SELECT *lecture*
FROM **Databases**

WHERE chapter LIKE '*Basic Concepts*'

Mihaela Elena Breabăn'

Plan

- Key terms: Database, DBMS
- Some history and current trends

- The relational model
- RDBMSs architecture

Database

- A collection of logically related (operational) data

- Examples?
- Ideas on how to store/organize information?
- Think about efficiency and security!

Database Management System (DBMS)

- An environment that provides efficient and secure methods for data storing and retrieval to a wide range of users

- Components:
  - Hardware
  - Software
  - Data
  - Users/Roles

DBMS Features

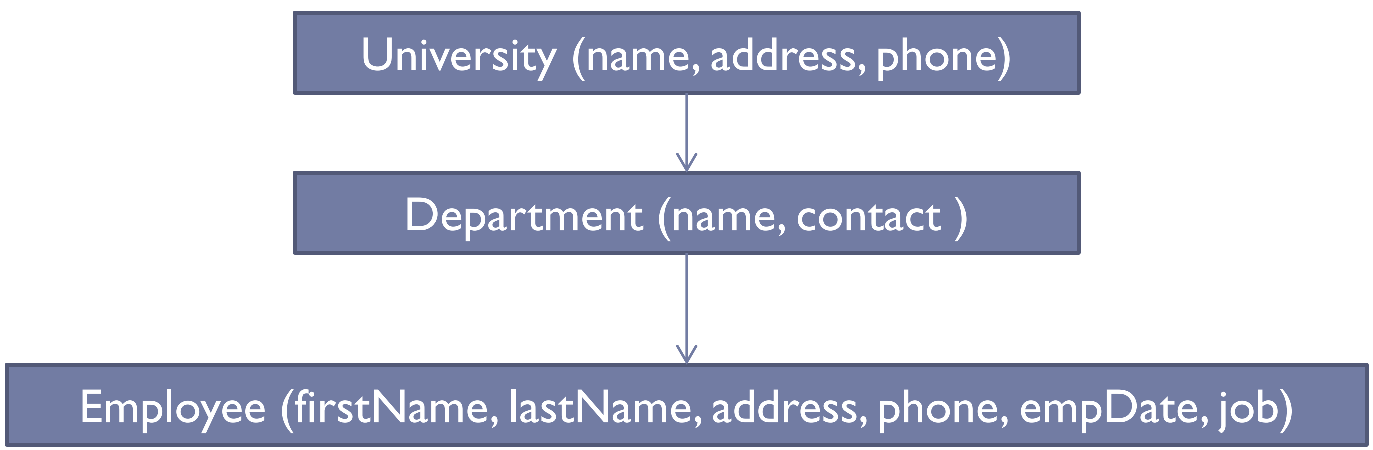
- Security
- Controlled database access
- Data storing, retrieval, update
- Integrity
- Transaction control
- Concurrent control
- Data backup and recovery
- Catalog (data dictionary)

DBMS History

- Hierarchical model (IBM's IMS, late '60s)
- Network Model (CODASYL 1969-1971)
- **Relational model (Codd, '70s)**
- Object-relational model ('90s)
- XML DBs (2000)
- NoSQL family

