مدیریت زمان در پایگاههای داده‌ی رابطه‌ای

چگونگی طراحی، به روز رسانی و پرس‌وجوی داده‌ی تمپورال

Managing Time
in Relational Database

تام جانستون
راندال ویز

مترجم: مهرداد عباس پناه

معاونت فناوری اطلاعات و ارتباطات ناجا
۱۳۹۳

سرشناسه: جانستون، تام، ۱۹۴۴ - م.
Johnston, Tom
عنوان و نام‌پدیدآور: مدیریت زمان در پایگاههای داده‌ی رابطه‌ای: چگونگی طراحی، به‌روزرسانی و پرس و جوی داده‌ای تمپورال/ تام جانستون؛ ویز: مترجم مهرداد عباس‌پناه؛ [برای] نیروی انتظامی جمهوری اسلامی ایران، معاونت فناوری اطلاعات و ارتباطات، مرکز تحقیقات کاربردی.
مشخصات نشر: تهران، منشور سمیر، ۱۳۹۲
مشخصات ظاهری: ۵۷۶ ص.: مصور، جدول.
شابک: ۹۷۸-۶۰۰-۹۴۱۹۷-۵-۷
وضعیت فهرست نویسی: فیپا
یادداشت: عنوان اصلی: Managing time in relational databases : how to design, update and query temporal data , c2010.
موضوع: پایگاه‌های اطلاعاتی موقتی
موضوع: پایگاه‌های ا... رابطه‌ای
موضوع: پایگاه ها... اطلاعاتی... مدیریت
موضوع: زبان‌های ... کامپیوتری
شناسه افزوده: ویس، ر...
شناسه افزوده: ...eis, Randal
شناسه افزوده: عباس‌پناه، مهرداد، ۱۳... - مترجم
شناسه افزوده: نیروی انتظامی جمهوری اسلامی ایران. معاونت فناوری اطلاعات و ارتباطات. مرکز تحقیقات کاربردی
رده بندی دیویی: ۰۰۵/۷۵۳
رده بندی کنگره: ۱۳۹۲ ۱۵ج۲پ/QA۷۶/۹
شماره کتابشناسی ملی: ۳۴۱۴۵۰۵

چگونگی طراحی، به روزرسانی و پرس‌وجوی داده‌ی تمپورال

- **صاحب اثر :** معاونت فناوری اطلاعات و ارتباطات ناجا
- **مؤلفین :** مهرداد عباس پناه
- **ناظر :** بهزاد لک
- **ویراستاران:** ادریس جعفری و فاطمه مرتضوی
- **طراح و صفحه آرایی :** محمد مهدی شعبانی
- **شمارگان :** ۱۰۰۰ نسخه
- **نشر و چاپ:** انتشارات منشور سمیر
- **نوبت چاپ :** اول
- **تاریخ چاپ :** تابستان ۱۳۹۳
- **قیمت :** ۳۰۰.۰۰۰ ریال

فهرست مطالب

بخش ۱

- فصل ۱. تاریخچه مختصری از مدیریت داده‌های تمپورال
- فصل ۲. رده‌بندی روش‌های مدیریت داده‌های تمپورال

پیشگفتار

در طول زمان داده‌های مربوط به مشتریان، محصولات، حساب‌ها و غیره تغییر می‌کنند. اما در اغلب موارد داده‌هایی که ما از این موجودیت‌ها نگهداری می‌کنیم، فقط وضعیت جاری آن‌ها را توصیف می‌کند. به این دلیل که هر تغییری که در این موجودیت‌ها ایجاد می‌شود، داده‌های مربوطه بروز رسانی شده و داده‌های قبلی از بین می‌روند. اما همه‌ی این موجودیت‌ها تاریخچه ای دارند، و بسیاری از آن‌ها در آینده نیز وجود خواهند داشت. داده‌های مربوط به گذشته و آینده هر دو مهم هستند.

داده‌های تاریخی را معمولاً با صرف زمان و تلاش کافی، می‌توان بازسازی کرد. البته کسب و کار در دنیای امروز اهمیت زیادی به دسترسی به داده‌های تاریخی و داده‌های مربوط به آینده بدون تأخیر و صرف هزینه می‌دهد. امروزه، بیش از پیش در محیط‌های کسب و کار به دسترسی مستقیم و فوری به داده‌های غیر جاری، البته با همان سهولت و سرعت داده‌های جاری، توجه می‌شود.

جدول‌های متعارف[1] به توصیف حالت جاری اشیاء می‌پردازند. اما جهت دسترسی به داده‌های توصیف‌کننده حالت قبلی موجودیت‌ها و چگونگی آن‌ها در آینده، تصور می‌شود که ترکیب داده‌های مربوط به گذشته، حال و آینده در همان جدول‌ها الزامی می‌باشد. به

---

[1]. Conventional table

جدول‌هایی که حالت گذشته، حال و آینده داده‌های توصیف کننده موجودیت‌ها را توصیف می‌کنند، جدول‌های نسخه بندی شده[1] گفته می‌شود.

