به نام خدا

دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

اصول طراحی پایگاه داده

پیاده سازی متدهای CRUD (فاز ۲)

استاد درس:

دکتر پوربهمن

مهلت ارسال پاسخ:

جمعه ۱۴۰۳/۱۰/۲۱ ساعت ۲۳:۵۹

مقدمه:

هدف از فاز دوم این پروژه، توسعه و تکمیل امکانات پایگاه داده است. در این مرحله، بر پیاده‌سازی عملیات اصلی مدیریت داده‌ها (CRUD) تمرکز می‌شود. همچنین، قابلیت‌های پیشرفته‌ای نظیر ایجاد توابع، viewها، triggerها، و اعمال محدودیت‌های سطح کاربری بر روی پایگاه داده اضافه خواهند شد. این فاز به گونه‌ای طراحی شده است که شما با فرآیند کامل توسعه یک پایگاه داده پیشرفته آشنا شوند و توانایی مدیریت و بهره‌برداری از آن در کاربردهای واقعی را کسب کنند.

تعریف کلی از پروژه:

در فاز دوم این پروژه، هدف اصلی پیاده‌سازی عملی و آزمایش قابلیت‌های پیشرفته پایگاه داده است. این قابلیت‌ها شامل موارد زیر هستند:

۱. پیاده‌سازی عملیات CRUD: نوشتن متدهای Insert، Select، Update و Delete در پایگاه داده با استفاده از زبان‌های برنامه‌نویسی مانند جاوا یا پایتون.
۲. اجرای نمونه کدهای CRUD: نوشتن کدهای عملیاتی برای چند مثال از عملیات Select، Insert، Update و Delete جهت اطمینان از عملکرد صحیح.
۳. کوئری‌های پیشرفته: اجرای کوئری‌هایی برای استخراج و تحلیل اطلاعات ذخیره‌شده در پایگاه داده.
٤. محدودیت‌های کاربری: تعریف محدودیت‌های دسترسی برای کاربران پایگاه داده، به‌گونه‌ای که برخی کاربران فقط مجاز به نوشتن یا خواندن باشند.
۵. ایجاد triggerها: تعریف رویدادهای خودکار که در واکنش به عملیات خاصی در پایگاه داده فعال می‌شوند.
٦. ایجاد توابع: نوشتن توابع سفارشی برای ساده‌سازی و بهبود عملیات تکراری در پایگاه داده.
۷. ایجاد viewها: طراحی ویوهایی که اطلاعات موردنیاز را با ساختار مشخص و استاندارد ارائه می‌دهند.

این فاز به گونه‌ای طراحی شده است که شما علاوه بر تسلط بر اصول اولیه پایگاه داده، توانایی مدیریت پیچیدگی‌های یک سیستم بانکی ساده را به دست آورید و درک عمیقی از کاربردهای عملی پایگاه داده پیدا کنند.

شرح پروژه:

قسمت اول

هدف این قسمت، پیاده‌سازی و آزمایش متدهای مدیریت داده‌ها (CRUD) در پایگاه داده است. این عملیات شامل Insert، Select، Update و Delete می‌شود. در این مرحله، تمرکز بر پیاده‌سازی متدهایی داینامیک و انعطاف‌پذیر است که امکان تعامل آسان و کارآمد با پایگاه داده را فراهم کنند.

