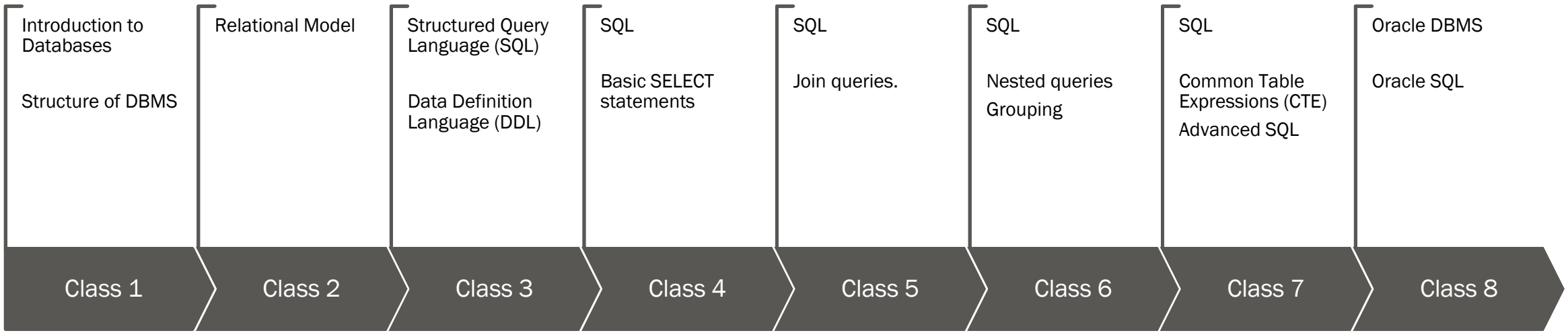
INTRODUCTION TO DATABASES USING ORACLE

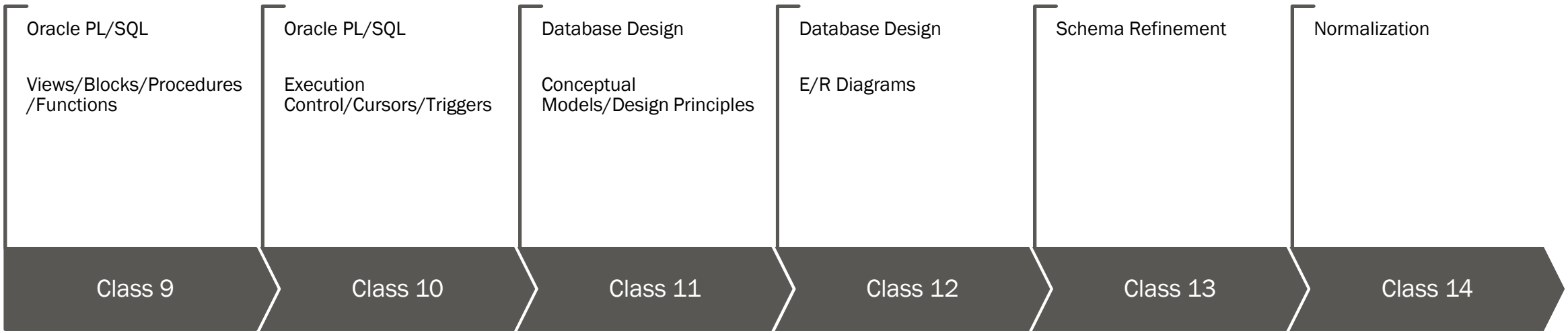
420-983-VA

INTRODUCTION

COURSE STRUCTURE



COURSE STRUCTURE

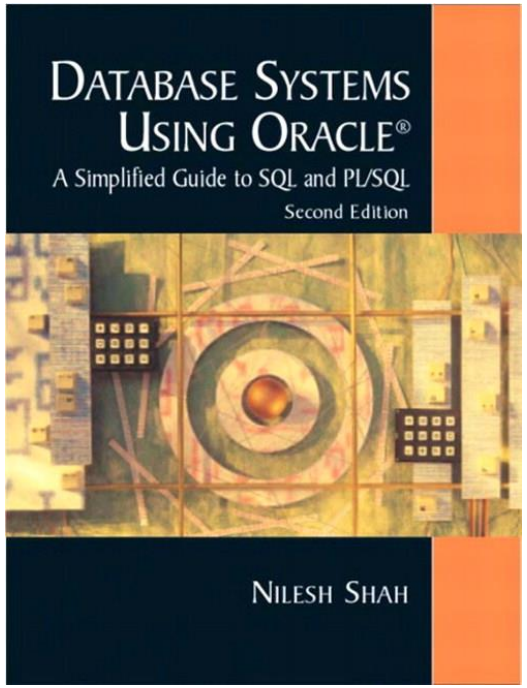


COURSE EVALUATION PROCEDURES

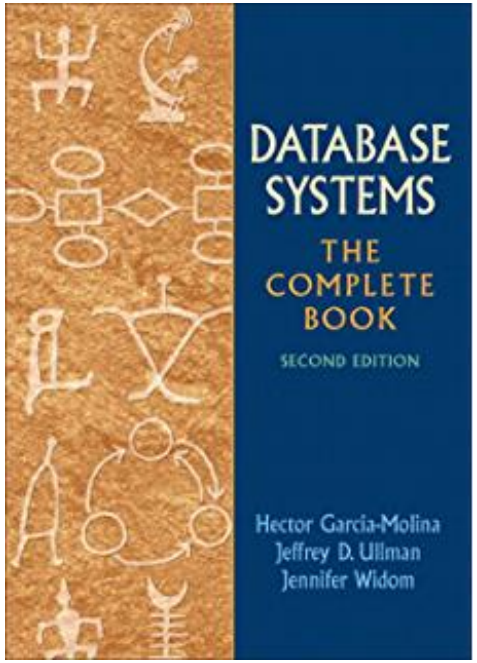
Evaluation	Weight
Labs	**10%**
Assignment 1	**15%**
Assignment 2	**15%**
Mid-Project	**30%**
Final-Project	**30%**

	Posted	Submitted
Assignment 1	: Week 1	Week 3
Assignment 2	: Week 3	Week 5
Mid Project	: Week 1	Week 4
Final Project	: Week 2	Week 5

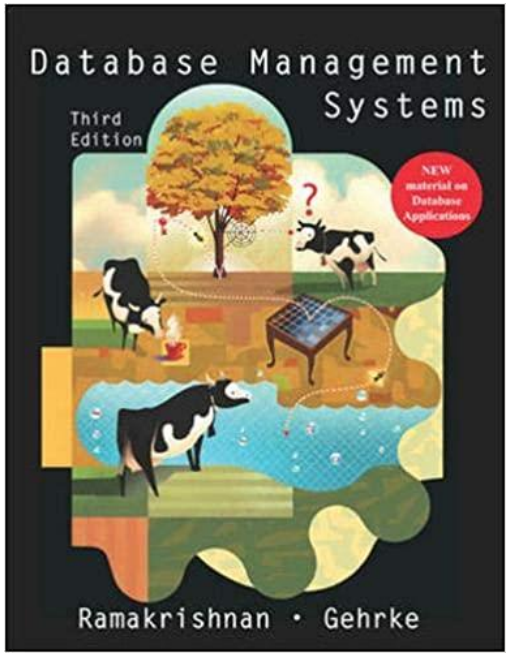
COURSE BIBLIOGRAPHY



Database Systems Using Oracle: A Simplified Guide to SQL and PL/SQL (2nd Edition)
Nilesh Shah
Publisher : Pearson; (May 13 2004)
ISBN-10 : 0131018574



Database Systems: The Complete Book
Hector Garcia-Molina, Jeffrey D. Ullman and Jennifer Widom
Publisher : Pearson; 2nd edition (June 5 2008)
ISBN-10 : 0131873253



Database Management Systems
Raghu Ramakrishnan, Johannes Gehrke
Publisher : McGraw-Hill Education; 3nd edition (August 14 2002)
ISBN-10 : 0072465638

WHAT IS A DATABASE?

- A collection of information that exists over a long period of time, often many years.
- The term database refers to a collection of data that is managed by a DBMS (Database Management System)

WHAT IS A DBMS?

- A very large, integrated collection of data.
- Models real-world enterprise.
    - Entities (e.g., students, courses)
    - Relationships (e.g., Madonna is taking CS564)
- A Database Management System (DBMS) is a software package designed to store and manage databases.

FILE VS. DBMS

A company has a large collection (say, 500 GB) of data on employees, departments, products, sales, and so on. This data is accessed concurrently by several employees. Questions about the data must be answered quickly, changes made to the data by different users must be applied consistently, and access to certain parts of the data (e.g., salaries) must be restricted.

