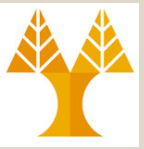
Department of Computer Science
University of Cyprus



EPL342 –Databases

Lecture 19: Internal DB Programming II

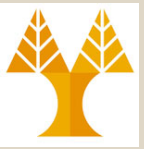
Internal DB Programming, Scripts/Batches in TSQL

(Chapter 10.1, Elmasri-Navathe 7ED +
TransactSQL Reference Guide

Demetris Zeinalipour

http://www.cs.ucy.ac.cy/courses/EPL342

Περιεχόμενο Διάλεξης



Ολοκλήρωση Διάλεξης 18.

Internal DB Programming II

- Εσωτερικός Προγραμματισμός ΣΔΒΔ
- **Scripts/Batches** σε TSQL
- **Stored Procedures (Sprocs)** σε TSQL
- **User Defined Functions (UDFs)** σε TSQL

Προγραμματισμός Λειτουργιών μιας DBMS

- Ενώ η **SQL** ξεκίνησε ως **δηλωτική γλώσσα** διατύπωσης επερωτήσεων στη συνέχεια επεκτάθηκε με **εντολές DDL** και οι σύγχρονες ΒΔ παρέχουν σήμερα και δυνατότητες για διαδικαστικό προγραμματισμό ΜΕΣΑ στην ίδια την βάση.
- Γενικά, υπάρχουν οι ακόλουθες κατηγορίες:
  - **Εσωτερικός Προγραμματισμός:** Scripts/Batches, Sprocs, UDFs, Views, Triggers, Assertions, κτλ
  - **Εξωτερικός Προγραμματισμός:** Μέσω *Γλώσσας Προγραμματισμού* (Host Language): Embedded SQL , Dynamic SQL, APIs και *Διεπαφών Βάσεων* (ODBC, JDBC, MS OLEDB, ADO.NET κτλ) ➔ επόμενη διάλεξη.
- Στην ερχόμενη διάλεξη θα ασχοληθούμε με τον Εξωτερικό Προγραμματισμό (Φροντιστήριο: JDBC)

Προγραμματισμός Λειτουργιών μιας DBMS

Εξωτερικός Προγραμματισμός (Μέσω Εφαρμογής)

DBMS

SProcs

UDFs

Views

Catalog/ Indexes

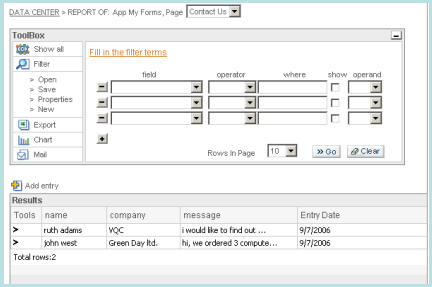
Tables

Triggers

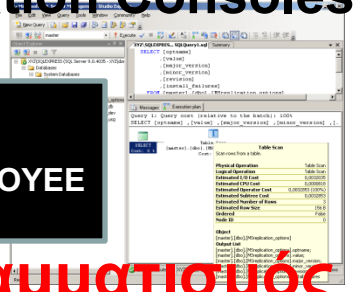
Interfaces: JDBC, ODBC, κτλ.

Programming Language (π.χ., JAVA, C, C#,…)

Web Languages (PHP, Python, ASP, Ruby, Perl)



Administration Consoles



```
# SQLCMD
1> SELECT *
FROM EMPLOYEE
2> go
```

Εσωτερικός Προγραμματισμός

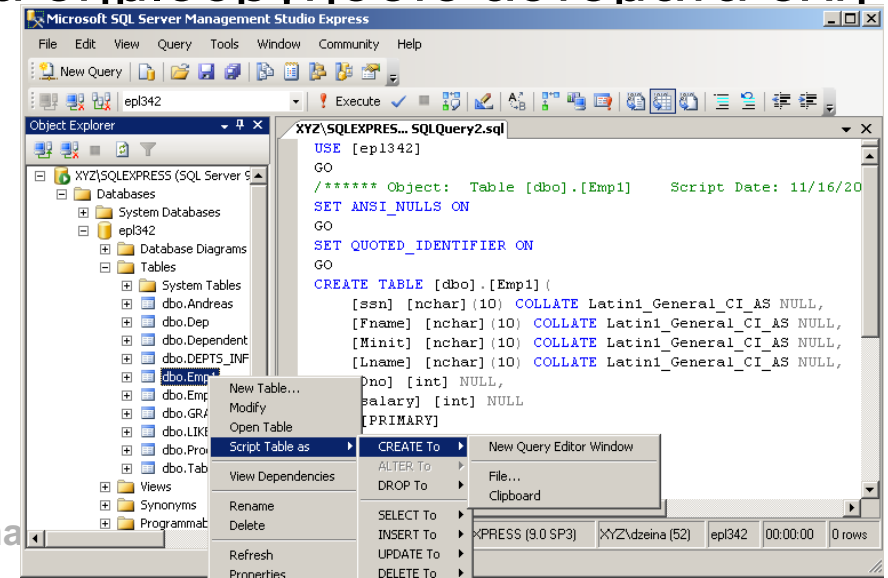
Εξωτερικός Προγραμματισμός (Διαδραστικός)

