Department of Computer Science
University of Cyprus

EPL342 –Databases

Lecture 3: Introduction III

System Concepts and Architecture

(Chapters 2.2-2.7, Elmasri-Navathe 5ED)

Διδάσκων: Παναγιώτης Ανδρέου

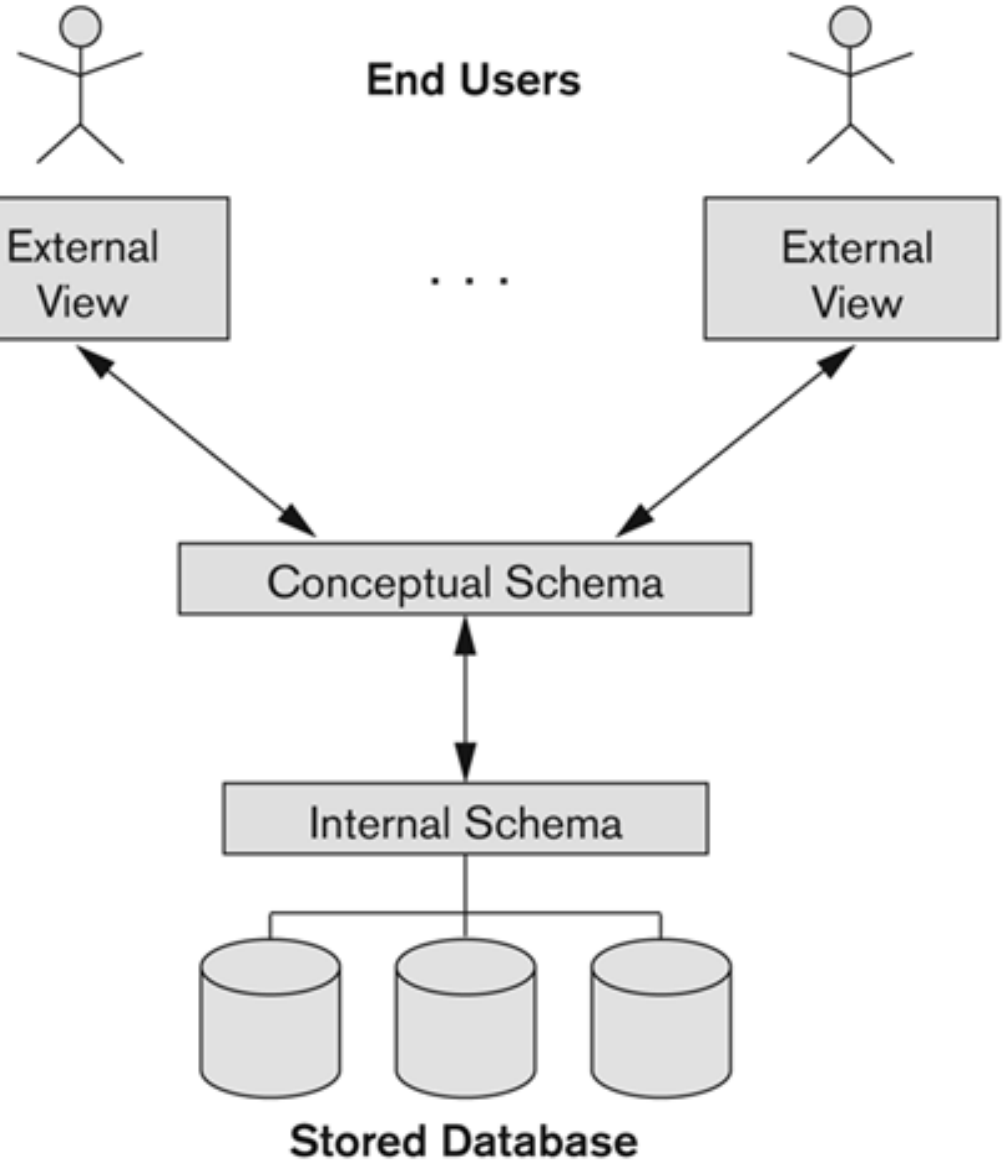
http://www.cs.ucy.ac.cy/courses/EPL342

Περιεχόμενο Διάλεξης

Κεφάλαιο 2

- 2.2) Αρχιτεκτονική **Τριών Σχημάτων** και **Ανεξαρτησία Δεδομένων**
- 2.3) **Γλώσσες** και **Διεπαφές** Βάσεων Δεδομένων
- 2.4-2.6) **Περιβάλλον & Αρχιτεκτονικές Συστημάτων Διαχ. Βάσεων Δεδ.**
  - **Διασύνδεση Συσκευών Αποθήκευσης**
  - **Συγκεντρωτικές**
  - **Πελάτη-Εξυπηρετητή (2-επιπεδων & N-επιπέδων)**
  - **Κατανεμημένες**

Αρχιτεκτονική Τριών-Σχημάτων



Εξωτερικό Επίπεδο

Εννοιολογικό Επίπεδο

Εσωτερικό Επίπεδο

Αρχιτεκτονική Τριών-Σχημάτων

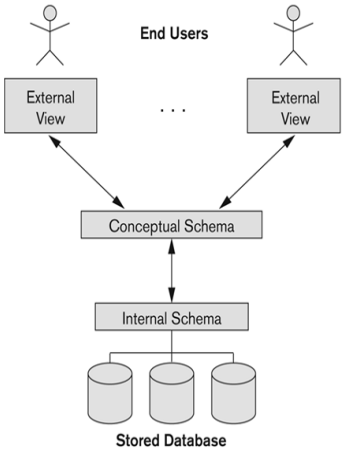
- **Αρχιτεκτονική Τριών Σχημάτων (Three-Schema Architecture):** Μια **αφαιρετική** αρχιτεκτονική ΣΔΒΔ (DBMS) η οποία αποσκοπεί στο να **διαχωρίσει** τις **εφαρμογές χρηστών** από τη **φυσική αποθήκευση** των δεδομένων, δηλ.
  - **Της Ανεξαρτησίας Προγράμματος-Δεδομένων.**
  - Της Υποστήριξης Πολλαπλών **Όψεων**.
- Μια τέτοια αρχιτεκτονική **δεν χρησιμοποιείται** ρητά σε κάθε εμπορική DBMS, αλλά είναι χρήσιμη για να επεξηγηθεί η οργάνωση βάσεων δεδομένων

Αρχιτεκτονική Τριών-Σχημάτων

- Η Αρχιτεκτονική αυτή ορίζει **Σχήματα (Schemas)** σε τρία διαφορετικά επίπεδα:
  - **Εσωτερικά Σχήματα:** Για περιγραφή των ***φυσικών δομών αποθήκευσης*** *(storage structures)* και των ***δομών ανάκτησης (access paths)***
    - Χρήση **Φυσικού Μοντέλου Δεδομένων** π.χ., τα δεδομένα είναι οργανωμένα σε αρχεία σωρών, ταξινομημένα αρχεία, κτλ.
  - **Εννοιολογικά Σχήματα:** Για περιγραφή της **δομής** και των **περιορισμών** σε μορφή **κατανοητή στους χρήστες**
    - Χρήση **Εννοιολογικού (Conceptual)** ή **Αναπαραστατικού (Representational)** Μοντέλου Δεδομένων.
  - **Εξωτερικά Σχήματα:** Για περιγραφή των **όψεων** που βλέπουν οι **χρήστες**.
    - Και πάλι χρήση εννοιολογικού σχήματος

Ανεξαρτησία Δεδομένων

- **Λογική Ανεξαρτησία Δεδομένων (Logical Data Independence)**
  - Η δυνατότητα αλλαγής του **εννοιολογικού σχήματος** χωρίς να απαιτείται να αλλάξει το **εξωτερικό σχήμα** (αυτό που βλέπει ο χρήστης)
- **Φυσική Ανεξαρτησία Δεδομένων (Physical Data Independence)**
  - Η δυνατότητα αλλαγής του **εσωτερικού σχήματος** χωρίς να απαιτείται να αλλάξει το **εννοιολογικό σχήμα**.



