

SQL Unit I

An Introduction to Relational Databases

Ryan Nixon

First, a note

- Multiple SQL applications may be used for this course. Because of this, instruction will cover the differences in syntax you may encounter in each.
- Supported implementations are SQLite, MySQL, Microsoft SQL Server, Microsoft Access and OpenOffice

First, a note

- When in doubt, use Microsoft Access. It is already installed in all UAA campus computer labs, including the CS lab.
- The CS lab also supports the other implementations (SQLite, MS SQL & MySQL)

Databases

- By definition: “A structured set of data held in a computer, especially one that is accessible in various ways.”
- Contains data in an organized manner and optimizes retrieval of that data
- Used anywhere that large sets of data need to be stored, organized, and/or retrieved by multiple users

Databases

- In essence: If Microsoft Excel can't do the job, you're better off with a database

“Relational”

- This course will focus specifically on relational databases
- Relational databases have that name because of their reliance on items that are interrelated
- Other databases, such as document-oriented databases, will not be covered

SQL

- “Structured Query Language”: A Language for Querying Structured data
- Used to create, update, delete, and retrieve information from the database
- Extremely flexible, but suffers from different implementations

Examples of Databases

- What databases are already out there?
What is the largest? The smallest?

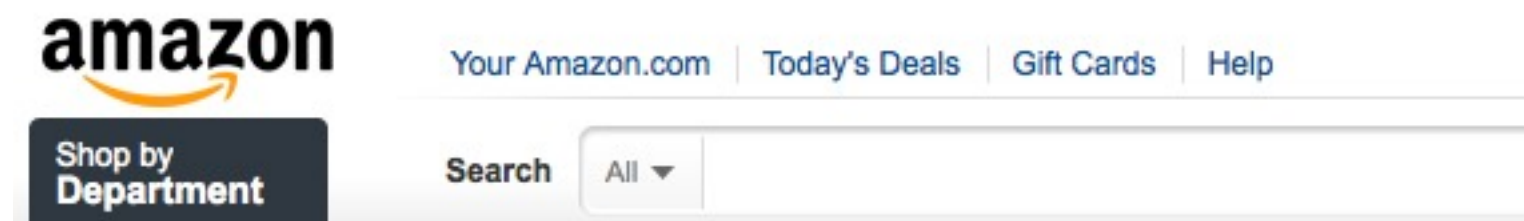
Examples of Databases

The Google logo, featuring the word "Google" in its characteristic multi-colored font (blue, red, yellow, blue, green, red).


Google Search

I'm Feeling Lucky

Examples of Databases



Examples of Databases


 LIBRARY OF CONGRESS

ASK A LIBRARIAN

DIGITAL COLLECTIONS

LIBRARY CATALOGS

About the Library






II


Search

All Formats ▾ Search Loc.gov


GO

Collection Highlights


 Print  Subscribe  Share/Save



American Memory



Prints & Photographs



Historic Newspapers

Topics

[American History](#)

[Arts & Culture](#)

[Government, Politics & Law](#)

[Maps & Geography](#)

[News & Journalism](#)

Examples of Databases



Database Structures

- Collections of tables, views, indexes, procedures, functions, etc. that are meant to hold or operate on the data
- Some names change according to the implementation. For example, “table” may be referred to as “file” or “relation”
- Tables contain the actual data, stored in rows or “records”

Tables & Entities

- Tables store “entities”, individual objects that have common characteristics
- Cats, Dogs, Humans, Birds. All are animals. They could fit in an “animal” table
- Entities can also be abstract, such as Work Days, or Weather Patterns (“work_day” and “weather_pattern”, respectively)

Naming Conventions

- Table names can be any sequence of most characters, but there are conventions to each name
- Table names generally follow these rules:
 - Lowercase
 - Underscores rather than spaces
 - Singular rather than plural

Naming Conventions

- Why plural? It's best to think of the table as the entity itself rather than a collection. This prevents confusion when discussing the table, i.e. "the animals table contains animals...s"
- It is also important as many software applications will automatically pluralize the table names when retrieving multiple records

Keys & Constraints

- In order to maintain table entities, constraints are applied to the attributes within those tables
- “Keys” are an important type of constraint that manage the relations between tables

