بسم الله الرحمن الرحيم

آموزشگاه تحلیل داده

تخصصی ترین مرکز برنامه نویسی و دیتابیس در ایران

آموزش اصول transactions در SQL SERVER

مدرس : مهندس افشین رفوآ

رمز فایل :tahlildadeh.com

مقدمه

تراکنش (transactions) به يك عمليات يا مجموعه اي از عملياتها ميگويند كه در يك مجموعه انجام شده و مسول انجام شدن عمليات ها استفاده هستند. به همين خاطر قوانيني هستند كه بايد آنها را رعايت كرد. همچنين راهكارهايي براي استفاده بهينه از transaction تعبيه شده است.

اين قواعد زير مخفف ACID تعريف ميشوند

١. Atomicity (A): چنانچه transaction شامل مجموعه اي از عمليات باشد همه آنها ميبايست كه اجرا شوند. در غيراين صورت , حتي اگر يك عمليات اجرا نشود بقيه آنها نيز اجرا نخواهند شد.

٢. Consistency (C) : عمليات ها بايد همسان باشند

٣. Isolation (I): زماني كه عمليات ها اجرا ميشوند , آنها ميبايست كه از يكديگر جدا باشند. اين به اين خاطر است كه از مخلوط شدن آنها در يك server يا يك database جلوگيري شود.

٤. Durability (D) : هنگام اجرا كردن عمليات ها , كامپيوتر تغييرات آنها را رصد ميكند. اگر اشكالي باعث توقف عمليات ها شده و كلمپيوتر restart شود بعد از restart شدن خودبه خود عمليات را ادامه ميدهد. بعد از اجرا شدن عمليات و اطمينان از بي نقص بودن بايد آن را save كرد و براي دفعات بعد به كار برد.

شروع يك transasction

قبل از ايجاد يك transactions , شما بياد عمليات را تعريف كنيد. براي مشخص كرد شروع يك transactions , قبل از اولين عمليات , كلمه ي BEGIN TRAN يا BEGIN TRANSACTION را به همراه فرمول زير تايپ كنيد:

```
BEGIN { TRAN | TRANSACTION }

  [ { transaction_name | @tran_name_variable }

   [ WITH MARK [ 'description' ] ]

  ]

[ ; ]
```

عبارت را با BEGIN TRAN يا BEGIN TRANSACTION شروع كرده و سپس ميتوانيد به دلخواه transaction_name را قرار دهيد. اگر از عبارات متني استفاده ميكنيد (char, nchar, varchar, or nvarchar) اسم transaction را برا اساس آنها انتخاب كرده ايد ميتوانيد transaction_name را حذف كرده و از اسم متغير استفاده كند. چنانچه مايل به شرح transaction در log file هستيد , كلمه ي WITH MARK را تايپ كرده و شرح را در فايل بنويسيد.

كد بين BEGIN TRAN يا BEGIN TRANSACTION قسمتي از transactionاست

پايان transaction

استفاده از transaction

بعد از تعريف عمليات , كه قسمتي از transaction است, موتور database ترتيب وار آنها را اجرا ميكند. شما بايد مكان پايان transaction ها رامشخص كنيد. براي يان كار BEGIN TRAN يا BEGIN TRANSACTION را تايپ كنيد :

```
BEGIN TRAN Name or BEGIN TRANSACTION Name

  Operations

COMMIT TRAN Name or COMMIT TRANSACTION Name
```