- **Παράδειγμα:** Θεωρήστε ότι οι εγγραφές της οντότητας **STUDENTS** είναι αποθηκευμένες στο δίσκο σε **τυχαία διάταξη**

Εάν προστεθεί ένα **ευρετήριο (index)** το οποίο μας επιτρέπει να βρίσκουμε γρηγορότερα τους φοιτητές βάσει της **ηλικίας** τους τότε θα έχει **αλλάξει το φυσικό σχήμα.**

Γλώσσες Βάσεων Δεδομένων

- Όταν ολοκληρωθεί η φάση της **εννοιολογικής μοντελοποίησης** των απαιτήσεων του χρήστη τότε ένας DBA ή DB Designer προχωρεί στην υλοποίηση της βάσης δεδομένων με τα ακόλουθα:

Α) Γλώσσα Ορισμού (Δομής) Δεδομένων

  - **Data Definition Language (DDL),**
  - Χρησιμοποιείται από τον DBA και τον DB Σχεδιαστή για τον ορισμό του **Εννοιολογικού Σχήματος** (**Αναπαραστατικού Μοντέλου**) μιας βάσης.

Β) Γλώσσα Χειρισμού (Επεξεργασίας) Δεδομένων

  - **Data Manipulation Language (DML), π.χ,,**
  - Χρησιμοποιείται στον ορισμό **ανακτήσεων (retrievals)** και **ενημερώσεων (updates)**

Γλώσσες Βάσεων Δεδομένων

- **Α) Γλώσσα Ορισμού Δεδομένων [Data Definition Language (DDL)]:**
  - Παράδειγμα σε SQL-DDL*:

```
CREATE TABLE products (
        product_no integer,
        name text,
        price numeric );
```

** Δημιουργεί ένα πίνακα products με 3 πεδία (γνωρίσματα)*

  - Σε πολλές DBMSs, το DDL χρησιμοποιείται επίσης για τον ορισμό των **εσωτερικών** (π.χ., indexes) και **εξωτερικών σχημάτων** (π.χ., views).

Π.χ., **CREATE VIEW** expensive_products AS

```
SELECT name, price
FROM products
WHERE price>100;
```

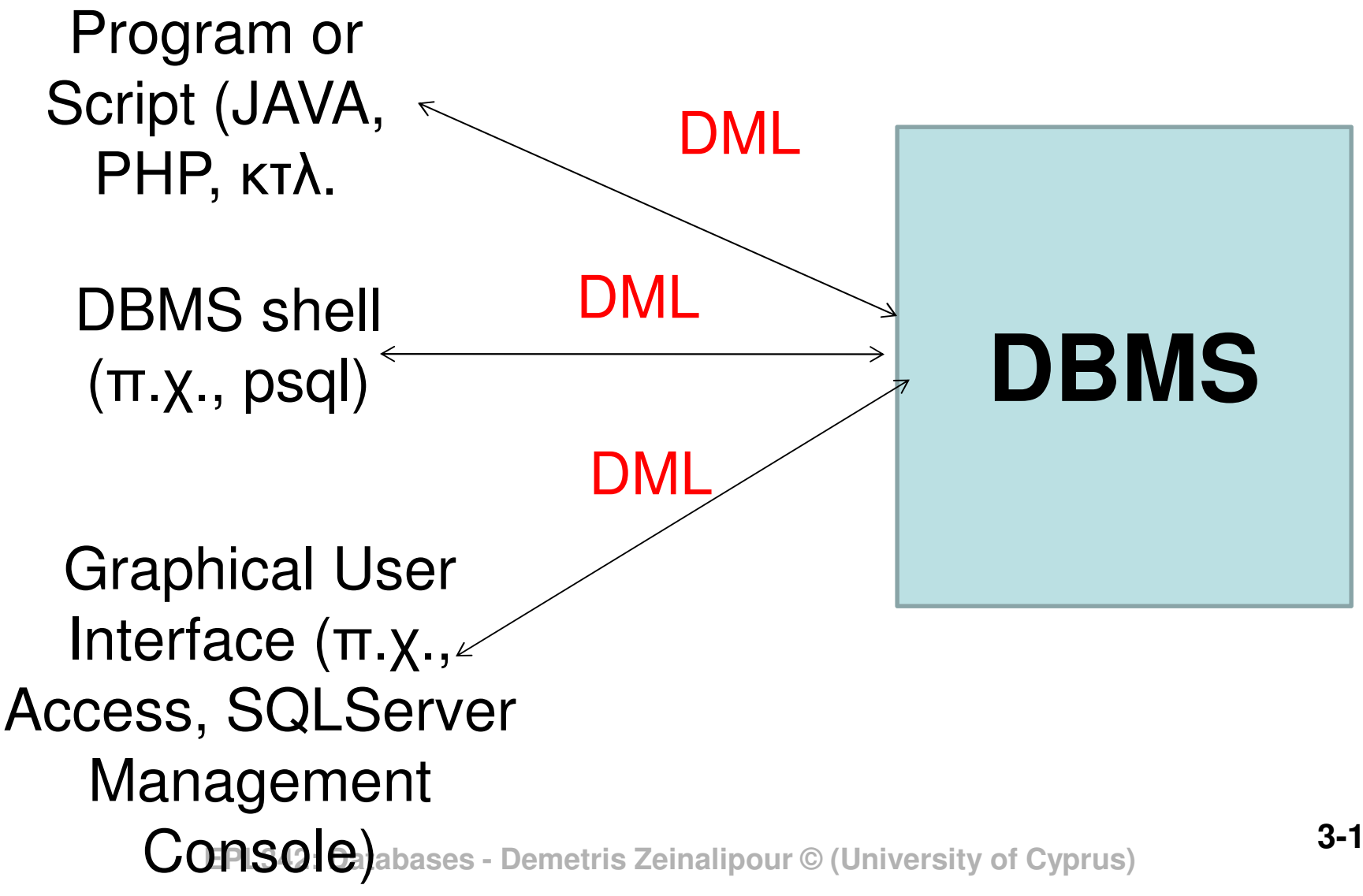
** Δημιουργεί ένα νοητό πίνακα που περιλαμβάνει μόνο τα ακριβά προϊόντα*

Γλώσσες Βάσεων Δεδομένων

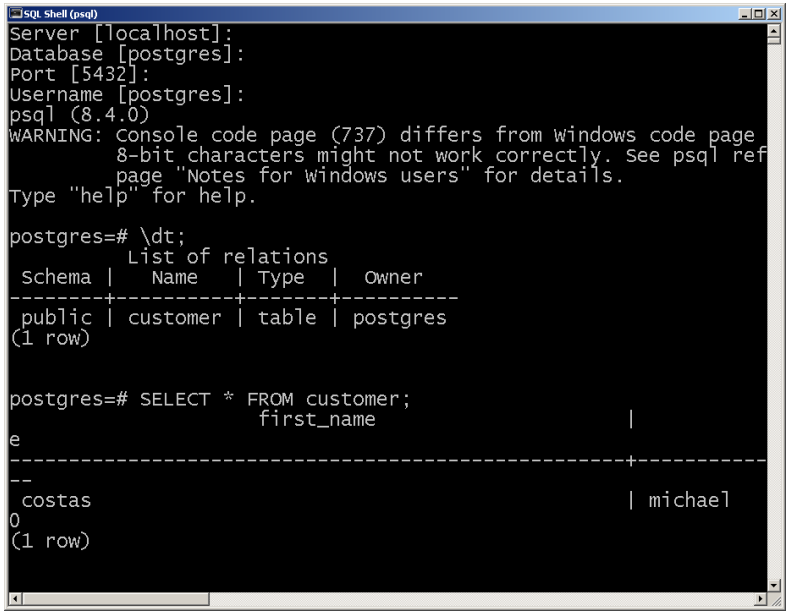
- **Γλώσσα Χειρισμού Δεδομένων [Data Manipulation Language (DML)]:**
  - Παράδειγμα SQL-DML*:
    - SELECT * FROM products;

    * Επιστρέφει όλα τα προϊόντα στον πίνακα products;
  - Οι εντολές DML (υπόγλωσσα δεδομένων) μπορούν να ενσωματωθούν σε μια **γλώσσα προγραμματισμού** (π.χ., C, C++, C#, Java, κτλ.).
  - Εναλλακτικά, μπορούμε να εκτελέσουμε τέτοιες εντολές απευθείας από τη γραμμή εντολών κάποιου **κελύφους SQL (psql/PostgreSQL, *SQL**Plus/Oracle, SQLCMD/SQLServer κτλ) ή** ακόμα και μέσω γραφικού περιβάλλοντος διαπροσωπείας

