9η Εργαστηριακή Άσκηση: Stored Procedures - Triggers

Σκοπός της παρούσας εργαστηριακής άσκησης, είναι η εξοικείωση του σπουδαστή με τη δημιουργία αποθηκευμένων διαδικασιών (Stored Procedures) και σκανδάλης (trigger) στον MICROSOFT SQL SERVER 2008.

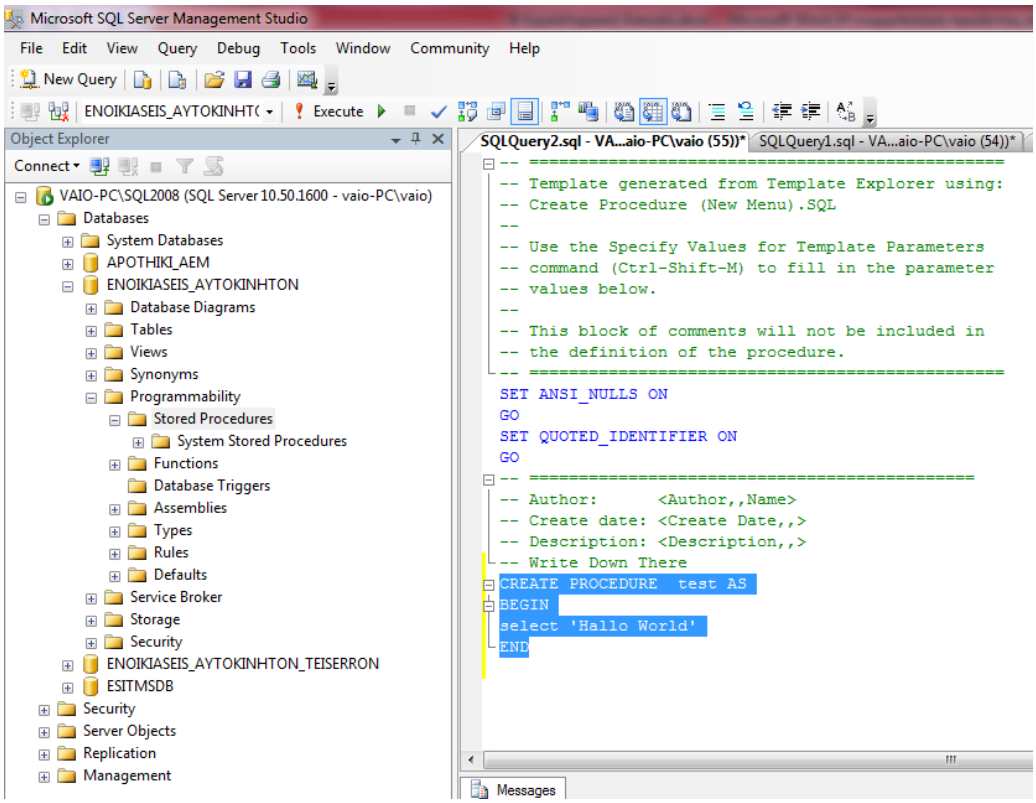
Αποθηκευμένες Διαδικασίες (Stored Procedures):

Μια store procedure είναι ένα πρόγραμμα που αποτελείται από SQL εντολές καθώς και εντολές προγραμματισμού. Εκτελείται ταχύτερα επειδή έχει προηγουμένως μεταγλωττιστεί στον database server και όχι στο πρόγραμμα εφαρμογής.

Παράδειγμα 1 :

