Carnegie Mellon University

Course Intro & Relational Model

Lecture #01

Database Systems
15-445/15-645
Fall 2017

Andy Pavlo
Computer Science Dept.
Carnegie Mellon Univ.

TODAY'S AGENDA

Wait List
Overview
Course Logistics
Relational Model
Homework #1

WAIT LIST

There are currently 130 people on the waiting list.

Max capacity is 90.

We will enroll people from the waiting list in the order that you complete Homework #1.

COURSE OVERVIEW

This course is on the design and implementation of disk-oriented database management systems.

This is not a course on how to use a database to build applications or how to administer a database.
→ See CMU 95-703 (Heinz College)

COURSE OUTLINE

Relational Databases
Storage
Execution
Concurrency Control
Recovery
Distributed Databases
Potpourri

COURSE LOGISTICS

Course Policies + Schedule:
→ Refer to course web page.

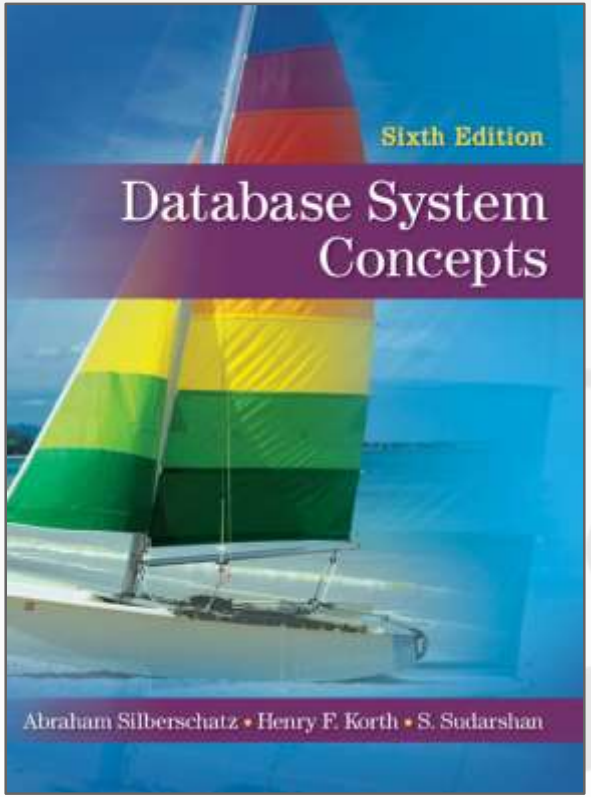
Academic Honesty:
→ Refer to CMU policy page.
→ If you're not sure, ask the professors.
→ Don't be stupid.

All discussion + announcements will be on Canvas.

TEXTBOOK

Database System Concepts
6th Edition
Silberschatz, Korth, & Sudarshan

We will also provide lecture notes that covers topics not found in textbook.



COURSE RUBRIC

Homeworks (15%)
Projects (45%)
Midterm Exam (20%)
Final Exam (20%)
Extra Credit (+10%)

HOMEWORKS

Six homework assignments throughout the semester.

First homework is a SQL assignment. The rest will be pencil-and-paper assignments.

All homeworks should be done individually.

PROJECTS

Four programming projects based on the SQLite DBMS.

→ You will build your own storage manager from scratch.

We will not teach you how to write/debug C++11 code.

See 2015 video from SQLite creator for more info.

LATE POLICY

You are allowed 4 slip days for either homeworks or projects.

You lose 25% of an assignment's points for every 24hrs it is late.

Mark on your submission (1) how many days you are late and (2) how many late days you have left.

PLAGIARISM WARNING

The homeworks and projects must be your own work.

You may not copy source code from other groups or the web.

Plagiarism will not be tolerated. See CMU's Policy on Academic Integrity for additional information.

EXAMS

Mid-term Exam (October 18)
Final Exam (End of Semester)

Closed book.
One sheet of handwritten notes.