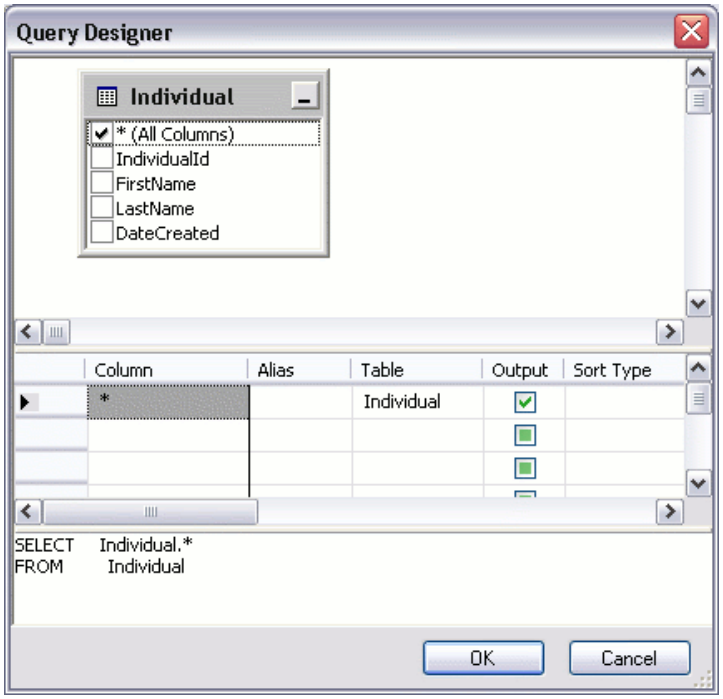
Γλώσσες Βάσεων Δεδομένων



Γλώσσες Βάσεων Δεδομένων



To psql (PostgreSQL) DBMS shell



SQL Query Builder στον SQLServer Management Studio

Τύποι DML

- **Υψηλού Επιπέδου ή Μη-διαδικαστικές (δηλωτικές):**
  - Π.χ., η SQL είναι μια τέτοια γλώσσα.
  - **Συνολοστρεφείς ("set"-oriented)**
    - *Επιστρέφουν τα αποτελέσματα σε σύνολα και όχι μια-πλειάδα-κάθε-φορά (record-at-a-time)*
  - **Δηλωτικές (declarative)**
    - Ο χρήστης δηλώνει **ΤΙ** θέλει να ανακτήσει και όχι **ΠΩΣ** θέλει να το ανακτήσει., Π.χ., SELECT * FROM product;
- **Χαμηλού Επιπέδου ή Διαδικαστικές (imperative):**
  - **Εγγραφοστρεφείς ("tuple"-oriented)**
    - Ανακτούν τα δεδομένα μια-εγγραφή-κάθε-φορά (record-at-a-time)
  - Παρέχουν βρόχους επανάληψης, εντολές επιλογής, λογικές παραστάσεις, κτλ μαζί με δείκτες πλοήγησης (cursors).
  - Προεκτάσεις της SQL (δηλ., η SQL3) παρέχουν τέτοιες δυνατότητες
    - Π.χ., ANSI **SQL/PSM (Persistent Stored Modules)**, **PL/SQL (Oracle)**, **TSQL (Microsoft SQL Server)**, **SPL (Informix)**, **PL/pgPSM (PostgreSQL)**

Τύποι DML

- Παράδειγμα Χαμηλού Επιπέδου (Διαδικαστικής) SQL (με χρήση του PL/pgPSM στη PostgreSQL)

  ** Δεν χρειάζεται να καταλάβετε την ακριβή λειτουργία του πιο κάτω κώδικα*

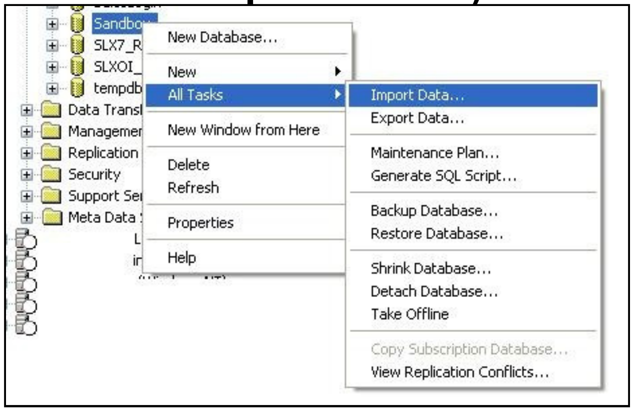
```
CREATE OR REPLACE FUNCTION hello(uid integer)
RETURNS varchar AS
$$
  BEGIN
    DECLARE real_name varchar;
    -- Get real name
    SET real_name = (SELECT name
                     FROM Users
                     WHERE Users.uid = hello.uid);
    RETURN 'Hello, ' || real_name;
  END;
$$ LANGUAGE plpgsql;

SELECT hello(123);
```

Ένα τέτοιο πρόγραμμα θα καταχωρείτω απευθείας στη βάση δεδομένων

Η Εκτέλεση του θα γινόταν με αποστολή μιας εντολής SQL, π.χ.,
SELECT hello(123);

Εργαλεία Ωφελιμότητας Βάσεων (Database System Utilities)

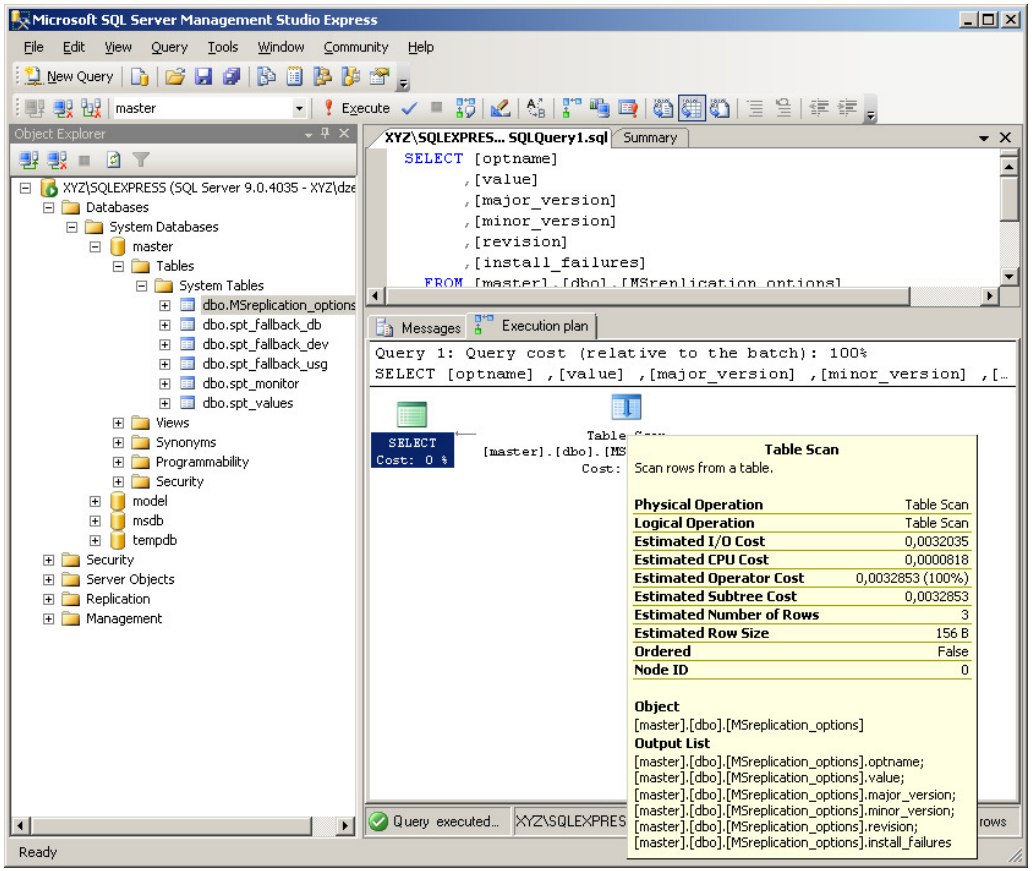
- Τέτοια εργαλεία συνοδεύουν τα εμπορικά και μη ΣΔΒΔ, και εκτελούν λειτουργίες όπως:
  - **Φόρτωση Δεδομένων** (συμπεριλαμβανομένου και της **μετατροπής** μεταξύ διαφορετικών τύπων βάσεων).
  - **Αντιγραφή/Επαναφορά (Backup/Restore)**
  - **Αναδιοργάνωση Αρχείων, κτλ.**
    - **(π.χ., Shrink, Detach)**
  - **Κτλ..**



