CS 250
Databases and DBMSs

Benjamin Dicken

Databases and DBMSs

- The word ***database*** is thrown around a lot, but it is often misused
- When learning about databases it is important to distinguish between a ***database*** and a ***DBMS***

Databases and DBMSs

- A ***Database (DB)*** is an organized collection of data, typically organized to model aspects of reality in a way that supports external processing
- A ***Database Management System (DBMS)*** is a computer software application that interacts with the user, other applications, and the *database* itself to capture and analyze data

Thanks, Wikipedia!

Databases and DBMSs

- A database is not a **program**
- It is a collection of information (typically one or more files on a computer) that represent reality
- We will discuss several ways in which databases can **model** reality

Databases and DBMSs

- A database could be:
  - Something a simple as a single CSV file, where each line represents an entity, and each column a bit of information!
  - A collection of file in a directory that have information related to each-other
  - An entire hard-drive with organized files and information
  - An entire building full of hard drives with petabytes of information!

Databases and DBMSs

- A **DBMS** *is* a computer program
- A user should interact with the DBMS, not the database
  - A DBMS is a “middle man” between the user and the database
  - A robust DBMS provides features to add, remove, retrieve, and process data in a database
  - Users sends ***queries*** to a DBMS to interact with the data held within it

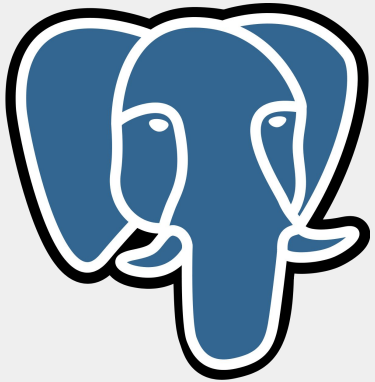
Databases and DBMSs

Databases and DBMSs

- Why is a DBMS necessary? Why Can't we just manually view and edit files using programs like Excel?
  - **Scale** is one issue
    - What if you have a billion pieces of information?
  - **Complexity** is another issue
    - What if you have very complex information and relationships to represent?
- Excel may not be practical

Databases and DBMSs

- Well-known DBMSs include
  - MySQL, SQLite, PostgreSQL, MongoDB, MariaDB, Microsoft SQL Server, Oracle, IBM DB2 ...



Databases and DBMSs

- A DBMS should provide functionality that allows for management of a database and its data
- These functionalities can be classified into four main functional groups

Databases and DBMSs

1) **Data definition** – Creation, modification and removal of definitions that define the organization of the data
2) **Update** – Insertion, modification, and deletion of the actual data
3) **Retrieval** – Providing information in a form directly usable or for further processing by other applications. The retrieved data may be made available in a form basically the same as it is stored in the database or in a new form obtained by altering or combining existing data from the database

