CompSci 516
Data Intensive Computing Systems

Lecture 1
Introduction and Data Models

Instructor: Sudeepa Roy

Course Website

- http://www.cs.duke.edu/courses/spring16/compsci516/

- Please check frequently for updates

Instructor

- Sudeepa Roy
  - sudeepa@cs.duke.edu
  - https://users.cs.duke.edu/~sudeepa/
  - office hour: Tuesdays 4:30-5:30 pm, LSRC D325

- About myself
  - Assistant Professor in CS
  - Joined in Fall 2015
  - Research interests:
    - Databases (theory and applications)
    - Data Analysis, causality, explaining answers
    - Uncertain data, crowd sourcing, data provenance

(½) TAs

- Junghoon Kang
  - jungkang@cs.duke.edu
  - office hour: Tuesdays 10:20-11:20 am, North N303B
- Xiaodan Zhu
  - xdzhu@cs.duke.edu
  - office hour: Wednesdays 3:20-4:20 pm, LSRC D301

Grading

- Five Homework: 35%
- Midterm: 25%
- Final: 35%
- Class Participation: 5%

Logistics

- Homework submission: Sakai
  - All enrolled and waitlisted students are already there
- Discussion forum: Piazza
  - All enrolled students are already there
  - Send me an email if you are enrolled in class (not waitlisted), but have not received a welcome email from Piazza

Homework

- Due in two weeks after they are posted
- No late days – contact the instructor if you have a *valid* reason to be late
- To be done individually
- Mix of coding and problem solving by hand
- The first homework will be about installing a DBMS, loading data into it, and running queries
  - to be released by the next class

Exams

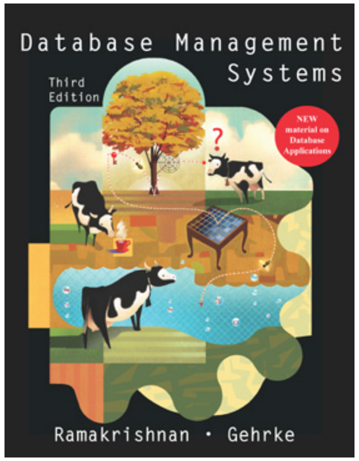
- Midterm – March 1
- Final – May 7

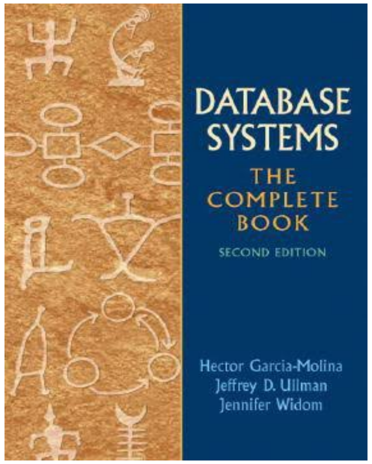
- In class
- Closed book, closed notes, no electronic devices
- Total weight: 25 + 35 % = 60 %
- You need to learn the material for the exams

Class Participation

- Ask questions
  - In class
  - On Piazza (as public whenever possible)
- Answer others' questions
- Try to avoid looking at laptop/phones in class unless you have to
- Please send (anonymous) feedback, suggestions, or concerns on Piazza
- You effort will be taken into account while deciding borderline cases for grades

Reading Material


Database Management Systems
Third Edition
NEW material on Database Applications
Ramakrishnan · Gehrke



- Book chapters will be posted
- Other books and research papers -- will be posted on the website
- No need to buy the books, but it will be good to consult them from time to time
- You should be prepared to do quite a bit of reading from books and papers

Duke Honor Code

- Read the honor code from http://integrity.duke.edu
- Any suspected case of violation of honor code will be pursued aggressively by the course staff and will be handled through official university channels
- You MUST submit your own solutions for homeworks and exams
  - You are encouraged to discuss course material with other students
  - If you have confusions regarding a homework question, ask on Piazza (as a public question if possible)
- If you are not sure whether something is allowed, ASK!

What is this course about?

- This is a graduate-level database course in CS
- We will cover principles and internals of database systems in depth
- We will also have an introduction to advanced research topics in databases (later in the course)
- NOTE: There will be changes from the Spring 2015 version
- Let me know if you have questions after the class

What will be covered?

