Διάλεξη 16: SQL DML IV, SQL DCL, SQL-TCL

Στην ενότητα αυτή θα μελετηθούν τα εξής επιμέρους θέματα:
Εισαγωγή στις έννοιες:

- Ενημέρωση Βάσης Δεδομένων (INSERT, UPDATE, DELETE)
- SQL-DCL (GRANT, DENY, REVOKE)
- SQL-TCL (BEGIN, COMMIT, ROLLBACK)

Διδάσκων: Παναγιώτης Ανδρέου

Ενημέρωση Βάσης Δεδομένων

- 3 είδη ενημερώσεων:
  - **INSERT: Εισάγει** καινούριες πλειάδες **σε ένα πίνακα**
  - **DELETE: Διαγράφει** πλειάδες **από ένα πίνακα**
  - **UPDATE: Ενημερώνει** υφιστάμενες πλειάδες **σε ένα πίνακα**
- Στην απλή τους μορφή, αλληλεπιδρούν με ένα πίνακα
  - π.χ., DELETE FROM EMPLOYEES WHERE DNO=5
- Μπορούν να χρησιμοποιηθούν με συνενώσεις για πιο περίπλοκες ενημερώσεις
  - π.χ.,

```
UPDATE      E
SET         E.DNAME = D.DNAME
FROM        EMPLOYEES E INNER JOIN
            DEPARTMENTS D ON E.DNO=D.DNO
```

INSERT – Εισαγωγή Δεδομένων

- Εισάγει πλειάδα(ες) σε ένα πίνακα
- Σύνταξη:
  INSERT INTO TABLE(<column-1>, <column-2>, <column-3>**)**
  VALUES (<value-1>, <value-2>, <value-3>**)**
- Παράδειγμα:
  INSERT INTO PERSON(PID, PNAME, GENDER**)**
  VALUES (5, 'Nikos', 'M'**)**

PERSON (P)		
PID	**PNAME**	**GENDER**
1	Andreas	M
2	Kostas	M
3	Maria	F
4	Eleni	F

PERSON (P)		
PID	**PNAME**	**GENDER**
1	Andreas	M
2	Kostas	M
3	Maria	F
4	Eleni	F
5	**Nikos**	**M**

INSERT (συν.)

- Επισημάνσεις
  - Αν δηλωθεί **INSERT INTO TABLE χωρίς στήλες τότε η SQL υποθέτει ότι στο VALUES θα δοθούν τιμές για όλες τις στήλες με τη σειρά που ορίστηκαν στον πίνακα**
    - Π.χ., **INSERT INTO PERSON** ~~**(**PID, PNAME, GENDER**)**~~
      VALUES (5, 'Nikos', 'M'**)**
    - Αν δεν δοθούν τιμές (π.χ., VALUES (5, 'Nikos')) τότε θα πάρουμε μήνυμα λάθους
  - Μπορούμε να μην εισάγουμε τιμές σε μεμονωμένες στήλες του πίνακα
    - Π.χ., **INSERT INTO PERSON (**PID, PNAME**)**
      VALUES (5, 'Nikos'**)**
    - Αν παραβιάζεται κάποιος περιορισμός (π.χ., CONSTRAINT GENDER NOT NULL) τότε δεν θα επιτραπεί η εισαγωγή

INSERT (συν.)

- Επισημάνσεις
  - T-SQL: ΔΕΝ μπορούμε να εισάγουμε τιμές σε πεδία τύπου **IDENTITY**, αφού η βάση δεδομένων τα θέτει αυτόματα
    - Π.χ., **INSERT INTO PERSON (**PID, PNAME**)** **--PID IDENTITY**
      →ERROR
    - Θα πρέπει να ΜΗΝ χρησιμοποιήσουμε το πεδίο τύπου IDENTITY
    - Δηλ., **INSERT INTO PERSON (**PNAME**)**
      VALUES ('Andreas')
  - Αν θέλουμε να απενεργοποιήσουμε προσωρινά το IDENTITY τότε μπορούμε να χρησιμοποιήσουμε την εντολή
    SET IDENTITY_INSERT [ON|OFF]
    - Π.χ., **SET IDENTITY_INSERT PERSON ON**
      INSERT INTO PERSON VALUES(1,'Andreas', 'M')
      SET IDENTITY_INSERT PERSON OFF