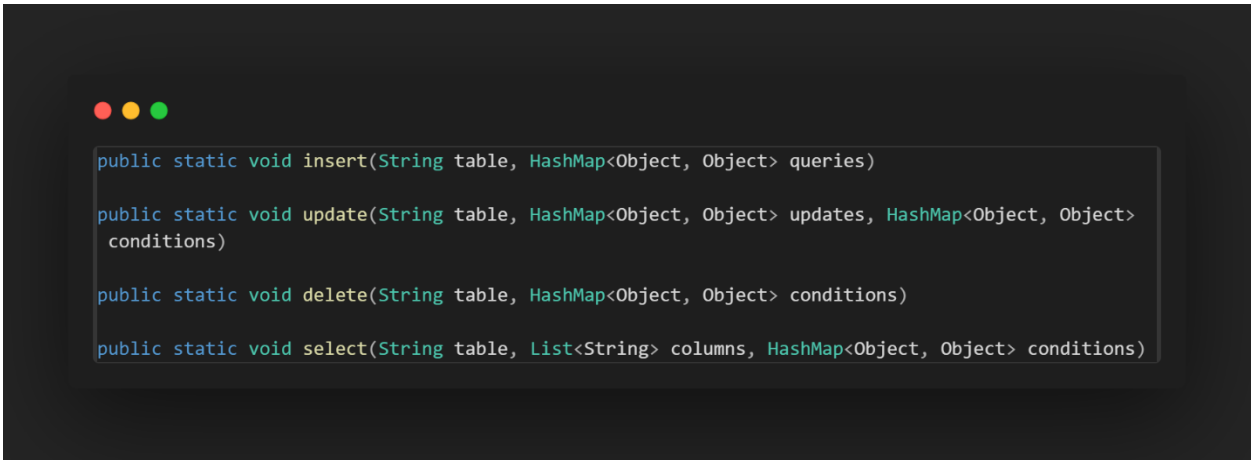
توجه داشته باشید برای اینکه بتوانید این کار را انجام دهید نیاز است package، library یا API مربوط به پایگاه‌داده و زبان برنامه نویسی‌ای که استفاده میکنید را نصب و import کنید.

مراحل انجام کار:

۱. **پیاده‌سازی متدهای CRUD:**
- توسعه متدهای Insert، Select، Update و Delete با استفاده از زبان برنامه‌نویسی انتخاب‌شده.
- تاکید بر داینامیک بودن متدها، به این معنا که متدها **باید پارامترهای متغیری** را بپذیرند تا بتوانند برای جداول و شرایط مختلف مورد استفاده قرار گیرند.
- اطمینان از ایمن بودن کوئری‌ها (مانند استفاده از Prepared Statements در SQL) برای جلوگیری از حملات SQL Injection.

۲. **آزمایش متدها با داده‌های آزمایشی:**
- اجرای چند نمونه عملیات CRUD روی جداول موجود در پایگاه داده.
- ثبت نتایج و تحلیل صحت عملکرد متدها از نظر کارایی و درستی داده‌ها.

توجه: این متدها باید به گونه‌ای طراحی شوند که قابلیت reuse و انطباق با تغییرات آینده را داشته باشند. Dynamic بودن متدها در این بخش ضروری است، زیرا سیستم باید برای عملیات متنوع و شرایط متفاوت آماده باشد.



```
public static void insert(String table, HashMap<Object, Object> queries)

public static void update(String table, HashMap<Object, Object> updates, HashMap<Object, Object>
 conditions)

public static void delete(String table, HashMap<Object, Object> conditions)

public static void select(String table, List<String> columns, HashMap<Object, Object> conditions)
```

۱- متدهایی که باید در کد (در اینجا جاوا) پیاده‌سازی شوند

قسمت دوم

در این قسمت از پروژه، نمونه کدهای عملیات CRUD برای جداول مختلف سیستم بانکی نوشته و اجرا می‌شود. هدف از این عملیات، اطمینان از عملکرد صحیح متدهای طراحی‌شده و بررسی جامعیت داده‌ها در پایگاه داده است.

عملیات Insert

یک رکورد جدید به پایگاه داده اضافه کنید که مشخصات آن شامل نام "David" و نام خانوادگی "Mills"، تاریخ تولد "۲ نوامبر ۱۹۸۵"، شماره تلفن "214555123"، ایمیل "davidmillss8@gmail.com"، و آدرس "12 Main St" باشد.

همچنین این شخص را به عنوان مشتری از نوع حقیقی ثبت کنید.

