**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 جدولBookCopyStatuses )

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 FALSEاست.

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 NULL است (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 Loanیک 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.

**TRIGGER Library**

1. **TR\_Loans\_AfterInsert\_UpdateBookCopyStatusToBorrowed**

هنگامی که یک نسخه‌ی 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 می‌شود.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.