مثال :

```
BEGIN TRAN Name or BEGIN TRANSACTION Name
```

Operations

COMMIT TRAN Name or COMMIT TRANSACTION Name

Consider the following example:

```
USE Exercise;
Go
CREATETABLEAdministration.Employees
(
EmployeeNumbernvarchar(10),
EmployeeNamenvarchar(50),
DateHireddate,
HourlySalarymoney
);
GO
INSERTINTOAdministration.Employees
VALUES(N'593705',N'Frank Somah',N'20061004', 26.15),
      (N'720947',N'Paul Handsome',N'20000802', 36.05);
GO
INSERTINTOAdministration.Employees(EmployeeName,EmployeeNumber,DateHired)
VALUES(N'Clarice Simms',N'971403',N'20011112');
GO
BEGINTRANSACTIONAddEmployees
INSERTINTOAdministration.Employees
VALUES(N'595002',N'John Meah',N'20000212', 32.25);
GO
INSERTINTOAdministration.Employees
VALUES(N'928375',N'Chuck Stansil',N'20080628');
GO
INSERTINTOAdministration.Employees
VALUES(N'792764',N'Orlando Perez',N'20000616', 12.95);
GO
COMMITTRANSACTIONAddEmployees;
GO
INSERTINTOAdministration.Employees(EmployeeName,EmployeeNumber,
            HourlySalary,DateHired)
VALUES(N'Gina Palau',N'247903', 18.85,N'20080612');
GO
```

ان کد از database engine درخواست ایجاد یك table به نام employee را در Administration schema ایجاد میکند. بعد از تشکیل table , باید اول یك record و سپس ۲ record تشکیل دهد. سپس بیاد یك transaction شامل ۳ record را انجام دهد. بعد از آن transaction , وارد کردن دادها با اضافه کردن recordها ادامه پیدا کرده براي درك بهتر ما یك error را در کد transation قرار میدهیم. کد بالا شکل زیر را نتیجه خواهد داد :