سپس برای همین شخص یک حساب بسازید که مشخصات آن شامل شماره حساب "1921072918"، نوع آن "سپرده"، مقدار موجودی اولیه حساب برابر ۵۰۰۰، تاریخ افتتاح آن را دلخواه و وضعیت حساب فعال باشد.

عملیات Update

آدرس همان شخص David Mills را به "456 Elm St" تغییر دهید. سپس وضعیت حساب او را غیرفعال تغییر دهید.

عملیات Delete

حساب بانکی David Mills را حذف کنید.

عملیات Select

چندین کاربر بانکی دیگر اضافه کنید و سپس نام و نام خانوادگی تمام این اشخاص را انتخاب کنید و نمایش دهید.

سپس چندین حساب بانکی دیگر اضافه کنید و تمامی حساب‌هایی که وضعیت آن‌ها فعال است را نمایش دهید.

این بخش از پروژه کمک می‌کند تا مطمئن شوند متدهای پیاده‌سازی شده برای عملیات CRUD به صورت دینامیک و کارا عمل می‌کنند و همچنین داده‌های وارد شده برای انجام کوئری‌ها و تحلیل‌های بعدی صحیح و قابل‌اعتماد هستند.

قسمت سوم

در این بخش از پروژه به اجرای کوئری‌های SQL می‌پردازیم. هدف این بخش، تمرین مهارت‌های کوئری‌نویسی است. شما موظف هستید بر اساس توضیحات داده شده کوئری‌های SQL مربوطه را بنویسند.

بخش اجباری:

۱- اضافه کردن یک فرد جدید در جدول Person با مشخصات دلخواه.
۲- افزودن یک حساب جدید برای یک فرد موجود در جدول Account.
۳- انتخاب و نمایش تمام تراکنش‌های مرتبط با یک حساب خاص از جدول Transaction.
۴- نمایش تمام وام‌های فعال از جدول Loan.
۵- نمایش تمام حساب‌هایی که موجودی آنها از یک مقدار مشخص بیشتر است.
۶- محاسبه موجودی کل حساب‌های هر فرد و نمایش نام و نام خانوادگی صاحب حساب به تفکیک نوع حساب.
۷- انتخاب نام، نام خانوادگی و مقدار وام کارمندانی که دارای وام‌های فعال هستند.
۸- نمایش نام، نام خانوادگی و تعداد حساب‌های مشتریانی که بیش از یک حساب دارند.

بخش امتیازی:

۱- نمایش مشتریانی که بیشترین تعداد وام‌های فعال را دارند.
۲- انتخاب ۵ وامی که کمترین تعداد اقساط پرداخت‌شده را داشته‌اند.
۳- نمایش نام، نام خانوادگی، شناسه وام و مقدار وام مشتریانی که اقساط وام‌های خود را به موقع پرداخت نکرده‌اند.
۴- نمایش نام، نام خانوادگی و موجودی ۵ مشتری که بالاترین موجودی را در حساب‌های خود دارند.

این بخش، شما را به چالش می‌کشد تا مهارت‌های خود را در نوشتن کوئری‌های مختلف بهبود دهید.

بدیهی است که باید رکوردهای مناسبی در دیتابیس خود وارد کرده باشید تا بتوانید خروجی ملموسی از این کوئری ها دریافت کنید.

کدهای این قسمت را در یک فایل با نام queries.sql ذخیره کنید و در zip نهایی خود قرار دهید.

