Databases

Part 1/2 The Relational Data Model

Kai Brünnler
CAS Applied Data Science
University of Bern

Material adapted from: Silberschatz, Korth, Sudarshan: Database System Concepts. 6th Edition.

Database Management System

- A DBMS consists of
 - A collection of interrelated data
 - A set of programs to access the data
 - An environment that is both convenient and efficient to use
- Classic Database Applications:
 - Banks, Airlines, Universities, Shops, Manufacturers, Human resources
- Our running Example: A University Database.
 - Some routine tasks:
 - add new student, register student for course, list course participants, assign grades, compute grade point averages, generate transcripts
 - Can you think of a data science task?

Drawbacks of File Systems

In the early days, database applications were built directly on top of file systems. But that leads to many problems.

- Data redundancy and inconsistency
 - Duplication of information in different files
 - Example: data of a student with double major is stored by both departments
- Difficulty in accessing data
 - Need to write a new program for each unexpected new task
 - Example: There is an application program to generate a list of all students. Suddenly, there is a new requirement to generate a list of all students with certain grade.

Drawbacks of File Systems

Data isolation

- Multiple files can have multiple file formats
- It becomes harder to write new applications to access the data
- Example: First, data was stored in CSV for interoperability, but later XML became popular, so new data was stored in XML.

Integrity problems

- Integrity constraints become "buried" in program code: they become hard to understand and hard to change
- Example: There are only two semesters in a year: "Fall" and "Spring". It is hard to identify all the places in the application code that ensure that.

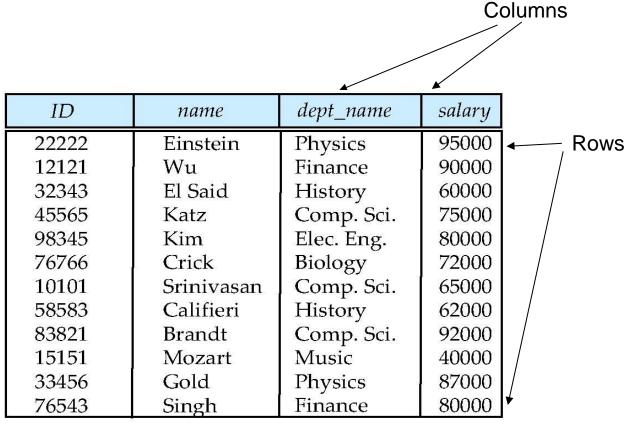
Drawbacks of File Systems

- Updates are not atomic
 - System failures may leave data in an inconsistent state
 - Example: A transfer of funds consists of two operations: subtract from one account, add to the other. The system might fail after the first operation.
- Problems with concurrent access
 - Concurrent access can lead to inconsistencies
 - Example: Two people reading a balance (say 100) and updating it by withdrawing money (say 50 each) at the same time
- Security problems
 - It is hard to provide a user with access to some, but not all, data
 - Example: A schedule planner needs to know the department of an instructor but is not allowed to see salary

Database systems were developed to solve these problems.

Relational Model

Example of tabular data in the relational model



(a) The *instructor* table

A Sample Relational Database

ID	name	dept_name	salary
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The instructor table

dept_name	building	budget
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The department table

Schema and Instance

- They are similar to type and value of a variable in programming languages
- Schema the logical structure of the database
 - Example:
 - instructor(ID, name, dept_name, salary)
 - department(dept_name, building, budget)
 - ... and the meaning of all those terms
- Instance the actual content of the database at a particular point in time
 - Example: the tables we have seen before

SQL

- We usually use **SQL** (Structured Query Language) to access a database
- SQL has two distinct parts:
 - The schema is modified by the Data Definition Language (DDL)
 - The instance is modified by the Data Manipulation Language (DML)

SQL: Data Definition Language

Defining the database schema and implementation details

```
Example: create table instructor (

ID char(5),

name varchar(20),

dept_name varchar(20),

salary numeric(8,2))
```

- The DDL can also specify integrity constraints:
 - Primary key constraint
 - Example: An ID uniquely identifies an instructor
 - Foreign key constraint
 - Example: The dept_name of an instructor must exist in the department relation

SQL: Data Manipulation Language

Example: Find the name of the instructor with ID 22222

```
select name
from instructor
where ID = '22222'
```

Example: Find the building of the instructor "Einstein"

Database Schema Design

Is there any problem with this schema?