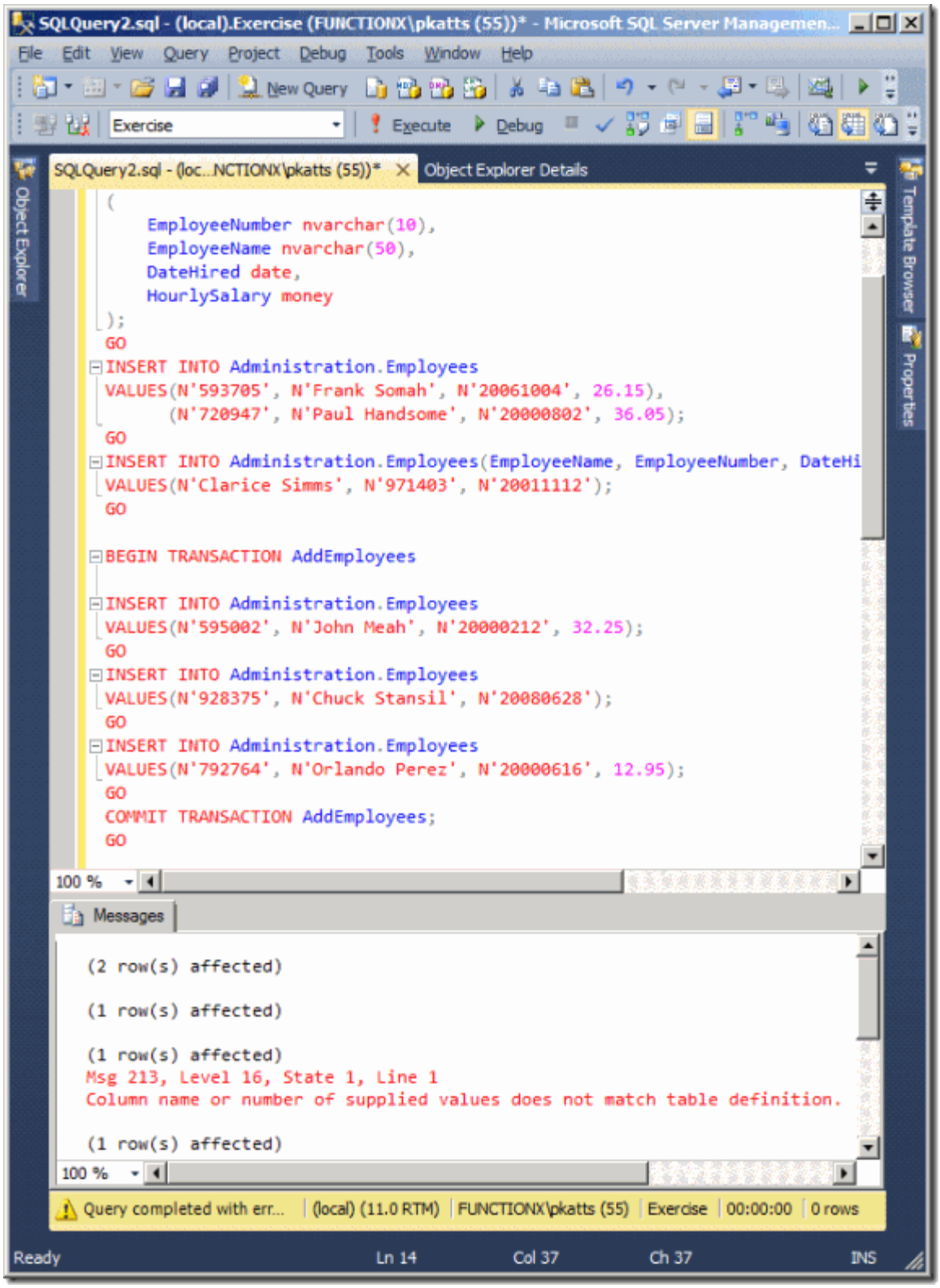
