مجله محاسبات نرم

SOFT COMPUTING JOURNAL

تارنمای مجله: scj.kashanu.ac.ir

دانشگاه کاشان
University of Kashan

بهبود زمان پاسخ‌دهی پرس‌وجوهای تحلیلی در انبارهٔ داده‌ای برخط با استفاده از الحاق دیدهای ذخیره‌شده✧

سید مجید شفائی[۱]*، کارشناسی ارشد، بابک وزیری[۲]، استادیار، سید مصطفی شفائی[۳]، کارشناسی ارشد

[۱و۲] دانشکده فنی و مهندسی، دانشگاه آزاد اسلامی واحد تهران مرکزی، تهران، ایران.

[۳] دانشکده کامپیوتر، دانشگاه تربیت دبیر شهید رجایی، تهران، ایران.

اطلاعات مقاله

تاریخچه مقاله:

دریافت ۰۷ دی ماه ۱۳۹۸

پذیرش ۲۰ شهریور ماه ۱۳۹۹

کلمات کلیدی:

انبارهٔ داده‌ای برخط

پردازش تحلیلی برخط

بخش‌بندی، دید ذخیره‌شده

مخزن داده

چکیده

یک انبارهٔ داده‌ای برخط مجموعه‌ای از داده‌های اخیر و داده‌های سلسله‌مراتبی است که برای اخذ تصمیمات، توسط مدیران از طریق ایجاد پرس‌وجوهای تحلیلی برخط مورد استفاده قرار می‌گیرد. داده‌هایی که از منابع داده‌ای واکشی می‌شوند و به انبارهٔ داده‌ای برخط وارد می‌شوند، روزبه‌روز در حال افزایش است و همچنین با افزایش حجم داده‌های ورودی به انبارهٔ داده‌ای برخط تداخل بین عملیات بارگیری و پردازش تحلیلی برخط بیش از پیش افزایش پیدا می‌کند. این دو چالش به مهم‌ترین مسائل در زمینهٔ انبار داده‌ای برخط تبدیل شده‌اند. در این مقاله، روشی برای بهبود زمان پاسخ‌دهی پرس‌وجوهای تحلیلی در معماری انبارهٔ داده‌ای برخط با استفاده از الحاق دیدهای ذخیره‌شده ارائه شده است. فرایند کار بدین صورت است که نتایج پرس‌وجوهای اجراشده در هر بخش برخط ذخیره می‌گردند و در هنگام انتقال داده به بخش بعدی، این نتایج از قبل محاسبه‌شده نیز منتقل می‌شوند. در هنگام انتقال داده، هر بخش برخط حاوی مجموع دادهٔ بخش قبلی خود در چندین مرحله از انتقال است. در نتیجه با انتقال داده، نتایج محاسبه‌شدهٔ پرس‌وجوها نیز منتقل می‌شوند و می‌توان بدون نیاز به اجرای دوبارهٔ پرس‌وجوها، نتایج محاسبه‌شدهٔ قبلی را با هم الحاق کرد و به نتیجهٔ دلخواه رسید. روش پیشنهادی منجر به کاهش زمان پاسخ‌دهی به پرس‌وجوهای تحلیلی و کاهش تداخل ورود داده با اجرای هم‌زمان و طولانی‌مدت پرس‌وجوها شده است. چالشی که این پژوهش با آن مواجه است، این است که روش پشنهادی بر روی حجم کمی از داده در بخش برخط، مورد استفاده قرار گیرد و همچنین چالش بعدی، شامل تغییرات مورد نیاز برای استفاده در دادهٔ بزرگ است.

۱. مقدمه

امروزه رقابت شدیدی بین مؤسسه‌ها و شرکت‌ها برای ارائهٔ بهترین خدمات به مشتری‌ها وجود دارد؛ ازاین‌رو، مدیران برای اخذ تصمیمات خود نیاز به سامانه تصمیم‌یار دارند. امروزه یکی از این سامانه‌های تصمیم‌یار انبارهٔ داده‌ای است که می‌توان از آن برای به‌دست‌آوردن نتایج تحلیلی بر روی داده‌های سلسله‌مراتبی (تاریخی) استفاده کرد.

رکوردها یا داده‌ها از طریق ابزاری به نام ETL[1] (استخراج، تغییر شکل و بارگیری) به‌روزرسانی می‌شوند. فرایند ETL

✧ نوع مقاله: پژوهشی

* نویسنده مسئول

پست‌های الکترونیک: sm.shafaei71@gmail.com (شفائی)

b.vaziri@iauctb.ac.ir (وزیری)

s.m.shafaei@outlook.com (شفائی)

1. Extract Transform and Load

نحوه ارجاع به مقاله: شفائی، سید مجید، وزیری، بابک، شفائی، سید مصطفی، «بهبود زمان پاسخ‌دهی پرس‌وجوهای تحلیلی در انباره داده‌ای برخط با استفاده از الحاق دیدهای ذخیره‌شده»، مجله محاسبات نرم، جلد ۸، شماره ۲، ص۲۲ـ۳۷، پاییز و زمستان ۱۳۹۸.

مسئول شناسایی، استخراج داده‌های مرتبط از منابع سامانهٔ پردازش تراکنش برخط، سفارشی‌سازی و ادغام این داده‌ها به داخل یک فرمت مشترک، تمیزکاری داده و همانند کردن آن‌ها به قالب مناسب، یکپارچه‌سازی برای بروزرسانی محیط دادهٔ انبارهٔ داده‌ای و در نهایت بارگیری دادهٔ قالب‌بندی‌شدهٔ نهایی به داخل پایگاه داده است [۱]. همچنین فازی به نام CDC[1] وجود دارد که چگونگی به‌دست‌آوردن (یا ضبط) تغییرات داده در منابع داده را انجام می‌دهد.

انبارهٔ داده‌ای سنتی داده‌های خودش را به‌طور دوره‌ای با زمان طولانی (برای مثال ماهانه، هفتگی و یا روزانه) در ساعات بارکاری کم (برای مثال در نیمه‌شب) به‌روزرسانی می‌کند.

انبارهٔ داده‌ای برخط ترکیبی از یک انبارهٔ داده‌ای و رفتار برخط است [۲]. در این نوع معماری، زمان به‌روزرسانی داده نسبت به معماری سنتی انبارهٔ داده‌ای کوتاه است (برای مثال از دقیقه تا ساعت و...). پرس‌وجوهای تحلیلی برخط بر روی انبارهٔ داده‌ای و بخش(های) برخط صورت می‌گیرد. اما در طراحی یک انبارهٔ داده‌ای برخط، با چندین چالش بزرگ مواجهیم. یکی از این چالش‌ها، کاهش زمان پاسخ‌دهی به پرس‌وجوهای تحلیلی برخط کاربران است. به‌دلیل اینکه اگر پرس‌وجوهای تحلیلی بر حجم وسیعی از انبارهٔ داده‌ای ایستا و بخش‌های برخط که دارای حجم زیادی از داده‌های ورودی هستند انجام گیرد، زمان انتظار پاسخ کاربر افزایش پیدا می‌کند. یکی دیگر از این چالش‌ها تداخل بین عملیات تحلیلی برخط با بارگیری داده است؛ زیرا در اینجا داده‌ها به‌صورت تقریباً برخط با زمان به‌روزرسانی پایین از منابع خارجی واکشی شده و به انبارهٔ داده‌ای بارگیری می‌شوند. پژوهش‌هایی در این زمینه در [۳ـ۶] صورت گرفته است که هرکدام توانسته‌اند تا حدودی این مشکلات را حل کنند.

برای تبدیل یک انبارهٔ داده‌ای سنتی به یک انبار داده‌ای برخط می‌توان یک بخش [۱، ۷، ۸ و ۹] یا چندین بخش [۲، ۱۰، ۱۱ و ۱۲] برخط به آن اضافه کرد. این بخش(ها) برخط ساختاری مشابه با انبارهٔ داده‌ای دارند. در این بخش‌ها هیچ گونه اندیس، کلید اصلی و قید از هر نوع ـ شامل قید جامعیت ارجاع- تعریف نمی‌شود [۱]. هرکدام از این بخش‌ها دارای یک زمان تأخیر برای بارگیری داده هستند. بدین صورت که بعد از سپری شدن این زمان، شروع به انتقال داده به بخش بعدی یا انبارهٔ داده‌ای ایستا می‌کنند. در معماری چندبخشی برخط، تعداد و زمان بارگیری یا تأخیر این نوع بخش‌ها به‌صورت پویا تعیین می‌شود [۲ و ۱۲].

در این پژوهش، راه‌حلی مؤثر برای بهبود پرس‌وجوهای عملیات تحلیلی برخط کاربران و کاهش مشکل تداخل بین عملیات تحلیلی برخط با بارگیری داده ارائه شده است. در رویکرد پیشنهادی با ایجاد دیدهای ذخیره‌شده از نتایج محاسبه‌شده و استفادهٔ آن‌ها در بخش‌های برخط و انتقال و الحاق آن‌ها با یکدیگر به سایر بخش‌های برخط باعث کاهش سرعت پاسخ‌دهی به پرس‌وجوهای تحلیلی کاربران شده است و در صورت استفاده از این دیدهای ذخیره‌شده، نیازی به مراجعه به بخش(های) برخط نیست و در این صورت عملیات بارگیری داده با عملیات تحلیلی برخط تداخلی ندارند. برای استفاده از الحاق دیدهای ذخیره‌شده می‌توان از عملگر Union استفاده کرد که باعث می‌شود دیدهای مشترک یک پرس‌وجو با یکدیگر (که در چند مرحلهٔ انتقال ذخیره شده‌اند) پیوند خورده و نتیجهٔ پرس‌وجو بدون نیاز به اجرای آن در بخش‌های برخط تهیه شود.

اجرای روش پیشنهادی در بخش‌های برخط با حجم پایین کارایی کمی دارد و حتی شاید باعث کاهش زمان پاسخ‌دهی شود. در روش پیشنهادی بایستی بخش‌های برخط دارای حجم نسبتاً زیادی باشند و اجرای الگوریتم روی بخش‌های برخط انتهایی بهترین کارایی را دارد؛ زیرا این بخش‌های برخط دارای حجم دادهٔ زیاد و تعداد دیدهای ذخیره‌شدهٔ بالایی (نتایج محاسبه‌شدهٔ بخش‌های برخط قبلی) هستند.

کمبود بعدی این پژوهش که احتمالاً با آن مواجه است، اجرای روش پیشنهادی بر روی دادهٔ بزرگ[2] و روش‌های مربوط به پیاده‌سازی و تست است. برای مثال می‌توان از Hadoop، اسپارک و... برای ارزیابی روش پیشنهادی بر روی دادهٔ بزرگ استفاده کرد.

1. Change Data Capture

2. Big Data

ادامهٔ مقاله بدین شرح سازماندهی شده است. بخش ۲ شامل پژوهش‌های پیشین در زمینهٔ انبارهٔ داده‌ای برخط ارائه شده است. بخش ۳ رویکرد پیشنهادی جدید شرح داده شده است، بخش ۴ شامل ارزیابی نتایج به‌دست‌آمده از رویکرد پیشنهادی و بخش آخر شامل نتیجه‌گیری است.

