Βάσεις Δεδομένων - 2

Έννοιες Συστήματος ΒΔ και Αρχιτεκτονικές

Δρ. Χαράλαμπος Καρανίκας

Περιεχόμενα

- Μοντέλα δεδομένων και οι κατηγορίες τους
- Ιστορία Μοντέλων Δεδομένων
- Σχήμα, Στιγμιότυπα, και καταστάσεις ΒΔ
- Αρχιτεκτονική Τριών-Επιπέδων
- Ανεξαρτησία Δεδομένων
- Γλώσσες και Διεπαφές ΣΔΒΔ
- Βοηθητικά προγράμματα και Εργαλεία Συστήματος ΒΔ
- Κεντρικές και Client-Server Αρχιτεκτονικές
- Κατανεμημένες Αρχιτεκτονικές

Μοντέλα δεδομένων 1/2

- Ένα **μοντέλο δεδομένων** είναι ένα σύνολο εννοιών που μπορούν να χρησιμοποιηθούν για την περιγραφή της δομής ενός database.

- **Δομή ενός database** περιλαμβάνει τύπους δεδομένων, σχέσεις και περιορισμούς που υφίστανται μεταξύ των δεδομένων, καθώς και βασικούς τελεστές για ανάκτηση και αλλαγή του database.

Μοντέλα δεδομένων 2/2

Τέτοιες έννοιες περιλαμβάνουν **έννοιες Δομής**

- **Οντότητες** (***Elements***) *και τους τύπους δεδομένων τους*
  - ***Ομάδες Οντοτήτων*** *(Εγγραφές, Πίνακες, κτλ)*
- ***Συσχετίσεις (Relationships***) μεταξύ των Οντοτήτων.
- **Περιορισμούς (Constraints)** τα οποία περιλαμβάνουν κάποιους κανόνες οι οποίοι πρέπει να τηρούνται πάντα για να είναι η ΒΔ σε μια συνεπή κατάσταση.

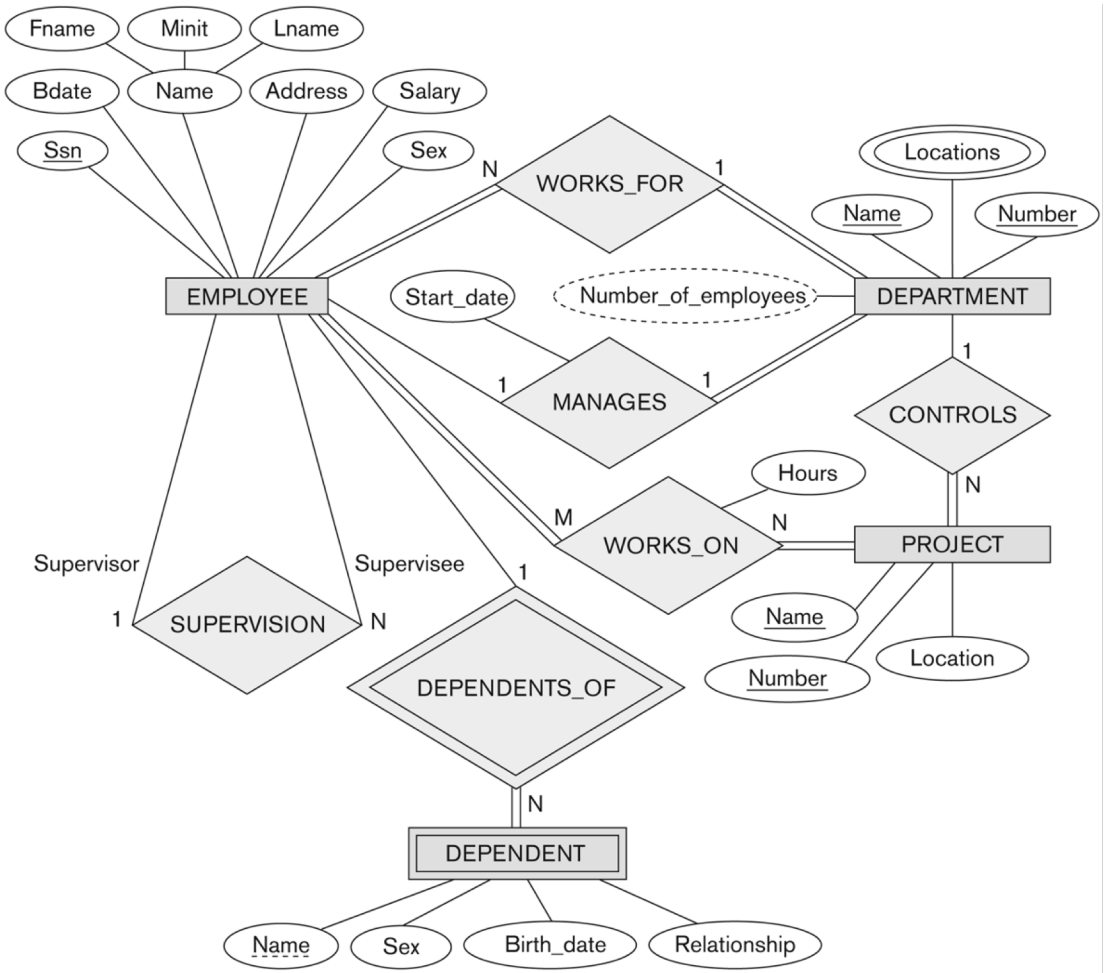
Τέτοιες έννοιες περιλαμβάνουν **έννοιες Τελεστών**

- Τελεστές **Ανάκτησης** και **Ενημέρωσης** Δεδομένων
- Χωρίζεται σε **βασικούς τελεστές** (insert, delete, update) και τελεστών χρηστών (π.χ., compute_student_gpa, update_inventory)

Κατηγορίες Μοντέλων

- **Υψηλού επιπέδου, εννοιολογικό μοντέλο (high-level or conceptual)**
  - Παρέχει έννοιες κοντά στον τρόπο που πολλοί χρήστες καταλαβαίνουν τα διάφορα δεδομένα
  - Π.χ., **Entity-Relationship Model**
- **Αναπαραστατικό Μοντέλο (representational or implementational)**
  - Παρέχει έννοιες που είναι μεν κατανοητές από τους χρήστες αλλά όχι πολύ απομακρυσμένες από το τρόπο αποθήκευσης
  - Π.χ., **Relational Model** and DB Schemas
- **Χαμηλού επιπέδου (low-level or physical)**
  - Παρέχει έννοιες που περιγράφουν τις λεπτομέρειες του πως τα δεδομένα είναι αποθηκευμένα στη δευτερεύουσα μνήμη
  - Π.χ., Specific Storage Model