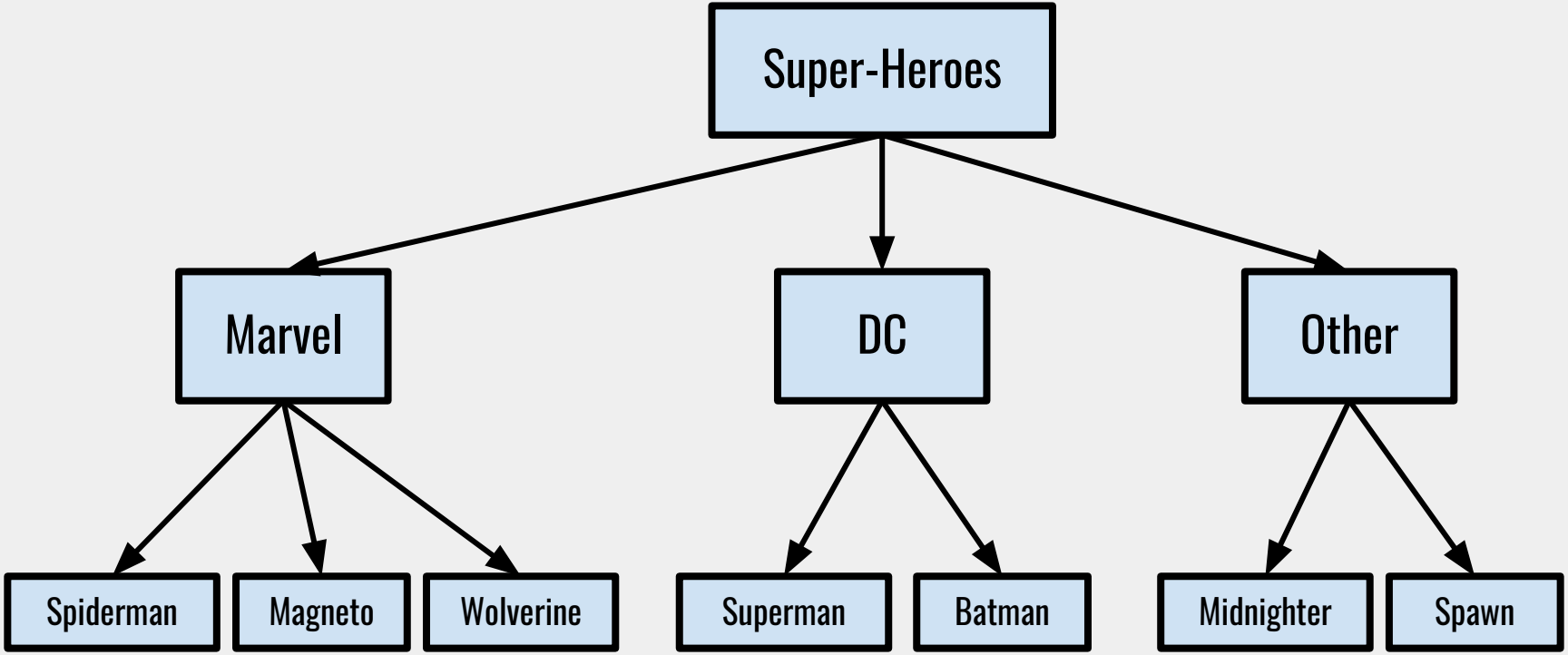
Databases and DBMSs

4) **Administration** – Registering and monitoring users, enforcing data security, monitoring performance, maintaining data integrity, dealing with concurrency control, and recovering information that has been corrupted by some event such as an unexpected system failure