System Utilities στον SQL Server

  - Πολλά **επιπλέον utilities** προσφέρονται ως **επεκτάσεις** σε μια DBMS (π.χ., SQL Server Management Studio). Τέτοιες προεκτάσεις συμπερ.: Αναφορές, Έλεγχος Επίδοσης, Έλεγχος Χρηστών, Συμπίεση Πινάκων, κτλ)

Εργαλεία Ωφελιμότητας Βάσεων (Database System Utilities)



Ανάλυση Εκτέλεσης Επερωτήσεων στον SQL Server 2008 (SQL Server Management Studio)

Αρχιτεκτονικές DBMS

Θα μελετήσουμε τρία επί μέρους θέματα:

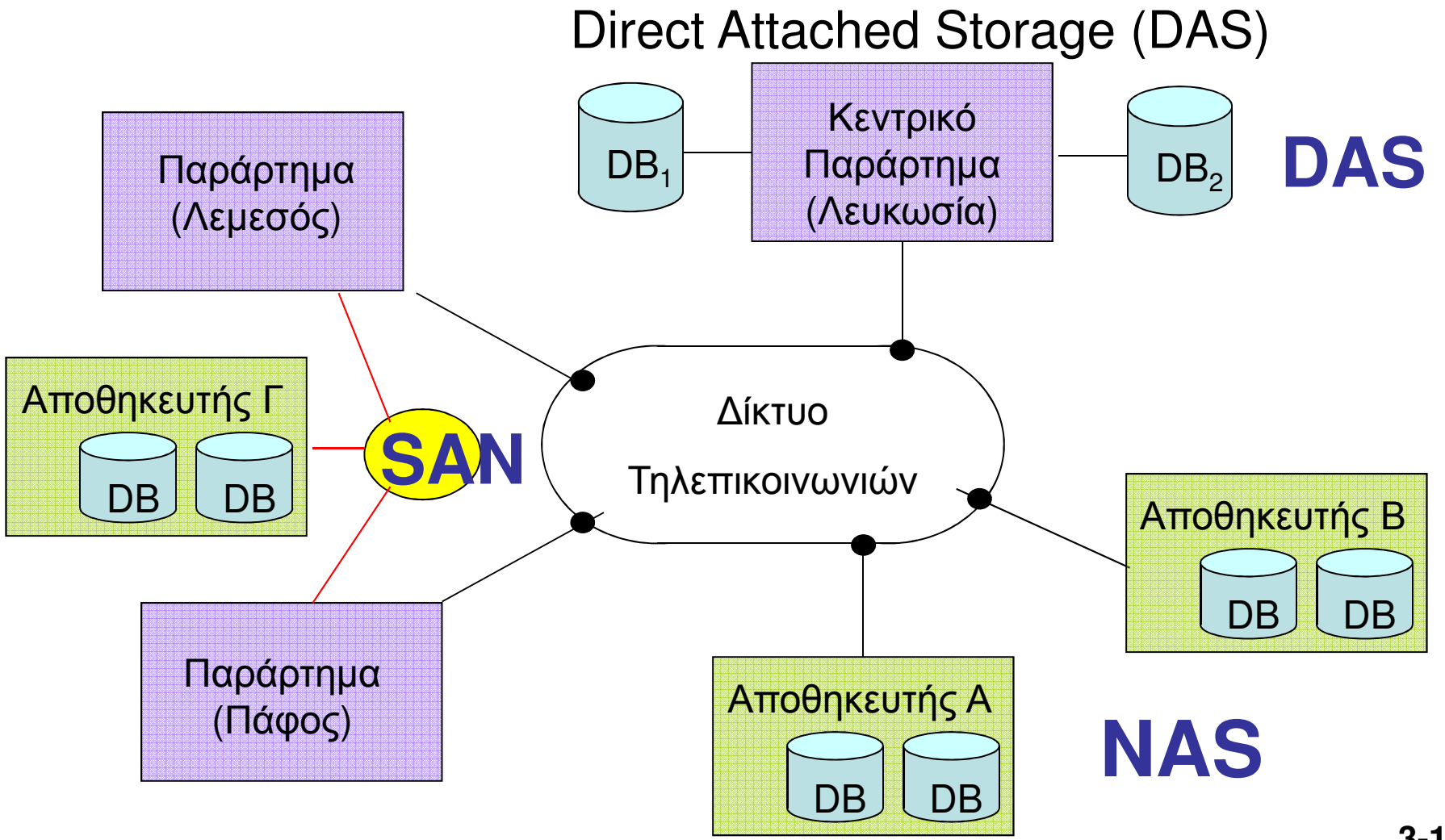
1. Διασύνδεση Συσκευών Αποθήκευσης (Δίσκων)
2. Διασύνδεση Πελάτη και DBMS
3. Διασύνδεση μεταξύ DBMS

Αρχιτεκτονικές DBMS
1) Διασύνδεση Συσκευών Αποθήκευσης

- **Απ' ευθείας** σύνδεσης μέσω κάποιας διεπαφής (αλλά όχι μέσω δικτύου) (π.χ., SCSI, SATA, κτλ.)
  - **Π.χ., Direct Attached Storage (DAS)**
- Σύνδεση μέσω Δικτύου με σύστημα αποθήκευσης
  - **Network attached storage (NAS)**
  - Χρησιμοποιείται σήμερα και σε μικρές επιχειρήσεις, οικίες με χρήση πρωτοκόλλων όπως TCP/IP, κτλ.

- Σύνδεση μέσω (οπτικού) κόμβου (switch) με αποθηκευτές
  - **Storage area network (SAN)**
  - Για μεγάλες επιχειρήσεις με πολλά TB δεδομένων (σήμερα υπάρχουν και λύσεις πάνω από TCP/IP το οποίο οδηγεί στη σύγκλιση NAS/SAN)

Αρχιτεκτονικές DBMS
1) Διασύνδεση Συσκευών Αποθήκευσης



Αρχιτεκτονικές DBMS
1) Διασύνδεση Συσκευών Αποθήκευσης

- **RAID*:** Είδος **DAS** όπου γίνεται διάταξη πολλαπλών δίσκων το οποίο δίνει την εντύπωση ενός **μεγάλου** δίσκου **υψηλών αποδόσεων**!
- **Στόχοι:**
  - Αύξηση **Επίδοσης**
    - Γιατί; Οι δίσκοι έχουν μηχανικά συστατικά ➔ είναι αργοί!
  - Αύξηση **Αξιοπιστίας**
    - Γιατί; Μηχανικά και Ηλεκτρονικά Συστατικά έχουν σφάλματα!

RAID*: Redundant Array of Independent* **(Inexpensive)** Disks

(Εφεδρικές Συστοιχίες Ανεξαρτήτων Δίσκων)

Αρχιτεκτονικές DBMS
2) Διασύνδεση Πελάτη και DBMS

A) Κεντρικοποιημένη (Centralized) DBMS:

- **Συνδυάζει τα πάντα σε ένα ενιαίο σύστημα**
  - Συμπεριλαμβανομένου: **Λογισμικού** DBMS, **υλικού**, **προγραμμάτων εφαρμογών** και **λογισμικό αλληλεπίδρασης** χρήστη.
- Οι Χρήστες συνδέονται μέσω **τερματικών** και η επεξεργασία γίνεται **κεντρικοποιημένα**.