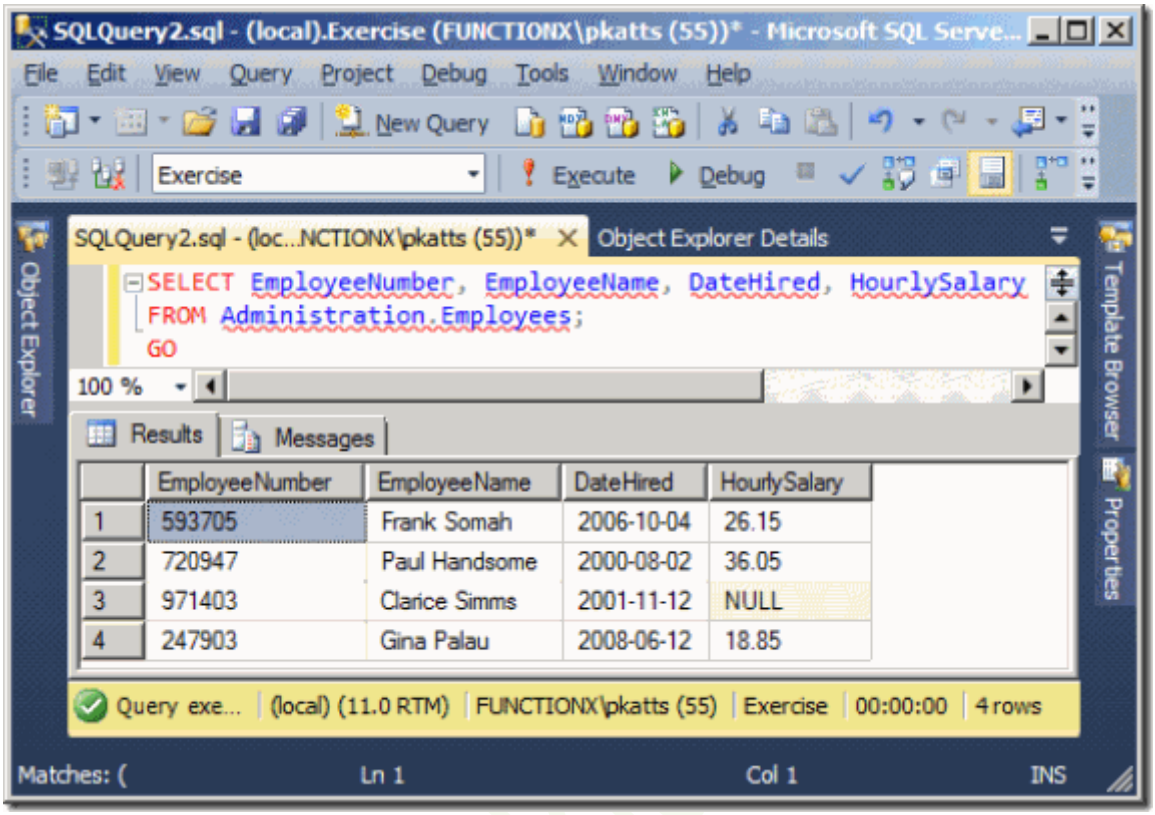