جدول‌های نسخه بندی شده، یکی از دو نوع جدول‌های یونی تمپورال[2] هستند. در این کتاب به کاربرد جدول‌های نسخه بندی که باعث کاهش هزینه‌ها و افزایش ارزش داده‌های تمپورال[3] می‌شود، می‌پردازیم. داده‌های تمپورال به داده‌هایی گفته می‌شود که حالت پدیده‌ها در گذشته، حال و گاهی اوقات آینده را توصیف می‌کنند. با ساده‌سازی طراحی، نگهداری و پرس‌وجوی داده‌های تمپورال، هزینه‌ها کاهش می‌یابند. همان‌طور که مشاهده خواهید کرد ارائه پاسخ‌های سریع‌تر و دقیق‌تر به پرس‌وجوهای دسترسی به داده‌های تمپورال،باعث بالا رفتن ارزش آن خواهد شد.

موضوع مهم دیگری که گاهی اوقات پیرامون داده‌ها مطرح است، تفسیر اشتباه داده‌ها می‌باشد. احتمال دارد داده‌های اشتباهی در خصوص وضعیت یک مشتری خاص ثبت شود. به طوری‌که، مثلاً، نشان دهد که یک مشتری VIP، فرد بد حسابی است. به محض پی بردن به چنین اشتباهی، سریعاً با به‌روز کردن رکوردهای اطلاعاتی مربوط به مشتری[4]، آن داده‌ها را اصلاح خواهیم کرد.

این عمل نه تنها اشتباه را اصلاح می‌کند بلکه آن را نیز می‌پوشاند. ممیزان اغلب می‌توانند با استفاده از پشتیبان‌ها و لاگفایل‌ها[5]، داده‌های اشتباه را بازسازی کنند. اما در پایگاه داده هیچ نشانی از رخداد اشتباه، نوع و زمان اشتباه یا مدت زمان عدم کشف آن برای نویسنده پرس‌وجو معمولی باقی نمی‌ماند.

خوشبختانه می‌توانیم کار بهتری هم انجام دهیم. به جای بازنویسی اشتباه، می‌توان سابقه اولیه مشتری و کپی اصلاح شده به همراه اطلاعاتی در خصوص زمان و مدت زمانی که تصور می‌شد سابقه اولیه درست بوده است، زمانی که اشتباه بودن آن مشخص شده است، و اقدامات انجام شده در راستای تصحیح آن را در همان جدول نگهداری کرد. ممیزان با ادامه ارائه دسترسی آسان و فوری همراه با قابلیت پرس‌وجوی مستقیم به داده‌هایی که درستی آن‌ها تأیید می‌شود، می‌توان همان سطح دسترسی را در داده‌هایی که تمپورال نیست در نظر گرفته می‌شدند و اکنون نادرستی آن‌ها ثابت شده است، در نظر گرفت.

---

[1] Versioned table
[2] uni- temporal table
[3] Temporal Data
[4] customer's record
[5] backups and logfiles

درباره عنوان این جدول اتفاق نظری وجود ندارد، لذا آن را جدول اظهار[1] می‌نامیم. مشاهده خواهید کرد که جدول‌های اظهار برای بازآفرینی گزارش‌ها و پرس‌وجوها در آینده ضروری می‌باشند. منظور از آینده، زمانی است که هدف ما بازیابی همه جانبه داده‌ها به همان صورت اولیه درج آن‌ها می‌باشد. جدول‌های اظهار دومین نوع از جدول‌های یونی تمپورال هستند. همان روش‌های مدیریتی داده‌ای که باعث کاهش هزینه و افزایش ارزش داده‌های نسخه بندی شده می‌شود، هزینه داده‌های اظهارشده را پایین و ارزش آن‌ها را بالا می‌برد.

جدول‌ها هم وجود دارد که اظهارها[2] و نسخه‌ها را به گونه‌ای با هم ترکیب می‌کنند که هر سطر از این جدول‌ها دربرگیرنده یک نسخه و یک اظهار باشد. این جدول‌ها داده‌های زیر را در بر می‌گیرند: ۱- داده‌هایی که در حال حاضر تصور می‌شود بیانگر حالت گذشته، حال و آینده باشند، ۲- داده‌هایی که زمانی تصور می‌شد بیانگر حالت گذشته، حال و آینده باشند و ۳- داده‌هایی که احتمال دارد در آینده تصور شود بیانگر حالت گذشته حال و آینده باشند. به جدول‌هایی از این قبیل که اظهارها- نسخه‌ها داده‌هایی در خصوص گذشته، حال و آینده اشیاء و همچنین تصور گذشته، حال و آینده آن اشیاء را در بر می‌گیرند، جدول‌های بای تمپورال گفته می‌شود.

