* Multiplicity
  + 1-1
    - 1 table or 2 tables with unique fk
  + 1-N
    - 2 tables fk not null
  + N-N
    - 3 tables with a middle “junction table”
* Triggers
  + Triggers replace the action of an INSERT, UPDATE or DELETE
  + Triggers can be BEFORE, AFTER or INSTEAD OF
* Procedures
  + Are like functions but allow any sql command including db write
  + They don’t have to return anything
  + You can only call them with EXECUTE, never inside a SELECT or any other command

CREATE PROCEDURE Movie.RenameMovies(@newname NVARCHAR(50), @rowschanged INT OUTPUT)

AS

BEGIN

//we can use WHILE loops, IF ELSE, TRY CATCH

BEGIN TRY

IF (EXISTS(SELECT COUNT(\*) FROM Movie.Movie))

BEGIN

SET @rowschanged = (SELECT COUNT(\*) FROM Movie.Movie

UPDATE Movie.Movie

SET Title = @newname

END

ELSE

BEGIN

RAISERROR(‘No movies found’, 16, 1)

END

END TRY

BEGIN CATCH

PRINT ERROR\_MESSAGE();

END CATCH

END

EXECUTE Movie.RenameMovies ‘Movie’

Transactions-

ACID

1. Transactions must be atomic
   1. All operations succeed 100% or do nothing

C- Consistent

1. Operations should not be allowed to violate constraints

I- Isolated

1. Even if multiple transactions run at the same time, it should be as if each was running alone
2. 4 levels of isolation
   1. Read\_uncommitted
      1. Doesn’t fix a dirty read
         1. See other transactions’ unfinished work
   2. Read\_committed
      1. Doesn’t fix non-repeatable read
         1. Could read a row twice and another finished transaction changed it in between my reads
   3. Repeatable\_read
      1. Doesn’t fix phantom read
         1. Other transactions can insert new rows that meet filtered conditions
   4. Serializable
      1. Treats all steps as though they were truly serialized

As isolation increases, performance decreases and the number of locks increases

D- Durable

1. Transaction is incomplete until it is written to persistent storage

Transactions are written like:

BEGIN TRANSACTION

COMMIT TRANSACTION

ROLLBACK TRANSACTION: will bring back a transaction if an error occurs