- Database concepts
  - Data Models, SQL, Views, Constraints, RA, Normalization
- Principles and internals of <u>d</u>ata<u>b</u>ase <u>m</u>anagement <u>s</u>ystems (DBMS)
  - Indexing, Query Execution-Algorithms-Optimization, Transactions, Parallel and Distributed Query Processing, Map Reduce
- Database theory and Advanced research topics in databases
  - Datalog, NOSQL, Data Integration, Uncertain Data, Data mining
  - Involves some Maths and analytical reasoning

- We will go fast for some basic topics and inject material from seminal and new research papers

Background

- You should have taken (or have the willingness to learn the content of) an undergraduate database course
  - and some concepts from a CS undergraduate data structure and algorithms course

- We will have a brief overview of the basic topics in class

Why should we care about databases?

- We are in a data-driven world
- “Big Data” is supposed to change the mode of operation for almost every single field
  - Science, Technology, Healthcare, Business, Manufacturing, Journalism, Government, Education, …
- We must know how to collect, store, process, and analyze such data

Why should we care about databases?

- From "Big Data" wiki:

Science

"The Large Hadron Collider experiments represent about 150 million sensors delivering data 40 million times per second. There are nearly 600 million collisions per second. If all sensor data were recorded in LHC, …. this is equivalent to 500 quintillion (5×10^{20}) bytes per day, almost 200 times more than all the other sources combined in the world."

Why should we care about databases?

Technology

- From "Big Data" wiki:
  - eBay.com uses two data warehouses at 7.5 PB (x 10^{12}) and 40PB as well as a 40PB Hadoop cluster for search, consumer recommendations, and merchandising
  - Facebook handles 50 billion photos from its user base
  - As of August 2012, Google was handling roughly 100 billion searches per month

Why should we care about databases?

- From "Big Data" wiki:
  - Healthcare: digitization of patient's data, prescriptive analytics
  - Media: Tailor articles and advertisements that reach targeted people, validate claims
    - "Computational Journalism" project in Duke DB group
  - Manufacturing: supply planning
  - Sports: improve training, understanding competitors

Healthcare

Media

Manufacturing

Sports

…..

Why should we care about databases?

- Simply storing such large datasets in a file does not work
  - Need efficient model, storage, and processing

- A DBMS takes care of such issues – the user only has to run queries to process such datasets
  - much simpler than writing low level code

Today

- DBMS
- Data Models

What is a Database?

- A database is a collection of data
  - typically related and describing activities of an organization
- A database may contain information about
  - Entities
    - students, faculty, courses, classroom
  - Relationships between entities
    - students' enrollment, faculty teaching courses, rooms for courses

Why use a DBMS

- i.e. why not use file system and a programming language?
- Suppose a company has a large collection of data on employees, departments, products, sales etc.
- Requirements:
  - Quickly answer questions on data
    - Note that all the data may not fit in main memory
  - Concurrent access: apply changes consistently
  - Restricted access (e.g. salary)

Why use a DBMS?

- A DBMS is a piece of software (i.e. a big program written by someone else) that makes these tasks easier
  - Quick access
  - Robust access
  - Safe access
  - Simpler access

Why use a DBMS?

1. Data Independence
   - Application programs should not be exposed to the data representation and storage
   - DBMS provides an abstract view of the data

2. Efficient Data Access
   - A DBMS utilizes a variety of sophisticated techniques to store and retrieve data (from disk) efficiently

Why use a DBMS?

3. Data Integrity and Security
   - DBMS enforces "integrity constraints" – e.g. check whether total salary is less than the budget
   - DBMS enforces "access controls" – whether salary information can be accesses by a particular user

4. Data Administration
   - Centralized professional data administration by experienced users can manage data access, organize data representation to minimize redundancy, and fine tune the storage

Why use a DBMS?

5. Concurrent Access and Crash Recovery
   - DBMS schedules concurrent accesses to the data such that the users think that the data is being accessed by only one user at a time
   - DBMS protects users from system failures

6. Reduced Application Development Time
   - Supports many functions that are common to a number of applications accessing data
   - Provides high-level interface
   - Facilitates quick and robust application development

When NOT to use a DBMS?

