طراحی و ساخت پایگاه داده

مقدمه

پروژه‌ی پایانی درس پایگاه داده از سه قسمت اصلی تشکیل شده است: طراحی و ساخت یک پایگاه داده‌ی رابطه‌ای ساده، طراحی و ساخت ETL و در قسمت آخر طراحی و ساخت انبار داده (Data Warehouse). هر کدام از این قسمت‌ها به تفصیل در زیر شرح داده شده است. اما قبل از تعریف قسمت‌های پروژه، خوب است که دو تعریف زیر را ببنید:

- انبار داده (Data Warehouse) منبعی است که محل جمع‌آوری داده از منابع و فایل‌های دیگر می‌باشد. هدف از این جمع‌آوری استفاده از این دادگان در کنار یکدیگر در جهت ارایه تحلیل‌های تجاری برای سازمان است.
- فرآیند ETL مخفف Extract, Transform and Load است (استخراج، پالایش و بارگذاری). از ETL در زمان ساخت انبار داده استفاده می‌شود تا دادگان از منابع مختلف استخراج شده و پس از پالایش در انبار داده بارگذاری شوند. این عملیات می‌تواند توسط یکی از ابزارهای تجاری ساخته شده برای این منظور یا Stored Procedureها یا توابع و کوئری‌ها و یا اسکریپت‌هایی که در زبان‌های برنامه‌نویسی رایج نوشته می‌شوند، انجام گیرد.

طراحی و ساخت پایگاه داده‌ی رابطه‌ای

(۳۰ نمره)

در قسمت اول این پروژه از شما خواسته شده یک پایگاه داده‌ی رابطه‌ای با نیازمندی‌های زیر طراحی کنید. کوئری‌های مربوط به ساخت آن را با $PostgreSQL$ بنویسید و مدل داده‌ای آن را رسم کنید و ضمیمه‌ی آن قرار دهید. دقت کنید پایگاه داده طراحی شده باید در **فرم نرمال ۵ (5NF)** باشد.

- یک کتاب‌خانه دارای تعدادی کتاب است. ممکن است از بعضی از کتاب‌ها بیش از یک جلد در کتاب‌خانه موجود باشد. هر کتاب دارای شابک (شماره‌ی استاندارد بین‌المللی کتاب یا همان ISBN)، عنوان، توضیح، شماره‌ی نسخه، زمان انتشار و نام انتشارات است. یک یا چند نویسنده دارد. به یک یا

چند ژانر تعلق دارد. به یک یا چند زبان نوشته شده است. می‌تواند ترجمه شده و یک یا چند مترجم داشته باشد.

- هر یک از اعضای کتاب‌خانه دارای مشخصات فردی مانند نام، تاریخ تولد، تاریخ عضویت، شماره‌ی عضویت، اطلاعات تماس و آدرس است. بدیهی است هر یک از اعضا می‌تواند یک یا چند نسخه از یک یا چند کتاب را تا موعد مشخصی به امانت بگیرد و در این زمان این نسخه‌ها قابلیت به امانت گذاشته شدن مجدد را ندارند.

طراحی و ساخت ETL

(۴۰ نمره)

در قسمت دوم شما خط لوله‌ی انتقال داده (ETL pipeline) را می‌سازید. خط لوله انتقال داده می‌بایست با برقراری اتصال به دو پایگاه داده که هر دو از نوع $PostgreSQL$ هستند داده‌ها را از مبدأ به مقصد منتقل کند. به ازای هر جدول در پایگاه داده‌ی مبدأ می‌بایست جدولی هم‌نام با آن در پایگاه داده‌ی مقصد وجود داشته باشد که بعد از انجام عمل انتقال دقیقاً برابر با پایگاه داده‌ی مبدأ باشد.

