استفاده از قابلیت‌های XML و دیدهای ذخیره‌شده در ایجاد یک معماری پایگاه داده تحلیلی تقریباً بی‌درنگ

سیدمصطفی شفائی، نگین دانشپور و سیدمجید شفائی

چکیده: یک چالش اساسی در حوزه سکوها و کاربردها چگونگی نمایش و ترکیب نتایج به دست آمده از بخش‌های بی‌درنگ و ایستا و همچنین کاهش زمان پاسخ‌دهی به پرس‌وجوهای پردازش تحلیلی برخط در یک پایگاه داده تحلیلی تقریباً بی‌درنگ است. بنابراین محتوای مناسب می‌تواند از طریق ایجاد یک واسط مشترک از نتایج پرس‌وجوها، در یک پایگاه داده تحلیلی تقریباً بی‌درنگ تولید شود. این مقاله معماریی ارائه می‌کند که شامل یک رویکرد واسط XML, XSLT برای تولید محتوای مناسب و همچنین ساخت دیدهای ذخیره‌شده در سمت مشتری می‌باشد. در این معماری با فراهم‌کردن یک الگوی تولید خروجی مبتنی بر HTML، نحوه انتشار و ترکیب نتایج ارائه می‌شود. همچنین دو رویکرد موازی برای ترکیب‌کردن نتایج بخش‌های بی‌درنگ و ایستا از پایگاه داده تحلیلی تقریباً بی‌درنگ برای معماری پیشنهاد می‌شود. در معماری ارائه‌شده، نقش اساسی XML و فناوری‌های وابسته به آن در تولید و نگهداری محتوا در یک پایگاه داده تحلیلی تقریباً بی‌درنگ مشخص می‌شود. نتایج آزمایش‌ها، کاهش زمان پاسخ‌دهی به پرس‌وجوهای پردازش تحلیلی برخط را از طریق دیدهای ذخیره‌شده در سمت سرور و مشتری نشان می‌دهد. توابع سود معرفی‌شده برای انتخاب دیدهای ذخیره‌شده در هر دو رویکرد دیدهای ذخیره‌شده در سمت مشتری و سرور، باعث بهبود فضای نگهداری می‌شود.

کلیدواژه: **پایگاه داده تحلیلی تقریباً بی‌درنگ، دید ذخیره‌شده، XML، XQuery، XSLT.**

۱- مقدمه

با به وجود آمدن انواع سکوها[1] و دستگاه‌های مختلف اعم از کامپیوترها و تلفن‌های هوشمند، نیاز است که خروجی مناسب برای هر یک از دستگاه‌ها و کاربردها تولید شود. با استفاده از XML[2] و XSLT و دیگر فناوری‌های توسعه داده شده برای تولید و تغییر محتوا، امکان ایجاد سامانه‌ای جهت انتشار داده در بسترهای متفاوت سخت‌افزاری و نرم‌افزاری به وجود می‌آید. بنابراین قادر خواهیم بود که نتایج به دست آمده از یک پایگاه داده تحلیلی[3] را با استفاده از XML، ابزارها و امکانات توسعه داده‌شده برای آن، دست‌کاری و تغییر دهیم. از آنجایی که تمامی نتایج پرس‌وجوهای تحلیلی از طریق یک واسط مجازی گذردهی می‌شوند، نیاز به پیاده‌سازی‌های متفاوتی جهت تولید خروجی مناسب وجود دارد. بنابراین باید از یک الگوی خاص تولید خروجی، جهت انتشار نتایج پرس‌وجوها و نیز ترکیب‌نمودن نتایج پرس‌وجوهای تحلیلی استفاده کرد. این مقاله با توسعه یک واسط XSLT، مشکل تولید محتوای گوناگون و همچنین یکپارچه‌سازی نتایج در یک پایگاه داده تحلیلی تقریباً بی‌درنگ[4] را برطرف کرده است. رویکردهای گوناگونی برای طراحی یک معماری پایگاه داده تحلیلی بی‌درنگ ارائه شده است. مقالات [۱] تا [۵] با ساختن یک یا چندین بخش بی‌درنگ[5]، توانایی استفاده از داده‌های تاریخی به همراه داده‌های اخیر را مهیا کرده‌اند. برای انجام عملیات پردازش تحلیلی برخط[6] و عمل یکپارچه‌سازی نتایج، از عملگر UNION ALL موجود در پایگاه داده‌ها استفاده شده است [۲] تا [۴]. این مقاله با ارائه یک رویکرد موازی، به پرس‌وجوهایی که به طور هم‌روند از چندین بخش استفاده می‌کنند پاسخ‌دهی می‌کند.

به طور دقیق‌تر، این مقاله به ارائه رویکرد واسط XSLT برای تولید و تغییر در محتوا و همچنین ایجاد دیدهای ذخیره‌شده[7] توزیع‌شده پرداخته است. با فراهم‌کردن نتایج پرس‌وجوها در قالب XML و تعریف یک الگو برای تولید خروجی، نتایج مورد نظر با واسط XSLT ایجاد و منتشر می‌شوند. در این مقاله با فراهم‌کردن یک الگوی تولید خروجی مبتنی بر HTML، نحوه انتشار و ترکیب نتایج نشان داده شده است. از آنجایی که ممکن است خروجی مورد نظر به فرمت XML باشد، دیدهای ذخیره‌شده در سمت مشتری نیز معرفی می‌شوند. این دیدها باعث کاهش زمان پاسخ‌دهی به پرس‌وجوها خواهند شد و شامل یک زمان انقضا برای نگهداری دیدها در سمت مشتری هستند. در معماری ارائه‌شده، نقش اساسی XML و فناوری‌های وابسته به آن در تولید و نگهداری محتوا در یک پایگاه داده تحلیلی تقریباً بی‌درنگ مشخص می‌شود. نتایج آزمایش‌ها، کاهش زمان پاسخ‌دهی به پرس‌وجوهای پردازش تحلیلی برخط را از طریق دیدهای ذخیره‌شده در سمت مشتری نشان می‌دهد.

این مقاله با به کارگیری قابلیت‌های XML در صدد ساخت یک معماری مجتمع برای پایگاه داده تحلیلی تقریباً بی‌درنگ است. این قابلیت‌ها به شرح زیر است:

- استفاده از فرمت XML جهت ذخیره‌سازی و یکپارچه‌سازی نتایج.
- ایجاد دید ذخیره‌شده در سمت مشتری به منظور استفاده از زبان XQuery جهت اعمال فیلتر بر روی پرس‌وجو.
- تولید محتوای گوناگون و متفاوت مانند HTML با استفاده از پردازنده XSLT.

این مقاله در تاریخ ۵ مرداد ماه ۱۳۹۵ دریافت و در تاریخ ۲۳ آذر ماه ۱۳۹۵ بازنگری شد. این پژوهش با حمایت مالی دانشگاه تربیت دبیر شهید رجایی طبق قرارداد شماره ۳۶۲ انجام گردیده است.

سیدمصطفی شفائی، دانشکده مهندسی کامپیوتر، دانشگاه تربیت دبیر شهیدرجایی، تهران، (email: s.m.shafaei@outlook.com).

نگین دانشپور، دانشکده مهندسی کامپیوتر، دانشگاه تربیت دبیر شهیدرجایی، تهران، (email: ndaneshpour@srttu.edu).

سیدمجید شفائی، دانشکده فنی و مهندسی، دانشگاه آزاد واحد تهران مرکزی، تهران، (email: sm.shafaei71@gmail.com).

1. Platform
2. Extensible Markup Language
3. Data Warehouse
4. Real-Time Data Warehouse
5. Real-Time Partition
6. OLAP
7. Materialized Views

مقاله به صورت چندین بخش سازمان‌دهی شده است. بخش دوم، خلاصه‌ای از کارهای پیشین را بررسی می‌کند. بخش سوم، معماری پیشنهادی را شرح می‌دهد. بخش چهارم، رویکردهای دیدهای ذخیره شده را بررسی می‌کند. بخش پنجم، آزمایش و نتایج معماری پیشنهادی را ارزیابی می‌کند. بخش ششم، نتیجه گیری است.

۲- کارهای پیشین

در سال ۲۰۰۸ نویسندگان [۱]، یک معماری تقریباً بی‌درنگ مبتنی بر حافظه نهان را ارائه کردند. این معماری شامل یک مؤلفه برای تملک داده از طریق سرویس‌های وب، مؤلفه گرفتن داده‌های تغییریافته، حافظه‌های نهان چندگانه و یک پایگاه داده تحلیلی است. تملک داده از طریق سرویس‌های وب برای جمع‌آوری داده‌های ناهمگون پیاده‌سازی شده است. در این معماری، داده‌های به روز شده مبتنی بر XML و صف پیام هستند. از XML به عنوان انتقال فرمت داده از منابع خارجی و حافظه‌های نهان داخلی استفاده شده است. همچنین از حافظه‌های نهان چندسطحی برای ذخیره داده‌های بی‌درنگ استفاده شده تا به روز رسانی داده‌ها و پرس‌وجوهایی که نیاز به داده‌های بی‌درنگ دارند را تسهیل کند. هر حافظه نهان یک صف پیام را نگه می‌دارد و به وسیله مدیر صف، پیام‌ها مدیریت می‌شوند. فرمت انتقال داده در صف به فرمت XML است. پیام‌ها از طریق صف‌های هر حافظه نهان وارد حافظه نهان بعدی می‌شوند و سپس بعد از عبور از تمامی حافظه‌های نهان، داده‌ها به داخل پایگاه داده تحلیلی ایستا بارگذاری می‌شود. سامانه از حافظه‌های نهان داده‌ای چندمرحله‌ای جهت ذخیره داده‌های بی‌درنگ استفاده می‌کند و به طور هم‌زمان حافظه‌های نهان متفاوت، داده‌هایی با درجه تازگی متفاوت را برای مدتی نگهداری می‌کنند. برای مثال در این سامانه ۰- chache، ۱- chache، ۲- chache و ۳- chache داده‌ها را در ۵، ۱۰، ۳۰ و ۶۰ دقیقه به طور جداگانه و به طور موقت نگهداری می‌کنند. این معماری فقط از XML برای انتقال داده بین حافظه‌های نهان بهره برده است در حالی که معماری پیشنهادی از قابلیت‌های XML به منظور کاهش زمان پاسخ‌دهی و تولید نتایج پویا استفاده کرده است.

در سال ۲۰۰۸ نویسندگان [۲] به ارائه یک روش برای بارگذاری داده در پایگاه داده تحلیلی پرداختند. شمای پایگاه داده تحلیلی از طریق جداول تکرار (Replica) ایجاد شده است. این روند با ساختن یک ساختار تکرار از تمامی جداول پایگاه داده تحلیلی که نهایتاً داده‌های جدید را دریافت می‌کند آغاز می‌شود. این جداول به صورت خالی، بدون تعریف اندیس، کلید اصلی، هرگونه قید و بدون جامعیت خارجی ایجاد می‌شوند. برای هر جدول، یک خصوصیت جدید باید ساخته شود که جهت ذخیره‌کردن شناسه ترتیبی یکتای هر سطر اضافه‌شده در جدول موقت استفاده شود. وجود این خصیصه شناسایی‌کننده یکتابودن سطرها، امکان شناسایی ترتیب دقیق هر سطر اضافه‌شده جدید را فراهم می‌آورد. پس از این که یک رکورد وارد قسمت بی‌درنگ شد ممکن است در طول یک مدت به روز رسانی‌های متفاوتی داشته باشد. با استفاده از شناسه ترتیبی، آخرین رکورد نگهداری شده و بقیه به روز رسانی‌های قبلی حذف می‌گردند. عیب این روش در این است که اگر به روز رسانی‌های یک رکورد جدید دیرتر از زمان بخش بی‌درنگ وارد شوند، دیگر قابل شناسایی نیستند زیرا رکورد قبلی وارد پایگاه داده تحلیلی شده است. این رویکرد قابل انجام برای معماری‌های چندبخشی نیست و از این رو معماری پیشنهادی از دو رویکرد دیدهای ذخیره‌شده برای استفاده در معماری‌های تک و چندبخشی بهره برده است.