علی‌رغم چندین دهه تحقیق در زمینه داده‌های تمپورال و افزایش آگاهی از ارزش دسترسی بلادرنگ[3] به آن‌ها، کار زیادی برای کمک به متخصصان فناوری اطلاعات جهت مدیریت داده‌های تمپورال در پایگاه داده‌های جهان واقعی انجام نشده است. یکی از دلایل این است که الحاق زمانی زبان SQL هنوز تأیید نشده است، در صورتی که پیشنهاد افزودن مشخصات تمپورال به زبان SQL، پانزده سال پیش ارائه شد. عرضه‌کنندگان سامانه مدیریت پایگاه داده(DBMS)[4] به دلیل عدم وجود استانداردهای راهنما، در تجهیز محصولات خود به پشتیبانی تمپورال کند عمل کرده‌اند.

در ضمن متخصصان فناوری اطلاعات پشتیبانی داخلی از نسخه برداری[5] را گسترش داده در حالی که به قابلیت بای تمپورال آن‌ها هیچ توجهی نکرده‌اند. در بسیاری از موارد حتی

---

[1] assertion table

[2] (assertion) ، یک اظهار قسمتی ازSQLاست که نسبت به اعمال صحیح شروط اطمینان حاصل می‌کند و یا فعالیت‌هایی که بر روی اشیاء پایگاه داده، در حال انجام می باشد را خاتمه می‌بخشد. البته می‌تواند به معنای قفل کردن تمام جداول و یا حتی کل پایگاه داده باشد.

[3] real-time access

[4] DBMS (DataBase Management System)

[5] Versioning

نمی‌دانند که قابلیت بای تمپورال چیست. در اکثر موارد، کاربران تجاری آن‌ها از مزایا و قابلیت‌های بای تمپورال،اطلاعی ندارند و نمی‌دانند که باید چنین کارکردی[1] را درخواست از بین کسانی که حداقل عبارت بای تمپورال به گوش آن‌ها آشنا است و کسانی که بای‌تمپورال را برایشان توضیح دادیم، دو پاسخ مشترک دریافت کردیم. یکی اینکه، آقای رالف کیمبل[2] با ارائه سه نوع از ابعاد در حال تغییر خود، این مسئله را مدت‌ها پیش حل کرده است. دوم اینکه، با [illegible] جدول‌هایی که تمایل داریم داده‌های تمپورال را به آن‌ها اضافه می‌کنیم، [illegible] به تمام کاربردهای تمپورال دست یابیم.

[illegible] دو پاسخ مذکور اشتباه می‌باشد. ابعاد در حال تغییر کند به هیچ وجه از مدیریت داده‌های بای تمپورال از نسخه‌برداری پشتیبانی نمی‌کنند. اتر هر دو روش در مدیریت نسخه‌ها استفاده می‌شود، همان‌طور که در ادامه خواهیم دید هر دو به دلیل کمبود پشتیبانی گسترده از نسخه‌بندی خاص از Asserted versioning با شکست مواجه می‌شوند.

اهداف کتاب

دسترسی بی وقفه به داده‌های تمپورال

توصیف نحوه مدیریت داده‌های بای تمپورال در پایگاه داده‌های رابطه‌ای یکی از اهداف این کتاب است. این مدیریت باید به گونه‌ای باشد که دسترسی به این داده‌ها و داده‌های جاری به صورت بی‌وقفه امکان‌پذیر باشد. منظور از «بی‌وقفه» این است که: الف)تراکنش‌های مربوطه به قدری ساده باشد تا هر کس که می‌تواند تراکنش‌های جدول‌های متعارف را بنویسد، آن‌ها را هم بتواند بنویسد. ب) پرس‌وجوهای مربوطه به قدری ساده باشد تا هر کس که می‌تواند پرس‌وجوهای جدول‌های متعارف را بنویسد، آن‌ها را هم بتواند بنویسد. ج) کارایی تراکنش‌ها و پرس‌وجوهای مربوط به این گونه داده‌ها با اندازه کارایی تراکنش‌ها و پرس‌وجوهایی که صرفاً داده‌های متعارف را هدف قرار می‌دهند، باشد.

---

[1] functionality

[2] Ralph Kimball

[3] هر دو صورت داده‌های تمپورال را در پایگاه داده‌های غیررابطه ای نیز می‌توان اجرا کرد. به همین خاطر، با یک سری از فایل‌های یک سطحی نیز قابل اجرا هستند. در واقع دلیل استفاده از ادبیات پایگاه داده رابطه ای این است که همه گیر شدن فن آوری پایگاه داده‌ی رابطه ای این واژه را به زبان مشترک بازارهای IT تبدیل کرده‌است.