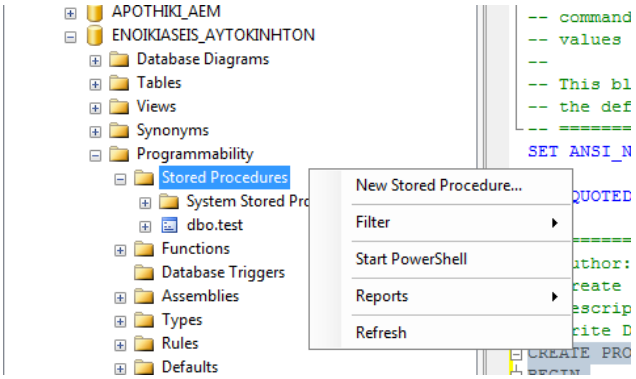
Μεταβαίνουμε στο SQL Server Management Studio, επιλέγουμε τη βάση μας και με την επιλογή "Stored Procedures" → Δεξί κλικ → New Stored Procedure (Βλ. Εικόνα 9.1) ανοίγει ένα παράθυρο όπου γράφουμε τον κώδικα της διαδικασίας

```
CREATE PROCEDURE  test AS
BEGIN
select "Hallo World"
END
```



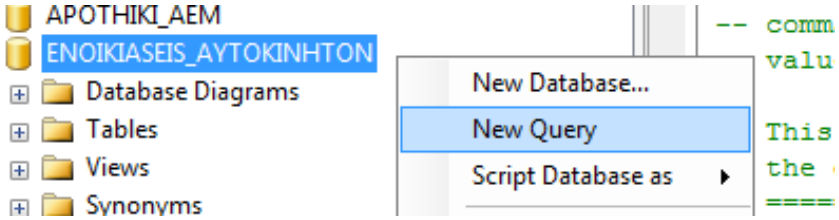
Εικόνα 9.1 Δημιουργία Αποθηκευμένης Διαδικασίας

Για να αποθηκευτεί η διαδικασία αυτή πιέζετε το κουμπί !Execute από την γραμμή εντολών και στη συνέχεια κάντε refresh στην επιλογή Stored Procedures.



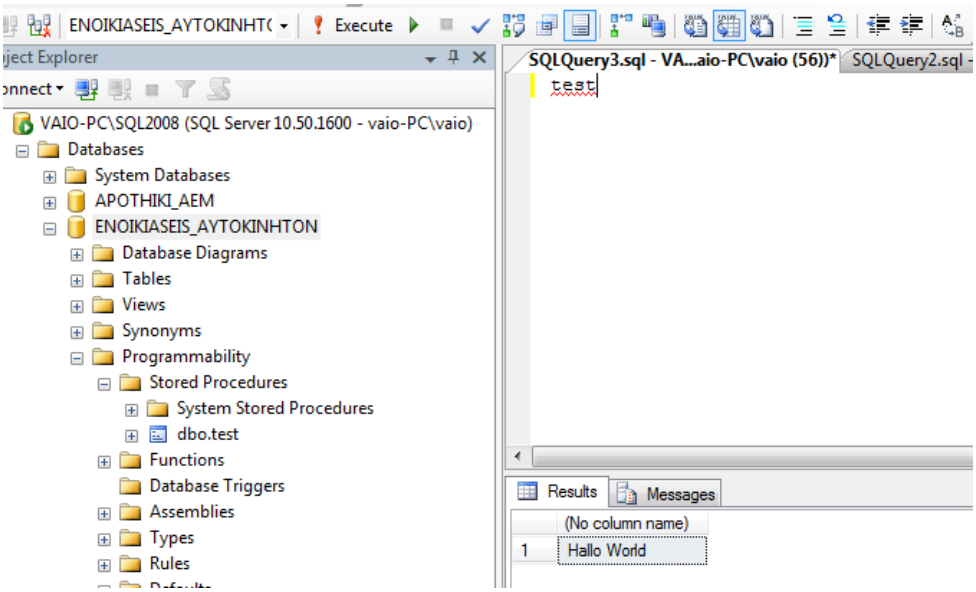
Εικόνα 9.2 Ανανέωση των Αποθηκευμένων Διαδικασιών

Για να εκτελεστεί η Αποθηκευμένη Διαδικασία που μόλις δημιουργήθηκε πιέζουμε Δεξί κλικ πάνω στη Βάση Δεδομένων και επιλέγουμε New Query



Εικόνα 9.3 Δημιουργία νέου ερωτήματος για εκτέλεση των Αποθηκευμένων Διαδικασιών

Γράφουμε το όνομα της διαδικασίας και την εκτελούμε πατώντας το !Execute από την γραμμή εντολών (εικονίδιο με θαυμαστικό) και βλέπουμε τα αποτελέσματα. Το συγκεκριμένο παράδειγμα τυπώνει “hello world”.