در سال ۲۰۱۴، [۳] یک معماری بی‌درنگی برای نگهداری داده‌های پویا و ایستا ارائه کرد. داده‌های ایستا در یک بخش و داده‌های پویا در بخش دیگر نگهداری می‌شوند. در هنگام ورود پرس‌وجو، پرس‌وجو به هر دو قسمت ایستا و پویا فرستاده شده و داده‌های آنها از طریق یک ادغام‌گر، ادغام می‌شوند. در این رویکرد اگر پرس‌وجو نیازی به داده‌های اخیر نداشته باشد، داده از بخش پویا واکشی و یکپارچه نیز می‌شود.

در سال ۲۰۱۳ یک معماری بی‌درنگ که فقط شامل یک بخش بی‌درنگ است، ارائه شده است. داده‌های اخیر وارد بخش بی‌درنگ شده و پس از مدتی به بخش پایگاه داده تحلیلی منتقل می‌شوند. در هنگام ورود پرس‌وجو، اگر پرس‌وجو به داده‌های اکنون نیاز داشت، پرس‌وجو به بخش بی‌درنگ ارسال می‌شود. با بررسی نیاز به داده اخیر، داده تاریخی به همراه داده اخیر یکپارچه می‌شود [۴]. معماری پیشنهادی این قابلیت را دارد که از یک یا چندین بخش داده‌ای به منظور مقایسه نتایج تحلیل در زمان‌های متفاوت استفاده کند. در معماری پیشنهادی، اجباری به استفاده از تمامی یا تعدادی از بخش‌های داده‌ای نیست بلکه پس از آماده‌شدن نتایج هر بخش، مؤلفه ادغام‌گر موجود در معماری پیشنهادی، نتایج را یکپارچه می‌کند.

مقاله [۵] در سال ۲۰۱۱ یک معماری تقریباً بی‌درنگ را که شامل چندین بخش بی‌درنگ است ارائه کرد. این معماری، درجه‌های مختلف تازگی داده را با مفهوم استفاده از چندین بخش بی‌درنگ مطرح کرده و از این رو معماری، شامل داده‌های اخیر با درجه تازگی متفاوت است. این رویکرد، حجم کاری را روی چندین بخش پخش کرده است. نویسنده روش خاصی را برای انجام پرس‌وجوی پردازش تحلیلی برخط پیشنهاد نکرده است. مقاله ارائه‌شده با پیشنهاد دو رویکرد موازی برای پاسخ‌دهی به پرس‌وجوها، قابلیت بازدهی را به این معماری نیز افزوده است. این معماری یک روش صریح برای پاسخ‌دهی به پرس‌وجوها را ارائه نکرده و از این رو معماری پیشنهادی دو رویکرد موازی برای پاسخ‌دهی به پرس‌وجوها را پیشنهاد داده است.

در سال ۲۰۱۲ نویسندگان [۶] یک معماری لایه‌ای برای پایگاه داده تحلیلی ارائه کردند. این رویکرد، یک معماری سازمانی مرجع را به صورت تعریف لایه‌های بیشتر و همراه با جزئیات هدفمند تعریف کرده است. در این معماری یک لایه برای تملک داده اضافه شده به طوری که پس از ورود داده به این لایه اگر اتصال با منابع داده قطع شود، داده در دسترس است. لایه اضافه بعدی، لایه انتشار است که شامل داده‌های هماهنگ و یکپارچه است. لایه انتشار امکان تحلیل را مستقیماً جدا از پایگاه داده تحلیلی خصوصی‌شده[1] به کاربران می‌دهد. مشکل این معماری وجود لایه‌های زیاد و عدم وجود حالت بی‌درنگ است. معماری پیشنهادی با اضافه‌کردن بخش‌های بی‌درنگ قابلیت استفاده از داده‌های اخیر به همراه داده‌های تاریخی را مهیا کرده است. معماری پیشنهادی این قابلیت را دارد که از هر بخش، داده‌ها را مستقیماً واکشی و در نهایت یکپارچه کند. معماری [۶] فقط اجازه دستیابی به داده‌های تمیز داده شده و داده‌هایی که منطق تجاری روی آنها اعمال شده را می‌دهد، در حالی که معماری پیشنهادی در این مقاله، شامل دادهای تمیز داده شده و قابل استفاده در هر بخش به منظور تحلیل در بازه‌های زمانی مختلف است.

در سال ۲۰۱۲ مؤلفین [۷] یک معماری مبتنی بر سرویس را ارائه کردند. در این معماری فرض بر این است که یک سازمان دارای منابع گران‌قیمت و پرارزش برای نگهداری پایگاه داده تحلیلی است ولی به

1. Data Mart

دلایلی پایگاه داده تحلیلی موجود در سازمان نیاز به اینچنین منابع قدرتمندی ندارد. این معماری با کمک سرویس‌های وب و پیام‌های SOAP در یک بستر وب پیاده‌سازی می‌شود. معایب این معماری شامل مواردی است که در ادامه شرح داده می‌شوند: ۱) حجم انبوه داده قابل انتقال به وسیله اینترنت نیست و بایستی از طریق جابه‌جایی فیزیکی به سایت فراهم‌کننده انتقال داده شود. ۲) روشی برای اعتبارسنجی داده ورودی وجود ندارد. معماری پیشنهادی قابلیت اجرا بر روی فضای ابری و یا شبکه را دارد. در معماری پیشنهادی نیازی به انتقال حجم انبوهی از داده در یک زمان طولانی و مشخص نیست زیرا این معماری از قابلیت بی‌درنگ بهره برده و همچنین قبل از درج داده، ورودی با استفاده از XML SCHEMA در معماری پیشنهادی بررسی می‌شود.

در سال ۲۰۰۹ نویسندگان [۸] از یک مدل داده‌ای به نام باینری رابطه‌ای به جای مدل معروف رابطه‌ای، جهت استفاده در شمای پایگاه داده تحلیلی استفاده کردند. آنها ادعا کردند که سرعت این روش نسبت به مدل رابطه‌ای بیشتر است و دیگر نیازی به استفاده از دیدهای ذخیره‌شده نیست. همچنین قابلیت انطباق‌پذیری با انواع مدل‌های منطقی مختلف (مثل ستاره‌ای و دانه برفی) را نیز دارد. مشکلات این روش عبارتند از ۱) رایج‌نبودن مدل باینری رابطه‌ای، ۲) در دسترس نبودن نرم‌افزارهای کار با این مدل، ۳) نبود زبان پرس‌وجو مختص این مدل، ۴) مشکلات مربوط به ترجمه SQL به مدل رابطه باینری و ۵) عدم وجود سامانه‌های مدیریت پایگاه داده رابطه باینری. معماری پیشنهادی بر روی یک مدل رابطه‌ای پیاده‌سازی شده و از این رو مشکلات موجود در مدل باینری رابطه‌ای در معماری پیشنهادی وجود ندارد. به منظور افزایش سرعت نیز از دو رویکرد استفاده از دیدهای ذخیره‌شده در سمت سرور و مشتری در معماری پیشنهادی استفاده شده که استفاده از مدل باینری رابطه‌ای در ازای مدل رابطه‌ای در معماری پیشنهادی را برطرف کرده است.

مقاله [۹] روشی ترکیبی از محیط ذخیره‌سازی خارجی پویا و یک فناوری تکرار آینه‌ای پویا را برای حل رقابت بین پرس‌وجوهای تحلیلی و به روز رسانی‌های پردازش تحلیلی برخط ارائه کرده است. همچنین برای پشتیبانی از تحلیل داده بلادرنگ، ETL را به دو نوع ETL بلادرنگ و ETL تاریخچه‌ای یا سنتی تقسیم کرده‌اند. در این راستا نیز کارهایی نظیر [۱۰] و [۱۱] انجام شده است.

در [۱۲] ابتدا الگوریتم انتخاب پویای دیدهای ذخیره‌شده از طریق انتخاب دیدهایی که از نتایج پرس‌وجوهای تحت قید زمان اجرا و فضای ذخیره‌سازی سامانه به دست آمده، ارائه شده است. مؤلفین سپس یک سیاست جدید برای مشخص‌کردن نگهداری پویای دیدهای ذخیره‌شده بر طبق تناوب دسترسی و بارکاری سامانه پیشنهاد داده‌اند. این معماری رویکردی را برای نگهداری دیدهای ذخیره‌شده در معماری پایگاه داده تحلیلی بی‌درنگ- بخش‌بندی شده- ارائه نکرده است. معماری پیشنهادی تابع سود و الگوریتمی را برای نگهداری دیدهای ذخیره‌شده در معماری پایگاه داده تحلیلی تقریباً بی‌درنگ- بخش‌بندی شده- در مقایسه با این معماری پیشنهاد داده است.

نویسندگان [۱۳] معماری و روشی را جهت تقسیم‌کردن داده‌های ورودی به دو نوع ارائه کرده‌اند. این دو نوع عبارتند از کلی که اغلب تغییر نمی‌کنند (یا در زمان‌های طولانی تغییر پیدا می‌کنند) و داده تراکنشی که منظم به روز می‌شود. به عنوان مثال جدول محصولات به عنوان داده کلی و جدول سفارش به عنوان داده تراکنشی در نظر گرفته می‌شود. به همین دلیل داده کلی که حجم کم و تقریباً ثابتی دارد می‌تواند به راحتی ذخیره شود و سرعت عملیات بالا رود. در این راستا [۱۴] تا [۱۶] روش‌هایی را برای تقسیم داده‌های ورودی که قابلیت تقریباً بی‌درنگ در آن مهیا شده است را ارائه کرده‌اند.

مقالات [۱۷] و [۱۸] چگونگی تأثیرات بارکاری پرس‌وجو و تأثیراتی که به وسیله فرایند ETL و عوامل تحت تأثیرگذاری مانند نوع استراتژی بارگیری، اندازه داده بارگیری شده، شاخص‌گذاری، محدودیت‌های یکپارچه‌سازی، فعالیت تازه‌سازی روی داده‌های خلاصه و بخش‌بندی جداول حقیقت است را مورد تجزیه و تحلیل قرار داده‌اند.

در سال ۲۰۰۳ ساخت یک ابزار مبتنی بر داده جریانی پیشنهاد شد که تازگی داده‌ها و ادغام پیوسته را ممکن ساخت. به علاوه معیارهای دیگر توسط نویسندگان ارزیابی شده است. هر چند این راه حل‌های مبتنی بر جریان، نتایج تقریبی را برمی‌گردانند، به علت داشتن نتایج دقیق با استفاده از جریان پیوسته هزینه زیادی متحمل خواهند شد [۱۹]. مقالات [۲۰] و [۲۱] به ارائه معماری‌هایی برای انجام عملیات ETL و بهینه‌سازی در راستای معماری پایگاه داده تحلیلی جریانی پرداخته‌اند.