Παράδειγμα **Εννοιολογικού** Μοντέλου



Υψηλού Επίπεδου

Παράδειγμα **Αναπαραστατικού** Μοντέλου

Ενδιάμεσου Επίπεδου

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Σχήμα και Κατάσταση της Βάσης Δεδομένων

- **Σχήμα της Βάσης** είναι η δομή της βάσης η οποία:
  - Περιλαμβάνει τύπους δεδομένων, σχέσεις, περιορισμούς στα δεδομένα
  - Είναι ανεξάρτητη οποιασδήποτε εφαρμογής
  - Σπάνια αλλάζει

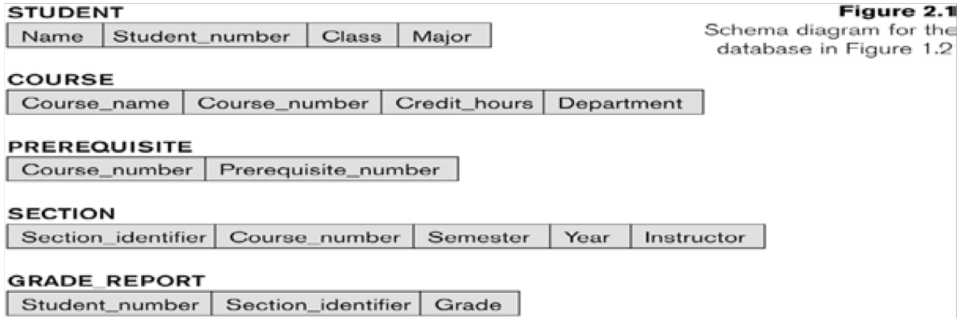


Figure 2.1
Schema diagram for the database in Figure 1.2

- **Κατάσταση ή Στιγμιότυπο της Βάσης**:
  - Είναι τα πραγματικά δεδομένα που υπάρχουν σε μια βάση δεδομένων, μια συγκεκριμένη χρονική στιγμή (DB instance, **occurrence** or **snapshot**)
  - Η Κατάσταση μια βάσης αλλάζει συχνά.

Σχήμα και Κατάσταση της Βάσης Δεδομένων

- Αρχική Κατάσταση ΒΔ (Initial Database State):
  - Αναφέρεται στην κατάσταση της βάσης δεδομένων όταν φορτώνεται αρχικά στο σύστημα. (Αρχικοποίηση)

- Έγκυρη κατάσταση (Valid State):
  - Μια κατάσταση που ικανοποιεί την δομή και τους περιορισμούς της βάσης δεδομένων.

- Το **Σχήμα** ονομάζεται επίσης και "**intension**" (σκοπός)
- Η **Κατάσταση** ονομάζεται επίσης και "**extension**" (επέκταση) του σχήματος.

Παράδειγμα ενός σχήματος vs κατάστασης ΒΔ

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Figure 2.1
Schema diagram for the database in Figure 1.2.

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 1.2
A database that stores student and course information.

Αρχιτεκτονική τριών επιπέδων

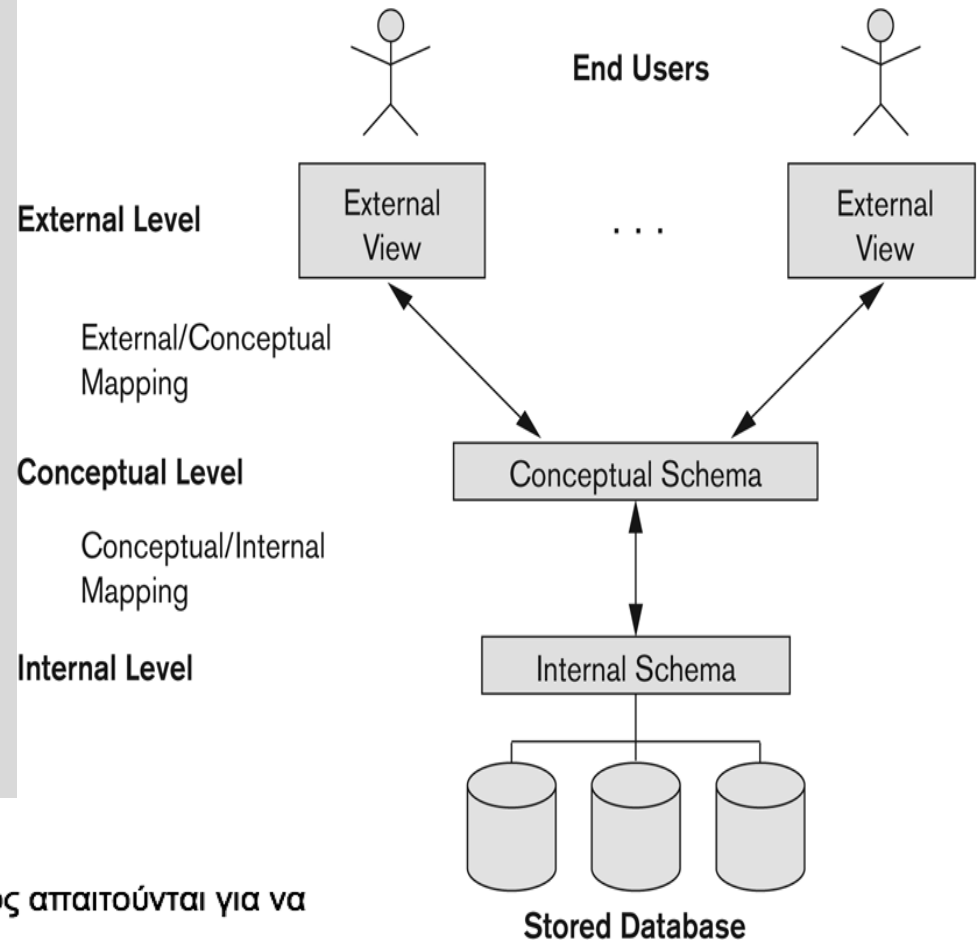
- Προτείνεται για να υποστηρίξει τα ακόλουθα χαρακτηριστικά των DBMS:
  - **Ανεξαρτησία προγραμμάτων-δεδομένων.**
  - Υποστήριξη **πολλαπλών όψεων** των δεδομένων. Πολλαπλά Views δεδομένων επιτρέπουν σε κάθε χρήστη/εφαρμογή να έχουν διαφορετική οπτική της βάσης

