گزارش پروژه

مریم علیپور ۹۶۱۲۰۳۷

فاز اول:

شمارهٔ استاندارد بین‌المللی کتاب یا **شابک** شماره‌ای متشکل از ۱۰ یا ۱۳ رقم که به کتاب‌های منتشرشده منسوب می‌شود و هر شماره منحصر به یک کتاب (یا یک نگارش یا نوع مشخص از آن کتاب) است، اما تجدید چاپ کتاب باعث تغییر شابک نمی‌شود. فلذا با تغییر شماره نسخه (ورژن) کتاب شابک آن هم عوض می شود. پس در جدول book هر کتاب با شابک یکتا یک رکورد می شود و اگر بطور مثال از آن کتاب ۱۰۰ جلد کتاب موجود داریم در جدول book_item ۱۰۰ رکورد ایجاد می شود که هر کدام یکی از این کتاب ها را نشان میدهد. در خصوص زبان کتاب توجه داشته باشید که هر ترجمه از کتاب خودش یک کتاب مجزا است و شابک خودش را دارد. در فایل database.sql کوئری های ساخت این اسکیما قرار گرفته است. برای سهولت کار ETL و عدم بررسی تمام رکوردهای دیتابیس و افزایش پرفورمنس، بر روی هر جدول ۳ عدد trigger از جنس insert, update و delete قرار گرفته که بعد از هر event این نوع event و pk آن رکورد و نام table به جدول temp_log به ترتیب insert می شود. نکته ی مهم این که همه ی constraint برای foreign key ها عملیات delete و update بصورت cascade است و در نتیجه بعد از حذف یا اپدیت شدن هر رکورد refrence رکوردهای متناظر هم حذف یا اپدیت می شوند و در جدول temp_log بوسیله ی trigger ها ثبت می شود.

فاز دوم:

بعد از طراحی انبار داده و مشخص کردن منطق ETLبرای این فرآیند یک کلاس پایتون نوشته شده است که اسکریپت کامنت گذاری شده. اما متاسفانه هنگام تست etl متوجه شدم دیتابیس هنگام حذف یا اپدیت refrence ها با cascade بجای اینکه اول رکورد های وابسته را در سایر جدول ها حذف کند خود refrence را حذف میکند و ترتیب ذخیر شدن logها را بهم میریزد! متاسفانه به دلیل محدویت زمان و انرژی نمی توانستم از اول دیتابیس و انبار داده و منطق etl را طراحی کنم و مجبور شدم از خود PostgreSQL محدویت های مربوط به foreign_key را دریافت کنم و تا یک لایه خودم وابستگی ها رو هندل کنم که کد را بسیار کثیف و ناخوانا کرد و احتمالا توش bug پیدا شه! بطور کلی برنامه بعد از وصل شدن به

دیتابیس و انبار داده، متادیتا و جدول temp_log را دریافت میکند و در مرحله پالایش یک لیست به ترتیب از کوئری هایی که باید در انبارداده اجرا شوند می سازد و در آخر اجرا میکند.

فاز سوم:

طرح کلی انبار داده مشابه دیتابیس اصلی است با این تفاوت که pk های جداول auto increment نیستند و همچنین بر delete و update ها دیگر cascde نیستند و جدول temp_log و trigger هم نداریم. متناظر با هر جدول یک جدول history قرار دارد که در آن رکوردهایی که اپدیت و دیلیت شدند با ویژگی event متمایز شدند و زمان وقوع هم ثبت می شود. در جدول های اصلی ستون created_at هم زمان اضافه شدن رکورد های جدید به دیتابیس را نشان میدهند که در نمودار umlنمایش ندادم (فایل اصلی حذف شد و نتونستم اضافه کنم) اما در کوئری های shema.sql آن می تونید مشاهده کنید این ستون را گذاشتم.