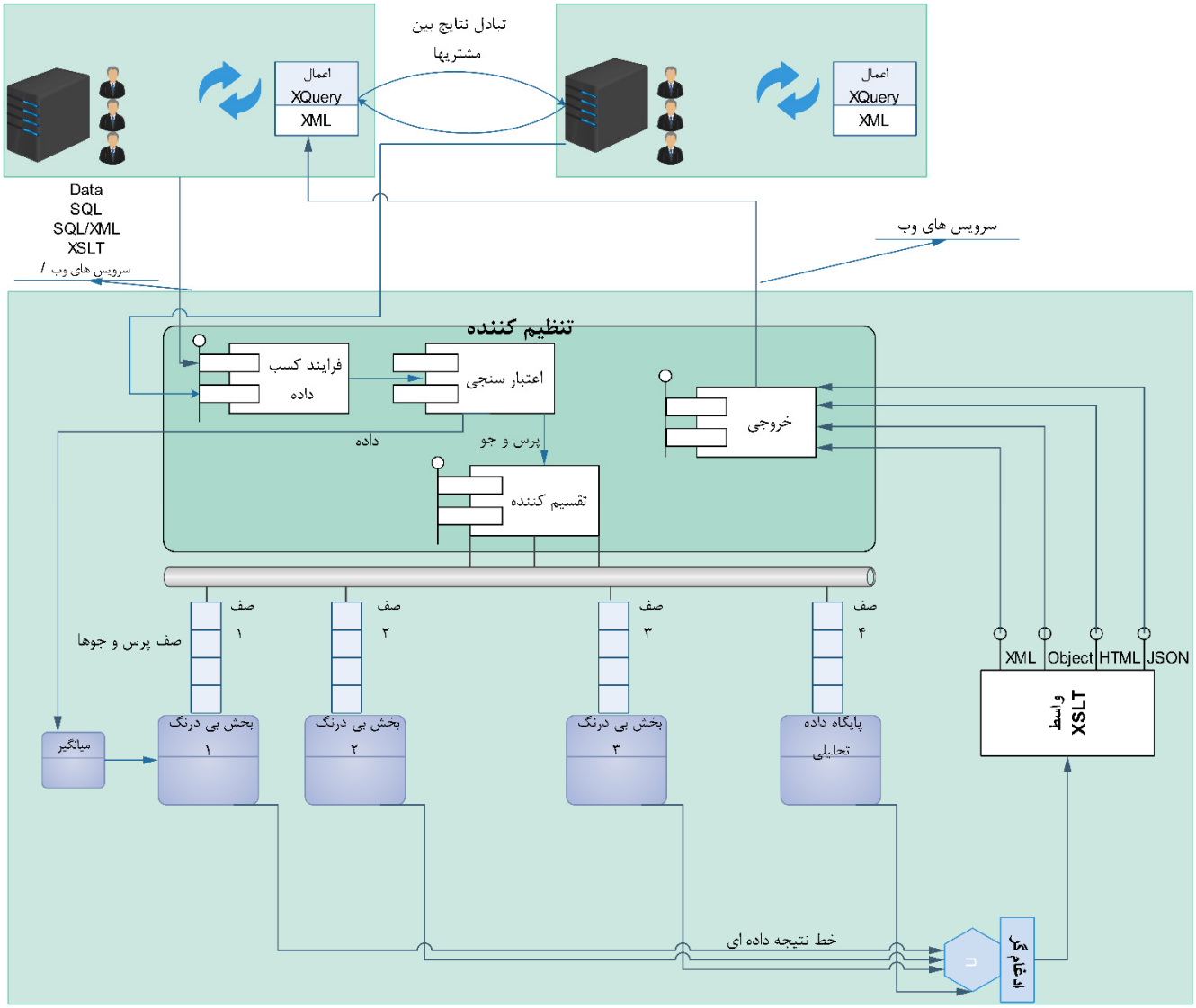
در سال ۲۰۰۷ نویسندگان مقاله به ارائه یک سرویس تجاری مبتنی بر معماری سرویس‌گرا پرداختند. در این لایه مدیریت داده شامل یک پایگاه داده تحلیلی تقریباً بی‌درنگ به علاوه مسیریابی، امنیت، پردازش پیام و قابلیت مدل‌سازی است [۲۲]. معماری پیشنهادی از سرویس‌های وب و XML برای انتقال داده و نیز افزایش سرعت پردازش پرس‌وجو از طریق معرفی دیدهای سمت مشتری در مقایسه با این معماری بهره برده است.

در [۲۳] نویسندگان به ارائه یک الگوریتم انتخاب مجموعه‌ای از دیدهای ذخیره‌شده با استفاده از رویکرد داده‌کاوی تکرار مجموعه پرداخته‌اند. در این رویکرد، مجموعه پرس‌وجوها به یک تراکنش پایگاه داده تبدیل می‌شوند. این الگوریتم فقط مجموعه‌ای از دیدهای ذخیره‌شده در بخش پایگاه داده تحلیلی را انتخاب می‌کند و به این دلیل الگوریتم ارائه‌شده قابلیت استفاده در معماری‌های بی‌درنگ- چندبخشی- را ندارد.

نویسندگان [۲۴] الگوریتم بهینه‌سازی ازدحام ذرات که یک الگوریتم تصادفی است را برای انتخاب بهینه مجموعه‌ای از دیدهای ذخیره‌شده بر روی یک چارچوب مشبک پیاده‌سازی کرده‌اند. سپس نتایج به دست آمده را با الگوریتم ژنتیک برای اثبات کارایی الگوریتم بهینه‌سازی ازدحام ذرات مقایسه نموده‌اند.

مقاله [۲۵] یک رویکرد خودکار را برای طراحی شمای یک پایگاه داده تحلیلی از طریق XML Schema موجود از منابع داده‌ای به فرمت اسناد XML پیشنهاد کرده است. این رویکرد از منابع XML Schema موجود، چندین طراحی مختلف برای شمای پایگاه داده تحلیلی تولید می‌کند. در معماری پیشنهادی، شمای پایگاه داده تحلیلی ثابت در نظر گرفته شده است. همچنین در معماری پیشنهادی چندین XML Schema بین مشتری‌ها به اشتراک گذاشته می‌شود تا داده‌های خود را منطبق با یکی از آنها به سرور ارسال کنند.

مقاله [۲۶] یک رویکرد مبتنی بر XML Schema را برای ساخت خودکار شمای ستاره‌ای پایگاه داده تحلیلی معرفی کرده است. این رویکرد



شکل ۱: معماری پیشنهادی.

با بهره‌گیری از استانداردهای UML، XML، QTV و XSLT اقدام به ساخت شمای پایگاه داده تحلیلی می‌کند.

معماری پیشنهادی در این مقاله، منطق و عملکرد پایگاه داده تحلیلی موجود را تغییر نمی‌دهد بلکه سعی می‌کند با استفاده از XML و قابلیت‌های آن، روشی را برای تولید و انتشار نتایج و همچنین کاهش زمان پاسخ‌دهی به پرس‌وجوها از طریق معرفی دیدهای ذخیره‌شده در سمت سرور، مشتری و روش‌های موازی پاسخ‌دهی به پرس‌وجوهای پردازش تحلیلی برخط ارائه نماید. معماری‌های دیگر، فقط از بخشی از قابلیت‌های XML مانند استخراج داده به پایگاه داده تحلیلی استفاده کرده‌اند. در این معماری نیازی نیست که پایگاه داده تحلیلی مبتنی بر XML ساخته شود بلکه از قابلیت‌های XML نظیر پردازه XSLT، XML Schema و زبان XQuery در یک پایگاه داده تحلیلی- رابطه‌ای- بی‌درنگ- استفاده می‌شود. این قابلیت‌ها فقط در معماری‌های مبتنی بر XML به منظور ذخیره‌سازی و جستجو در یک پایگاه داده تحلیلی مبتنی بر XML مشاهده می‌شود. تابع سود انتخاب دیدهای ذخیره‌شده سمت سرور پیشنهادی این مقاله، ارزش بخش‌های بی‌درنگ و پایگاه داده تحلیلی را برای نگهداری دیدهای ذخیره‌شده در نظر می‌گیرد، زیرا از یک فضای مشترک برای نگهداری همه دیدهای ذخیره‌شده استفاده می‌شود. در مقالات ارائه‌شده، روش‌هایی را برای در نظر گرفتن حجم بخش‌های بی‌درنگ مد نظر قرار نداده‌اند. مقالات ارائه‌شده صرفاً دیدهای ذخیره‌شده در بخش پایگاه داده تحلیلی را نگهداری می‌کنند. مقاله پیشنهادی الگوریتم و تابع سودی را برای نگهداری مناسب دیدهای ذخیره‌شده در همه بخش‌های داده‌ای پیشنهاد داده و همچنین تابع سود انتخاب دیدهای ذخیره‌شده در سمت مشتری پیشنهادی در این مقاله از زمان انقضای افزایشی برای نگهداری دیدهای ذخیره‌شده استفاده می‌کند. الگوریتم دیدهای ذخیره‌شده در سمت مشتری به صورت توزیع‌شده عمل می‌کند و از این رو این مقاله به ارائه یک الگوریتم جدید با به کارگیری قابلیت توزیع‌شده پرداخته است. این مقاله با یکپارچه‌سازی قابلیت‌های XML و دیدهای ذخیره‌شده در یک پایگاه داده تحلیلی رابطه‌ای- بی‌درنگ، یک معماری مجتمع را پیشنهاد کرده است.

۳- معماری پیشنهادی

ساختار معماری که شامل قسمت‌های بخش‌ها، فرایند پردازش پرس‌وجو بر روی بخش‌ها و رابط XSLT است در شکل ۱ نشان داده شده است. هر بخش به عنوان معرف داده‌هایی با عمر زمانی متفاوت (زمان سپری‌شده استخراج داده از پایگاه داده به نخستین بخش) است که شامل یک جدول حقیقت و جداول ابعاد است. برای سادگی الگوریتم ادغام، استراتژی نگهداری یک شمای ستاره‌ای برای هر بخش اتخاذ شده

Scheduling OLAP Queries Algorithm

Assumptions
O = OLAP query
Partitions[] // keep data freshness

```
1.  foreach (partition p in Partitions) do
2.        Slaves.CreateSlaveTask(p, O)
3.  end for
4.   // Wait for all of the results and then integrate them
5.  Slaves.WaitAll()
6.  Result = Merge(Slaves)
7.  // Sending integrated results into XSLT interface
8.  IXSLT(Result, outputFormat)

9.  //Creates and assigns sub query task to a partition
10.  function CreateSlaveTask(Partition p, OLAP o)
11.       Result = DBMS.Partitions[p].Execute(o)
12.       return Result
13. end function

14. // It is used to merge the results from all of the salves
15. function Merge(Slaves[] s)
16.       Merge input according to aggregation function
          or union the results into xml format
17. end function
```

شکل ۲: الگوریتم موازی برای پرس‌وجوهای پردازش تحلیلی برخط.

که در ادامه به توضیح هر یک از بخش‌ها می‌پردازیم.

فرایند کسب داده که همانند الگوی Master/Slaves است، مسئول دریافت داده‌های ورودی است. این فرایند از طریق مؤلفه فرایند کسب داده موجود در معماری که همانند Master است مدیریت می‌شود. داده‌ها از طریق سرویس‌های وب به این مؤلفه تحویل داده می‌شوند و این مؤلفه با ایجاد زیروظیفه‌ها[1]، داده‌های ورودی را از هر منبع دریافت می‌کند. منابع ممکن است از تگ‌هایی مختلف برای تحویل داده استفاده کنند که زیروظیفه‌های ایجادشده، مسئول بررسی اعتبار آنها هستند. مؤلفه اعتبارسنجی مسئول اعتبارسنجی داده‌های ورودی است.

مؤلفه تقسیم‌کننده همانند Master و بخش‌ها نیز Slaveها هستند. Slaveها زیروظیفه‌هایی برای انجام عملیات پردازش تحلیلی برخط بر روی هر بخش هستند. پس از انجام عمل پرس‌وجو توسط هر زیروظیفه، نتایج به مؤلفه ادغام‌گر، جهت ادغام فرستاده می‌شوند و پس از عمل ادغام، نتیجه یک‌پارچه شده که به فرمت XML است به رابط XSLT جهت تولید محتوای مناسب ارسال می‌شود.

با فراهم‌کردن فرمت XML از نتایج یک‌پارچه شده از هر Slave، مؤلفه XSLT وظیفه تولید محتوای مناسب را بر عهده دارد. برای ایجاد یک الگوی XSLT جهت تولید محتوا به یک پیاده‌سازی نیاز است که این پیاده‌سازی به واسط XSLT تحویل داده می‌شود.

