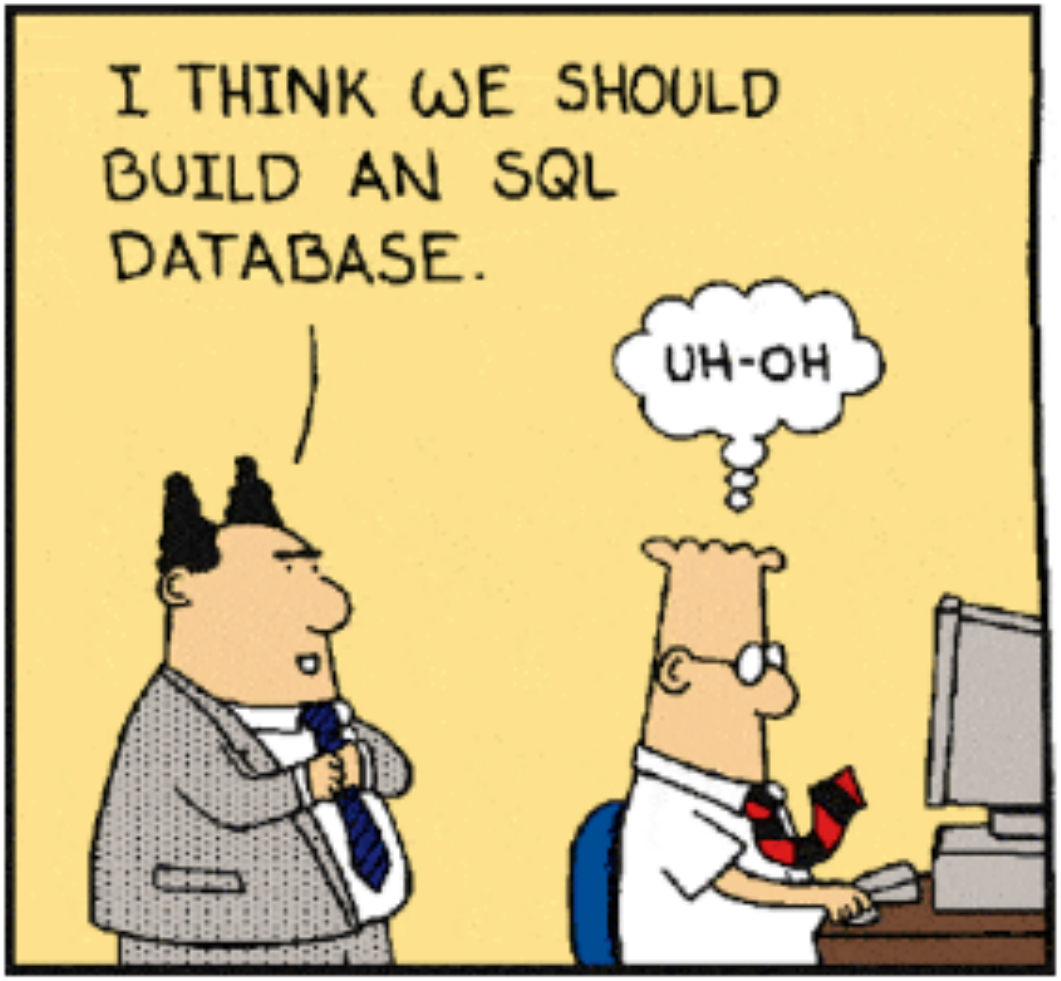