The First Decade B.C.*
Hierarchical and network DB systems

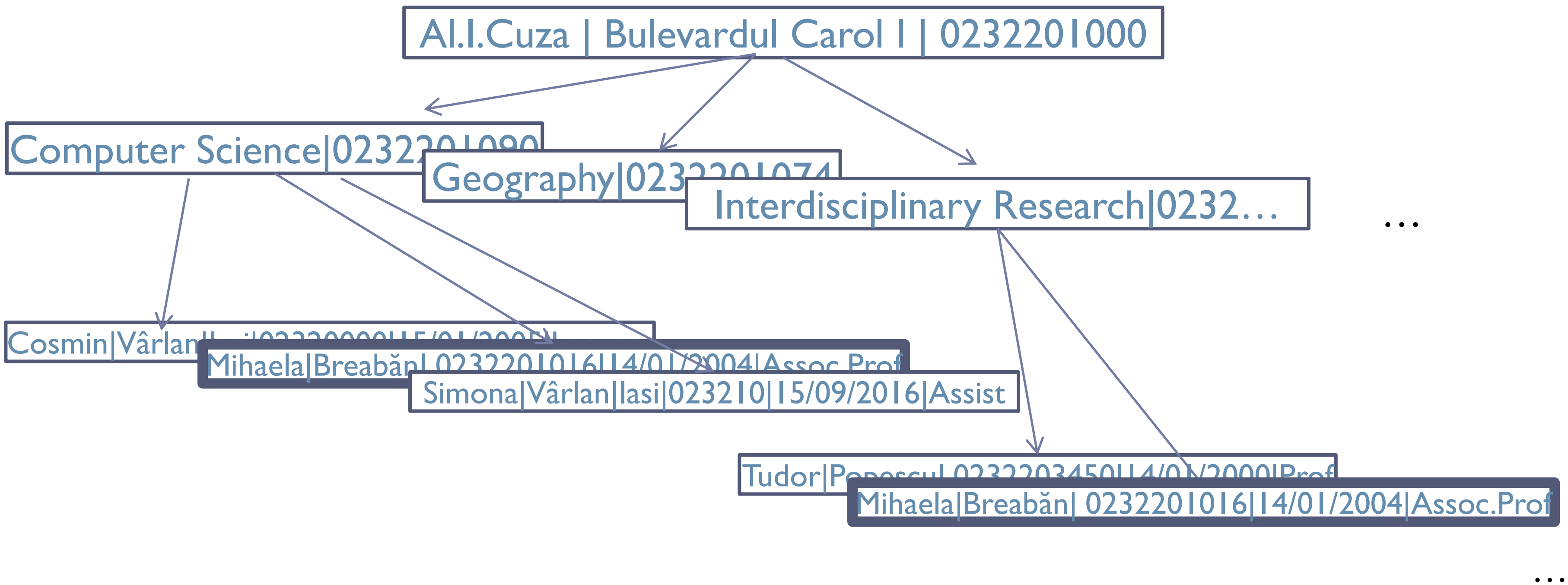
- Data organized as records which are stored as nodes in graphs
- *Navigational* APIs
  - Programmers had to (carefully!) scan or search for records, follow parent/child structures or pointers, and maintain code when anything physical changed



* inspired by *Michael J. Carey*'s talk at ICDE 2016: *Beyond Rows and Columns: Is the Fourth Time the Charm?*

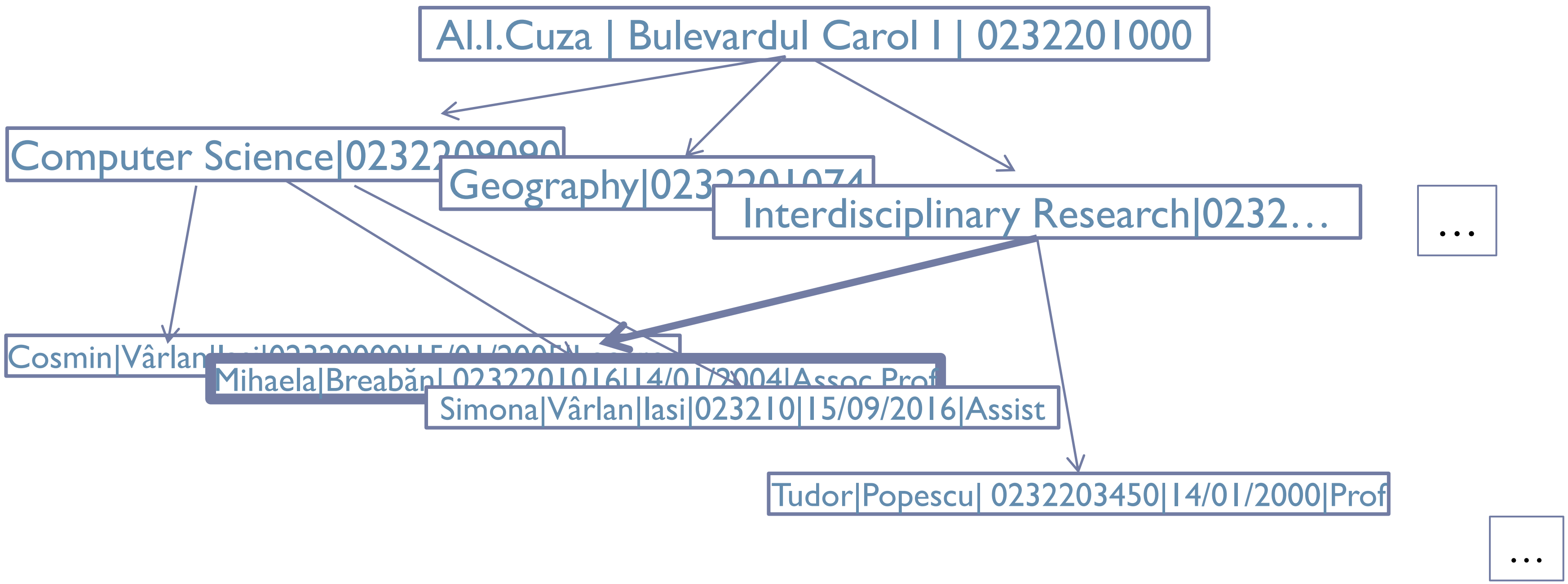
Hierarchical model (IBM's IMS, '60s)