نتایج به دست آمده از پرس‌وجوی پردازش تحلیلی برخط می‌تواند از دو طریق یکپارچه شود: ۱) از طریق افزودن نتایج به یکدیگر و ۲) از طریق محاسبه توابع تجمعی. رویکرد اول، نتایج هر بخش را به انتهای دیگری اضافه می‌کند و رویکرد دوم نتایج به دست آمده از هر بخش را بر حسب توابع تجمعی ادغام می‌کند. با فرض این که جداول حقیقت بخش‌ها بر روی n بخش با جداول $T_1,T_2,\ldots,T_n$ توزیع شده است، تابع $f(T_i,c)$ نتیجه تابع تجمعی بر جدول T_i را بر حسب بعد C نشان می‌دهد. روابط (۱) و (۲) فرمول محاسبه توابع تجمعی و افزودن نتایج به یکدیگر را به ترتیب نشان می‌دهند

$$SUM(T,c)=\sum_{1}^{n} SUM(T_i,c) \qquad (۱)$$

$$SUM(T,c)= SUM(T_i,c)\cup SUM(T_{i+1},c)\cup\ldots\cup SUM(T_n,C) \qquad (۲)$$

الگوریتم دو رویکرد یکپارچه‌سازی داده‌ها در شکل ۲ نشان داده شده است.

واسط XSLT فرایند تولید محتوا به فرمت‌های گوناگون مورد نیاز مصرف‌کننده را فراهم می‌کند. در این واسط از پردازنده XSLT جهت تولید محتوا استفاده شده است. پردازنده XSLT جهت تولید خروجی از داده ورودی به فرم XML استفاده می‌کند و از این رو نتایج پرس‌وجوهای پردازش تحلیلی بر خط پایگاه داده تحلیلی توسط مؤلفه ادغام‌گر از طریق فرمت XML یک‌پارچه می‌شوند. از این طریق قابلیت ارسال نتایج به مشتری و همچنین تولید محتوای مناسب به وجود می‌آید. امکان تولید هم‌زمان چندین محتوا برای دستگاه‌های مختلف از طریق منبع XML وجود دارد و در واقع یک انتشاردهنده پویا برای منبع XML ایجاد می‌شود.

٤- رویکردهای دیدهای ذخیره‌شده

دو رویکرد دیدهای ذخیره‌شده برای معماری پیشنهاد شده که رویکرد نخست، دیدهای ذخیره‌شده با استفاده از فرکانس تکرار پرس‌وجو و رویکرد دوم، دیدهای ذخیره‌شده در سمت مشتری است. دیدهای ذخیره‌شده با استفاده از فرکانس تکرار پرس‌وجو به صورت مجزا برای هر بخش ایجاد می‌شود. بخش‌هایی که زمان پاسخ‌گویی به پرس‌وجوها در آنها به مراتب زمان‌بر است از این رویکرد استفاده می‌کنند. در صورت نبود فضای کافی برای ذخیره‌سازی دیدهای ذخیره‌شده از رویکرد دیدهای ذخیره‌شده در سمت مشتری استفاده می‌شود.

٤-۱ دیدهای ذخیره‌شده با استفاده از فرکانس تکرار پرس‌وجو

هر پرس‌وجو نیاز به درجه‌های تازگی داده و یا بخش‌ها برای پاسخ‌دهی دارد. مجموعه پرس‌وجوهای ورودی کاربران در صف مؤلفه تقسیم‌کننده که در شکل ۳ نشان داده شده قرار می‌گیرند و سپس این مؤلفه اقدام به ساخت وظیفه‌ها جهت اجرای پرس‌وجوها بر روی بخش‌های مناسب می‌کند. ممکن است برخی از این پرس‌وجوها تکراری و یا قابل پاسخ‌دهی با دیگر پرس‌وجوهای داخل صف باشند. شکل ۳-الف مجموعه چند پرس‌وجوی داخل صف و شکل ۳- ب نحوه تخصیص وظایف را نشان می‌دهد. Q(index) شماره و ترتیب ورود پرس‌وجو، p(index) بخش(های) مورد نیاز پرس‌وجو و T(index) شماره وظیفه‌ای که ایجاد خواهد شد را نشان می‌دهد.

هر پرس‌وجو شامل چندین زیر پرس‌وجو (بخش‌های مورد نیاز) بوده و برای هر زیر پرس‌وجو همان طور که در الگوریتم ۱ نشان داده شده است یک وظیفه ایجاد می‌شود. پرس‌وجوی Q۱ نیاز به اجرای پرس‌وجو بر روی تمامی بخش‌ها دارد پس ۴ وظیفه برای اجرای موازی ایجاد می‌شود. پرس‌وجوی Q۲ نیز همانند پرس‌وجوی Q۱ برای اجرا سه وظیفه ایجاد

1. Slaves

Q1: p1, p2, p3, p4

Q2: p1, p2, p3

Q3: p3, p4

Q4: p1, p2

Qn: ...

(الف)

Q1:	T1	T2	T3	T4
Q2:	T5	T6	T7	
Q3:			T8*	T9*
Q4:	T10	T11		

(ب)

شکل ۳: (الف) مجموعه پرس‌وجوهای ورودی کاربران و (ب) وظیفه‌هایی که ایجاد خواهند شد.

می‌کند. پرس‌وجوی Q۳ یک پرس‌وجوی تکراری و یا قابل پاسخ‌گویی با پرس‌وجوی Q۱ است، پس تا زمانی که هر کدام از وظیفه‌های T۳ و T۴ از Q۱ پاسخ‌دهی نشده‌اند، وظیفه‌های T۸ و T۹ از پرس‌وجوی Q۳ معلق خواهند بود. پس از این که وظیفه‌های T۳ و T۴ از پرس‌وجوی Q۱ پاسخ‌دهی شدند، برای هر یک از آنها یک دید ذخیره‌شده از نتایج آنها ایجاد می‌گردد. وظیفه‌های T۸ و T۹ از پرس‌وجوی Q۳ تا زمان به وجود نیامدن دیدهای ذخیره‌شده ایجاد نمی‌شوند. دلایل به تعویق انداختن وظیفه‌های T۸ و T۹ از پرس‌وجوی Q۳ این موارد است: ۱) از نتایج دید ذخیره‌شده استفاده شود، ۲) ایجاد وظیفه جدید سربار دارد و ۳) سامانه تعداد محدودی از وظیفه‌های هم‌روند را می‌تواند داشته باشد. زمانی که وظیفه‌های T۳ و T۴ از پرس‌وجوی Q۱ وظیفه خود را تمام کردند، این وظیفه‌ها از بین نخواهند رفت بلکه برای اجرای وظیفه‌های T۸ و T۹ از پرس‌وجوی Q۳ مورد استفاده قرار می‌گیرند.

الگوریتم دیدهای ذخیره‌شده با استفاده از فرکانس تکرار پرس‌وجو در الگوریتم شکل ۴ نشان داده شده است. این الگوریتم از یک فضای ذخیره‌سازی مشترک برای نگهداری دیدهای ذخیره‌شده در هر بخش (بخش‌های انتخاب‌شده جهت ایجاد دید به منظور افزایش سرعت) استفاده می‌کند. این الگوریتم مبتنی بر فرکانس تکرار و دستیابی پرس‌وجوها در بخش‌ها و نیز وظیفه‌های ایجادشده است. از آنجایی که فرکانس دستیابی پرس‌وجو در بخش‌ها مد نظر قرارگرفته است این الگوریتم در صورت خالی‌نبودن فضای ذخیره‌سازی از یک روش اولویت‌بندی برای انتخاب دید جهت نگهداری استفاده می‌کند. در این روش از دو پارامتر شامل پرس‌وجوهایی با فرکانس دستیابی پایین و بخش‌های با حجم پایین برای ساخت یک لیست مرتب استفاده می‌شود. اگر فضای ذخیره‌سازی پر شود، دید ذخیره‌شده‌ای از فضای ذخیره‌سازی حذف می‌گردد که کمترین فرکانس دستیابی و در کم‌حجم‌ترین بخش قرار دارد. هر مجموعه پرس‌وجو به نام یک جلسه به عنوان ورودی الگوریتم مورد استفاده قرار می‌گیرد و الگوریتم بر روی این مجموعه ورودی کار می‌کند. یک هدف از ساختن یک مجموعه ورودی از پرس‌وجوها، محدودبودن منابع سامانه

DVMP Algorithm

Assumptions
Q = Query
S = Space of saving view
M = Materialized view as output
Qu = Queue

```
1.  M = {}
2.  while(Qui is not empty) do
3.    if ((vi ∉ M)) then
4.        Compute BS(vi, M)
5.        while((Svi + Sm) > S) do
6.           // v ∉ Ts can be dropped.
7.           select v ∈ M and BS(v, M) < BS(vi, M)
             and v ∉ Ts and BS(v, M) is minimum
8.           M = M – {v}
9.        end while
10.       if ((Svi + Sm) < S) then
11.          M = M U {vi}
12.          end if
13.  end if
14. end while
```

شکل ۴: الگوریتم DVMP.

در ساخت و توازی وظیفه‌ها است. الگوریتم DVMP[1] به طور پویا دیدها را انتخاب و ذخیره می‌کند، به طور پیوسته پرس‌وجوهای ورودی را بازبینی می‌کند و بهترین مجموعه دیدها را با در نظر داشتن محدودیت فضا و وظیفه‌هایی که از دیدهای ذخیره‌شده قبلی استفاده می‌کنند، ذخیره می‌سازد.

این الگوریتم به این صورت عمل می‌کند که تا زمانی که فضای کافی موجود و تابع سود بیشترین باشد، نتایج پرس‌وجو ذخیره می‌گردد. به این دلیل تابع سود برای هر پرس‌وجوی ورودی مورد بررسی قرار می‌گیرد که ساختن دید هزینه‌بر است، پس تا زمانی که واقعاً نیاز به ساخت یک دید نیست از این کار صرف نظر می‌شود. اگر تابع سود مینیمم و فضای کافی موجود نباشد باید از روش جایگذاری استفاده شود و این کار با استفاده از تابع سود صورت می‌گیرد. الگوریتم، حالت جاری دیدهای ذخیره‌شده، دید مورد نیاز جدید برای ذخیره‌سازی، محدودیت فضا و محدودیت حذف دیدهای ذخیره‌شده جاری که برخی از وظیفه‌های جاری نیاز به استفاده از آنها را دارند، به عنوان ورودی دریافت می‌کند. محدودیت حذف دیدهای ذخیره‌شده جاری که برخی از وظیفه‌های جاری نیاز به استفاده از آنها را دارند به این دلیل به عنوان یکی از محدودیت‌ها مد نظر قرار گرفته که وظیفه‌های استفاده‌کننده از دید، به سرعت کار خود را به دلیل استفاده از نتایج آماده درون دید تمام می‌کنند. همچنین فرصت برای ساخت دید جدید به علت تکرار آن در آینده نیز مهیا است. هر دید ذخیره‌شده در صورتی که سود کمتری از نتیجه جدید داشته باشد، برای حذف مورد بررسی قرار می‌گیرد. در قدم اول، مجموعه‌ای از دیدها ساخته می‌شود. اگر در طول این فرایند نتوانیم دیدهایی برای حذف بیابیم، جستجو متوقف شده و نتیجه دید، ذخیره نمی‌شود. در هر مجموعه پرس‌وجو، فرکانس تکرار پرس‌وجو به مجموعه بعدی نیز انتقال داده می‌شود و اگر پرس‌وجویی تکرار در لیست داشته باشد، به فرکانس قبلی پرس‌وجو اضافه

1. Dynamic View Materialization in Partitions

A Algorithm

Assumptions
R = Result of an OLAP query
S = Space of client node
Th = Threshold value
C = An incremental value
1. **if** ((Svi + Sm) < S) **then**
2. M = M U {vi}
3. **if** (Vi.IsTransmited) **then**
4. Vi.ExpiredTime = Vi.ExpiredTime * C
5. **else**
6. Vi.ExpiredTime = Th
7. **end if**
8. **else**
9. **while**((Svi + Sm) > S) **do**
10. *// Remove MV(s) that have the lowest expired time;*
11. **select** v ∈ M **and** BC(v, M) < BC(vi, M) **and** BC(v, M) **is minimum**
12. M = M – {v}
13. **end while**
14. **if** ((Svi + Sm) < S) **then**
15. M = M U {vi}
16. **end if**
17. **end if**

شکل ۵: الگوریتم A.