EXTRA CREDIT

Pick a DBMS and get standard database benchmarks to run on it.

→ Can be either OLTP or OLAP system.
→ We already have an open-source testing framework that you can use.
→ We will give you EC2 credits.
→ Groups of at most three people.

We will provide more information later in the semester.

NUODB

Databases

DATABASE

Organized collection of inter-related data that models some aspect of the real-world.

Databases are core the component of most computer applications.

DATABASE EXAMPLE

Create a database that models a digital music store.

Things we need store:

→ Information about Artists
→ What Albums those Artists released
→ The Tracks on those Albums

ENTITY-RELATIONSHIP DIAGRAM

Artists have names, year that they started, and country of origin.

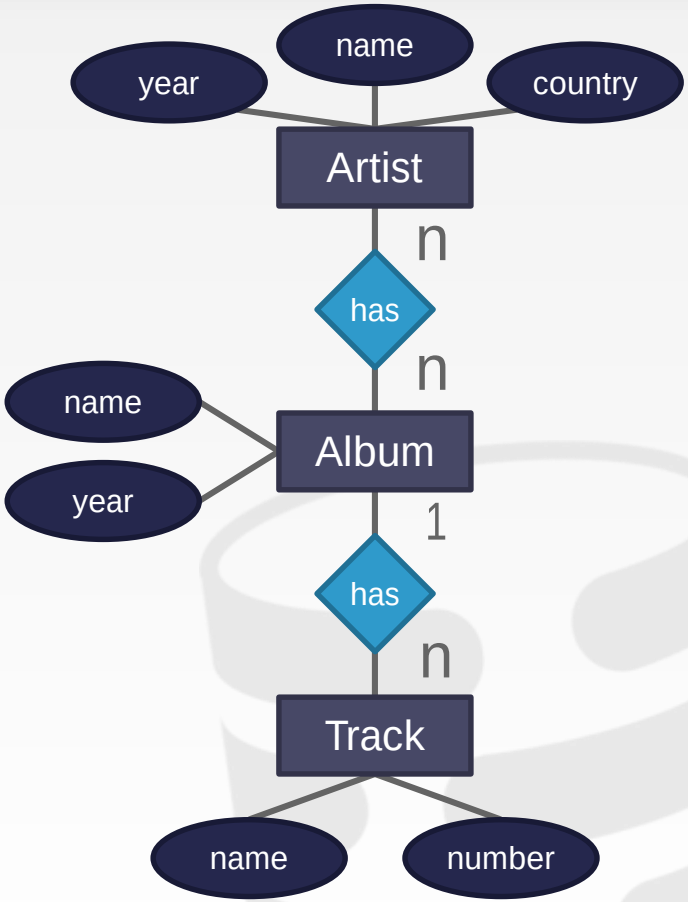
Albums have names, release year.

Tracks have a name and number.

An Album has one or more Artists.

An Album has multiple Tracks.

A Track can appear only on one Album.



FLAT FILE STRAWMAN

Store the data in comma-separated value (CSV) files.

→ Use a separate file per entity.

→ The application has to parse the files each time they want to read/update records.

Artist(name, year, country)

```
"Wu Tang Clan",1992,"USA"
"Notorious BIG",1992,"USA"
"Ice Cube",1989,"USA"
```

Album(name, artist, year)

```
"Enter the Wu Tang","Wu Tang Clan",1993
"St.Ides Mix Tape","Wu Tang Clan",1994
```

FLAT FILE STRAWMAN

Store the data in comma-separated value (CSV) files.

→ Use a separate file per entity.

→ The application has to parse the files each time they want to read/update records.

Example: Get the year that Ice Cube went solo.

Artist(name, year, country)

```
"Wu Tang Clan",1992,"USA"
"Notorious BIG",1992,"USA"
"Ice Cube",1989,"USA"
```

```
for line in file:
  record = parse(line)
  if “Ice Cube” == record[0]:
    print int(record[1])
```

FLAT FILES: DATA INTEGRITY

How do we ensure that the artist is the same for each album entry?