- Able to model only one-to-many relationships



Network model
(Charles Bachman - CODASYL 1969-1971)

- Extends the hierarchical model by allowing many-to-many relationships



Relational model (Edgar Frank Codd – '70)

IBM's System R (SEQUEL)
Berkley's Ingres
Oracle, Postgres, SQLServer, MySQL…

- ***Everything is (logical) rows and columns!***
- ***Flat structures: columns are atomic (1NF)***
- ***Data INDEPENDENCE!***

1. Information Rule
2. Guaranteed Access Rule
3. Comprehensive Data Sub-language Rule
4. View Update Rule
5. High Level Insert, Update and Delete
6. Physical Data Independence
7. Logical Data Independence
8. Integrity Independence
9. Non Subversion Rule
10. Systematic Treatment of Null Values
11. Database Description Rule
12. Distribution Independence

- Components:
  - *Tables* as data structures
  - Constraints on data stored in tables
  - Logical relationships between tables (via Keys)
  - Methods to build new tables from existing ones (operators in relational algebra)

Relational model (cont.)

Departments

name	contact	deptID
Computer Science	0232201090	1
Geography	0232201074	2
Interdisciplinary Research	0232201102	3

Employees

firstName	lastName	address	phone	empDate	job	empID
Cosmin	Vârlan	Iasi	0232100000	15/01/2005	Lecturer	11
Mihaela	Breabăn	Iasi	0232201016	14/01/2004	Assoc.Prof	22
Simona	Vârlan	Iasi	0232100000	15/09/2016	Assist.	33
Tudor	Popescu	Iasi	0232203450	14/01/2000	Professor	44

Dept_Emp

deptID	empID
1	11
1	22
1	33
3	22
3	44

Object-relational model ('90s)

Postgres, Starburst, UniSQL, Illustra, DB2, Oracle

- Related to OO programming languages
  - Methods ("behavior") as well as data in the DBMS
  - *User-defined functions (UDTs/UDFs) & aggregates,* nested tables

Employees

Name		address	phone	empDate	job	empID
Cosmin	Vârlan	Iasi	0232200000	15/01/2005	Lecturer	11
Mihaela	Breabăn	Iasi	0232201016	14/01/2004	Assoc.Prof	22
Simona	Vârlan	Iasi	0232100000	15/09/2016	Assist.	33
Tudor	Popescu	Iasi	0232203450	14/01/2000	Professor	44

Departments

name	contact	Employees		
Computer Science	0232201090	11	22	33
Geography	0232201074			
Interdisciplinary Research	0232/201102	22	44	

XML databases (2000)

Natix, Timber, Ipedo, MarkLogic, BaseX; DB2, Oracle, SQL Server

- Flexible data model
  - Origins in document markup (SGML)
  - Nested data
  - Schema variety/optionality
- New declarative query language (XQuery)
  - Designed both for querying and transformation
  - Early standardization effort (W3C)

```
<Department id=1>
        <name>Computer Science</name>
        <contact>0232201090</contact>
        <employees>
                <employee> 11</employee>
                <employee> 22</employee>
                <employee> 33</employee>
        </employees>
…
```

```
<Employee id=11>
        <firstName>Mihaela</firstName>
        <lastName>Breabăn</lastName>
        <phone>0232201016</phone>
        <address>Iasi<address>
…
```

NoSQL databases (Not only SQL) …current investments…

- The movement defined by "what it's not"
- Triggered by the development of web
- Developed in the *distributed systems* community
- From "how to store it?" to "how to use it?"