INSERT με SELECT

- Το INSERT χρησιμοποιείται για την εισαγωγή μίας-μίας πλειάδας
  - Εξαίρεση: TSQL 2008 προσφέρει την δυνατότητα εισαγωγής πολλαπλών πλειάδων με μια εντολή INSERT, π.χ., INSERT INTO EMPLOYEE **VALUES (1,'Andreas','M'), (2,'Eleni','F'), …;**
- Για να εισάγουμε πολλαπλές πλειάδες με το INSERT, μπορούμε να κάνουμε χρήση του SELECT:
  - Εισαγωγή δεδομένων από άλλο πίνακα (INSERT SELECT)

    Παράδειγμα:
    ```
    INSERT INTO PERSON
    SELECT EID, ENAME, GENDER
    FROM EMPLOYEE
    ```

    Εισαγωγή δεδομένων από διάφορες πηγές (INSERT SELECT UNION SELECT)

    Παράδειγμα:
    ```
    INSERT INTO PERSON
    SELECT 1, 'Andreas', 'M'
    UNION
    SELECT 2, 'Eleni', 'F'
    ```

DELETE – Διαγραφή Δεδομένων

- Η εντολή **DELETE** διαγράφει πλειάδες από μια σχέση βάσει κάποιας ορισμένης συνθήκης:

DELETE [FROM] <table-name>

[WHERE <condition>]

- Π.χ., DELETE FROM EMPLOYEE where Dno=5;
- **Επισημάνσεις**

1. Εάν δεν οριστεί ο όρος WHERE, τότε σβήνονται **ΟΛΑ** ολα τα δεδομένα (πλειάδες) μιας σχέσης.
    - Π.χ., «DELETE FROM EMPLOYEE» (αντίστοιχο αποτέλεσμα με την TRUNCATE EMPLOYEE μόνο που θα καταγράφεται ένα log record για κάθε διαγραφή)
    - Το **DROP** από την άλλη σβήνει τόσο τα **δεδομένα** όσο και το σχήμα της βάσης από τον **κατάλογο του συστήματος**.

DELETE (συν.)

- **Επισημάνσεις**
  2. Οι **κανόνες αναφορικής ακεραιότητας** επιβάλλονται από την βάση κατά την διαγραφή
     - π.χ., το σύστημα απαγορεύει την διαγραφή ενός **EMPLOYEE** που αναφέρεται από την σχέση **WORKS_ON**.
  3. Δεν υπάρχει η έννοια της **διαγραφής** από **πολλαπλούς πίνακες.** Κάθε διαγραφή **αναφέρεται σε 1 πίνακα**

U4A: DELETE FROM EMPLOYEE
WHERE LNAME='Brown’

- Ωστόσο, εάν ορίζονται **Εντολές Ενεργοποίησης Αναφοράς**, τότε μια διαγραφή μπορεί να προκαλέσει μια ή περισσότερες αλυσιδωτές διαγραφές
  - π.χ., εάν έχουμε **ON DELETE CASCADE** στην **WORKS_ON(ssn) (με αναφορά στον EMPLOYEE(ssn)) τότε** η διαγραφή ενός **EMPLOYEE** σβήνει και την αντίστοιχη πλειάδα από την **WORKS_ON**.

UPDATE – Ενημέρωση Δεδομένων

- Η εντολή **UPDATE** χρησιμοποιείται για να **ενημερώνει** την τιμή **προκαθορισμένων γνωρισμάτων** μιας **σχέσης** βάσει κάποιας συνθήκης.

  UPDATE <table-name>

  SET <column>=<value> [,<column>=<value>]

  [WHERE <condition>]

- Παράδειγμα:

UPDATE PERSON
SET GENDER='F'
WHERE PID=3

PERSON (P)		
PID	**PNAME**	**GENDER**
1	Andreas	M
2	Kostas	M
3	**Maria**	**M**

PERSON (P)		
PID	**PNAME**	**GENDER**
1	Andreas	M
2	Kostas	M
3	**Maria**	**F**

UPDATE (συν.)