What if somebody overwrites the album year with an invalid string?

How do we store that there are multiple artists on an album?

FLAT FILES: IMPLEMENTATION

How do you find a particular record?

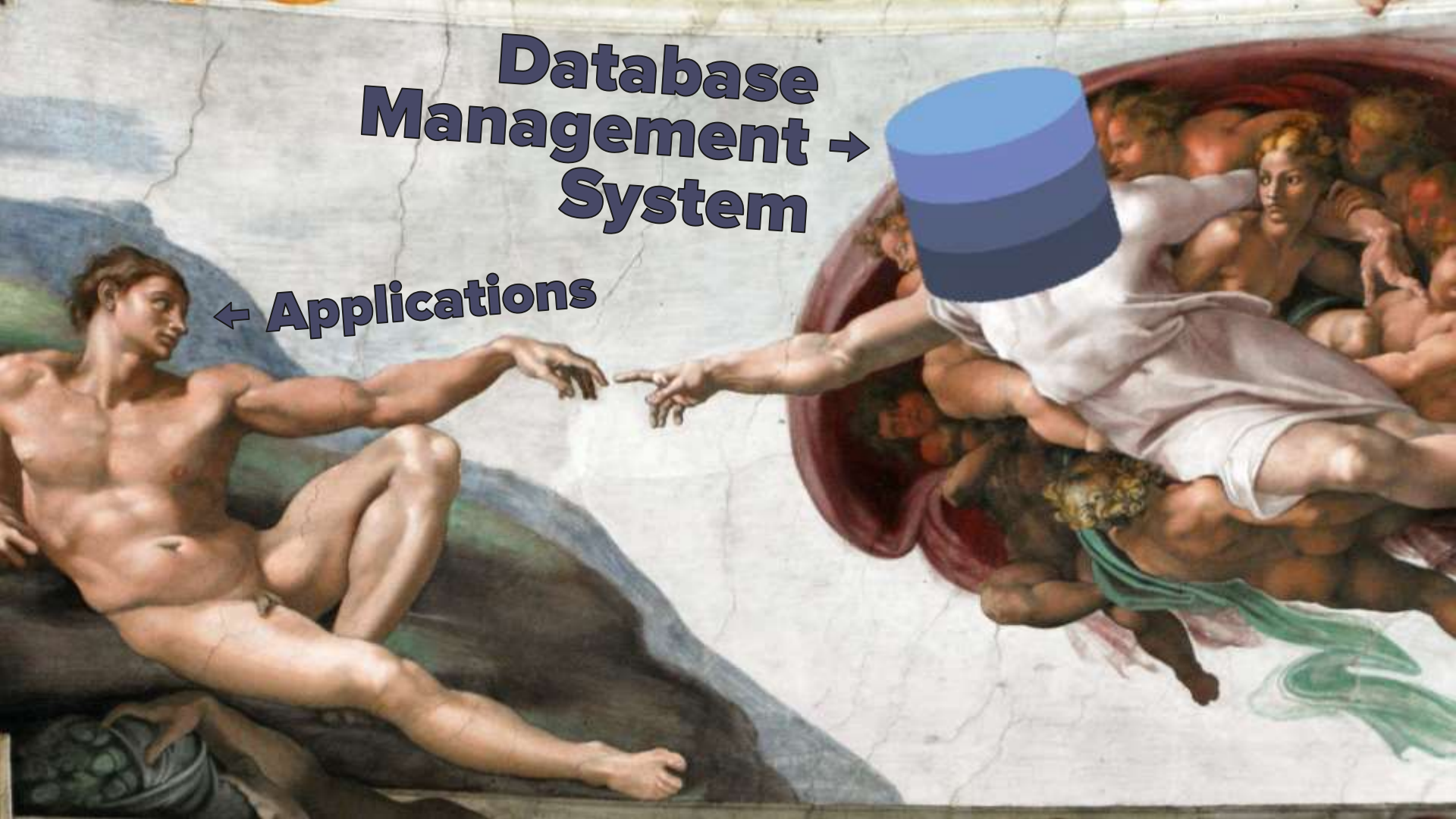
What if we now want to create a new application that uses the same database?

