

Week 2

Data Models

CPSC 50900

Last time we studied storage models

- Storage models concern
 - How the files that store the data are arranged on a disk
 - How the data within each file are organized with respect to other data
- We saw five different storage models
 - Hierarchical
 - Network
 - Object-Oriented
 - NoSQL
 - Relational
- We saw several advantages to the Relational Model
- We saw that a Relational Database Management System can maximize these advantages and offer additional benefits

Storage models are physical models.

But how do we model what the data means?

- *Data Models* help us capture essential elements of the real world in our data
- Coming up with data models involves
 - Identifying the kinds of things we want to store data about
 - What aspects of those things we want to store
 - How different kinds of things are related to each other
- Data Models are the focus for this week.

Three kinds of data models

- Conceptual
- Logical
- Physical

Conceptual data model

- Most basic and abstract
- Focuses on
 - Entities
 - The things we are storing
 - Relationships
 - How those things are related
- No talk of attributes yet
- Audience: business stakeholders and data architects

Example of a conceptual model: a library management system

- Entities:
 - Books
 - Authors
 - Borrowers
- Relationships
 - Books are written by authors
 - Books are borrowed by borrowers

Logical data model

- Focuses on
 - Entities
 - Keys
 - Relationships
 - Attributes
- Without focusing on physical implementation details
- Audience: data architects and database designers
- Involves the process of *normalization*
 - A process of mapping attributes to entities to minimize redundancy

Example of a logical model: a library management system

- Books
 - Title,
 - Genre
 - Author
- Authors:
 - Name
 - Country
- Borrowers:
 - Name
 - MembershipDate

Example of a logical model: a library management system

- Books
 - BookID (PK)
 - Title
 - Genre
 - Author
- Authors:
 - AuthorID (PK)
 - Name
 - Country
- Borrowers:
 - BorrowerID (PK)
 - Name
 - MembershipDate

Add keys to uniquely identify each entity

These are called primary keys (PK)

Physical data model

- Most specific / concrete
- Translates the data to model to optimized storage with a particular DBMS
- For relational models, this involves mapping
 - Entities to tables
 - Attributes to columns of tables
 - Relationships to columns that are duplicated in two tables
 - foreign keys
 - Adding **bridge entities** for many-to-many relationships
- Also considers optimization
 - Indexes
 - Views
 - Triggers
 - Partitions
 - Clustering
 - Denormalization
- End product: SQL scripts for building the database

Example of a physical model: a library management system

- Books
 - BookID (PK)
 - Title,
 - Genre
 - Author
- Authors:
 - AuthorID (PK)
 - Name
 - Country
- Borrowers:
 - BorrowerID (PK)
 - Name
 - MembershipDate

Relationships:

A book has an author:
AuthorID

Add a foreign key (FK) to
create this relationship.

Example of a logical model: a library management system

- Books
 - BookID (PK)
 - Title,
 - Genre
 - AuthorID (FK)
- Authors:
 - AuthorID (PK)
 - Name
 - Country
- Borrowers:
 - BorrowerID (PK)
 - Name
 - MembershipDate

Relationships:

A book has an author:

AuthorID (FK)

Example of a logical model: a library management system

- Books
 - BookID (PK)
 - Title,
 - Genre
 - AuthorID (FK)

Relationships:

A book has an author: AuthorID (FK)

- Authors:
 - AuthorID (PK)
 - Name
 - Country

A borrower can borrow many books

A book can be borrowed by many borrowers

- Borrowers:
 - BorrowerID (PK)
 - Name
 - MembershipDate

Example of a logical model: a library management system

- Books
 - BookID (PK)
 - Title,
 - Genre
 - AuthorID (FK)

Relationships:

A book has an author: AuthorID (FK)

- Authors:
 - AuthorID (PK)
 - Name
 - Country

A borrower can borrow many books

A book can be borrowed by many borrowers

- Borrowers:
 - BorrowerID (PK)
 - Name
 - MembershipDate

We probably will need
a place to store
borrowing data

Example of a physical model: adding a bridge entity to library management system

- Books
 - BookID (PK)
 - Title,
 - Genre
 - AuthorID (FK)
- Authors:
 - AuthorID (PK)
 - Name
 - Country
- Borrowers:
 - BorrowerID (PK)
 - Name
 - MembershipDate
- BookLending
 - Borrower
 - Book
 - DateLent
 - DateDue