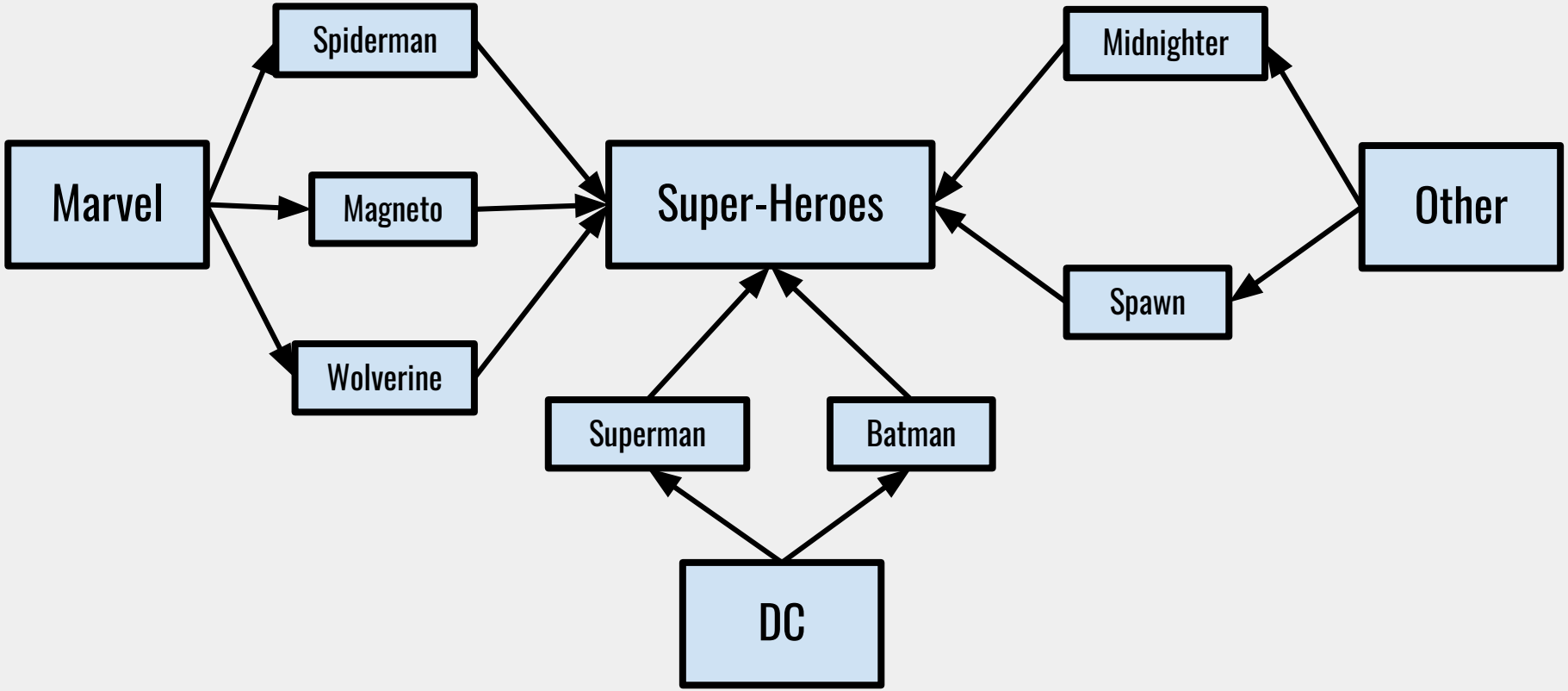
Databases and DBMSs

- Four main structure *types* of databases
  - Relational databases
  - Hierarchical databases
  - Network databases
  - Object-oriented databases