What drawbacks do you see by storing this data in operating system files?

FILE VS. DBMS

- We probably do not have 500 GB of main memory to hold all the data. We must therefore store data in a storage device such as a disk or tape and bring relevant parts into main memory for processing as needed.

- We have to write special programs to answer each question a user may want to ask about the data. These programs are likely to be complex because of the large volume of data to be searched.

FILE VS. DBMS

- We must protect the data from inconsistent changes made by different users accessing the data concurrently. If applications must address the details of such concurrent access, this adds greatly to their complexity.

- We must ensure that data is restored to a consistent state if the system crashes while changes are being made.

- Operating systems provide only a password mechanism for security. This is not sufficiently flexible to enforce security policies in which different users have permission to access different subsets of the data.

WHY USE A DBMS ?

- Allow users to create new databases and specify their schemas (logical structure of the data), using a specialized data-definition language (DDL).
- Give users the ability to query the data (a “query” is database lingo for a question about the data) and modify the data, using an appropriate language, often called a query language or data-manipulation language (DML).

WHY USE A DBMS ?

- Support the storage of very large amounts of data over a long period of time, allowing efficient access to the data for queries and database modifications.
- Enable durability, the recovery of the database in the face of failures, errors of many kinds, or intentional misuse.
- Control access to data from many users at once, without allowing unexpected interactions among them (isolation) and without actions on the data to be performed partially but not completely (atomicity).

A HISTORICAL PERSPECTIVE

early 1960s – first general-purpose DBMS, designed by Charles Bachman at General Electric, called integrated data store. It formed the basis for the network data model, which was standardized by the Conference on Data Systems Languages (CODASYL). Bachman was the first recipient of ACM's Turing Award (the computer science equivalent of a Nobel Prize) in 1973.

late 1960s – IBM developed the Information Management System (IMS) DBMS, used even today in many major installations. IMS formed the basis for an alternative data representation framework called the hierarchical data model.

A HISTORICAL PERSPECTIVE

1970s – Edgar Codd, at IBM's San Jose Research Laboratory, proposed a new data representation framework called the relational data model. It sparked the rapid development of several DBMSs based on the relational model, along with a rich body of theoretical results that placed the field on a firm foundation. Codd won the 1981 Turing Award for his seminal work. Database systems matured as an academic discipline, and the popularity of relational DBMSs changed the commercial landscape.

1980s – The SQL query language for relational databases, developed as part of IBM's System R project, is now the standard query language. SQL was standardized in the late 1980s

A HISTORICAL PERSPECTIVE

1990s – Considerable research was carried out into more powerful query languages and richer data models, with emphasis placed on supporting complex analysis of data from all parts of an enterprise. Several vendors (e.g., IBM's, DB2, Oracle 8, Informix 2 UDS) extended their systems with the ability to store new data types such as images and text, and to ask more complex queries. Specialized systems have been developed by numerous vendors for creating data warehouses, consolidating data from several databases, and for carrying out specialized analysis.

2000s – DBMSs have entered the Internet Age. While the first generation of websites stored their data exclusively in operating systems files, the use of a DBMS to store data accessed through a Web browser is becoming widespread. Queries are generated through Web-accessible forms and answers are formatted using a markup language such as HTML to be easily displayed in a browser. All the database vendors are adding features to their DBMS aimed at making it more suitable for deployment over the Internet.

ADVANTAGES OF A DBMS

- Data Independence: Application programs should not, ideally, be exposed to details of data representation and storage, The DBMS provides an abstract view of the data that hides such details

- Efficient Data Access: A DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently. This feature is especially important if the data is stored on external storage devices.

- Data Integrity and Security: If data is always accessed through the DBMS, the DBMS can enforce integrity constraints. For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded. Also, it can enforce access controls that govern what data is visible to different classes of users.

ADVANTAGES OF A DBMS

- Data Administration: When several users share the data, centralizing the administration of data can offer significant improvements. Experienced professionals who understand the nature of the data being managed, and how different groups of users use it, can be responsible for organizing the data representation to minimize redundancy and for fine-tuning the storage of the data to make retrieval efficient.

- Concurrent Access and Crash Recovery: A DBMS schedules concurrent accesses to the data in such a manner that users can think of the data as being accessed by only one user at a time. Further, the DBMS protects users from the effects of system failures.

- Reduced Application Development Time: Clearly, the DBMS supports important functions that are common to many applications accessing data in the DBMS.