Επισημάνσεις:

- Το **WHERE** χρησιμοποιείται για να **προσδιοριστεί το σύνολο** εγγραφών που πρέπει να **ενημερωθεί**
  - Μπορεί να συνδυασθεί με συνενώσεις για περίπλοκες ενημερώσεις με πολλαπλούς πίνακες

```
Π.χ.,   UPDATE    E
        SET       E.DNAME = D.DNAME
        FROM      EMPLOYEES E INNER JOIN
                  DEPARTMENTS D ON E.DNO=D.DNO
```

- Το **SET** προσδιορίζει τα **γνωρίσματα** που πρέπει να ενημερωθούν και την νέα τους τιμή.
- Η ενημέρωση γίνεται μόνο σε ένα πίνακα
- **Οι κανόνες ακεραιότητας επιβάλλονται αυτόματα**

Χρήσιμες Εντολές – BULK INSERT

- Μια εξαιρετικά χρήσιμη εντολή είναι η **BULK INSERT** η οποία εισάγει **μαζικά** δεδομένα από αρχεία κειμένου (αντίστοιχα η **BCP (windows command line utility)** εξάγει μαζικά δεδομένα):
  - Π.χ., bcp "SELECT FirstName, LastName FROM TABLE ORDER BY LastName" queryout Contacts.txt -c -T
- **Στον SQL Server:**
  - **BULK INSERT** OrdersBulk **FROM** 'c:\data.csv'

    WITH (FIRSTROW=2, FIELDTERMINATOR=',', ROWTERMINATOR='\n')

  Υποθέστε ότι έχουν προηγηθεί τα ακόλουθα:
  - **CREATE TABLE** OrdersBulk(CustomerID INT, CustomerName VARCHAR(32), OrderID INT, OrderDate SMALLDATETIME)
  - Το **data.csv** θεωρήστε ότι έχει την ακόλουθη δομή:

    # CustomerID,CustomerName,OrderID,OrderDate

    1,foo,5,20031101

    3,blat,7,20031101

    5,foobar,23,20031104

Χρήσιμες Εντολές - OPENROWSET

- Η OPENROWSET επιτρέπει να αντλήσουμε δεδομένα από μια άλλη πηγή (π.χ., txt αρχείο, άλλη OLE-DB βάση δεδομένων, κτλ)
- Παράδειγμα join με πίνακα άλλης βάσης

```
USE Northwind;
GO
SELECT c.*, o.*
FROM Northwind.dbo.Customers AS c
  INNER JOIN OPENROWSET('Microsoft.Jet.OLEDB.4.0',
  'C:\Program Files\Microsoft
   Office\OFFICE11\SAMPLES\Northwind.mdb';'admin';'', Orders)
  AS o
  ON c.CustomerID = o.CustomerID ;
GO
```

Γλώσσα Ορισμού Ελέγχων (DCL)

- **GRANT**: Δίνει πρόσβαση σε κάποιο αντικείμενο της βάσης δεδομένων σε κάποιους χρήστες ή ρόλους

```
GRANT     [INSERT| UPDATE | DELETE]
ON        [TABLE | PROCEDURE | <object>
TO        [USER | ROLE];
```

- **DENY**: Απαγορεύει πρόσβαση σε κάποιο αντικείμενο της βάσης δεδομένων σε κάποιους χρήστες ή ρόλους

```
DENY      [INSERT| UPDATE | DELETE]
ON        [TABLE | PROCEDURE | <object>
TO        [USER | ROLE];
```

- **REVOKE**: Αναστέλλει την τελευταία εντολή πρόσβασης ή απαγόρευσης

Γλώσσα Χειρισμού Εντολών (TCL)

- Ομαδοποίηση εντολών (COMMIT) και εκτέλεση τους σαν μία εντολή. Σε περίπτωση αποτυχίας, μπορούν να αναιρεθούν όλες οι εντολές (ROLLBACK).
- Παράδειγμα σε SQL-TCL:

BEGIN TRANSACTION

INSERT INTO STUDENTS(ID, NAME) VALUES(13, Andreas)

INSERT INTO STUDENT_INTERESTS(ID, INTEREST) VALUES(13, 'Football')

INSERT INTO STUDENT_INTERESTS(ID, INTEREST) VALUES(13, 'Rugby')

COMMIT TRANSACTION

IF @@ERROR<>0

ROLLBACK TRANSACTION