تاریخ دریافت مقاله: ۹۶/۰۱/۱۱
تاریخ پذیرش مقاله: ۹۶/۰۳/۲۴

ارزیابی و مقایسه کارایی پایگاه داده‌های کلید–مقدار با هدف انتخابِ مبتنی بر نیاز

سامان کشوری*
دانشجوی کارشناسی ارشد نرم افزار- گروه کامپیوتر - دانشگاه جامع امام حسین (ع)- تهران- ایران
پست الکترونیکی: SaKeshvari@ihu.ac.ir

محمدعلی جوادزاده
استادیار گروه کامپیوتر - دانشگاه جامع امام حسین (ع)- تهران- ایران
پست الکترونیکی: MJavadzad@ihu.ac.ir

مهدی نقوی
استادیار گروه کامپیوتر- دانشگاه جامع امام حسین (ع)- تهران- ایران
پست الکترونیکی: Naghavi@aut.ac.ir

چکیده

یکی از دغدغه‌های توسعه‌دهندگان نرم‌افزار انتخاب پایگاه داده متناسب با نیازهای طراحی در پروژه‌های ایشان است. با توجه به تازگی، کاربرد وسیع و تنوع در پایگاه داده‌های کلید–مقدار[1] در سال‌های اخیر، نیاز به انتخاب بهترین نمونه از این بانک اطلاعاتی بیش‌ازپیش اهمیت دارد. ارزیابی و مقایسه کارایی بانک‌های اطلاعاتی کلید–مقدار راهی برای انتخاب پایگاه داده مناسب در هر کاربرد است. برای این منظور معیارهایی برای مقایسه انتخاب شده‌اند که برگرفته از نیازهای بانک‌های اطلاعاتی توزیع‌شده هستند. سپس چهار پایگاه داده ردیس، رایاک، آمازون داینامو و مِمکَش که از پرکاربردترین بانک‌های اطلاعاتی کلید–مقدار هستند، انتخاب شده و مورد ارزیابی قرارگرفته‌اند. در ادامه با مقایسه جزئی‌تر به دلیل مشابهت در ساختار و کاربرد، بانک‌های اطلاعاتی ردیس با ممکش و رایاک با داینامو در بخش‌های جداگانه مقایسه شده‌اند تا دقیق‌تر بتوان مقایسه را انجام داد. با مقایسه این چهار بانک اطلاعاتی می‌توان بهترین آن‌ها را با توجه به نیازها و کاربردهای برنامه در استفاده از بانک‌های اطلاعاتی کلید–مقدار، از بین این چهار بانک اطلاعاتی انتخاب کرد تا به بهترین کارایی در استفاده از بانک‌های اطلاعاتی کلید–مقدار رسید.

کلیدواژه‌ها: پایگاه داده کلید–مقدار، پایگاه داده توزیع‌شده، بانک اطلاعاتی NoSQL.

مقدمه

با توجه به رشد استفاده کاربران از اینترنت و تنوع در تولید داده‌ها و همچنین افزایش حجم اطلاعات، نیاز به روش‌های نوینی جهت ذخیره‌سازی، مدیریت و پردازش این حجم از اطلاعات وجود دارد. با توجه به ضعف‌هایی که سیستم‌های ذخیره‌سازی سنتی دارند، این نوع ساختارها جوابگوی برخی از نیازهای امروزی نیستند و به‌منظور رفع نیاز ذخیره‌سازی، از سیستم‌های فایلی توزیع شده

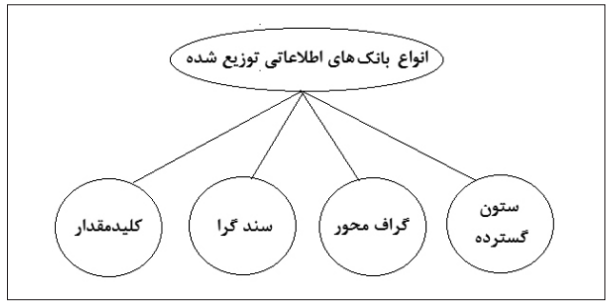
* نویسندهٔ مسئول

1- Key-Value

استفاده می‌شـوند [۱]. در رابطه با بانک‌های اطلاعاتی در سـال ۲۰۰۰ میلادی، اریک بروئر[2] با ارائه نظریه CAP[3] به کمبودهـا و محدودیت‌های مدل رابطه‌ای در سیسـتم‌های توزیع‌شده اشـاره کرد [۲]. با توجه به این نظریه، ثبات و قابلیت دسترسی بالا، هر دو در یک پایگاه داده موجود در یک شبکه گسترده و وسـیع قابل فراهم شدن نیستند. این نظریه، باعث شد تا توجه‌ها به سمت داده‌های گسترده در سطح شبکه جلب شده و مدل‌هایی با تأکید بر توزیع‌شدگی و دسـترس‌پذیری بالا به‌عنوان نیازمنـدی اصلی معرفی شـوند و ثبات به‌عنوان اولویت بعـدی که امکان به تأخیر انداختن آن در مقایسـه با سایر اولویت‌ها نیز وجود دارد، معرفی شد [۳]. به همین دلیل، عبارت NoSQL مفهومی است که جهت مشـخص کردن پایگاه‌های داده‌ای به کار می‌رود که با پایگاه‌های داده‌ای رابطه‌ای سنتی متفاوت هستند [۴]. این پایگاه‌های داده در اغلب موارد دارای سـطر و ستون نبوده و عملیات الحاق در آن‌ها بی‌معنی بوده و به‌صورت افقی مقیاس‌پذیر هسـتند [۵]. در سـال ۲۰۱۲ میلادی، کار روی پروژه‌ای بانام UnQL[4] آغاز شـد که هدف آن، تدوین اسـتاندارد زبان پرس‌وجو در پایگاه‌های داده‌ای NoSQL اسـت. این زبان مجموعه‌ای ورای SQL به شـمار آمده و زبان SQL را می‌توان نمونه بسیار محدودشده آن به شمار آورد. بااین‌حال، زبان UnQL دسـتورهای تعریف داده[5] را پوشـش نخواهـد داد [۶]. بانک‌های اطلاعاتـی NoSQL، با توجه به کاربردشـان به چهار دسـته تقسیم می‌شوند که در شـکل ۱ نشـان داده شـده‌اند. بانک‌های اطلاعاتی سـندگرا اقلام داده‌ای را در سـندها نگهداری می‌کنند که ایـن مدل داده‌ای را جهت تهیـه برنامه‌های مدیریت محتوا به مدلی مناسـب تبدیل کرده است [۷]. بانک‌های اطلاعاتی MongoDB, CouchBase, CouchDB نمونه‌هایی از این نوع بانک‌های اطلاعاتی هستند. بانک اطلاعاتی ستون گسترده از سـتون‌های چندبعدی (ابر ستون) علاوه بر ستون‌های یک‌بعـدی جهت ذخیره‌سـازی اسـتفاده می‌کننـد که در داده‌های حجیم بسـیار کاربرد دارند و سـرعت بازیابی اطلاعـات در آن‌هـا بالاسـت [۸]. Cassandra و Hbase دو نمونه از این مـدل بانک‌های اطلاعاتی هسـتند. بانک‌های اطلاعاتی مبتنی بر گراف که از نظریه گراف پیروی می‌کنند جهت ردیابی ارتباطات بین اطلاعات طراحی‌شده‌اند و برای برنامه‌های شبکه‌های اجتماعی در عملیاتی همچون سیستم نظرگذاری، دنبال کـردن، عضویت در گروه‌های اجتماعی و علاقه‌مندی‌ها بسیار مفید هستند [۹]. Neo4j نمونه موفق پیاده‌سازی این مدل داده‌ای است. اخیراً بانک‌های اطلاعاتی چنـد مدلی نیز به وجود آمده که از چندین نوع از این بانک اطلاعاتی پشـتیبانی می‌کننـد [۱۰]. از این نـوع بانک‌های اطلاعاتی می‌تـوان MarkLogic, OrientDB, Adabas را نام برد [۱۱].