قسمت چهارم

در این بخش، با هدف دسترسی ساده‌تر و خوانایی بهتر داده‌ها، چند View طراحی و به پایگاه داده اضافه خواهد شد. این ویوها به صورت مجازی ایجاد شده و به کاربران اجازه می‌دهند بدون نیاز به کوئری‌های پیچیده، اطلاعات موردنظر خود را مشاهده کنند.

customer_accounts view:

هدف از این ویو، نمایش اطلاعات کامل مشتریان به همراه حساب‌های مرتبط و موجودی هر حساب است. این ویو باید شامل ستون‌های زیر باشد:

- اطلاعات شخصی مشتری (نام، نام خانوادگی، شماره تماس)
- شماره حساب
- نوع حساب
- موجودی حساب

bank_transactions view:

این ویو اطلاعات مربوط به تراکنش‌ها را نمایش می‌دهد و بانک مربوطه را نیز مشخص می‌کند.
ستون‌های پیشنهادی برای این ویو شامل موارد زیر است:

- نام بانک
- شناسه تراکنش
- شماره حساب مبدأ
- شماره حساب مقصد
- مبلغ تراکنش
- تاریخ تراکنش

bank_member view:

این ویو برای نمایش اطلاعات کارکنان و مشتریان یک بانک طراحی شده است. ستون‌های این ویو عبارتند از:

- نام بانک
- مشخصات شخص (نام، نام خانوادگی، شناسه شخص)
- نقش فرد در بانک (کارمند یا مشتری)
- اطلاعات تماس (ایمیل، شماره تلفن)

این بخش به شما کمک می‌کند تا مفهوم View را بهتر درک کرده و از آن برای ساده‌سازی دسترسی به داده‌ها استفاده کنید.

شما باید با استفاده از **کوئری‌های SQL**، این ویوها را بر اساس ساختار پایگاه داده و داده‌های موجود ایجاد کنید.

کدهای این قسمت را در یک فایل با نام views.sql ذخیره کنید و در zip نهایی خود قرار دهید.

قسمت پنجم

در همه شرکت‌ها محدودیت‌هایی برای امنیت کاربران و حفظ درستی داده‌ها وجود دارد که دسترسی برخی از کارکنان با توجه به وظایفی که دارند محدود می‌شود و فقط یکی از عملیات‌ها را می‌توانند انجام دهند.

در این مرحله قصد داریم که یک کاربر جدید در DBMS درست کنیم و دسترسی‌های write آن به تمام جداول، view ها و... را از بین ببریم و صرفاً اجازه بدهیم که دسترسی read داشته باشد. این کاربر با نام کاربری Blazkowicz و رمز: William1939 باید بتواند وارد DBMS شود و صرفاً دسترسی read کردن از تمام جداول را داشته و نمی‌تواند write را انجام دهد.

کدهای این قسمت را در یک فایل با نام access.sql ذخیره کنید و در zip نهایی خود قرار دهید.

قسمت ششم

در این بخش، مجموعه‌ای از تریگرها و توابع برای مدیریت و بهبود عملکرد پایگاه داده طراحی می‌شوند. هدف اصلی این قسمت، تعریف فرآیندهای خودکار و توابع سفارشی برای ساده‌سازی عملیات پیچیده و تضمین صحت داده‌ها است.

تعریف تریگرها

۱. **ثبت تاریخ ایجاد حساب جدید:**

- این تریگر هنگام درج یک رکورد جدید در جدول Account فعال می‌شود.
- به‌طور خودکار تاریخ ایجاد حساب را در فیلد DateOpened ثبت می‌کند.
- هدف: جلوگیری از نیاز به ثبت دستی تاریخ ایجاد حساب و اطمینان از صحت داده‌ها.

۲. **جلوگیری از حذف مشتری با وام‌های فعال:**

- این تریگر هنگام تلاش برای حذف یک رکورد از جدول Customer فعال می‌شود.
- بررسی می‌کند که مشتری موردنظر هیچ وام فعالی در جدول Loan نداشته باشد. در غیر این صورت، عملیات حذف را لغو می‌کند.
- هدف: حفظ یکپارچگی داده‌ها و جلوگیری از ایجاد ناسازگاری در روابط پایگاه داده.

۳. **به‌روزرسانی موجودی حساب پس از انجام تراکنش:**

- این تریگر پس از درج یک تراکنش جدید در جدول Transaction فعال می‌شود.
- به‌طور خودکار موجودی حساب‌های مبدأ و مقصد را در جدول Account به‌روزرسانی می‌کند.
- هدف: اطمینان از هماهنگی بین داده‌های تراکنش‌ها و حساب‌ها.