Εσωτερικός Προγραμματισμός μιας DBMS

- Σήμερα θα δούμε πως μπορούμε να προγραμματίσουμε τον SQL Server με τους ακόλουθους τρόπους:
  A. **Scripts / Batches:** Απλή ακολουθία SQL εντολών με τελεστές έλεγχου, επανάληψης, μεταβλητές, κτλ. (αντίστοιχη λογική με shell scripts στο UNIX)
  B. **Stored Procedures (Sprocs):** Ακολουθίες SQL εντολών με παραμέτρους αποθηκευμένες στην βάση.
  C. **User Defined Functions (UDFs):** Επιτρέπουν στον χρήστη να ορίζει συναρτήσεις που μπορούν να χρησιμοποιηθούν στα πλαίσια του SELECT ή WHERE (π.χ., MetersToInches(decimal(10,3))

A) Scripts σε TSQL

- **Scripts: Ακολουθία (T)SQL Εντολών** η οποία **αποθηκεύεται** σε ένα **αρχείο** για **επαναχρησιμοποίηση**.
- Παραδείγματα Χρήσης:
  - **Backup / Restore** πινάκων ή ολόκληρης της DB.
    - Θυμηθείτε το Northwind.sql (Εργαστήριο 8) το οποίο χρησιμοποιήσατε για να δημιουργήσετε αυτόματα όλη την βάση.
  - **Επανάληψη Συχνών Λειτουργιών** (Συντήρηση, κτλ)



A) Scripts σε TSQL (Μεταβλητές)

Παράδειγμα Script σε TSQL

Set current Database (otherwise script will be executed on whatever database is currently open)

USE epl342;

Declare Variable with Name TEST of TYPE int
(default = NULL)

DECLARE @TEST int

Assign Value to Variable TEST

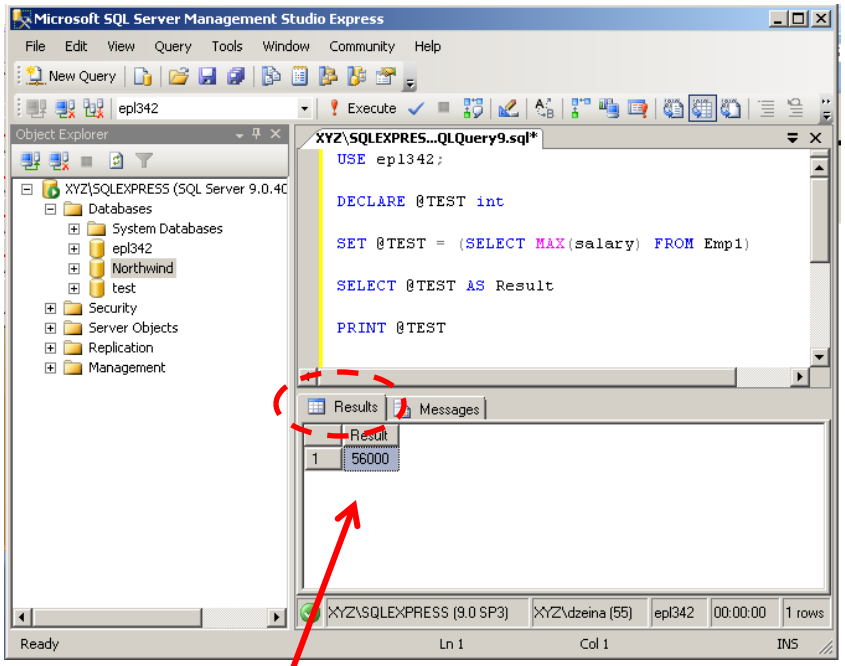
SET @TEST = (SELECT MAX(salary) FROM Emp1)