۲. پیشینهٔ پژوهش

Weiping Qu و همکارانش در [۳]، مدل تصویر لحظه‌ای را که در سمت منبع جریان ETL مستقر است، در کار خود گسترش داده‌اند. تصویر لحظه‌ای بر روی جداول انبارهٔ داده‌ای گرفته می‌شود تا بتوان یک پرس‌وجوی معین را مستقل از به‌روزرسانی‌های هم‌زمان پاسخ داد. قبل از پاسخ‌گویی به یک پرس‌وجو، ابتدا جداول مربوط در لحظه‌ای که پرس‌وجو واردشده است، دقیقاً با تعداد دلتا رکورد از منبع بروز می‌شوند. نگهداری تصویر لحظه‌ای به‌وسیلهٔ یک خط لولهٔ دوبارهٔ محاسبه‌شوندهٔ افزایشی انجام می‌شود که از طریق یک مجموعه دلتا (مجموعه‌ای از تعداد رکورد) پی‌درپی متعلق به پرس‌وجوهای واردشونده، پر می‌شود. همچنین از یک زمان‌بند برای کار خود استفاده کرده‌اند.

Wei He و Lizhen Cui در [۷]، یک روش پیشنهادی برای انجام موازی پردازش تحلیلی برخط بر روی کارایی نودها مطلع ارائه کرده‌اند. در این پژوهش دربارهٔ یک رویکرد موازی سبک‌وزن به همراه یک پیش‌بینی کارایی و یک مدل درجه‌بندی کردن داده بر طبق خصوصیات نودها بحث شده است. بر این اساس، کدام جداول حقیقت با مقیاس بزرگ بایستی به چندین نود از طریق فرایند ETL موازی جزءبندی بشوند و بتوان پرس‌وجوهای تحلیلی را به‌صورت موازی بر روی این نودها که داده در آن‌ها بخش‌بندی شده‌اند، انجام داد.

Tanvi Jain و همکاران در [۱۳]، روشی پیشنهاد داده‌اند که در آن دادهٔ حیاتی از غیرحیاتی تمیز داده می‌شوند. بعضی از داده‌های حیاتی نیاز به‌روزرسانی به شیوه‌ای برخط دارند. در کار آن‌ها یک رویکرد برای تشخیص دادهٔ حیاتی به‌وسیلهٔ دو عامل «تأثیر از یک به‌روزرسانی» و «تعداد رکوردهای متأثر» انجام می‌شود. دادهٔ حیاتی تشخیص داده‌شده در جداول موقت ذخیره می‌شوند. این جداول موقت به‌صورت برخط به‌روز می‌شوند (برای مثال روزانه) و داده‌های باقی‌مانده به‌صورت عادی به‌روز خواهند شد (برای مثال هفتگی).

Yingchi Mao و همکاران در [۴]، روشی ترکیبی از محیط ذخیره‌سازی پویا خارجی و یک فناوری تکرار آینه‌ای پویا برای حل رقابت بین پرس‌وجوهای تحلیلی برخط و به‌روزرسانی پردازش تحلیلی برخط ارائه کرده‌اند. همچنین برای پشتیبانی از تحلیل داده برخط، ETL را به دو نوع تقسیم کرده‌اند: ETL برخط و ETL تاریخچه‌ای یا سنتی.

Issam Hamdi و همکاران در [۵]، ابتدا الگوریتم انتخاب پویا دیدهای ذخیره‌شده را ارائه کرده‌اند که این دیدها از نتایج پرس‌وجوهای تحت قید زمان اجرا و قید فضای ذخیره‌سازی سیستم انتخاب می‌شوند. سپس یک سیاست جدید برای مشخص کردن نگهداری پویا دیدهای ذخیره‌شده بر طبق تناوب دسترسی و بارکاری سیستم پیشنهاد داده‌اند.

Ziyu Lin و همکاران در [۶]، روش جدیدی به نام دید مبتنی بر لایه پیشنهاد داده‌اند که در آن، سازگاری گزارش‌گیری از داده به شکلی مؤثر و مناسب حفظ می‌شود. و ایدهٔ اصلی، جلوگیری داده از درگیر شدن یک پرس‌وجو پردازش تراکنش برخط از تغییر کردن با استفاده از مکانیسم قفل‌گذاری و پرهیز از برخورد عملیات خواندن و نوشتن با کمک مکانیسم لایه‌ای است.

M. Asif Naeem و همکاران در [۱۰]، معماری و روشی برای تقسیم کردن داده‌های ورودی به دو نوع که شامل کلی که اغلب تغییر نمی‌کنند (یا در زمان‌های طولانی تغییر پیدا می‌کنند)، و دادهٔ تراکنشی که به‌طور منظم به‌روز می‌شوند، ارائه کرده‌اند. برای مثال جدول محصولات به‌عنوان دادهٔ کلی و جدول سفارش به‌عنوان دادهٔ تراکنشی در نظر گرفته می‌شود. به همین دلیل، دادهٔ کلی که حجم کم و تقریباً ثابتی دارد، می‌تواند به‌راحتی کش شود و سرعت عملیات بالا رود.

Shao YiChuan و Xingjia Yao در [۱۱]، استراتژی انبارهٔ داده‌ای برخط را که مبتنی بر مکانیسم تکرار آینهٔ دوبل است، معرفی کرده‌اند. این استراتژی از دو مرحله تشکیل شده است. ابتدا از مکانیسم تکرار آینهٔ دوبل برای توانایی در بارگیری

پیوستهٔ داده در انبارهٔ داده‌ای برخط با کمترین تأثیر در زمان اجرای پرس‌وجو استفاده کرده‌اند. سپس میانگیرهای چندسطحی را به داخل ساختار انبارهٔ داده‌ای که مبتنی بر جزءبندی برخط است یکپارچه کرده‌اند و فرایند طراحی و پیاده‌سازی را با جزئیات ارائه داده‌اند.

قسمت عمدهٔ تحقیقات وابسته به انبارهٔ داده‌ای به مدیریت اختصاصی انبارهٔ داده‌ای می‌پردازد. اکثر تحقیقات بر روی ابزار جلویی و ادبیات در دسترس مرتبط با ابزارهای انتهایی که خیلی محدود هستند، متمرکز شده‌اند [۱۴].

معماری انبارهٔ داده‌ای برخط شامل دیدها و رویکردهای زیادی است. قبل از ساخت RTDW[1]، نیازمندی‌ها باید به‌طور شفاف برای ساخت یک انبارهٔ داده‌ای برخط تجزیه و تحلیل شوند. چنین نیازمندی‌هایی توسط Kimball و Cacerta در سال ۲۰۰۴ ارائه شده است [۱۵].

فناوری ETL عموماً در حالت برون‌خطی کار می‌کنند [۱۶]. اما فناوری‌های برون‌خطی چندین اشکال برحسب تأخیر و قابلیت اعتماد دارند [۱۰]. یک راه عملی برای برخط کردن، کوتاه‌سازی دورهٔ بارگذاری انبارهٔ داده‌ای است. این رویکرد به‌عنوان پایگاه داده تقریباً برخط یا دسته‌ای کوچک ETL توسط Caserta و Kimball در سال ۲۰۰۴ معرفی شده است. همچنین به معرفی Hub-spoke و معماری باس برای ارتقای ابزارهای ETL سنتی پرداخته‌اند [۱۵].

Nickerson Ferreira و همکاران چگونگی تأثیرات بارکاری پرس‌وجو و تأثیراتی که به‌وسیلهٔ فرایند ETL و عوامل تحت‌تأثیری مانند نوع استراتژی بارگیری، اندازهٔ دادهٔ بارگیری، شاخص‌گذاری، محدودیت‌های یکپارچه‌سازی، فعالیت تازه‌سازی بر روی داده‌های خلاصه و بخش‌بندی جداول حقیقت را مورد تجزیه و تحلیل قرار داده‌اند. در نهایت، معیارهای آزمایشی را ارزیابی کرده و نشان داده‌اند که بخش‌بندی مهم‌ترین عامل برای قابلیت ارائه انبار داده‌ای تقریباً برخط است [۱۷].

Zhu و Liu در سال ۲۰۰۸ راه‌حل مبتنی بر SOA (معماری سرویس‌گرا) معرفی کرده‌اند. در این رویکرد در ابتدا، داده از میان وب‌سرویس استخراج و نتایج در میانگیرها ذخیره می‌شوند. چندین سطح میانگیر متفاوت در خصوص سطح‌های به‌روزرسانی، برای مثال ۵، ۱۰، ۳۰ و ۶۰ دقیقه وجود دارد. بعد از عبور از تمامی میانگیرها، داده‌ها در انبارهٔ داده‌ای ذخیره می‌شود. این راه‌حل همچنین دارای یک مؤلفه است که اطلاعات را با یکدیگر پیوند می‌دهد. این پیشنهاد، معماری و ساختار پایگاه داده را تغییر داده است [۲].

نویسندگان در [۱، ۱۲، ۱۳ و ۱۸] پژوهش‌هایی مشابه، برای پرداختن به رویکردهای مبتنی بر جداول موقتی ارائه کرده‌اند. برای نمونه Zuters در سال ۲۰۱۱ بخش‌های دانه‌دانه فاصله–زمانی را اضافه کرده‌اند. برای مثال یکی برای ساعت واپسین و یکی برای روز واپسین. محدودیت اصلی این راه حل‌ها عدم وجود تجزیهٔ مناسب سرویس‌های بارگذاری داده و پرس‌وجوهای تحلیلی است؛ زیرا آن‌ها عملیات خود را بر روی نمونهٔ پایگاه داده یکسان و در ماشین یکسان انجام می‌دهند [۱۲]. Santos و Bernardino یک متدولوژی بارگذاری انبارهٔ داده‌ای که بیش از چهار ناحیهٔ عملیاتی مختلف را پوشش می‌دهد، ارائه کرده‌اند: ۱. شمای منطبق انبارهٔ داده‌ای؛ ۲. رویهٔ بارگذاری ETL؛ ۳. پرس‌وجو منطبق OLAP؛ ۴. بسته‌بندی و دوباره بهینه‌سازی پایگاه انبارهٔ داده‌ای برخط [۱].

Tjoav و Nguyen در سال ۲۰۰۳ ساخت یک ابزار مبتنی بر دادهٔ جریانی را پیشنهاد کرده‌اند که تازگی داده‌ها و ادغام پیوسته را ممکن ساخته است. به‌علاوه معیارهای دیگر توسط نویسندگان ارزیابی شده است. هرچند این راه حل‌های مبتنی بر جریان، نتایج تقریبی را برمی‌گردانند، به‌علت داشتن نتایج دقیق با استفاده از جریان پیوسته، هزینهٔ زیادی متحمل خواهند شد [۱۹].