در این مقاله به ویژگی‌هـای بانک‌های اطلاعاتی کلید–مقدار و مقایسـه کارایی چهار نـوع پرکاربرد آن پرداخته شـده اسـت. این بانک‌های اطلاعاتی در مواقعی که پنهان کـردن محتوا، ارجاعات سـریع به داده‌هـا و نمایه‌گذاری مورد نیاز است کاربرد دارند. با توجه به تنوع در بانک‌های اطلاعاتی کلید–مقدار، انتخاب مناسب‌ترین آن‌ها با توجه به نیازمندی‌های پـروژه نرم‌افزاری در افزایش کارایی مورد نیاز تأثیر بسزایی دارد. در همین راستا در این مقاله و در بخش ۲، سـاختار، ویژگی‌ها، ذخیره‌سازی، مقیاس‌پذیری و سـازگاری این نوع بانک‌های اطلاعاتی بیان شده است. سـپس در بخش ۳ با توجه به میزان کاربرد انواع مختلف این نوع بانک اطلاعاتی به ویژگی‌های چهار بانک اطلاعاتی پرکاربـرد (رایـاک، ردیس، ممکـش و آمـازون داینامو) پرداخته شده اسـت. انتخاب بین این چهار بانک اطلاعاتی در کاربردهـای مربوط به این نوع بانک اطلاعاتی با توجه به نوپا بودن آن‌ها یکی از چالش‌های استفاده از بانک‌های اطلاعاتی NoSQL است؛ به همین دلیل در بخش ۴ با ارزیابی و مقایسه‌ای که در این مقاله انجام شده –با برشمردن نقاط ضعـف و قوت هرکـدام از این چهار بانـک اطلاعاتی در

2- Eric Brewer
3- Consistency- Availability- Partition tolerance
4- Unstructured data Query Language
5- Data Definition Language (DDL)



شکل ۱: انواع بانک اطلاعاتی استفاده شده در ساختار NoSQL

قالب مقایسه- می‌توان یکی از این بانک‌های اطلاعاتی را با توجه به کاربرد بانک‌های کلید-مقدار و نیازمندی‌های برنامه انتخاب کرد تا بهترین بازدهی را در استفاده از بانک اطلاعاتی داشت. در بخش ۵ نیز نتیجه‌گیری ارائه خواهد شد.

۲- پایگاه داده کلید- مقدار

در این مدل پایگاه‌های داده، جفت‌های داده‌ای کلید-مقدار به‌صورت جدول درهم‌سازی[6] توزیع‌شده ذخیره می‌شوند. ازآنجاکه سرویس بازیابی مشابه جدول درهم‌سازی را فراهم می‌سازد، جهت بازیابی کافی است کلید موردنظر درخواست شود تا با سرعتِ بالا داده‌های ذخیره‌شده برگشت داده شود. این سرعت بالا به دلیل ساده بودن عملکرد بازیابی اطلاعات در این نوع بانک‌های اطلاعاتی است [۱۲]. در این نوع پایگاه داده نوع داده‌ای که در قسمت مقدار واردشده است اهمیتی نداشته و سیستم مدیریت مقدار مورد نظر را به برنامه تحویل داده و در برنامه عملیات مورد نیاز جهت نشان دادن آن داده صورت می‌پذیرد.

در پایگاه‌های داده کلید-مقدار، هر ماشین عضو خوشه پایگاه داده موجود در سیستم می‌تواند مقدار مرتبط با یک کلید را به‌صورت بهینه بازیابی کند. مسئولیتِ نگهداری نگاشت کلیدها و مقدارها در میان گره‌های موجود توزیع‌شده است، به‌طوری‌که هرگونه تغییر در مجموعه گره‌ها، کم‌ترین میزان قطع شدن خدمات سیستم را به بار خواهد آورد [۱۳]. از کاربردهای معمول این نوع بانک اطلاعاتی، پنهان کردن محتوا، ارجاعات سریع به داده‌ها، نمایه‌گذاری و ساختار کلید-مقدار است که در مقدار آن می‌تواند قالب‌های مختلف ازجمله JSon, BSon, XML را داشت که در ارتباطات برنامه و بانک اطلاعاتی مؤثر است.

مزیت این پایگاه‌های داده در مقابل بانک‌های رابطه‌ای، تأخیر کم، توزیع‌پذیری و جایگزینی پذیری بالا است که با استفاده از یک واسط برنامه کاربردی[7] ساده قابل‌دسترسی هستند. از ضعف‌های این نوع می‌توان عدم داشتن نمای طراحی پایگاه داده را نام برد [۱۴]. در جدول ۱ نام و میزان مقبولیت ده بانک اطلاعاتی که ساختار کلید-مقدار دارند مشاهده می‌شوند.

جدول ۱: بانک‌های اطلاعاتی کلید- مقدار [۱۵]

نام	رتبه	رتبه در کل۱
Redis	۱	۱۲
Memcached	۲	۲۲
Amazon DynamoDB2	۳	۲۸
Riak	۴	۳۱
Ehcache	۵	۳۹
Hazelcast	۶	۴۰
OrientDB2	۷	۵۱
Oracle Coherence	۸	۵۶
Berkeley DB	۹	۵۹
Aerospike	۱۰	۶۲

۱- منظور، رتبه در بین کل پایگاه‌های داده اعم از رابطه‌ای، ستون گسترده، مبتنی بر گراف، کلید-مقدار و سندگرا است [۱۵].

۲- این نوع از بانک‌های اطلاعاتی علاوه بر ساختار کلید-مقدار از ساختارهای دیگر هم پشتیبانی می‌کنند*

*- http://nosql-database.org/index.html

۲-۱- مقیاس‌پذیری[8] و سازگاری[9] در بانک‌های کلید-مقدار

بانک‌های اطلاعاتی کلید-مقدار به دلیل عملکرد ساده بسیار مقیاس‌پذیر هستند. برخلاف بانک‌های اطلاعاتی رابطه‌ای، بانک‌های کلید-مقدار به‌صورت عمودی رشد

6- Hash

7- API (Application programming interface)
8- Scalability
9- Consistency

جدول دانشجو

ID	Name	Family	Age
2536	Saman	Keshvari	Null
2537	Ali	Null	24

جدول واسط دانشجو و کتاب

Student_id	Book_id
2536	4182
2537	4023

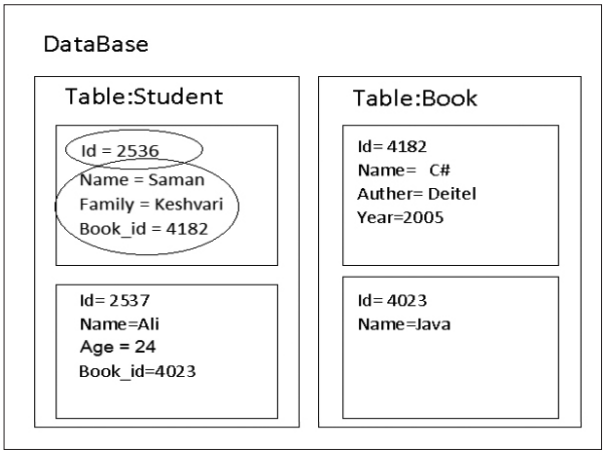
جدول کتاب

ID	Name	Author	Year
4182	C#	Deitel	2005
4023	Java	Null	Null

شکل ۲: نحوه طراحی بانک اطلاعاتی کتابخانه در ساختار رابطه‌ای

نمی‌کنند. این دسته از بانک‌های اطلاعاتی بدون نیاز به طراحی مجدد بین چندین ماشین یا دستگاه گسترش[10] پیدا می‌کنند. مقیاس‌پذیری در این بانک‌های اطلاعاتی دارای هزینه پایین‌تری نسبت به بانک‌های اطلاعاتی رابطه‌ای است [۱۶]. جهت گسترش پایگاه داده‌های رابطه‌ای گاهی نیاز به توقف کارساز جهت تغییر الگوی[11] پایگاه داده وجود دارد. یک پایگاه داده کلید-مقدار منحنی هزینه را خطی نگه‌داشته و از بروز منحنی نمایی جهت افزایش هزینه جلوگیری می‌کند، این نتیجه به دلیل طراحی این نوع بانک‌های اطلاعاتی است زیرا به‌گونه‌ای طراحی‌شده‌اند که بدون هیچ‌گونه الگوی از پیش تعیین‌شده‌ای داده‌ها را مدیریت نمایند. برخی از پایگاه داده‌های کلید-مقدار، مانند دیگر پایگاه داده‌های NoSQL، برای سازگاری نهایی طراحی‌شده‌اند. به این معنی که نسخه‌ای که از داده‌ها در بین گره‌های مختلف ذخیره‌شده‌اند بلافاصله به‌روز شوند. این در حالی است که در برخی از این نوع بانک‌های اطلاعاتی این‌گونه نیست.