- DBMS is optimized for certain kind of workloads and manipulations
- There may be applications with tight real-time constraints or a few well-defined critical operations
- Abstract view of the data provided by DBMS may not suffice
- To run complex, statistical/ML analytics on large datasets
- Will see some examples later in the course

Data Model

- Applications need to model some real world units
- Entities:
  - Students, Departments, Courses, Faculty, Organization, Employee, …
- Relationships:
  - Course enrollments by students, Product sales by an organization

- A data model is a collection of high-level data description constructs that hide many low-level storage details

Data Model

Can Specify:

1. Structure of the data
   - like arrays or structs in a programming language
   - but at a higher level (conceptual model)
2. Operations on the data
   - unlike a programming language, not any operation can be performed
   - allow limited sets of queries and modifications
   - a strength, not a weakness!
3. Constraints on the data
   - what the data can be
   - e.g. a movie has exactly one title

Important Data Models

- Structured Data
- Semi-structured Data
- Unstructured Data

What are these?

Important Data Models

- Structured Data
  - All elements have a fixed format
  - Relational Model (table)
- Semi-structured Data
  - Some structure but not fixed
  - Hierarchically nested tagged-elements in tree structure
  - XML
- Unstructured Data
  - No structure
  - text, image, audio, video

Relational Data Model

- Proposed by Edward (Ted) Codd in 1970
  - won Turing award for it
- Motivation:
  - Simplicity
  - Better logical and physical data independence

Relational Data Model

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith1@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

- The data description construct is a Relation
  - Represented as a table
  - Basically a set of records

Relational Data Model

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith1@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