Zhou و همکاران در سال ۲۰۱۱ یک روش برای دریافت داده‌های تغییریافته که مبتنی بر تجزیه و تحلیل فایل تاریخچه و گوش فرادادن به فایل تاریخچه است ارائه کرده‌اند. این روش مبتنی بر فایل تاریخچهٔ تراکنش‌های DBMS، به‌منظور نظارت بر پایگاه داده است. زمانی که داده‌های منابع سیستم تغییر کند،

1. Real-Time Data Warehouse

عمل برخط ضبط می‌شود و بر روی برخی از داده‌ها پالایش انجام می‌شود [۲۰].

Jorge Santos و همکاران در سال ۲۰۱۱ راه‌حلی ساده، سریع و مؤثر مبتنی بر تکرار و جداول موقتی برای تغییر یک انبارهٔ داده‌ای تجاری سنتی به یک انبارهٔ داده‌ای برخط ارائه کرده‌اند که توانایی در بارگذاری پیوستهٔ داده و در دسترس‌پذیر بودن عملیات OLAP در زمان‌بندی ۷/۲۴ (ساعت/روز) را دارد [۹].

Ferreira و Furtado در سال ۲۰۱۳ یک مؤلفهٔ انبارهٔ داده‌ای پویا و یک انبارهٔ داده‌ای ایستا برای نمایش داده‌های یکپارچه‌شدهٔ اخیر و داده‌های تاریخی و چگونگی ادغام این مؤلفه‌ها با یکدیگر ارائه کرده‌اند [۲۱].

Abrahiem در سال ۲۰۰۷، میان‌افزار سرویس تجاری تحت معماری سرویس‌گرا ارائه کرده‌اند که قادر است ارتباطی بین لایهٔ جلویی و لایه‌های عقبی تجمع SOA که کاربران را با یک لایهٔ مدیریتی متصل کرده است برقرار کند. این لایهٔ مدیریت داده شامل یک انبارهٔ داده‌ای تقریباً برخط به‌علاوهٔ مسیریابی، امنیت، پردازش پیام و قابلیت مدل‌سازی است [۲۲].

Golab و Johnson در سال ۲۰۱۳ چهار مورد در رابطه با انبارهٔ داده‌ای جریانی را مورد بررسی قرار داده‌اند که عبارت است از: ۱. ایجاد انگیزهٔ نیاز به سامانه‌های انبارهٔ داده‌ای جریانی برای استفاده در مثال‌هایی از دنیای واقعی. این موارد در آزمایش‌هایی در نظارت از مراکز داده و شبکه انجام داده‌اند، کسب کرده‌اند؛ ۲. توضیح چندین معماری ممکن برای انبارهٔ داده‌ای جریانی؛ ۳. بحث دربارهٔ مشکلات متنوع در زبان‌های پرس‌وجو، بهینه‌سازی کارایی و کیفیت جریان داده؛ ۴. نتیجه‌گیری با بحث دربارهٔ مشکلات موجود [۲۳].

Gorawski و Gorawska در سال ۲۰۱۴، اولین پیاده‌سازی از فرایند جریان ETL را که در اصل مدل و مفهومی از انبارهٔ داده‌ای جریانی است ارائه کرده‌اند. همچنین به توضیح یک پیشرفت بر روی موتور جریان ETL پرداخته‌اند. فرایند جریانی ETL، بارگذاری داده‌های برخط را بدون هیچ‌گونه وقفه در پردازش و هدایت داده‌ها ممکن می‌سازد و فرایند تصمیم‌گیری را حمایت می‌کند [۲۴].

در [۱۸] نویسندگان پژوهشی در مورد راه‌های ممکن تبدیل یک معماری انبارهٔ داده‌ای به یک معماری باقابلیت برخط ارائه کرده‌اند. در اینجا یک بخش برخط برای بارگیری دادهٔ اخیر قبل از بارگیری آن در انبارهٔ داده‌ای ایستا معرفی شده است. همچنین به معرفی مؤلفهٔ ترکیب‌گر که باعث ترکیب نتایج پرس‌وجوهای تحلیلی که کاربران به انبارهٔ داده‌ای ایستا و بخش برخط فرستاده‌اند، بحث شده است.

شفائی و همکاران در [۲۵] معماری ارائه کرده‌اند که شامل یک رویکرد واسط XML, XSLT برای تولید محتوای مناسب و همچنین ساخت دیدهای ذخیره‌شده در سمت مشتری و سرور به‌منظور کاهش زمان پاسخ‌دهی به پرس‌وجوهاست. همچنین دو رویکرد موازی برای ترکیب کردن نتایج بخش‌های بی‌درنگ و ایستا از پایگاه دادهٔ تحلیلی تقریباً بی‌درنگ برای معماری پیشنهاد شده است. در معماری ارائه‌شده، نقش اساسی XML و فناوری‌های وابسته به آن، در تولید و نگهداری محتوا و کاهش زمان پاسخ‌دهی در یک پایگاه دادهٔ تحلیلی تقریباً بی‌درنگ مشخص شده است.

شفائی و وزیری در [۲۶] راه حلی برای استفاده از عملگرهای برش قطعه‌ای و برش ورقه‌ای پیشنهاد داده‌اند. رویکرد پیشنهادی بدین صورت است که اگر پرس‌وجویی در حال اجرا بر روی حجم زیادی از داده با زمان اجرای طولانی باشد که در این حین پرس‌وجوهایی تسلیم شوند که شامل پرش قطعه‌ای و برش ورقه‌ای پرس و جوی در حال اجرا باشند، این‌گونه پرس‌وجوها معلق شده و از نتیجهٔ پرس‌وجوی ذخیره‌شدهٔ نخست استفاده می‌کنند. بدیهی است که در این صورت از اجرای زیر پرس‌وجو بر روی حجم زیادی از داده جلوگیری به عمل آمده و در نتیجه زمان انتظار پاسخ برای کاربر کاهش پیدا خواهد کرد و از سخت‌افزار به‌صورت بهینه استفاده خواهد شد.

شفائی و دانش‌پور در [۲۷] معماری جهت اضافه کردن قابلیت پردازش موازی در انجام پرس‌وجوها در بخش‌های برخط را پیشنهاد کرده‌اند. قابلیت پردازش موازی در انجام پرس‌وجوها این امکان را می‌دهد که در یک لحظه چندین پرس‌وجو به‌طور هم‌زمان پاسخ داده شوند. این معماری در عوض استفاده از

عملگر UNION ALL پایگاه داده، از سیستم صف‌بندی استفاده کرده است. استفاده از عملگر UNION ALL موجب می‌شود که تا زمانی که کار تمامی بخش‌ها آماده نشود، خروجی بخش‌های آماده‌شده منتظر دیگر بخش‌ها بمانند. ازاین‌رو استفاده از رویکرد صف‌بندی باعث می‌شود که به‌محض آماده شدن هر یک از نتایج بخش‌ها، پردازندهٔ منطقی به پرس‌وجوی بعدی اختصاص داده شود.

شفائی و همکاران در [۲۸] یک معماری انبارهٔ داده‌ای تقریباً برخط همراه با داده‌های خارجی و هستان‌شناسی را برای تحلیل و مقایسهٔ پرس‌وجوهای مدیران، تحلیلگران و کاربران با تحلیل همبستگی میان داده‌های پایگاه داده تحلیلی با داده‌های خارجی ارائه کرده‌اند. این معماری با بهره بردن از مزیت‌های استفاده از هستان‌شناسی در ذخیره و توصیف داده‌های خارجی و نیز افزودن دانش در دادهٔ داخلی و خارجی سعی بر تسهیل کار کاربران، جلوگیری از تکرار و قابل‌فهم کردن اصطلاحات که منجر به کاهش زمان پاسخ‌دهی به پرس‌وجوهای تحلیلی می‌شود، بهره برده است.

همچنین اطلاعات مفیدی در [۳۲ـ۳۷] در زمینهٔ فاز انتقال برخط، معماری لامبادا و داده‌های بزرگ آورده شده است. در [۴۲] روشی مبتنی بر قطعه‌بندی برای محرمانگی پرس‌وجوها تشریح شده است که برای حفاظت از پرس‌جوها در پژوهش جاری می‌تواند مفید واقع شود. جدول (۱) مقایسه و خلاصه‌ای از رویکردهای انجام‌شده توسط سایر پژوهشگران را نشان داده است.

جدول (۱): خلاصه و مقایسهٔ بعضی از رویکردها

راه حل	مرجع	مزایا	معایب
CDC	[۱]، [۱۳]، [۲۰] و [۳۸]	• تغییرات جدید نیازمند تغییر در برنامهٔ کاربردی نیست. • امکان شناسایی تاریخچهٔ تغییرات. • را اندازی آسان • قابلیت اطمینان بالا	• اضافه کردن سربار در هنگام عملیات نوشتن • خواندن‌ها را نمی‌تواند ردیابی کند. • در اکثر مواقع تغییر توسط چه کسی را ردیابی نمی‌کند.
تک‌بخش برخط	[۱]، [۳]، [۴]، [۶]، [۷]، [۸]، [۹]، [۱۳]، [۱۸] و [۲۱]	• در دسترس بودن دادهٔ اخیر • کمتر کردن تداخل بین ورود و اجرای پرس‌وجو	• در دسترس نبودن دادهٔ برخط در زمان‌های اخیر متفاوت • در صورت حجم بالا از داده، منجر به افزایش زمان پاسخ‌دهی می‌شود. • در صورت حجم پایین، منجر به غیرقابل کاربردی بودن استفاده از دیدهای ذخیره‌شده می‌شود.
چندین بخش برخط	[۲]، [۱۰]، [۱۱] و [۱۲]	• در دسترس بودن دادهٔ برخط در زمان‌های اخیر متفاوت • کمتر کردن تداخل بین ورود و اجرای پرس‌وجو • استفادهٔ بهینه از دیدهای ذخیره‌شده • اجرای پرس‌وجو فقط بر روی دادهٔ اخیر	• در صورت ورود نرخ دادهٔ پایین، کاربرد چندانی ندارد. • افزایش حجم نگهداری داده در چند بخش برخط و دیدهای ذخیره‌شده
تصویر لحظه‌ای	[۳] و [۳۸]	• روشی برای برخط بودن انبارهٔ داده‌ای • قابل استفاده در انواع پایگاه داده	• روش‌های پیاده‌سازی سخت • کاهش زمان پاسخ‌دهی بدون استفاده از دیدهای ذخیره‌شده • ذخیره شدن ستون اضافه به جداول
کار موازی	[۷]، [۲۵]، [۲۶]، [۲۷] و [۴۱]	• افزایش سرعت پاسخ‌دهی • استفادهٔ بهینه از دیدهای ذخیره‌شده • حداکثر استفاده از سخت‌افزار •	• امکان ایجاد بن‌بست • تنظیم پیکربندی متفاوت برای هر محیط سخت‌افزاری و پایگاه داده نسبت به تعداد کاربران • کاهش زمان پاسخ‌دهی نسبت به کارهای موازی سبک‌وزن
دید ذخیره‌شده	[۵]، [۶]، [۲۵] و [۲۷]	• افزایش سرعت پاسخ‌دهی • ذخیره شدن نتایج محاسبه‌شده برای آرشیو	• حجم ذخیره‌سازی بالا • نیاز به الگوریتمی برای حذف دیدهای ذخیره
انبارهٔ داده‌ای ایستا و ETL	[۱۵]، [۱۶]، [۱۷]، [۳۹]، [۴۰] و [۴۱]	• دسترسی به دادهٔ تمیز شده • شاخص‌گذاری بهتر • پردازش پرس‌وجوها با روش‌های متفاوت • بهبود هوش تجاری (بهبود تصمیم‌گیری برای اخذ تصمیمات متوسط و طولانی‌مدت) • افزایش کارایی در سامانه و پرس‌وجوها • دسترسی به دادهٔ ناهمگون از منابع دادهٔ متفاوت	• افزایش مالکیت (هزینهٔ بالا و طولانی‌مدت برای ساخت) • سازگار شدن با سامانه‌های موجود • هزینهٔ بالا در نگهداری • نامناسب برای کمک به تصمیم‌گیری کوتاه‌مدت • دشواری در اضافه شدن منابع دادهٔ جدید