DESCRIBING AND STORING DATA IN A DBMS

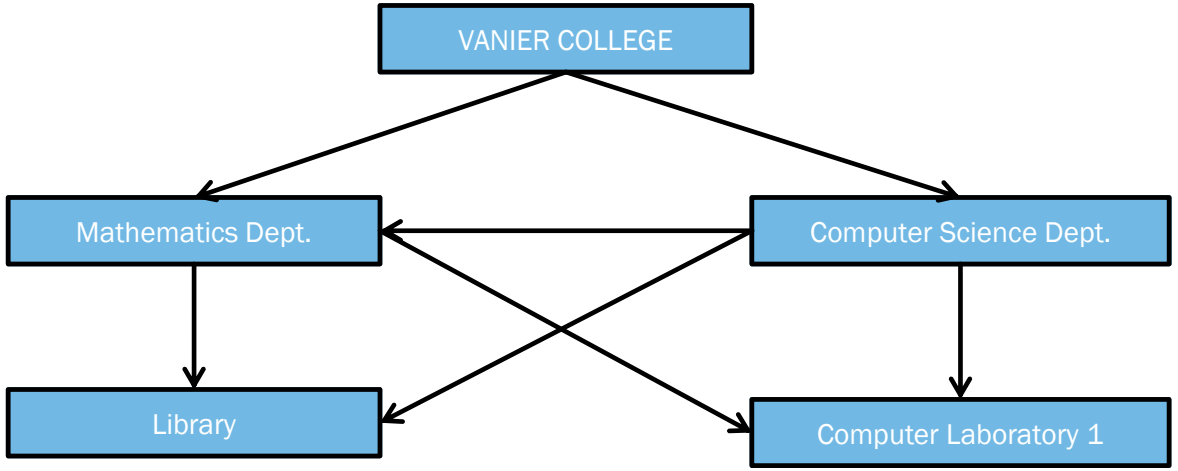
The user of a DBMS is ultimately concerned with some real-world enterprise, and the data to be stored describes various aspects of this enterprise. For example, there are students, faculty, and courses in a university, and the data in a university database describes these entities and their relationships.

A data model is a collection of high-level data description constructs that hide many low-level storage details. A DBMS allows a user to define the data to be stored in terms of a data model. Most database management systems today are based on the relational data model.

A semantic data model is a more abstract, high-level data model that makes it easier for a user to come up with a good initial description of the data in an enterprise. A widely used semantic data model called the entity-relationship (ER) model allows us to pictorially denote entities and the relationships among them.

THE NETWORK MODEL

In this model the schema is viewed as a graph in which object types are nodes and relationship types are arcs. The graph is not restricted to being a hierarchy or lattice. The network model replaces the hierarchical model with a graph thus allowing more general connections among the nodes. The main difference of the network model from the hierarchical model is its ability to handle many to many relationships. In other words it allow a record to have more than one parent.



THE NETWORK MODEL

Advantages

Conceptual simplicity - Just like the hierarchical model, the network model is also conceptually simple and easy to design.

Capability to handle more relationship types - The network model can handle the one to many and many to many relationships which is real help in modeling the real life situations.

Ease of data access - The data access is easier and flexible than the hierarchical model.

Data integrity - The network model does not allow a member to exist without an owner.

Data independence - The network model is better than the hierarchical model in isolating the programs from the complex physical storage details.

Database standards

Disadvantages

System complexity - All the records are maintained using pointers and hence the whole database structure becomes very complex.

Operational Anomalies - The insertion, deletion and updating operations of any record require large number of pointers adjustments.

Absence of structural independence - structural changes to the database is very difficult.

THE NETWORK MODEL

Existing implementations:

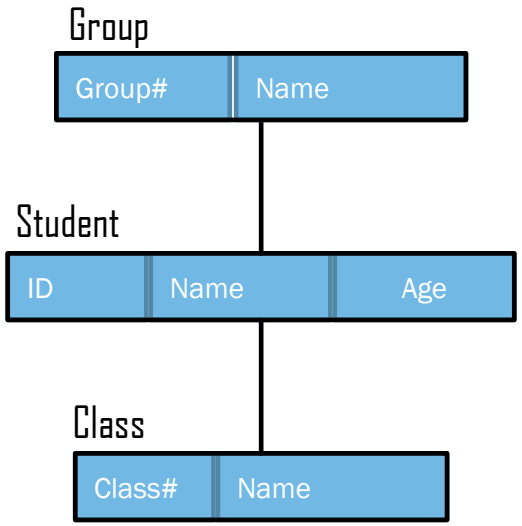
IDS (**I**ntegrated **D**ata **S**tore), Charles Bachman, 1965 - 1975

IDMS (**I**ntegrated **D**atabase **M**anagement **S**ystem), Computer Associates. Started from IDs, some IDMS versions are still running today on IBM mainframes.