می‌گردد. به عنوان یک مثال ساده اگر پرس‌وجوهای Q۱، Q۲ و Q۳ که نتایج آنها ذخیره شده و به ترتیب ۵، ۳، ۲ تکرار در مجموعه پرس‌وجوی اول داشته‌اند و همان پرس‌وجوها در مجموعه پرس‌وجوی دوم به ترتیب ۲، ۳، ۶ تکرار داشته باشند و هزینه ساخت دید در آنها یکسان باشد، فرکانس نهایی آنها به ترتیب برابر با ۷، ۶ و ۸ است. پس تابع سود نیز در این مجموعه پرس‌وجوی جدید به نفع پرس‌وجوی Q۱ خواهد بود. در این الگوریتم حجم بخش‌ها به عنوان یک عامل مهم در نگهداری و یا حذف دیدها مؤثر است و بخش‌هایی که حجم بیشتری از داده را داشته باشند، مقدار بالاتری برای تابع سود دارند. تابع سود الگوریتم در (۳) نشان داده شده است

$$BS(v,M)=(\sum_{i=1}^{n} f_{qoldi}+f_{q\,new})\times c(v_i,P) \qquad (۳)$$

در (۳) $c(v_i,P)$ هزینه اجرای پرس‌وجو در بخش، n تعداد جلسات، f_{qold} فرکانس پرس‌وجو در نشست‌های قبلی، $f_{q\,new}$ فرکانس پرس‌وجو در نشست جاری، S_v فضای ذخیره‌سازی مشترک دیدهای ذخیره‌شده و شرط $v\in T_s$ یک قانون برای جلوگیری از حذف دید ذخیره‌شده است. این بدین معنی است که اگر همیشه هزینه ساخت دید جدید بیشتر از دیدهای موجود که وظیفه‌هایی منتظر استفاده از آنها هستند است، ساخت دید جدید رد شود. اگر هزینه ساخت متغیر باشد، شرط موجود در الگوریتم حذف و سود ساخت دید جدید محاسبه می‌شود.

۴-۲ دیدهای ذخیره‌شده در مشتری‌ها

هدف از این رویکرد به دست آوردن فضای ذخیره‌سازی بیشتر برای دیدهای ذخیره‌شده است. از آنجایی که یکی از فاکتورهای مهم در ساخت و نگهداری دیدهای ذخیره‌شده، میزان فضای در دسترس است استفاده از بخشی از فضای ذخیره‌سازی مشتری‌ها برای نگهداری دیدهای ذخیره‌شده مفید خواهد بود. این رویکرد شبیه به پروتکل اشتراک فایلBitTorrent برای دانلود فایل‌ها در سطح شبکه سراسری و یا محلی است. فقط کافی است مشتری‌هایی که قابلیت نگهداری دیدها را دارند از یک برنامه جانبی برای انجام مدیریت و نگهداری دیدهای ذخیره‌شده استفاده کنند. این برنامه نیز می‌تواند از طریق سرویس‌دهنده برای مشتری فراهم شود. استفاده از سرویس‌دهنده برای مدیریت دیدهای ذخیره‌شده در سازمان‌ها که از یک شبکه محلی بهره می‌برند پیشنهاد می‌شود.

۴-۲-۱ مدیریت دیدهای ذخیره‌شده در مشتری‌ها

دیدهای ذخیره‌شده شامل یک پارامتر زمان انقضا هستند که این پارامتر در زمان ذخیره نتایج به عنوان یک دید به آن نسبت داده می‌شود. زمان انقضا برای به روز رسانی دیدهای ذخیره‌شده استفاده می‌شود و پس از اتمام این زمان، دید مذکور از مشتری حذف می‌گردد.

زمانی که یک دید برای اولین بار در یک مشتری ذخیره می‌شود مقدار ثابتی به عنوان زمان انقضا به آن دید نسبت داده می‌شود. اگر دید ذخیره‌شده مورد دستیابی و یا به مشتری دیگر منتقل شود، زمان انقضای قبلی با یک مقدار افزایشی اضافه می‌گردد و سپس به عنوان زمان انقضای فعلی دید ذخیره‌شده ثبت می‌گردد. این مقدار افزایشی باعث می‌شود آن دیدهایی که مکرراً مورد دستیابی قرار می‌گیرند، شانس بیشتری را برای نگه داشته شدن داشته باشند. با هر بار دستیابی به دید ذخیره‌شده، مقدار افزایشی نیز افزایش پیدا می‌کند تا این که به یک آستانه مشخص برسد که در این صورت دیگر این مقدار، افزایش پیدا نمی‌کند و ثابت می‌ماند. هر مشتری نیز یک پارامتر فضای قابل دسترس برای ذخیره دیدهای ذخیره‌شده دارد. الگوریتم دیدهای توزیع‌شده در شکل ۵ آمده است.

الگوریتم A برای حالتی است که نتیجه پرس‌وجو برای نخستین بار در یک مشتری ذخیره می‌گردد. اگر مشتری‌ای (چه خود مشتری و دیگر مشتری‌ها) پرس‌وجویی که قابل پاسخ‌گویی با یکی از دیدهای ذخیره‌شده در یکی از مشتری‌ها است را داشته باشد، مشتری مبدأ به طور مستقیم نتیجه را به مشتری مقصد ارسال می‌کند. برای انجام فیلترهایی بر روی دید ذخیره‌شده در سمت مشتری (به عنوان مثال DrillDown) از زبان XQuery جهت اعمال پرس‌وجو بر روی دیدهای ذخیره‌شده در سمت مشتری استفاده می‌شود. زمانی که نتیجه پرس‌وجو از طریق یک دید ذخیره‌شده پاسخ‌دهی شود دید ذخیره‌شده در سمت مبدأ، حذف می‌شود و نسخه ارسال‌شده به مشتری مقصد به عنوان دید ذخیره‌شده در آن مشتری با زمان انقضای قبلی ذخیره می‌گردد. هدف از حذف دید از مشتری مبدأ، جلوگیری از وجود تکرار در مشتری‌ها است. زمان انقضای قبلی به دلیل جلوگیری از کهنگی نتیجه، حفظ می‌شود و از این طریق می‌تواند دید را پس از اتمام زمان انقضا حذف و یا به روز رسانی نمود. اگر در هنگام ساخت دید جدید یا انتقال یک دید به مشتری دیگر، مشتری، فضای ذخیره‌سازی کافی را برای ذخیره‌سازی این دید نداشته باشد، تابع سود محلی برای انتخاب دیدهای ذخیره‌شده در سمت مشتری به منظور حذف آنها تصمیم‌گیری می‌کند.

فرمول (۴) تابع سود انتخاب دید محلی را نشان می‌دهد. تابع سود، زمان انقضای دید داده‌شده (T) به تابع را در مشتری مشخص می‌کند. تا زمانی که فضای کافی برای دید جدید مهیا نشود، دیدهایی با زمان انقضای کمتر حذف می‌گردند. در دیدهایی که زمان انقضای بیشتری دارند، این زمان انقضا با توجه به فرکانس دستیابی به آنها افزایش پیدا کرده است و به همین دلیل شانس بیشتری برای نگه داشته شدن دارند. الگوریتم B در شکل ۶ نحوه استفاده از دیدهای ذخیره‌شده را نشان داده است

B Algorithm

Assumptions
O = OLAP query
A = Source client
B = Destination client
S = Server
One O has been sent by B to S

```
1.  // Server checks is there a MV that is equal with O
2.  if (S.FindMV(O)) then
3.      S.A.Send(O, B)
4.      S.A.Remove(O)
5.      S.B.CreateMV(O)
6.  // Server checks is there a MV is similar with O
7.  else if (S.FindOptMV(O)) then
8.      S.A.Filter(O).Send(B)
9.      S.A.Remove(O)
10.     S.B.CreateMV(O)
11. else
12.     S.Execute(O)
13.     S.Send(B)
14.     S.B.Run(algorithm A)
15. end if
```

شکل ۶: الگوریتم B.

$$BC(v,M)=T \qquad (۴)$$

الگوریتم B، ۲ مشتری A و B و یک سرور را جهت اجرای یک پرس‌وجوی پردازش تحلیلی برخط نشان می‌دهد. یک پرس‌وجوی پردازش تحلیلی برخط از طریق مشتری B به سرور ارسال شده و نتیجه این پرس‌وجو در مشتری A وجود دارد. سرور با بررسی لیست دیدهای ذخیره‌شده در مشتری‌ها، وجود نتیجه را در مشتری A متوجه می‌شود. اگر پرس‌وجو نیازی به پالایه نداشته باشد، نتیجه از مشتری A به B ارسال می‌شود. دید موجود در مشتری A حذف و در مشتری B با همان تاریخ انقضای قبلی ذخیره می‌گردد و در غیر این صورت، سرور مجبور به اجرای پرس‌وجو خواهد بود.

۵- آزمایش

رویکردهای موازی و دیدهای ذخیره‌شده در سمت مشتری و سرور در یک پایگاه داده تحلیلی تقریباً بی‌درنگ پیاده‌سازی و اجراشده است. بخش‌بندی پایگاه داده تحلیلی تقریباً بی‌درنگ با قرار دادن ۴ بخش به حجم GB ۶ صورت گرفته است.

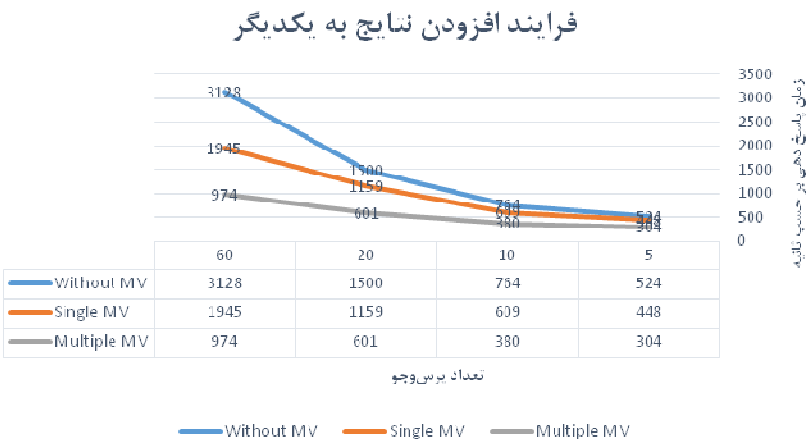
در آزمایش از ارزیاب SSB[1] جهت ساخت شمای ستاره‌ای، پرکردن داده بخش‌ها و پرس‌وجوهای پردازش تحلیلی برخط استفاده شده است. SSB شامل ۴ بعد (Part, Customer, Date, Supplier) و یک جدول حقیقت (LineOrder) است و به منظور ارزیابی زمان پاسخ‌دهی به پرس‌وجوها، از ۱۳ پرس‌وجوی پردازش تحلیلی برخط با اعمال فیلترهای مختلف استفاده شده است. از TPC-H dbgen به منظور پرکردن داده در ابعاد و جدول حقیقت استفاده شده است. جزئیات بیشتر در مورد SSB در [۱۸] موجود است. Intel (R) Core (TM) i۷ ۲٫۲۰ GHz با GB ۴ حافظه اصلی و Microsoft SQL Server ۲۰۱۴، پیکربندی سامانه مورد استفاده بوده است.