- پنهان‌سازی[1] ساختارهای داده و پردازش‌های داده‌های تمپورال

هدف دوم کتاب توصیف چگونگی پنهان‌سازی پیچیدگی‌های مدیریتی داده‌های یونیتمپورال و بای تمپورال می‌باشد. این پیچیدگی‌ها در هیچ جای دیگری به خوبی کتاب دکتر ریچارد سنادگراس، چاپ ۱۰ سال پیش، به تصویر کشیده نشده‌اند. وی یکی از کارشناسان زبده در این حوزه به شمار می‌رود.

دکتر سنادگراس در این کتاب با عنوان"توسعه‌ی کاربردهای پایگاه داده‌ای زمان گرا در SQL"[2] مثال‌های فراوانی از طرح‌های تمپورال و SQL را در چندین سامانه مدیریت پایگاه داده‌ای مختلف ارائه می‌دهد. وجود سامانه‌های مدیریت پایگاه داده‌ای مختلف در راه‌اندازی یونی‌تمپورال و بای تمپورال و به ویژه اعمال محدودیت‌های لازم در هنگام به وجود آوردن و نگهداری داده‌ها ضروری می‌باشند. بسیاری از این مثال‌های SQL در چندین خط نوشته شده و بسیار پیچیده هستند.

این دسته از کدها از آن دسته کدهایی نیستند که هر بار و با ایجاد هر برنامه جدید در پایگاه داده‌ای از نو نوشته می‌شوند بلکه کدی است که صرف‌نظر از برنامه‌هایی که از پایگاه داده استفاده می‌کنند، یکپارچگی آن‌را تضمین می‌کند. لذا تا وقتی که فروشندگان چنین کدی را بر محصولات سامانه مدیریت پایگاه داده‌ای خود ارائه نکنند، این کد باید به عنوان فصل مشترک میان کاربری‌ها و سامانه‌های مدیریت پایگاه داده‌ای مدیریت کننده پایگاه داده باقی بماند. این کد یک کد پایه است که در برنامه‌های کاربردی چندگانه بکار می‌رود در حالی که مستقل از برنامه‌های کاربردی که از آن استفاده خواهند کرد به وجود آمده و نگهداری شده است. به کد پایه ای که این نقش را ایفا می‌کند اغلب لایه دسترسی به داده[3] یا زیرچوب سرویس پایایی و پرس‌وجو[4] گفته می‌شود.

بنابراین به این نتیجه می‌رسیم که بهترین روش ارائه کارکرد تمپورال بر پایگاه داده‌های تحت DBMS/SQLهای امروزی، پنهان‌سازی این پیچیدگی‌ها است. versioning Asserted این کار را انجام می‌دهد.Asserted versioning در همین راستا، راه‌حل سازمانی را برای این مسئله مدیریت داده‌های تمپورال به وجود می‌آورد و در نتیجه از تعامل معنایی و فیزیکی داده‌های تمپورال در سرتاسر پایگاه‌های داده‌ی موجود در واحد سازمانی پشتیبانی می‌کند.

---

[1] Encapsulation

[2] DevelopingTime-Oriented Database Applications in SQL

۴ مورگان کوفمن، سان فرانسیسکو، ۲۰۰۰

[3] Data access lager

[4] Persistence and query service framework

Asserted versioning، پیچیدگی‌های مربوط به طراحی، نگهداری و پرس‌وجوی داده‌های یونی‌تمپورال وبای تمپورال را پنهان‌سازی می‌کند.پنهان‌سازی پیچیدگی‌های مربوط به طراحی به این معنی است که طراحان داده ملزم به طراحی ساختارهای داده‌های تمپورال نیستند. در عوض، مشخصات اعلانی[1] جایگزین کار طراحی می‌شود. این مشخصات علاوه بر مسائل دیگر تعیین می‌کنند که چه موجودیت‌های مدل داده منطقی باید هنگام ایجاد فیزیکی، به جدول‌های بای تمپورال تبدیل شوند؛ کلیدهای تجاری منحصر به شیء[2] نمایش داده شده در سطر یا سطرهایی وجود دارند و میان کدام جفت از جدول‌ها نوعی از یکپارچگی ارجاعی زمانی وجود دارد.