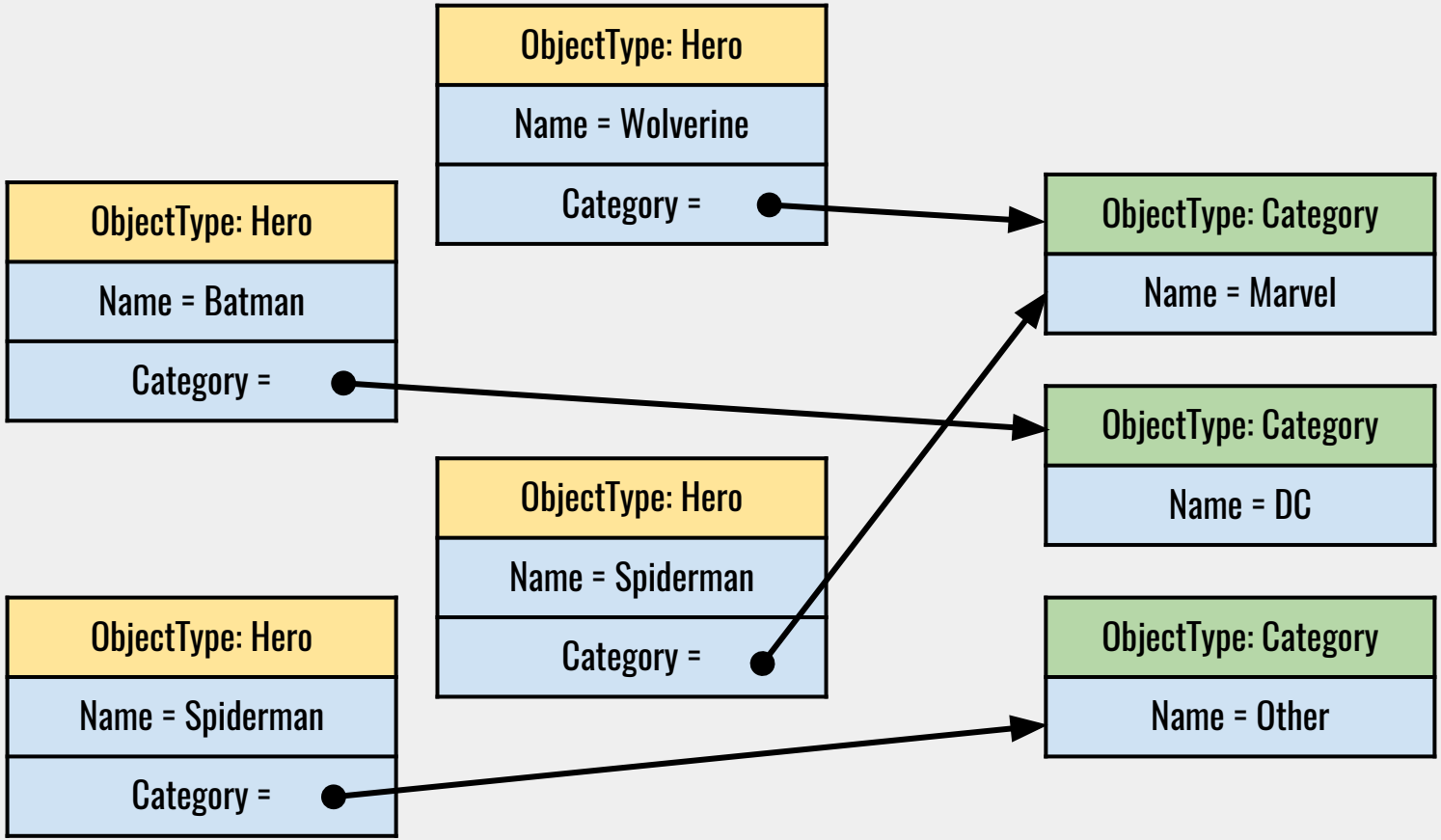
Hierarchical Model



Network Model



Object-oriented Model



Relational Model

Name
Batman
Superman
Spiderman
Spawn
Midnighter
Magneto
Wolverine

Name
Marvel
DC
Other

Relational Model

Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

Name	CatID
Marvel	1
DC	2
Other	3

Relational Model

Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

Name	CatID
Marvel	1
DC	2
Other	3

Databases and DBMSs

- Four main structure *types* of DBMSs
  - Relational databases
  - Hierarchical databases
  - Network databases
  - Object-oriented databases