Αρχιτεκτονική τριών επιπέδων

- Ορίζει σχήματα DBMS σε **τρία** επίπεδα :
  - **Το Εξωτερικό(ή όψη)Επίπεδο (external or view level)**
    - Είναι ένα (ιδεατό) υποσύνολο του σχήματος της βάσης δεδομένων και αντιστοιχεί σε ένα πλαίσιο πεδίων που χρησιμοποιεί ο χρήστης ή μια ανεξάρτητη εφαρμογή
  - **Το Λογικό Επίπεδο (conceptual level)**
    - Περιγράφει τη δομή του database για όλους τους χρήστες
    - Κρύβει τις λεπτομέρειες αποθήκευσης
    - Υψηλού επιπέδου μοντέλο ή Αναπαραστασιακό Μοντέλο
  - **Το Εσωτερικό Επίπεδο (internal level)**
    - Έχει ένα εσωτερικό σχήμα το οποίο περιγράφει τη δομή της αποθήκευσης των δεδομένων. Περιγράφει τον τρόπο με τον οποίο τα δεδομένα δομούνται και αποθηκεύονται στις φυσικές συσκευές

Παράδειγμα Αρχιτεκτονικής τριών επιπέδων

- **Επίπεδο όψεων (view):**
  - CSMajors
  - MathMajors
- **Λογικό επίπεδο: όλο το σχήμα της βάσης**
  - Courses (CourseNo,CourseName,Credits,Dept)
  - Student (StudentID,Lname,Name,Class,Major)
  - GradeReport (StudentID,CourseNo,Grade,Term)
- **Φυσικό επίπεδο:**
  - how these tables are stored, how many bytes / attribute etc.



Αντιστοιχίσεις μεταξύ των επιπέδων του σχήματος απαιτούνται για να μετατρέπουν τις αιτήσεις και τα δεδομένα.

- Π.χ., Τα δεδομένα που εξάγονται από το εσωτερικό επίπεδο του ΣΔΒΔ μορφοποιούνται για να ταιριάζουν στην εξωτερική όψη του χρήστη (π.χ. Μορφοποίηση των αποτελεσμάτων μιας SQL επερώτησης για εμφάνιση σε μια ιστοσελίδα)

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 Συστήματος Βάσης Δεδομένων (Database Languages)

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:

Γλώσσες Συστήματος Βάσης Δεδομένων (Database Languages)

- Γλώσσα ορισμού δεδομένων - Data Definition Language (DDL)
  - Δίνει τη δυνατότητα δημιουργίας του ΣΧΗΜΑΤΟΣ
- Γλώσσα επεξεργασίας δεδομένων - Data Manipulation Language (DML)
  - Εντολές για πρόσβαση, ανάκτηση, εισαγωγή, αφαίρεση δεδομένων
- Σε ορισμένα ΣΔΒΔ υπάρχουν ξεχωριστά:
  - Γλώσσα ορισμού φυσικής αποθήκευσης (Storage Definition Language – SDL) Περιγράφει το εσωτερικό σχήμα
  - Γλώσσα ορισμού όψεων (View Definition Language – VDL) Ορίζει τις διάφορες όψεις και τη μεταφράζει από την όψη στο λογικό σχήμα

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*

Γλώσσα επεξεργασίας 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

Τύποι DML

- Δηλωτικές, υψηλού επιπέδου ή μη-διαδικαστικές (π.χ., SQL → declarative)
  - Ο χρήστης καθορίζει το **Τι** και όχι το **Πως**
  - Γλώσσες που χειρίζονται ένα σύνολο από εγγραφές ταυτοχρόνως (“set-at-a-time”)

- Διαδικαστικές (procedural)
  - Ο χρήστης καθορίζει το **Πως** να αποκτηθεί μια πληροφορία
  - Συνήθως είναι γλώσσες που χειρίζονται «εγγραφή μετά εγγραφή» (“record-at-a-time”) Π.χ., Ιεραρχικό ή Δικτυωτό Μοντέλο
  - Παρέχουν βρόχους επανάληψης, εντολές επιλογής κτλ.
  - Προεκτάσεις της SQL (π.χ., SQL3) παρέχουν τέτοιες δυνατότητες π.χ., ANSI SQL/PSM (Persistent Stored Modules), PL/SQL (Oracle), TSQL (Microsoft SQL Server)