پنهان‌سازی پیچیدگی پشتیبانی و پرس‌وجوها[3] همان‌طور که پیش‌تر هم اشاره شد، به این معنی است که در به روز رسانی و حذف در جدول‌های بای تمپورال و پرس‌وجوی آن‌ها به قدری ساده است که بتواند آن‌ها را در جدول‌های غیرتمپورال بنویسد، بتواند آن‌ها را در جدول‌های زمانی Asserted versioningنیز بنویسد. پنهان‌سازی پیچیدگی پشتیبانی، در چارچوب Asserted versioning (AVF) که به توسعه آن مشغول هستیم، به وسیله یک API فراهم می‌شود. فراخوانی‌های[4] API این است جایگزین فراخوانی‌های مستقیمSQL بومی شود. البته به شرطی که الحاق تمپورال به SQL توسط کمیته استاندارد تایید و تأمین‌کنندگان آن را پیاده‌سازی نمایند.

AVF با عمل کردن در قالب یک چارچوب بالای صر داده‌های موجود در جدول‌های رابطه‌ای را حفظ نمی‌کند، بلکه اظهارها و نسخه‌ها را حفظ می‌کند و محدودیت‌های معنایی را میان سطرهای دارای شباهت تمپورال به یکپارچگی موجودیت و یکپارچگی ارجاعی، اعمال می‌کند.

Asserted versioning با عمل کردن در چارچوب خدمات پرس جو و جهت دسترسی به داده‌های جاری با استفاده از دسته‌ای از دیدها[6] پرس‌وجوها را پنهان می‌کند. این جلوه‌ها به تمامی پرس‌وجوهای مقابل داده‌های جاری این امکان را می‌دهند تا بدون انجام هیچ اصلاحی بی وقفه به فعالیت خود ادامه دهند. علاوه بر پنهان‌سازی، پرس‌وجوهایی که به دسترسی بی وقفه به هر ترکیب و یا دامنه‌ای از داده‌های گذشته، حال و آینده در یکی از دو بُعد تمپورال یا هر دو بعد تمپورال نیاز دارند پرس‌وجوهای لازم فراهم شده است.

---

[1] Declarative specifications
[2] object
[3] Maintenance encapsulation and query encapsulation
[4] calls
[5] temporal extensions to SQL
[6] view

در جدول‌های نسخه اظهارشده وجود داده‌های بای تمپورال معنایی مناسب تضمین شده است و لذا پرس‌وجو از آن جدول‌ها به طرز قابل توجهی ساده خواهد بود و صرفاً به افزودن یکی دو نقطه یا یک دوره از مستندهای تمپورال به پرس‌وجو دیگری از داده‌های جاری،نیاز می‌باشد.

- **زمینه‌سازی سازمانی**[1]

سومین هدف این کتاب تشریح نحوه اجرای مدیریت داده‌های تمپورال به عنوان یک راه‌حل سازمانی می‌باشد. البته، روش دیگر، اجرای تدریجی آن به صورت یک سری از راه‌حل‌های تاکتیکی است.برخی از راه‌حل های تاکتیکی که برای برنامه‌ها و پایگاه داده‌های متفاوت به صورت پروژه‌های نرم‌افزاری ایجاد می‌شوند، از یک سری مفاهیم تمپورال پشتیبانی می‌کنند که دیگر راه‌حل‌های تاکتیکی، آن پشتیبانی را انجام نمی‌دهند. در جاهایی که از معانی یکسان پشتیبانی می‌شود، ضرورتاً کدهای پشتیبان آن‌ها معمولاً متفاوت خواهد بود، حتی در برخی موارد این تفاوت بسیار زیاد است. در اکثر موارد، کُدی که از مفاهیم تمپورال پشتیبانی می‌کند در برنامه‌هایی جای گرفته‌اند که از معانی مختص یک کاربری خاص پشتیبانی می‌کنند. این معانی هیچ ارتباطی به تمپورال بودن ندارند. حتی پرس‌وجوهای یکپارچه‌ای که سعی بر پیوند داده‌های تمپورال در پایگاه داده‌های تاکتیکال شده با استفاده از راه‌حل‌های تاکتیکی مختلف دارند به اجبار با شکست مواجه خواهند شد. این پرس‌وجوها در صورت اتصال از طریق یک راه حل سازمانی یکپارچه[2] چندین برابر پیچیده‌تر خواهند شد.

Asserted versioning همان راه‌حل سازمانی مدنظر است. در این روش جدول‌های پایگاه داده‌ای که به عنوان یک جدول نسخه اظهار شده ایجاد می‌شوند و یا به آن تبدیل می‌شوند از تمامی معانی بای تمپورال پشتیبانی خواهند کرد. یک واحد یکسان از کُد یعنی چارچوب Asserted versioning و یا اجرای شما از این مفاهیم ازتمامی جدول‌های نسخه اظهارشده در سراسر پایگاه داده‌های پشتیبانی می‌کند.

