجام پذیرد که این کار می تواند با الگوریتم خوشه بندی انجام پذیرد پس از این عملیات طبقه بندی سخت افزار ها در چند مرتبه ارزشی طبقه بندی گردد به طور مثال داده دارای ارزش بالا در پایگاه داده های طلایی (حافظه های فلش) و پایگاه داده نقره ای (هارد با سرعت 15K) و پایگاه داده برنزی (هاردهای با سرعت 10K) و ... و سپس استفاده زا تکنیک شاردینگ بر مبنای کلید برای توزیع داده ها در بخش های مختلف توزیع شده، با این روش پیشنهادی که پیچیدگی زیادی خواهد داشت می توان با توزیع مناسب بار بروی سرور ها بخش زیادی از منابع را بهینه نمود و سرعت پاسخ گویی کاربران را افزایش داد.

منابع

[1] Bahaa Mahmoud Abdelhafiz, 2020. Distributed Database Using Sharding DatabaseArchitecture
2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)
[2] Ashis Kumar Samanta, Bidut Biman Sarkar, Nabendu Chaki 2018. Query Performance Analysis of NoSQL and Big Data
Fourth International Conference on Research in Computational Intelligence and Communication Networks
(ICRCICN).
[3] Mayur M Patil, Akkamahadevi Hanni, CH Tejeshwar,Priyadarshini Patil 2017. A qualitative analysis of the performance of MongoDB vs MySQL Database based on insertion and retriewal operations using a web/android application to explore Load Balancing International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC 2017)
[4] Ragul R, Arokia Paul Rajan R, 2020. Efficient Horizontal Scaling of Databases using Data Sharding Technique, International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075
mongoDB [5] Mrugesh P Patel,Mosin I. Hasan ,Hemant D. Vasava, 2014. Performance improvement of sharding in
using k-mean clustering algorithm, International Journal of Advance Engineer ing and Research Development
(IJAERD)
[6] Caio H. Costa, Jo˜ao Vianney B. M. Filho, Paulo Henrique M. Maia and Francisco Carlos M. B. Oliveira., 2015. Sharding by Hash Partitioning A Database Scalability Pattern to Achieve Evenly Sharded Database Clusters, 17th International Conference on Enterprise Information Systems
[7] Uwe Hohenstein and Michael C. Jaeger., 2014. Economical AspectsofDatabaseSharding
4thInternationalConferenceonCloudComputingandServicesScience (CLOSER-2014),
[8] Adam Krasuski1 and Marcin Szczuka, 2012. Knowledge Driven Query Sharding?