LibraryFunction

1- Library.CountAvailableBookCopies(@BookID INT)

یک تابع اسکالر برای محاسبه تعداد نسخه های موجود از یک کتاب مشخص است.

آیدی کتاب را به عنوان ورودی می گیرد و با تابع COUNT و کوئری روی جدول BookCopies تعداد نسخه ها را به عنوان خروجی بر می گرداند. وضعیت نسخه (CopyStatusID) مربوط به وضعیت "Available" باشد.(از جدول BookCopyStatuses)

2. Library.HasMemberOverdueBooks(@MemberID INT)

یک تابع اسکالر است که بررسی می کند آیا عضو کتابخانه با آیدی داده شده کتابی امانت گرفته دارد که قبل از تاریخ مشخص شده تحویل نداده است یا خیر. یعنی خروجی تابع TRUEیا FALSE است.

این تابع چک می کند که حساب شخص فعال باشد و تاریخ بازگشت کتاب NULL باشد و تاریخ تحویل قبل از تاریخ امروز باشد.

3- Library.GetMemberActiveLoanCount(@MemberID INT)

یک تابع اسکالر است که که تعداد کتاب‌هایی را که یک عضو فعال کتابخانه در حال حاضر امانت دارد، محاسبه می‌کند.

این تابع فقط اعضایی را در نظر می‌گیرد که وضعیت حسابشان Active باشد. در جدول Library.Loans جستجو می‌کند و تعداد رکوردهایی را می‌شمارد که ReturnDate آن‌ها NULLاست (یعنی هنوز برگردانده نشده اند) و مربوط به همان عضو باشند.

4- Library.IsBookCopyAvailable(@CopyID INT)

این تابع یک تابع اسکالر است که بررسی می‌کند آیا یک نسخه خاص از کتاب هم‌اکنون واقعا در دسترس است یا نه.

اگر نسخه دارای وضعیت Available باشد و در حال حاضر در امانت نباشد. (یعنی هیچ ردیفی در جدول Loans برای آن با ReturnDate IS NULL وجود نداشته باشد.) true یا همان یک را بر می گرداند.

Library PROCEDURE

1- Library.BorrowBook

این یک پراسیجر برای امانت دادن کتاب به عضو کتابخانه است.

از طریق آیدی ممبر ابتدا چک می کند که حساب آن فعال باشد و وجود داشته باشد. سپس از طریق ایدی نسخه کتاب چک می کند که این نسخه موجود است یا خیر. اگر شرط ها برقرار بود در جدول Loan یک رکورد مناسب اضافه می شود.

همچنین در جدول لاگ این رویداد ثبت می شود.

در این پراسیجر از تراکنش استفاده شده که تضمین کند تمام عملیات یا کامل انجام می‌شود یا هیچ‌کدام اجرا نمی‌شود.

2- Library.ReturnBook

این پراسیجر مسئول ثبت بازگشت یک نسخه خاص از کتاب در سیستم امانت کتابخانه است. اگر نسخه مورد نظر در حال حاضر امانت باشد، عملیات بازگشت را انجام می‌دهد، جریمه تأخیر را محاسبه می‌کند و همه چیز را در جدول لاگ ثبت می‌کند.

در این پراسیجر از تراکنش استفاده شده که تضمین کند تمام عملیات یا کامل انجام می‌شود یا هیچ‌کدام اجرا نمی‌شود.

اگر تاریخ فعلی از DueDat گذشته باشد، جریمه روزانه ضربدر تعداد روزهای تأخیر برای جریمه محاسبه می‌شود. و جدول Loan هم به روزرسانی می شود و فیلد ReturnDate را برابر تاریخ فعلی می گذارد.

3- Library.AddBookWithCopies

این پراسیجر برای افزودن یک کتاب جدید به سیستم کتابخانه همراه با چند نسخه طراحی شده است. با اجرای این پراسیجر، اطلاعات کتاب در جدول Books و نسخه‌های فیزیکی آن در جدول BookCopies ثبت می‌شود. همچنین، در صورت موفقیت، اطلاعات مربوطه در جدول لاگ سیستم نیز ثبت می‌شود.

اگر کتابی با ISBN واردشده قبلاً وجود داشته باشد، پیام خطا داده و تراکنش لغو می‌شود.

اگر وضعیت Available در جدول BookCopyStatuses وجود نداشته باشد، عملیات متوقف می‌شود.