این کُد از لحاظ فیزیکی از کد برنامه مجزا خواهد بود. در نتیجه، تمامی پیچیدگی‌های منطقی نگهداری داده‌های تمپورال از برنامه کاربردی حذف و در عوض با یک فراخوانی API[4] در AVF[3] از برنامه‌های کاربری انجام می‌شود. پرس‌وجوهای یکپارچه از داده‌های تمپورال، نیازی به در برگیری دستکاری‌های موردی ندارند. تنها هدف این دستکاری‌ها حل اختلافات

[1] Enterprise Contextualization
[2] unified enterprise solution
[3] AVF
[4] API Call

میان اجراهای مختلف از یک معنای تمپورالو یا مقیاس‌بندی. یک کاربری قدرتمند دریک جدول نسبت به جدول دیگر با قابلیت بیان ضعیف‌ترتغییرات را در بر نخواهد گرفت.

Asserted versioning به عنوان یک راه‌حل سازمانی یه منزله پلی به سوی آینده است. آینده‌ای که در آن قابلیت تمپورال توسط سامانه‌های مدیریت داده‌ای تجاری تامین می‌شود ومستقیماً از طریق تراکنش‌های SQL[1] و پرس‌وجوها قابل دسترسی خواهد بود. لیکن، اکنون Asserted versioning، را می‌توان با سرعت و بر اساس اولویت‌های انتخابی هر سازمانی اجرا کرد ، و این بدین معنی است که بخش تجاری با استفاده از آن خود را با حذف پشتیبانی مرحله‌ای از داده‌های تمپورال از کاربری‌ها و جایگزینی آن با عبارات توصیفی فراخوانی می‌کندتاAVFرا راه‌اندازی کند برای آینده آماده می‌کنند. در صورت انتخاب، دگردیسی نهایی حاصل از Asserted versioning درراه‌حل‌های اجرای تجاری که در ورای فراخوانی‌های API و جلوه‌ها پنهان شده است تقریبا برای همه روشن و آشکار خواهد بود.

اما، استراتژی‌های دیگری هم برای مهاجرت وجود دارد. یکی از این استراتژی‌ها این است که AVF در محل خود باقی بماند و اجازه دهد نسخه‌های آینده AVF کُد خود را متوقف کنند و در عوض یک پشتیبانی تمپورال به سبک جعبه از این سامانه‌های مدیریت داده‌ای آینده را هنگام فراهم آوردن پشتیبانی توسط عرضه کنندگان به کار گیرند. همان‌طور که مشاهده خواهید کرد به‌ویژه در فصل‌های ۱۲، ۱۳، ۱۶،Asserted versioning یک کارکرد برای تمپورال مهم را ارائه می‌دهد که هنوز موضوع بحث محققان علوم کامپیوتر نشده است.

یک کسب و کار در حالی که چارچوب Asserted versioning را در جای خود حفظ کرده است می‌تواند آن کارکرد مهم را هنگام جابجایی به کارکردهای تجاری ویژگی‌های خاص تمپورال پشتیبانی کند و این کار را بدون نیاز به اصلاح کد برنامه انجام دهد.

---

[1] با وجود این‌که هنوز استاندارد SQL پسوندهای تمپورال را جهت پوشش داده‌های بای تمپورال د ر برنمی‌گیرد، شرکت Oracle از چندین جنبه‌ی بای تمپورال در مد یر فضای‌کار[11] گرمی حمایت کرده‌است. در کتاب دیگری که بر وب‌سایت Elsevier و همچنن در آدرس Asserted Versioning@.com موجود است به بازبینی مدیر فضای کار و مقایسه آن با AV پرداخته‌ایم.

1 Workspace- Manager

درونی‌سازی دسته‌های داده‌ای Pipeline[1]

هدف آخر کتاب توصیف نحوه انتقال تراکنش‌های به تعلیق در آمده [2]به جدول‌های تولید هدف این تراکنش‌ها، و نحوه حفظ تراکنش‌های نوشته شده در همین جدول‌ها می‌باشد. تراکنش‌های به تعلیق در آمده به عبارات درج، به روز رسانی و حذفی گفته می‌شود که نوشته شده ولی هنوز به برنامه کاربردی که با پایگاه داده تولیدی در ارتباط است، ارسال نشده است. گاهی اوقات فایل‌های تراکنش یکجا [3]در خارج از پایگاه داده هدف گردآوری شده اند. اغلب، فایل‌های تراکنش یکجا در جدول‌های تراکنش یکجا گردآوری شده اند. تراکنش‌های نوشته شده به روی داده‌های در شرف درج، و تصاویر اولیه داده‌های در شرف حذف و یا به روزرسانی گفته می‌شود.