Εικόνα 9.4 Εκτέλεση των Αποθηκευμένων Διαδικασιών

Λόγος χρήσης των αποθηκευμένων διαδικασιών (Stored Procedures) :

Κάθε διαδικασία εκτελείται στον Database Server, επομένως είναι ανεξάρτητη από την πλατφόρμα εφαρμογής. Επιπλέον λαμβάνουμε ταχύτατα τα αποτελέσματα για τον λόγο ότι η εκτέλεση γίνεται στον Database Server και δεν χρειάζεται η επικοινωνία μέσω δικτύου με την πλατφόρμα εφαρμογής. Στο παράδειγμα 1, εκτελέσαμε Procedure μιας εντολής η οποία θα μπορούσε ναν γραφεί και έξω από το block εντολών. Για να εκτελέσουμε πολλές εντολές σε μια διαδικασία πρέπει υποχρεωτικά να βρίσκονται μέσα σε ένα μπλοκ begin - end. Με τον ίδιο τρόπο που περιγράψαμε πριν δημιουργήστε τις παρακάτω διαδικασίες.

Παράδειγμα 2 :

```
CREATE PROCEDURE  test2 AS
BEGIN
select 'ASKISI 9 VASEIS DEDOMENON II'
select 'Δημιουργία Αποθηκευμένων Διαδικασιών'
END
```

Χρήση μεταβλητών σε διαδικασία :

Οι μεταβλητές δηλώνονται με το σύμβολο "@" πριν τον ορισμό τους.

Παράδειγμα 3 :

```
CREATE PROCEDURE test3 (@var1 INT,@var2 INT)  AS
BEGIN
      SELECT @var1+@var2;
END
GO
```

την καλούμε με 2 ορίσματα :

```
test3 4,6
```

αποτέλεσμα :

```
10
```

Παράδειγμα 4 :

Παράδειγμα διαδικασίας που επηρεάζει τις τιμές του πίνακα :

```
CREATE PROCEDURE AFXISI_TIMIS (@POSOSTO INT) AS
BEGIN
UPDATE AYTOKINITA
      SET TIMH_ENOIKIASIS = TIMH_ENOIKIASIS*(1+@POSOSTO/100)
END
GO
```

Έστω ότι θέλουμε να γίνει αύξηση τιμών κατά 10%. Παρατηρούμε τις τιμές του πίνακα πριν την αύξηση.

Εφόσον γράψουμε τον παραπάνω κώδικα και πατήσουμε !Execute και μετά Refresh μεταβαίνουμε σε νέο Query για να καλέσουμε και να εκτελέσουμε τη διαδικασία γράφοντας «AFXISI_TIMIS 10». Πατώντας !Execute (Εικονίδιο θαυμαστικού), εκτελείται η διαδικασία. Παρατηρούμε τις νέες τιμές ενοικίασης στον πίνακα οι οποίες θα είναι αυξημένες κατά 10%.

Σκανδάλες (Triggers):

Trigger (Σκανδάλη), ονομάζεται ένα αντικείμενο Βάσης Δεδομένων, το οποίο απευθύνεται σε συγκεκριμένο πίνακα και εκτελείται όταν συμβεί ένα συγκεκριμένο γεγονός σε αυτόν τον πίνακα.

Η δήλωση μιας σκανδάλης ακολουθεί το εξής πρότυπο:

```
CREATE  TRIGGER trigger_name ON tbl_name
AFTER INSERT, DELETE, UPDATE
      BEGIN
      …….
      …….
      END
```

όπου:

- trigger_name : Το όνομα μιας νέας σκανδάλης
- tbl_name : Το όνομα του πίνακα που αφορά η σκανδάλη
- AFTER INSERT, DELETE, UPDATE : Το πότε θα εκτελεστούν οι εντολές της σκανδάλης.