Keys

Primary Key (“Identity”)

- Must be unique
- Can apply to many attributes in a table
- Often a number that automatically increases with each row
- Usually named “id”

Keys

Foreign Key (“Reference”)

- Contents must be empty or a value from another table
- Often references the primary key of the other table
- Usually suffixed with “_id” to denote that it is pointing at another key

Constraints

- **Not Null** - Value must not be empty
 - Note that “”, 0, or FALSE are not considered to be null
- **Unique** - No two values may be exactly the same
- **Default** - If a value is not provided, it will take on a default value

Table Relationships

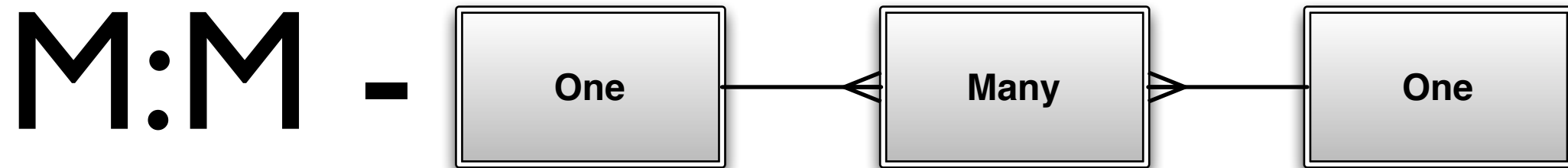
- All tables in a database have connections. These data relationships are extremely important to the system as without them the data would not relate
- The three major types of relationships are One to One, One to Many, and Many to Many
- Feel free to chuckle now before moving on



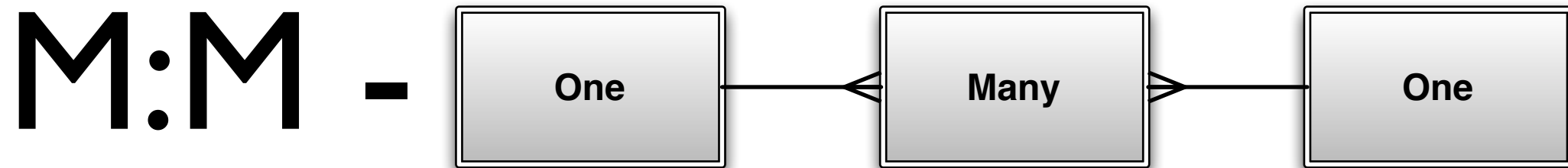
- One to many relationships are the most common relation in databases
- One entity may hold/relate to multiple other entities
- In this case, the primary key of the “One” table would be a foreign key in the “Many” table



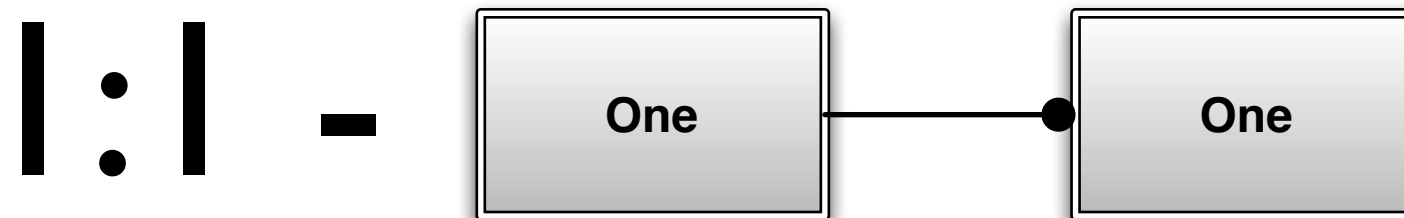
- *One* classroom contains *Many* students
- *One* Car manufacturer creates *Many* models, and distributes *Many* cars per year
- *One* Artist might write *Many* Albums, which contain *Many* tracks