۳. روش پیشنهادی

۳. ۱. ایدهٔ اصلی

در انبارهٔ داده‌ای داده‌ها توسط فاز ETL از منابع داده واکشی شده و از این فاز عبور می‌کنند، سپس پس از تمیزکاری و تغییر قالب، وارد بخش نخست در معماری انبارهٔ داده‌ای چندبخشی برخط می‌شوند و بعد از سپری شدن زمانی معین در هر بخش، داده‌ها به بخش بعدی انتقال داده می‌شوند تا در نهایت به انبارهٔ داده‌ای ایستا برسند و داده‌ها به‌طور دائم در آن باقی می‌مانند. تعداد این بخش‌ها به‌صورت پویا تعیین می‌شود و همچنین زمان تأخیر هر بخش برخط نیز به‌صورت پویا و اختیاری است. زمان تأخیر هر بخش می‌تواند از دقیقه تا روز باشد؛ به‌عبارت دیگر، در هر بخش دادهٔ واردشونده به‌اندازهٔ زمان تأخیر تعیین‌شده توسط مدیر پایگاه نگهداری می‌شود. در این حین از داده‌های ذخیره‌شده در این بخش‌ها می‌توان برای اعمال پرس‌وجوهای تحلیلی برخط استفاده کرد. همان طور که گفته شد، کاربر می‌تواند پرس‌وجوی خود را تنها بر روی یک یا تعدادی معین از این بخش‌های برخط تسلیم کند. برای مثال اگر بخش برخط دوم داده‌هایی با زمان تأخیر ۲ ساعت نگهداری کند و کاربر بخواهد تنها به داده‌های اخیر (حدوداً ۲ ساعت پیش) دسترسی داشته باشد، پرس‌وجوی تحلیلی خود را بر روی این بخش برخط و بخش برخط قبلی برای اجرا تسلیم می‌کند. ساختار این بخش‌ها مشابه شمای ستاره‌ای انبارهٔ داده‌ای ایستاست. در اینجا برای نگهداری موقت داده‌هایی که نتوانستند به بخش بعدی به‌علت تداخل عملیات تحلیلی برخط و بارگیری داده منتقل شوند، از بخشی‌هایی با ساختار مشابه انبارهٔ داده‌ای ایستا استفاده شده است که در واقع نقش بافر یا میانگیر را بر عهده دارند. قالب انتقال داده به‌صورت رکوردی و یا انتقال انبوه است. اساس کار این نوع معماری‌ها در شکل (۱) نشان داده شده است [۲ و ۱۲].

در معماری انبارهٔ داده‌ای چندبخشی زمانی که عملیات پردازش تحلیلی برخط بر روی تمامی بخش‌ها (یا یکی از آن‌ها) انجام می‌شود، ممکن است پرس‌وجوهایی مشابه تسلیم انبارهٔ داده‌ای بشوند که به‌ازای هرکدام از این پرس‌وجوها بایستی تک‌تک آن‌ها بر روی این بخش(های) برخط انجام گیرد که باعث افزایش زمان پاسخ خواهد شد. و از سوی دیگر، زمانی که داده‌ها به بخش بعدی منتقل می‌شوند باعث افزایش حجم بخش بعدی می‌شود و در نتیجه زمان انجام عملیات پردازش تحلیلی برخط چندین برابر خواهد شد.

برای غلبه بر این چالش، معماری انبارهٔ داده‌ای چندبخشی با استفاده از الحاق دیدهای ذخیره‌شده بهبود داده شده است. رویکرد پیشنهادی در شکل (۲) نشان داده شده است. سؤال اصلی که به وجود خواهد آمد، این است که «پرس‌وجوهایی که در یک بخش انجام شده است و هنگامی‌که داده به بخش بعدی منتقل می‌شود، آیا راهکاری وجود دارد که بتوان از این داده و پرس‌وجوهای انجام‌شده، برای افزایش سرعت عملیات پردازش تحلیلی برخط بهره جست؟»

ایدهٔ اصلی و جواب به سؤال پرسیده‌شده این است که می‌توان عملیات تحلیلی برخطی که در بخش مربوط انجام شده است، نتایج آن‌ها را در دید ذخیره‌شده، ذخیره کرد و در هنگام انتقال داده به بخش بعدی این دیدها نیز منتقل بشوند.

هدف اصلی از رویکرد جدید عبارت است از:

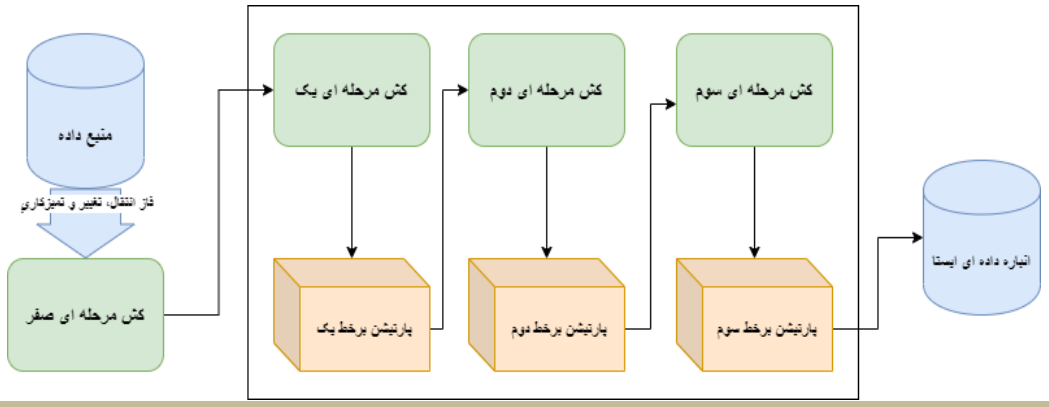
- افزایش سرعت پاسخ‌دهی به عملیات تحلیلی برخط کاربران؛
- کاهش تداخل (یا رقابت) بین عملیات تحلیلی برخط و بارگیری داده.

۲.۳. سازمان و جزئیات رویکرد

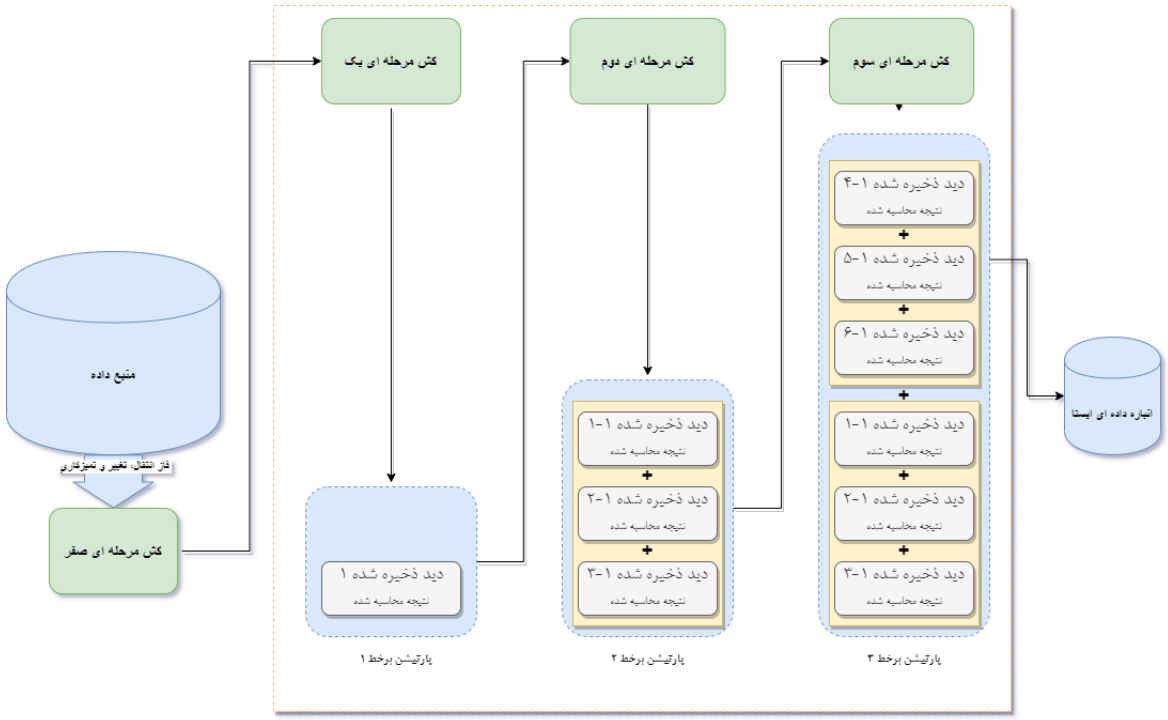
در وهلهٔ اول نیاز است تا نیازمندی‌های این رویکرد ذکر شود. این نیازمندها عبارت‌اند از:

- حجم نسبتاً بالای داده‌های ورودی از منابع داده به بخش نخست؛
- وجود حداقل دو بخش برخط؛
- تعداد نسبتاً بالای عملیات تحلیلی برخط مشابه.

به هر میزانی که حجم دادهٔ ورودی بالاتر باشد و نیز تعداد عملیات تحلیلی برخط مشابه زیاد باشد، رویکرد پیشنهادی باعث کاهش بیشتر زمان پاسخ به کاربر خواهد شد.



شکل (۱): معماری عمومی انبارهٔ داده‌ای برخط چندبخشی [۱۲]



شکل (۲): الحاق دیدهای ذخیره‌شده با **یکدیگر** و انتقال آن‌ها به بخش‌های برخط بعدی

در این پژوهش، یکی از اهداف ذکرشده، افزایش سرعت پاسخ‌دهی به عملیات تحلیلی کاربران است. در انبارهٔ داده‌ای برخط یکی از مهم‌ترین کارهای انجام‌شده توسط پژوهشگران، سعی در رسیدن به این هدف بوده است؛ زیرا در انبارههای داده‌ای برخط با ورود دادهٔ جدید و افزایش تداخل، و با توجه به ماهیت برخط بودن، نیاز به افزایش سرعت پاسخ‌دهی برای کاربران احساس می‌شود. در انبارهٔ داده‌ای غیر برخط با توجه به ورود دادهٔ جدید در ساعات غیر عملیاتی بودن سامانه و محاسبهٔ نتایج از قبل، نیاز به به‌روز بودن دیدهای ذخیره‌شده به‌صورت برخط نیست. در کل، سه خصوصیت برای انبارهٔ داده‌ای برخط نیاز است که شامل دسترس‌پذیری بالا، تأخیر کم و مقیاس‌پذیری است [۳۶]. رویکرد پیشنهادی به دو مورد اول دست یافته است. مورد سوم را می‌توان به‌عنوان کار در آینده (دادهٔ بزرگ) در نظر گرفت. همچنین در [۴۳ و ۴۴] روش‌هایی برای رتبه‌بندی و خوشه‌بندی تشریح شده است که می‌توان برای انتخاب پرس‌وجوها بر اساس معیارهای متفاوت بهره گرفت.