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

```
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);

Διεπαφές ΣΔΒΔ - DBMS Interfaces

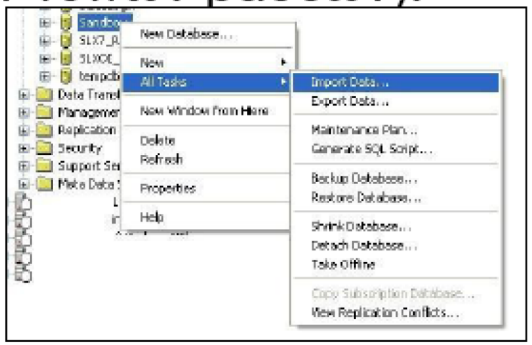
- Stand-alone query language interfaces
  - Example: Entering SQL queries at the DBMS interactive SQL interface (e.g. SQL*Plus in ORACLE)
- Programmer interfaces for embedding DML in programming languages
- User-friendly interfaces
  - Menu-based, forms-based, graphics-based, etc.

- Programmer interfaces for embedding DML in a programming languages:
  - **Embedded Approach**: e.g embedded SQL (for C, C++, etc.), SQLJ (for Java)
  - **Procedure Call Approach**: e.g. JDBC for Java, ODBC for other programming languages
  - **Database Programming Language Approach**: e.g. ORACLE has PL/SQL, a programming language based on SQL; language incorporates SQL and its data types as integral components

Βοηθητικά προγράμματα ΣΔΒΔ - Database System Utilities

Τέτοια εργαλεία συνοδεύουν τα εμπορικά και μη ΣΔΒΔ, και εκτελούν λειτουργίες όπως:

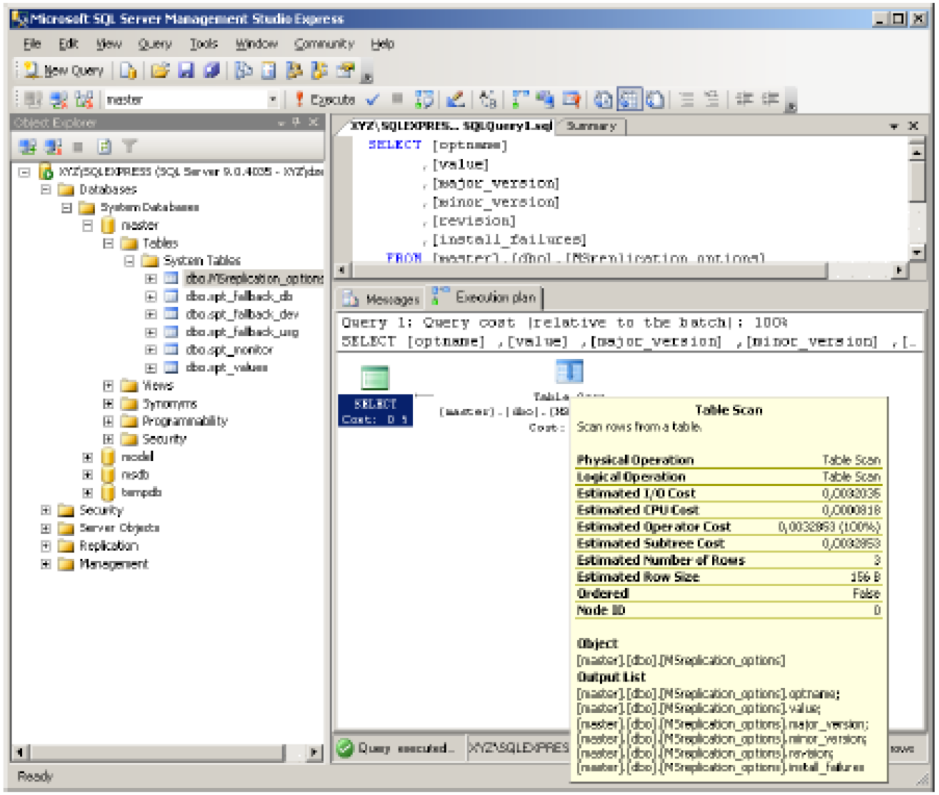
- **Φόρτωση Δεδομένων** (συμπεριλαμβανομένου και της **μετατροπής** μεταξύ διαφορετικών τύπων βάσεων).
- **Αντιγραφή/Επαναφορά (Backup/Restore)**
- **Αναδιοργάνωση Αρχείων, κτλ.**
  - **(π.χ., Shrink, Detach)**
- **Κτλ..**



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

Βοηθητικά προγράμματα ΣΔΒΔ - Database System Utilities



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

Άλλα εργαλεία

- Application Development Environments and CASE (computer-aided software engineering) tools:
- Παραδείγματα:
  - PowerBuilder (Sybase)
  - JBuilder (Borland)
  - JDeveloper 10G (Oracle)

Τυπικά Εργαλεία Λειτουργικών Μονάδων DBMS

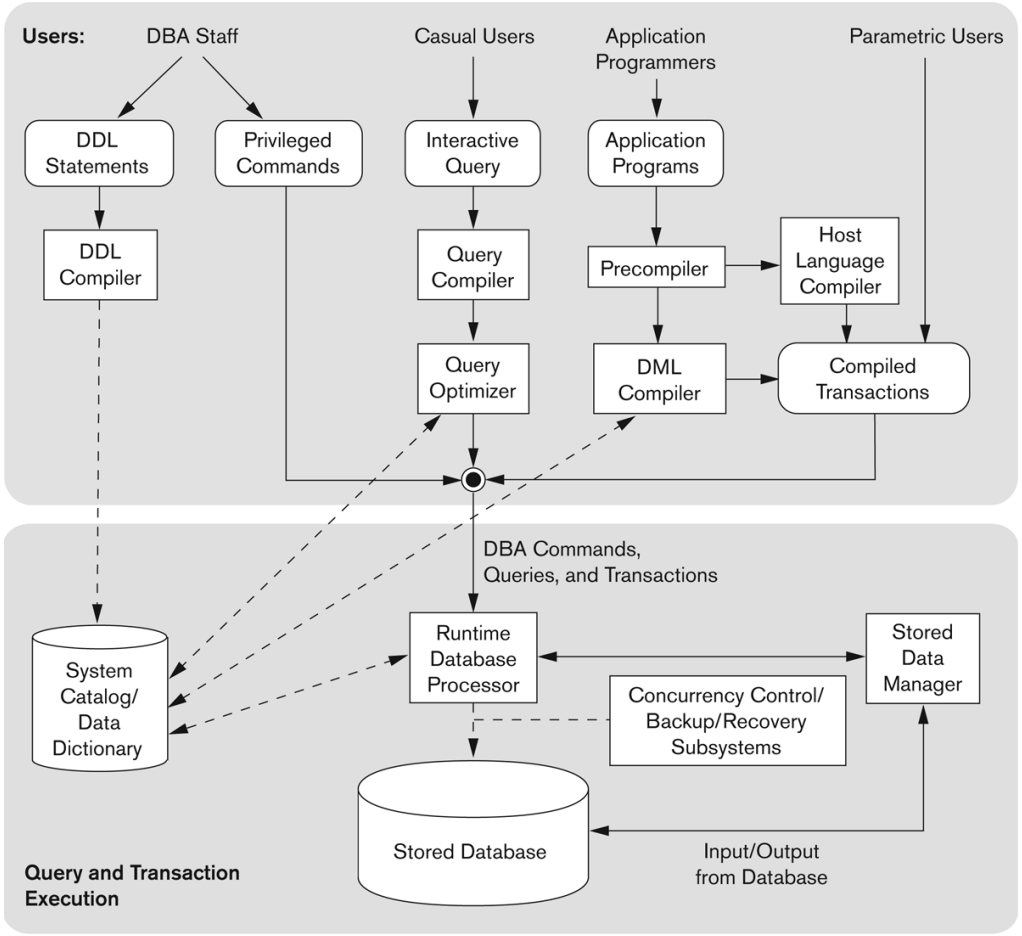


Figure 2.3
Component modules of a DBMS and their interactions.

Κεντρική και Client-Server Αρχιτεκτονική ΣΔΒΔ

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**.

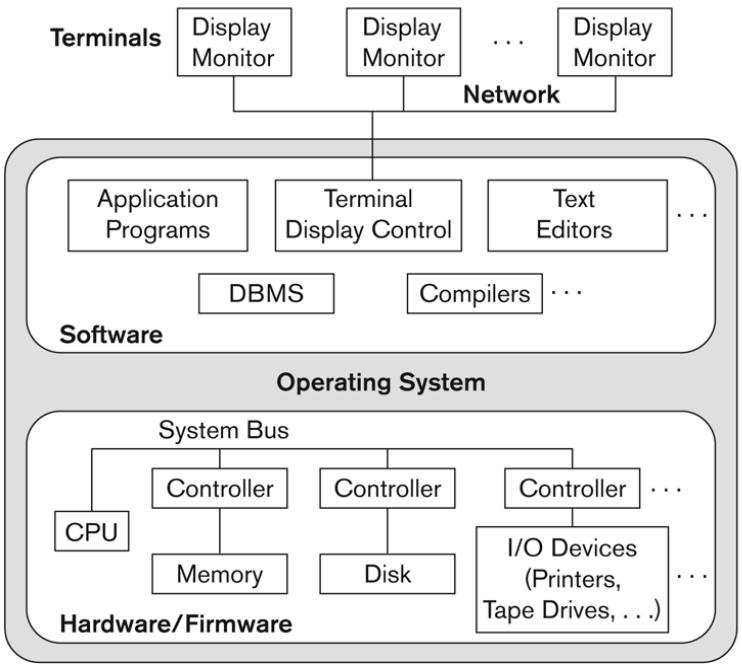


Figure 2.4
A physical centralized architecture.

Mainframe Databases

Βασική αρχιτεκτονική 2 – επιπέδων Client - Server

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.

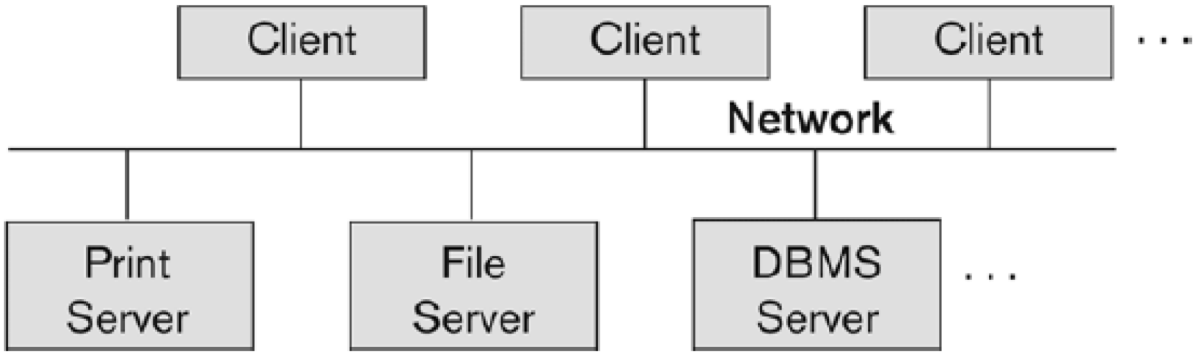


Figure 2.5 Logical two-tier client/server architecture.

Πελάτης (Client)

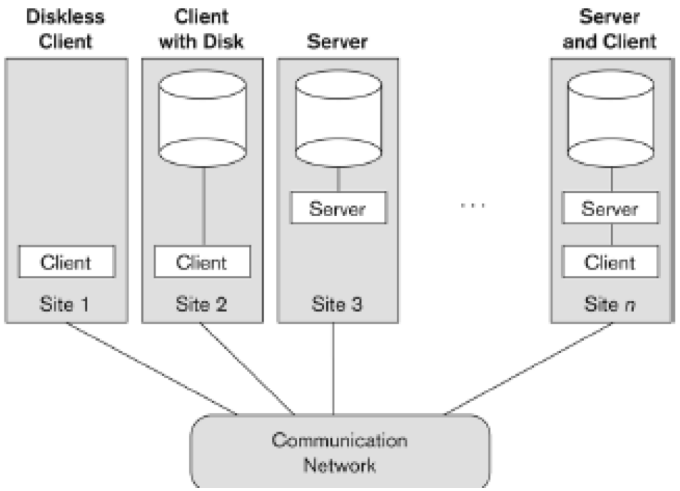
Πελάτης (Client): Εάν πληροί τα ακόλουθα:

- **Προϋπόθεση Α:** Διαθέτουν κάποιας μορφής τοπική **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.

DBMS Server 1/2

- **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.



DBMS Server 2/2

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)

- *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
```