Table نهايي به شكل زير خواهد بود:



توجه کنید که کد transaction کامل نبوده و record تشکیل نمیشود

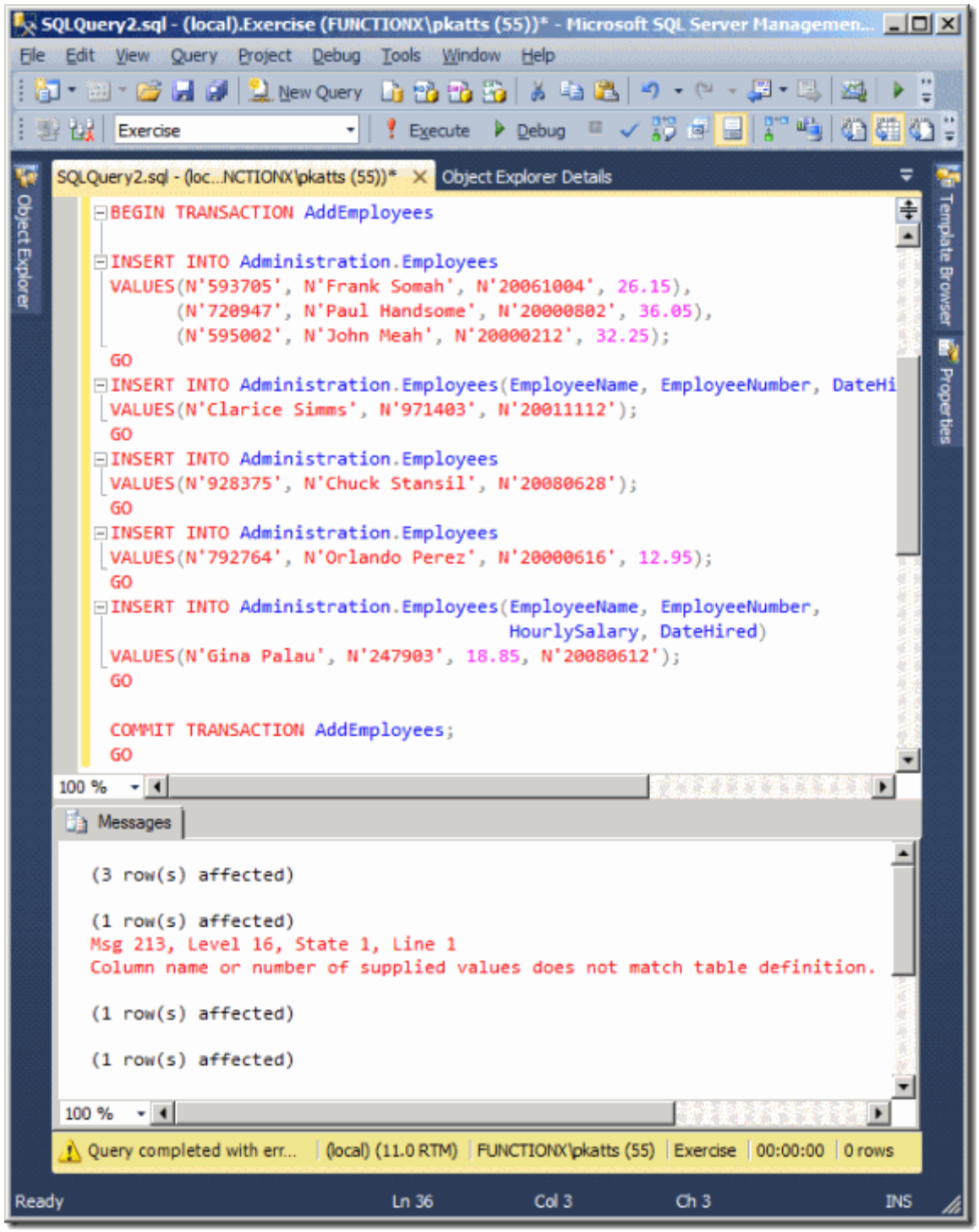
عقبگرد یك transasction

کد زیر را در نظر بگیرید

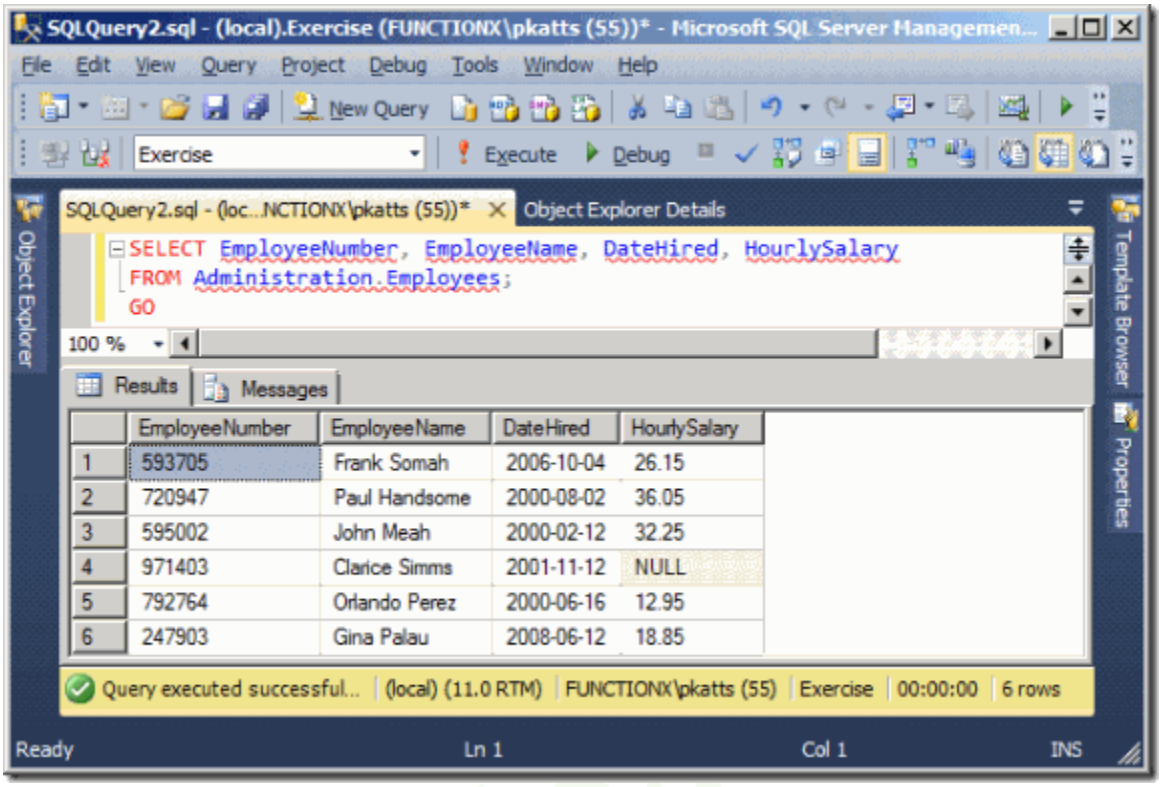
```
USE Exercise;
GO
DROPTABLEAdministration.Employees
GO
CREATETABLEAdministration.Employees
(
EmployeeNumbernvarchar(10),
EmployeeNamenvarchar(50),
DateHireddate,
HourlySalarymoney
);
GO
BEGINTRANSACTIONAddEmployees
INSERTINTOAdministration.Employees
VALUES(N'593705',N'Frank Somah',N'20061004', 26.15),
      (N'720947',N'Paul Handsome',N'20000802', 36.05),
      (N'595002',N'John Meah',N'20000212', 32.25);
GO
```

```
INSERTINTOAdministration.Employees(EmployeeName,EmployeeNumber,DateHired)VALUES(N'Clarice
Simms',N'971403',N'20011112');
GO
INSERTINTOAdministration.Employees
VALUES(N'928375',N'Chuck Stansil',N'20080628');
GO
INSERTINTOAdministration.Employees
VALUES(N'792764',N'Orlando Perez',N'20000616', 12.95);
GO
INSERTINTOAdministration.Employees(EmployeeName,EmployeeNumber,
                                   HourlySalary,DateHired)
VALUES(N'Gina Palau',N'247903', 18.85,N'20080612');
GO
COMMITTRANSACTIONAddEmployees;
GO
```