سناریوهای افزودن نتایج به یکدیگر و محاسبه توابع تجمعی برای بررسی آزمایش الگوریتم DVMP مورد نظر قرار گرفته‌اند. در سناریوی اول از الگوریتم افزودن نتایج به یکدیگر و در سناریوی دوم نیز از الگوریتم محاسبه توابع تجمعی استفاده شده است. همچنین الگوریتم افزودن نتایج به یکدیگر نیز برای بررسی آزمایش دیدهای ذخیره‌شده توزیع‌شده به کار گرفته شده است. معیار ارزیابی نیز مقایسه کاهش زمان پاسخ‌دهی به پرس‌وجوهای تحلیلی با تعداد پرس‌وجوهای متفاوت است.

زمان اجرا برای ۳ آزمون مختلف (با تعداد پرس‌وجوهای مختلف) در آزمون‌ها صورت پذیرفته است. آزمون‌ها شامل ۵، ۱۰، ۲۰ و ۶۰ تعداد پرس‌وجوی هم‌زمان پردازش تحلیلی برخط بوده است. این سه آزمون شامل اجرای پرس‌وجوهای پردازش تحلیلی برخط بدون استفاده از دیدهای ذخیره‌شده (Without MV)، با استفاده از دید ذخیره‌شده فقط در بخش پایگاه داده تحلیلی (Single MV) و با استفاده از دیدهای ذخیره‌شده در همه بخش‌ها (Multiple MV) است. زمان اجرا برای نمودار شکل ۷ با استفاده از الگوریتم افزودن نتایج به یکدیگر (Union) نشان داده شده است. نمودار شکل ۸، زمان اجرا را برای الگوریتم محاسبه توابع تجمعی (AF) نشان می‌دهد. نمودار شکل ۹ نیز مقایسه زمان اجرای دو الگوریتم افزودن نتایج به یکدیگر (Union) و محاسبه توابع تجمعی (AF) را با یکدیگر نشان داده است. همان طور که در نمودار شکل‌های ۷ و ۸ مشاهده می‌شود با افزایش تعداد پرس‌وجوها زمان اجرا کاهش پیدا می‌کند. رویکرد استفاده از MMV به دلیل ساخت دیدهای ذخیره‌شده در همه بخش‌ها، زمان اجرایی به مراتب کمتر نسبت به رویکردهای SMV و WMV دارد. رویکرد نیز SMV زمان پاسخ‌دهی کمتری نسبت به رویکرد WMV دارد. همان طور که از نتایج نمودار شکل‌های ۷ و ۸ مشاهده می‌شود به هر میزانی که تعداد دیدهای ذخیره‌شده افزایش پیدا کند، زمان اجرا کاهش پیدا می‌کند. مقایسه بین رویکردهای MMV و SMV نشان می‌دهد که ساخت دیدهای ذخیره‌شده در آخرین بخش (پایگاه داده تحلیلی) تأثیر زیادی در زمان اجرا دارد. بخش‌هایی که حاوی حجم بیشتری از داده هستند، تأثیر بیشتری بر زمان پاسخ‌دهی به پرس‌وجوها می‌گذارند و بنابراین استفاده از دیدهای ذخیره‌شده در آخرین بخش تأثیر چشم‌گیری بر روند کاهش زمان پاسخ‌دهی به پرس‌وجوها دارد. به همین دلیل تابع سود انتخاب دیدهای ذخیره‌شده، تأثیر حجم بخش‌ها را در محاسبه مقدار نهایی تابع، مد نظر قرار داده است.

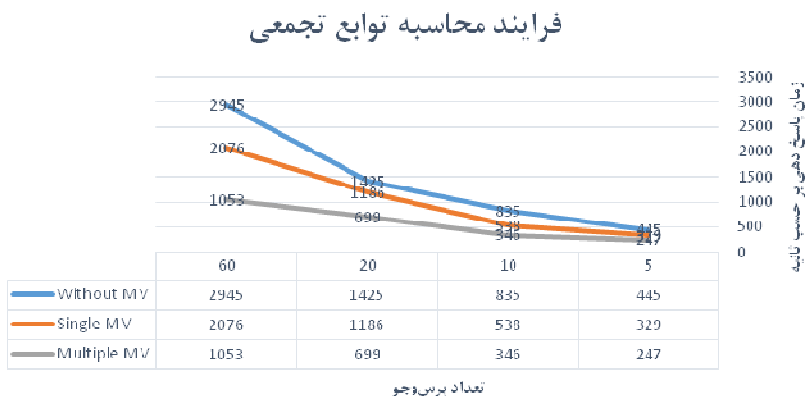
همان طور که در نمودار شکل ۹ مشاهده می‌شود، زمان اجرا در آزمون‌های مختلف در دو رویکرد افزودن نتایج به یکدیگر و محاسبه توابع تجمعی، متغیر است. دلیل این امر این است که تعداد رکوردهای برگشتی برای انجام عمل ادغام در پرس‌وجوهای پردازش تحلیلی برخط به مراتب کم بوده است. زمان واکشی داده (اجرای پرس‌وجو بر روی بخش‌ها) در هر دو رویکرد تقریباً برابر و به شرایط کنونی پایگاه داده بستگی دارد. نمودار شکل ۱۰ مقایسه بین زمان اجرای پرس‌وجوها با استفاده از دیدهای ذخیره‌شده و عدم استفاده از دیدهای ذخیره‌شده در سمت مشتری را نشان می‌دهد. همان طور که در این نمودار مشاهده می‌شود با افزایش تعداد پرس‌وجوها در رویکرد استفاده از دیدهای ذخیره‌شده در سمت مشتری، زمان پاسخ‌دهی به پرس‌وجو کاهش پیدا می‌کند. به هر میزانی که تعداد دیدهای ذخیره‌شده در سمت مشتری افزایش پیدا کند، زمان پاسخ‌دهی به پرس‌وجوها کم می‌شود. به ازای تعداد ۶۰ پرس‌وجوی

1. Star Schema Benchmark



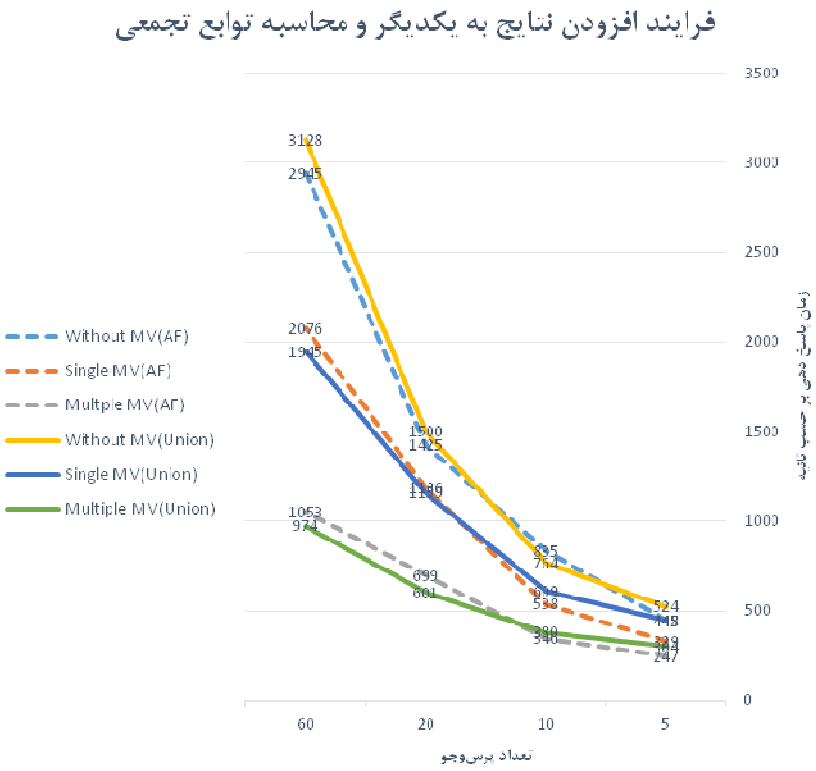
	60	20	10	5
Without MV	3128	1500	754	524
Single MV	1945	1159	609	448
Multiple MV	974	601	380	304

شکل ۷: ارزیابی زمان پاسخ‌دهی به پرس‌وجو برای رویکرد افزودن نتایج به یکدیگر.



	60	20	10	5
Without MV	2945	1425	835	445
Single MV	2076	1186	538	329
Multiple MV	1053	699	345	247

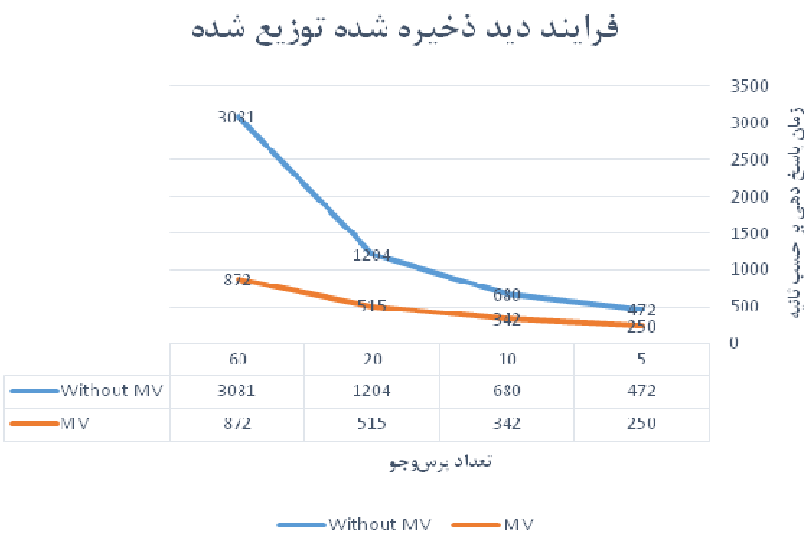
شکل ۸: ارزیابی زمان پاسخ‌دهی به پرس‌وجوها در رویکرد محاسبه توابع تجمعی.



شکل ۹: مقایسه زمان اجرای پاسخ‌دهی به پرس‌وجوها در هر دو رویکرد افزودن نتایج به یکدیگر و محاسبه توابع تجمعی.

پردازش تحلیلی برخط، زمان اجرا کاهش چشم‌گیری پیدا کرده است. کاهش زمان پاسخ‌دهی به پرس‌وجوها به دلیل وجود پرس‌وجوهای مشابه روی داده و بنابراین تعداد دستیابی به دیدهای ذخیره‌شده در سمت مشتری نیز افزایش پیدا کرده است.

برای آزمون فرض می‌کنیم که کاربر یک پرس‌وجوی پردازش تحلیلی برخط در رابطه با بیشترین فروش محصولات خاصی را در سال‌های گذشته تاکنون درخواست کرده است. در ساده‌ترین حالت، پرس‌وجو از طریق تنها بخش آخر پایگاه داده تحلیلی پاسخ داده می‌شود و دیگر



شکل ۱۰: زمان اجرای پرس‌وجو با استفاده از دیدهای ذخیره‌شده در سمت مشتری.

```
<Results>
  <result1>
    <row>
      <SalesAmount>8849840820.3822</SalesAmount>
      <DiscountAmount>951326529.8939</DiscountAmount>
      <quantitiy>529269</quantitiy>
      <numberOfOrder>535525</numberOfOrder>
      <Count>11049</Count>
      <customer_key>1</customer_key>
      <first_name>Louie</first_name>
      <last_name>Espinosa</last_name>
      <email_address>Chong76@example.com</email_address>
      <state>4</state>
      <town>3</town>
      <address>1023 E Glenwood Court</address>
    </row>
    <row>
        <SalesAmount>7131737784.0021</SalesAmount>
```

شکل ۱۱: نمونه‌ای از نتیجه پرس‌وجو در فرمت XML.