انتقال داده در این خط لوله انتقال باید به صورت رکورد به رکورد انجام شود (مثلاً اجازه ندارید پایگاه داده مقصد را کلاً پاک کنید و پایگاه داده‌ی مبدأ را جای آن کپی کنید؛ چون شما ممکن است بخواهید بر روی رکوردها پالایشی (transform) اعمال کنید یا حجم داده به شما این اجازه را ندهد). برای انتقال داده‌ها از هر جدول پایگاه داده‌ی مبدأ به متناظر آن در مقصد باید به موارد زیر توجه کرد:

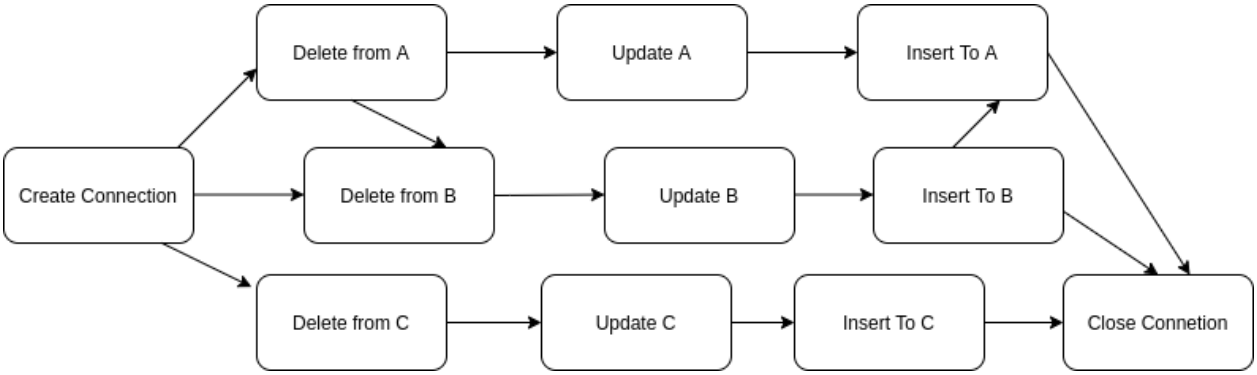
- ممکن است پس از آخرین اجرای خط لوله‌ی انتقال داده رکوردهایی از پایگاه داده‌ی مبدأ حذف شده باشند. پس می‌بایست در پایگاه داده‌ی مقصد نیز حذف شوند. (Delete)
- ممکن است پس از آخرین اجرای خط لوله‌ی انتقال داده رکوردهایی در پایگاه داده‌ی مبدأ دچار تغییر شده باشند. در این رکوردها primary key که می‌تواند شامل یک ستون یا ترکیبی از ستون‌ها باشد بدون تغییر باقی می‌ماند و یک یا چند عدد از دیگر ستون ها دچار تغییر شده‌اند. می‌بایست این رکوردها در پایگاه داده‌ی مقصد نیز به‌روزرسانی شوند. (Update)
- ممکن است پس از آخرین اجرای خط لوله‌ی انتقال داده رکوردهایی به پایگاه داده‌ی مبدأ اضافه شده باشند. پس می‌بایست در پایگاه داده‌ی مقصد نیز اضافه شوند. (Insert)

این رکوردها در هر یک از این شرایط باید شناسایی شوند. این که در هر بار اجرای خط لوله‌ی انتقال داده تمام رکوردهای پایگاه داده‌ی مقصد حذف شده و تمام رکوردهای پایگاه داده‌ی مبدأ به مقصد انتقال یابند در حجم کلان داده ممکن نیست و در این پروژه نیز پذیرفته نمی‌شود.

از طرفی باید به این نکته توجه کنید که ممکن است دو جدول به هم وابسته باشند. یعنی در یکی از آن‌ها (الف) foreign key ای به جدول دیگر (ب) وجود داشته باشد. بدیهی است اگر نیاز است رکوردی به جدول (الف) اضافه شود ابتدا می‌بایست رکورد متناظر با آن به جدول (ب) اضافه شده باشد و اگر نیاز است از جدول (ب) رکوردی حذف شود ابتدا می‌بایست رکورد متناظر با آن از جدول (الف) حذف شده باشد.