در بانک‌های اطلاعاتی کلید مقدار انتخاب کلید از اهمیت بالایی برخوردار است، ممکن است توسعه‌دهنده از کلید افزایشی جهت مقداردهی به کلیدها استفاده کند مانند عملکردی که کلید اصلی در بانک‌های اطلاعاتی رابطه‌ای ایفا می‌کند که این موجب کاهش کارایی و افزایش میزان خطا هنگام توزیع شدن بانک اطلاعاتی در میان کارسازهای مختلف شود؛ به‌عنوان راه‌حل می‌توان از تنها یک کارساز جهت افزایش مقادیر شناسه‌ها استفاده نمود که این خود می‌تواند باعث ایجاد گلوگاه در این کارساز شود. راه‌حل دیگر استفاده از شناسه‌های اتفاقی و طولانی مانند MD5 یا SHA1 است که جهت توسعه سیستم کارایی بالاتری نسبت به حالت قبلی دارند در استفاده ازاین‌گونه کارها ردیس



شکل ۳: نحوه طراحی بانک اطلاعاتی کتابخانه در ساختار کلید- مقدار

به دلیل سادگی مدل داده‌ای و سرعت بالا عملکرد بسیار خوبی دارد.[12]

۲-۲. نحوه ذخیره‌سازی بانک‌های کلید-مقدار

اگر پایگاه داده کتابخانه شکل ۲ فرض شود، در ساختار کلید- مقدار کافی است برای دانشجویان و کتاب‌ها یک کلید در نظر گرفته شود. به‌عنوان‌مثال در شکل ۳ کلید برای دانشجویان و کتاب‌ها یک حوزه (id) نشان داده‌شده است ولی قسمت مقدار آن‌ها با یکدیگر متفاوت است. به‌عنوان‌مثال برای کتاب پارامترهای متفاوت از دانشجو وجود دارد. حتی در موجودیت دانشجو نیز امکان وجود حوزه‌های متفاوتی وجود دارد مثلاً برای دانشجوی اول: نام، نام خانوادگی، شناسه کتاب ولی برای دانشجوی دوم: نام، سن و شناسه کتاب در نظر گرفته‌شده درصورتی‌که در سیستم رابطه‌ای در خانه‌های خالی ردیف‌های جدول، مقادیر Null قرار می‌گیرد که این موجب بهنجار نبودن بانک اطلاعاتی می‌شود. باید توجه داشت که مقدار کلید به ازای هر موجودیت در بانک اطلاعاتی کلید-مقدار به‌صورت یکتاست [۱۷].

10- Scale Over
11- Schema
12- http://redis.io/documentation

۳- ساختار چندین بانک اطلاعاتی کلید-مقدار

در این مقاله، به ساختار چهار بانک اطلاعاتی کلید-مقدار پراستفاده مطابق با جدول ۱ پرداخته شده است. این بانک‌های اطلاعاتی رایاک[۱۳]، ردیس[۱۴]، ممکش[۱۵] و داینامو آمازون[۱۶] هستند که در ادامه ساختار و ویژگی‌های هرکدام آمده است.

۳-۱- بانک اطلاعاتی رایاک

پایگاه داده رایاک به‌عنوان یک راه‌حل برای مشکلات داده‌های حجیم بر اساس طراحی داینامو آمازون[۱۷] در سال ۲۰۰۹ توسط شرکت فناوری باشو[۱۸] ارائه شد [۱۸]. رایاک تحت دو نوع مجوز است که یک مجوز تحت لیسانس آپاچی۲ به‌صورت متن‌باز ارائه‌شده است و مجوز دیگر مربوط به شرکت باشو است که توانایی نظارت بر SNMP را به توسعه‌دهنده می‌دهد. تفاوت دیگر این دو در میزان انتشار داده‌ها در کارسازهای مختلف است. رایاک در سیستم‌عامل‌های مختلفی همچون لینوکس، مک، BSD و سولاریس[۱۹] اجرا می‌شود و واسط برنامه کاربردی آن تحت پروتکل HTTP با زبان‌های مختلفی وجود دارد [۱۹]. رایاک توسط زبان برنامه‌نویسی ارلنگ[۲۰] نوشته‌شده است که از ویژگی‌های این زبان می‌توان به همزمانی، ارتباطات توزیعی جامد[۲۱]، تحمل‌پذیری خطا نام برد. وبگاه‌ها و شرکت‌های بزرگی همچون Github, Ask.com, Yandex[۲۲], Comcast, Voxer, Rovio[۲۳], Disqus, Comcast, که دارای سیستم‌های بزرگ ذخیره‌سازی چندین ترابایت داده هستند، از رایاک استفاده می‌کنند.

طراحی داینامو به‌گونه‌ای است که توانایی پاسخ به درخواست‌های در مقیاس بالا را دارد. رایاک توانایی تحمل خطای بالایی از طریق تکرار خودکار و توزیع داده‌ها در سراسر خوشه‌های سیستم توزیع‌شده دارد؛ مانند داینامو، رایاک نیز به‌سرعت به درخواست‌های خواندن و نوشتن با حجم بالا در حد ترابایت پاسخ می‌دهد. پایگاه داده رایاک توانایی ذخیره‌سازی هر نوع داده‌ای در قسمت مقدار را دارد که این موجب شده که این پایگاه داده به‌راحتی می‌تواند رشد کند. در برنامه‌های کاربردی که به خرابی بسیار حساس هستند، یا برنامه‌هایی که دارای حجم، سرعت و تنوع بالا در داده‌ها هستند بهتر است از رایاک استفاده کنند. این بانک اطلاعاتی علاوه بر این‌که از سازوکار نگاشت-کاهش استفاده می‌کند، از جستجوی کامل با استفاده از یک الگوریتم قوی[۲۴] نیز پشتیبانی می‌کند. در رایاک داده‌ها به دو صورت می‌توانند ذخیره شوند، زمانی که نیاز به ذخیره دائم داده است داده‌ها بر روی دیسک ذخیره می‌شوند، درصورتی‌که نیاز به پردازش‌های درون حافظه‌ای بود می‌توان رایاک را به‌گونه‌ای تنظیم کرد که از حافظه جهت ذخیره‌سازی داده‌ها استفاده کند.

تفاوتی که این پایگاه داده با سایر پایگاه‌های داده‌ای NoSQL دارد در انتخاب گره اصلی است که اگر ارتباط با گره اصلی قطع شود، نزدیک‌ترین گره همسایه وظیفه سرویس‌دهی را بر عهده می‌گیرد. در رایاک می‌توان روی مقادیر کلیدها نیز شاخص تعریف کرد که سرعت جستجوی مقادیر در آن بسیار بالاست [۱۴]. در نسخه دوم رایاک امکانات خوبی جهت جستجوی متن تعبیه‌شده است. امکان اجرای پردازش‌های دسته‌ای[۲۵] از طریق اسپارک[۲۶] و اتصال به ردیس هم در رایاک فراهم‌شده است.[۲۷]

۳-۲- بانک اطلاعاتی ردیس

نسخه اول ردیس در سال ۲۰۰۹ توسط سالواتوره[۲۸] ایجادشده است [۲۰]. ردیس یک کارساز پایگاه داده باقابلیت خوشه‌بندی، متن‌باز و درون حافظه اصلی است.