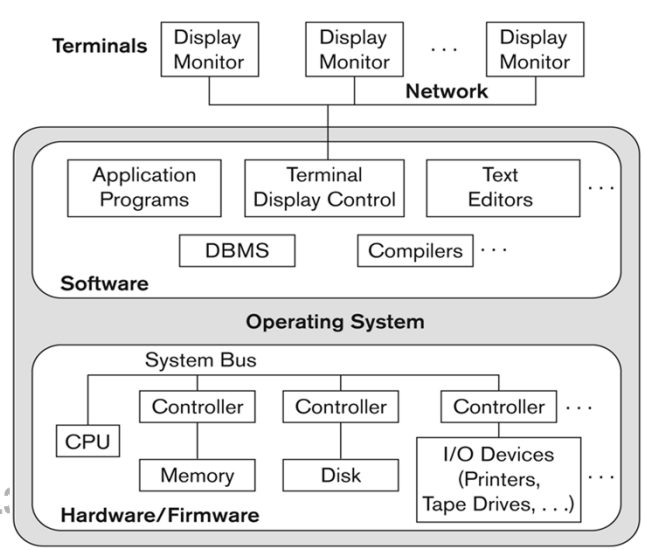


Figure 2.4
A physical centralized architecture.

Mainframe Databases

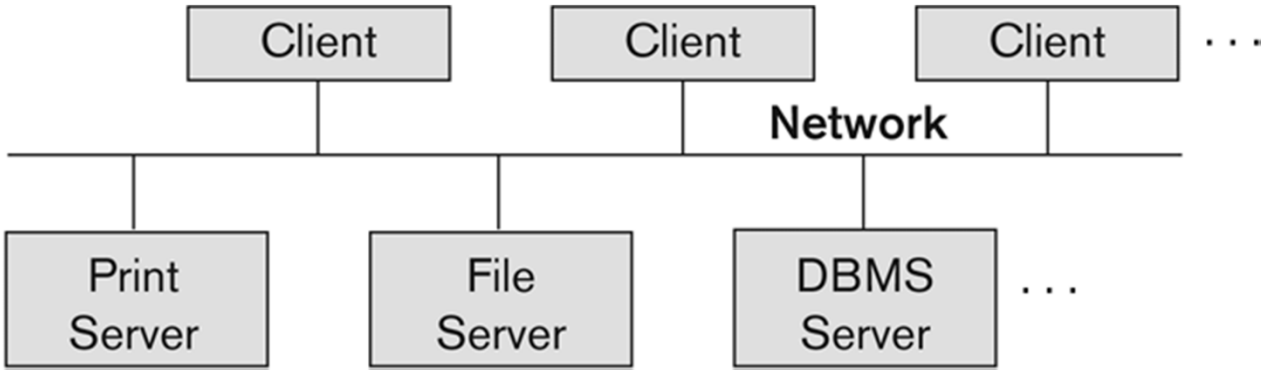
Αρχιτεκτονικές DBMS
2) Πελάτη/Εξυπηρετητή 2-επιπέδων

Β) Βασική Αρχιτ. Πελάτη-Εξυπ. 2-επιπέδων

- **2-επιπέδων:** Το **λογισμικό** είναι **κατανεμημένο** πάνω από **δυο συστήματα**: τον **πελάτη** και τον **εξυπηρετητή**

- Οι πελάτες ζητούν **υπηρεσίες** από τους εξυπηρετητές (διακομιστές) όπως απαιτείται.
  - Π.χ, Print servers, File servers, DBMS servers, Web servers, Email servers, κτλ.

Figure 2.5
Logical two-tier client/server architecture.



- Θα ορίσουμε τώρα τι είναι πελάτης και τι διαθέτης….

Αρχιτεκτονικές DBMS
2) Πελάτη/Εξυπηρετητή 2-επιπέδων

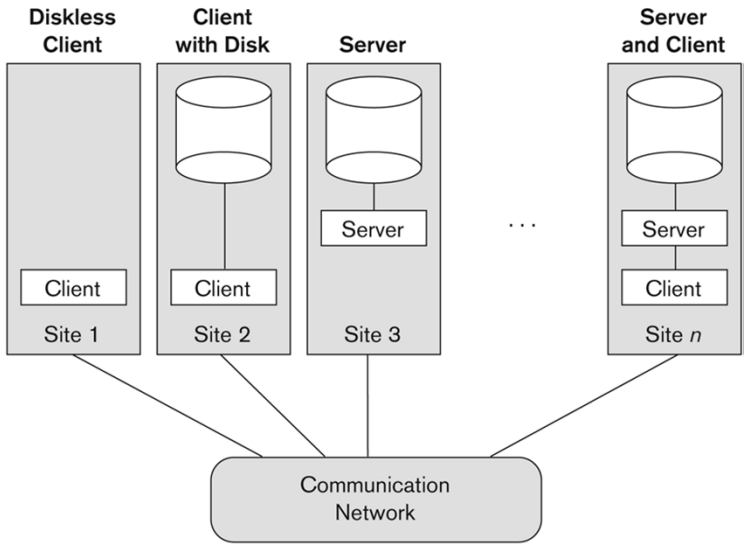
Πελάτης (Client): Εάν πληροί τα ακόλουθα:

- **Προϋπόθεση Α:** Διαθέτουν κάποιας μορφής τοπικής **μονάδας επεξεργασίας**
  - Π.χ., **PC**, **Workstations**, **Smartphones** αλλά ακόμη και **μηχανές χωρίς δευτερεύουσα μνήμη** τα οποία τρέχουν το λογισμικό του «Πελάτη»
    - Δεν είναι δηλαδή απλά τερματικά (γιατί θα ήταν Κεντρικοποιημένη Αρχ.)
- **Προϋπόθεση Β:** Συνδεδεμένα με τους εξυπηρετητές μέσω κάποιου είδους **δικτύου**.
  - LAN, WIFI, κτλ.
- **Προϋπόθεση Γ:** Διαθέτουν **διεπαφές (interfaces)** στο χρήστη μέσω των οποίων μπορεί να έχει **πρόσβαση στις υπηρεσίες** του εξυπηρετητή.

Αρχιτεκτονικές DBMS
2) Διαθέτης DBMS

- **Διαθέτης (Server) ή Διακομιστής:** Ένα σύστημα το οποίο παρέχει το **υλικό** και το **λογισμικό** για την **παροχή υπηρεσιών** προς τους **πελάτες**.
  - **Υπηρεσίες:** Εκτέλεση Επερωτήσεων και Δοσοληψιών
- Ένα μηχάνημα μπορεί να φέρει ένα ή περισσότερους ρόλους, π.χ.,:

Figure 2.6
Physical two-tier client/server architecture.



Αρχιτεκτονικές DBMS
2) Διαθέτης DBMS

- Οι **Σχεσιακοί (Relational) DBMS** servers ονομάζονται συχνά και **SQL servers**, **query servers**, ή **transaction servers**
- Οι εφαρμογές των πελατών κάνουν χρήση κάποιας **βιβλιοθήκης (Application Program Interface - API)** για να συνδεθούν με ένα εξυπηρετητή με κάποιο προτυποποιημένο τρόπο:
  - **ODBC**: **Open Database Connectivity** standard (υλοποιήσεις για όλα τα Λειτουργικά Συστήματα)
  - **JDBC: JAVA DB Connectivity** για πρόσβαση από προγράμματα JAVA (η JAVA είναι cross-platform)
- Τόσο ο πελάτης όσο και ο εξυπηρετητής πρέπει να είναι εφοδιασμένοι με **κατάλληλο λογισμικό** για χρήση των πιο πάνω **standards** (ακολουθεί παράδειγμα)

Αρχιτεκτονικές DBMS
2) Παράδειγμα Σύνδεσης Πελάτη-Εξυπηρετητή

- *Παράδειγμα χρήσης **JDBC driver** σε ένα πελάτη γραμμένο σε **JAVA** για σύνδεση με βάση **PostgreSQL** εγκατεστημένη στο **ίδιο μηχάνημα**.*

```
import java.sql.*;  // Αντίστοιχος Μηχανισμός στο .NET Framework

public static void main (String[] args) {
    Class.forName("org.postgresql.Driver");
    Connection conn = DriverManager.getConnection(
     "jdbc:postgresql:test", “some-user”, “some-password”);
     String query = "SELECT * FROM EMPLOYEE WHERE employeeid=";
     query += “14;” // παράδειγμα δυναμικής SQL (παράγεται διαφορετική
     εντολή SQL ανάλογα με την ροη εκτέλεσης του προγράμματος
     Statement stmt = conn.createStatement();
     ResultSet rs =; stmt.executeQuery(query)
     while ( rs.next() ) { ….} // Επεξεργασία αποτελεσμάτων στη JAVA
```

Αρχιτεκτονικές DBMS
2) Πελάτη-Εξυπηρετητή 2-επιπέδων

- Ένα πρόγραμμα πελάτη μπορεί να συνδέεται με πολλαπλά DBMSs, τα οποία ονομάζονται κάποτε **πηγές δεδομένων**

- **Πηγές Δεδομένων (Data Sources)** μπορεί να είναι από **απλά αρχεία** μέχρι δεδομένα εξειδικευμένου **λογισμικού** το οποίο διαχειρίζεται δεδομένα (όχι απαραίτητα DBMS)
  - Π.χ., Αρχεία Excel, Αρχεία Κειμένου, XML, κτλ.