رویکرد پیشنهادی بدین صورت عمل می‌کند که داده‌هایی که در بخش نخست (و سایر بخش‌ها) وجود دارند، پردازش

تحلیلی برخط بر روی این داده‌ها در حال اجرا هستند. زمانی که این عملیات تمام می‌شوند، نتایج تحلیلی محاسبه‌شدهٔ آن‌ها، در جداولی به‌عنوان دید ذخیره‌شده، ذخیره می‌شوند. اگر پرس‌وجویی مشابه به بخش مربوط تسلیم شود، دیگر نیازی به محاسبهٔ پردازش تحلیلی برخط از ابتدا نیست بلکه نتیجهٔ مورد نظر، از دید ذخیره‌شده به دست آورده می‌شود. در اینجا برای به‌روزرسانی دید ذخیره‌شده با داده‌های ورودی جدید که بیشتر در بخش برخط نخست مورد توجه است، می‌توان از روش‌های مهر زمانی یا روش پیشنهادی در [۲۹] استفاده کرد.

همان طور که در شکل (۲) می‌بینید، هنگامی که زمان انتقال دادهٔ بخش برخط نخست فرامی‌رسد، شروع به انتقال انبوه داده به بخش بعدی می‌کند. در این هنگام، دیدهای ذخیره‌شده‌ای که از نتایج محاسبه‌شدهٔ پردازش تحلیلی برخط در این بخش برخط به وجود آمده‌اند، به بخش بعدی منتقل می‌شوند؛ زیرا این دیدهای ذخیره‌شده حاصل از اجرای پرس‌وجوهای تحلیلی، بر روی همین داده‌ها هستند. پس دیگر نیازی نیست پرس‌وجوهایی که از قبل با همین داده‌ها انجام شده‌اند، دوباره محاسبه شوند. با ذخیرهٔ نتایج پرس‌وجوها می‌توان از اجرای دوبارهٔ آن‌ها در بخش بعدی جلوگیری به عمل آورد. این دیدها از قبل یا به‌روزرسانی شده‌اند یا نیاز به به‌روزرسانی دارند؛ در هر صورت آخرین به‌روزرسانی بر روی دیدها انجام می‌گیرد و سپس منتقل می‌شوند. منظور از انتقال دید ذخیره‌شده، به‌صورت فیزیکی نیست بلکه یک انتقال منطقی صورت می‌گیرد به‌نحوی که اشاره‌گر آن‌ها به بخش برخط بعدی جهت استفاده، اشاره می‌کند. در این صورت، زمانی برای انتقال دید ذخیره‌شده وجود ندارد. این کار از طریق ساخت یک فراداده انجام می‌شود که اطلاعات مربوط به دیدهای ذخیره‌شده را خود نگهداری می‌کند. این دیدهای ذخیره‌شده بعد از انتقال از بخش مربوطه به بخش بعدی، اعتباری در این بخش به‌دلیل ورود داده‌های جدید و تغییر نتایج محاسبه‌شده نخواهند داشت. از این پس برای به‌دست‌آوردن نتایج پردازش تحلیلی برخط در بخش برخط بعدی، نیاز به محاسبه نیست بلکه از دید ذخیره‌شده استفاده خواهد شد.

بعد از چندین انتقال داده به بخش بعدی (بخش‌های برخط دوم و به بعد) داده‌ها و به همین ترتیب دیدهای ذخیره‌شده نیز روی هم انباشته می‌شوند. اگر از این دیدهای ذخیره‌شده استفاده نشود، با بارگیری چندبارهٔ هر بخش برخط، حجم داده‌ها چندین برابر می‌شود و در نتیجه زمان محاسبهٔ پرس‌وجوهای تحلیلی برخط افزایش پیدا می‌کند. برای به‌دست‌آوردن نتیجهٔ پردازش تحلیلی برخط، از الحاق دیدهای ذخیره‌شده معادل با این پرس‌وجو با یکدیگر، استفاده شده و نتیجه به کاربر برگردانده می‌شود. در اینجا می‌توان دو سیاست را دنبال کرد: در سیاست نخست، همان طور که گفته شد دیدهای ذخیره‌شدهٔ معادل با این پرس‌وجو با یکدیگر الحاق می‌شوند تا نتیجهٔ خروجی حاصل شود. در سیاست دوم، قبل از اینکه اشاره‌گر دیدهای ذخیره‌شده تغییر کند، ابتدا برای هر پرس‌وجوی تحلیلی که حاوی چندین دید ذخیره‌شده است، دیدهای ذخیره‌شدهٔ هر پرس‌وجو را با یکدیگر الحاق می‌کنیم؛ به‌عبارت دیگر هر پرس‌وجوی تحلیلی که حاوی چندین دید ذخیره‌شده است، با الحاق آن‌ها به یکدیگر، تبدیل به یک دید ذخیره‌شده می‌شوند. در نتیجه نیاز به محاسبهٔ پردازش تحلیلی برخط بر روی کل بخش برخط که با چندین انتقال داده به‌صورت پشت سر هم پرشده است، نیست. در این صورت زمان انتظار کاربر به پاسخ عملیات پردازش تحلیلی خود به‌شدت کاهش پیدا می‌کند. برای مثال اگر در بخش برخط نخست، ۲ میلیون رکورد وجود داشته باشد پس از سه انتقال داده به بخش برخط بعدی تعداد رکورد در این بخش برخط ۶ میلیون می‌شود. و تعداد دیدهای ذخیره‌شده به ازای هر پرس‌وجو در اینجا سه دید است، که با الحاق آن‌ها با یکدیگر پاسخ به‌سرعت تهیه می‌شود. شبه کد مربوط به این روال در شکل (۳) آمده است.

در خط اول شبه کد، ابتدا بررسی می‌کند که آیا این پردازش تحلیلی برخط قبلاً اجرا شده و نتیجهٔ آن ذخیره‌شده است یا خیر. اگر اجرا نشده باشد آن را اجرا می‌کند که در این صورت الگوریتم دوم که فرایند مربوط به ساخت دید ذخیره‌شده است، اجرا می‌شود (خط شمارهٔ ۲) در غیر این صورت در یک حلقه (خط شمارهٔ ۴) از جدولی که مشخصات دیدهای ذخیره‌شده در

آن نگهداری می‌شود (Result Computed Table)، دیدهای ذخیره‌شده معادل با پرس‌وجوی و شمارهٔ بخش برخط ورودی را واکشی کرده و در متغیر MV قرار می‌دهد (خط شمارهٔ ۵). سپس از طریق متد join نتایج محاسبه‌شده را با یکدیگر الحاق کرده و در متغیر ارجاعی join Result قرار می‌دهد (خط شمارهٔ ۶) و آن را برگشت می‌دهد (خط شمارهٔ ۸).

```
The Join Materialized View Algorithm.
Input: Query Q; MaterializedView joinResult, MV;
Number PNO:
Output: Materialized View
Begin
 1. if (is Q exist == false) then
 2.    Run(Q); % or Run The CreateMV
ALGORITHM(Q)
 3. else
 4.    for each MV in ResultComputedTable do
 5.          MV ← Find equivalent of MV(Q, PNO)
 6.          joinResult ← Join(MV, joinResult);
 7.    end for
 8.    return joinResult;
 9.end if
 End
```

شکل (۳): الگوریتم الحاق دیدهای ذخیره‌شده برای تولید نتیجهٔ خروجی

برای فهم بهتر می‌توان خط شمارهٔ ۶ را به این صورت توضیح داد که نتایج محاسبه‌شدهٔ پرس‌وجوی قبلی (که در جداول موقت) ذخیره‌شده را با استفاده از عملگر Union به یکدیگر الحاق کرد و به نتیجهٔ کامل پرس‌وجو رسید؛ بدون اینکه نیاز به اجرای پرس‌وجو در بخش برخط باشد (منظور از MV1-3 این است که شمارهٔ پرس‌وجو، ۱ و شمارهٔ انتقال، ۳ است؛ به‌عبارت دیگر پرس‌وجو شمارهٔ ۱ سه بار در بخش برخط یک، اجرا شده است و در هر بار انتقال داده به بخش بعدی، نتیجهٔ ذخیره‌شدهٔ این پرس‌وجو نیز انتقال پیدا کرده است).

```
SELECT * FROM MV1-1 --The result of the first transfer
UNION ALL
SELCT * FROM MV1-2 --The result of the second transfer
UNION ALL
SELCT * FROM MV1-3 --The result of the third transfer
.....
```

بعد از اینکه تمامی داده‌ها از بخش‌های برخط عبور کردند و به انبارهٔ داده‌ای ایستا رسیدند، این داده‌ها به‌طور مانا در انبارهٔ داده‌ای ایستا ذخیره می‌شوند. در اینجا می‌توان دیدهای ذخیره‌شده را که تا این مرحله ایجاد شده‌اند حذف کرد. همچنین می‌توان این دیدهای ذخیره‌شده را به انبارهٔ داده‌ای ایستا منتقل کرد. در این صورت بهتر است الگوریتمی برای نگهداری این دیدهای ذخیره‌شده ایجاد شود یا از الگوریتم‌های موجود در این زمینه استفاده گردد.

یکی از چالش‌هایی که ممکن است ظاهر شود، تعداد بالای دیدهای ذخیره‌شده‌ای است که ساخته می‌شوند. برای غلبه بر این چالش، دیدهای ذخیره‌شدهٔ مورد علاقه تعریف می‌شوند. در این رویکرد، مدیر پایگاه به صلاحدید خود یک مجموعه از پرس‌وجوهای پردازش تحلیلی برخط مورد علاقهٔ کاربران را علامت‌گذاری می‌کند و تنها در این صورت آن‌ها تبدیل به دید ذخیره‌شده می‌شوند. در رویکرد دوم، زمان اجرای پرس‌وجوهای پردازش تحلیلی برخط محاسبه می‌شود و سپس اگر این زمان از زمان آستانه که توسط مدیر پایگاه تعیین شده است بزرگ‌تر بود، این پرس‌وجو پردازش تحلیلی برخط تبدیل به دید ذخیره‌شده می‌شود. می‌توان این دو رویکرد را با یکدیگر تلفیق کرد؛ که در شکل (۴): الگوریتم ساخت دید ذخیره‌شده) شبه کد مربوطه آورده شده است. اگر از این رویکرد استفاده شود، بایستی در هر مرحله از انتقال (منطقی) دیدهای ذخیره‌شده، دید ذخیره‌شدهٔ مرحلهٔ قبل معادل با این پرس‌وجو جهت وجود یا عدم وجود در بخش برخط بعدی بررسی شود. این کار از طریق اطلاعات موجود در فرادادهٔ دیدهای ذخیره‌شده انجام می‌شود. در صورت عدم وجود دید ذخیره‌شده در مرحلهٔ قبل انتقال، بایستی تمامی دیدهای ذخیره‌شده در مراحل قبل حذف شوند. این کار بدین منظور انجام می‌شود که اگر در یک مرحله از انتقال، دید ذخیره‌شده‌ای ساخته نشده باشد، نمی‌توان جواب کامل بخش برخط را با الحاق دیدهای ذخیره‌شده به دست آورد؛ زیرا تعداد دیدهای ذخیره‌شده برای الحاق کافی نیست. به‌عبارت دیگر بخشی از جواب پرس‌وجو موجود نیست.