هنگام اجرا در query editor نتیجه زیر به دست خواهد آمد



توجه کنید که در کد transaction یک errorوجود دارد. record ایجاد شده به صورت زیر است :



ملاحظه میکنید که با وجود transaction , error اجرا شده و بخشي که خطا دارد نادیده گرفته خواهد شد.

در بعضي موارد براي به کار بردن قواعد اشاره شده در بالا , ممکن است که در صورت وجود خطا در یک section , از اجراي transaction صرف نظر کنید. به عبارت دیگر , یک Transaction باید کاملا درست باشد یا اصلا اجرا نشود.براي درخواست از database engine براي تایید درستي آن , اصطلا آن را عقبگرد میکنیم. به این منظور به جاي اجراي از عبارت ROLLBACK TRANSACTION به صورت زیر استفاده میکنیم :

```
ROLLBACK { TRAN | TRANSACTION }

    [ transaction_name | @tran_name_variable

    | savepoint_name | @savepoint_variable ]

[ ; ]
```

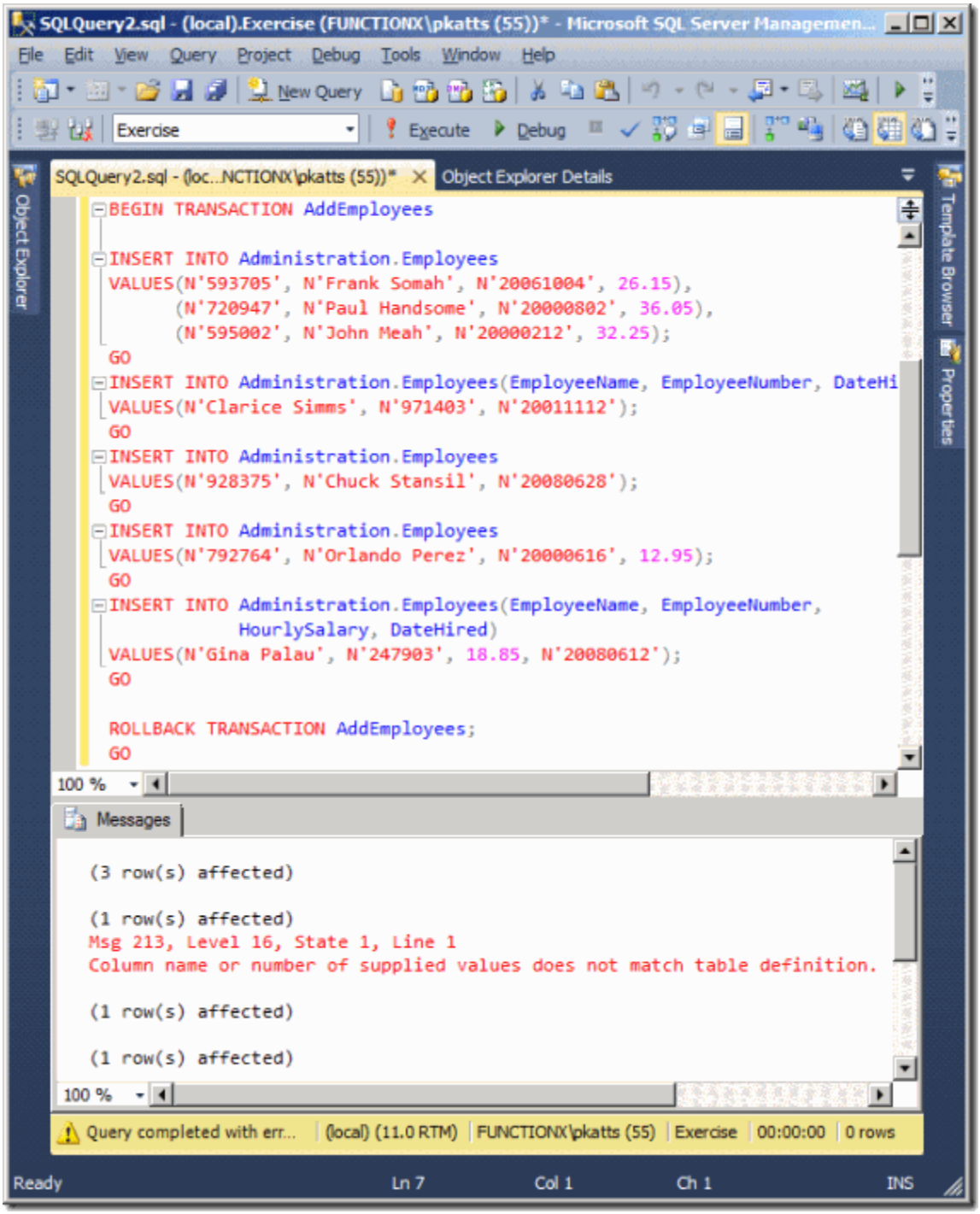
عبارت را با ROLLBACK TRAN یا ROLLBACK TRANSACTIONشروع کنید. در صورت اسم داشتن transaction, اسم یا متغیر حاوي اسم را تایپ کنید.. در صورت تمایل براي ذخیره از عبارت savepoint_name یا @savepoint_variable استفاده کنید.

مثال زیر نشان میدهد که یک transaction در صورت موفق نبودن باید عقبگرد کند.