۴. **بررسی موجودی کافی قبل از انجام تراکنش:**

- این تریگر قبل از درج یک رکورد جدید در جدول Transaction فعال می‌شود.
- بررسی می‌کند که موجودی حساب مبدأ برای انجام تراکنش کافی باشد. در صورت ناکافی بودن موجودی، عملیات را لغو می‌کند.

- هدف: جلوگیری از انجام تراکنش‌هایی که باعث ایجاد مانده منفی در حساب‌ها می‌شوند.

تعریف توابع

1. **محاسبه موجودی کل حساب‌های یک مشتری:**
   - این تابع با دریافت CustomerID، موجودی کل حساب‌های متعلق به آن مشتری را محاسبه و بازمی‌گرداند.
   - کاربرد: ارائه گزارش‌های مالی و تحلیل وضعیت مالی مشتریان.
2. **بررسی وضعیت یک وام خاص:**
   - این تابع با دریافت LoanID، وضعیت وام (فعال یا تسویه‌شده) را بر اساس تاریخ پایان و وضعیت پرداخت‌ها بازمی‌گرداند.
   - کاربرد: نظارت بر وضعیت وام‌ها و جلوگیری از ناهماهنگی داده‌ها.
3. **محاسبه تعداد وام‌های فعال یک مشتری:**
   - این تابع تعداد وام‌های فعال یک مشتری خاص را بر اساس CustomerID محاسبه می‌کند.
   - کاربرد: تحلیل رفتار اعتباری مشتریان و مدیریت ریسک.
4. **محاسبه مجموع پرداخت‌های انجام‌شده برای یک وام:**
   - این تابع مجموع پرداخت‌های انجام‌شده برای یک وام خاص را با دریافت LoanID محاسبه می‌کند.
   - کاربرد: بررسی عملکرد پرداخت و برنامه‌ریزی تسویه وام.
5. **دریافت نام مشتری بر اساس شناسه:**
   - این تابع با دریافت CustomerID، نام و نام خانوادگی مشتری را بازمی‌گرداند.
   - کاربرد: تسهیل عملیات مربوط به شناسایی مشتریان در گزارش‌ها یا پردازش داده‌ها.

این بخش شما را قادر می‌سازد تا با تعریف و استفاده از تریگرها و توابع صحت و یکپارچگی داده‌ها را تضمین کنید و فرآیندهای پیچیده را ساده‌تر کرده و از بازنویسی کدهای تکراری جلوگیری کنید.

کدهای این قسمت را در یک فایل با نام functions.sql ذخیره کنید و در zip نهایی خود قرار دهید.

نکات مربوط به تحویل:

- این پروژه در انتها، تحویل به صورت مجازی در google meet یا skype خواهد داشت و عدم تحویل پروژه به منظور از دست دادن تمام نمره‌ی آن خواهد بود.
- تمرین شما تحویل آنلاین خواهد داشت؛ بنابراین از استفاده از کدهای یکدیگر یا کدهای موجود در وب که قادر به توضیح دادن عملکرد آن‌ها نیستید، بپرهیزید.
- ابهامات خود را با تدریسیاران درس مطرح کنید تا آن‌ها در سریعترین زمان ممکن به شما پاسخ دهند. همینطور میتوانید مشکلات خود را با طراحان پروژه با آیدی HolyBardia@، roza_gp@ و ZeinabKarakani@ مطرح کنید.
- تاکید می‌شود که به هیچ وجه نباید از ORM ها استفاده کنید.

مواردی که باید ارسال شوند:

- فایل کدهای خود که در آن متدهای CRUD را ساخته‌اید
- فایل کدهای SQL
- یک فایل زیپ با نام studentID_PRJ_P2.zip که شامل تمام موارد بالا و گزارش شماست.