-			i
П		1	
L			
			J

ID	пате	salary	dept_name	building	budget
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Database Schema Design

- The previous database schema is bad:
 - Repetition of information:
 - the budget of a certain department is stored redundantly for each instructors in the department
 - Difficulty of storing information:
 - how to store the budget of a department without instructors?
- How would you improve the schema?

Database Management Software

- Relational Database Software
 - Commercial



- Oracle, IBM DB2, MS SQL Server
- Open Source
 - Postgres, MySQL
- What's the most widely deployed database software?

SQLite

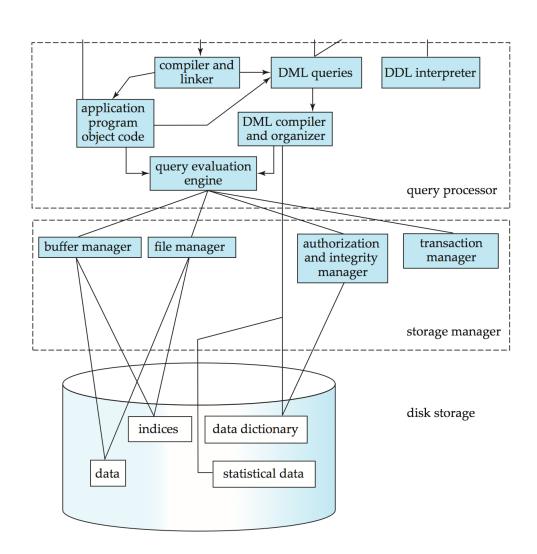
- SQLite is an open source SQL database library
- Less than 500KB
- "a replacement for fopen()"

- The most widely deployed database, used in:
 - Android, iPhone
 - Windows 10, MacOS
 - Firef Chrome, Safari
 - Skype, Dropbox, Adobe Reader
 - etc.

Structure of a DBMS

A database system consists of two main components:

- The Query Processor translates queries into an efficient sequence of operations at the physical level.
- The Storage manager stores data and executes operations at the physical level.



The course Relation

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

The section Relation

course_id	sec_id	semester	year	building	room_number	time_slot_id
BIO-101	1	Summer	2009	Painter	514	В
BIO-301	1	Summer	2010	Painter	514	A
CS-101	1	Fall	2009	Packard	101	Н
CS-101	1	Spring	2010	Packard	101	F
CS-190	1	Spring	2009	Taylor	3128	E
CS-190	2	Spring	2009	Taylor	3128	A
CS-315	1	Spring	2010	Watson	120	D
CS-319	1	Spring	2010	Watson	100	В
CS-319	2	Spring	2010	Taylor	3128	C
CS-347	1	Fall	2009	Taylor	3128	A
EE-181	1	Spring	2009	Taylor	3128	C
FIN-201	1	Spring	2010	Packard	101	В
HIS-351	1	Spring	2010	Painter	514	C
MU-199	1	Spring	2010	Packard	101	D
PHY-101	1	Fall	2009	Watson	100	A

The teaches Relation

ID	course_id	sec_id	semester	year
10101	CS-101	1	Fall	2009
10101	CS-315	1	Spring	2010
10101	CS-347	1	Fall	2009
12121	FIN-201	1	Spring	2010
15151	MU-199	1	Spring	2010
22222	PHY-101	1	Fall	2009
32343	HIS-351	1	Spring	2010
45565	CS-101	1	Spring	2010
45565	CS-319	1	Spring	2010
76766	BIO-101	1	Summer	2009
76766	BIO-301	1	Summer	2010
83821	CS-190	1	Spring	2009
83821	CS-190	2	Spring	2009
83821	CS-319	2	Spring	2010
98345	EE-181	1	Spring	2009