4- Library. RecommendBooksToStudent

این پراسیجر، برای یک دانشجوی خاص با استفاده از (StudentID) حداکثر ۳ کتاب پیشنهادی بر اساس سابقه امانت‌گیری دانشجویان مشابه ارائه می‌دهد.
الگوریتم آن بر پایه Collaborative Filtering طراحی شده است.مراحل این الگوریتم :

1. گرفتن لیست کتاب های امانت گرفته شده توسط دانشجوی فعلی
2. پیدا کردن دانشجویانی که با او اشتراک کتاب داشته اند.(حداقل دو کتاب مشترک)
3. استخراج لیست کتابهایی که این دانشجویان گرفته اند ولی دانشجوی فعلی هنوز نگرفته است .
4. مرتب سازی این لیست بر اساس فراوانی امانت یا تعداد تکرار در بین کاربران مشابه
5. ارائه ۳ کتاب برتر به عنوان خروجی نهایی

5- Library.GenerateAllStudentBookRecommendations

این پروسیجر به‌صورت خودکار برای همه‌ی دانشجویان فعال در سیستم کتابخانه، کتاب‌هایی را که ممکن است به آن‌ها علاقه‌مند باشند توصیه می‌کند. این پیشنهادات بر اساس رفتار دانشجویان مشابه (یعنی کسانی که کتاب‌های مشابهی را قرض گرفته‌اند) ایجاد می‌شوند.

۱. پاک‌سازی توصیه‌های قبلی:
ابتدا همه‌ی رکوردهای جدول Library.BookRecommendations حذف می‌شوند تا فقط توصیه‌های به‌روز در سیستم باقی بمانند.

۲. ایجاد جدول موقت:
یک جدول موقت برای ذخیره‌ی خروجی‌های موقتی از پروسیجر Library.RecommendBooksToStudent ایجاد می‌شود.

۳. گردش روی دانشجویان فعال:
با استفاده از یک cursorروی همه‌ی دانشجویانی که وضعیت آن‌ها "Active" است، گردش انجام می‌شود.

۴. اجرای پروسیجر توصیه:
برای هر دانشجو:

- جدول موقت پاک‌سازی می‌شود.
- پروسیجر RecommendBooksToStudent اجرا شده و خروجی آن در جدول موقت ذخیره می‌شود.
- داده‌های جدول موقت به جدول دائمی BookRecommendations منتقل می‌شوند، همراه با شناسه دانشجو و تاریخ ایجاد توصیه.

۵. مدیریت خطا:
در صورتی که اجرای پروسیجر برای یک دانشجو با خطا مواجه شود، پیام خطا ثبت می‌شود و پردازش برای سایر دانشجویان ادامه می‌یابد.

Library TRIGGER

1. TR_Loans_AfterInsert_UpdateBookCopyStatusToBorrowed

هنگامی که یک نسخه ی کتاب توسط یک عضو امانت گرفته می‌شود (در جدول Loans درج می‌شود)، این تریگر وضعیت آن نسخه را در جدول BookCopies به Borrowed تغییر می‌دهد.

تریگر روی جدول Library.Loans تعریف شده است و با رویداد AFTER INSERT فعال می شود.یعنی هنگامی که رکوردی به جدول Loan اضافه می شود به طور اتوماتیک وضعیت این نسخه از کتاب را در جدول BookCopies به Borrowed تغییر می‌دهد.

2.TR_Loans_AfterUpdate_HandleReturnAndUpdateBookCopy Status

این تریگر به‌صورت خودکار پس از به‌روزرسانی جدول Library.Loans اجرا می‌شود و زمانی فعال می‌شود که مقدار ReturnDate برای یک کتاب ثبت شود، یعنی کتاب بازگردانده شود.

تغییر وضعیت نسخه کتاب به Available در جدول BookCopies در صورتی که کتاب واقعاً بازگردانده شده باشد.

تریگر فقط زمانی اجرا می‌شود که ستون ReturnDate در عملیات UPDATE تغییر کرده باشد. بررسی می‌شود که آیا نسخه کتابی که ReturnDate آن از NULL به مقدار غیر NULL تغییر یافته، وجود دارد یا نه.

3. TR_Books_LogChanges

این تریگر به صورت خودکار تمام تغییرات مربوط به جدول Library.Book شامل افزودن، ویرایش، و حذف کتاب‌ها را لاگ می‌کند.

با استفاده از inserted و deleted نوع عملیات تشخیص داده می‌شود.

4. TR_LibraryMembers_LogChanges

این تریگر به صورت خودکار پس از درج یا به‌روزرسانی رکوردی در جدول Library.LibraryMembers اجرا می‌شود و فعالیت مربوط به اعضای کتابخانه را در جدول لاگ ثبت می‌کند.

اگر AccountStatusID تغییر کند، تغییر وضعیت حساب از مقدار قبلی به جدید ثبت می‌شود.