13- Riak
14- Redis(Remote Dictionary Server)
15- Memcached
16- Amazon DynamoDB
17- Amazon Dynamo
18- Basho Technologies
19- Solaris
20- Erlang
21- Solid distributed Communication

۲۲- موتور جستجو روسی
۲۳- شرکت تولیدکننده بازی پرنده‌های خشمگین (Angry Birds)

24- Robust
25- Batch Processing
26- http://basho.com/products/spark/
27- http://docs.basho.com
28- Salvatore Sanfilippo

ردیس تا جولای ۲۰۱۵ توسط Pivotal, VMware پشتیبانی می‌شد ولی هم‌اکنون توسط آزمایشگاه ردیس[۲۹] پشتیبانی می‌شود. ردیس توسط زبان ANSI C نوشته‌شده است و بر روی سیستم‌عامل‌های لینوکس، مک BSD و ویندوز اجرا می‌شود. برای ردیس کتابخانه‌هایی به زبان‌های مختلفی نوشته‌شده است.[۳۰]

ردیس توانایی ذخیره‌سازی و پاسخگویی به درخواست‌ها توسط حافظه اصلی را دارد که موجب شده این بانک اطلاعاتی با سرعت بالا عمل کند [۲۱]. این بانک اطلاعاتی امکان ذخیره‌سازی داده‌ها بر روی حافظه جانبی را نیز دارد که مانند اغلب بانک‌های اطلاعاتی NoSQL با تکثیر داده‌ها موجب یکپارچگی آن‌ها می‌شود. علاوه بر استفاده از ردیس به‌عنوان یک پایگاه داده کلید مقدار می‌توان از آن به‌عنوان کارساز ساختمان داده[۳۱] نیز استفاده نمود. از نسخه ۲,۶ ردیس به بعد زبان پرس‌وجو لوآ[۳۲] به‌صورت توکار[۳۳] در آن تعبیه‌شده است که موجب شده پرس جوهای پیشرفته‌ای به‌راحتی روی این بانک اطلاعاتی اعمال نمود. زمانی که تعداد کاربران یک سامانه بالا رود و حجم درخواست‌ها زیاد شود، باید با استفاده از یک تکنیک خرد کردن مناسب یک کلید تولید کرد که بر اساس آن درخواست‌هایی که به کارسازهای مختلف می‌شود، پاسخ داده شود. نقطه قوت ردیس امکان استفاده از تکنیک‌های مختلف برای خرد کردن است.

تفاوت عمده بین ردیس و دیگر سیستم‌های پایگاه داده این است که ردیس علاوه بر مقدار رشته، مقادیر و ساختمان داده‌های زیر را نیز پشتیبانی می‌کند [۲۲]:

- رشته
- فهرستی از رشته‌ها[۳۴]
- مجموعه‌ای از رشته‌ها[۳۵]
- جداول درهم‌سازی[۳۶]

29- Redis Lab
30- http://redis.io/clients
31- Data Structure Server
32- Lua
33- Built-In
34- Lists
35- Sets
36- Hash Tables

- HyperLogLogs[۳۷]

۳-۳. بانک اطلاعاتی مِمکَش

ممکش در سال ۲۰۰۳ توسط شرکت Danga Interac-tive با زبان C نوشته‌شده و به‌صورت متن‌باز تحت مجوز BSD ارائه شد [۲۳]. این بانک اطلاعاتی در اصل برای وبگاه مجله لایوجورنال[۳۸] ایجاد شده که در سیستم‌عامل‌های Free-BSD، لینوکس، یونیکس، ویندوز و OS X اجرا می‌شود. نمونه وبگاه‌ها و شرکت‌هایی همچون Wikipedia, Youtube, Flickr, Bebo, Twitter, Typepad, Yellowbot, LiveJournal Digg, WordPress.com, Craigslist, Mixi از ممکش در طراحی خود استفاده می‌کنند.[۳۹]

ممکش یک سیستم با کارایی بالا، با توانایی نهان‌سازی شیء با حافظه توزیع‌شده برای استفاده در بالا بردن سرعت برنامه‌های وب پویا[۴۰] با کاهش بار پایگاه داده است. این بانک اطلاعاتی، در مدیریت داده‌هایی نظیر متغیرهای یک نشست[۴۱] در وب، قفل‌های داده‌ای و آمار کوتاه‌مدت به‌خوبی عمل می‌کند. ممکش تنها برای وبگاه‌هایی مفید است که تعداد درخواست‌های زیادی در واحد زمان از چندین کاربر برخط به سمت کارساز می‌رود، درحالی‌که در هر درخواستی چندین بار از پایگاه داده استفاده می‌شود. ممکش جهت ذخیره‌سازی شمارنده‌های موقت و صفحات HTML ارائه‌شده به‌خوبی عمل می‌نماید زیرا تنها از حافظه اصلی که سرعت غیرقابل مقایسه با حافظه جانبی که بانک‌های اطلاعاتی جهت نگهداری مقادیر بهره می‌گیرند، استفاده می‌کند. این بانک اطلاعاتی برای استفاده در برنامه‌هایی که درخواست زیادی داشته طراحی‌شده و در پردازش‌ها از چند ریسه‌ای[۴۲] استفاده می‌کند، این ویژگی موجب شده که به درخواست‌ها بسیار سریع پاسخ دهد. استفاده ممکش از حافظه را می‌توان محدود کرد، این بخش‌بندی استفاده از حافظه می‌تواند به امنیت داده‌های

۳۷- این مدل داده برای تخمین زدن و برآورد تقریبی کاردینالیتی به کار می‌رود.
38- http://www.livejournal.com/
39- code.google.com/p/memcached/wiki/NewStart
40- Dynamic Web Application
41- Session
42- Multi thread

جدول ۲: دستورات بانک اطلاعاتی ممکش

دستور	عملیات
Get	برگشت دادن مقادیر کلید داده‌شده به دستور
Set	ذخیره داده (جایگزینی در صورت وجود داده)
Add	ذخیره داده در صورت عدم وجود داده از قبل
Replace	جایگزینی مقدار در مقدار کلید داده‌شده به دستور
append	اضافه کردن بایت به مقدار موجود
Prepend	عکس عملیات append
Cas	مخفف Compare-and-Swap

کارساز در صورت اجرای چندین وبگاه یا برنامه بر روی آن، کمک بسزایی کند.

عملیات نوشتن در مقیاس بزرگ در ممکش به‌خوبی صورت می‌پذیرد. برای داده‌های پویایی که در آن‌ها تغییرات زیادی صورت نمی‌پذیرد این بانک اطلاعاتی به‌خوبی عمل کرده که این ویژگی موجب عملکرد خوب آن در خواندن اطلاعات شده است، مثلاً در پروفایل کاربری که در شبکه‌های اجتماعی استفاده می‌شود اطلاعاتی نظیر نام، عکس و نام مستعار یا در وبگاه‌ها اطلاعاتی نظیر درباره ما یا تماس با ما دارای اطلاعاتی هستند که تغییر چندانی ندارند، لذا جهت ذخیره‌سازی آن‌ها می‌توان از ممکش استفاده نمود. در این پایگاه داده درواقع جفت کلید و مقدار با استفاده از جدول درهم‌سازی ذخیره‌سازی می‌شود که قسمت مقدار می‌تواند توسعه پیدا کند. حداکثر اندازه کلید ۲۵۰ بایت، شامل تمام نویسه‌ها غیر از نویسه‌های کنترلی و فاصله[43] است. حداکثر مقدار در ممکش ۱ مگابایت بوده که شامل داده‌های دلخواه کاربر است. ممکش علاوه بر این‌که داده‌ها را در سیستم‌های مختلف توزیع می‌کند از حافظهٔ نهان توزیع‌شده نیز بهره می‌برد که موجب استفاده از سراسر فضای حافظه سیستم‌ها می‌شود [۱۴].