What if two threads try to write to the same file at the same time?

FLAT FILES: DURABILITY

What if the machine crashes while we're updating a record?

What if we want to replicate the database on multiple machines for high availability?


Database Management System →
← Applications

DATABASE MANAGEMENT SYSTEM

A DBMS is software that allows applications to store and analyze information in a database.

A general-purpose DBMS is designed to allow the definition, creation, querying, update, and administration of databases.

DATABASE MANAGEMENT SYSTEM

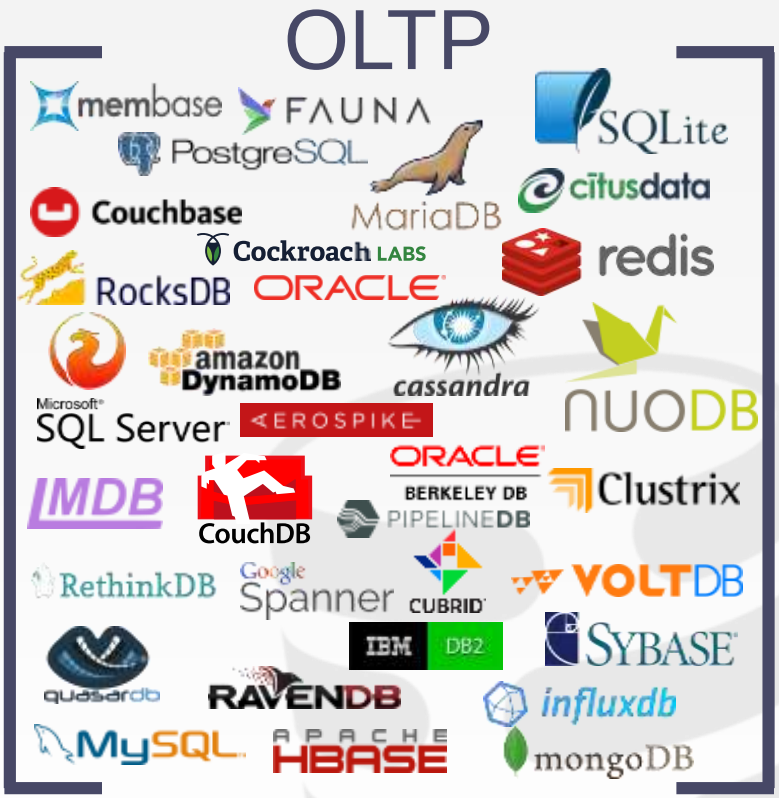
DBMSs are used in almost every application, web site, software system that you can think of.

Think about the other types of software that CMU SCS does not dedicate entire courses to...

DBMS TYPES: TARGET WORKLOADS

On-line Transaction Processing

→ Fast operations that only read/update a small amount of data each time.



DBMS TYPES: TARGET WORKLOADS

On-line Transaction Processing

→ Fast operations that only read/update a small amount of data each time.

On-line Analytical Processing

→ Complex queries that read a lot of data to compute aggregates.

DBMS TYPES: TARGET WORKLOADS

On-line Transaction Processing

→ Fast operations that only read/update a small amount of data each time.

On-line Analytical Processing

→ Complex queries that read a lot of data to compute aggregates.