با اقتباس استعاره [4] در بسیاری از کاربری‌های لجستیک، تراکنش‌های به تعلیق در آمده را به عنوان تراکنش‌هایی که در نقاط مختلفی از Pipelineهای درونی وجود دارند در نظر می‌گیریم و تراکنش‌های [illegible] را داده‌هایی می‌دانیم که به نوعی لاگ فایل تخصیص یافته‌اند و همراه با Pipeline [illegible] در این مسیر حرکت می‌کنند. لذا اگر بتوانیم تراکنش‌های به تعلیق در آمده را [illegible] هدف آنها پیاده‌سازی کنیم و تراکنش‌های نوشته شده را در همان جدول‌ها نگه داریم، با [illegible] این استعاره دسته‌های داده‌ای Pipeline را درونی‌سازی کرده‌ایم.

داده‌های تولید، علاوه بر جدول‌های تولید، فایل‌های تراکنش یکجا به روزرسانی کننده آن‌ها و لاگ فایل‌ها نگهداری کننده تاریخچه این به روزرسانی‌ها در دسته‌های داده‌ای دیگر نیز وجود دارد. در برخی از جدول‌های تولید، جدول‌های تاریخچه وجود دارد که در آن‌ها تمامی نسخه‌های گذشته داده‌های جدول‌های تولید نگهداری می‌شود. گاهی اوقات یک گروه از سطرها در یک یا چند جدول تولید قفل شده و سپس به مکان فیزیکی دیگری کپی می‌شوند. داده‌ها پس از انجام کارهای مربوط به این محدوده مرحله بندی، به نقطه اولیه خود برگردانده می‌شوند و بر روی کپی‌های قفل شده اولیه آن داده‌ها قرار می‌گیرند.

در دنیای امروز مدیریت داده‌های فناوری اطلاعات، حجم زیادی از بودجه عملیاتی صرف مدیریت این دسته‌های داده‌ای فیزیکی چندگانه[5]می‌شود که داده‌های تولید در سرتاسر آن

---

[1]Internalization of Pipeline Datasets
[2]Pending transactions
[3]Batch transaction
[4]Borrowing a metaphor common
[5]Multiplephysical datasets

گسترده شده‌اند. به عبارتی، این کل کار عملیات فناوری اطلاعات[1] می‌باشد. سپس برنامه عملیات فناوری اطلاعات و سیستم‌های مدیریت جریان کار مختلف سعی می‌کنند که به روزرسانی‌ها را با این دسته‌های داده‌ای پراکنده هماهنگ کنند تا این به روزرسانی‌ها در یک توالی صحیح رخ دهند و نتایج صحیحی به بار آید. ابزارهای دیگری که یک‌دسته داده‌ی یکپارچه، متوالی و هماهنگ را در سرتاسر سیستم دسته داده‌ای و Pipelineها تضمین می‌کنند، رهاسازی سازمانه مدیریت پایگاه داده را در بر می‌گیرد که با پیش نیازها یا پس نیازها، مدیریت غیرهمزمان تراکنش و تغذیه غیر همزمان هماهنگ دستی از یک پایگاه داده به پایگاه داده دیگر مرتبط می‌باشند.

هزینه نگهداری این فرآیندها و محیط‌ها بالا و نیز منشاء خطا هستند. برای مثال، در جدول‌های تاریخچه و کارهای در حال انجام در محدوده‌های مرحله‌بندی بیرونی و یک سری از دسته‌های داده‌ای که به تعلیق در آمده، اعمال تغییر در یک واحد معنایی منفرد اطلاعات مانند نوع رویه در یک بیمه بیمه‌ای[2]، نیازمند اعمال در بسیاری از کپی‌های فیزیکی آن اطلاعات می‌باشد. ممکن است بخشی از این دسته‌های داده‌ای حتی با وجود کمک رهاناها و دیگر فرآیندهای اتوماتیک نشده باشد. این چشم پوشی بیشتر در محدوده‌های مرحله‌بندی بیرونی وجود دارد که شیشه بر آن‌ها وجود ندارند و به همین دلیل بخشی از فعالیت نگهداری برنامه‌ریزی شده منظم نیز می‌باشند. اگر هماهنگی غیرهم‌زمان باشد یعنی بخشی از واحد تجزیه نشدنی منفرد و مجزای کار نباشد، آنگاه با تأخیری مواجه هستیم که یک دوره زمانی است که در آن پایگاه داده و یا یک یا چند از پایگاه‌های داده‌ای در یک حالت متناقض قرار دارند. همچنین، در بازیافت خطا باید این به صورت دستی مد نظر قرار داده شوند و در حالی که به روز رسانی‌های متعدد دسته‌های داده‌ای به صورت دستی یا نیمه اتوماتیک انجام می‌گیرد، بازیافت این خطاها اتوماتیک نبوده و اغلب به صورت دستی انجام می‌شوند.