THE HIERARCHICAL MODEL

Data is organized in a tree-like structure. Data is stored as records connected to each other through links. A record is a collection of fields, each field containing only one value. The type of a record defines which fields the record contains.

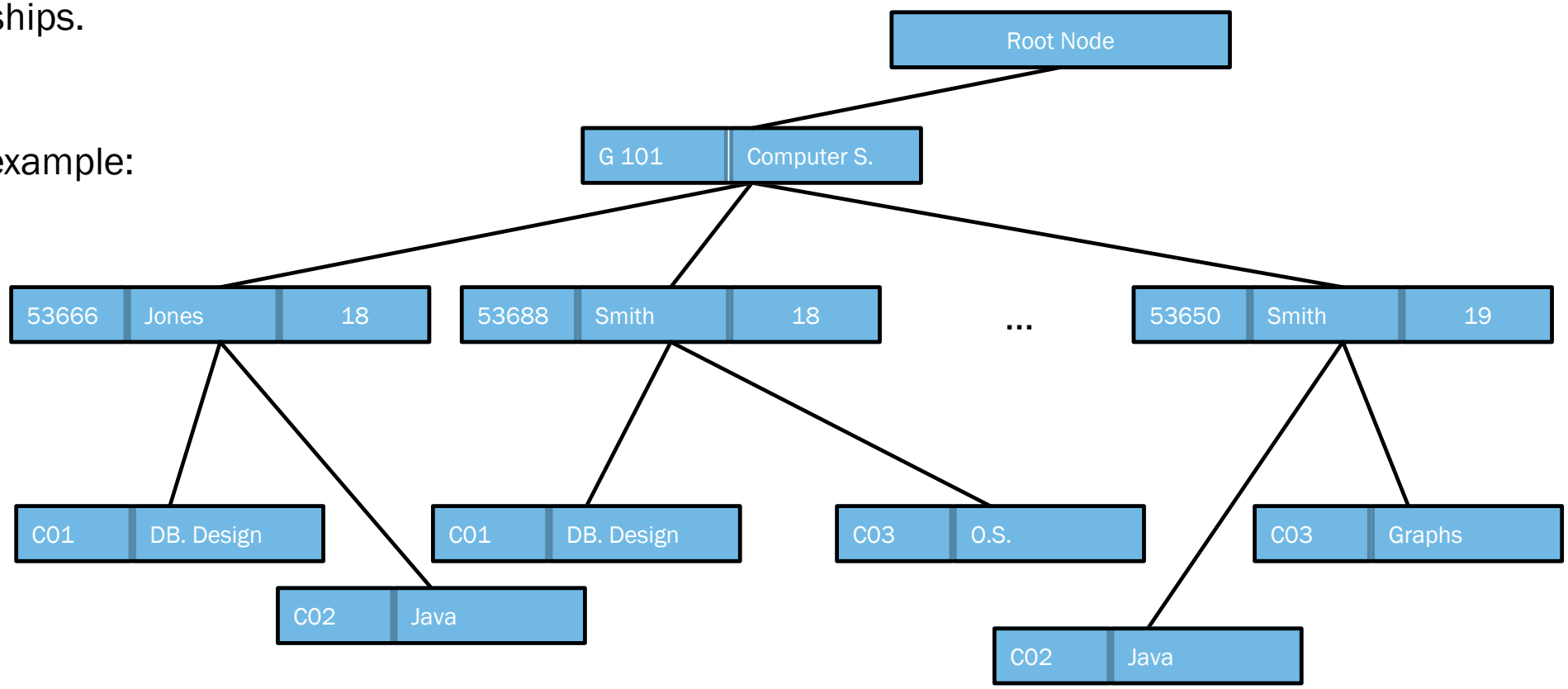
Simple example:



THE HIERARCHICAL MODEL

A hierarchical database model must have only one parent for each child node but parent nodes can have more than one child. Multiple parents are not allowed. The first node of the tree is called the root node. When data needs to be retrieved then the whole tree is traversed starting from the root node. This model represents one- to- many relationships.