این بانک اطلاعاتی از خوشه‌بندی[44] حافظه جهت ذخیره‌سازی کلید و مقدار استفاده می‌کند که این خوشه‌ها ممکن است دارای فضای حافظه برابر یا متفاوت باشند، به‌عنوان‌مثال در یک میزبان[45] ممکن است دارای ۱ گیگابایت حافظه بوده درحالی‌که میزبان دیگر در خوشه دارای ۲ گیگابایت حافظه است. ممکش سربار پردازشی پایینی داشته و روی پردازنده فشار زیادی نمی‌آورد، زیرا پردازش‌ها در ممکش بیشتر در سمت مشتری[46] انجام می‌شوند. این بانک اطلاعاتی از دودسته دستور ذخیره‌سازی و بازیابی بهره می‌گیرد که دستورات آن در جدول ۲ مشاهده می‌شوند.

ممکش توانایی اجرای چندین دستور get[47] به‌صورت موازی را دارد که این ویژگی سرعت بیشتری در پاسخ‌دهی نسبت به دستور get تکی دارد [۲۴]. این بانک اطلاعاتی به‌صورت افقی می‌تواند رشد می‌کند، بنابراین می‌توان به اندازه مورد نیاز به آن میزبان اضافه کرد. جهت اضافه کردن کارساز می‌توان از سیستم فایل‌هایی نظیر AWS شرکت آمازون که به‌خوبی در معماری ابری عمل می‌نماید یا هدوپ[48] شرکت آپاچی یا نمونه‌های مشابه آن‌ها با توجه به نیاز استفاده نمود.

۳-٤. بانک اطلاعاتی داینامو آمازون

داینامو در سال ۲۰۱۲ توسط شرکت آمازون به‌صورت تجاری ارائه شد [۲۵] که سرعت تضمین‌شده‌ای در مقیاس بالای حجم ترافیک و داده دارد. منظور از حجم بالا در ترافیک تعداد زیاد درخواست خواندن و نوشتن است که به سمت پایگاه داده آمده که در عمل داینامو به‌گونه‌ای طراحی‌شده است که به تعداد ۱۰۰,۰۰۰ درخواست خواندن و ۱۰۰,۰۰۰ درخواست نوشتن در ثانیه پاسخ دهد. منظور از حجم بالا در داینامو، حجم بیش از ۱۰۰ترابایت است که داینامو تضمین کرده که با سرعت تضمین‌شده‌ای آن را مدیریت می‌کند؛ لذا زمانی ذخیره و مدیریت حجم‌هایی در این حدود نیاز است، بهترین گزینه بانک اطلاعاتی داینامو است[49]. شرکت آمازون برای برنامه‌های کوچک‌تر چندین

43- White Space
44- Clustering

45- Host
46- Client_based
47- Multi-Gets
48- Hadoop
49- https://aws.amazon.com/dynamodb/pricing/

ابزار دیگر دارد که اگر نیاز کاربر در آن حجم نبود با توجه بـــه نیازمندی خود می‌تواند یکی از آن‌ها را انتخاب کرده و استفاده نماید.

در داینامو باید هنگام ایجاد بانک اطلاعاتی واحد ظرفیت خواندن و نوشتن[50] را تعیین نمود که میزان ظرفیت خواندن و نوشـــتن به ازای هـــر ۴کیلوبایت در ثانیه را مشـــخص می‌کند؛ یعنی اگر واحد خواندن، نوشـــتن یک در نظر گرفته شود در بانک اطلاعاتی در یک ثانیه مقدار ۴ کیلوبایت داده می‌توان نوشت. خواندن اطلاعات در بانک اطلاعاتی داینامو به دو نوع خواندن قوی[51] و خواندن مشروط[52] مطرح است، در خوانـــدن قوی آمازون تضمین می‌کند که داده‌ای که در اختیار کاربر قرار می‌دهد داده به‌روز است ولی در خواندن مشروط این تضمین را نمی‌دهد که داده‌های که در اختیار کاربر قرار می‌دهد الزاماً نسخه به‌روز آن داده است. دلیل این موضوع بحث تکثیر داده‌ها[53] از داده‌هاست که آمازون هر داده را در ســـه کارســـاز ذخیره می‌کند. در این حالت اگر یک کارســـاز به هر دلیلی از کار افتاد سایر کارسازها توانایی پاســـخ به آن را دارد. در حالت خواندن قوی الزاماً داده از حداقل دو منبع از سه منبع خوانده می‌شود ولی در حالت مشروط ممکن است تنها از یک کارساز می‌خواند که الزاماً نسخه به‌روز نیست [۲۶].

در داینامـــو هنگام ایجاد بانک اطلاعاتی، طراح باید کلید درهم‌ســـازی که معادل کلید اصلی در بانک‌های اطلاعاتی رابطه‌ای اســـت را انتخاب کند. کلید درهم‌سازی را می‌توان بـــه دو صورت ســـاده و دارای محـــدوده[54] انتخاب نمود؛ تفاوت این دو زمانی مشـــخص می‌شـــود که پرس و جوی خوانـــدن از بانک اطلاعاتی اجرا شـــود، در حالتی که برای جدول درهم‌سازی و محدوده انتخاب‌شده بر اساس حوزهٔ محدوده پاســـخ‌ها مرتب می‌شـــوند ولی در حالت ســـاده هیچ‌گونه ترتیبی در پاســـخ وجود ندارد. کلید درهم‌سازی یکی از سه نوع داده رشته، عدد و دودویی است[55]. اگر نیاز به جســـتجو بر اســـاس یک حوزه دیگر غیر از کلید اصلی در برنامه وجود داشـــت باید در زمان طراحی آن حوزه را به‌عنوان شاخص ثانویه محلی[56] به داینامو معرفی کرده و تعیین کرد چه حوزه‌هایی از آن برگشت داده شوند. در این حالت داینامو برای آن قسمتی که شاخص گذاری کرده یک جدول شامل شـــاخص و ویژگی‌های تعیین‌شده در همان بخش ایجاد کرده و به درخواست‌ها پاسخ می‌دهد [۲۶]. اگر پس از طراحی بانک اطلاعاتی نیاز به جســـتجو بر اساس حوزهٔ دیگری نیز نیـــاز بود، آن حوزه را می‌توان به‌عنوان شاخص ثانویه عمومی تعریف نمود. حداکثر تعداد شاخص ثانویه عمومی در هر جدول، پنج عدد است.

نکته مهمی که این بانک اطلاعاتی را از ســـایر نمونه‌ها مجزا کرده است ارائه این بانک اطلاعاتی به‌عنوان سرویس اســـت که موجب شـــده کاربران درگیر مســـائل ریز بانک اطلاعاتی توزیع‌شده از قبیل چگونگی توزیع و ایجاد بانک اطلاعاتـــی و از همه مهم‌تـــر هزینه نگهداری نباشـــند. این ســـرویس در وبگاه شـــرکت آمازون[57] ارائه‌شده و کاربر کافی است به آن مراجعه کرده و پایگاه داده خود را ایجاد نماید. زمانی که دسترســـی به داده‌ها به‌صورت یکنواخت است – به‌گونه‌ای که دسترســـی‌ها بین بخش‌های مختلف تقسیم می‌شوند– بانک اطلاعاتی داینامو عملکرد مناسبی از خود نشـــان می‌دهد. اگر در برنامه داده‌هایی وجود داشت که در دســـترس بودن آن‌ها خیلی مهم بود بهتر اســـت از سیســـتم‌های نهان‌سازی داده همچون ممکش یا ردیس که در بخش‌های قبلی معرفی شـــدند، استفاده شود. همچنین، حافظه نهان قابل ارتجاع[58] شـــرکت آمازون در کنار بانک اطلاعاتی داینامو استفاده کرد به‌گونه‌ای که اگر داده در آن‌ها نبود آنگاه به بانک اطلاعاتی داینامو جهت واکشـــی داده‌ها مراجعه کرد. زمانی که در سیستم تراکنش، گزارش‌گیری و یا پرس‌وجوهای پیچیده‌ای نیاز بود بهتر اســـت از بانک

50- Read/Write Capacity Unit
51- Strongly consistent read
52- Eventually consistent read
53- Replication
54- Hash and Range

55-http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/DataModel.html
56- Local Secondary Index
57- http://console.aws.amazon.com
58- Elastic catch