- Τα DBMSs ξεκίνησαν ως **κεντρικοποιημένα, στη συνέχεια** αποκεντρώθηκε το **UI** και οι **εφαρμογές**. Η εκτέλεση **επερωτήσεων (queries)** και οι δοσοληψίες παρέμειναν στο διαθέτη.
  - Η **SQL** μπορεί να **θεωρηθεί ότι είναι το «πρωτόκολλο»** μεταξύ του **πελάτη** και του **εξυπηρετητή** (αυτό με το οποίο ζητά λειτουργίες ο πελάτης από την εξυπηρετητή)

Αρχιτεκτονικές DBMS
2) Πελάτη/Εξυπηρετητή 3-επιπέδων

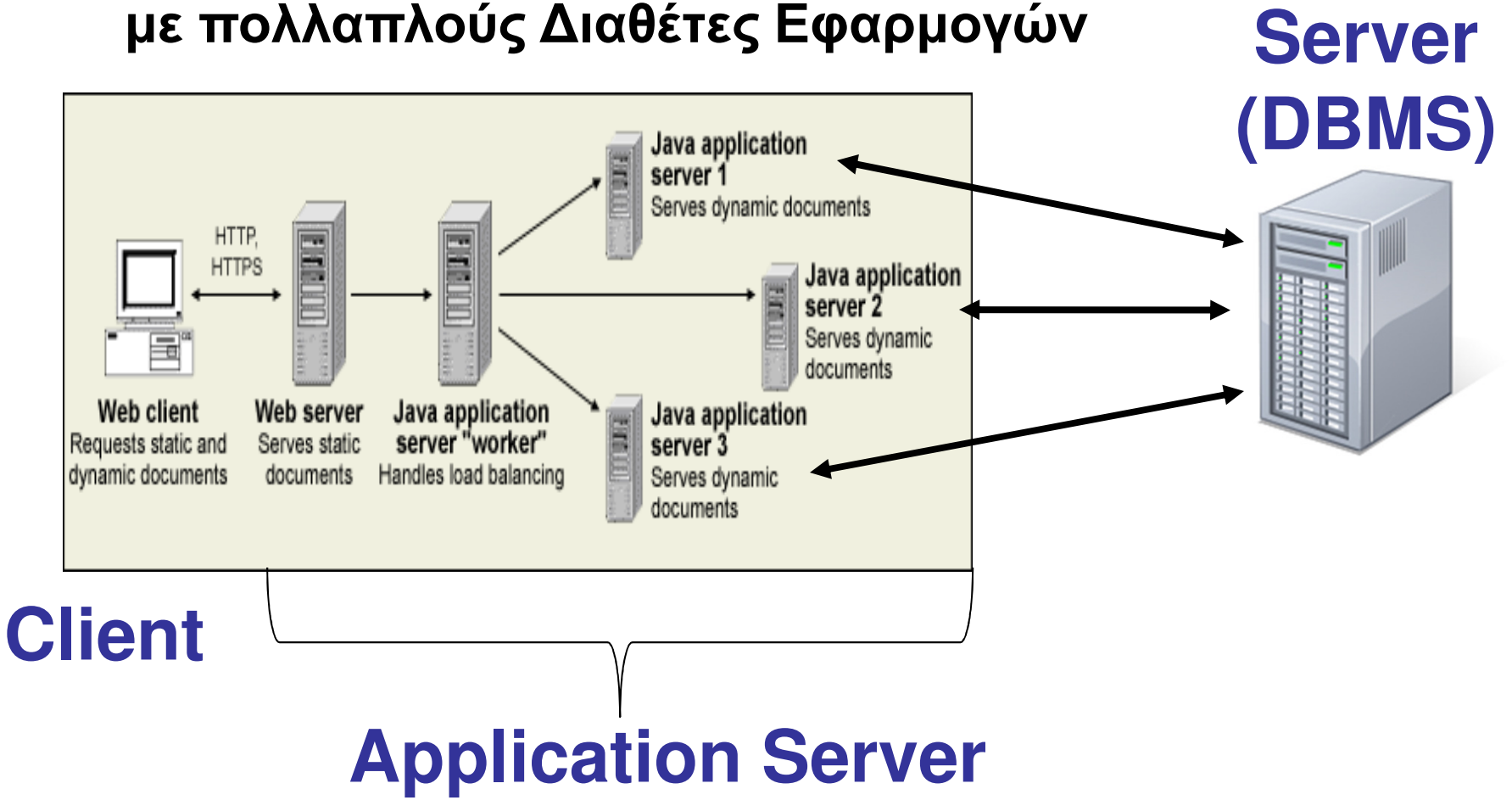
Γ) Πελάτη-Εξυπ. 3-επιπέδων

- Η έννοια αυτή **δημιουργήθηκε** με την εξέλιξη του **Web.**
- **3-επιπέδων:** Το λογισμικό είναι κατανεμημένο πάνω από 3 συστήματα: τον **Πελάτη**, τον **Application Server** (**Διαθέτης Εφαρμογών** ή Web Server) και τον **Εξυπηρετητή**