Display the TEST Variable with Column Name Result

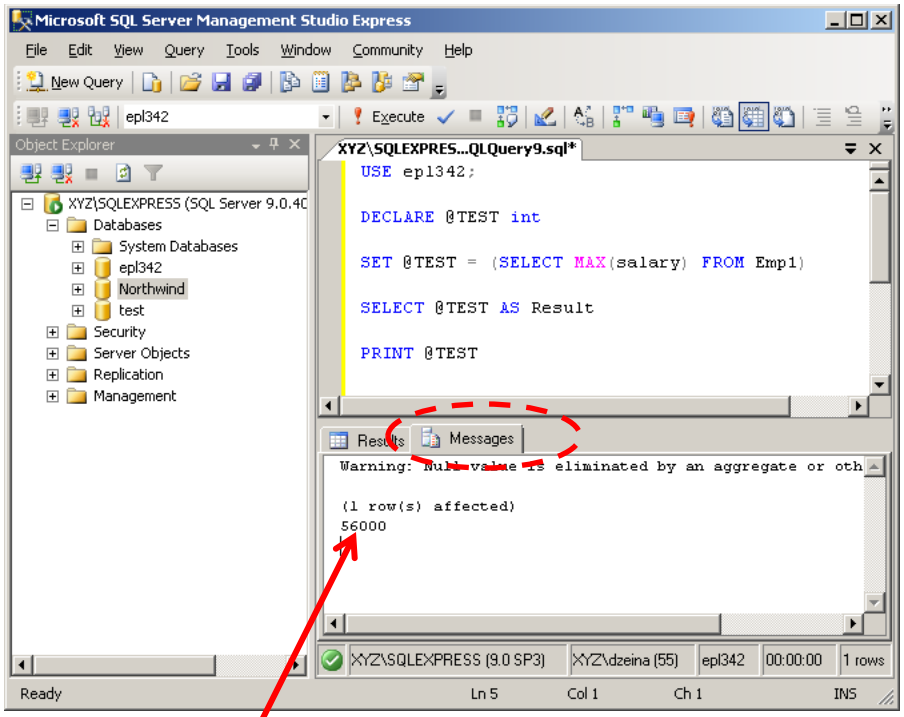
SELECT @TEST AS Result -- Αντίστοιχο του **STDOUT**

PRINT @TEST – Αντίστοιχο του **STDERR.** Το μήνυμα είναι μέχρι 8000 χαρακτήρες και επιστρέφεται στον χρήστη.

A) Scripts σε TSQL (Μεταβλητές)

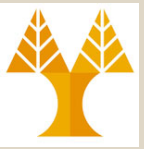


SELECT



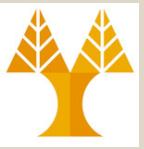
PRINT (δες και RAISERROR για try…catch)

A) Scripts σε TSQL



- Χαρακτηριστικά των Scripts
  1. Τα Scripts **δημιουργούνται** και **εκτελούνται** από κάποιο **χρήστη** ή ως μέρος άλλου **script**.
  2. Το Script εκτελείται **γραμμή-γραμμή** από **πάνω** προς τα **κάτω** (η σύνταξη ελέγχετε πριν την εκτέλεση) από τον μεταφραστή της βάσης.
  3. Εάν **προκύψει λάθος (runtime λάθος)** τότε **ΑΚΥΡΩΝΕΤΑΙ** ολόκληρο το script.
     - Ένα Script εκτελείται ως μια δοσοληψία (transaction). Συνεπώς αποτελεί μια **ατομική πράξη**: “Είτε εκτελείται ολόκληρο ή καθόλου”!
     - Μετά από κάποιο λάθος **επαναφέρεται** (**ROLLBACK**) πίσω στην αρχική κατάσταση η βάση δεδομένων.

A) Scripts σε TSQL
(Παράδειγμα με χρήση @@IDENTITY)



Παράδειγμα Εισαγωγής Συσχετιζόμενων Δεδομένων σε Δυο Πίνακες (Order και OrderDetails).

```
USE Northwind
DECLARE @NewOrderID int

INSERT INTO Orders(CustomerID, OrderDate)
VALUES (15, DATEADD(day,-1,GETDATE()))

SET @NewOrderID = @@IDENTITY
-- ή SELECT @NewOrderID = @@IDENTITY

INSERT INTO [Order Details](OrderID, ProductID, UnitPrice, Quantity)
VALUES (@NewOrderID, 1, 50, 25)

SELECT 'The OrderID of the INSERTed row is ' + CONVERT(varchar(8),
    @NewOrderID)
```