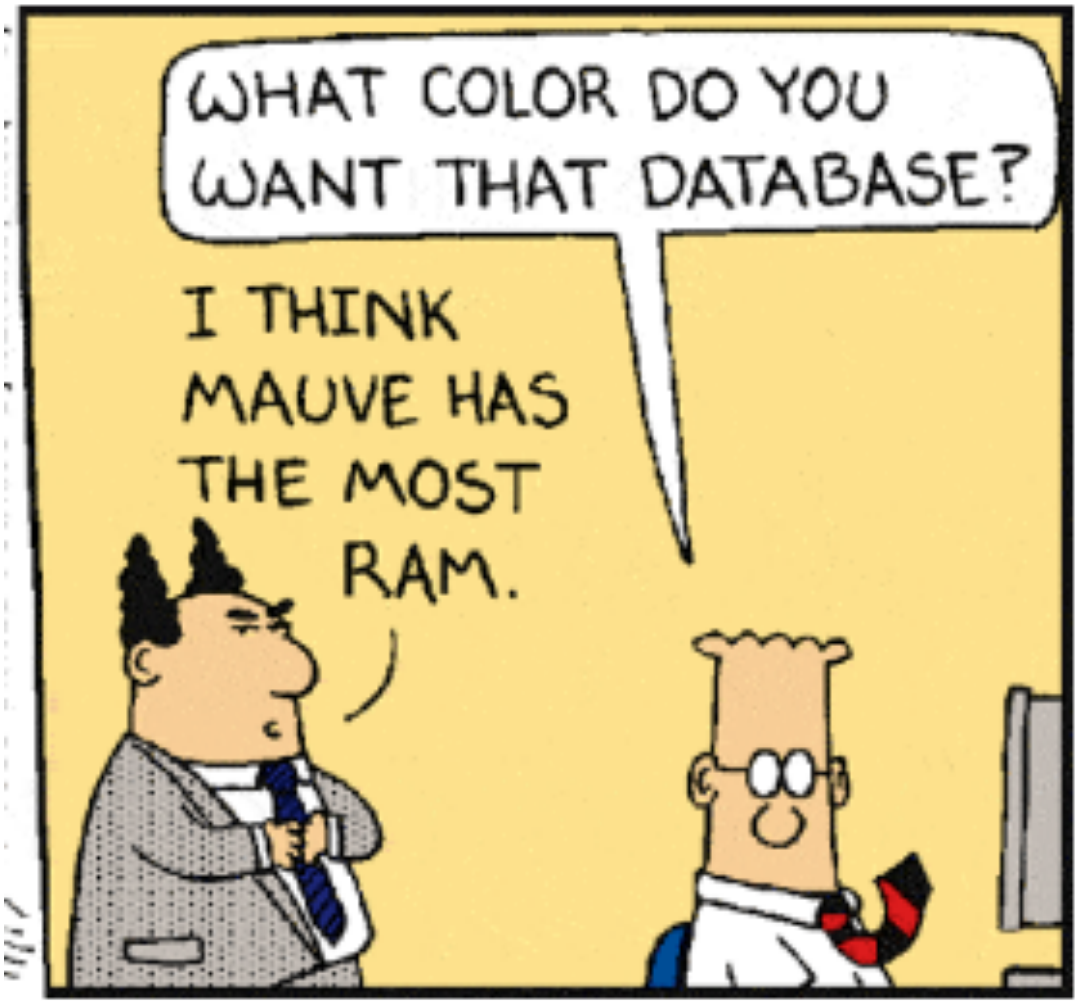