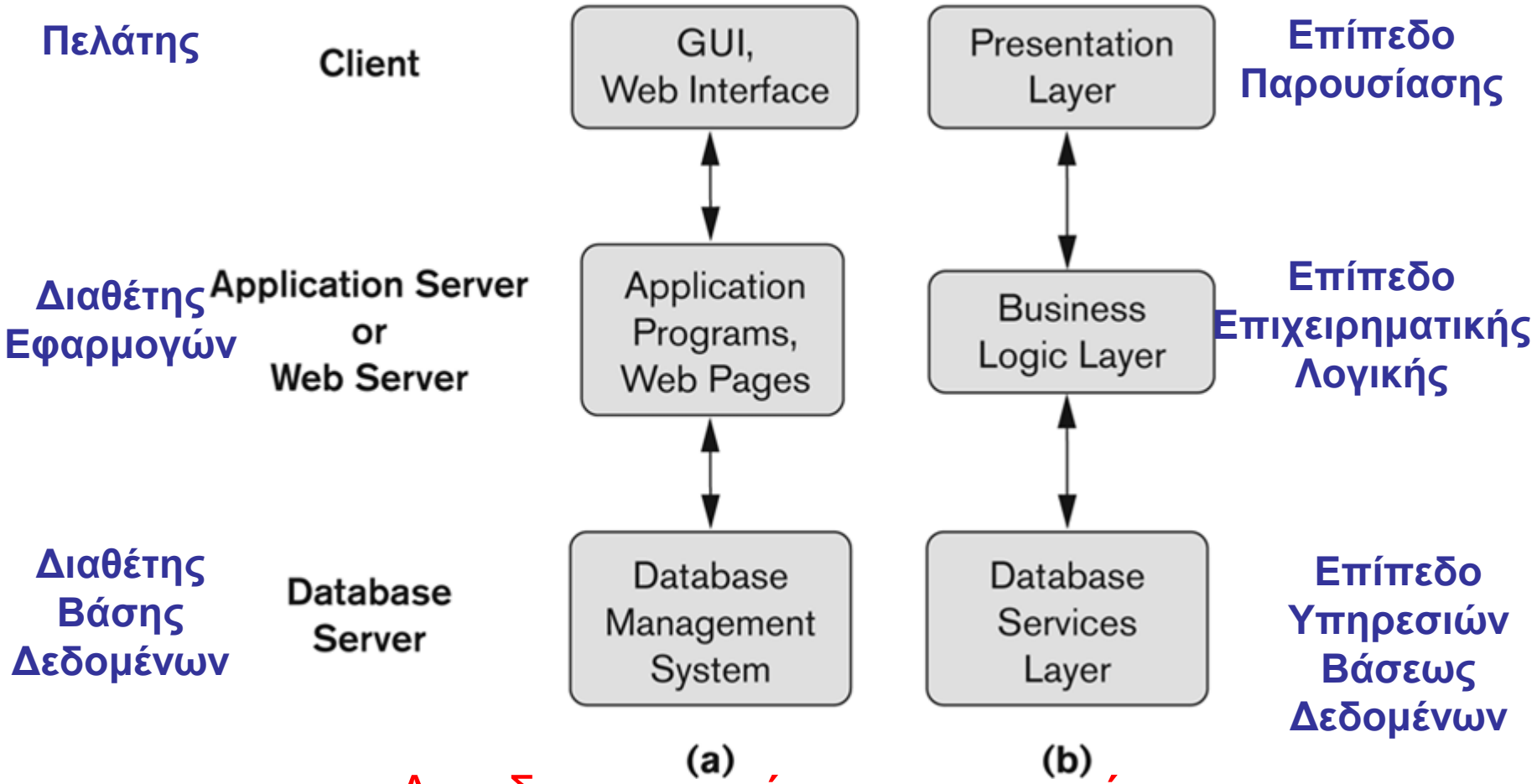
- Οι πελάτες **ζητούν υπηρεσίες** από τους **Application Servers**, αυτοί με την **σειρά τους** από τους **εξυπηρετητές**.
- Οι **Application Servers** επιβάλλουν διάφορους υψηλού επιπέδου **περιορισμούς** που επιβάλει **ένας οργανισμός** (π.χ., διαδικασίες, κανονισμούς, κτλ) και κυρίως **ασφάλεια**
  - Ένας πελάτης **ΔΕΝ** μπορεί να έχει **άμεση πρόσβαση στη DBMS**!
  - Π.χ, IBM Websphere Server, Oracle Weblogic Server, .NET Framework, Adobe's JRUN, Apple's WebObjects, SAP Netweaver,

Αρχιτεκτονικές DBMS
2) Πελάτη/Εξυπηρετητή 3-επιπέδων

Παράδειγμα Αρχιτεκτονικής 3-επιπέδων με πολλαπλούς Διαθέτες Εφαρμογών



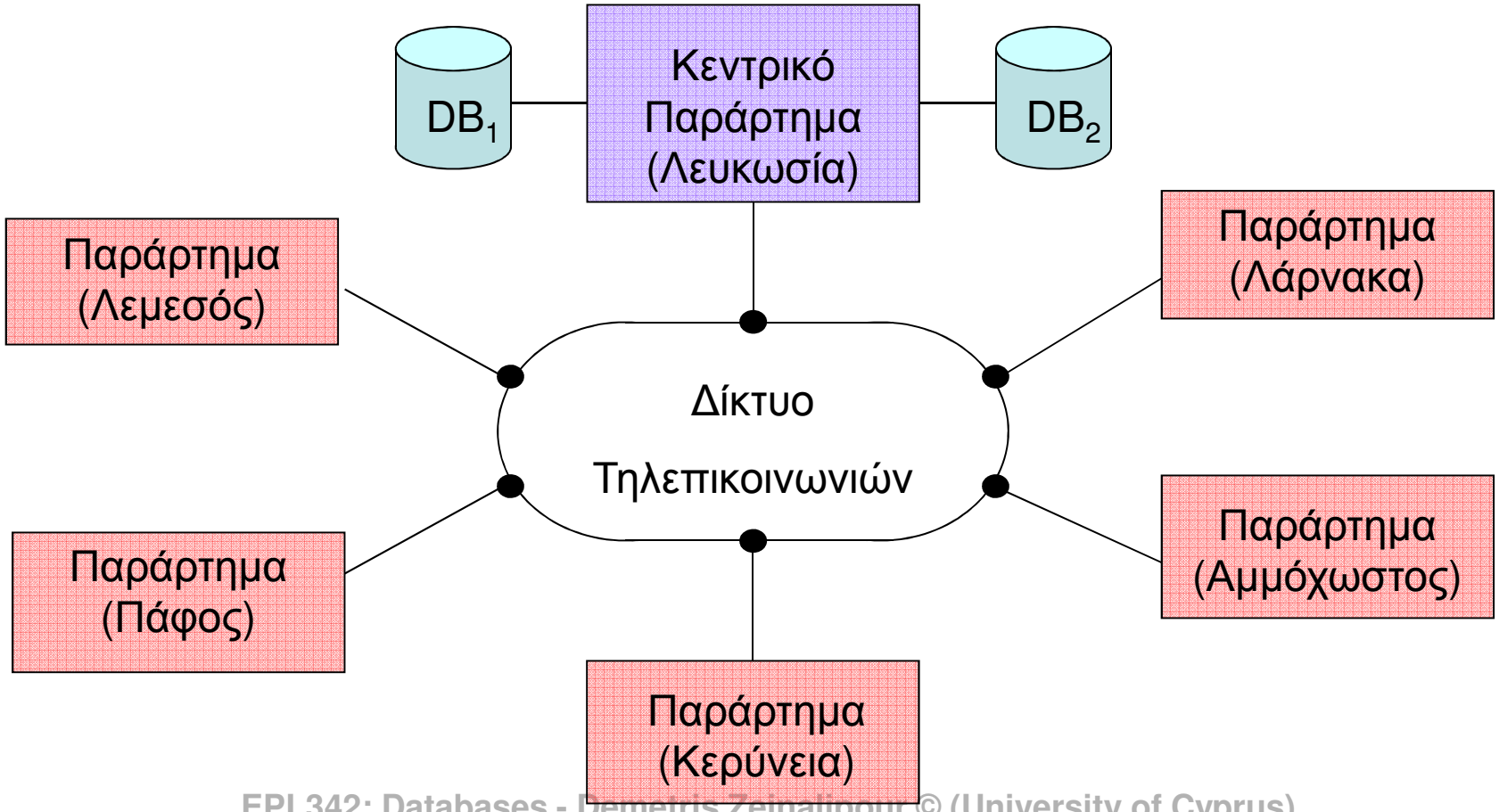
Αρχιτεκτονικές DBMS
2) Πελάτη/Εξυπηρετητή 3-επιπέδων