شکل (۴)، نخست وجود پرس‌وجو پردازش تحلیلی برخط بررسی می‌شود و در صورت عدم وجود، الگوریتم ادامه پیدا می‌کند (خط شمارهٔ ۱). سپس یک تایمر برای محاسبهٔ زمان اجرای پرس‌وجو پردازش تحلیلی برخط شروع می‌شود (خط

شمارهٔ ۲). آنگاه پرس‌وجو پردازش تحلیلی برخط شروع به اجرا می‌کند و نتیجهٔ آن در tmpResult به‌طور موقت ذخیره می‌شود (خط شمارهٔ ۳). بعد تایمر متوقف می‌شود (خط شمارهٔ ۴) و زمان اجرای پرس‌وجو پردازش تحلیلی برخط با آستانه و اینکه جزء پرس‌وجوهای مورد علاقه است، مقایسه می‌شود (خط شمارهٔ ۵). در صورتی که شرط برقرار باشد، نتیجهٔ محاسبه‌شده به‌صورت یک دید ذخیره‌شده به همراه شمارهٔ بخش برخط در فراداده ذخیره می‌گردد (خط شمارهٔ ۶ و ۷). هدف از مقایسهٔ زمان اجرا با زمان آستانه، این است که آیا با توجه به زمان اجرای این پرس‌وجو نیاز به ذخیرهٔ نتیجه وجود دارد یا خیر. برای مثال، اگر زمان آستانه را ۱۰ دقیقه تنظیم کرده باشند و زمان اجرای پرس‌وجو ۴ دقیقه باشد، آنگاه با توجه به عدم طولانی بودن اجرای پرس‌وجو، نیاز به ساخت دید محاسبه‌شده ندارد. اما اگر در مراحل بعدی اجرای همین پرس‌وجو، زمان اجرا بیش از ۱۰ دقیقه به طول بینجامد، نتیجهٔ این پرس‌وجو برای اجرای بعدی ذخیره می‌شود.

زمانی که از واژهٔ دید ذخیره‌شده استفاده می‌شود این است که «نتیجهٔ یک پرس‌وجو» بایستی ذخیره گردد. برای نگهداری نتیجهٔ یک پرس‌وجو می‌توان از یک جدول فیزیکی استفاده کرد یا از هر روش دیگری که سرعت بیشتری داشته باشد.

```
The CreateMV Algorithm.
Input: Query Q; MaterializedView tmpResult;
Number PNO; Time threshold:
Output: Materialized View
Begin
 1. If( is Q exist == false) then
 2.    The timer starts to work
 3.    tmpResult ← Compute Q;
 4.    time ← The timer stops and get interval
 5.    if ( time >= threshold and is Q Interested) then
 6.       Create Materialized View from tempResult
 7.       Save PNO
 8.    end if
 9. end if
End
```

شکل (۴): الگوریتم ساخت دید ذخیره‌شده

این پیشنهاد باعث کاهش تداخل بین عملیات تحلیلی برخط و بارگیری داده می‌شود. زمانی که عملیات تحلیلی برخط بر روی بخش‌ها در حال اجرا هستند، بارگیری دادهٔ جدید با این روند تداخل پیدا می‌کند. اما در روش پیشنهادی به‌دلیل اینکه برای پاسخ‌گویی به پرس‌وجوهای پردازش تحلیلی برخط از دیدهای ذخیره‌شده استفاده می‌شود و دیگر لزومی به مراجعه به بخش‌های برخط (به‌خصوص بخش‌های دوم و به‌بعد) نیست، نتایج از الحاق دیدهای ذخیره‌شده به دست آورده می‌شود. در این صورت تداخل بین عملیات بارگیری داده و عملیات تحلیلی برخط به‌طور فراوانی کاهش پیدا می‌کند.

خلاصه‌ای از خصوصیات به‌دست‌آمده از رویکرد پیشنهادی جدید:

- کاهش زمان پاسخ عملیات تحلیلی برخط به کاربران؛
- کاهش تداخل بین عملیات تحلیلی برخط و عملیات بارگیری داده؛
- کنترل تعداد دیدهای ذخیره‌شده؛
- بهبود معماری‌های انبارهٔ داده‌ای برخط پیشین؛
- ایجاد الگوریتمی برای استفاده از دیدهای ذخیره‌شده در مرحلهٔ آخر (کار در آینده).

۴. نتایج آزمایشگاهی

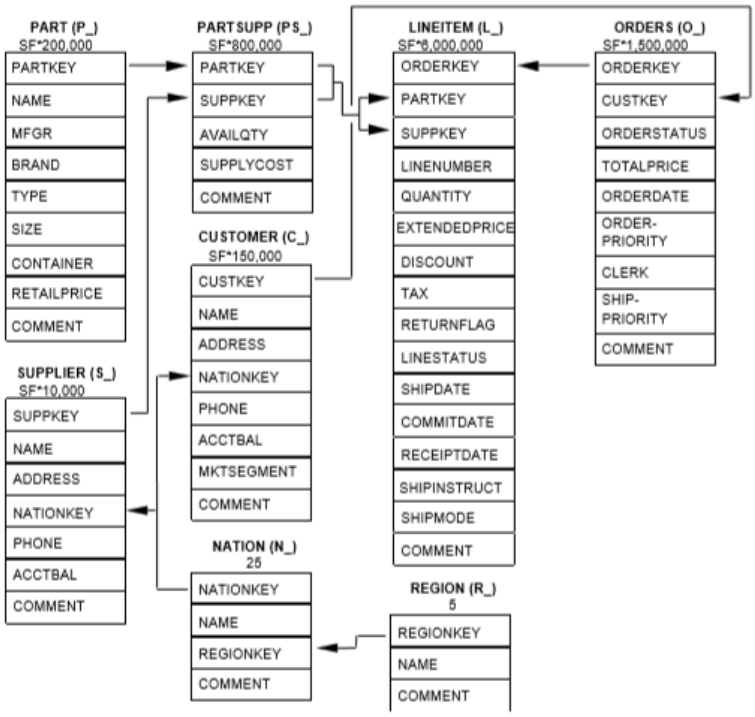
این پژوهش توسط ارزیاب SSB [۳۰] که نمونهٔ تطبیق داده شده از ارزیاب TPC-H [۳۱] است، مورد ارزیابی قرار گرفته است. این ارزیاب یک انبارهٔ داده‌ای با شمای ستاره‌ای است که شامل چهار بعد (جدول) به‌نام‌های مشتری، تهیه‌کننده، قسمت، تاریخ و یک بعد حقیقت به نام اقلام سفارش است. حجم پایگاه داده در جدول (۲) نشان داده شده است.

نام جدول	supplier	part	lineItem	dates	customer
تعداد رکورد	۶۰۰۰۰	۱۰۰۰۰۰۰	۲۰۰۰۰۰۰۰	۲۵۵۶	۹۰۰۰۰۰
حجم ذخیره‌سازی (مگابایت)	۵/۹۳۸	۱۱۳/۲۲۷	۱۸۶۰/۱۲۵	۰/۲۵۸	۹۳/۴۵۳

جدول (۲:) مشخصات ساختار پایگاه

```
--ROLAP Q4.3 (13)

select dates.d_year,
part.p_mfgr,
part.p_category,
part.p_brand1,
supplier.s_region,
supplier.s_nation,
supplier.s_city ,
sum(lo_revenue - lo_supplycost)

from dbo.dates as dates, dbo.lineorder as lineorder, dbo.customer as customer, dbo.part as
part, dbo.supplier as supplier

where lineorder.lo_orderdate = dates.d_datekey and
dates.d_year in (1997, 1998) and
lineorder.lo_custkey = customer.C_CUSTKEY and
customer.c_region = 'AMERICA' and
lineorder.lo_partkey = part.P_PARTKEY and
part.p_mfgr = 'MFGR#1' and
part.p_category = 'MFGR#14' and
part.p_brand1 in ('MFGR#141', 'MFGR#1410', 'MFGR#1411', 'MFGR#1412', 'MFGR#1413', 'MFGR#1414',
'MFGR#1415', 'MFGR#1416', 'MFGR#1417', 'MFGR#1418', 'MFGR#1419', 'MFGR#142', 'MFGR#1420',
'MFGR#1421', 'MFGR#1422', 'MFGR#1423', 'MFGR#1424', 'MFGR#1425', 'MFGR#1426', 'MFGR#1427',
'MFGR#1428', 'MFGR#1429', 'MFGR#143', 'MFGR#1430', 'MFGR#1431', 'MFGR#1432', 'MFGR#1433',
'MFGR#1434', 'MFGR#1435', 'MFGR#1436', 'MFGR#1437', 'MFGR#1438', 'MFGR#1439', 'MFGR#144',
'MFGR#1440', 'MFGR#145', 'MFGR#146', 'MFGR#147', 'MFGR#148', 'MFGR#149') and
lineorder.lo_suppkey = supplier.S_SUPPKEY and
supplier.s_region = 'AMERICA' and
supplier.s_nation = 'UNITED STATES' and
supplier.s_city in ('UNITED ST0', 'UNITED ST1', 'UNITED ST2', 'UNITED ST3', 'UNITED ST4',
'UNITED ST5', 'UNITED ST6', 'UNITED ST7', 'UNITED ST8', 'UNITED ST9')

group by dates.d_year, part.p_mfgr, part.p_category, part.p_brand1, supplier.s_region,
supplier.s_nation, supplier.s_city;
```

شکل (۵): نمونه‌ای از پرس‌وجو

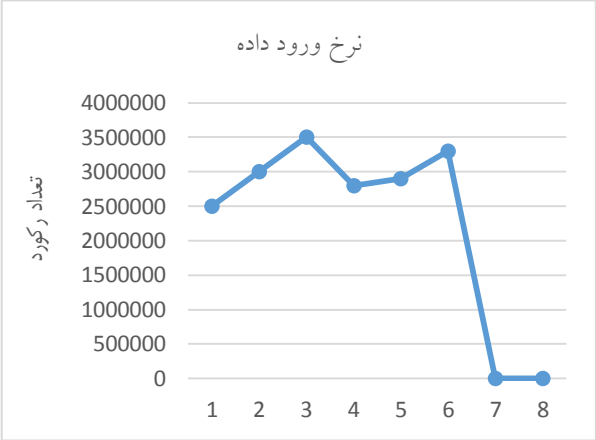


شکل (۶): شمای جداول ارزیاب

نوع پرس‌وجوها به‌صورت تصادفی و با تغییر در شرایط و گروه‌بندی در یک فایل XML به تعداد مورد نظر ذخیره شده و از آن‌ها برای ارزیابی استفاده شده است. نمونه‌ای از پرس‌وجو در شکل (۵) آورده شده است. همچنین شمای ارزیاب در شکل (۶) نشان داده شده است. ارزیابی انجام‌شده بر روی سامانه‌ای با یک پردازنده Intel(R) Core(TM) i7 2.20GHz و حافظهٔ اصلی با ظرفیت ۸ گیگابایت آزمایش شده است. همچنین میزبانی پایگاه داده بر روی SQL Server 2014 و از زبان برنامه‌نویسی C# استفاده شده است.