Attribute/ Column/ Field

Tuple/ Row/ Record

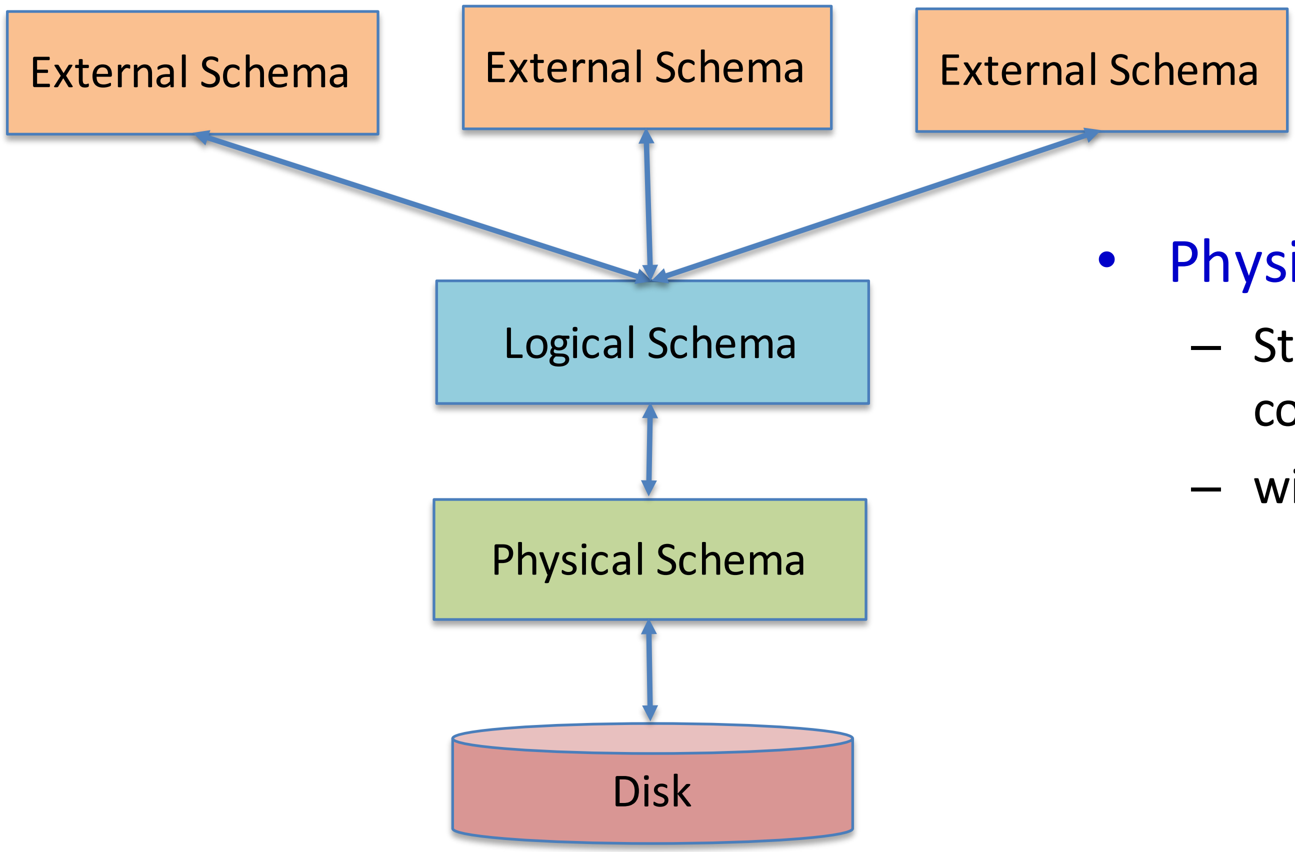
Value

- Schema:
  - A template for describing an entity/relationship (e.g. students)

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

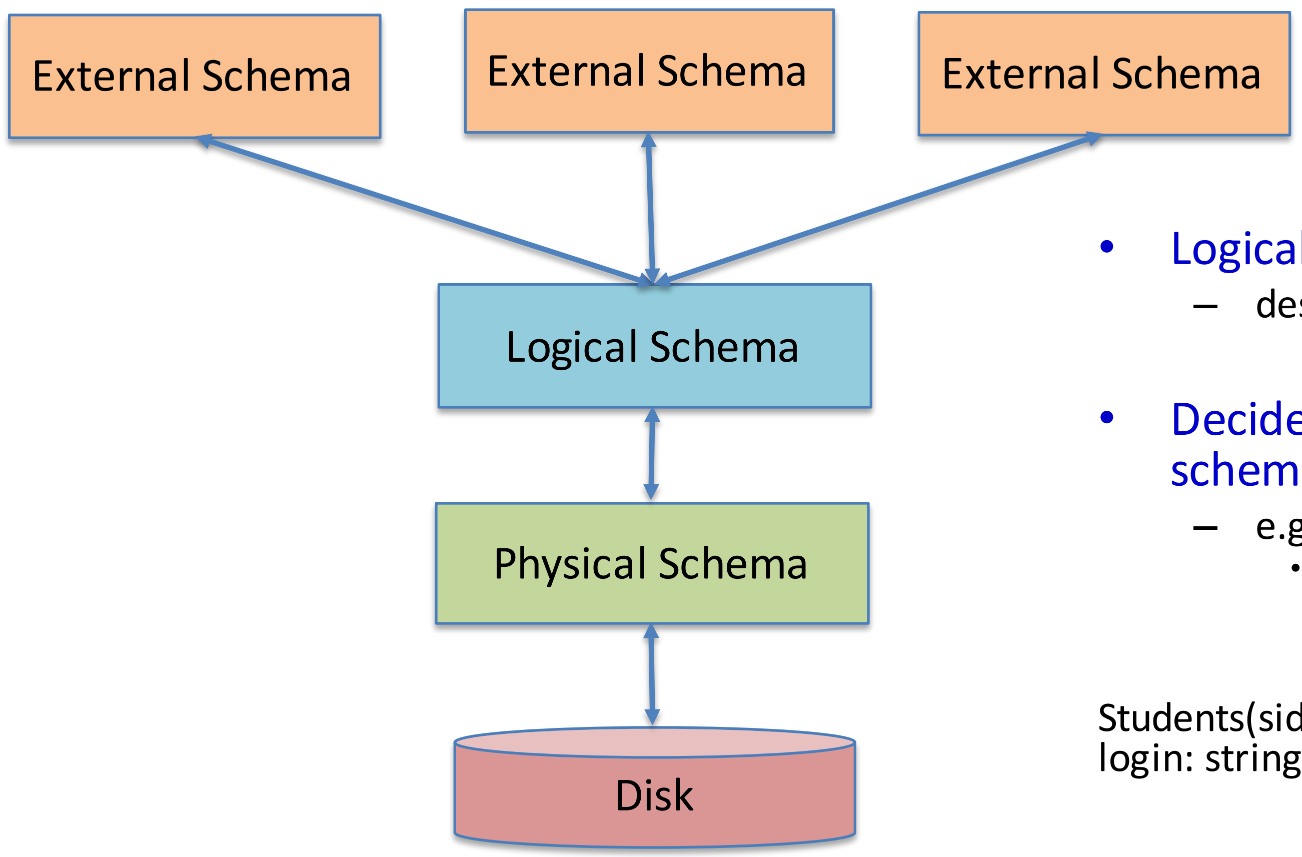
What is a poorly chosen attribute in this relation?