Example of a physical model: adding a bridge entity to library management system

- Books
 - BookID (PK)
 - Title,
 - Genre
 - AuthorID (FK)
- Authors:
 - AuthorID (PK)
 - Name
 - Country
- Borrowers:
 - BorrowerID (PK)
 - Name
 - MembershipDate
- BookLending
 - BookLendingID (PK)
 - BorrowerID (FK)
 - BookID (FK)
 - DateLent
 - DateDue

Example of a physical model: Optimization of library management system

- Add index on AuthorID in the Book table for faster lookup by author
- Add index on BorrowerID and BookID in the BookLending table for faster lookup by borrower and book

Example of a physical model:
Write the SQL scripts to build the actual physical database

```
CREATE TABLE Books (  
    BookID INT PRIMARY KEY AUTO_INCREMENT,  
    Title VARCHAR(255),  
    Genre VARCHAR(50),  
    AuthorID INT,  
    FOREIGN KEY (AuthorID) REFERENCES  
Authors(AuthorID)  
);
```

```
CREATE TABLE Borrowers (  
    BorrowerID INT PRIMARY KEY  
AUTO_INCREMENT,  
    Name VARCHAR(255),  
    MembershipDate DATE  
);
```

```
CREATE TABLE Authors (  
    AuthorID INT PRIMARY KEY  
AUTO_INCREMENT,  
    Name VARCHAR(255),  
    Country VARCHAR(100)  
);
```

Let's do more examples,
focusing on conceptual models

Conceptual model - review

- What are we storing data about?
 - What are the *entities*?
- How are those things related to each other?
 - What are the *relationships* among the entities?

Example: Movie Rentals

- Concerning movie rentals, what might we store data about?
 - The movie
 - The renter
 - The streaming service we rented it from
 - The rental reservation
- These are the *entities* that make up our data model

How are the entities related to each other in the movie rental example?

- A renter makes several rental reservations.
Each rental reservation is made by a renter.
- A rental reservation is made from one streaming service.
A streaming service fulfills several rental reservations.
- A rental reservation is made for one movie.
A movie may be rented through several rental reservations over time.

Example: University course catalog

- What are the entities we want to store data about?
 - Academic department
 - Course
 - Course Section
 - Student
 - Teacher

What are the relationships among these entities in the University catalog example?

- A department offers many courses
Each course is offered by one department
- A course has many course sections.
Each course section belongs to a course.
- A course section is taught by one teacher.
A teacher can teach many course sections.
- A student takes many course sections.
Each course section is taken by many students.

Example: Auto insurance

- What are the entities we want to store data about?
 - Customer
 - Driver
 - Agent
 - Insurance plan
 - Coverage
 - Auto

What are the relationships among the entities in the insurance example?

- A customer has perhaps several insurance plans.
Each insurance plan covers one customer.
- Each driver is assigned to a car.
A car can be assigned to several drivers.
- An insurance plan covers several drivers.
A driver is covered by one insurance plan.
- An insurance plan offers several types of coverages.
A type of coverage could be offered by several insurance plans.
- An agent sells many insurance plans.
Each insurance plan is sold by one agent.

Relationships have connectivity

- Three kinds of connectivity:
 - One-to-one (1:1)
 - One-to-many (1:M)
 - Many-to-many (M:N)

Example: Identify the connectivity of each relationship

- A department offers many courses
Each course is offered by one department
- A course has many course sections.
Each course section belongs to a course.
- A course section is taught by one teacher.
A teacher can teach many course sections.
- A student takes many course sections.
Each course section is taken by many students.

Relationships can be mandatory or optional

- This is called *optionality*
- A relationship could have different optionality in each direction.
- For example:
 - A course section must have a teacher (mandatory)
 - A teacher doesn't need to teach any course sections (optional)

Example: Describe the optionality in each case

- The relationship between movie and rental agreement
- The relationship between customer and insurance plan
- The relationship between student and course section

How do we share conceptual models (and logical and physical models for that matter) conveniently?