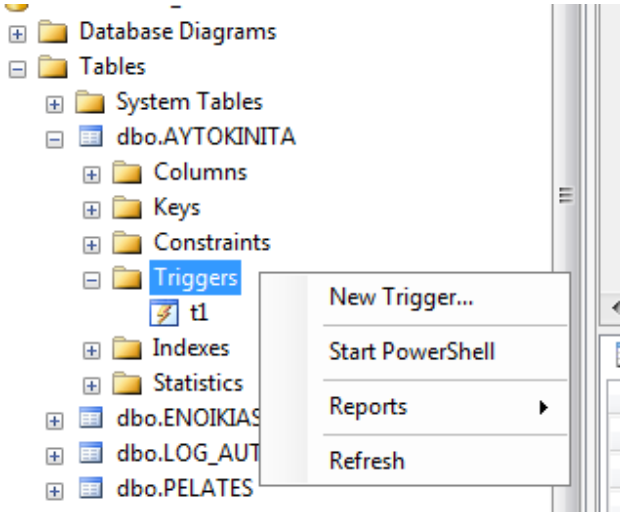
Για εκτέλεση πολλαπλών ενεργειών, όπως και στις Stored Procedures, χρησιμοποιείται η δομή BEGIN {εντολές} END.

Παράδειγμα 5 :

Έστω ότι θέλουμε να υπολογίζεται αυτόματα το πεδίο του πίνακα ENOIKIASH, TELIKO_POSO να ισούται με το άθροισμα του TELIKO_POSO συν XREOSH_ZHMIAS σε κάθε ενέργεια INSERT, UPDATE, DELETE στον πίνακα.

Δημιουργία σκανδάλης :

Αφορά τον πίνακα ENOIKIASH, οπότε ή δημιουργούμε ένα νέο Query ή επιλέγουμε ΑΥΤΟΚΙΝΗΤΑ → Triggers → Δεξί κλικ → New Trigger (βλ. Εικόνα 9.9)



Εικόνα 9.5 Δημιουργία Νέας Σκανδάλης

Στο παράθυρο που εμφανίζεται γράφουμε τον παρακάτω κώδικα:

```
CREATE TRIGGER TEL_POSO ON ENOIKIASH
AFTER INSERT, UPDATE, DELETE
AS
BEGIN
UPDATE ENOIKIASH SET TELIKO_POSO=POSO_PLHROMHS + XREOSH_ZHMIAS
END
```

Εφόσον γράψαμε τον κώδικα του Trigger, ελέγχουμε αν είναι σωστή η σύνταξη πατώντας "!Execute" και έχει δημιουργηθεί επιτυχώς η σκανδάλη. Για να δούμε τα αποτελέσματα θα δημιουργήσουμε μια νέα εγγραφή στον πίνακα ENOIKIASH. Αφού εισάγουμε τα απαραίτητα πεδία, φτάνοντας στο πεδίο "TELIKO_POSO" πατώντας το πλήκτρο "TAB", παρατηρούμε ότι το πεδίο υπολογίζεται αυτόματα

Παράδειγμα 6 :

Εισάγουμε ένα νέο πεδίο στον πίνακα PELATES, με όνομα HLIKIA τύπου INT, στο οποίο θέλουμε να υπολογίζεται αυτόματα η ηλικία του κάθε πελάτη(βλ Εικόνα 1.9.14), από την ημερομηνία γέννησης που υπάρχει ήδη στον πίνακα. Για να δημιουργηθεί αυτή η σκανδάλη πρέπει να γνωρίζουμε δύο συναρτήσεις :

- DATEDIFF (YEAR, HMEPOMHNIA_ΑΡΧΗΣ , HMEPOMHNIA_ΤΕΛΟΥΣ)

η οποία υπολογίζει τη διαφορά δυο ημερομηνιών (HMEPOMHNIA_ΑΡΧΗΣ , HMEPOMHNIA_ΤΕΛΟΥΣ) σε χρόνια, εβδομάδες μέρες (YEAR, WEEK, DAY).

- GETDATE () η οποία επιστρέφει την τρέχουσα ημερομηνία.