Simple example:



THE HIERARCHICAL MODEL

Advantages

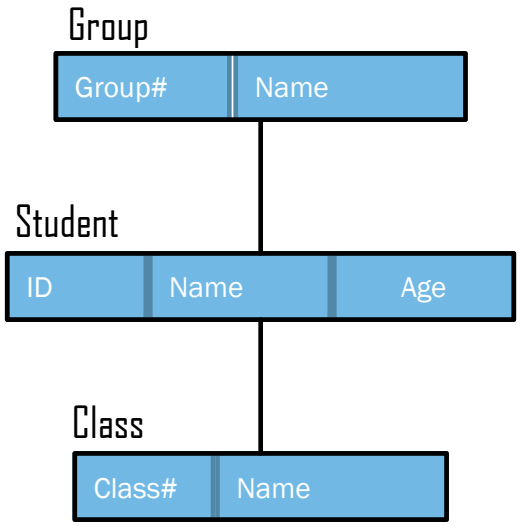
- Data can be retrieved easily due to the explicit links present between the table structures.
- Referential integrity is always maintained i.e. any changes made in the parent table are automatically updated in a child table.
- Promotes data sharing.
- It is conceptually simple due to the parent-child relationship.
- Database security is enforced.
- Efficient with 1: N relationships.
- A clear chain of command or authority.
- Increases specialization.
- High performance.
- Clear results.

Disadvantages

- If the parent table and child table are unrelated then adding a new entry in the child table is difficult because additional entry must be added in the parent table.
- Complex relationships are not supported.
- Redundancy which results in inaccurate information.
- Change in structure leads to change in all application programs.
- M: N relationship is not supported.
- No data manipulation or data definition language.
- Lack of standards.
- Poor flexibility.

THE HIERARCHICAL MODEL

Existing implementations: IBM's IMS (Information Management System).



This structure can be specified in IMS as follow:

```
1   DBD NAME=EDUBD
2   SEGM NAME=Group,BYTES=256
3   FIELD NAME=(Group#,SEQ),BYTES=3,START=1
4   FIELD NAME=Name,BYTES=33,START=4
5   SEGM NAME=Student,PARENT=Group,BYTES=38
6   FIELD NAME=(ID,SEQ),BYTES=3,START=1
7   FIELD NAME=Name,BYTES=33,START=4
8   FIELD NAME=Age,BYTES=2,START=37
9   SEGM NAME=Class,PARENT=Student,BYTES=36
10  FIELD NAME=(Class#,SEQ),BYTES=3,START=1
11  FIELD NAME=Name,BYTES=33,START=4
```

THE RELATIONAL MODEL

A description of data in terms of a data model is called a schema. In the relational model, the schema for a relation specifies its name, the name of each field (or attribute or column), and the type of each field.

As an example, student information in a university database may be stored in a relation with the following schema:

Students(*sid*: string, *name*: string, *login*: string, *age*: integer, *gpa*: real)

Below is an example of an instance of the Students relation:

SID	NAME	LOGIN	AGE	GPA
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	John	john@music	20	1.8
53832	Phil	phil@music	18	2

THE RELATIONAL MODEL

SID	NAME	LOGIN	AGE	GPA
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	John	john@music	20	1.8
53832	Phil	phil@music	18	2

Each row in the Students relation is a record that describes a student. Every row follows the schema of the Students relation. The schema can therefore be regarded as a template for describing a student.

THE RELATIONAL MODEL

SID	NAME	LOGIN	AGE	GPA
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	John	john@music	20	1.8
53832	Phil	phil@music	18	2

We can make the description of a collection of students more precise by specifying integrity constraints, which are conditions that the records in a relation must satisfy. For example, we could specify that every student has a unique SID value. Observe that we cannot capture this information by simply adding another field to the Students schema.

Relational model is used in numerous systems: DB2, Informix, Oracle, SQL Server, Sybase, Microsoft Access, Teradata, MySQL,...

DESCRIBING AND STORING RELATIONAL DATA

For example, there are students, faculty, and courses in a university, and the data in a university database describes these entities and their relationships, as below:

SID	NAME	LOGIN	AGE	GPA
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	John	john@music	20	1.8
53832	Phil	phil@music	18	2

Is this a good design?