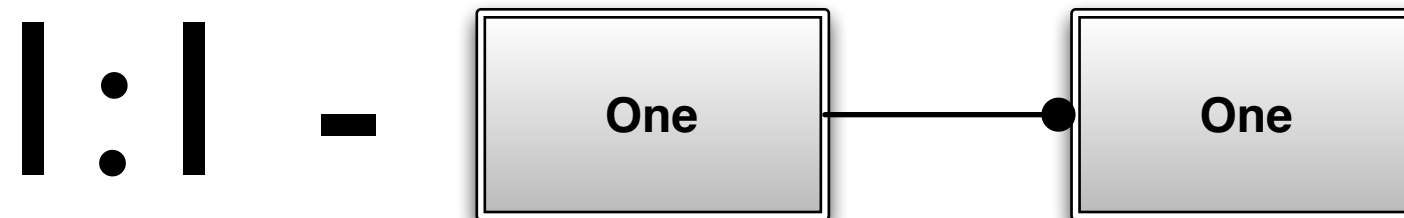
- Many to Many tables allow the relationship to go both ways and thus are extensions of 1:M relationships
- By placing both of the primary keys as foreign keys in a middle table, both “One” tables can have many of each other



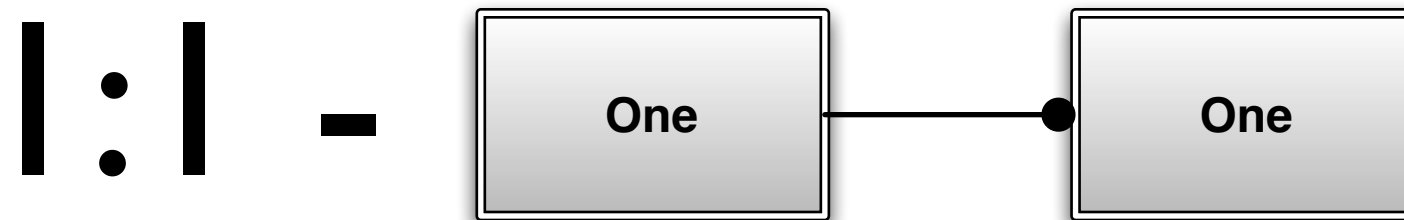
- Students may register for *Many* classes, and classes may have *Many* students registered
- Cars may be taken to *Many* repair shops, and repair shops may handle *Many* cars
- Music companies may contract *Many* artists, and artists may contract with *Many* different music companies



- One to one relationships are quite different. They are best used to provide additional data about an entity that might not apply to all the rows
- Think of them as “is a” relationships



- There are also two ways to define them
 - Make the primary key of one the primary key of the other
 - Make the primary key of one a foreign key in the other and make the foreign keys *unique*. This allows for future behavior if the relationship might ever become 1:M



- A student *is a* person that attends classes
- A convertible *is a* car with a removable top
- A one-hit-wonder *is an* artist with a single hit song

Forms of Integrity

- The stability of the table structure and data is known as “integrity”
- There are 3 forms of integrity in a relational database: Data Integrity, Entity Integrity, and Referential Integrity

Data Integrity

- Data integrity is the requirement that all data in all fields of a database be valid and correct.
- If an attribute has a Not Null constraint and a user enters garbage data to satisfy it, this would violate data integrity
- A better value would be “”, or 0 (provided that 0 does not have meaning)

Entity Integrity

- Every table has to have a primary key field, and no part of the primary key field can be null for any record in the table
- This is generally enforced directly by the database

Referential Integrity

- Every value that appears in a foreign key field also has to appear as a value in the corresponding primary key field
- In other words, you won't be able to enter a value in the keyed field unless the item is first present in the other table

Data Types

- Data types further enforce values, making them match known formats such as dates, times, decimal numbers, etc.
- This is done both to ensure data is valid but also to optimize the storage and retrieval of those values
- These fields will be covered in more detail later in the course

Data Types

- Numeric
 - bigint (± 9 quintillion)
 - int (± 2 billion)
 - smallint (± 32 thousand)
 - tinyint (0 - 255)
 - bit (0-1)

Data Types

- Approximate Numerics
 - float, double (precision specified)
- Exact Numerics
 - decimal, numeric (precision specified)

Remember

- Unit 1 assignment up by Friday
- No class on Monday. Enjoy your Labor day!
- Meeting in SSB 112 lab on Wednesday