پس در نتیجه می‌بایست گرافی جهت‌دار و بدون دور (DAG) از گره‌های ریزدانه‌ی وظایف برای انجام به ترتیب آن‌ها ایجاد شود. برای مثال در پایگاه داده‌ی زیر جدول A به جدول B وابسته است و جدول C از هر دو آن‌ها مستقل است. اجرای درست ETL با این DAG امکان پذیر است:

(برای راهنمایی: می‌توانید از topological sort برای پیمایش این DAG استفاده کنید.)



توجه کنید که در مراحل مختلف پیاده‌سازی ETL برای به دست آوردن لیست جداول، برای به دست آوردن انواع Keyها و وابستگی بین آن‌ها و ... به هیچ عنوان نباید از Hard code استفاده کنید و می‌بایست با استفاده از کوئری مناسب به پایگاه داده آن‌ها را به دست آورید.

پیشنهاد می‌شود برای پیاده‌سازی ETL از زبانِ $python$ استفاده کنید. این یک پیشنهاد است، می‌توانید پیاده‌سازی‌های دیگری، در یکی از سه زبان C ،C++ و Java نیز داشته باشید!

شما می‌بایست در انتها source code خود به همراه داکیومنتی که برای توضیح آن تهیه کرده‌اید، تحویل دهید.

طراحی و ساخت انبار داده

(۳۰ نمره)

در این قسمت از پروژه روی یکی از نیازمندی‌های اصلی طراحی انبار داده که سفر به گذشته است، تمرکز می‌کنیم! ما دوست داریم بدانیم دیتابیس مبدأ در تاریخ معینی در گذشته در چه حالتی قرار داشته است (شامل چه رکوردهایی بوده است). پس می‌بایست:

- به ازای هر رکوردی که Insert می‌شود بدانیم دقیقاً در چه زمانی به انبار داده اضافه شده است. می‌توان در هر جدول یک ستون برای این کار اضافه کرد. (رکوردی که این ستون را دارد با رکورد متناظری که در دیتابیس مبدأ این ستون را ندارد برابر در نظر گرفته می‌شود). این یک پیشنهاد است، می‌توانید پیاده‌سازی‌های دیگری نیز داشته باشید!
- به ازای هر رکورد که در دیتابیس مبدأ Delete یا Update می‌شود باید این رکورد در انبار داده حفظ شود ولی از جدول هم‌نام متناظر به جدول دیگری که برای تاریخچه در نظر گرفته شده است انتقال یابد. باید مشخص باشد چه زمانی از جدول اصلی خارج شده است و علت آن چه بوده است. (حذف از جدول اصلی یا به روزرسانی آن) این یک پیشنهاد است، می‌توانید پیاده‌سازی‌های دیگری نیز داشته باشید!

این دو شرط برای سفر به گذشته کافی هستند. توضیح دهید که چگونه با استفاده از این دو شرط می‌توان در انبار داده به گذشته سفر کرد! سپس کوئری‌های مربوط به ساخت آن را با $PostgreSQL$ بنویسید و مدل داده‌ای آن را رسم کنید و ضمیمه‌ی آن قرار دهید.

در انتها شما باید یک فایل `zip` شامل ۳ پوشه‌ی Database و ETL و Data Warehouse آپلود کنید. پوشه‌های Database و Data Warehouse هر کدام شامل یک فایل `schema.sql` برای کوئری‌های مربوط به ساخت و یک فایل `data_model.pdf` برای رسم مدل داده‌ای هستند. پوشه‌ی Data Warehouse علاوه بر این‌ها شامل یک فایل `answer.pdf` نیز هست که توضیح شما برای سوال مطرح شده را در بر دارد. پوشه‌ی ETL شامل source code و یک فایل `pdf` برای توضیحات مربوط به ساختار کد است.