- SQL is about traditional relational DBMS and not the SQL language
- NoSQL is not really about SQL, but developing data management systems that are not relational.

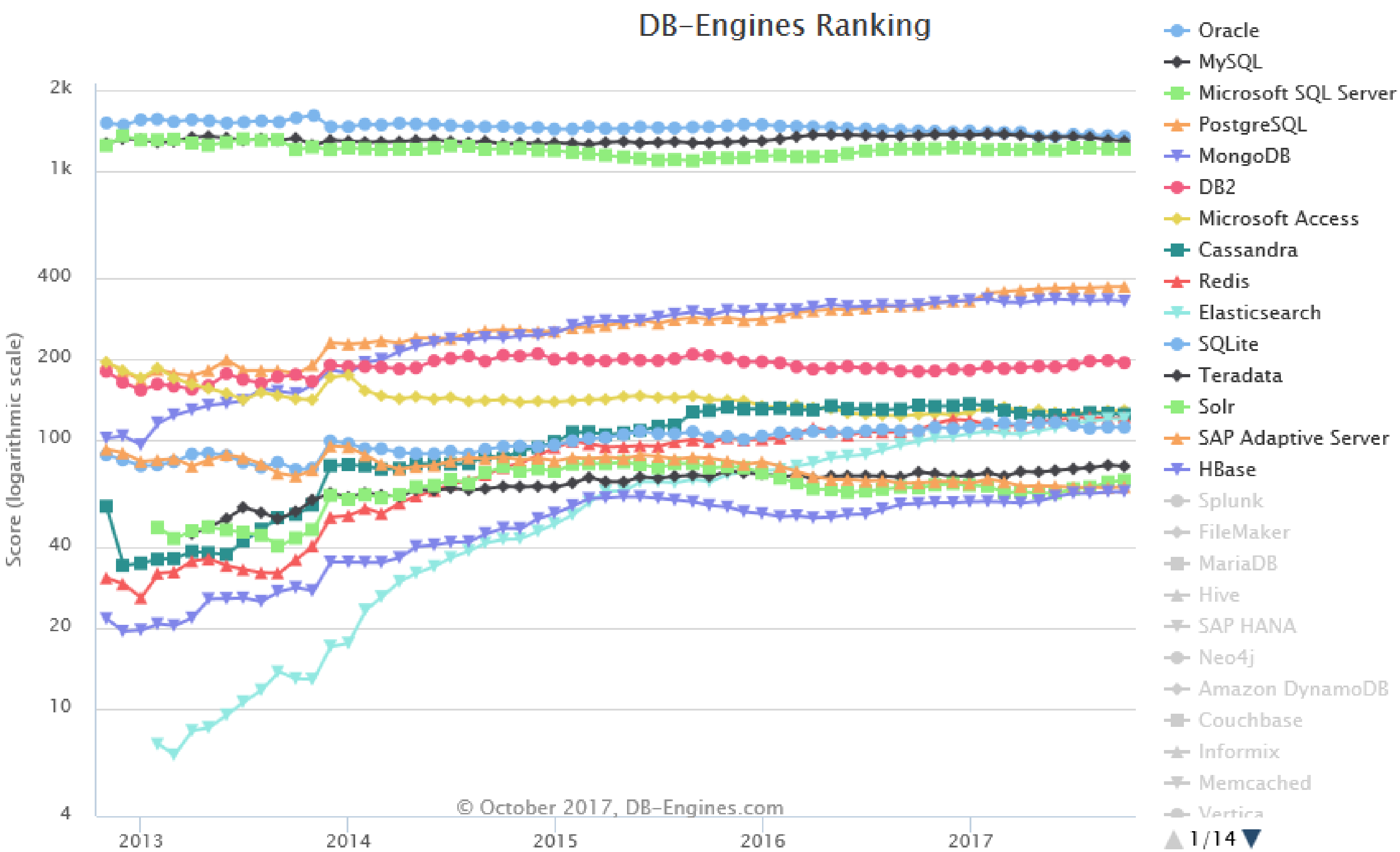
NoSQL systems

Two MySQL DBAs walk to a NoSQL bar. They left disappointed: they couldn't find any tables!