Databases and DBMSs

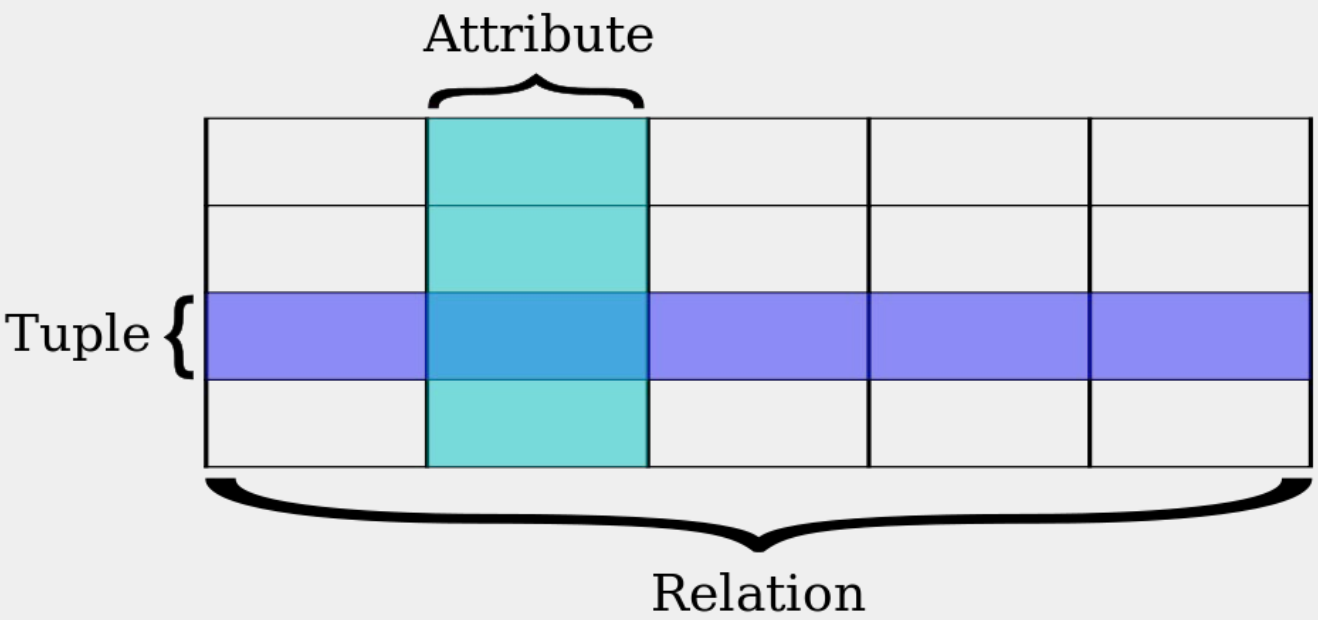
- Four main structure *types* of DBMSs
  - **Relational databases**
  - Hierarchical databases
  - Network databases
  - Object-oriented databases

Mostly talking about these in cs250

Databases and DBMSs

- A relational database is made up of... ***relations***
- In simple terms, A ***relation*** is a tabular representation of an entity
- ***Relations*** have ***attributes*** and ***tuples***
  - An ***attribute*** of a relation is a particular type of information that is stored about the entity the relation models (similar to a ***column*** in a table)
  - A ***tuples*** is a specific instance of the entity that the ***relation*** represents (similar to a ***row*** in a table)

Databases and DBMSs



Databases and DBMSs

- This is an over-simplification, but helpful for this class since we are not covering relational databases in detail:
  - **Relation → Table**
  - **Tuple → Row**
  - **Attribute → Column**

Databases and DBMSs

- Important to differentiate between **Schema** and **Data**
  - **Schema:** The specification of the structure and design of a particular database
  - **Data:** The entries stored in the database

Relational Model

This is the **Schema**

Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

Name	CatID
Marvel	1
DC	2
Other	3

Relational Model

This is the **Data**

Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

Name	CatID
Marvel	1
DC	2
Other	3

Databases and DBMSs

Using the previous example...

This is a **Relation (Table)**

Superhero		
Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

This is an **Attribute (Column)**

Superhero		
Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

This is an **Attribute Title**

Superhero		
Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

This is a **Tuple (Row)**

Superhero		
Name	HeroID	CatID
Batman	1	2
Superman	2	2
Spiderman	3	1
Spawn	4	3
Midnighter	5	3
Magneto	6	1
Wolverine	7	1

Databases and DBMSs

- It is the task of a database administrator to model some real-world information and interactions with the relational model
- Typically, each real-world ***entity*** has it's own relation
- We can make connections between relations with **foreign keys**

Databases and DBMSs

How to model the following real-world information with a relational model?

Christian Bale played ***Batman*** in ***Batman Begins*** (directed by ***Chris Nolan)***
Henry Cavill played ***Superman*** in ***Batman v Superman*** (directed by ***Zack Snyder***)
Tim Holland played ***Spider-man*** in the ***Captain America: Civil War*** (directed by ***Andy Russo***)
Michael Jai White played ***Spawn*** in ***Spawn*** (directed by ***Mark A.Z. Dippe***)
Ian McKellen played ***Magneto*** in ***X-Men*** (directed by ***Bryan Singer***)
Hugh Jackman played ***Wolverine*** in ***X-Men*** (also directed by ***Bryan Singer***)
Nobody played the super-hero ***Midnighter*** in any movies

Databases and DBMSs

- In this set of info, there are three types of entities
  - **People**
  - **Characters** (could be subdivided into actors and directors)
  - **Movies**
- There are many ways this could be modeled...