اطلاعاتی داینامو اســتفاده نشود. با توجه به این‌که بیشتر پرس‌وجوهــای پیچیده بر روی داده‌هــای غیرفعال[۵۹] زده می‌شــود، بهتر اســت که این داده‌ها بر روی سیستم‌های دیگری منتقل‌شده و کارســازهایی که به درخواست‌ها بر روی داده‌های فعال پاسخگو هســتند را درگیر این موارد نکــرد. بانک اطلاعاتــی داینامو علاوه بر نــوع کلید مقدار به‌عنــوان یکی از بانک‌های اطلاعاتی ســندگرا نیز در نظر گرفته می‌شــود که این موجب محبوبیت آن شــده است[۶۰]. بانک اطلاعاتی داینامو از جدول اقلام داده‌ای[۶۱] تشکیل‌شده است؛ اقلام داده‌ای شامل یک کلید اصلی یا ترکیبی و تعدادی ویژگی که حداکثر ۴۰۰ کیلوبایت است، می‌شود. منظور از ویژگی ترکیب نام و مقدار است که مقادیر ممکن، اسکالر[۶۲] (عدد، رشته، دودویی و مقادیر بولی) یا مجموعه داده است. مجموعه داده خود شامل دودسته همگن و ناهمگن می‌شود که مجموعه‌های همگن شامل مجموعه‌های عددی، دودویی و رشته‌ای بوده و مجموعه‌های ناهمگن شامل فهرست‌ها و نقشه‌های ناهمگن[۶۳] هستند؛ ولی محدودیتی در ایجاد انواع داده‌ای در طراحی بانک اطلاعاتی وجود ندارد [۲۷].

۴- مقایسه بانک‌های اطلاعاتی کلید- مقدار

در این مقاله چهار بانک اطلاعاتی کلید مقدار بررســی شــد که نتایج بررســی‌ها به‌صورت مقایســه در جدول ۳ آمده اســت. معیارهای موجود در این جدول شامل موارد مدیریتی و نیازمندی‌هایی اســت که در بانک‌های اطلاعاتی توزیع شــده NoSQL مطــرح هســتند [۲][۴][۷][۹][۱۳].[۶۴] اطلاعات این جدول با توجه به تحقیقات صورت گرفته در هرکدام از بانک‌های اطلاعاتی ارائه‌شــده و مورد مقایســه قرارگرفته‌انــد. در ادامه بــه موارد مهــم در این جدول با جزئیات بیشتری پرداخته شده و درنهایت به دلیل شباهت ممکش و ردیس در ساختار و عملکرد، این دو باهم مقایسه شــده و در بخشــی جداگانه، رایــاک و داینامو به‌صورت جزئی‌تر مقایسه شده‌اند.

در رابطه با شاخص ثانویه در بخش داینامو توضیحات ارائه شــد همان‌طور که در جدول دیده می‌شــود دو بانک اطلاعاتــی رایاک و داینامو دارای این خاصیت هســتند که نشــان می‌دهد جســتجو در این بانک‌های اطلاعاتی دارای تنوع و سرعت بیشتری است.

تحمل‌پذیــری خطا در رایــاک با برچســب‌گذاری در نوشتن پشتیبانی می‌شــود، در این روش توسعه‌دهنده با تعیین مقــدار w تعداد گره‌هایی را که بــا تکثیر داده‌ها در آن‌ها عملیات نوشــتن پایان می‌یابد مشخص می‌کند[۶۵]. در بانک اطلاعاتی داینامو روشــی برای تحمل‌پذیری در برابر خطا قــرار داده‌اند و در ردیس از روش ثبت وقایع و نقطه بررسی[۶۶] برای این کار استفاده می‌شود.

جهت کنترل هم‌رونــدی در تراکنش‌ها ردیس تراکنش را به یک ریســه نگاشــت می‌کند ولی در رایاک و داینامو از روش قفــل خوش‌بینانه اســتفاده می‌شــود که با کمک ذخیره‌سازی چندگانه فراهم می‌شود. قبل از تثبیت داده‌های تغییریافته، هر تراکنش با تراکنش‌هایی که به‌صورت موازی با تراکنــش موردنظر بــر روی مجموعه داده مشــخص تغییرات ایجاد می‌کنند که ممکن است منجر به بروز تصادم شوند، بررسی می‌شــود که اگر تراکنش با تصادم همراه بــود تراکنش به حالت اولیه بازمی‌گــردد. این روش برای مجموعــه تراکنش‌هایی که در آن‌ها امــکان بروز تصادم کم اســت، به‌خوبی عمل می‌کند. در این روش بررســی و بازگشت دارای هزینه کمتری نسبت به قفل‌گذاری مجموعه داده به‌منظور دسترسی اختصاصی داراست.

تنها رایاک روش نگاشــت-کاهش را پشتیبانی می‌کند، ولی بانک اطلاعاتی داینامو از روشــی با عنوان نگاشــت-کاهش کشسانی[۶۷] پشتیبانی می‌کند که با این روش می‌توان مقادیر زیادی از داده‌ها را با توزیع محاسبات در یک خوشه

59- Archive
60- http://aws.amazon.com/documentation/dynamodb
61- Data Items
62- Scaler
63- heterogeneous
۶۴- این اطلاعات شــامل ویژگی‌هایی اســت که نســخه‌های بانک‌های اطلاعاتی در زمان نگارش مقاله دارند و ممکن است در نسخه‌های آتی تغییر کنند.
65-https://docs.basho.com/riak/kv/2.1.4/developing/app-guide/replication-properties/#w-value-and-write-fault-tolerance
66- Logging and Check Point
67- Elastic MapReduce

جدول ۳: مقایسه چهار بانک اطلاعاتی کلید-مقدار بررسی‌شده در مقاله با معیارهای مختلف

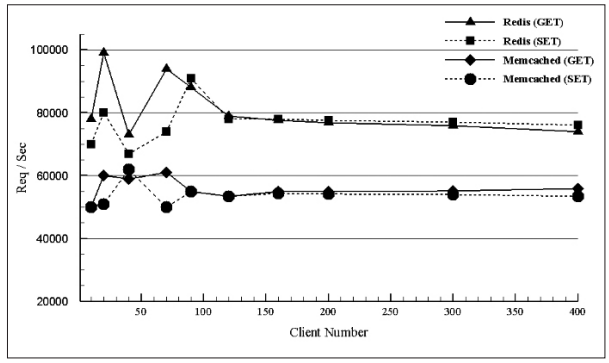
معیار	رایاک	ردیس	ممکش	بانک اطلاعاتی داینامو
توسعه‌دهنده	Basho Technologies	Salvatore Sanfilippo (Redis Lab)	Danga Interactive	Amazon
سال تولید	۲۰۰۹	۲۰۰۹	۲۰۰۳	۲۰۱۲
مجوز	متن‌باز	متن‌باز	متن‌باز	تجاری
ارائه به‌عنوان سرویس	خیر	خیر	خیر	بله
زبان پیاده‌سازی	Erlang	C	C	نامشخص
سیستم‌عامل‌های کارساز	لینوکس، OS X	BSD, OS X ویندوز، لینوکس	FreeBSD, Linux, OS X لینوکس، یونیکس، ویندوز	میزبان شده۱
پشتیبانی از نوع داده	خیر	بله (به‌صورت جزئی)	خیر	بله
روش دست‌یابی و واسط برنامه کاربردی	HTTP API, Native Erlang Interfrace	قواعد اختصاصی (RESP2)	قواعد اختصاصی	RESTful, HTTP, API
نوشتار سمت کارساز	JavaScript And Erlang	Lua	ندارد	ندارد
شاخص ثانویه	دارد	ندارد	ندارد	دارد
پشتیبانی از کلید ترکیبی	دارد	-	-	دارد
حداکثر اندازه قطعه به ازای هر کلید	64MB	512 MB	1 MB	1 MB
روش کنترل هم‌روندی	قفل‌گذاری خوش‌بینانه	ریسه‌های تکی	قفل‌گذاری ریزساختار	قفل‌گذاری خوش‌بینانه
روش تحمل‌پذیری خطا	برچسب نوشتن	ثبت وقایع، نقطه بررسی	-	دارد
نگاشت-کاهش	دارد	ندارد	ندارد	ندارد
تجزیه‌ناپذیری	دارد	دارد	دارد	دارد
ثبات (سازگاری)۳	بردار-زمان	درنهایت سازگار	دارد	درنهایت سازگار
جداسازی۴	دارد	دارد	دارد	دارد
ماندگاری۵	دارد	دارد	ندارد	دارد
سازوکار دسترسی کاربر	ندارد	کنترل با کمک رمزعبور ساده	استفاده از SASL6	کنترل به روش 7IAM AWS آمازون
جستجوی کامل متنی	دارد	ندارد	دارد	دارد
پشتیبانی از گراف	دارد	ندارد	ندارد	دارد
پیشروی افقی	دارد	ندارد	دارد	دارد
تکثیر داده‌ها	دارد	دارد	ندارد	دارد

که در کارسازهای مجازی ابری آمازون اجرا می‌شوند را تحلیل و پردازش کرد، این خوشه‌ها در چارچوب هدوپ اداره می‌شوند.

