# Βάσεις Δεδομένων – 6

## SQL – A Relational Database Language – Μέρος Β

Δρ. Χαράλαμπος Καρανίκας

# Επεξεργασία δεδομένων με SQL

# SQL Data Manipulation Language

UPDATE, DELETE, LIMIT,
MIN(), MAX() COUNT(), AVG() και SUM() Συναρτήσεις
LIKE, IN, BETWEEN, AS (Aliases), JOIN

# UPDATE Statement

- Η εντολή UPDATE χρησιμοποιείται για να τροποποιήσει τις υπάρχουσες εγγραφές σε έναν πίνακα.
- Προσοχή κατά την ενημέρωση των εγγραφών σε έναν πίνακα: Το **WHERE** καθορίζει τις εγγραφές που πρέπει να ενημερωθούν. Αν παραλείψετε το WHERE, όλες οι εγγραφές στον πίνακα θα ενημερωθούν.

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

```
UPDATE Customers
SET ContactName = 'Alfred Schmidt', City= 'Frankfurt'
WHERE CustomerID = 1;

UPDATE Customers
SET ContactName='Juan'
WHERE Country='Mexico';
```

# DELETE Statement

- Η εντολή DELETE χρησιμοποιείται για τη διαγραφή υπαρχόντων εγγραφών σε έναν πίνακα.
- Το **WHERE** καθορίζει τις εγγραφές που πρέπει να διαγραφούν. Εάν παραλείψετε το WHERE, όλες οι εγγραφές στον πίνακα θα διαγραφούν.

```
DELETE FROM table_name
WHERE condition;
```

```
DELETE FROM Customers
WHERE CustomerName='Alfreds Futterkiste';
```

- Είναι δυνατή η διαγραφή όλων των γραμμών σε έναν πίνακα χωρίς διαγραφή του πίνακα. Αυτό σημαίνει ότι η δομή του πίνακα, τα χαρακτηριστικά και τα ευρετήρια θα παραμείνουν.

```
DELETE FROM table_name;
```

# LIMIT

- Η MySQL υποστηρίζει το LIMIT για να επιλέξετε έναν περιορισμένο αριθμό εγγραφών.
- Είναι χρήσιμο σε μεγάλους πίνακες με χιλιάδες εγγραφές. Η επιστροφή μεγάλου αριθμού εγγραφών μπορεί να επηρεάσει την απόδοση.

```
SELECT column_name(s)
FROM table_name
WHERE condition
LIMIT number;
```

```
SELECT * FROM Customers
LIMIT 3;
```

# MIN() και MAX() Συναρτήσεις

- Η συνάρτηση **MIN ()** επιστρέφει τη μικρότερη τιμή της επιλεγμένης στήλης και η **MAX()** αντίστοιχα την μεγαλύτερη.

```
SELECT MIN(column_name) ή MAX(column_name)
FROM table_name
WHERE condition;
```

```
SELECT MIN(Price) AS SmallestPrice
FROM Products;

SELECT MAX(Price) AS LargestPrice
FROM Products;
```

# COUNT(), AVG() και SUM() Συναρτήσεις

- Η συνάρτηση COUNT () επιστρέφει τον αριθμό των εγγραφών που αντιστοιχούν σε συγκεκριμένα κριτήρια.

```
SELECT COUNT(column_name)
FROM table_name
WHERE condition;
```

- Η συνάρτηση AVG () επιστρέφει τη μέση τιμή μιας αριθμητικής στήλης.

```
SELECT AVG(column_name)
FROM table_name
WHERE condition;
```

- Η συνάρτηση SUM () επιστρέφει το συνολικό άθροισμα μιας αριθμητικής στήλης.

```
SELECT SUM(column_name)
FROM table_name
WHERE condition;
```

# Τελεστής LIKE 1/3

- Ο τελεστής LIKE χρησιμοποιείται στο WHERE για να αναζητήσετε ένα συγκεκριμένο πρότυπο (pattern) σε μια στήλη.
- Σε συνδυασμό με το LIKE χρησιμοποιούνται:
  - **«%»** Το σύμβολο ποσοστού αντιπροσωπεύει μηδέν, ένα ή πολλαπλούς χαρακτήρες
  - **«_»** Η κάτω παύλα αντιπροσωπεύει ένα μοναδικό χαρακτήρα
- Μπορείτε επίσης να συνδυάσετε οποιονδήποτε αριθμό συνθηκών χρησιμοποιώντας τους τελεστές AND ή OR.

```
SELECT column1, column2, ...
FROM table_name
WHERE columnN LIKE pattern;
```