2-επιπέδων Client-Server αρχιτεκτονική

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.

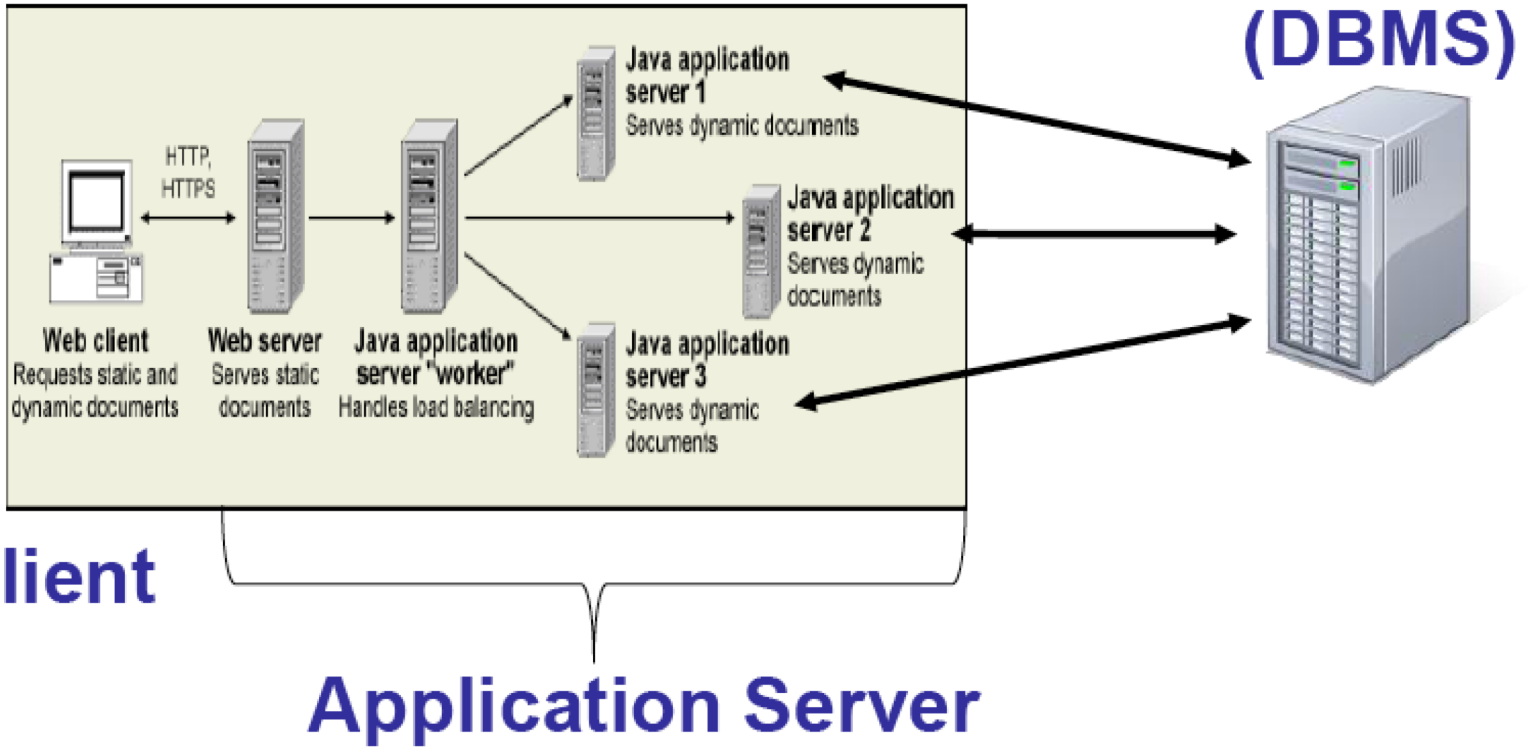
Τα DBMSs ξεκίνησαν ως **κεντρικοποιημένα, στη συνέχεια** αποκεντρώθηκε το **UI** και οι **εφαρμογές**. Η εκτέλεση **επερωτήσεων (queries)** και οι δοσοληψίες παρέμειναν στο διαθέτη.