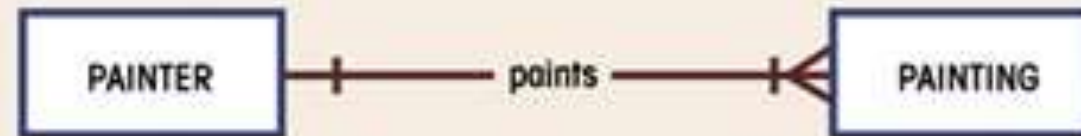
Entity Relationship Diagrams

An Entity Relationship Diagram (ERD) can show entities and relationships

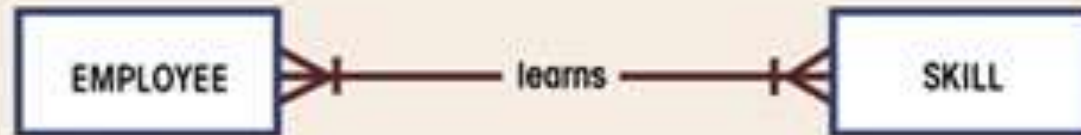
- An entity-relationship diagram (ERD) shows
 - Entities
 - Relationships
 - Attributes
- Can be used for all three kinds of data models – conceptual, logical, physical

Crow's Foot Notation for ERDs

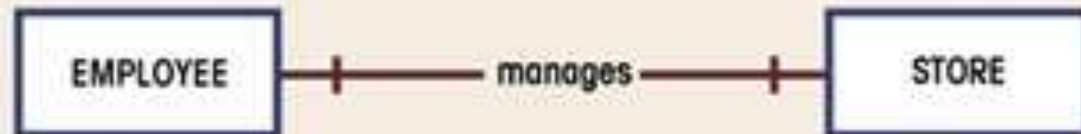
A one-to-many (1:M) relationship: A PAINTER can paint many PAINTINGs;
each PAINTING is painted by one PAINTER



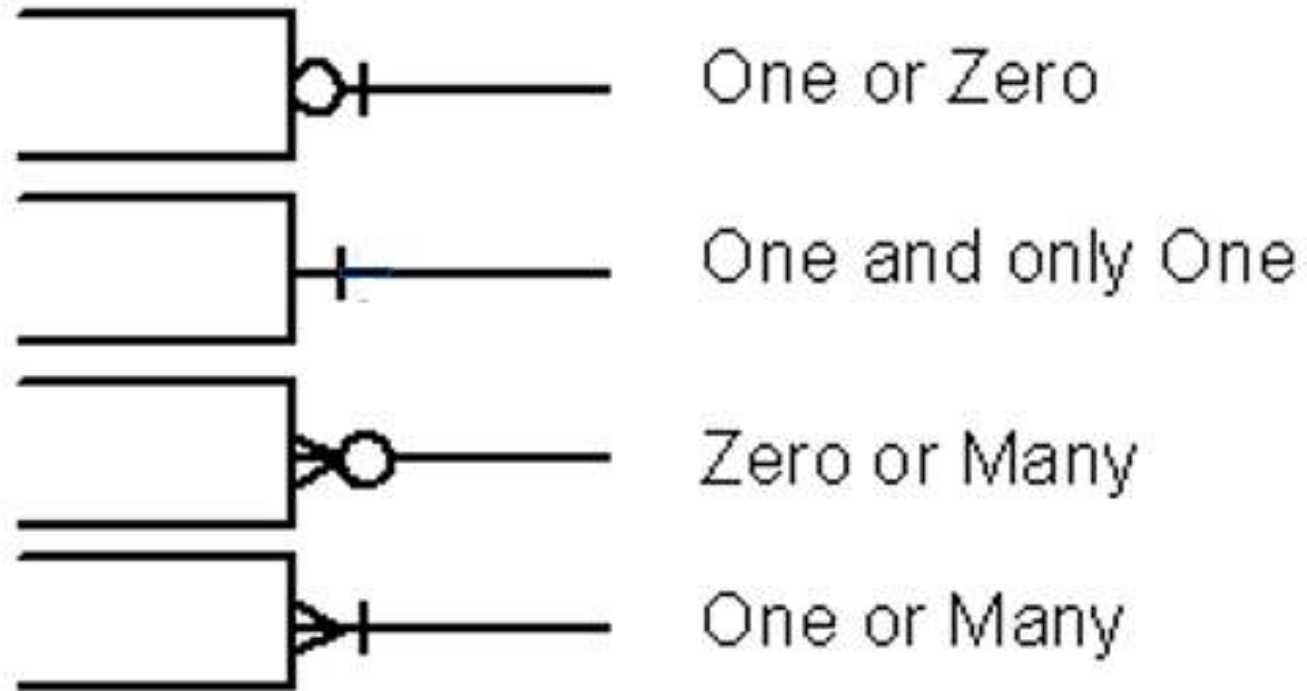
A many-to-many (M:N) relationship: an EMPLOYEE can learn many SKILLs;
each SKILL can be learned by many EMPLOYEEs