# Τελεστής LIKE – Παραδείγματα 2/3

- WHERE CustomerName LIKE 'a%' : Βρίσκει οποιεσδήποτε τιμές ξεκινούν με "a"
- WHERE CustomerName LIKE '%a' : Βρίσκει τιμές που τελειώνουν με "a"
- WHERE CustomerName LIKE '%or%' : Βρίσκει οποιεσδήποτε τιμές έχουν "or" σε οποιαδήποτε θέση
- WHERE CustomerName LIKE '_r%' : Βρίσκει οποιεσδήποτε τιμές που έχουν "r" στη δεύτερη θέση
- WHERE CustomerName LIKE 'a_%_%' : Βρίσκει οποιεσδήποτε τιμές που ξεκινούν με "a" και έχουν μήκος τουλάχιστον 3 χαρακτήρων
- WHERE ContactName LIKE 'a%o' : Βρίσκει οποιεσδήποτε τιμές ξεκινούν με "a" και τελειώνουν με "o"

# Τελεστής LIKE - Παραδείγματα 3/3

```sql
SELECT * FROM Customers
WHERE CustomerName LIKE 'a%';

SELECT * FROM Customers
WHERE CustomerName LIKE '%a';

SELECT * FROM Customers
WHERE CustomerName LIKE '%or%';

SELECT * FROM Customers
WHERE CustomerName LIKE '_r%';

SELECT * FROM Customers
WHERE CustomerName LIKE 'a_%_%';

SELECT * FROM Customers
WHERE ContactName LIKE 'a%o';
```

```sql
SELECT * FROM Customers
WHERE CustomerName NOT LIKE 'a%';
```

# Τελεστής IN

- Ο τελεστής IN σας επιτρέπει να καθορίσετε πολλαπλές τιμές στο WHERE.
- Ουσιαστικά είναι συντομογραφία για πολλαπλές συνθήκες OR.

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (value1, value2, ...);
```

```
SELECT column_name(s)
FROM table_name
WHERE column_name IN (SELECT STATEMENT);
```

```
SELECT * FROM Customers
WHERE Country IN ('Germany', 'France', 'UK');
```

```
SELECT * FROM Customers
WHERE Country NOT IN ('Germany', 'France', 'UK');
```

```
SELECT * FROM Customers
WHERE Country IN (SELECT Country FROM Suppliers);
```

# Τελεστής BETWEEN 1/2

- Ο τελεστής BETWEEN επιλέγει τιμές μέσα σε ένα δεδομένο εύρος τιμών. Οι τιμές μπορεί να είναι αριθμοί, κείμενο ή ημερομηνίες.

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

```
SELECT * FROM Products
WHERE Price BETWEEN 10 AND 20;
```

```
SELECT * FROM Products
WHERE Price NOT BETWEEN 10 AND 20;
```

```
SELECT * FROM Products
WHERE (Price BETWEEN 10 AND 20)
AND NOT CategoryID IN (1,2,3);
```

# Τελεστής BETWEEN 2/2

```
SELECT * FROM Products
WHERE ProductName BETWEEN 'Carnarvon Tigers' AND 'Mozzarella di Giovanni'
ORDER BY ProductName;
```

```
SELECT * FROM Orders
WHERE OrderDate BETWEEN #07/04/1996# AND #07/09/1996#;
```

# SQL AS (Aliases) 1/2

- Χρησιμοποιούνται για να δώσουν σε ένα πίνακα ή μια στήλη ενός πίνακα ένα προσωρινό όνομα και υπάρχουν μόνο για τη διάρκεια του ερωτήματος.
- Μπορεί να είναι χρήσιμα όταν:
  - Υπάρχουν περισσότεροι από ένας πίνακας που εμπλέκονται στο ερώτημα
  - Συναρτήσεις χρησιμοποιούνται στο ερώτημα
  - Τα ονόματα των στηλών είναι μεγάλα ή δεν είναι πολύ ευανάγνωστα
  - Δύο ή περισσότερες στήλες συνδυάζονται μαζί

```
SELECT column_name AS alias_name
FROM table_name;
```

```
SELECT column_name(s)
FROM table_name AS alias_name;
```

# SQL AS (Aliases) – Παραδείγματα 2/2

```
SELECT CustomerID as ID, CustomerName AS Customer
FROM Customers;

SELECT CustomerName AS Customer, ContactName AS [Contact Person]
FROM Customers;

SELECT CustomerName, CONCAT(Address,', ',PostalCode,', ',City,',
',Country) AS Address
FROM Customers;
```

```
SELECT o.OrderID, o.OrderDate, c.CustomerName
FROM Customers AS c, Orders AS o
WHERE c.CustomerName="Around the Horn" AND c.CustomerID=o.CustomerID;
```