- 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)

3-επιπέδων Client-Server αρχιτεκτονική

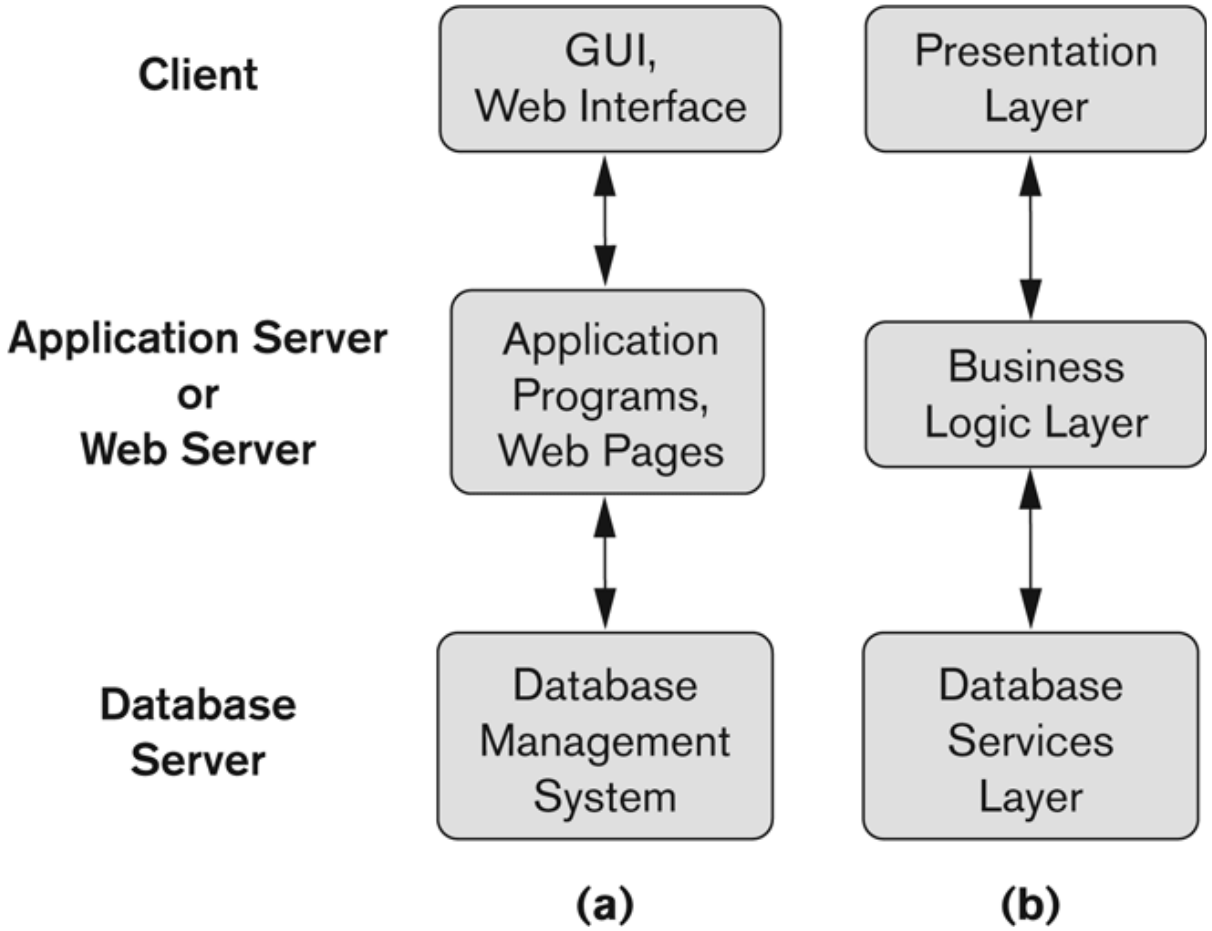
- Σύνηθες για Web applications
- Ενδιάμεσο επίπεδο που ονομάζεται Application Server ή Web Server:
  - Αποθηκεύει το λογισμικό σύνδεσης στο web και το τμήμα της επιχειρηματικής λογικής της εφαρμογής που χρησιμοποιείται για προσπέλαση των αντίστοιχων δεδομένων από τον διακομιστή της βάσης δεδομένων
  - Λειτουργεί σαν αγωγός αποστολής μερικώς επεξεργασμένων δεδομένων μεταξύ του διακομιστή της βάσης δεδομένων και του πελάτη.
- Σε τρία επίπεδα Αρχιτεκτονική μπορεί να ενισχύσει την ασφάλεια:
  - Database server only accessible via middle tier
  - Clients cannot directly access database server