Οπότε ο κώδικας της σκανδάλης για τον πίνακα PELATES, διαμορφώνεται όπως παρακάτω :

```
CREATE TRIGGER HLIKIA_PELATH ON [dbo].[PELATES]
FOR INSERT, UPDATE, DELETE
AS
BEGIN
      UPDATE PELATES SET HLIKIA =
                  DATEDIFF ( YEAR , HM_GENNISIS , GETDATE() )
END
```

Καταχωρώντας μια νέα εγγραφή στον πίνακα πελάτες παρατηρούμε ότι υπολογίζεται η ηλικία του πελάτη αυτόματα.

Παράδειγμα 7 :

Δημιουργούμε έναν πίνακα με όνομα LOG_AUTO στον οποίο με την ενεργοποίηση μιας σκανδάλης (TRIGGER) που θα εφαρμόζεται πάνω στον βασικό πίνακα ΑΥΤΟΚΙΝΗΤΑ, θα αποθηκεύονται οι τροποποιήσεις που θα γίνονται στο πεδίο ΤΙΜΗ_ΕΝΟΙΚΙΑSIS του πίνακα ΑΥΤΟΚΙΝΙΤΑ.

Ο πίνακας LOG_AUTO περιλαμβάνει :

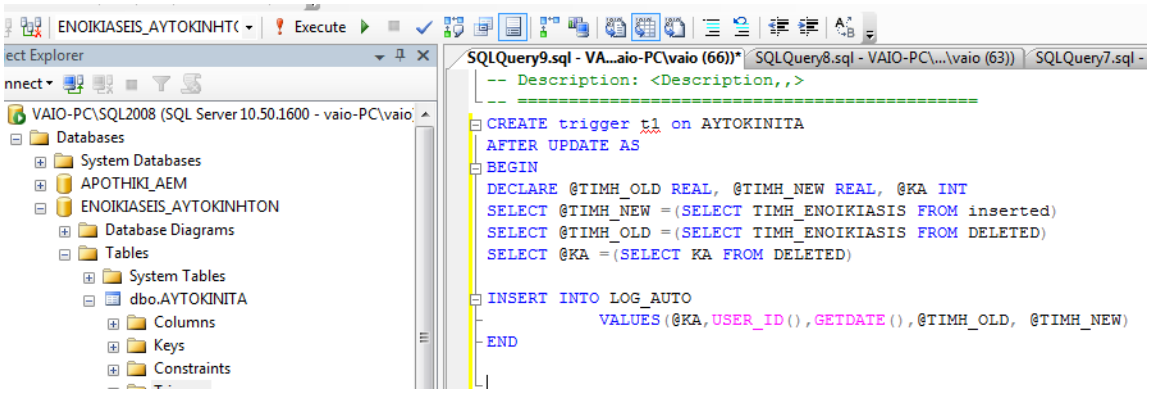
```
AUTO_NUM τον αριθμό πινακίδας του αυτοκινήτου,
USER_ID τον κωδικό του χρήστη που κάνει αλλαγές στον πίνακα,
HMEROMHNIA που έχει η αλλαγή στον πίνακα,
OLD η παλιά τιμή ενοικίασης,
NEW η νέα τιμή ενοικίασης
```

Ο πίνακας μπορεί να δημιουργηθεί είτε χειροκίνητα, όπως δημιουργήσαμε πίνακες στις πρώτες εργαστηριακές ασκήσεις, είτε εκτελώντας το παρακάτω Query :

```
CREATE TABLE LOG_AUTO(
      AUTO_NUM varchar(10),
      USER_ID varchar(20),
      HMEROMHNIA datetime,
      OLD float,
      NEW float)
```

Δημιουργούμε λοιπόν ένα Trigger για τον πίνακα “ΑΥΤΟΚΙΝΙΤΑ” με τον παρακάτω κώδικα :