بخش‌های بی‌درنگ مورد بررسی قرار نمی‌گیرند. نتیجه نیز فقط به فرمت XML منتشر می‌شود که نمونه‌ای از آن در شکل ۱۱ مشاهده می‌شود.

چهار مسئله باید در مورد معماری ارائه‌شده مورد توجه قرار بگیرد. نخست، داده‌های اخیر مورد پرس‌وجو قرار نمی‌گیرند، داده‌های اخیر در این آزمایش شامل داده‌هایی با درجه تازگی دیروز و امروز هستند که در بخش‌های بی‌درنگ و پایگاه داده تحلیلی قرار دارند. دوم، درخواست‌کننده ممکن است کاربر و یا یک سامانه نرم‌افزاری باشد و اگر فرض کنیم نتیجه پرس‌وجو بایستی از طریق مرورگر به کاربر نمایش داده شود، پس یک خروجی به فرمت HTML مورد نیاز است. سوم این که اگر این پرس‌وجو پیش‌تر توسط درخواست‌کننده‌ای دیگر مورد پاسخ‌دهی قرار گرفته است می‌توان از نتایج آن استفاده کرد. مورد چهارم، پرس‌وجوی فعلی قابلیت استفاده از دید ذخیره‌شده‌ای که پیش‌تر ایجاد شده را دارد ولی به این دلیل که پرس‌وجوی فعلی شامل یک و یا چند فیلتر اضافه نسبت به پرس‌وجوی دید ذخیره‌شده است، قابلیت استفاده از نتایج دید قبلی برای پاسخ‌دهی به پرس‌وجوی فعلی امکان‌پذیر نیست.

به این دلایل برای مورد نخست از یک پایگاه داده تحلیلی تقریباً بی‌درنگ که شامل چندین بخش برای نگهداری داده‌ها با زمان‌های متفاوت تازگی است استفاده شده است. دو مدل موازی پاسخ‌دهی به پرس‌وجوهای پردازش تحلیلی برخط شامل افزودن نتایج به یکدیگر و محاسبه توابع تجمعی به منظور تسریع در زمان پاسخ‌دهی به پرس‌وجوها و بالابردن قابلیت انعطاف در نحوه یکپارچه‌سازی نتایج استفاده شده است.

مورد دوم، پردازه XSLT مسئول تبدیل تگ‌های XML به تگ‌های مناسب فرمت خروجی است و از این رو تولید خروجی توسط پردازه XSLT انجام می‌پذیرد. برای مورد سوم از دیدهای ذخیره‌شده در سمت مشتری استفاده شده است. از آنجایی که فضای ذخیره‌سازی سرور برای دیدهای ذخیره‌شده محدود است و همچنین زمان اعتبار دیدهای ذخیره‌شده نیز باید در نظر گرفته شود، از دیدهای ذخیره‌شده در سمت مشتری که حاوی یک زمان انقضا هستند استفاده گردیده و فرمت دیدهای ذخیره‌شده XML در نظر گرفته شده است. با این رویکرد، مشکلی در نحوه ذخیره‌سازی، انجام پرس‌وجو، تبدیل به فرمت‌های دیگر و همچنین انتقال داده به وجود نمی‌آید. همان گونه که ذکر شد فرمت دیدهای ذخیره‌شده در سمت مشتری XML است، پس می‌توان از زبان XQuery برای برطرف نمودن مورد چهارم استفاده کرد. بدین صورت که قبل از ارسال نتیجه به مشتری دیگر، پرس‌وجوی مناسب بر روی دید ذخیره‌شده توسط XQuery اجرا و سپس نتیجه ارسال شود.

برای مجموع ۲۰ پرس‌وجوی در حال ورود از سمت ۲ مشتری به سرور، مجموع اقدامات زیر انجام می‌شود. فرض بر این است که سرور هیچ دید ذخیره‌شده‌ای را نگهداری نمی‌کند. سرور آن را با استفاده از منبع اعتبارسنجی موجود از طریق XML Schema پس از دریافت هر ورودی بررسی می‌کند. ورودی معماری می‌تواند شامل داده‌هایی برای ذخیره در پایگاه داده تحلیلی و یا پرس‌وجوی پردازش تحلیلی برخط به همراه فراداده مناسب آن باشد. با ورود اولین پرس‌وجو که در شکل ۱۲ نشان داده شده است، سرور ورودی را اعتبارسنجی و مقادیر را از آن استخراج می‌کند.

ورودی یک دستور که در شکل ۱۲ نشان داده شده است شامل یک پرس‌وجوی پردازش تحلیلی برخط، حالت پردازش پرس‌وجو (افزودن نتایج به یکدیگر و محاسبه توابع تجمعی)، نیازهای درجه تازگی داده و خروجی نتیجه است که به ترتیب در تگ‌های cmd، process، part و output درج شده‌اند. برای پاسخ‌گویی به پرس‌وجو، نخست سرور دید مناسب ذخیره‌شده در سمت مشتری را انتخاب می‌کند. در صورت عدم وجود دید ذخیره‌شده در سمت مشتری، پرس‌وجو را با استفاده از الگوریتم فرایند موازی پرس‌وجو پاسخ‌دهی می‌کند. در این حالت، پس از پاسخ‌دهی به پرس‌وجو، نتیجه به فرمت XML جهت ترکیب نتایج به دست آمده از بخش‌ها و ارسال آن به مشتری و یا تولید محتوا از طریق واسط XSLT تبدیل می‌شود. در مورد پرس‌وجوی ۱بخشی از این نتیجه در شکل ۱۳

```
<Commands count="20" type="Group">
	<cmd ID="3" group="4" ip="192.168.100.1">
		select dates.d_year,
		customer.c_region ,
		part.p_mfgr ,
		part.p_category ,
		supplier.s_region,
		supplier.s_nation,
		sum(cast(lo_revenue as bigint) - lo_supplycost) as sum,
		COUNT_BIG(*) AS Count

		from dbo.dim_DATES_Stage_0 as dates, dbo.fact_LINEORDER_Stage_0 as lineorder,
dbo.dim_CUSTOMER_Stage_0 as customer, dbo.dim_PART_Stage_0 as part, dbo.dim_SUPPLIER_Stage_0 as
supplier

		where lineorder.lo_orderdate = dates.d_datekey and
		dates.d_year in (1998, 1998) and
		lineorder.lo_custkey = customer.C_CUSTKEY and
		customer.c_region = 'AMERICA' and
		lineorder.lo_partkey = part.P_PARTKEY and
		part.p_mfgr in ('MFGR#2', 'MFGR#4') and
		part.p_category in ('MFGR#11', 'MFGR#12', 'MFGR#13', 'MFGR#14', 'MFGR#15',
'MFGR#21', 'MFGR#22', 'MFGR#23', 'MFGR#24', 'MFGR#25') and
		lineorder.lo_suppkey = supplier.S_SUPPKEY and
		supplier.s_region = 'AMERICA' and
		supplier.s_nation in ('ARGENTINA', 'BRAZIL', 'CANADA', 'PERU', 'UNITED STATES')

		group by d_year, c_region, p_mfgr, p_category, s_region, s_nation
	</cmd>
	<process>Union</process>
	<part>#2#3</part>
	<output>
		</html>
	</output>
  </Commands>
```

شکل ۱۲: نمونه‌ای از پرس‌وجوی کاربر.

result2

d_year	c_region	p_mfgr	p_category	s_region	s_nation	sum	Count
1998	AMERICA	MFGR#2	MFGR#21	AMERICA	CANADA	976787435	343
1998	AMERICA	MFGR#2	MFGR#23	AMERICA	BRAZIL	918795494	326
1998	AMERICA	MFGR#2	MFGR#24	AMERICA	UNITED STATES	1065952092	371
1998	AMERICA	MFGR#2	MFGR#22	AMERICA	PERU	987296441	339
1998	AMERICA	MFGR#2	MFGR#23	AMERICA	ARGENTINA	1066809740	363
1998	AMERICA	MFGR#2	MFGR#23	AMERICA	UNITED STATES	948619862	349
1998	AMERICA	MFGR#2	MFGR#25	AMERICA	CANADA	974025940	350
1998	AMERICA	MFGR#2	MFGR#21	AMERICA	PERU	968171224	328
1998	AMERICA	MFGR#2	MFGR#24	AMERICA	ARGENTINA	1028172461	371
1998	AMERICA	MFGR#2	MFGR#24	AMERICA	BRAZIL	971833733	354
1998	AMERICA	MFGR#2	MFGR#25	AMERICA	PERU	850935802	310
1998	AMERICA	MFGR#2	MFGR#23	AMERICA	CANADA	1062868419	367
1998	AMERICA	MFGR#2	MFGR#22	AMERICA	UNITED STATES	1059251842	367
1998	AMERICA	MFGR#2	MFGR#23	AMERICA	PERU	994833503	344

شکل ۱۳: بخشی از نتیجه به دست آمده از واسط XSLT.

نمایش داده شده است.

اگر نتیجه پرس‌وجوی درخواستی در یک مشتری وجود داشته باشد، سرور نتیجه را از طریق مشتری حاوی نتیجه پرس‌وجو به مشتری مصرف‌کننده ارسال می‌کند. اگر فرض کنیم که نتیجه ذخیره‌شده در مشتری به صورت مجموع فروش محصولات MFGR#۲ و MFGR#۳ در یک تاریخ مشخص است و پرس‌وجوی جدید شامل فقط مجموع فروش محصول MFGR#۲ در همان تاریخ است، مشتری با اعمال این شرط از طریق زبان XQuery نتیجه را دوباره محاسبه و به مشتری

مصرف‌کننده ارسال می‌کند. در صورتی که درخواست جدید، نیاز به یک فرمت خروجی مانند HTML داشته باشد و دید ذخیره‌شده برای پاسخ‌گویی به این پرس‌وجو نیز در دسترس باشد، نتیجه دید ذخیره‌شده به قسمت XSLT سرور مستقیماً ارسال می‌شود و سپس محتوای تولیدشده به وسیله XSLT به مشتری مصرف‌کننده ارسال می‌شود که نمونه خروجی آن مانند شکل ۱۳ است. هر نتیجه که در مشتری ذخیره می‌گردد شامل یک تاریخ انقضا است و به عنوان مثال، نتیجه پرس‌وجوی اول در مشتری درخواست‌کننده ذخیره می‌گردد. اگر تا ۲۰امین پرس‌وجو نیازی به این دید ذخیره‌شده نباشد، پس این دید عملاً کاربردی نخواهد داشت. با استفاده از زمان انقضای تعریف‌شده برای دیدهای ذخیره‌شده، این دید پس از اتمام زمان انقضا حذف می‌گردد. در هنگام ارسال نتیجه یک دید ذخیره‌شده به مشتری دیگر، دید ذخیره‌شده در مشتری ارسال‌کننده حذف می‌شود و بنابراین برای پاسخ‌دهی به این پرس‌وجو، نتیجه دید مشتری ۱ که به مشتری ۲ ارسال شد، از مشتری ۱ حذف شد. دید ارسال‌شده به مصرف‌کننده نیز با تاریخ انقضای قبلی ذخیره می‌گردد. تا اتمام پاسخ‌دهی به این ۲۰ پرس‌وجو، نتایج بین این دو مشتری در حال تبادل است و از این رو فضای ذخیره‌سازی مشتری‌ها نیز به خوبی به وسیله زمان انقضا و جایگزینی دیدهای ذخیره‌شده مدیریت می‌شود. ترافیک شبکه نیز به خوبی مدیریت می‌شود به این دلیل که نتیجه مورد نیاز مصرف‌کننده پس از انتقال به عنوان یک دید ذخیره‌شده نگهداری می‌شود، پس داده اضافی منتقل نمی‌شود. پس از اتمام پاسخ‌دهی به ۲۰ پرس‌وجو، مادامی که زمان انقضای دیدهای ذخیره‌شده تمام نشده باشد، با ورود هر پرس‌وجوی دیگر، این دیدهای ذخیره‌شده قابل استفاده هستند.