Παράδειγμα Αρχιτεκτονικής 3-επιπέδων με πολλαπλούς Διαθέτες Εφαρμογών

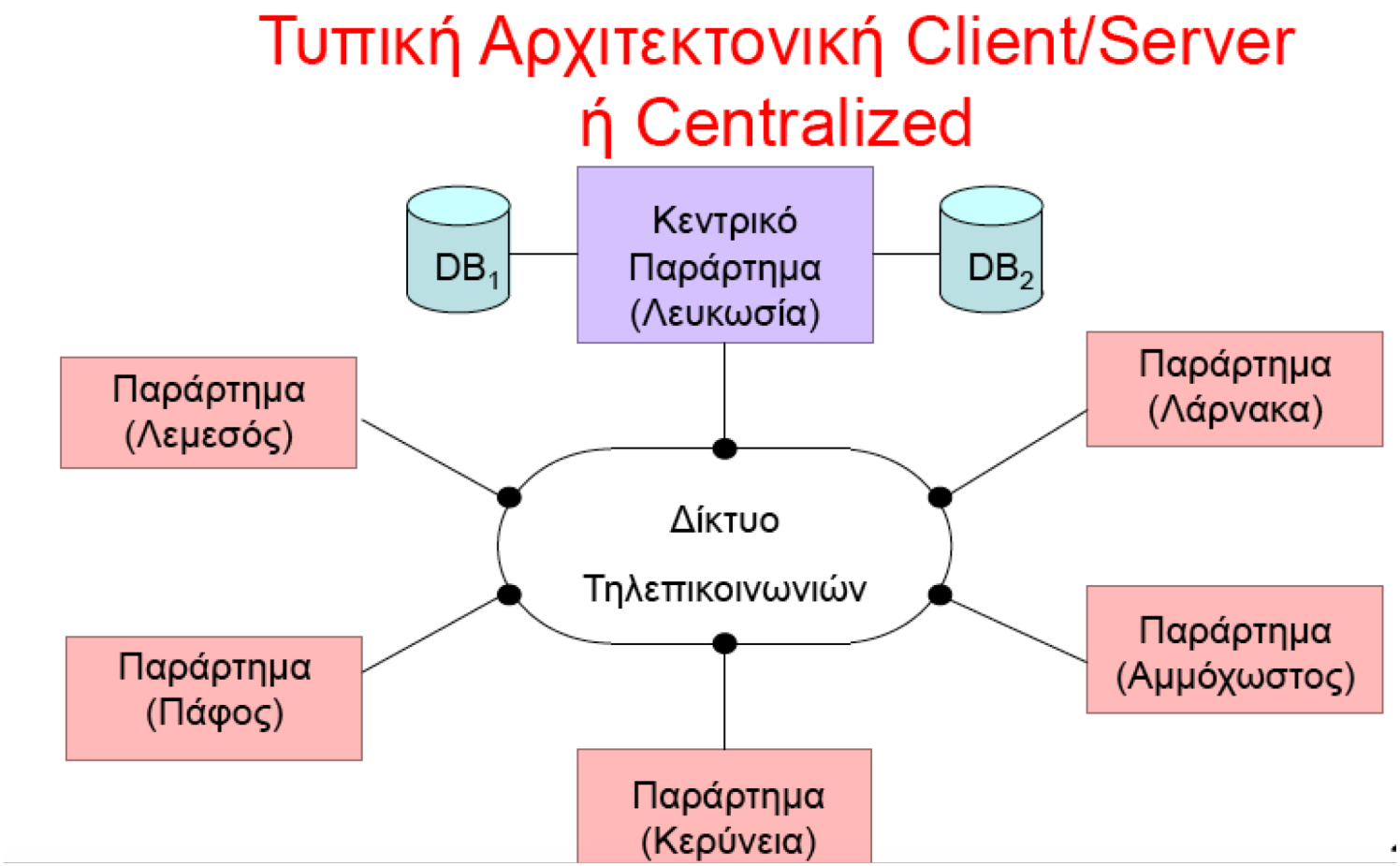


3-επιπέδων Client-Server αρχιτεκτονική

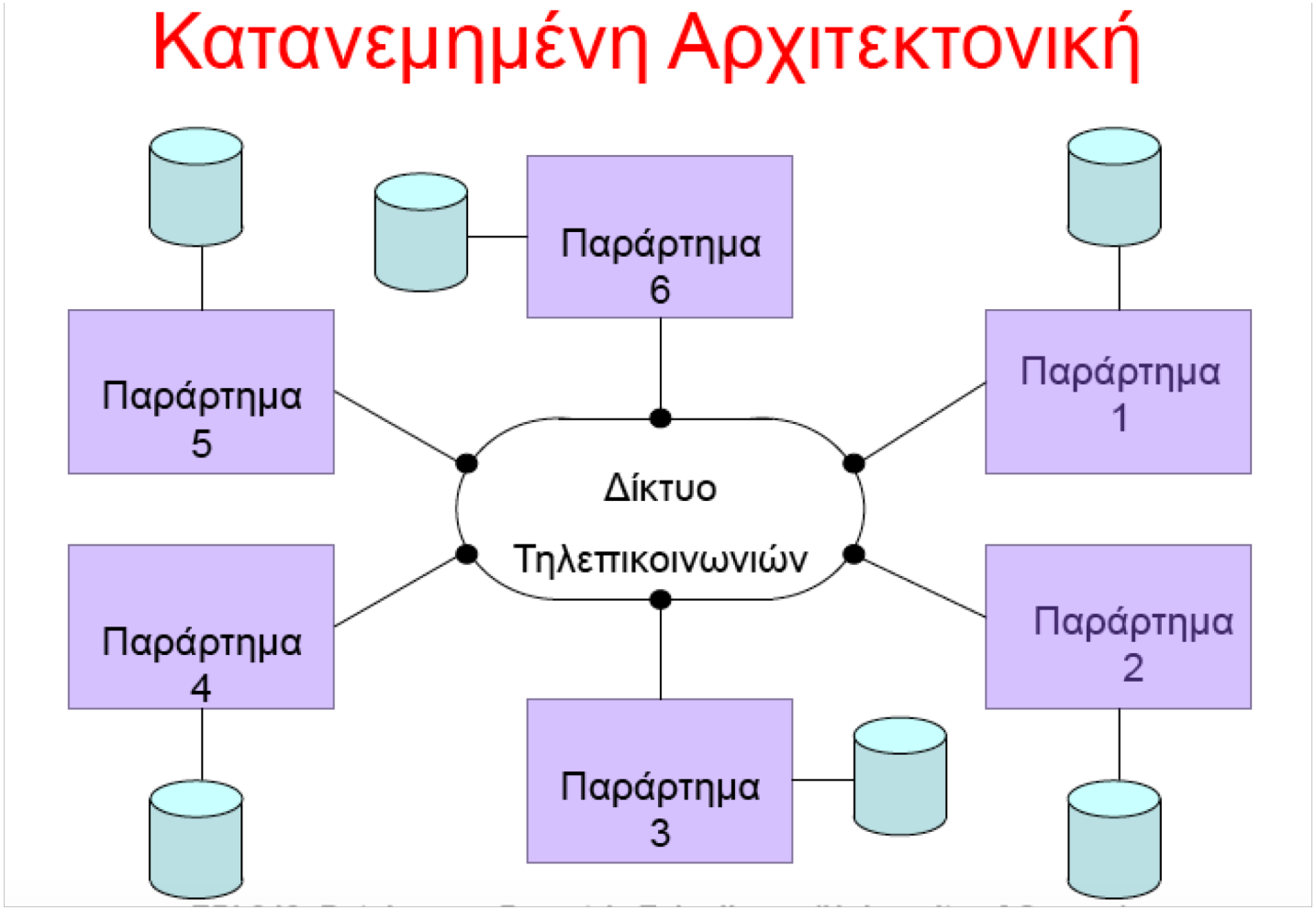
Figure 2.7
Logical three-tier client/server architecture, with a couple of commonly used nomenclatures.