Δυο διαφορετικές αναπαραστάσεις της αρχιτεκτονικής 3 επιπέδων

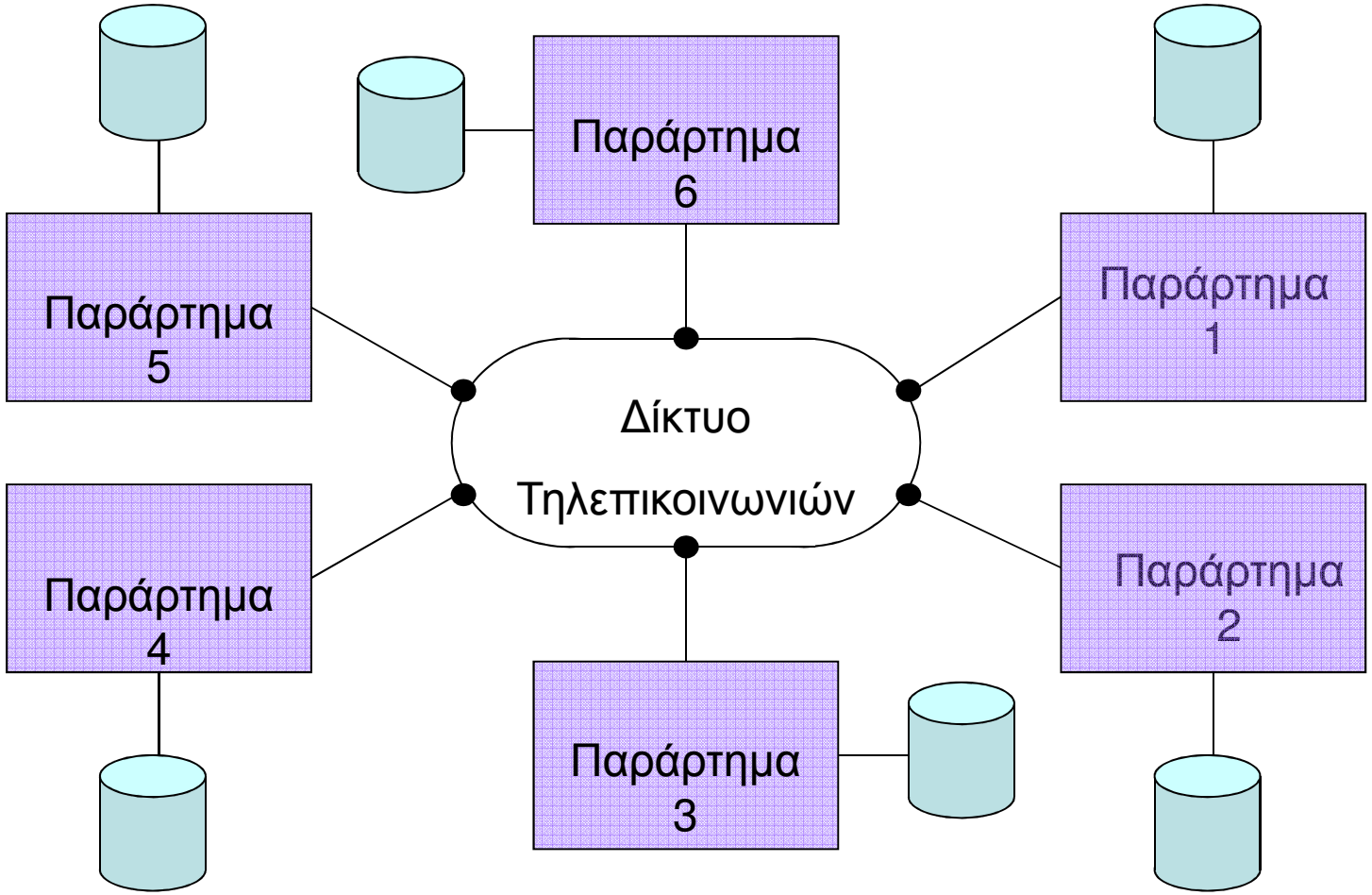
Αρχιτεκτονικές DBMS
3) Διασύνδεση μεταξύ DBMS

Τυπική Αρχιτεκτονική Client/Server ή Centralized



Αρχιτεκτονικές DBMS
3) Διασύνδεση μεταξύ DBMS

Κατανεμημένη Αρχιτεκτονική



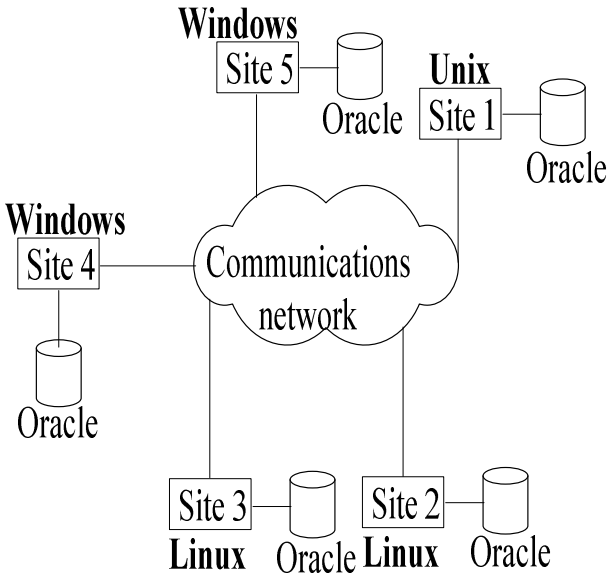
Αρχιτεκτονικές DBMS
3) Διασύνδεση μεταξύ DBMS

- **Κατανεμημένη Βάση (DDB)**
  - Μια **συλλογή** από **λογικά** συσχετιζόμενες αλλά **ανεξάρτητες** βάσεις δεδομένων προσβάσιμες πάνω από ένα **δίκτυο υπολογιστών**.
- **Κατανεμημένο Συσ. Διαχ. Βασ. Δεδ. (DDBMS)**
  - Α **γενικό λογισμικό σύστημα** το οποίο διαχειρίζεται κατανεμημένες βάσεις κάνοντας την **κατανομή** (δεδομένων, φόρτου, κτλ.,) **διαφανή** στο **χρήστη**.
- Οι περισσότερες βάσεις (εμπορικές και μη) **προσφέρουν** κάποιου είδους **δυνατότητες/επεκτάσεις** για να μετατραπεί η βάση σε **κατανεμημένη βάση δεδομένων.**
- **Βασικό Πρόβλημα:** Η έλλειψη κοινών αποδεκτών προτύπων (δηλ., η κατανομή περιορίζεται κυρίως μεταξύ ομογενών (όμοιων) βάσεων)

Αρχιτεκτονικές DBMS
3) Διασύνδεση μεταξύ DBMS

Ομογενής (Homogeneous): Κάθε site τρέχει την **ίδια DBMS** (π.χ., όλες είναι εφοδιασμένες με Oracle)

- Το Λειτουργικό Σύστημα **μπορεί ωστόσο να διαφέρει μεταξύ των sites:**
  - Π.χ., ΌΛΑ τα sites τρέχουν **Oracle** xor **DB2**, xor **Sybase** xor κάποιο άλλο DBMS.
  - Τα Λειτουργικά Συστήματα των κόμβων μπορεί να είναι ένα κράμα από **Linux, Window, Unix,** etc.

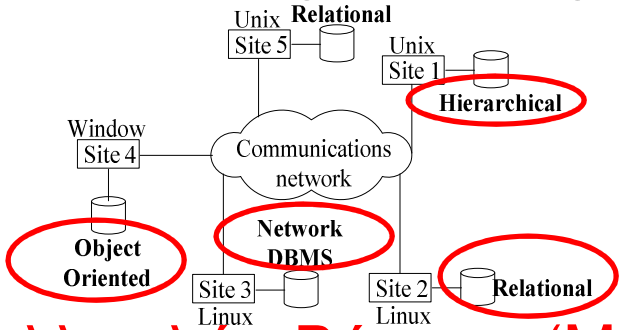


Oracle Βάση

Αρχιτεκτονικές DBMS
3) Διασύνδεση μεταξύ DBMS

Ετερογενής (Heterogeneous): Διαφορετικά sites τρέχουν διαφορετικά DBMSs (ή ακόμη non-relational DBMSs).

- **Τύποι Ετερογενών Βάσεων**
  - **Ομόσπονδα (Federated): Ένα Σχήμα.** Κάθε site μπορεί να τρέχει διαφορετικό DBMS αλλά η πρόσβαση στη πληροφορία **οργανώνεται μέσω ενός ενιαίου σχήματος (schema)**
    - Υπάρχει κεντρικοποιημένη διαχείριση **ασφάλειας**.
    - Δεν υπάρχει πολύ τοπική αυτονομία σε κάθε βάση.
  - **Πολλαπλών Βάσεων (Multidatabase): Καθόλου Σχήμα.** Δεν υπάρχει ένα καθολικό εννοιολογικό σχήμα. Για πρόσβαση στη πληροφορία οργανώνεται από τις εφαρμογές.



Παράδειγμα Πολλαπλών Βάσεων (Multidatabase)

Αρχιτεκτονικές DBMS
3) Διασύνδεση μεταξύ DBMS

- Στις κατανεμημένες βάσεις σημαντικό είναι το θέμα της **Κατάτμησης (Fragmentation)** και **Αντίγραφα (Replication)**
  - Π.χ., Οι πίνακες **EMPLOYEE**, **PROJECT**, and **WORKS_ON** μπορεί να **κατατμηθούν οριζόντια (fragmented horizontally)** και να **αντιγραφούν (replication)** για αύξηση **επίδοσης**, **αξιοπιστίας** , κτλ. οπως πιο κάτω: