CSE544
Data Management

Lecture 1: Relational Data Model

Outline

- Introduction, class overview
- Database management systems (DBMS)
- The relational model

Course Staff

- Instructor: Dan Suciu
  - Office hours: Mondays, 2:30-3:20
  - Location: CSE 662

- TA: Walter Cai
  - Office hours: Thursdays, 10:00-10:50
  - Location: CSE 220

Goals of the Class

- Relational Data Model
  - Data models, data independence, declarative query language.
- Relational Database Systems
  - Storage, query execution and optimization, transactions
  - Parallel data processing, column-oriented db etc.
- Transactions
  - Optimistic/pessimistic concurrency control
  - ARIES recovery system
- Miscellaneous

A Note for Non-Majors

- For the Data Science option: take 414
- For the Advanced Data Science option: take 544

- 544 is an *advanced* class, not intended as an introduction to data management research
- Does not cover fundamentals systematically, yet there is an exam testing those fundamentals

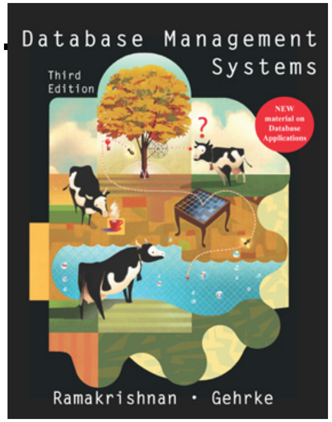
- Unsure? Look at the short quiz on the website.

Class Format

- Two lectures per week:
  - MW 10:00-11:20, CSE2 04
- Two makeup lectures:
  - Th 1/16, Th 1/23: 10:30-11:50, CSE2 371

Readings and Notes

- Background readings from the following book
  - Database Management Systems. **Third Ed.** Ramakrishnan and Gehrke. McGraw-Hill. [recommended]


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

- Readings are based on papers
  - Mix of old seminal papers and new papers
  - Papers are available on class website
- Lecture notes (the slides)
  - Posted on class website after each lecture

Class Resources

Website: lectures, assignments, videos

- http://www.cs.washington.edu/544

Mailing list on course website

Piazza: discuss assignments, papers, etc

Evaluation

- Assignments 30%
- Exam 30%
- Project 30%
- Paper reviews + class participation 10%

Assignments – 30%

- **HW1**: Use a DBMS
- **HW2**: Datalog
- **HW3**: Build a simple DBMS
- **HW4**: Data analysis in the cloud

- See course calendar for deadlines
- Late assignments w/ ***very*** valid excuse

Exam – 30%

- March 16, 8:30-10:20 CSE2 G04

Project – 30%

- Topic
  - Choose from a list of mini-research topics (will update the list)
  - Or come up with your own
  - Can be related to your ongoing research
  - Can be related to a project in another course
  - Must be related to databases / data management
  - Must involve either research or significant engineering
  - Open ended

- Final deliverables
  - Posters: Friday, March 6, 10am – 2pm in the CSE Atrium
  - Short conference-style paper (6 pages)

Project – 30%

- Dates posted on course website
  - **M1**: form groups
  - **M2**: Project proposal
  - **M3**: Milestone report
  - **M4**: Poster presentation
  - **M5**: Project paper
- We will provide feedback throughout the quarter

Paper reviews – 10%

- Recommended length: ½ page – 1 page
  - Summary of the main points of the paper
  - Critical discussion of the paper
  - Suggested discussion points will be posted for some papers

- Grading: credit/no-credit

- Submit review 12h before lecture

Class Participation

- Because
  - We want you to read & think about the material
- Expectations
  - Ask questions, raise issues, think critically
  - Learn to express your opinion
  - Respect other people's opinions
- Most students get full credit for class participation, but I may penalize students who don't attend lectures or don't participate

Now onward to the world of databases!

Let’s get started

- What is a database?
  - A collection of files storing related data
- Give examples of databases
  - Accounts database; payroll database; UW’s students database; Amazon’s products database; airline reservation database
  - Your ORCA card transactions, Facebook friends graph, past tweets, etc

Data Management

- **Entities**: employees, positions (ceo, manager, cashier), stores, products, sells, customers.

- **Relationships**: employee positions, staff of each store, inventory of each store.

- What operations do we want to perform on this data?

- What functionality do we need to manage this data?

Database Management System

- A DBMS is a software system designed to provide data management services

- Examples of DBMS
  - Oracle, DB2 (IBM), SQL Server (Microsoft),
  - PostgreSQL, MySQL,…

- Several types of architectures (next)

Single Client

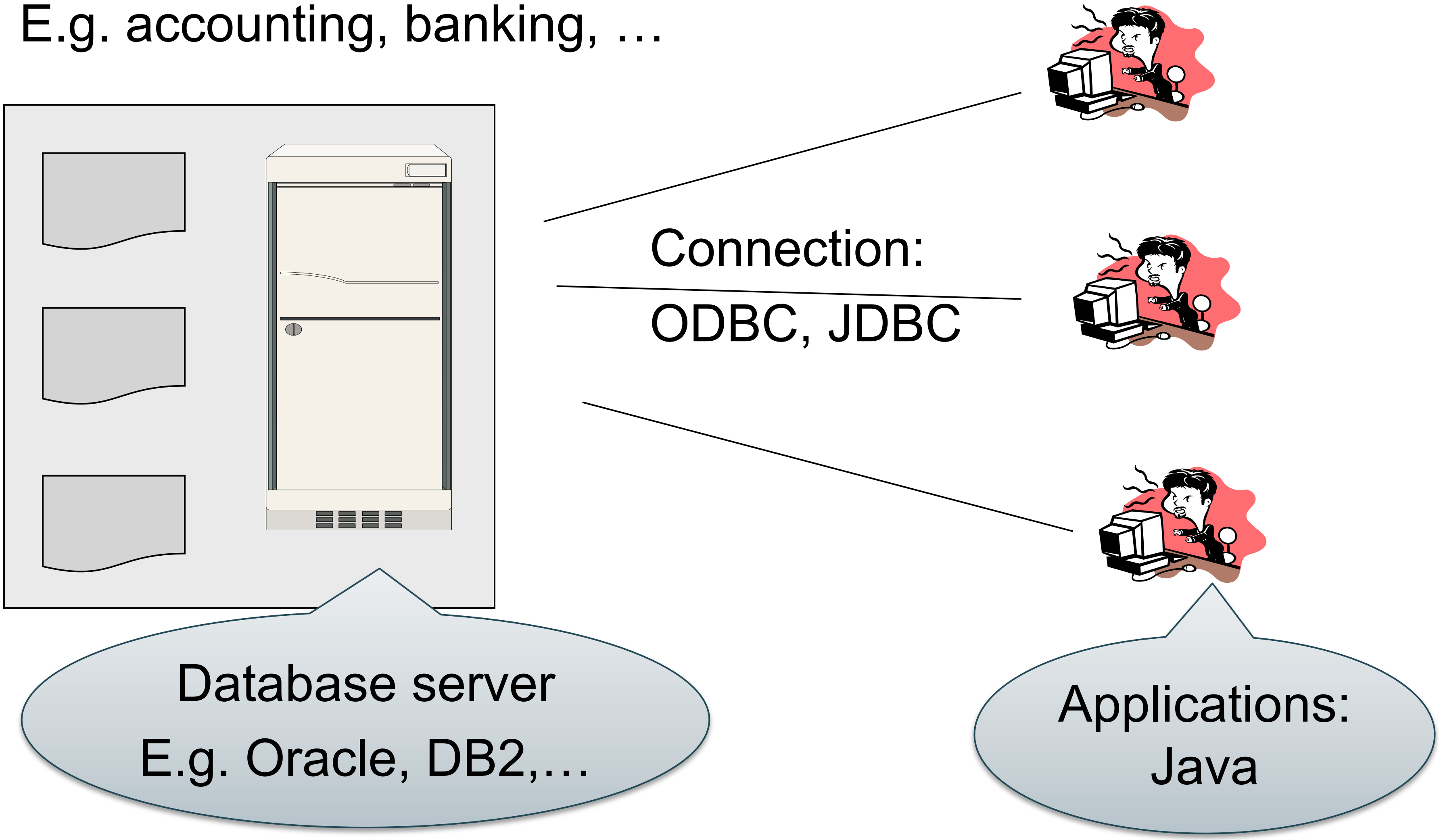
E.g. data analytics

Application and database on the same computer

E.g. sqlite, postgres

Two-tier Architecture Client-Server

E.g. accounting, banking, …



Connection:

ODBC, JDBC

Database server

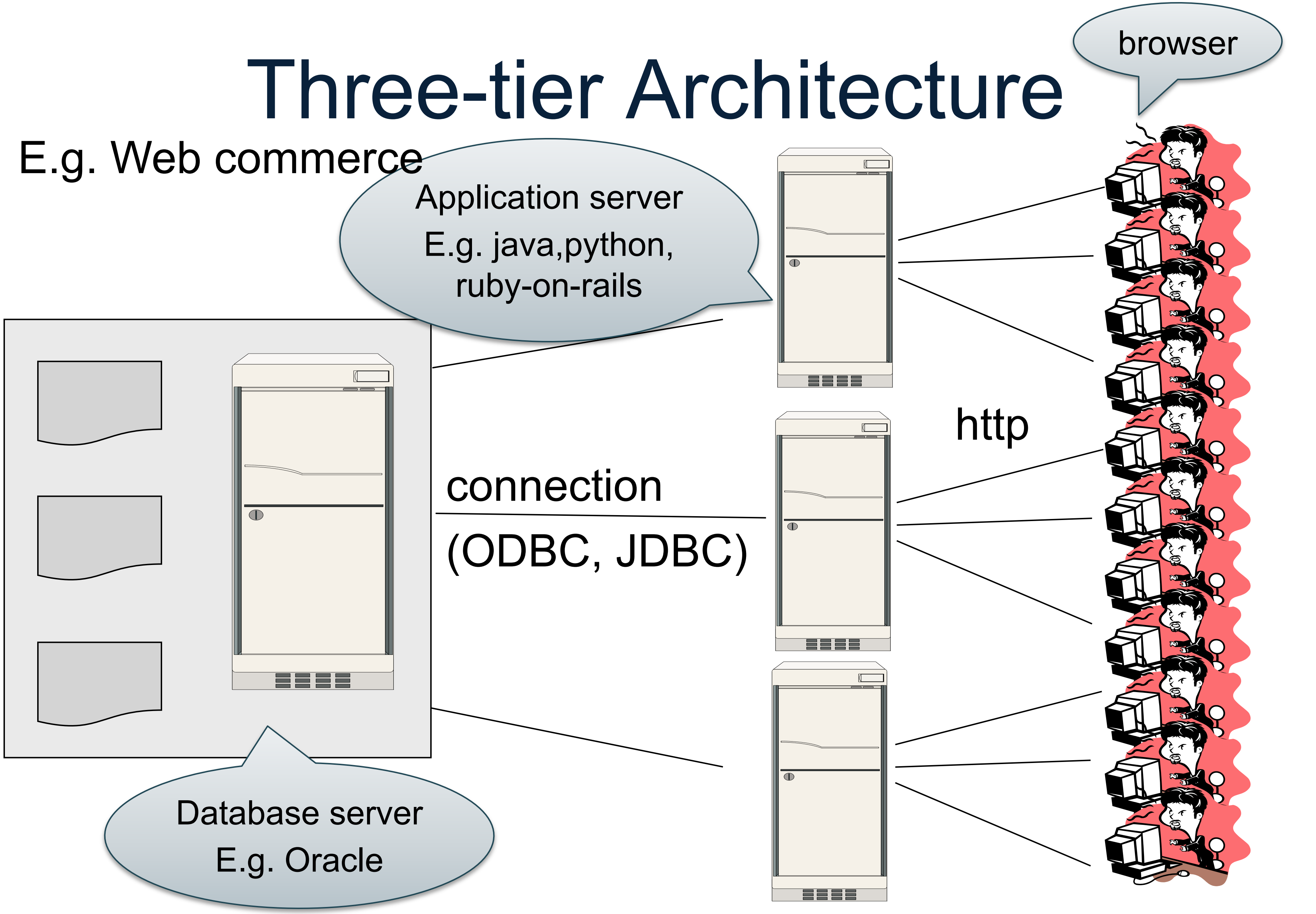
E.g. Oracle, DB2,…

Applications:

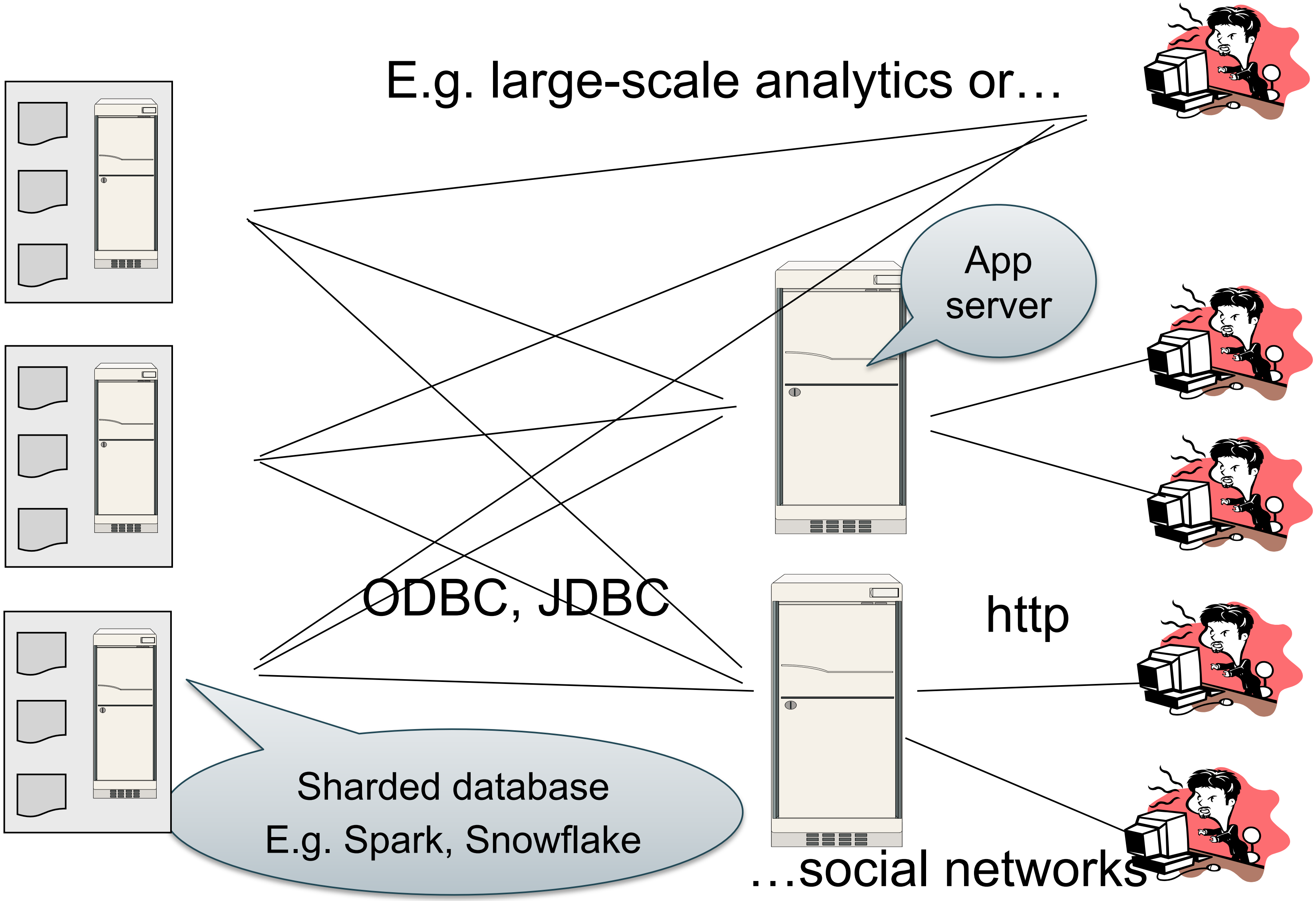
Java

Three-tier Architecture

E.g. Web commerce



Cloud Databases



E.g. large-scale analytics or…

App server

ODBC, JDBC

http

Sharded database

E.g. Spark, Snowflake

…social networks

Workloads

- OLTP – online transaction processing

- OLAP – online analytics processing, a.k.a. Decision Support

Main DBMS Features

- Data independence
  - Data model
  - Data definition language
  - Data manipulation language
- Efficient data access
- Data integrity and security
- Data administration
- Concurrency control
- Crash recovery

Relational Data Model

Data Model

An abstract mathematical concepts that defines the data *and* the queries

Data models:

- Relational (this course)
- Semistructured (XML, JSon, Protobuf)
- Graph data model
- Object-Relational data model

Definition

- **Database is collection of relations**

- Relation is a table with rows & columns
  - SQL uses the term “table” to refer to a relation

- Relation R is subset of D_1 x D_2 x … x D_n
  - Where D_i is the domain of attribute **i**
  - **n** is number of attributes of the relation

Example

- Relation schema

Supplier(sno: integer, sname: string, scity: string, sstate: string)

- Relation instance

sno	sname	scity	sstate
1	s1	city 1	WA
2	s2	city 1	WA
3	s3	city 2	MA
4	s4	city 2	MA

sno is called a key (what does it mean?)

SQL

```
CREATE TABLE supplier (
  sno INT PRIMARY KEY,
  sname TEXT,
  scity TEXT,
  sstate TEXT
);
```

sno	sname	scity	sstate
1	s1	city 1	WA
2	s2	city 1	WA
3	s3	city 2	MA
4	s4	city 2	MA

```
insert into supplier values (1, 's1', 'city1', 'WA');
insert into supplier values (2, 's2', 'city1', 'WA');
insert into supplier values (3, 's3', 'city2', 'WA');
insert into supplier values (4, 's4', 'city2', 'WA');
```

Example

- Two relations

Supplier(sno: integer, sname: string, scity: string, sstate: string)
Product(pno: integer, pname: string, p_sno: integer)

sno	sname	scity	sstate
1	s1	city 1	WA
2	s2	city 1	WA
3	s3	city 2	MA
4	s4	city 2	MA

pno	pname	p_sno
50	iPhone	3
60	iPad	2
70	Dell	3

p_sno is called a foreign key (what does it mean?)

SQL

```
CREATE TABLE supplier (
  sno INT PRIMARY KEY,
  sname TEXT,
  scity TEXT,
  sstate TEXT
);
```

```
CREATE TABLE product (
  pno INT PRIMARY KEY,
  pname TEXT,
  p_sno INT REFERENCES supplier
);
```

sno	sname	scity	sstate
1	s1	city 1	WA
2	s2	city 1	WA
3	s3	city 2	MA
4	s4	city 2	MA

pno	pname	p_sno
50	iPhone	3
60	Dell	2
70	iPad	3

Discussion of the Relational Model

- Relations are *flat* = called 1st Normal Form
- A relation may have a key, but no other FD's = either 3rd Normal form, or Boyce Codd Normal Form (BCNF) depending on some subtle details

[discuss on the white board]

Other Models: Semistructured

- E.g. you will encounter this in HW1:

```
<article mdate="2011-01-11" key="journals/acta/GoodmanS83">
   <author>Nathan Goodman</author>
   <author>Oded Shmueli</author>
   <title>NP-complete Problems Simplified on Tree Schemas.</title>
   <pages>171-178</pages>
   <year>1983</year>
   <volume>20</volume>
   <journal>Acta Inf.</journal>
   <url>db/journals/acta/acta20.html#GoodmanS83</url>
   <ee>http://dx.doi.org/10.1007/BF00289414</ee>
</article>
```

Integrity Constraints

- Condition specified on a database schema
- Restricts data that can be stored in the database instance
- DBMS enforces integrity constraints
- E.g. domain constraint, key, foreign key

Constraints are part of the data model

Key Constraints

- **Key constraint:** “certain minimal subset of fields is a unique identifier for a tuple”

- **Candidate key**
  - Minimal set of fields
  - That uniquely identify each tuple in a relation

- **Primary key**
  - One candidate key can be selected as primary key

Foreign Key Constraints

- Field that refers to tuples in another relation

- Typically, this field refers to the primary key of other relation

- Can pick another field as well (but check documentation)

Key Constraint SQL Examples

```
CREATE TABLE Part (
  pno integer,
  pname varchar(20),
  psize integer,
  pcolor varchar(20),
  PRIMARY KEY (pno)
);
```

Key Constraint SQL Examples

```
CREATE TABLE Supply(
  sno integer,
  pno integer,
  qty integer,
  price integer
);
```

Key Constraint SQL Examples

```
CREATE TABLE Supply(
   sno integer,
   pno integer,
   qty integer,
   price integer,
   PRIMARY KEY (sno,pno)
);
```

Key Constraint SQL Examples

```
CREATE TABLE Supply(
   sno integer,
   pno integer,
   qty integer,
   price integer,
   PRIMARY KEY (sno,pno),
   FOREIGN KEY (sno) REFERENCES Supplier,
   FOREIGN KEY (pno) REFERENCES Part
);
```

Key Constraint SQL Examples

```
CREATE TABLE Supply(
   sno integer,
   pno integer,
   qty integer,
   price integer,
   PRIMARY KEY (sno,pno),
   FOREIGN KEY (sno) REFERENCES Supplier
                        ON DELETE NO ACTION,
   FOREIGN KEY (pno) REFERENCES Part
                        ON DELETE CASCADE
);
```

General Constraints

- Table constraints serve to express complex constraints over a single table

```
CREATE TABLE Part (
  pno integer,
  pname varchar(20),
  psize integer,
  pcolor varchar(20),
  PRIMARY KEY (pno),
  CHECK ( psize > 0 )
);
```

- It is also possible to create constraints over many tables