Διασύνδεση μεταξύ DBMS 1/3



Διασύνδεση μεταξύ DBMS 2/3



Διασύνδεση μεταξύ DBMS 3/3

Κατανεμημένη Βάση (DDB)

– Μια **συλλογή** από **λογικά** συσχετιζόμενες αλλά **ανεξάρτητες** βάσεις δεδομένων προσβάσιμες πάνω από ένα **δίκτυο υπολογιστών**.

Κατανεμημένο Συσ. Διαχ. Βασ. Δεδ. (DDBMS)

– **γενικό λογισμικό σύστημα** το οποίο διαχειρίζεται κατανεμημένες βάσεις κάνοντας την **κατανομή** (δεδομένων, φόρτου, κτλ.,) **διαφανή** στο **χρήστη**.

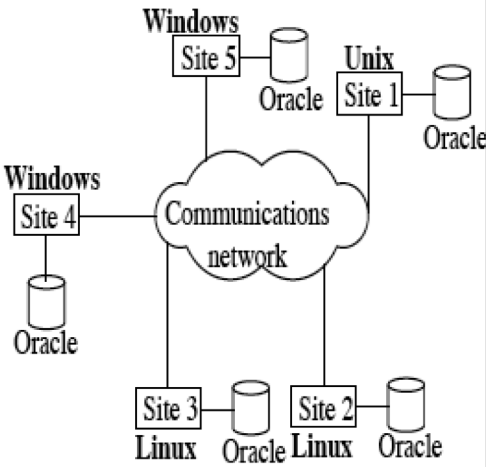
Οι περισσότερες βάσεις (εμπορικές και μη) **προσφέρουν** κάποιου είδους **δυνατότητες/επεκτάσεις** για να μετατραπεί η βάση σε **κατανεμημένη βάση δεδομένων.**

Βασικό πρόβλημα: Η έλλειψη κοινών αποδεκτών προτύπων. Η κατανομή περιορίζεται κυρίως μεταξύ ομογενών (όμοιων) βάσεων.

Κατανεμημένες βάσεις – Ομογενής

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.

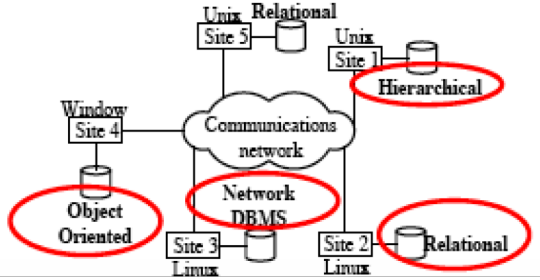


Oracle Βάση

Κατανεμημένες βάσεις – Ετερογενής

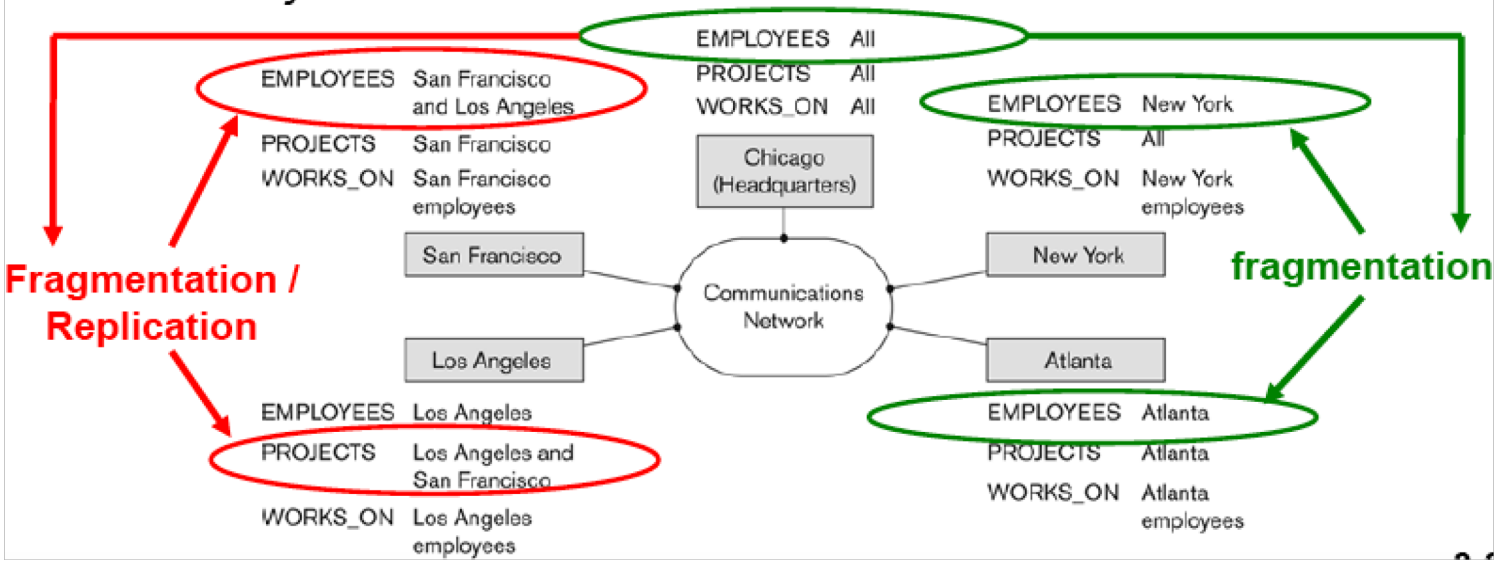
Ετερογενής (Heterogeneous): Διαφορετικά sites τρέχουν διαφορετικά DBMSs (ή ακόμη non-relational DBMSs).

- **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.



Κατανεμημένες βάσεις – βασικά θέματα

- 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:



Περίληψη

- Μοντέλα δεδομένων και οι κατηγορίες τους
- Ιστορία Μοντέλων Δεδομένων
- Σχήμα, Στιγμιότυπα, και καταστάσεις ΒΔ
- Αρχιτεκτονική Τριών-Σχημάτων
- Ανεξαρτησία Δεδομένων
- Γλώσσες και Διεπαφές ΣΔΒΔ
- Βοηθητικά προγράμματα και Εργαλεία Συστήματος ΒΔ
- Κεντρικές και Client-Server Αρχιτεκτονικές
- Κατανεμημένες Αρχιτεκτονικές