**بسمه تعالی**



برنامه نویسی بانک اطلاعاتی SQL Server

**نویسنده : محمد حسین فخرآوری**

**به دام انداختن خطا**

یکی از مهمترین مباحث در مدیریت تراکنش در SQL Server در حقیقت مدیریت خطا در تراکنش ها می باشد

* انواع روش های به دام انداختن خطا (Error Trapping)

1. @@Error
2. TRY…CATCH

* مفهوم سطح شدت خطا (Error Severity Level)
* استفاده از تراکنش در TRY…CATCH و برعکس
* خطاهایی که به دام نمی افتند

SQL Serverبطور پیش فرض در صورت بروز خطا در داخل یک تراکنش، آن را بصورت اتوماتیک و ضمنی برای شما ROLLBACK نخواهد کرد. پس ناچاریم با به دام انداختن خطا ها کار Rollback و صدور پیغام مقتضی را خود به عهده بگیریم. بنابراین لازم است کمی با بحث Error Handling در T-SQL آشنا شوید.

**بلاک  Try – Catch**

ساختار بلاک Try -Catch بسیار ساده هست و از دو قسمت تشکیل شده , یکی که Statement های ما رو در بر میگیرد و دیگری بلاکی که در صورت بروز خطا آن را کنترل میکند . یک دستور Try – Catch ساده به شرح زیر خواهد بود .

* میتوانید جهت بررسی خطا به این دستور مراجعه کنید. SELECT \* FROM master.dbo.sysmessages

**مثال**

|  |
| --- |
| CREATE TABLE [dbo].[Test]  (  [Id] [INT] IDENTITY(1,1) NOT NULL PRIMARY KEY,  [Product] [NVARCHAR](50) NULL,  [Price] [INT] NULL,  [Color] [NVARCHAR](50) NULL  )  CREATE PROC B  AS  BEGIN TRY  BEGIN TRANSACTION  INSERT INTO [Test]([Product],[Price],[Color])VALUES('name','Num eroor','name2')  COMMIT  RETURN 1  END TRY  BEGIN CATCH  ROLLBACK  RETURN -1  END CATCH  DECLARE @Row INT  EXEC @Row = B  SELECT @Row |
| DECLARE @ID int  BEGIN TRAN  INSERT INTO [Test]([Product],[Price],[Color]) VALUES('name','88','name2')    SET @ID = SCOPE\_IDENTITY()  IF @ID <= 0 ROLLBACK  COMMIT    SELECT @ID |
| DECLARE @ID INT,@ID2 INT  BEGIN TRAN  INSERT INTO [Test]([Product],[Price],[Color]) VALUES('name','88','name2')  SELECT @ID = SCOPE\_IDENTITY()    IF @ID <= 0 ROLLBACK    INSERT INTO [Test]([Product],[Price],[Color]) VALUES('name','88','name2')  SELECT @ID2 = SCOPE\_IDENTITY()  IF @ID2 <= 0 ROLLBACK  COMMIT  SELECT @ID,@ID2 |
| BEGIN TRAN  SELECT \* FROM [Test] WHERE ID=55  IF @@TRANCOUNT <> 0  COMMIT  ELSE  ROLLBACK |
| CREATE PROC T  AS  DECLARE @ID INT  BEGIN TRAN  INSERT INTO [Test]([Product],[Price],[Color]) VALUES('name','88','name2')  SET @ID = SCOPE\_IDENTITY()    IF @ID <= 0 BEGIN ROLLBACK SELECT -1 END  COMMIT  SELECT 1    --EXEC T |

|  |
| --- |
| DECLARE @a INT  BEGIN TRY  SET @a = 1/0  END TRY  BEGIN CATCH  SELECT  ERROR\_NUMBER() AS ErrorNumber,  ERROR\_SEVERITY() AS ErrorSeverity,  ERROR\_STATE() AS ErrorState,  ERROR\_PROCEDURE() AS ErrorProcedure,  ERROR\_LINE() AS ErrorLine,  ERROR\_MESSAGE() AS ErrorMessage;  END CATCH |

**بکار گیری Try – Catch در نمایش خطا**

|  |
| --- |
| DECLARE @t TABLE(i int);  INSERT INTO @t VALUES ('string\_value');  SELECT ERROR\_MESSAGE() AS ErrorMessage  **خطا**  Msg 245, Level 16, State 1, Line 2  Conversion failed when converting the varchar value 'string\_value' to data type int. |

[

**راه حل**

|  |
| --- |
| DECLARE @t TABLE(i int);  BEGIN TRY  INSERT INTO @t VALUES ('string\_value');  END TRY  BEGIN CATCH  SELECT ERROR\_MESSAGE() AS ErrorMessage  END CATCH  **نتیجه**  Conversion failed when converting the varchar value 'string\_value' to data type int. |

**چاپ مسیج در صورت خطا**

|  |
| --- |
| IF OBJECT\_ID(N'Categories') IS null  RAISERROR ('IS null',10,2)  ELSE  RAISERROR ('IS Not null',10,2) |

**نکته**

|  |
| --- |
| SELECT 1 / 0  IF @@ERROR > 0 SELECT @@ERROR |

اولین @@error آخرین شماره خطا را نگه داشته است که عددی بزرگتر از صفر است.

پس شرط دستور IF صحیح است در نتیجه @@error انتخاب می شود، اما دومین @@error که بعد از بررسی دستور IF اجرا می شود بر خلاف تصور شماره خطا مربوط به آخرین دستور که شرط IF باشد را نگهداشته است که عدد صفر است.

به بیان دیگر دستور IF مقدار @@error را reset کرده است.

* بعد از اینکه در بلاک CATCH خطا مدیریت شد کنترل منتقل می شود به اولین عبارت T-SQL که در ادامه ی عبارت END CATCH قرار دارد. در بلاک TRY عبارات T-SQL که بعد از عبارتی که موجب بروز خطا شد قرار دارند اجرا نخواهند شد.
* اگر هیچ خطایی داخل بلاک TRY وجود نداشته باشد کنترل پاس داده می شود به عبارتی که بلافاصله بعد از دستور END CATCH وجود دارد.

**سطح شدت خطا (Error Severity Level)**

Error ها دارای پارامتری به نام Severity هستند که شدت خطا را تعیین می کند. خطاهایی با شدت 10 و کمتر از آن به عنوان هشدار (Warning) یا پیام های اطلاع راسانی در نظر گرفته می شوند و Error هایی با سطح شدت 11 و بالاتر به عنوان Error محسوب می شوند .

نکته ای که قصد بیان آن را داشتم این است که @@error شماره خطاهایی با شدت 10 و کمتر از آن را برنمیگرداند.