Current Date Function

Assign value to variable

***@@:** System Function (last recorded Identity)*

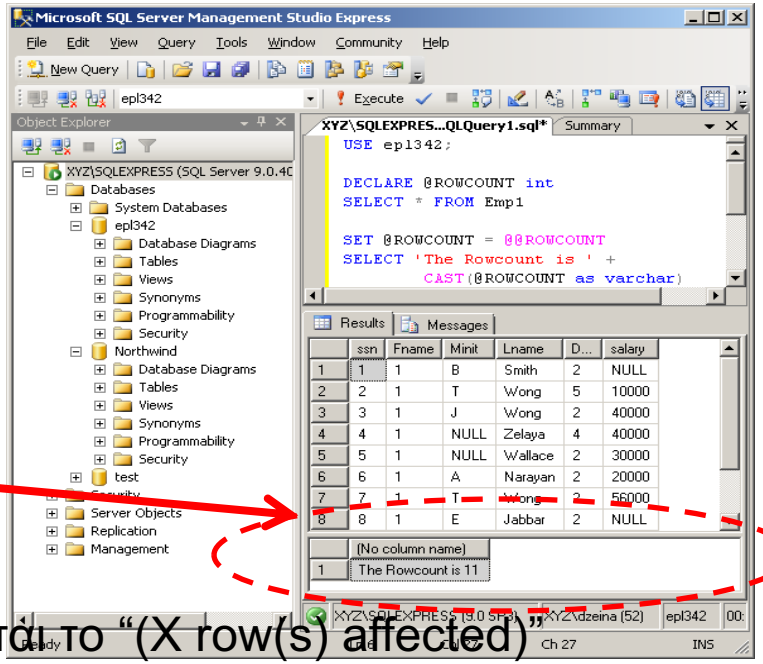
Casting integer to string

A) Scripts σε TSQL

(Χρήση Συνάρτησης Συστήματος @@ ROWCOUNT)

- Η συνάρτηση συστήματος **@@ROWCOUNT** σας επιστρέφει τον **αριθμό των πλειάδων** που **επηρεάστηκαν** ή **διαβάστηκαν** από την τελευταία **SQL επερώτηση**.
  - Θυμηθείτε το μήνυμα: "(X row(s) affected)" που προκύπτει μετά από ανάγνωση/αλλαγή δεδομένων ενός πίνακα

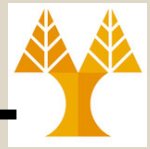
```
USE epl342;

DECLARE @ROWCOUNT int
SELECT * FROM Emp1

SET @ROWCOUNT = @@ROWCOUNT
SELECT 'The Rowcount is' +
        CAST(@ROWCOUNT as varchar);
```



Με την εντολή "**SET NOCOUNT ON**" δεν τυπώνεται το "(X row(s) affected)"

- Θα μας απασχολήσει αργότερα στα stored procedures … για να μην αποστέλνονται αποτελέσματα για από συνεχόμενες εντολές στον χρήστη

Α) Δέσμες (Batches) σε TSQL



- Τα **Batches (Δέσμες)** είναι Scripts εντολών TSQL τα οποία διαχωρίζονται με την εντολή **GO** και τα οποία εκτελούνται ανεξάρτητα μεταξύ τους (δηλαδή όχι στα πλαίσια του ίδιου transaction)
- **Παράδειγμα:**

```
USE epl342;
GO -- αποστολή δέσμης εκφράσεων TSQL στον SQL Server
INSERT INTO Emp1(SSN) VALUES ('4411111993')
GO -- αποστολή δέσμης εκφράσεων TSQL στον SQL Server
INSERT INTO Emp1(SSN) VALUES ('3311111993')
GO -- δεν πρέπει να υπάρχουν άλλες εντολές στην ίδια γραμμή με το GO!
```

- Η εντολή **GO** λειτουργεί ΜΟΝΟ στο πλαίσιο του **SQL Management Studio** ή της **sqlcmd** (ή **osql**).
  - Σε προγράμματα γίνεται κάτι αντίστοιχο μέσω εξειδικευμένων εντολών, π.χ. στη JAVA: **stmt.executeUpdate(query);**

Α) Δέσμες (Batches) σε TSQL

- Κάποιες **εντολές** είναι **αναγκαστικό** να είναι μέρος του **δικού** τους **Batch** (δηλαδή **πρεπει** να **ακολουθούνται** από GO).
  - Μερικές από αυτές είναι:
    - **CREATE TRIGGER**
    - **CREATE VIEW**
    - **CREATE PROCEDURE** ➔ θα το δούμε σε λίγο
- **Συντακτικά Λάθη** ελέγχονται όπως και στα Scripts πριν την εκτέλεση ολόκληρου του Batch
- Εάν προκύψουν **Runtime λάθη** σε ένα batch στο σημείο Χ τότε δεν εκτελείται καμία εντολή μετά το Χ.
  - Εντολές πριν το Χ δεν γίνονται ROLLBACK και αλλάζουν μόνιμα την κατάσταση της βάσης.