I THINK WE SHOULD BUILD AN SQL DATABASE.
UH-OH


DOES HE UNDERSTAND WHAT HE SAID OR IS IT SOMETHING HE SAW IN A TRADE MAGAZINE AD?


WHAT COLOR DO YOU WANT THAT DATABASE?
I THINK MAUVE HAS THE MOST RAM.

Databases

Standard stuff

- Class webpage: `cs.rhodes.edu/db`
- Textbook: get it somewhere; used is fine
  - Stay up with reading!
- Prerequisite: CS 241
- Coursework:
  - Homework, group project, midterm, final
- Be prepared to bring laptops every so often.

Group project

- You will design and implement your own database-driven website.
- Ideas: shopping, auctions, write a better BannerWeb, library/bibliography system, reviews a la Yelp, bank, finance/stocks, job postings, social networking a la Facebook, recipes, movies, apartments, …
- Groups: probably 4-5 people, formed on your own.
- Spread out over the whole semester; check-ins along the way.

Why study databases?

- Academic reasons
- Programming reasons
- Business (get a job) reasons
- Student reasons

What will you learn?

- Database design
  - How do you model your data so it can be stored in a database?
- Database programming
  - How do I use a database to ask it questions?
- Database implementation
  - How does the database itself work; i.e., how does it store, find, and retrieve data efficiently?

What is the goal of a database?

- Electronic record-keeping, enabling **fast** and **convenient** access to the information inside.
- DBMS = Database management system
  - Software that stores individual databases and knows how to search the information inside.
  - RDBMS = Relational DBMS
  - Examples: Oracle, MS SQL Server, MS Access, MySQL, PostgreSQL, IBM DB2, SQLite

DBMS Features

- Support massive amounts of data
  - Giga-, tera-, petabytes
- Persistent storage
  - Data continues to live long after program finishes.
- Efficient and convenient access
  - Efficient: don't search the entire thing to answer a question!
  - Convenient: allow users to ask questions as easily as possible.
- Secure, concurrent, and atomic access

Example: build a better BannerWeb

- Professors offer classes, students sign up, get grades
- What are some questions we (students or faculty) could ask?
  - Find my GPA.
  - …
- Why are security, concurrency, and atomicity important here?

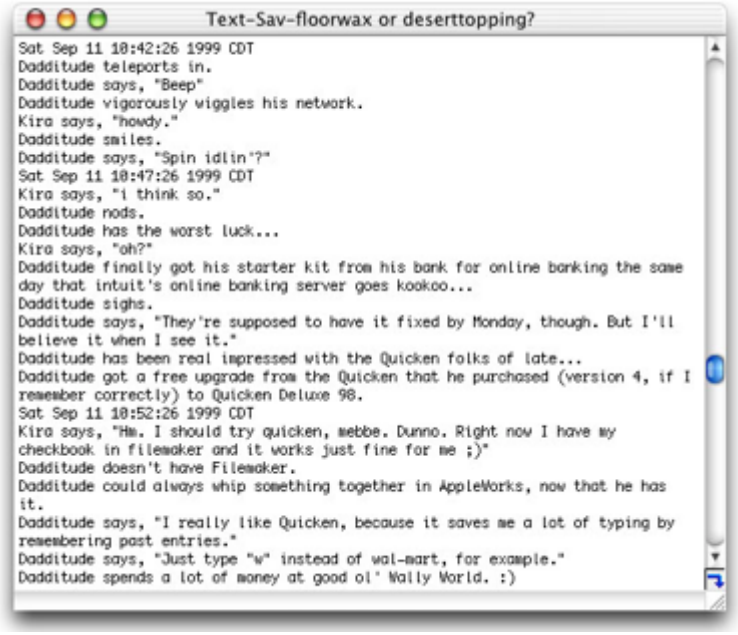
Obvious solution: Folders

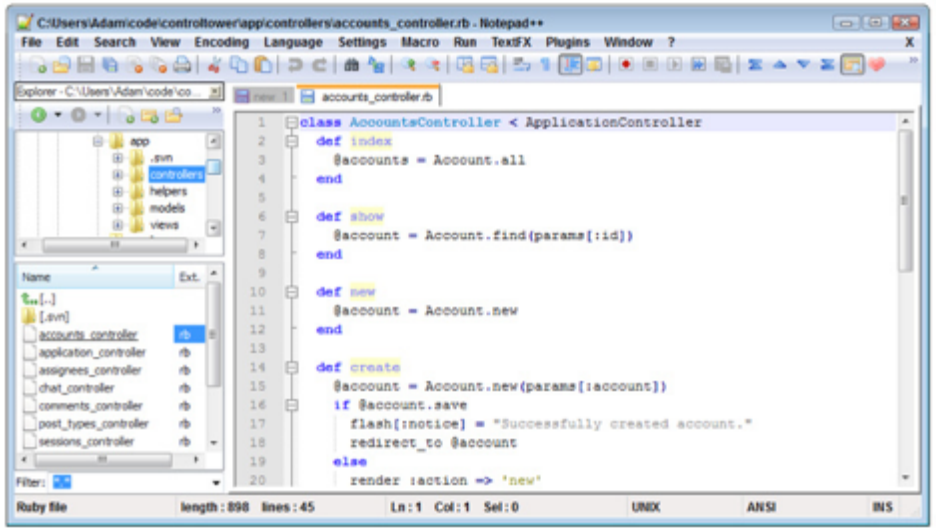
- Advantages?