٤-١- مقایسه ردیس و ممکش

ردیس و ممکش هردو متن‌باز هستند و داده‌ها را در حافظه اصلی ذخیره کرده که این ویژگی باعث شده بانک‌های اطلاعاتی مناسبی جهت استفاده در نهان‌سازی باشند. ممکش و ردیس دارای کتابخانه‌هایی هستند که به توسعه‌دهنده اجازه می‌دهند به‌راحتی با آن‌ها ارتباط برقرار کنند که این ساده بودن راه‌اندازی آن‌ها باعث محبوبیت آن‌ها شده است. سازندگان ممکش بیشتر به ثبات و بهینه بودن تأکیددارند و توجه کمتری به افزودن ویژگی جدید داشته‌اند.

کارایی در استفاده از حافظه: برای جفت کلید-مقدار



شکل ۴: مقایسه کارایی بانک‌های اطلاعاتی ردیس و ممکش

ساده ممکش در استفاده از حافظه کاراتر است ولی اگر از درهم‌سازی‌های ردیس استفاده شود، ردیس کارآمدتر خواهد شد.

ماندگاری: در ممکش داده‌ها ناپایدار هستند زیرا در حافظه اصلی نگهداری می‌شوند و با راه‌اندازی مجدد سیستم، از بین خواهند رفت و بازسازی حافظهٔ نهان هزینه پردازش بالایی دارد ولی ردیس می‌تواند داده‌های پایدار را مدیریت کند. همگام‌سازی داده‌های حافظه اصلی و ثانویه را در کمتر از دو ثانیه انجام می‌دهد.

تکثیر داده‌ها: در ممکش تکرار داده‌ها وجود ندارد ولی در ردیس از روش فرمانده-فرمانبر[68] استفاده می‌شود که این ویژگی منجر به ایجاد نسخه‌ای از داده‌های کارساز در هرکدام از مشتری‌ها می‌شود.

نحوه ذخیره‌سازی: ممکش مقادیر را در حافظه ذخیره می‌کند و هیچ اطلاعاتی را هنگام خواندن مشتری به‌صورت مستقیم از بانک اطلاعاتی واکشی نمی‌کند. از طرفی ردیس همانند یک بانک اطلاعاتی که در حافظه مقیم است و با درخواست داده جفت کلید و مقدار را که در حافظه مقیم هستند، در اختیار کاربر قرار می‌دهند، توسعه‌دهندگان از ردیس جهت تجزیه‌وتحلیل در برنامه‌های بی‌درنگ استفاده می‌کنند.

جهت مقایسه بهتر ردیس و ممکش عملیات Set و Get در این بانک‌های اطلاعاتی در شرایط یکسان انجام‌شده که در شکل ۴ نشان داده‌شده است. در این آزمایش از نسخه ۲,۰ ردیس و ۱,۴,۵ ممکش و از یک GCC[69] یکسان استفاده‌شده است. در آزمایش داده‌های با ظرفیت ۳۲ بایتی استفاده‌شده است. با توجه به این‌که ردیس شش سال پس از ممکش به وجود آمده و دارای ویژگی‌های بیشتری نسبت به آن است، عده‌ای ردیس را نسخه توسعه‌یافته ممکش می‌دانند، ولی مطالعات ما نشان می‌دهد که در دو مورد، استفاده ممکش بهتر از ردیس عمل می‌کند: اول اگر نیاز به نهان‌سازی داده‌های ایستا و کوچک مانند تکه کدهای HTML بود به دلیل مشابهت ممکش به ردیس در استفاده از مدیریت حافظه درونی و مصرف کمتر حافظه نسبت به ردیس بهتر عمل می‌کند. در ممکش تنها نوع داده رشته وجود دارد که این موجب شده که در مرتب کردن داده‌هایی که فقط خوانده می‌شوند مناسب باشد، زیرا رشته میزان پردازش کمتری نیاز دارد. با توجه به طراحی ساده‌تر ممکش نسبت به ردیس، ممکش مقیاس‌پذیری افقی بالاتر و بهتری نسبت به ردیس دارد. در غیرازاین دو حالت بهتر است از ردیس استفاده شود.

٤-٢- مقایسه رایاک و داینامو آمازون

بانک اطلاعاتی رایاک متن‌باز است ولی بانک اطلاعاتی داینامو آمازون تحت مجوز تجاری آمازون ایجادشده است. داینامو به‌عنوان سرویس بانک اطلاعاتی را در اختیار توسعه‌دهندگان قرار می‌دهد و تنها نسخه محلی جهت راه‌اندازی و آشنایی به توسعه‌دهندگان ارائه می‌دهد، به همین دلیل ارزیابی کارایی آن نسبت به سایرین مشکل است، بااین‌حال در ادامه این دو با معیارهای مختلف با یکدیگر مقایسه شده‌اند.

ثبات داده: رایاک جهت تضمین ثبات داده از ساختار ساعت-بردار[70] استفاده می‌کند. این ویژگی باعث می‌شود که درخواست‌های مشتری همواره نوشته شوند. در زمان خواندن به‌وسیله نرم‌افزار یا کدهای مشتری، داده‌های بازیابی شده که با توجه به قدمت و اندازه نگهداری شده‌اند، به‌روزترین داده است. البته این ویژگی را می‌توان

68- Master-Slave

69- Gnu C Compiler

70- Vector-Clock

در رایـــاک غیرفعال نمـــود. در داینامو ویژگـــی «درنهایت ســـازگار شدن»[71] استفاده می‌شـــود. به این معنا که گاهی درخواســـت‌های نوشـــتن در کپی‌ها با تأخیر همراه است. در هنگام خواندن داینامو ســـعی می‌کند به‌روزترین نسخه را در اختیـــار کاربر قرار دهد ولی گاهی نســـخه‌ای که در اختیار کاربر قرار می‌گیرد الزاماً آخرین نســـخه نیســـت. داینامو دارای دستورهایی است که جهت خواندن و نوشتن دسته‌ای[72] مناسب هستند.

مدل ذخیره‌ســـازی: رایاک دارای سیستم ذخیره‌سازی محلی توسعه‌پذیر است که به توسعه‌دهندگان اجازه استفاده از افزونه ذخیره‌سازی درونی را متناسب با نیازهای پروژه می‌دهد. در داینامو اقلام داده‌ای در دیسک‌های حالت‌جامد[73] ذخیره می‌شوند و نسخه‌های آن‌ها در مناطق در دسترس نگهداری می‌شوند.

توســـعه‌پذیری: در رایاک گره فرمانـــده وجود ندارد و گره‌هـــا همانند حلقه به یکدیگر متصل هســـتند که این امر موجب می‌شـــود که گـــره جدید به‌راحتی بـــه حلقه اضافه شـــود. در داینامـــو قبل از ایجـــاد بانک اطلاعاتـــی، میزان خواندن و نوشتن و در پی آن نیازمندی‌های سخت‌افزاری و پارتیشن‌بندی مشخص می‌شوند.