Α) Scripts σε TSQL (Δυναμική SQL σε TSQL)

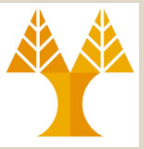
- **Δυναμική SQL (Dynamic SQL):** Εκφράσεις SQL που παράγονται κατά την εκτέλεση ενός script ή προγράμματος με χρήση της εντολής **EXEC**
  - Χρήσιμες εάν δεν είναι γνωστή εκ των προτέρων η SQL έκφραση.
  - **Σημείωση:** Όλα τα προηγούμενα παραδείγματα ήταν με στατικές εκφράσεις SQL (Static SQL)
- Παράδειγμα

```
USE epl342;
GO
DECLARE @Salary int;
SET @Salary = (SELECT MAX(salary) FROM EMP1);
EXEC ('SELECT * FROM Emp1 WHERE salary=' + @Salary);
GO
```

Έκφραση SQL που παράγεται δυναμικά κατά την εκτέλεση

concat

A) Scripts σε TSQL (Δυναμική SQL σε TSQL)



-- Δημιουργία Πίνακα Ως Χρήστης DBO (Database Owner)

```
USE epl342;
GO
EXECUTE ('CREATE TABLE SalesTable (SalesID int, SalesName
    varchar(10));')
AS USER = 'dbo';
GO
```

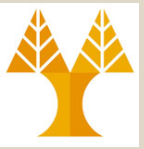
Το EXEC είναι συντομογραφία του EXECUTE

Επισημάνσεις για το EXEC

- Εκτελείται κάτω από με τα **ίδια δικαιώματα** με τον Script που το καλεί.
- Το EXEC τρέχει με τα **ίδιο connection** με το πρόγραμμα που το καλεί.
- Εάν θα γίνει σύμπτυξη με συνάρτηση, τότε αυτή πρέπει να γίνει πριν την κλήση της EXEC.
  - **ΛΑΘΟΣ:** EXEC ('SELECT * FROM Emp1 WHERE date=' **+ GetDate()**);

Εάν ήταν @DATEVAL δεν θα είχε πρόβλημα

A) Scripts σε TSQL (Έλεγχος Ροής σε TSQL)



- **Τελεστής Έλεγχου**
  - **IF <Boolean Expression>**
    <SQL statement> | BEGIN <code series> END
    ELSE
    <SQL statement> | BEGIN <code series> END

Επισημάνσεις

- Προφανώς επιτρέπεται και η **εμφώλευση** αυτού του τελεστή τόσο με τον εαυτό του όσο και με άλλους τελεστές.
- Εάν το <Boolean Expression> είναι NULL τότε είναι FALSE η λογική συνθήκη.
- **ΛΑΘΟΣ**: IF @myvar=NULL ➔ **ΣΩΣΤΟ:** IF @myvar IS NULL
- Υπάρχει και η **CASE** (δηλ., αντίστοιχο της SWITCH στη C) http://msdn.microsoft.com/en-us/library/ms181765.aspx

A) Scripts σε TSQL (Έλεγχος Ροής σε TSQL)

```
USE AdventureWorks;
GO
SELECT   ProductNumber, Category =
    CASE ProductLine
      WHEN 'R' THEN 'Road'
      WHEN 'M' THEN 'Mountain'
      WHEN 'T' THEN 'Touring'
      WHEN 'S' THEN 'Other sale items'
      ELSE 'Not for sale'
    END,  Name, SSN
FROM Production.Product
ORDER BY ProductNumber;
GO
```

Η συνθήκη μπορούσε να είναι και πιο σύνθετη, π.χ., (ProductNumber % 2) = 0

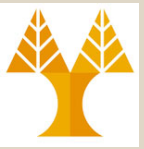
A) Scripts σε TSQL (Επανάληψη σε TSQL)

- **Τελεστής Επανάληψης** σε TSQL

```
USE epl342
GO
DECLARE @var INT

WHILE 1 = 1
BEGIN
    WAITFOR DELAY '00:00:01'
    SET @var = (SELECT MAX(salary) FROM EMP1)
END
```

Αναμονή 1 δευτερολέπτου μέχρι την επόμενη κληση

A) Scripts σε TSQL (Επανάληψη σε TSQL)



- **Τελεστής Επανάληψης**
  - **WHILE Boolean_expression**

```
{
    sql_statement |
    statement_block |    -- several statements in BEGIN  … END
    BREAK |
    CONTINUE
}
```