- MapReduce
  - No data model, data stored in files
  - User provides specific functions
  - System provides data processing "glue", fault-tolerance, scalability, designed for large scale analysis
- Key-value stores
  - ideal for retrieving specific data records from a large set of data
  - Data model: (key, value) pairs
  - Operations: Insert(key,value), Fetch(key), Update(key), Delete(key)
  - Examples: Google BigTable, Amazon Dynamo, Cassandra, Voldemort, HBase,
- Document stores
  - Data model: similar to key-value stores except value is a document in some form (e.g. JSON, xml), also Fetch based on document contents
  - Examples: CouchDB, MongoDB, SimpleDB
- Graph databases
  - Data model: nodes and edges; Nodes may have properties (including ID);Edges may have labels or roles
  - Examples: Neo4j, FlockDB, Pregel

back to RELATIONAL

If you still need a reason…



http://db-engines.com/en/ranking_trend

Relational databases
Concepts

- ***Relation*** = ***Table***
- ***Attribute*** = ***Column***
- ***Domain*** – the range of values allowed for an attribute
- ***Tuple*** = ***Record*** = ***Row*** in a table
- ***Relational database*** – a collection of relations with distinct names
- ***Relational schema*** – the definition of a relation: its name and its set of attributes with their domains
- ***Database schema*** - the set of relational schemas forming the database
- ***Database instance*** – content of a database at a given moment in time

Constraints for relations

- Attribute names are unique in a relational schema
- Every cell contains atomic values (1NF)
- The values of the attributes are constrained to the specified domains
- The order of the tuples is not significant
- (There are no duplicate tuples)

Keys

- ***Superkey*** – an atributte or a set of attributes that uniquely identifies a tuple in a relation
- ***Candidate key*** – a superkey for which no proper subset is also a superkey
- ***Primary key***– a candidate key designated by the database designer to uniquely identify the tuples in a relation
- ***Alternate key*** – candidate key that was not selected as primary
- ***Foreign key*** – an attribute or a set of attributes in a relation that references a candidate key in another relation

Integrity constraints

- Primary keys cannot contain NULLs
- The value of a foreign key must match at least one tuple in the referenced relation; otherwise it must be NULL
- More on constraints in lecture 8…

Views

- Relations/tables have their tuples/records physically stored in the database
- The VIEW looks and behaves like a relation/table but is just the result of some operations (queries) on existing tables.
- They are a mechanism for implementing Codd's 7th requirement (data independence)

RDBMS components
Hardware

- Requirements
  - Persistent data
  - Large volumes
  - Quick access

- Ranges from a standard PC to a server or a network (distributed database)

RDBMS components
Software

- Ensures the interaction between users and data by means of query languages:
  - SQL-DDL (data definition language)
    - Responsible for meta-data
  - SQL-DML (data manipulation language)
    - Responsible for storing, updating, retrieving operational data