(A)

Character
Name
Batman
Superman
Spiderman
Spawn
Midnighter
Magneto
Wolverine

Actor
Name
Chris Bale
Henry Cavill
Tim Holland
Michael Jai White
Ian McKellen
Hugh Jackman

Director
Name
Joss Whedon
Chris Nolan
Bryan Singer
Zack Snyder
Andy Russo
Mark A.Z. Dippe

Movie
Title
Cap Amer: Civil War
Batman Begins
X-Men
X-Men 2
Batman v Superman
Spawn

(B)

Character
Name
Batman
Superman
Spiderman
Spawn
Midnighter
Magneto
Wolverine

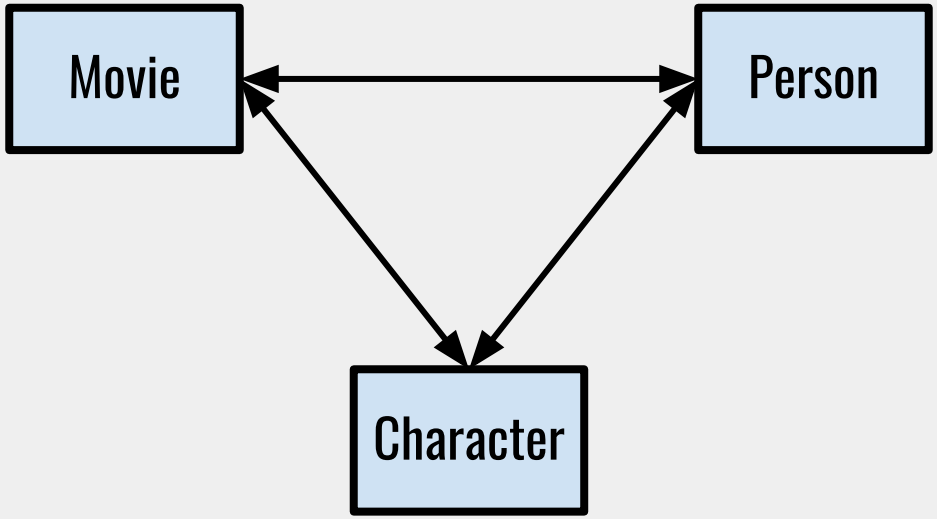
Person
Name
Chris Bale
Henry Cavill
Tim Holland
Michael Jai White
Ian McKellen
Hugh Jackman
Joss Whedon
Chris Nolan
Bryan Singer
Zack Snyder
Andy Russo
Mark A.Z. Dippe

Movie
Title
Cap Amer: Civil War
Batman Begins
X-Men
X-Men 2
Batman v Superman
Spawn

Databases and DBMSs

- We will stick with (B)
  - Separating actors and directors adds extra complexity
- We also need to model the following **relationships**
  - **Character** <-> **Person**
  - **Person** <-> **Movie** (actor)
  - **Person** <-> **Movie** (director)
  - **Movie** <-> **Character**

Databases and DBMSs



Character <-> Person

- How do we represent the relationship between **Person** and **Character?**
- It is easy if the mapping is 1-to-1 ....

Character	
Name	CID
Batman	1
Superman	2
Spiderman	3
Spawn	4
Midnighter	5
Magneto	6
Wolverine	7

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Character <-> Person

Character	
Name	CID
Batman	1
Superman	2
Spiderman	3
Spawn	4
Midnighter	5
Magneto	6
Wolverine	7

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Character <-> Person

Character		
Name	CID	PID
Batman	1	7
Superman	2	10
Spiderman	3	4
Spawn	4	3
Midnighter	5	?
Magneto	6	5
Wolverine	7	6

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Character <-> Person

Character		
Name	CID	PID
Batman	1	7
Superman	2	10
Spiderman	3	4
Spawn	4	3
Midnighter	5	?
Magneto	6	5
Wolverine	7	6