- Disadvantages?

Obvious solution++

- Text files and Python/C++/Java programs





Obvious solution++

- Let's use CSV:

```
Hermione,Granger,R123,Potions,A
Draco,Malfoy,R111,Potions,B
Harry,Potter,R234,Potions,A
Ronald,Weasley,R345,Potions,C
```

```
Hermione,Granger,R123,Potions,A
Draco,Malfoy,R111,Potions,B
Harry,Potter,R234,Potions,A
Ronald,Weasley,R345,Potions,C
Harry,Potter,R234,Herbology,B
Hermione,Graner,R123,Herbology,A
```

File 1:

```
Hermione,Granger,R123
Draco,Malfoy,R111
Harry,Potter,R234
Ronald,Weasley,R345
```

File 2:

```
R123,Potions,A
R111,Potions,B
R234,Potions,A
R345,Potions,C
R234,Herbology,B
R123,Herbology,A
```

Problems

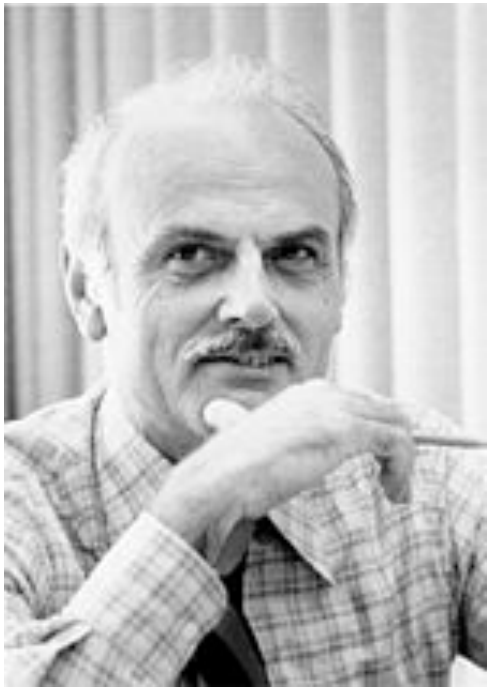
- Inconvenient – need to know Python/C++/Java to get at data!
- Redundancy/inconsistency
- Integrity problems
- Atomicity problems
- Concurrent access problems
- Security problems

Why are there problems?

- Two main reasons:
  - The description of how the files are laid out is buried within the Python/C++/Java code itself (if it's documented at all)
  - There is no support for **transactions** (supporting concurrency, atomicity, integrity, and recovery)
- **DBMSs handle exactly these two problems.**

Relational database systems

- Edgar F. Codd was a researcher at IBM who conceived a new way of organizing data based on the mathematical concept of a ***relation***.
- Relation: a set of ordered tuples (oh, no, CS172 stuff…)



- RDBMS = Relational database management system.
- The relational model uses relations (aka tables) to structure data.
- Grades relation:

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

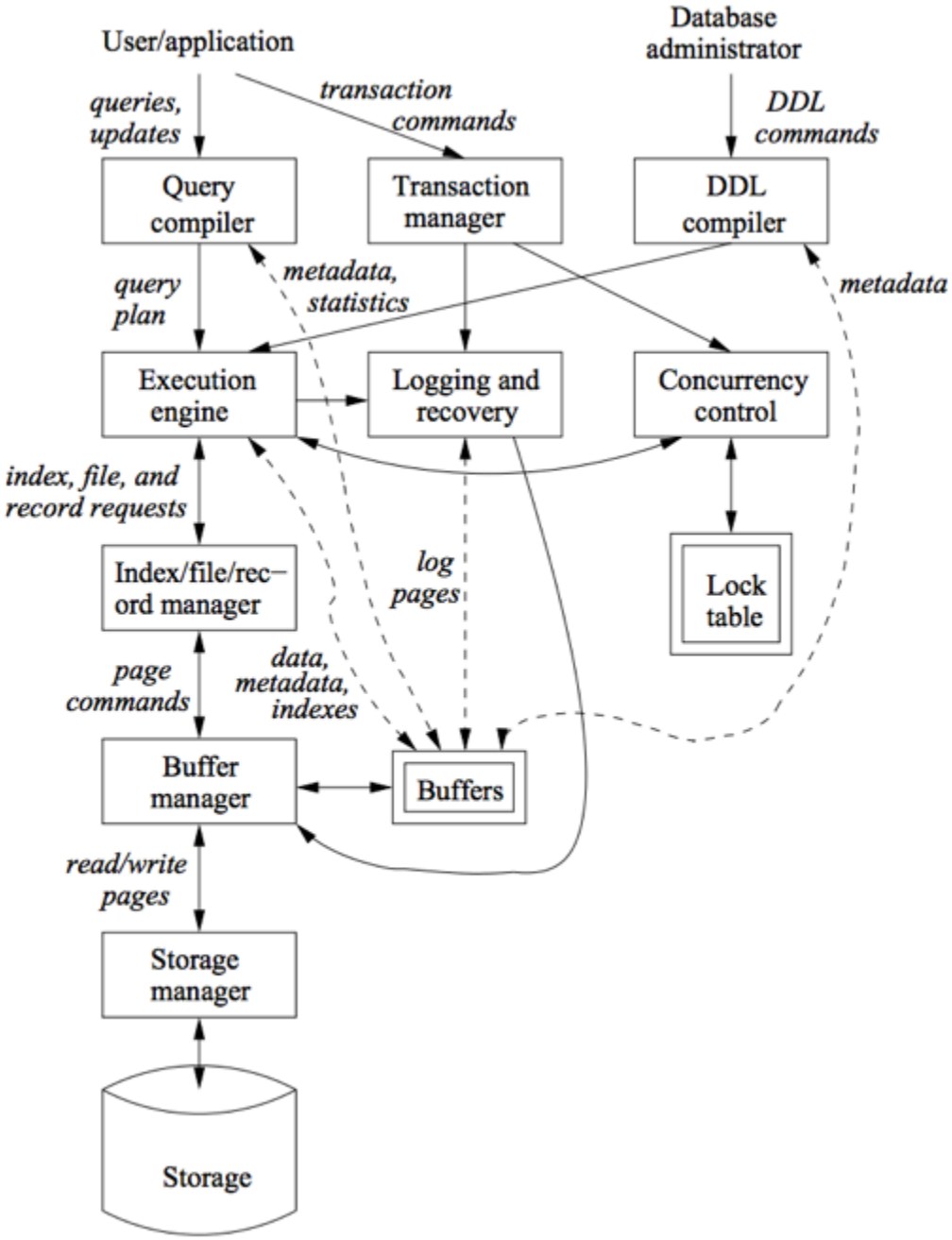
First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

- Relational model is an abstraction.
- Separates the **logical view** (as viewed by the DB user) from the **physical view** (DB's internal representation of the data)

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

- Structured query language (SQL) for accessing/modifying data:
- Find all students who are getting a B.
  - `SELECT First, Last FROM Grades WHERE Grade = "B"`


Huh?
Made by vrbtny


User/application
Database administrator
queries, updates
transaction commands
DDL commands
Query compiler
Transaction manager
DDL compiler
query plan
metadata, statistics
metadata
Execution engine
Logging and recovery
Concurrency control
index, file, and record requests
log pages
Index/file/record manager
Lock table
page commands
data, metadata, indexes
Buffer manager
Buffers
read/write pages
Storage manager
Storage

Transaction processing

- One or more DB operations can be grouped into a **transaction**.
- For a DBMS to properly implement transactions:
- **A**tomicity: All-or-nothing execution of transactions.
- **C**onsistency: A DB can have consistency rules that should not be violated.
- **I**solation: Each transaction must **appear** to be executed as if no other transactions are happening simultaneously.
- **D**urability: Any changes a transaction makes must never be lost.

On to the real stuff now…

Data Models

- A way of describing data.
  - Better: a description of how to conceptually structure the data, what operations are possible on the data, and any constraints on the data.
- Structure: how we view the data abstractly
- Operations: what is possible to do with the data?
- Constraints: how can we control what data is legal and what is not?

Relational model

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

- Structure: relation (table)
- Operations: relational algebra (select certain rows, certain columns, where properties are true/false)
- Constraints: can enforce restrictions like Grade must be in {A, B, C, D, F}

Other models

- Semi-structured data that is still “structured” but not in relational format.
  - XML, JSON
- Object databases, or object-relational
- Graph databases
- NoSQL, NewSQL

Semi-structured model

- Structure: Trees or graphs
  - e.g., XML
- Operations: Follow paths in the implied tree from one element to another.
  - e.g., XQuery
- Constraints: can constrain data types, possible values, etc.
  - e.g., DTDs (document type definition), XML Schema

Object-relational

- Similar to relational, but
  - Values in a table can have their own structure, rather than being simple strings or ints.
  - Relations can have associated methods.

Relational model is most common

- Simple: built around a single concept for modeling data: the relation or table.
  - A relational database is a collection of relations.
  - Each relation is a table with rows and columns.
  - An RDBMS can manage many databases at once.
- Supports high-level programming language (SQL)
  - Limited but useful set of operations.
- Has elegant mathematical theory behind it.

Relation Terminology

- Relation == 2D table
  - **Attribute** == column name
  - **Tuple** == row (not the header row)
- Database == collection of relations

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

Relation Terminology

- A relation includes two parts:
  - The relation **schema** defines the column headings of the table (attributes/fields)
  - The relation **instance** defines the data rows (tuples, rows, or records) of the table.

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

Schema

- A schema is written by the name of the relation followed by a parenthesized list of attributes.
  - `Grades(First, Last, Course, Grade)`
- A **relational database schema** is the set of schemas for all the relations in a DB.

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

Domains

- A relational DB requires that every component of a row (tuple) have a specific elementary data type, or **domain**.
  - string, int, float, date, time (no complicated objects!)

Grades(First:string, Last:string, Course:string, Grade:char)

Equivalent representations of a relation

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

```
Grades(First, Last, Course, Grade)
```

- Relation is a ***set*** of tuples, not a list.
- Attributes in a schema are a ***set*** as well.
  - However, the schema specifies a "standard" order for the attributes.
- How many equivalent representations are there for a relation with *m* attributes and *n* tuples?

Degree and cardinality

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

- **Degree/arity** of a relation is the number of attributes in a relation.
- **Cardinality** is the number of tuples in a relation.

Keys to a good relation(ship)

Keys of a relation

- Keys are a kind of **integrity constraint**.
- A set of attributes K forms a key for a relation R if:
  - we forbid two tuples in an instance of R to have the same values for all attributes of K.

First	Last	Course	Grade
Hermione	Granger	Potions	A
Draco	Malfoy	Potions	B
Harry	Potter	Potions	A
Ronald	Weasley	Potions	C

`Grades(First, Last, Course, Grade)`

Keys of a relation

- Keys help associate tuples in different relations.

SID	First	Last
123	Hermione	Granger
111	Draco	Malfoy
234	Harry	Potter
345	Ronald	Weasley

SID	CRN	Grade
123	777	A
111	777	B
234	777	A
345	777	C

CRN	Name	Semester	Year
777	Potions	Fall	1997
888	Potions	Spring	1997
999	Transfiguration	Fall	1996
789	Transfiguration	Spring	1996

Example

- Let's expand these relations to handle the kinds of things you'd like to see in BannerWeb.
- Keep track of students, professors, courses, who teaches what, enrollments, pre-requisites, grades, departments & their chairs.
  - Only one chair per department.
  - Student cannot enroll in multiple copies of the same course in one semester.
  - Other constraints that are logical.