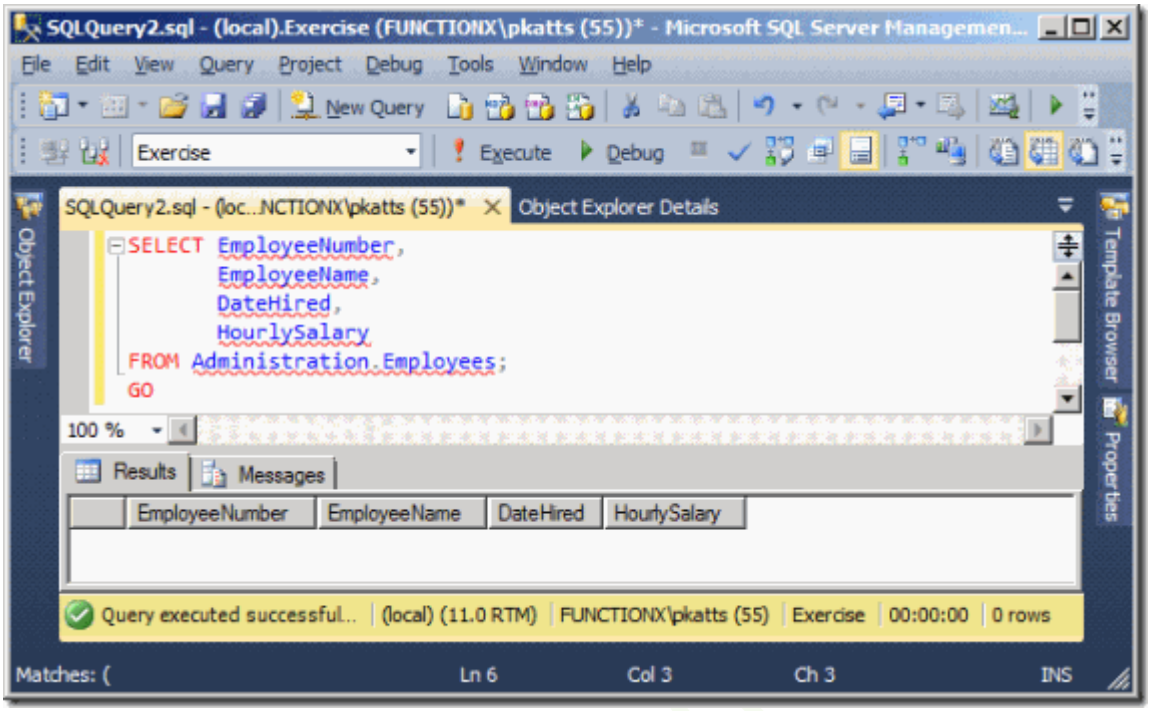
```
USE Exercise;
GO
DROPTABLEAdministration.Employees
GO
CREATETABLEAdministration.Employees
(
EmployeeNumbernchar(10),
EmployeeNamenvarchar(50),
DateHireddate,
HourlySalarymoney
);
GO
BEGINTRANSACTIONAddEmployees
INSERTINTOAdministration.Employees
VALUES(N'593705',N'Frank Somah',N'20061004', 26.15),
      (N'720947',N'Paul Handsome',N'20000802', 36.05),
      (N'595002',N'John Meah',N'20000212', 32.25);
GO
INSERTINTOAdministration.Employees(EmployeeName,EmployeeNumber,DateHired)
VALUES(N'Clarice Simms',N'971403',N'20011112');
GO
INSERTINTOAdministration.Employees
VALUES(N'928375',N'Chuck Stansil',N'20080628');
GO
INSERTINTOAdministration.Employees
VALUES(N'792764',N'Orlando Perez',N'20000616', 12.95);
GO
INSERTINTOAdministration.Employees(EmployeeName,EmployeeNumber,
            HourlySalary,DateHired)
VALUES(N'Gina Palau',N'247903', 18.85,N'20080612');
GO
ROLLBACKTRANSACTIONAddEmployees;
GO
```

این کد ابتدا یك table به اسم Employee تشکیل میدهد. سپس از database engine درخواست اضافه کردن چند record میکند. تشکیل record ها در transaaction با استفاده از گزینه ي rollback انجام پذیر است.

دقت کنید که یك error در transaction وجود دارد



زماني كه كد بالا اجرا ميشود table تشكيل ميشود چون خارج transaction قرار دارد. ولي table نهايي خالي خواهد بود



به خاطر وجود يك error كل transaction رد خواهد شد

Controlling a Transaction's Isolation Level

در database يك record آشفته (dirty) ناميده خواهد شد , اگر پس از آخرين استفاده از table محتواي آن تغيير داده شود.

موقع تشكيل يك transaction شما ميتوانيد طوري database engine را تنظيم كنيد كه transaction را اجرا يا آن را رد كند. براي استفاده از اين گزينه از فرمول زير استفاده كنيد:

```
SET TRANSACTION ISOLATION LEVEL

  { READ UNCOMMITTED

  | READ COMMITTED

  | REPEATABLE READ

  | SNAPSHOT

  | SERIALIZABLE
```

}

[;]

ابتدا از SET TRANSACTION ISOLATION LEVEL استفاده کرده سپس از يك مقدار استفاده كنيد

READ UNCOMMITTED : اين مقدار از database درخواست ميكند كه record هاي را بخواند ولي آنها را اجرا نكند.

READ COMMITTED : اين مقدار مشخص ميكند كه transaction , record هاي آشفته را از باقي transaction ها جدا كند و همچنين دسترسي آنها را از ان record قطع كند

Snapshot

serializable