DESCRIBING AND STORING RELATIONAL DATA

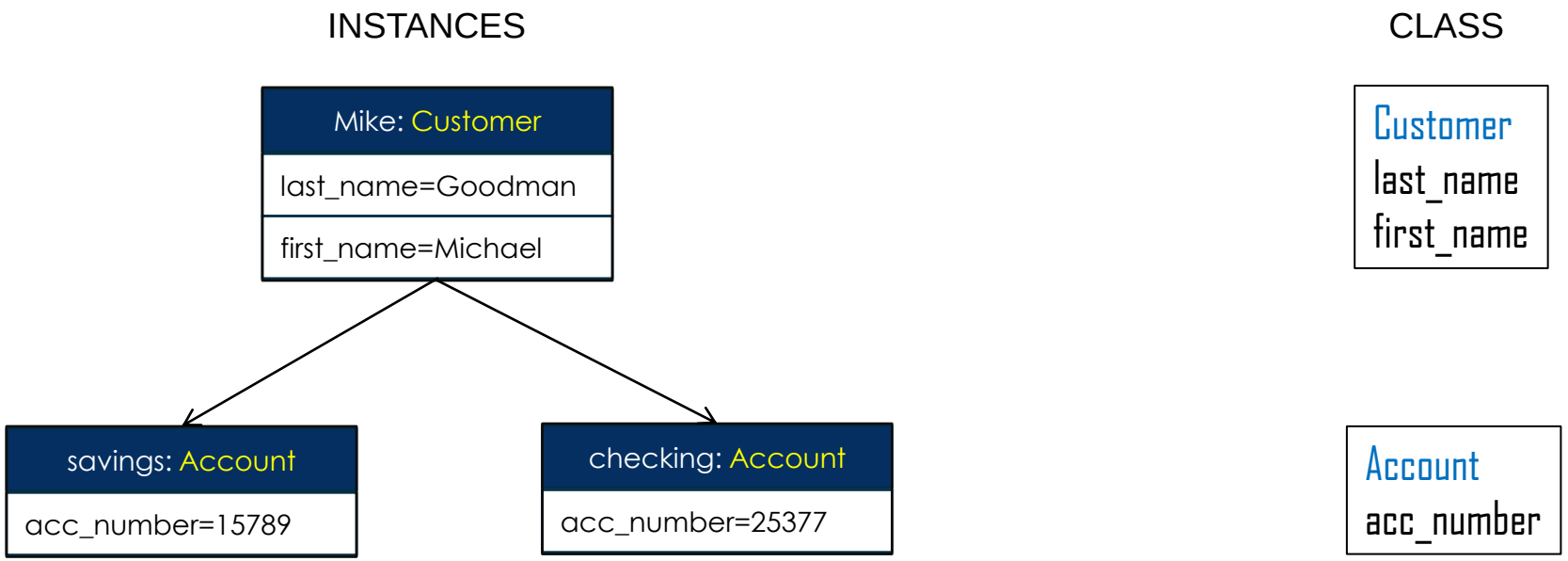
For example, there are students, faculty, and courses in a university, and the data in a university database describes these entities and their relationships, as below:

SID	NAME	LOGIN	AGE	GPA
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith@math	19	3.8
53831	John	john@music	20	1.8
53832	Phil	phil@music	18	2

Poor Design: You should not create a field such as age, whose value is constantly changing. A better choice would be date of birth; if needed age can be computed from this.

THE OBJECT-ORIENTED MODEL

Information is represented in the form of objects as used in Object-Oriented programming. Object-oriented databases allow definition of objects, which are different from database objects.



THE OBJECT-ORIENTED MODEL

Existing Implementations:

1. ObjectStore (1988 -)
2. Versant (1988 -)
3. ObjectDB (2003 -)

THE OBJECT-ORIENTED MODEL

An Object relational model is a combination of a Object oriented database model and a Relational database model. So, it supports objects, classes, inheritance etc. just like Object Oriented models and has support for data types, tabular structures etc. like Relational data model.

Advantages.

- Inheritance - allows its users to inherit objects, tables etc. so that they can extend their functionality..
- Complex Data Types - complex data types can be formed using existing data types.
- Extensibility - functionality of the system can be extended in Object relational data model

Disadvantages.

- Complicated - data model can get quite complicated and difficult to handle at times as it is a combination of the Object oriented data model and Relational data model and utilizes the functionalities of both of them.

THE OBJECT-RELATIONAL MODEL

Existing Implementations:

1. Illustra – a fork from Postgres, currently part of Informix

2. Oracle