این نوع پراکندگی داده‌های تولید بر نویسندگان پرس‌وجوها تاثیر می‌گذارد. نویسندگان پرس‌وجو برای دستیابی به اطلاعاتی که جستجو می‌کنند باید از این دسته‌های داده‌ای پراکنده آگاه باشند، زیرا پذیرش این‌که تمامی داده‌هایی که از فیلتر کیفی‌سازی نویسندگان پرس‌وجوهای آن‌ها می‌گذرد در یک مکان جمع شده‌اند برای آن‌ها سخت است. در این دسته‌های داده‌ای، تفاوت‌هایی در مرحله چرخه حیات دسته‌های داده‌ای مختلف (مانند تراکنش‌های به‌تعلیق

[1] IT Operations
[2] historytables
[3] the policy type of aninsurance policy

درآمده، تراکنش‌های نوشته‌شده و نسخه‌های گذشته، حال، یا جاری و...) وجود دارد. در این دسته‌های داده‌ای، وجود سطحی از افزونگی اجتناب ناپذیر است. اغلب هیچ جدولی تمامی موارد یک نوع خاص (مانند تمامی رویه‌ها)، لازم در تامین نیازهای یک پرس‌وجو را در بر نمی‌گیرد.

به دنیایی از داده‌های شرکتی[1] غیر ضروری فکر کنید، دنیایی که در آن تمامی دسته‌های داده‌ای Pipeline یک جدول مقصد و یا مبدا یکسانی دارند. در آن فضا، نگهداری داده‌ها به معنای «بفرست و فراموش کن[2]» می‌باشد و فعالیتی نیست که در آن در ابتدا تراکنش‌های حفظ داده‌ها به وجود آمده و سپس باید در یک سری از نقاط بهبودی و استراحت واسط تراکنش جا شده تا اینکه بلاخره در جدول‌های هدف خود پیاده‌سازی شوند. در این فضا، یک پرس‌وجو هرگز خاطر نادیده گرفتن بعضی از داده‌های مرتبط در معرض خطر قرار نمی‌گیرد. در این فضا، چگونگی تولید تمامی داده‌های مرتبط با اشیا خود را دارند.

Asserted versioning به عنوان [illegible] و نرم‌افزار[3]

این کتاب به ارائه مفاهیمی می‌پردازد که براساس آن‌ها یک شرکت خاص می‌تواند این راه را انتخاب کند که چارچوب خاصی برای مدیریت داده‌های تمپورال خود بسازد. در ضمن این کتاب نرم‌افزاری را توصیف می‌کندکه همزمان با نگارش این کتاب در حال تولید آن می‌باشیم. نمونه اولیه این نرم‌افزار در آدرس Asserted versioning.com موجود است که کاربران علاقمند می‌توانند تراکنش‌های نگهداری و پرس‌وجوها [illegible] پایگاه داده‌ای بای تمپورال را ارائه دهند. این نرم افزار با عنوان Asserted versioningFramework یا AVFجدول‌های بای تمپورال مشابه مدل‌های به وجود آوردن چارچوب‌های پایگاه‌های [illegible] غیر تمپورال را از مدل‌های داده‌های منطقی متعارف به وجود می‌آورد. طراحان داده صریحاً مشخص کنند که کدام موجودیت‌ها در مدل منطقی باید به صورت جدول‌های بای تمپورال [illegible] تا به عنوان متا داده[4] پارامترهای دیگری را به وجود آورند که به AVF نشان دهد [illegible] جدول‌ها را مدیریت کند. هیچ طرح تمپورال خاصی برای انجام دادن وجود ندارد.

---

[1] corprrate
[2] submit it and forget it
[3] Asserted Versioning as Methodology andas Software
[4] metadata

از قابلیت‌های این نرم افزار در نمونه‌ای اخیر، تولید طرح‌ها و یک کُد می‌باشد. این کُد قواعد اعمال معانی داده‌های تمپورال را صرفاً از مدل‌های داده اروین[1] اجرا می‌کند و شدیداً به صفات تعریف شده توسط کاربر اروین و زبان نگارش ماکرو وابسته است. شرکت Computer Associates در طی ساخت این نرم‌افزار یکسری منابع فنی را در اختیار ما قرار داد و انتظار می‌رود که به هنگام عرضه آن در بازار نیز همچنان روابط نزدیک خود را حفظ کنیم.

جهت کسب اطلاعات بیشتر در خصوص Asserted versioning و نمونه اولیه این محصول به نشانی وبسایت Asserted versioning.com مراجعه کنید. همچنین، در طی چندین سمینار به تبیین این مفاهیم و نمایش اجرای کاربردی نرم‌افزار پرداخته شده‌است.

این سمینارها در وبسایت AssertedVersioning.com و نوشته‌های مورگان کافمن در www. elsevierdirect.com/companion/9780123750419. در دسترس می‌باشند.

[1] Erwin