نرخ ورود داده به بخش برخط اول و سپس انتقال آن به بخش دوم در نمودار شکل (۷) نشان داده شده است. این عمل در هشت مرحله انجام شده است که در هر مرحله رکوردها وارد بخش برخط شده‌اند. ورود داده از منبع داده که در اینجا یک پایگاه داده با ساختار ارزیاب است استفاده شده است. دلیل استفاده از ساختار منبع داده به شکل انبارهٔ داده این است که فرض شده است داده از فاز ETL (برای تمیزکاری داده و تغییر قالب داده به قالب انبارهٔ داده‌ای) عبور کرده و به بخش برخط رسیده است. در این پژوهش اجرای پرس‌وجوی تحلیلی مهم است و فاز ETL برای تغییر قالب منبع به مقصد و... مهم نیست. ورود داده در هر بخش با توجه به زمان ورود انجام می‌شود. برای مثال هر ۱۰ دقیقه یک بار، داده از منبع داده خوانده می‌شود و به بخش برخط منتقل می‌شود و هر بخش برخط قبل از ورود داده به خود، دادهٔ موجود را به بخش بعدی ارسال می‌کند تا در نهایت به انبارهٔ داده‌ای ایستا برسد. در این آزمایش از سه بخش برخط و انبارهٔ داده‌ای ایستا استفاده شده است. همان طور که در نمودار شکل (۷) مشاهده می‌شود، نرخ ورود داده تا مرحلهٔ ۶ انجام شده است ولی در مراحل ۷ و ۸ نرخ ورود داده صفر است. برای اینکه در ارزیابی این پژوهش نمی‌توان داده‌ها را به‌صورت همیشگی و پیوسته وارد کرد، در مراحل ۷ و ۸ ورود داده متوقف می‌شود. با این حال داده‌های واردشده از قبل، هنوز در بخش‌های برخط وجود دارند و به انبارهٔ داده‌ای ایستا منتقل نشده‌اند، به همین دلیل تا زمان انتقال، این ارزیابی شامل مراحل ۷ و ۸ نیز می‌شود.

به‌ازای ورود هر مرحله از داده، پرس‌وجوهای پردازش تحلیلی برخط بر روی بخش(ها) برخط در حال اجرا هستند. تعداد این پرس‌وجوهای پردازش تحلیلی برخط از ۸ تا ۶۰ عدد به‌ازای هر مرحله در حال اجرا هستند. این پرس‌وجوها به‌طور تصادفی در ابتدای کار ایجاد شده و تسلیم انبارهٔ داده‌ای می‌شوند. در نمودار شکل (۸) نمودار مربوط نشان داده شده است:

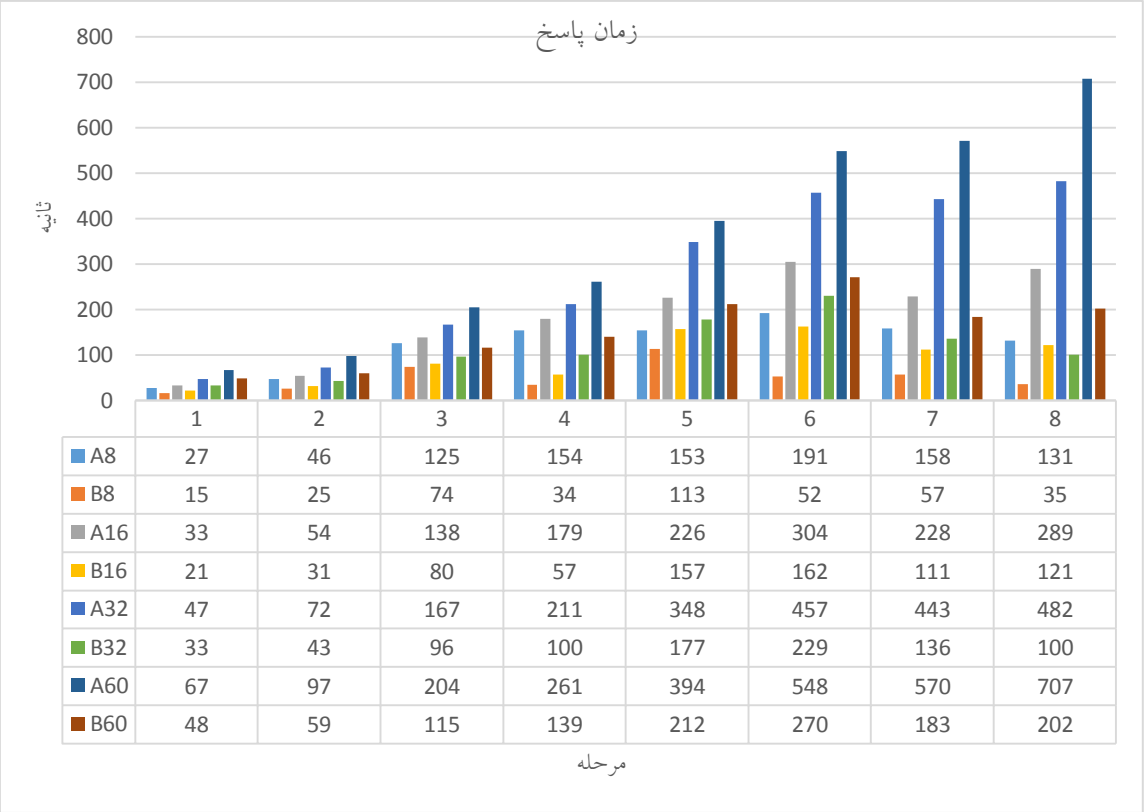


شکل (۷): نرخ ورد داده

همان طور که از نمودار مشخص است، در هر مرحله از بارگیری داده (۸ مرحله) زمان اجرای پرس‌وجوهای پردازش تحلیلی برخط به تعداد ۸، ۱۶، ۳۲ و ۶۰ عدد که تسلیم انبارهٔ داده‌ای شده‌اند، با روش معمولی (جهت اختصار A) و روش پیشنهادی (جهت اختصار B) نشان داده شده است. منظور از روش معمولی، استفاده از بخش‌های برخط بدون استفاده از دیدهای ذخیره‌شده و تجمیع آن‌هاست که در شکل (۱) نشان داده شده است. برای مثال، زمان اجرای ۱۶ پرس‌وجو پردازش تحلیلی برخط در پنجمین زمان انتقال در رویکرد معمولی (A16) برابر است با ۲۲۶ ثانیه و در رویکرد پیشنهادی (B16) برابر است با ۱۵۷ ثانیه است. همان طور که نمودار نشان می‌دهد، زمان اجرای پرس‌وجو پردازش تحلیلی برخط در روش پیشنهادی کاهش پیدا کرده است. تعداد اجرای پرس‌وجو پردازش تحلیلی برخط در کنار اختصار نوشته شده است.

برای درک ساده‌تر و نتیجه‌گیری کلی، متوسط زمان اجرای ۸ مرحله از پرس‌وجو پردازش تحلیلی برخط به تعداد ۸، ۱۶، ۳۲ و ۶۰ ، در نمودار شکل (۹) نشان داده شده است.

برای مثال، همان طور که از نمودار شکل (۷) مشاهده می‌شود، متوسط زمان اجرا در رویکرد معمولی (A) با تعداد ۳۲ پرس‌وجو تحلیلی برخط برابر است با ۲۷۸ ثانیه و در رویکرد پیشنهادی (B) برابر با ۱۱۶ ثانیه است. در کل، آنچه از نمودار می‌توان به دست آورد، زمان پاسخ پرس‌وجوهای تحلیلی برخط کمتر از نصف کاهش پیدا کرده است.



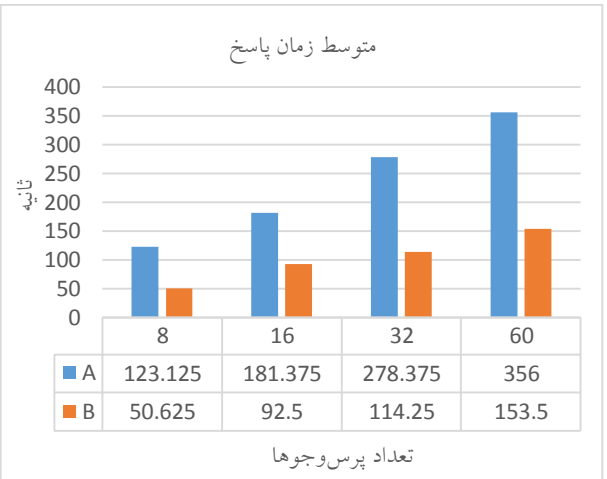
	1	2	3	4	5	6	7	8
A8	27	46	125	154	153	191	158	131
B8	15	25	74	34	113	52	57	35
A16	33	54	138	179	226	304	228	289
B16	21	31	80	57	157	162	111	121
A32	47	72	167	211	348	457	443	482
B32	33	43	96	100	177	229	136	100
A60	67	97	204	261	394	548	570	707
B60	48	59	115	139	212	270	183	202

مرحله

شکل (۸): زمان پاسخ به پرس‌وجوهای تحلیلی برخط در هشت مرحله به تعداد ۸، ۱۶، ۳۲ و ۶۰ پرس‌وجو (A روش معمولی، B روش پیشنهادی)

۵. نتیجه‌گیری

در این پژوهش، معماری انبارهٔ داده‌ای برخط، مورد مطالعه و تجزیه و تحلیل قرار گرفته و منجر به ارائهٔ یک پیشنهاد مبنی بر کاهش زمان پاسخ عملیات تحلیلی برخط کاربران و کاهش تداخل عملیات تحلیلی برخط با عملیات بارگیری شده است. در این رویکرد با به‌کارگیری ایجاد دیدهای ذخیره‌شده از نتایج محاسبه‌شدهٔ پرس‌وجوهای تحلیلی اجراشده بر روی بخش(ها) و الحاق این دیدها با یکدیگر و برگشت نتیجهٔ پرس‌وجو تحلیلی، توانسته است این نوع معماری‌های برخط مورداستفاده در انبارهٔ داده‌ای را بهبود ببخشد. همچنین در این پژوهش، رویکردی برای کنترل تعداد دیدهای ذخیره‌شده پیشنهاد شده است که تابع قید زمان اجرای پرس‌وجو تحلیلی و پرس‌وجوهای تحلیلی مورد علاقهٔ انتخاب‌شده توسط مدیر پایگاه هستند.