با توجه به نمودار شکل‌های ۷ و ۸، زمان اجرای پرس‌وجوهایی که از دیدهای ذخیره‌شده در هر دو رویکرد افزودن نتایج به یکدیگر و محاسبه توابع تجمعی استفاده کرده‌اند، کاهش پیدا کرده است. همان طور که انتظار می‌رفت، زمان پاسخ‌دهی به پرس‌وجوهایی که از دیدهای ذخیره‌شده در همه بخش‌ها استفاده کرده‌اند، نسبت به زمان پاسخ‌دهی به پرس‌وجوهایی که فقط از دیدهای ذخیره‌شده در بخش پایگاه داده تحلیلی استفاده کرده‌اند، بسیار کمتر است. زمان پاسخ‌دهی به پرس‌وجوها در هر دو رویکرد به ازای تعداد پرس‌وجوهای مختلف متغیر بوده است.

٦- نتیجه‌گیری

در این مقاله از رویکردهای تولید محتوا به منظور ارائه خروجی به فرمت مناسب برای استفاده کننده استفاده شده است. پاسخ‌گویی موازی به پرس‌وجوهای پردازش تحلیلی برخط به واسطه افزایش قدرت انعطاف و تسریع در پاسخ‌دهی به پرس‌وجوها به کار گرفته شده است. دیدهای ذخیره‌شده در سمت مشتری جهت کاهش زمان پاسخ‌دهی به پرس‌وجوها و همچنین داشتن فضای بیشتر برای ذخیره دیدهای ذخیره‌شده ایجاد شده است. مجموع استفاده از این رویکردها موجب شد که یک معماری پایگاه داده تحلیلی تقریباً بی‌درنگ با قابلیت استفاده از XML ساخته شود. نخست، معماری پایگاه داده تحلیلی تقریباً بی‌درنگ برای استفاده در این رویکردها پیشنهاد شد. سپس الگوریتم پاسخ‌گویی به پرس‌وجوها به صورت موازی از طریق افزودن نتایج به یکدیگر و محاسبه توابع تجمعی جهت یکپارچه‌سازی نتایج شرح داده شد. طبق آزمون انجام‌شده توسط ارزیاب SSB، استفاده از دیدهای ذخیره‌شده در سمت سرور در هر دو رویکرد پاسخ‌گویی به پرس‌وجوها باعث کاهش زمان پاسخ به پرس‌وجوهای پردازش تحلیلی برخط می‌شود. استفاده از دیدهای ذخیره‌شده در سمت مشتری نیز باعث بهبود زمان پاسخ‌دهی به پرس‌وجوها و استفاده از فضای ذخیره‌سازی می‌شود. در بررسی موردی، تعاملات بین مشتری‌ها با یکدیگر و مشتری‌ها با سرور از طریق ذخیره‌سازی و استفاده از دیدهای ذخیره‌شده توضیح داده شد. با قراردادن زمان انقضا برای دیدهای ذخیره‌شده، چنانچه دید ذخیره‌شده مورد استفاده قرار نگیرد، آن دید حذف و فضای کافی برای ساخت دیدهای جدید به وجود می‌آید. با انتقال نتیجه به مشتری مقصد، دید ذخیره‌شده در مشتری مبدأ حذف می‌گردید تا از تکرار و استفاده از فضای ذخیره‌سازی ممانعت به عمل آید.

به عنوان کار آتی، الگوریتم‌هایی برای انتخاب نودهایی که توانایی نگهداری دیدهای ذخیره‌شده را دارند اتخاذ می‌گردد. از این طریق تعداد مشتری‌های نگه‌دارنده دید کم شده و مدیریت و افزایش سرعت در انتقال نتایج به دیگر مشتری‌های نزدیک به یکدیگر بیشتر می‌شود.

مراجع

[1] Y. Zhu, Lei An, and S. Liu, "Data updating and query in real-time data warehouse system," in *Proc. of Int. Conf. on Computer Science and Software Engineering*, vol. 5, pp. 1295-1297, Dec. 2008.

[2] R. J. Santos and J. Bernardino, "Real-time data warehouse loading methodology," in *Proc. of ACM Int. Symp. on Database Engineering & Applications,* pp. 49-58, Sept. 2008.

[3] A. Cuzzocrea, N. Ferreira, and P. Furtado, "Enhancing traditional data warehousing architectures with real-time capabilities," in *Proc. of Int. Symp. on Methodologies for Intelligent Systems,* pp. 456-465, Jun. 2014.

[4] M. Obali, B. Dursun, Z. Erdem, and A. K. Görür, "A real time data warehouse approach for data processing," in *Proc. of Signal Processing and Communications Applications Conf., SIU'13*, 4 pp., Apr. 2013.

[5] J. Zuters, "Near real-time data warehousing with multi-stage trickle and flip," in *Proc. of Int. Conf. on Business Informatics Research,* . pp. 73-82, Oct. 2011.

[6] T. Winsemann, V. Koppen, and G. Saake, "A layered architecture for enterprise data warehouse systems," in *Proc. of Int.Conf. on Advanced Information Systems Engineering,* pp. 192-199, Jun. 2012.

[7] Y. Sharma, R. Nasri, and K. Askand, "Building a data warehousing infrastructure based on service oriented architecture," in *Proc. of IEEE Int. Conf. on Cloud Computing Technologies, Applications and Management, ICCCTAM'12*, pp. 82-87, Dec. 2012.

[8] V. Gonzalez-Castro, L. M. MacKinnon, and M. del Pilar Angeles, "An alternative data warehouse reference architectural configuration," in *Proc. of British National Conf. on Databases,* pp. 33-41, Jul. 2009.

[9] Y. Mao, *et al.*, "Dynamic mirror based real-time query contention solution for support big real-time data analysis," in *Proc. of 2nd Int. Conf. on Information Technology and Electronic Commerce, ICITEC'14*, pp. 229-233, Dec. 2014.

[10] W. Qu, V. Basavaraj, S., Shankar, and S. Dessloch, "Real-time snapshot maintenance with incremental ETL pipelines in data warehouses," in *Proc. of Int. Conf. on Big Data Analytics and Knowledge Discovery,* pp. 217-228, Sept. 2015.

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

[12] I. Hamdi, E. Bouazizi, and J. Feki, "Dynamic management of materialized views in real-time data warehouses," in *Proc. of 6th Int. Conf. on Soft Computing and Pattern Recognition, SoCPaR'14,* pp. 168-173, Aug. 2014.

[13] T. Jain, "Refreshing data warehouse in near real-time," *Int. J. of Computer Applications*, vol. 46, no. 18, pp. 24-28, May 2012.

[14] M. Asif Naeem, G. Dobbie, and G. Webber, "An event-based near real-time data integration architecture," in *Proc. of 12th Enterprise Distributed Object Computing Conf. Workshops,* pp. 401-404, Sept. 2008.

[15] S. Yi Chuan and X. Yao, "Research of real-time data warehouse storage strategy based on multi-level caches," *Physics Procedia*, vol. 25, pp. 2315-2321, Jan. 2012.

[16] R. J. Santos, J. Bernardino, and M. Vieira, "24/7 real-time data warehousing: a tool for continuous actionable knowledge," in *Proc. of 35th Annual Computer Software and Applications Conf.*, pp. 279-288, Jul. 2011.

[17] N. Ferreira, P. Martins, and P. Furtado, "Near real-time with traditional data warehouse architectures: factors and how-to," in

Proc. of ACM 17th Int. Database Engineering & Applications Symp., pp. 68-75, Oct. 2013.

[18] P. O'Neil, E. O'Neil, X. Chen, and S.Revilak, "The star schema benchmark and augmented fact table indexing," in *Proc. of Technology Conf. on Performance Evaluation and Benchmarking,* pp. 237-252, Aug. 2009.

[19] M. Nguyen Tho and A. Min Tjoa, "Zero-latency data warehousing for heterogeneous data sources and continuous data streams," in *Proc. of 5th Int. Conf. on Information Integration and Web-Based Applications Services*, pp. 55-64, Sept. 2003.

[20] L. Golab and T. Johnson, "Data stream warehousing," in *Proc. of IEEE 30th Int. Conf. on Data Engineering*, pp. 949-952, Mar./Apr. 2014.

[21] M. Gorawski and A. Gorawska, "Research on the stream ETL process," *Proc. of Int. Conf. Beyond Databases, Architectures and Structures,* pp. 61-71, Jan. 2014.

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

[23] M. K. Sohrabi and V. Ghods, "Materialized view selection for data warehouse using frequent itemset mining," *J. of Computers*, vol. 11, no. 1, pp. 140-148, Mar. 2016.

[24] A. Gosain, "Materialized cube selection using particle swarm optimization algorithm," *Procedia Computer Science*, vol. 79, pp. 2-7, Jan. 2016.

[25] T. Win, "Conversion of XML schema to data warehouse schema using automatic approach," *International J. of Computer Applications*, vol. 108, no. 15, pp. 12-18, Dec. 2014.

[26] Z. Ouaret, R. Chalal, and O. Boussaid, "An approach for generating an XML data warehouse schema using model transformation language," *Journal of Digital Information Management*, vol. 12, no. 6, pp. 407-420, Dec. 2014.

سیدمصطفی شفائی تحصیلات خود را در مقطع کاردانی کامپیوتر- نرم‌افزار در سال ۱۳۹۰ با کسب رتبه دوم در دانشگاه فنی و حرفه‌ای شهید جباریان، و کارشناسی مهندسی کامپیوتر- نرم‌افزار در سال ۱۳۹۲ با کسب رتبه اول در دانشگاه علم و فرهنگ به پایان رسانده است، و در سال ۱۳۹۵ کارشناسی ارشد خود در رشته مهندسی کامپیوتر- نرم‌افزار را از دانشگاه تربیت دبیر شهید رجایی اخذ کرده است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: پایگاه داده تحلیلی، هستان‌شناسی و هوش تجاری.

نگین دانشپور استادیار دانشکده مهندسی کامپیوتر دانشگاه تربیت دبیر شهید رجایی می‌باشد. نام‌برده تحصیلات خود را در مقطع کارشناسی مهندسی کامپیوتر- سخت‌افزار در سال ۱۳۷۸ با کسب رتبه اول در دانشگاه شهیدبهشتی، و کارشناسی ارشد مهندسی کامپیوتر- نرم‌افزار در سال ۱۳۸۱ در دانشگاه صنعتی امیرکبیر به پایان رسانده است، و در سال ۱۳۸۹ دکتری خود در رشته مهندسی کامپیوتر- نرم‌افزار را از دانشگاه صنعتی امیرکبیر اخذ کرده است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: پایگاه داده تحلیلی، سیستم‌های تصمیم‌یار، پیش پردازش داده‌ها، و داده- کاوی.

سیدمجید شفائی تحصیلات خود را در مقطع کاردانی کامپیوتر-نرم‌افزار در سال ۱۳۹۰ در دانشگاه فنی و حرفه‌ای خوارزمی ملایر، و کارشناسی مهندسی کامپیوتر- نرم‌افزار در سال ۱۳۹۲ در دانشگاه علم و فرهنگ به پایان رسانده است و در سال ۱۳۹۵ کارشناسی ارشد خود در رشته مهندسی کامپیوتر-نرم‌افزار را از دانشگاه آزاد اسلامی واحد تهران مرکزی اخذ کرده است. زمینه‌های تحقیقاتی مورد علاقه ایشان عبارتند از: پایگاه داده تحلیلی، هستان‌شناسی و هوش تجاری.