A one-to-one (1:1) relationship: an EMPLOYEE manages one STORE;
each STORE is managed by one EMPLOYEE



Showing optionality on an ERD



Example: Draw ERD for Movie Rental Conceptual Model

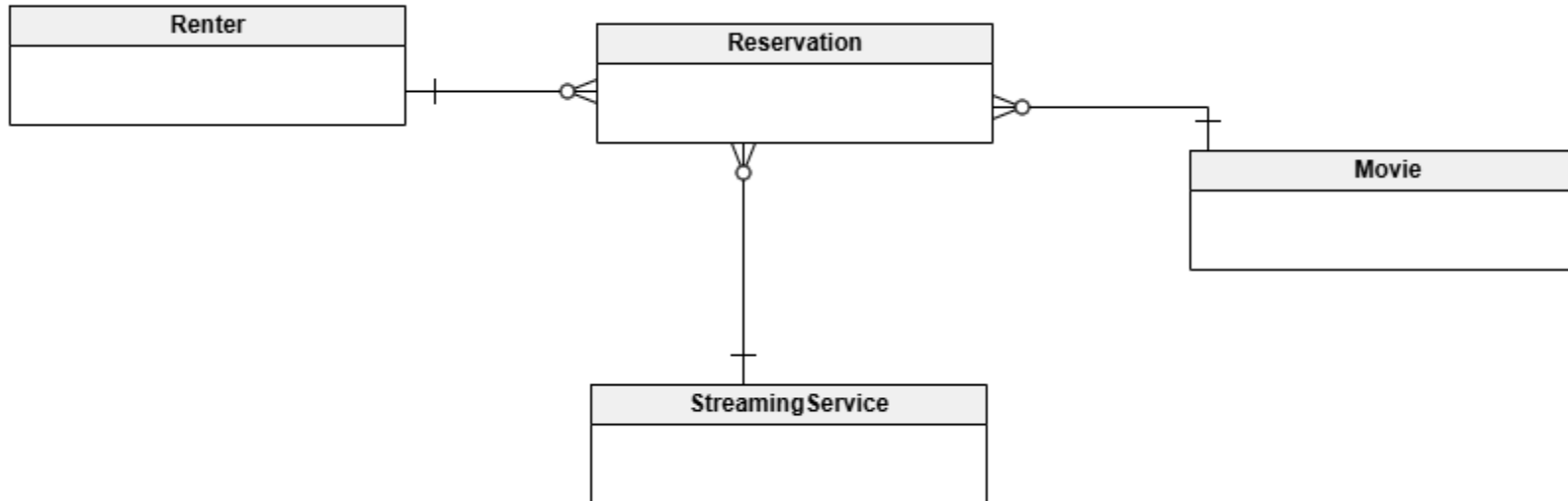
Concerning movie rentals, what might we store data about?

- The movie
- The renter
- The streaming service we rented it from
- The rental reservation

A renter makes several rental reservations.
Each rental reservation is made by a renter.

A rental reservation is made from one streaming service.
A streaming service fulfills several rental reservations.

A rental reservation is made for one movie.
A movie may be rented through several rental reservations over time.



Example: Draw an ERD for the University Catalog Example

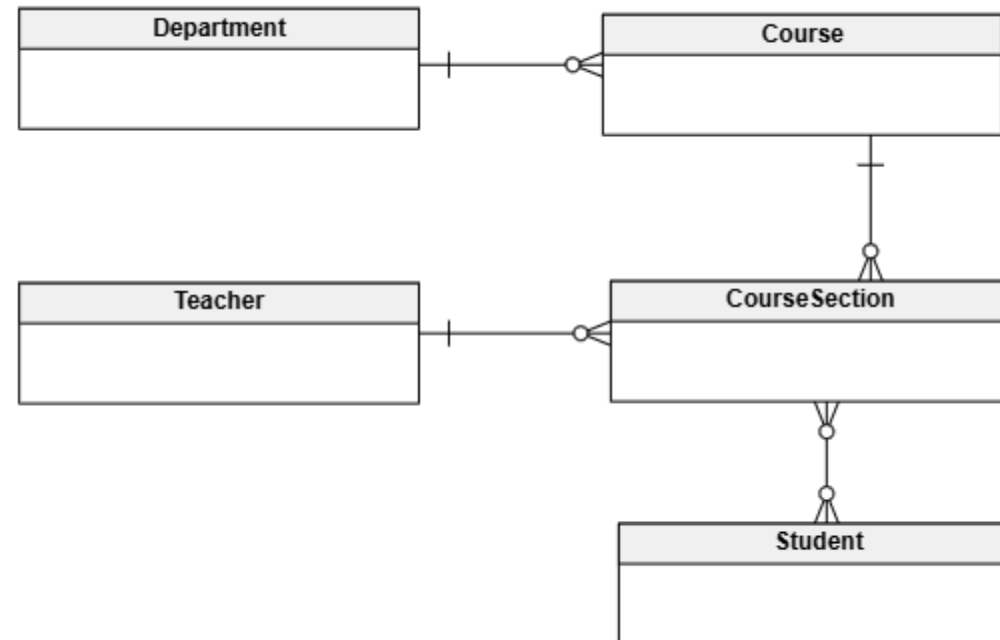
Academic department
Course
Course Section
Student
Teacher