	8	16	32	60
A	123.125	181.375	278.375	356
B	50.625	92.5	114.25	153.5

تعداد پرس‌وجوها

شکل (۹): متوسط زمان پاسخ پرس‌وجوهای تحلیلی برخط ۸ مرحله با تعداد ۸، ۱۶، ۳۲ و ۶۰ پرس‌وجو

به‌طور خلاصه، در روش پیشنهادی برای کسب کارایی بهتر، بایستی نرخ ورود دادهٔ بالا و تعداد پرس‌وجوهای تحلیلی برخط مشابه به تناسب بالا باشد. در این صورت کارایی روش پیشنهادی به‌سمت کارایی بهتر افزایش پیدا می‌کند.

مراجع

[1] Santos R. J. and Bernardino J., *"Real-time data warehouse loading methodology"*, Proceedings of the 2008 international symposium on Database engineering & applications, ACM, pp. 49-58, 2008.

[2] Zhu Y., An L. and Liu S., *"Data updating and query in real-time data warehouse system"*, Computer science and software engineering, 2008 international conference on, IEEE, pp. 1295-1297, 2008.

[3] Qu W., Basavaraj V., Shankar S. and Dessloch S., *"Real-Time Snapshot Maintenance with Incremental ETL Pipelines in Data Warehouses"*, International Conference on Big Data Analytics and Knowledge Discovery, Springer, pp. 217-228, 2015.

[4] Mao Y., Min W., Wang J., Jia B. and Jie Q., *"Dynamic mirror based real-time query contention solution for support big real-time data analysis"*, Information Technology and Electronic Commerce (ICITEC), 2014 2nd International Conference on, IEEE, pp. 229-233, 2014.

[5] Hamdi I., Bouazizi E. and Feki J., *"Dynamic management of materialized views in real-time data warehouses"*, Soft Computing and Pattern Recognition (SoCPaR), 2014 6th International Conference of, IEEE, pp. 168-173, 2014.

[6] Lin Z., Lai Y., Lin C., Xie Y. and Zou Q., *"Maintaining internal consistency of report for real-time OLAP with layer-based view"*, Asia-Pacific Web Conference, Springer, pp. 143-154, 2011.

[7] He W. and Cui L., *"A Parallel Approach for Real-Time OLAP Based on Node Performance Awareness"*, High Performance Computing, Springer, pp. 75-88, 2013.

[8] Obali M., Dursun B., Erdem Z. and Görur A. K., *"A real time data warehouse approach for data processing"*, Signal Processing and Communications Applications Conference (SIU), 2013 21st, IEEE, pp. 1-4, 2013.

[9] Santos R. J., Bernardino J. and Vieira M., *"24/7 Real-Time Data Warehousing: A Tool for Continuous Actionable Knowledge"*, 2011 IEEE 35th Annual Computer Software and Applications Conference, IEEE, pp. 279-288, 2011.

[10] Naeem M. A., Dobbie G. and Webber G., *"An event-based near real-time data integration architecture"*, 2008 12th Enterprise Distributed Object Computing Conference Workshops, IEEE, pp. 401-404, 2008.

[11] Yichuan S. and Yao X., *"Research of Real-time Data Warehouse Storage Strategy Based on Multi-level Caches"*, Physics Procedia, pp. 2315-2321, 2012.

[12] Zuters J., *"Near real-time data warehousing with multi-stage trickle and flip"*, International Conference on Business Informatics Research, Springer, pp. 73-82, 2011.

[13] Jain T., *"Refreshing datawarehouse in near real-time"*, International Journal of Computer Applications, vol. 46, 2012.

[14] Jia R., Xu S. and Peng C., *"Research on Real Time Data Warehouse Architecture"*, International Conference on Information Computing and Applications, Springer, pp. 333-342, 2013.

[15] Kimball R. and Caserta J., *"The Data WarehouseETL Toolkit: Practical Techniques for Extracting, Cleaning, Conforming, and Delivering Data"*, John Wiley & Sons, 2011.

[16] Chappell D. A., *"Enterprise Service Bus"*, O'Reilly Media, 2004.

[17] Ferreira N., Martins P. and Furtado P., *"Near real-time with traditional data warehouse architectures: factors and how-to"*, Proceedings of the 17th International Database Engineering & Applications Symposium, ACM, pp. 68-75, 2013.

[18] Cuzzocrea A., Ferreira N. and Furtado P., *"Enhancing Traditional Data Warehousing Architectures with Real-Time Capabilities"*, International Symposium on Methodologies for Intelligent Systems, Springer, pp. 456-465, 2014.

[19] Tho M. N. and Tjoa A. M., *"Zero-latency data warehousing for heterogeneous data sources and continuous data streams"*, 5th International Conference on Information Integrationand Web-based Applications Services, pp. 55-64, 2003.

[20] Zhou H., Yang D. and Xu Y., *"An ETL strategy for real-time data warehouse"*, Practical applications of intelligent systems, Springer, pp. 329-336, 2011.

[21] Ferreira N. and Furtado P., *"Real-time data warehouse: a solution and evaluation"*, International Journal of Business Intelligence and Data Mining, vol. 8, pp. 244-263, 2013.

[22] Abrahiem R., *"A new generation of middleware solutions for a near-real-time data warehousing architecture"*, 2007 IEEE International Conference on Electro/Information Technology, IEEE, pp. 192-197, 2007.

[23] Golab L. and Johnson T., *"Data stream warehousing"*, 2014 IEEE 30th International Conference on Data Engineering, IEEE, pp. 1290-1293, 2014.

[24] Gorawski M. and Gorawska A., *"Research on the stream ETL process"*, International Conference: Beyond Databases, Architectures and Structures, Springer, pp. 61-71, 2014.

[۲۵] شفائی، سید مصطفی، دانش‌پور، نگین، شفائی، سید مجید، *«استفاده از قابلیت‌های XML و دیدهای ذخیره‌شده در ایجاد یک معماری پایگاه داده تحلیلی تقریباً بی‌درنگ»*، نشریه مهندسی برق و مهندسی کامپیوتر ایران، سال پانزدهم، شماره ۱-ب، صفحات ۱۴-۲۶، ۱۳۹۶.

[۲۶] شفائی، سید مجید، وزیری، بابک، *«بهبود زمان پاسخ‌دهی پرس‌وجوهای موازی در معماری انبارهٔ داده‌های سنتی و برخط»*، اولین کنفرانس بین‌المللی هوش تجاری ایران، تهران، ۱۳۹۵.

[۲۷] شفائی، سید مصطفی، دانش‌پور، نگین *«بهبود زمان پاسخ‌دهی در پایگاه داده تحلیلی بی‌درنگ با استفاده از قابلیت پردازش موازی»*، نهمین کنفرانس داده‌کاوی ایران، ۱۳۹۴.

[۲۸] شفائی، سید مصطفی، دانش پور، نگین و شفائی، سید مجید، *«معماری پایگاه داده تحلیلی تقریباً بی‌درنگ مبتنی بر*

برون‌سپاری‌شده»، مجله محاسبات نرم، جلد هشتم، شماره ۱، ص ۲-۱۳، ۱۳۹۸.

[۴۳] نبی‌لو، مریم، دانشپور، نگین، «*ارائهٔ یک الگوریتم خوشه‌بندی برای داده‌های دسته‌ای با ترکیب معیارها*»، مجله محاسبات نرم، جلد پنجم، شماره ۱، ص ۱۴ـ۲۵، ۱۳۹۵.

[۴۴] سالخورده حقیقی، مهدی، ساجدی، فاطمه، «*ارزش‌گذاری ارجاعات غیرمستقیم در شبکه‌های استنادی با استفاده از تلفیق داده‌ها*»، مجله محاسبات نرم، جلد ۷، شماره ۲، ص ۳۶-۴۶، ۱۳۹۷.

هستان‌شناسی»، نشریه مهندسی برق و مهندسی کامپیوتر ایران، سال پانزدهم، شمارهٔ ۲-ب، صفحات ۸۵ـ۱۰۱، ۱۳۹۶.

[29] Santos R. J. and Bernardino J., *"A Query Cache Tool for Optimizing Repeatable and Parallel OLAP Queries"*, International Conference on Database and Expert Systems Applications, Springer, pp. 143-152, 2009.

[30] O'neil P., O'neil E., Chen X. and Revilak S., *"The star schema benchmark and augmented fact table indexing"*, Technology Conference on Performance Evaluation and Benchmarking, Springer, pp. 237-252, 2009.

[31] COUNCIL T. P. P., *"TPC-H benchmark specification"*, Published at http://www.tcp.org/hspec.html, 2008.

[32] Ali A. R., "*Real-time big data warehousing and analysis framework*," 2018 IEEE 3rd International Conference on Big Data Analysis (ICBDA), Shanghai, pp. 43-49, 2018.

[33] Cuzzocrea A. and Moussa R., "*Towards Lambda-Based Near Real-Time OLAP over Big Data*," 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), Tokyo, pp. 438-441, 2018.

[34] Wibowo A. and Akbar S., "*Handling of internal inconsistency OLAP - Based lock table using Message Oriented Middleware in near real time data warehousing*," 2015 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, pp. 329-334, 2015,

[35] Fikri N., Rida M., Abghour N. and et al. "*An adaptive and real-time based architecture for financial data integration*", J Big Data 6, 97, 2019.

[36] Machado G.V., Cunha Í., Pereira A.C.M. and et al. *"DOD-ETL: distributed on-demand ETL for near real-time business intelligence"*. J Internet Serv A, pp. 10, 21, 2019.

[37] Muddasir N. M., Raghuveer K. "*Study of methods to achieve near real time ETL*", In: 2017 international conference on current trends in computer, electrical, electronics and communication (CTCEEC), Mysore, India, pp. 436–41, 2017.

[38] Shi J., Bao Y., Leng F., Yu G., *"Study on log-based change data capture and handling mechanism in real-time data warehouse."* In: Computer Science and Software Engineering, 2008 International Conference On. New York: IEEE, pp. 478–81, 2008.

[39] Garani G., Chernov A., Savvas I. and Butakova M., "*A Data Warehouse Approach for Business Intelligence*," IEEE 28th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), Napoli, Italy, pp. 70-75.2019.

[40] Farooqui N. A. and Mehra R., "*Design of A Data Warehouse for Medical Information System Using Data Mining Techniques*," 2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC), Solan Himachal Pradesh, India, pp, 2018.

[۴۱] نجفی، حامد، دانشپور، نگین، «*بهبود فرایند استخراج، تبدیل و بارگذاری در پایگاه داده تحلیلی با کمک پردازش موازی*»، مجله محاسبات نرم، جلد چهارم، شماره ۲، ص ۱۸ـ۳۱، ۱۳۹۴.

[۴۲] باباخانی، معصومه، محمدپورزنجانی، داود، صفری، لیلا، «*روشی برای حفاظت از الگوی دسترسی در داده‌های*