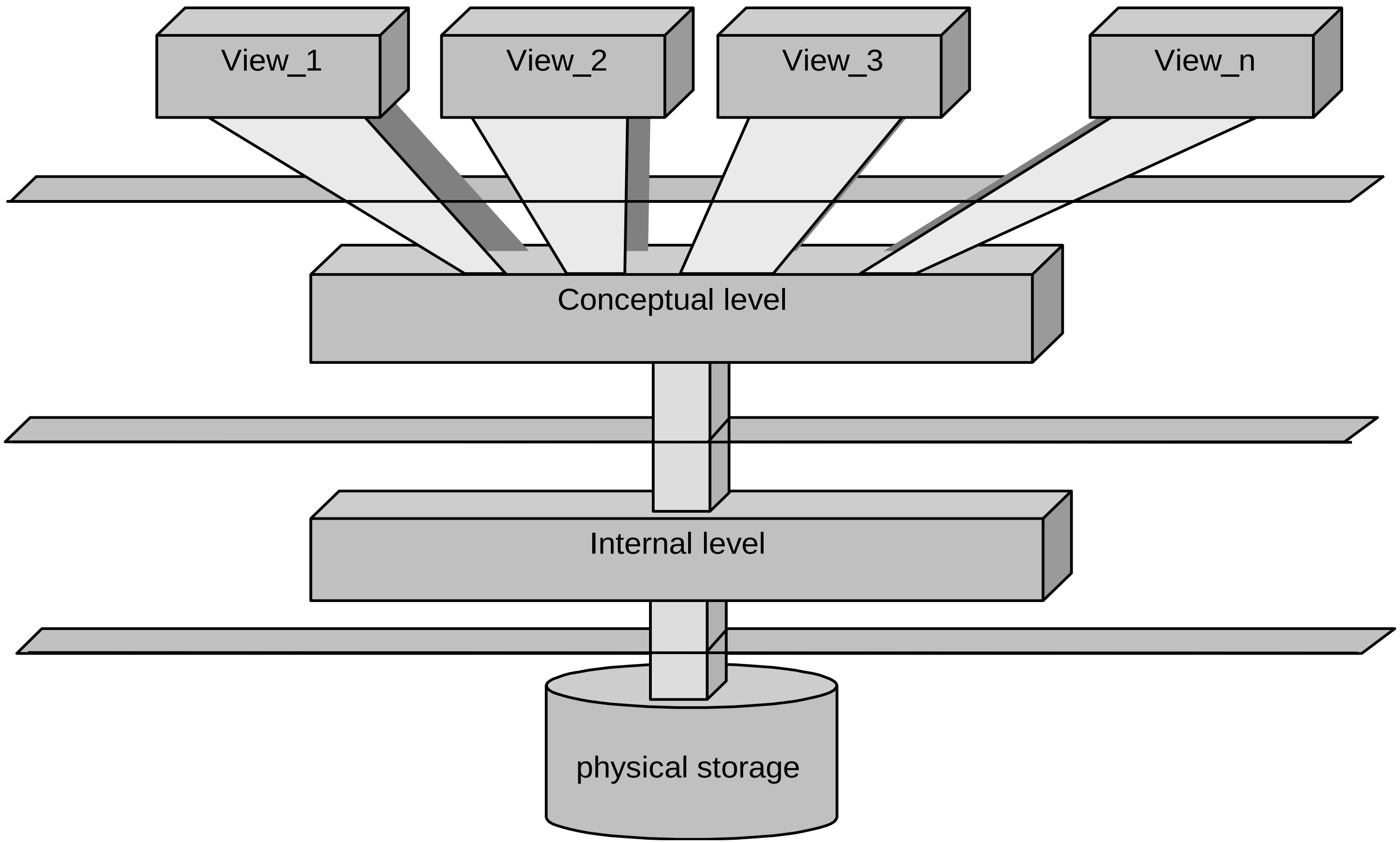
- Non-procedural approach

RDBMS components Users/Roles

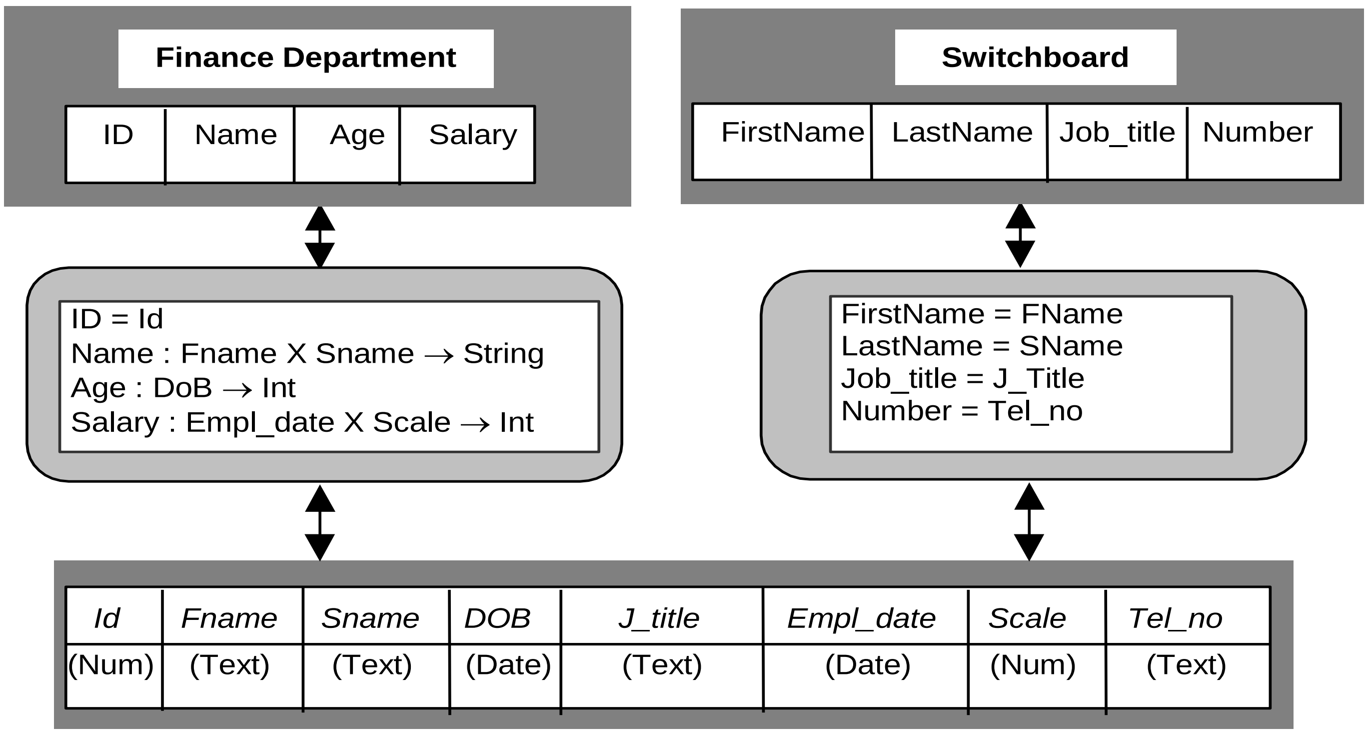
- *Database administrators (DBAs)*
- *Database designer*
- *Application programmers*
- *End users (SQL)*

☺ *Nobody knows what a DBA does, but every company needs to hire one, because no one can afford to hire two.*

ANSI-SPARC architecture of RDBMSs



External level/conceptual level mapping



Example from Thomas Connolly, Caroline Begg: "*Database Systems. A practical approach to design, implementation and management*". Ed. Addison Wesley

Conceptual level/internal level mapping

Id	*Fname*	*Sname*	*DOB*	*J_title*	*Empl_date*	*Scale*	*Tel_no*
(Num)	(Text)	(Text)	(Date)	(Text)	(Date)	(Num)	(Text)