A department offers many courses.
Each course is offered by one department.

A course has many course sections.
Each course section belongs to a course.

A course section is taught by one teacher.
A teacher can teach many course sections.

A student takes many courses.
Each course is taken by many students.



Example: Draw an ERD for the insurance example

Customer
Driver
Agent
Insurance plan
Coverage
Auto

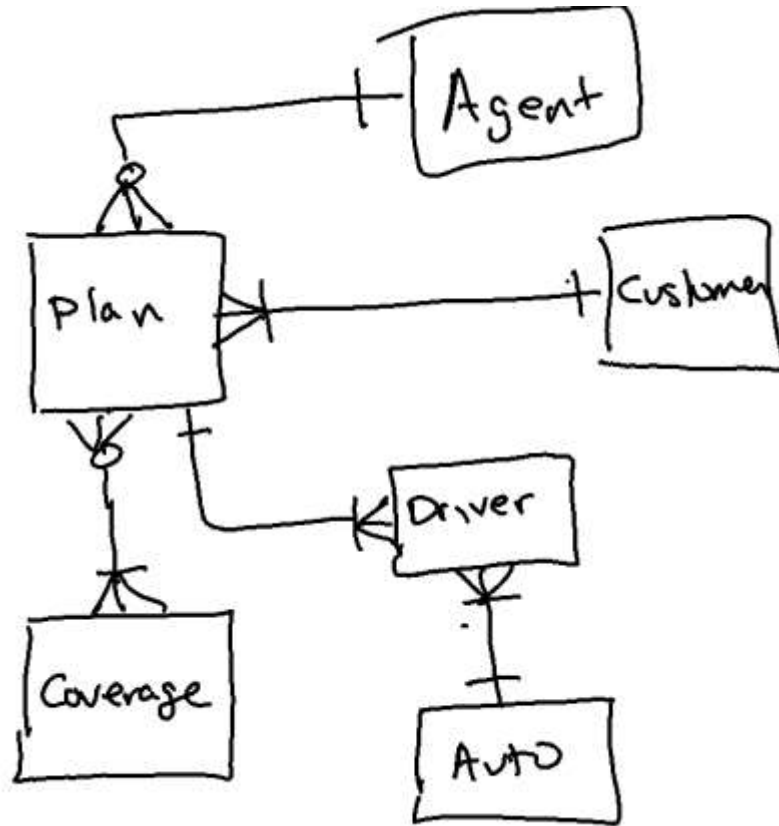
A customer has perhaps several insurance plans.
Each insurance plan covers one customer.

Each driver is assigned to a car.
A car can be assigned to several drivers.

An insurance plan covers several drivers.
A driver is covered by one insurance plan.

An insurance plan offers several types of coverages.
A type of coverage could be offered by several insurance plans.

An agent sells many insurance plans.
Each insurance plan is sold by one agent.



Professional drawing tool – Vertabelo

<https://www.vertabelo.com/>



- Can draw a conceptual model and then automatically generate a physical model from it.
- Register for the student account.

Create a trial account

First and last name:


Email:

Password:

☐ I agree to [Terms of Service](#) and [Privacy Policy](#)

[Create account](#) [Cancel](#)

Student or lecturer?
[Register for an academic account.](#)



Summary

- We learned how data models are used to capture the meaning of a data set.
- There are three types of data models:
 - Conceptual
 - Logical
 - Physical
- We focused on conceptual
 - Entities
 - Relationships
- We learned how to draw ERDs that express the logical model
- Next week – we'll add attributes to the entity sets, including data types
 - Logical model
 - Physical model