Levels of Abstractions in a DBMS



- Physical schema
  - Storage as files, row vs. column store, indexes
  - will discuss these soon

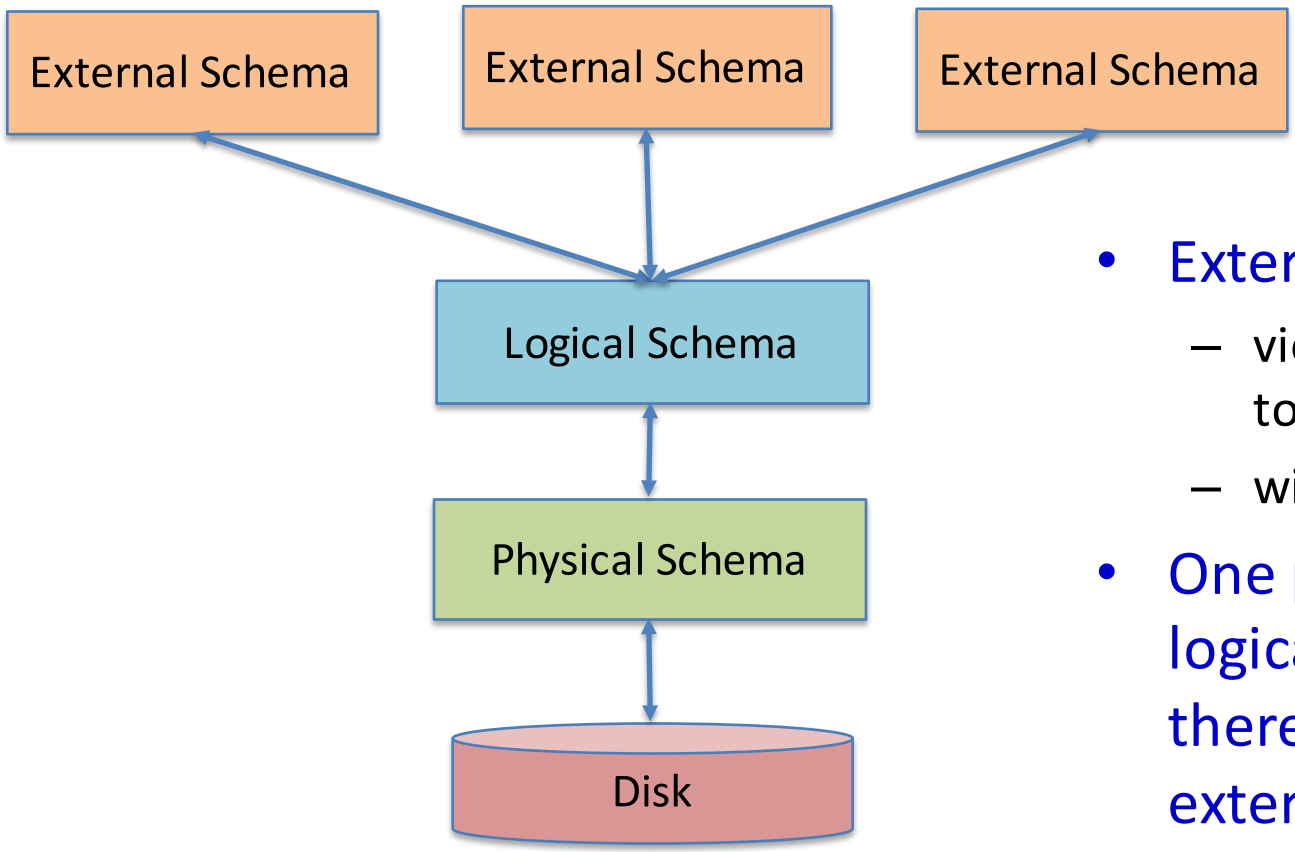
Levels of Abstractions in a DBMS



- Logical/Conceptual schema
  - describes the stored data
- Decided by conceptual schema design
  - e.g. ER Diagram
    - not covered in this course

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

Levels of Abstractions in a DBMS



- External schema
  - views of the database to different users
  - will discuss this soon
- One physical and logical schema but there can be multiple external schemas

Data Independence

- Application programs are insulated from changes in the way the data is structured and stored

- A very important property of a DBMS

- Logical and Physical

Logical Data Independence

- Users can be shielded from changes in the logical structure of data
- e.g. Students:

  Students(sid: string, name: string, login: string, age: integer, gpa: real)
- Divide into two relations

  Students_public(sid: string, name: string, login: string)

  Students_private(sid: string, age: integer, gpa: real)
- Still a "view" Students can be obtained using the above new relations
- A user who queries this view Students will get the same answer as before

Physical Data Independence

- The conceptual schema insulates users from changes in physical storage details
  - how the data is stored on disk
  - the file structure
  - the choice of indexes

- The application remains unaltered
  - But the performance may be affected by such changes

Semi-structured Data and XML

- XML: Extensible Markup Language
- Will not be covered in detail in this class, but many datasets available to download are in this form
  - You will download the DBLP dataset in XML format and transform into relational form (in HW 1)
  - Some help in a tutorial, but need to learn the rest yourself
- Data does not have a fixed schema
  - “Attributes” are part of the data
  - The data is “self-describing”
  - Tree-structured

XML: Example

Attributes

```
<article mdate="2011-01-11” key="journals/acta/Saxena96">
    <author>Sanjeev Saxena</author>
    <title>Parallel Integer Sorting and Simulation Amongst CRCW
        Models.</title>
    <pages>607-619</pages>
    <year>1996</year>
    <volume>33</volume>
    <journal>Acta Inf.</journal>
    <number>7</number>
    <url>db/journals/acta/acta33.html#Saxena96</url>
    <ee>http://dx.doi.org/10.1007/BF03036466</ee>
</article>
```

Elements

Attribute vs. Elements

- Elements can be repeated and nested
- Attributes are unique and atomic

Why XML?

+ Serves as a model suitable for integration of databases containing similar data with different schemas

+ Flexible – easy to change the schema and data

- Makes query processing more difficult

XML to Relational Model

- Problem 1: Repeated attributes

```
<book>
    <author>Ramakrishnan</author>
    <author>Gehrke</author>
    <title>Database Management Systems</title>
    <pubisher> McGraw Hill
</book>
```

What is a good relational schema?

XML to Relational Model

- Problem 1: Repeated attributes

```
<book>
    <author>Ramakrishnan</author>
    <author>Gehrke</author>
    <title>Database Management Systems</title>
    <pubisher> McGraw Hill</publisher>
</book>
```

Title	Publisher	Author1	Author2

XML to Relational Model

- Problem 1: Repeated attributes

```
<book>
    <author>Garcia-Molina</author>
    <author>Ullman</author>
    <author>Widom</author>
    <title>Database Systems – The Complete Book</title>
    <pubisher>Prentice Hall</publisher>
</book>
```

Does not work

Title	Publisher	Author1	Author2

XML to Relational Model

Book

BookId	Title	Publisher
b1	Database Management Systems	McGraw Hill
b2	Database Systems – The Complete Book	Prentice Hall

BookAuthoredBy

BookId	Author
b1	Ramakrishnan
b1	Gehrke
b2	Garcia-Molina
b2	Ullman
b2	Widom

XML to Relational Model

- Problem 2: Missing attributes

```
<book>
        <author>Ramakrishnan</author>
        <author>Gehrke</author>
        <title>Database  Management Systems</title>
        <pubisher> McGraw Hill
        <edition>Third</edition>
</book>
<book>
        <author>Garcia-Molina</author>
        <author>Ullman</author>
        <author>Widom</author>
        <title>Database  Systems – The Complete
Book</title>
        <pubisher>Prentice  Hall</publisher>
</book>
```

BookId	Title	Publisher	Edition
b1	Database Management Systems	McGraw Hill	Third
b2	Database Systems – The Complete Book	Prentice Hall	null

Summary

- Relational data model is the most standard for database managements
  - semi-structured model/XML is also used in practice
  - unstructured data (text/photo/video) is unavoidable, but won't be covered in this class
- A DBMS provides data independence and insulates the application programmer from many low level details
- We will learn about those low level details as well as high level data management in this course