Table_Employees <*implemented as*>
ARRAY[n] OF struct STAFF

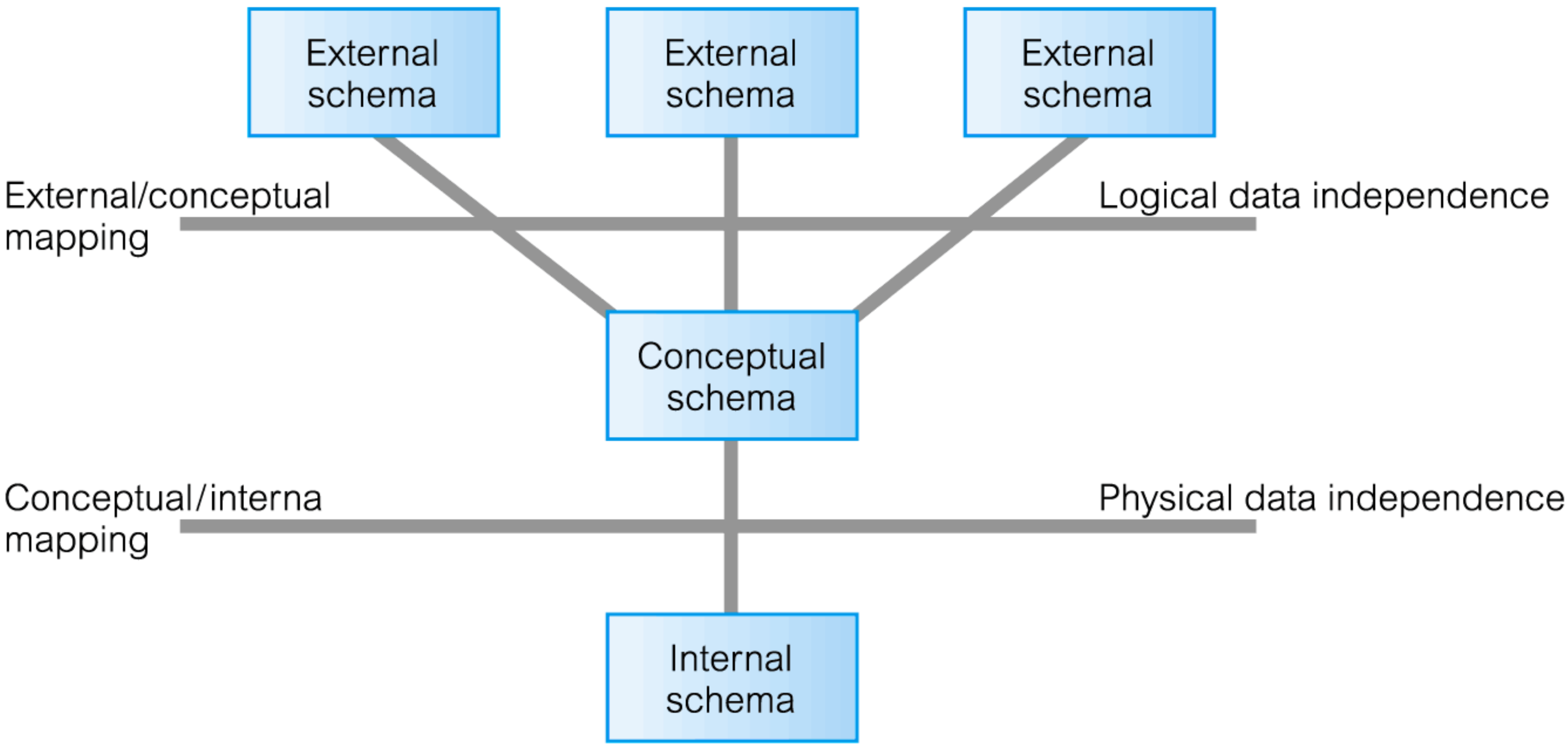
```
struct STAFF Table_Employees [5000];

struct STAFF {
        int         ID;
        char        Fname[20];
        // ...
        char        Tel_no[15];
};

struct INDEXS {
        int         ID;
        int         Index ;
} Index_Employees [n];
```

the information about staff is **physically** implemented by means of an array

other structures, not derived from the logical level, might be used at the physical level (e.g. indexes)

Example from Thomas Connolly, Caroline Begg: "*Database Systems. A practical approach to design, implementation and management*". Ed. Addison Wesley

Architectural schemas



RDBMSs Features

- Data consistency
  - As opposed to *eventual consistency* in NoSQL systems
  - Constraints on data, referential integrity
- Sharing of data
- Support for transactions (ACID over BASE properties)
- Improved data accessibility and efficiency
  - Declarative queries (SQL)
  - Views
  - Transparent indexing (physical data independence)
  - Query optimization and execution
- Increased concurrency
- Improved security
- Backup and recovery services

Bibliography

- Thomas Connolly, Caroline Begg: "*Database Systems. A practical approach to design, implementation and management*". Ed. Addison Wesley

- E. F. Codd: *A Relational Model of Data for Large Shared Data Banks*. CACM 13(6): 377-387 (1970)
- E. F. Codd(1985). "Is Your DBMS Really Relational?" and "Does Your DBMS Run By the Rules?" *ComputerWorld*, October 14 and October 21.
- E. F. Codd. 1990. *The Relational Model for Database Management: Version 2*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

- http://computing.derby.ac.uk/c/codds-twelve-rules/