Primary Keys

- For each table, the database designer chooses a primary means of identifying a tuple: the **primary key**.
- Notation: underline the attributes which form the primary key.
 - instructor (<u>ID</u>, name, dept_name, salary)
 - section (<u>course_id</u>, <u>sec_id</u>, <u>semester</u>, <u>year</u>, building, room_number, time_slot_id)
 - teaches (<u>ID</u>, <u>course_id</u>, <u>sec_id</u>, <u>semester</u>, <u>year</u>)
- The primary key constraint ensures that two tuples can not have the same values for the primary key attributes
 - For example, the database will reject adding a new instructor with an existing ID

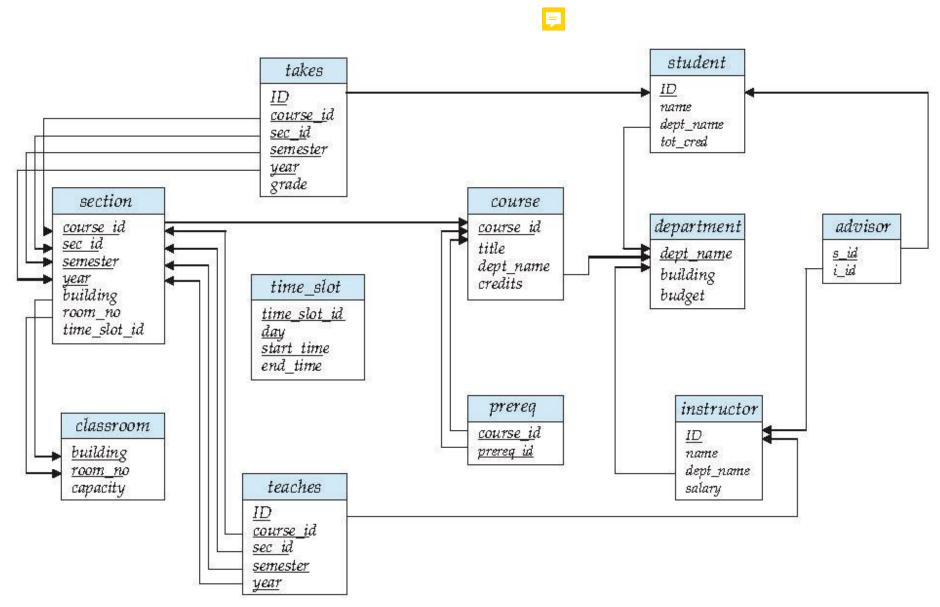
Foreign Keys

- A foreign key is a set of attributes in one relation which is the primary key of another relation
 - Example: the dept_name attribute in the instructor relation is the primary key of the department relation
- Notation:

```
instructor (<u>ID</u>, name, dept_name, salary)
```

- □ dept_name → department □
- instructor is called the referencing relation
- department is called the referenced relation
- The foreign key constraint says that attribute values of the foreign key in the referencing relation have to occur in the referenced relation
 - The database will reject adding an instructor with a department which is not in the department relation

Schema Diagram for University Database



Other Data Models

- We only talk about the relational data model.
 - It essentially says "data is a set of tables".
 - Most data fits well into tables how about your data?
- There are also other, non-relational, data models
 - Semi-structured data "data is a set of tree-like documents" (like JSON or XML). Example: MongoDB
 - Graph data, or object-oriented data
 "data is pieces of data which include pointers to other pieces of data"
 Example: Neo4J
 - Key-value stores, "data is tables with two columns"
 Example: Redis