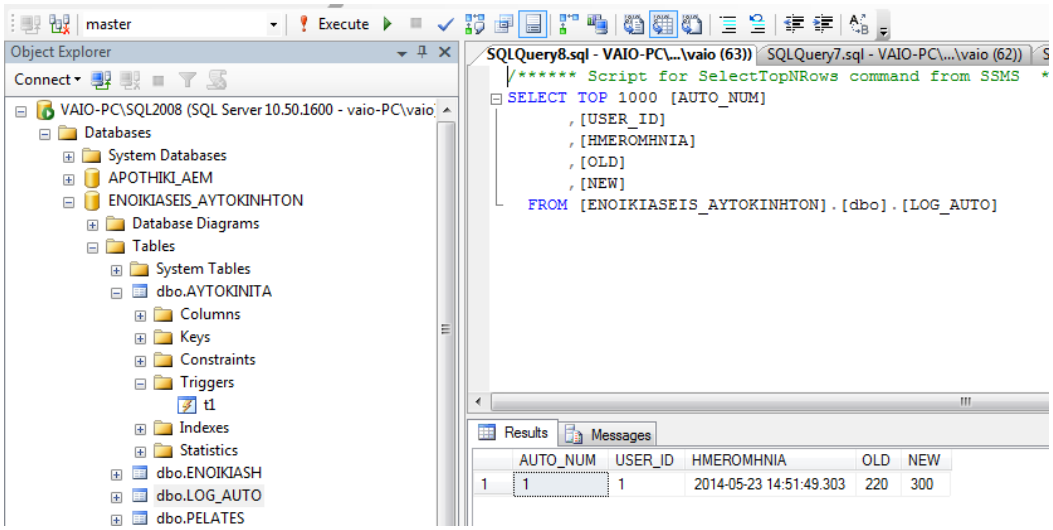
```
CREATE trigger t1 on AYTOKINITA
AFTER UPDATE AS
BEGIN
DECLARE @TIMH_OLD REAL, @TIMH_NEW REAL, @KA INT
SELECT @TIMH_NEW =(SELECT TIMH_ENOIKIASIS FROM inserted)
SELECT @TIMH_OLD =(SELECT TIMH_ENOIKIASIS FROM DELETED)
SELECT @KA =(SELECT KA FROM DELETED)

INSERT INTO LOG_AUTO
          VALUES(@KA,USER_ID(),GETDATE(),@TIMH_OLD, @TIMH_NEW)
END
```



Εικόνα 9.6 Εντολές Σκανδάλης με μεταβλητές και τους πίνακες Inserted, Deleted.

Αλλάζουμε τη τιμή ενοικίασης του FIAT STILO από 220 σε 300 και του MERCEDES SLK από 77 σε 100 αφού πρώτα πατήσουμε κλικ πάνω στον πίνακα ΑΥΤΟΚΙΝΙΤΑ→Edit Top 200Raws

Ανοίγουμε τον πίνακα AUTO_LOG και παρατηρούμε τις καταχωρήσεις που έκανε αυτόματα, με τη βοήθεια της σκανδάλης :



Εικόνα 9.7 Ενεργοποίηση Σκανδάλης.

Το παράδειγμα αυτό δείχνει το πώς μπορούν να χρησιμοποιηθούν οι σκανδάλες για να υλοποιήσουν ένα ημερολόγιο παρακολούθησης της κίνησης των συναλλαγών των χρηστών πάνω σε έναν πίνακα.

Βασική ιδέα των TRIGGER είναι η διαχείριση των προσωρινών πινάκων INSERTED και DELETED όπου καταχωρούνται προσωρινά οι κάθε αλλαγές που επιχειρεί να κάνει κάποιος χρήστης.

Σε κάθε τροποποίηση της στήλης TIMH_ENOIKIASIS χρησιμοποιώντας την πρόταση UPDATE ενεργοποιείται η σκανδάλη. Έτσι, οι τιμές των γραμμών των εικονικών πινάκων deleted και inserted εκχωρούνται στις αντίστοιχες μεταβλητές @TIMH_old, @TIMH_new και @auto_number. Οι εκχωρημένες τιμές, μαζί με το όνομα χρήστη και την τρέχουσα ημερομηνία και ώρα, εισάγονται κατόπιν στον πίνακα LOG_AUTO.