```
SELECT Orders.OrderID, Orders.OrderDate, Customers.CustomerName
FROM Customers, Orders
WHERE Customers.CustomerName="Around the
Horn" AND Customers.CustomerID=Orders.CustomerID;
```

# SQL JOIN 1/2

- Το JOIN χρησιμοποιείται για να συνδυάσει σειρές (εγγραφές) από δύο ή περισσότερους πίνακες, με βάση μια «σχετική» στήλη μεταξύ τους.

| OrderID | CustomerID | OrderDate |
|---|---|---|
| 10308 | 2 | 1996-09-18 |
| 10309 | 37 | 1996-09-19 |
| 10310 | 77 | 1996-09-20 |

| CustomerID | CustomerName | ContactName | Country |
|---|---|---|---|
| 1 | Alfreds Futterkiste | Maria Anders | Germany |
| 2 | Ana Trujillo Emparedados y helados | Ana Trujillo | Mexico |
| 3 | Antonio Moreno Taquería | Antonio Moreno | Mexico |

- Η στήλη "CustomerID" στον πίνακα "Orders" αναφέρεται στο "CustomerID" στον πίνακα "Customers". Η συσχέτιση μεταξύ των δύο παραπάνω πινάκων είναι η στήλη "CustomerID".

# SQL JOIN 2/2

| OrderID | CustomerID | OrderDate |
|---|---|---|
| 10308 | 2 | 1996-09-18 |
| 10309 | 37 | 1996-09-19 |
| 10310 | 77 | 1996-09-20 |

| CustomerID | CustomerName | ContactName | Country |
|---|---|---|---|
| 1 | Alfreds Futterkiste | Maria Anders | Germany |
| 2 | Ana Trujillo Emparedados y helados | Ana Trujillo | Mexico |
| 3 | Antonio Moreno Taquería | Antonio Moreno | Mexico |

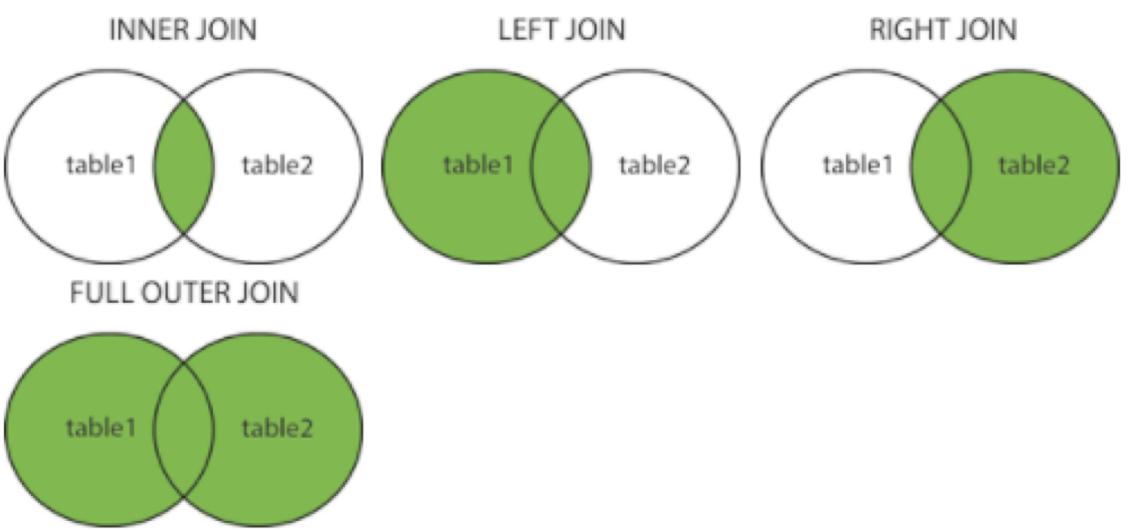
```
SELECT Orders.OrderID, Customers.CustomerName, Orders.OrderDate
FROM Orders
INNER JOIN Customers ON Orders.CustomerID=Customers.CustomerID;
```

- Και θα παράγει κάτι σαν αυτό:

| OrderID | CustomerName | OrderDate |
|---|---|---|
| 10308 | Ana Trujillo Emparedados y helados | 9/18/1996 |
| 10365 | Antonio Moreno Taquería | 11/27/1996 |
| 10383 | Around the Horn | 12/16/1996 |
| 10355 | Around the Horn | 11/15/1996 |
| 10278 | Berglunds snabbköp | 8/12/1996 |

# Διαφορετικοί τύποι SQL JOIN

- **(INNER) JOIN**: Επιστρέφει εγγραφές που έχουν αντίστοιχες τιμές και στους δύο πίνακες
- **LEFT (OUTER) JOIN**: Επιστρέφει όλες τις εγγραφές από τον αριστερό πίνακα και τις αντίστοιχες εγγραφές από το δεξιό πίνακα. Το αποτέλεσμα είναι NULL από τη δεξιά πλευρά, αν δεν υπάρχει αντιστοιχία.
- **RIGHT (OUTER) JOIN**:Επιστρέφει όλες τις εγγραφές από το δεξιό πίνακα και τις αντίστοιχες εγγραφές από τον αριστερό πίνακα
- **FULL (OUTER) JOIN:** Επιστρέφει όλες τις εγγραφές όταν υπάρχει αντιστοιχία στον αριστερό ή στον δεξί πίνακα



# JOIN με τρεις πίνακες

- Η ακόλουθη εντολή SQL επιλέγει όλες τις παραγγελίες με πληροφορίες πελατών και αποστολέα:

```
SELECT Orders.OrderID, Customers.CustomerName, Shippers.ShipperName
FROM ((Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID)
INNER JOIN Shippers ON Orders.ShipperID = Shippers.ShipperID);
```