۵- نتیجه‌گیری و کارهای آتی

بانک‌هـــای اطلاعاتی NoSQL به چهار دســـته ســـتون گســـترده، ســـندگرا، مبتنی بر گراف و کلید-مقدار تقسیم می‌شوند که اخیراً بانک‌های اطلاعاتی به وجود آمده‌اند که از چند مدل از این انواع پشـــتیبانی می‌کننـــد. در این مقاله تمرکز بر بانک‌های اطلاعاتی کلید-مقدار بود. با مقایســـه ویژگی‌ها و کاربردهای چهار بانک اطلاعاتی رایاک، ردیس، داینامو آمـــازون و ممکش که از نوع بانک اطلاعاتی کلید-مقدار هستند، سعی در ارائه بهترین بانک اطلاعاتی با توجه به نیازمندی‌ها و کاربرد پروژه بود. با توجه به ساختار و کاربرد مشابه، ردیس و ممکش با یکدیگر و رایاک و داینامو نیز با یکدیگر مقایسه شـــدند. با مقایسه انواع ویژگی‌های این بانک‌های اطلاعاتی در مقایسه ردیس و ممکش تنها در دو حالت، نهان‌سازی داده‌های ایستا و مقیاس‌پذیری افقی اســـتفاده از ممکش بهتر است در سایر نیازمندی‌ها ردیس بهتر عمل می‌کند. در مقایسه رایاک و داینامو به دلیل ارائه داینامو به‌صورت ســـرویس توسط آمازون درصورتی‌که امکانات ســـخت‌افزاری و راه‌اندازی بانک اطلاعاتی نیست اســـتفاده از ایـــن بانک اطلاعاتـــی به رایـــاک ترجیح داده می‌شود، ولی رایاک متن‌باز بوده و توسعه‌دهنده نسبت به آمازون دارای قدرت مانور بیشـــتری اســـت و به ساختار مشخصی محدود نیست. نکته دیگر حداکثر اندازه قطعه در این دو بانک اطلاعاتی اســـت که رایاک ۶۴مگابایت بوده و داینامو ۱مگابایت است که این ویژگی باعث شده که رایاک برای پردازش‌های دسته‌ای بهتر عمل کند. در کارهای آتی می‌توان ســـایر انواع بانک‌های اطلاعاتی را مورد ارزیابی قرار داد تا در هرکدام از انواع با توجه به نیاز بتوان بهترین انتخاب را داشت.

71- Eventual Consistency
72- Batch
73- Solid State Disk

منابع

[۱] کشوری، ســـامان. نقوی، مهدی، کشوری، ساناز، «معرفی، بررسی و مقایسه سیســـتم فایل‌های توزیع‌شـــده»، اولین همایش ملی فناوری‌هـــای نوین رایانه و توسعه پایدار، دانشگاه شهید بهشتی، تهران، ۱۳۹۴.

[2] Brewer, Eric A.: "Towards Robust Distributed Systems" Portland, Oregon, July 2000. – Keynote at the ACM Symposium on Principles of Distributed Computing (PODC) on 2000- 07-19.

[۳] کشوری، ســـامان. صابری، حسین، کشوری، ســـاناز، «نقش نظریه CAP و همزیســـتی مســـالمت‌آمیز در انتخاب بانک‌های اطلاعاتی»، ســـومین کنفرانس بین‌المللی پژوهش‌های کاربردی در مهندســـی کامپیوتـــر و فناوری اطلاعات، دانشگاه تربیت مدرس، تهران، ۱۳۹۴.

[4] Pokorný, J., "Database technologies in the world of big data." In Proceedings of the 16th International Conference on Computer Systems and Technologies (CompSysTech '15), Boris Rachev and Angel Smrikarov (Eds.). ACM, New York, NY, USA, 1-12, 2015.

[5] Abhinay B. A., Akshata B. A., Karuna C. Gull., "Growth of New Databases & Analysis of NOSQL Datastores", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 3, pp. 6, June 2013.

[6] Kempe, Sh., "UnQL: A Standardized Query Language for NoSQL Databases", [Online], Available: http://www.dataversity.net/unql-a-standardized-query-language-for-nosql-databases/ , (2012, March 29), Retrieved 26 February 2016.

[7] Srivastava, P., Goyal, S., Kumar, A., "Analysis of various NoSql database," presented at the Green Computing and Internet of Things (ICGCIoT), Noida, 2015.
[8] Nowosielski, A., Kowalski, P. A., Kulczycki, P., "The column-oriented data store performance considerations," 2016 Federated Conference on Computer Science and Information Systems (FedCSIS), Gdansk, pp. 877-881, 2016.
[9] Chandra, D. G., "BASE analysis of NoSQL database", Future Generation Computer Systems, vol. 52, pp. 13 - 21, November 2015.
[10] Oliveira, F. R., Cura, L. d. V., "Performance Evaluation of NoSQL Multi-Model Data Stores in Polyglot Persistence Applications." 20th International Database Engineering & Applications Symposium (IDEAS '16), Evan Desai (Ed.). ACM, New York, NY, USA, 230-235, 2016.
[11] Jayathilake, D., Sooriaarachchi, C., ET. Al. "A Study into the Capabilities of NoSQL Databases in Handling a Highly Heterogeneous Tree", 2012 IEEE 6th International Conference on Information and Automation for Sustainability, 2012.
[12] Yuan, X., Wang, X., Wang, C., Qian, Ch., Lin, J., "Building an Encrypted, Distributed, and Searchable Key-value Store." 11th ACM on Asia Conference on Computer and Communications Security (ASIA CCS '16). ACM, New York, NY, USA, 547-558, 2016.
[13] Han, J., Haihong, E., ET. Al. "Survey on NoSQL database", Pervasive Computing and Applications (ICPCA), 6th International Conference on Port Elizabeth. PP. 363-366, IEEE, 26-28 Oct, 2011.
[14] Redmond, E., Wilson, J. R., Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement, 352 pages, Pragmatic Bookshelf; 1 edition, 2012.
[15] solid IT gmbh. "DB-Engines Ranking of Key-value Stores"[Online]. Available: http://db-engines.com/en/ranking/key-value+store , [Accessed: 27 February 2016].
[16] Stonebraker, M. "SQL databases v. NoSQL databases", Communications of the ACM, New York, NY, USA, Vol. 53, pp. 4, 10-11, 2010.
[17] S. N., Swaminathan, R. Elmasri, "Quantitative Analysis of Scalable NoSQL Databases" IEEE International Congress on Big Data (BigData Congress), San Francisco, CA, pp. 323-326, 2016.
[18] Basho Technologies, "Distributed, fault tolerant key-value store", [Online]. Available: www.basho.com/products/riak-overview , [Accessed: 28 March 2017].
[19] Redmond, E. "A Little Riak Book", Ebook, 1.4.0 2013-06-07, 65 pages, URL: http://littleriakbook.com/ , Basho engineer, 2014.
[20] Salvatore Sanfilippo, "In-memory data structure store, used as database, cache and message broker", [Online]. Available: www.redis.io, [Accessed: 28 March 2017].
[21] Carlson, J. L. "Redis in Action." Manning Publications Co., Greenwich, CT, USA, 2013.
[22] Seguin, K., "The Little Redis Book", licensed under the Attribution-NonCommercial 3.0 Unported license, URL: http://github.com/karlseguin/the-little-redis-book , 2012.
[23] Interactive, D. "In-memory key-value store", originally intended for caching, [Online]. Available: www.memcached.org , [Accessed: 28 March 2017].
[24] Berezecki, M., Frachtenberg, E., Paleczny, M., Steele, K. "Many-core key-value store." In Proceedings of the 2011 International Green Computing Conference and Workshops (IGCC '11). IEEE Computer Society, Washington, DC, USA, 1-8, 2011.
[25] Amazon, Hosted, scalable database service by Amazon with the data stored in Amazons cloud, [Online]. Available: www.aws.amazon.com/dynamodb , [Accessed: 28 March 2017].
[26] Amazon, "Amazon DynamoDB Pricing", [Online]. Available: https://aws.amazon.com/dynamodb/details/ , [Accessed: 26 February 2016].
[27] DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., Vogels, W. "Dynamo: amazon's highly available key-value store." In Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles (SOSP '07). ACM, New York, NY, USA, 205-220, 2007.