Hybrid Transaction + Analytical Processing

→ OLTP + OLAP together on the same database instance

HTAP

DBMS TYPES: DATA MODEL

Relational ← **Most DBMSs**

Key/Value

Graph

Document

Column-family

Array / Matrix

Hierarchical

Network

DBMS TYPES: DATA MODEL

Relational

Key/Value
Graph
Document
Column-family

← NoSQL

Array / Matrix
Hierarchical
Network

DBMS TYPES: DATA MODEL

Relational
Key/Value
Graph
Document
Column-family
Array / Matrix ← **Machine Learning**
Hierarchical
Network

DBMS TYPES: DATA MODEL

Relational
Key/Value
Graph
Document
Column-family
Array / Matrix
Hierarchical
Network

← **Obsolete / Rare**

RELATIONAL MODEL

A relation is unordered set that contain the relationship of attributes that represent entities.

A tuple is a sequence of attribute values in the relation.

Integrity Constraints:
→ Primary Keys
→ Foreign Keys

Artist(name, year, country)

name	year	country
Wu Tang Clan	1992	USA
Notorious B.I.G.	1992	USA
Ice Cube	1989	USA

RELATIONAL MODEL: PRIMARY KEYS

A relation's primary key uniquely identifies a single tuple.

Some DBMSs support auto-generation of unique integer primary keys:

→ **SEQUENCE** (SQL:2003)
→ **AUTO_INCREMENT** (MySQL)

Artist(name, year, country)

name	year	country
Wu Tang Clan	1992	USA
Notorious B.I.G.	1992	USA
Ice Cube	1989	USA

RELATIONAL MODEL: PRIMARY KEYS

A relation's primary key uniquely identifies a single tuple.

Some DBMSs support auto-generation of unique integer primary keys:

→ **SEQUENCE** (SQL:2003)
→ **AUTO_INCREMENT** (MySQL)

Artist(id, name, year, country)

id	name	year	country
123	Wu Tang Clan	1992	USA
456	Notorious B.I.G.	1992	USA
789	Ice Cube	1989	USA

RELATIONAL MODEL: FOREIGN KEYS

A foreign key specifies that an attribute from one relation has to map to a tuple in another relation.

Artist(id, name, year, country)

id	name	year	country
123	Wu Tang Clan	1992	USA
456	Notorious B.I.G.	1992	USA
789	Ice Cube	1989	USA

Album(id, name, artists, year)

id	name	artists	year
11	Enter the Wu Tang	123	1993
22	St.Ides Mix Tape	???	1994

RELATIONAL MODEL: FOREIGN KEYS

A foreign key specifies that an attribute from one relation has to map to a tuple in another relation.

ArtistAlbum(artist_id, album_id)

artist_id	album_id
123	11
123	22
789	22

Artist(id, name, year, country)

id	name	year	country
123	Wu Tang Clan	1992	USA
456	Notorious B.I.G.	1992	USA
789	Ice Cube	1989	USA

Album(id, name, artists, year)

id	name	artists	year
11	Enter the Wu Tang	123	1993
22	St.Ides Mix Tape	???	1994

RELATIONAL MODEL: FOREIGN KEYS

A foreign key specifies that an attribute from one relation has to map to a tuple in another relation.

ArtistAlbum(artist_id, album_id)

artist_id	album_id
123	11
123	22
789	22

Artist(id, name, year, country)

id	name	year	country
123	Wu Tang Clan	1992	USA
456	Notorious B.I.G.	1992	USA
789	Ice Cube	1989	USA

Album(id, name, year)

id	name	year
11	Enter the Wu Tang	1993
22	St.Ides Mix Tape	1994

RELATIONAL MODEL: QUERIES

The relational model is independent of any query language implementation.

SQL is the de facto standard.

Next Class: We will define an algebra + calculus for querying relations.

```
for line in file:
  record = parse(line)
  if “Ice Cube” == record[0]:
    print int(record[1])
```

```
SELECT year FROM artists
 WHERE name = "Ice Cube“;
```

CONCLUSION

Databases are ubiquitous.

Relational databases are the most common data model because it is the most flexible.

HOMEWORK #1

Write SQL queries to perform basic data analysis on court data.

I will not be teaching basic SQL. Read the textbook.

Due: Wed Sept 13th @ 11:59pm

http://15445.courses.cs.cmu.edu/fall2017/homework1