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Movie <-> Character

- How do we represent the relationship between **Movie** and **Character?**
- In this case, the mapping is not 1-to-1
  - Magneto is in **X-Men** and **X-Men 2**
  - Wolverine is in **X-Men** and **X-Men 2**
  - Batman is in **Batman Begins** and **Batman v Superman**
- We need to use an extra table to represents relationships that are 1-to-many or many-to-many

Movie <-> Character

Character		
Name	CID	PID
Batman	1	7
Superman	2	10
Spiderman	3	4
Spawn	4	3
Midnighter	5	?
Magneto	6	5
Wolverine	7	6

CharacterIn	
CID	MID

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Movie <-> Character

Character		
Name	CID	PID
Batman	1	7
Superman	2	10
Spiderman	3	4
Spawn	4	3
Midnighter	5	?
Magneto	6	5
Wolverine	7	6

CharacterIn	
CID	MID
1	2
1	5
2	5
3	1
4	6
6	3
6	4
7	3
7	4

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Movie <-> Character

Character		
Name	CID	PID
Batman	1	7
Superman	2	10
Spiderman	3	4
Spawn	4	3
Midnighter	5	?
Magneto	6	5
Wolverine	7	6

CharacterIn	
CID	MID
1	2
1	5
2	5
3	1
4	6
6	3
6	4
7	3
7	4

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Person <-> Movie (Actor)

- How do we represent the relationship between **Person** and **Movie** (actors)?
- This mapping is also not 1-to-1 (Ian McKellen, Hugh Jackman)

Person <-> Movie (Actor)

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

ActsIn	
PID	MID

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Person <-> Movie (Actor)

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

ActsIn	
PID	MID
3	6
4	1
5	3
5	4
6	3
6	4
7	2
10	5

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Person <-> Movie (Actor)

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

ActsIn	
PID	MID
3	6
4	1
5	3
5	4
6	3
6	4
7	2
10	5

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Person <-> Movie (Director)

- How do we represent the relationship between **Person** and **Movie** (directors)?
- This mapping is also not 1-to-1 (because of Bryan Singer)
- It is many-to-many

Person <-> Movie (Director)

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Directs	
PID	MID

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Person <-> Movie (Director)

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Directs	
PID	MID
11	1
8	2
9	3
9	4
12	6
2	5

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Person <-> Movie (Director)

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Directs	
PID	MID
11	1
8	2
9	3
9	4
12	6
2	5

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Person	
Name	PID
Joss Whedon	1
Zack Snyder	2
Michael Jai White	3
Tim Holland	4
Ian McKellen	5
Hugh Jackman	6
Chris Bale	7
Chris Nolan	8
Bryan Singer	9
Henry Cavill	10
Andy Russo	11
Mark A.Z. Dippe	12

Character		
Name	CID	PID
Batman	1	7
Superman	2	10
Spiderman	3	4
Spawn	4	3
Midnighter	5	?
Magneto	6	5
Wolverine	7	6

CharacterIn	
CID	MID
1	2
1	5
2	5
3	1
4	6
6	3
6	4
7	3
7	4

Directs	
PID	MID
11	1
8	2
9	3
9	4
12	6
2	5

ActsIn	
PID	MID
3	6
4	1
5	3
5	4
6	3
6	4
7	2
10	5

Movie	
Title	MID
Cap Amer: Civil War	1
Batman Begins	2
X-Men	3
X-Men 2	4
Batman v Superman	5
Spawn	6

Putting it all Together

Databases and DBMSs

- All values in a particular column in a relational database table do not need to be unique
- There may be duplicate rows in a table as well

Databases and DBMSs

- Relationships
  - **1-to-1:** Do not need extra table for relationship
  - **1-to-many:** Sometimes want extra table for relationship (but not necessary)
    - If only need to go one-way, no need
    - If want to go both ways easily, then need!
  - **many-to-many:** Always have extra relation for relationship

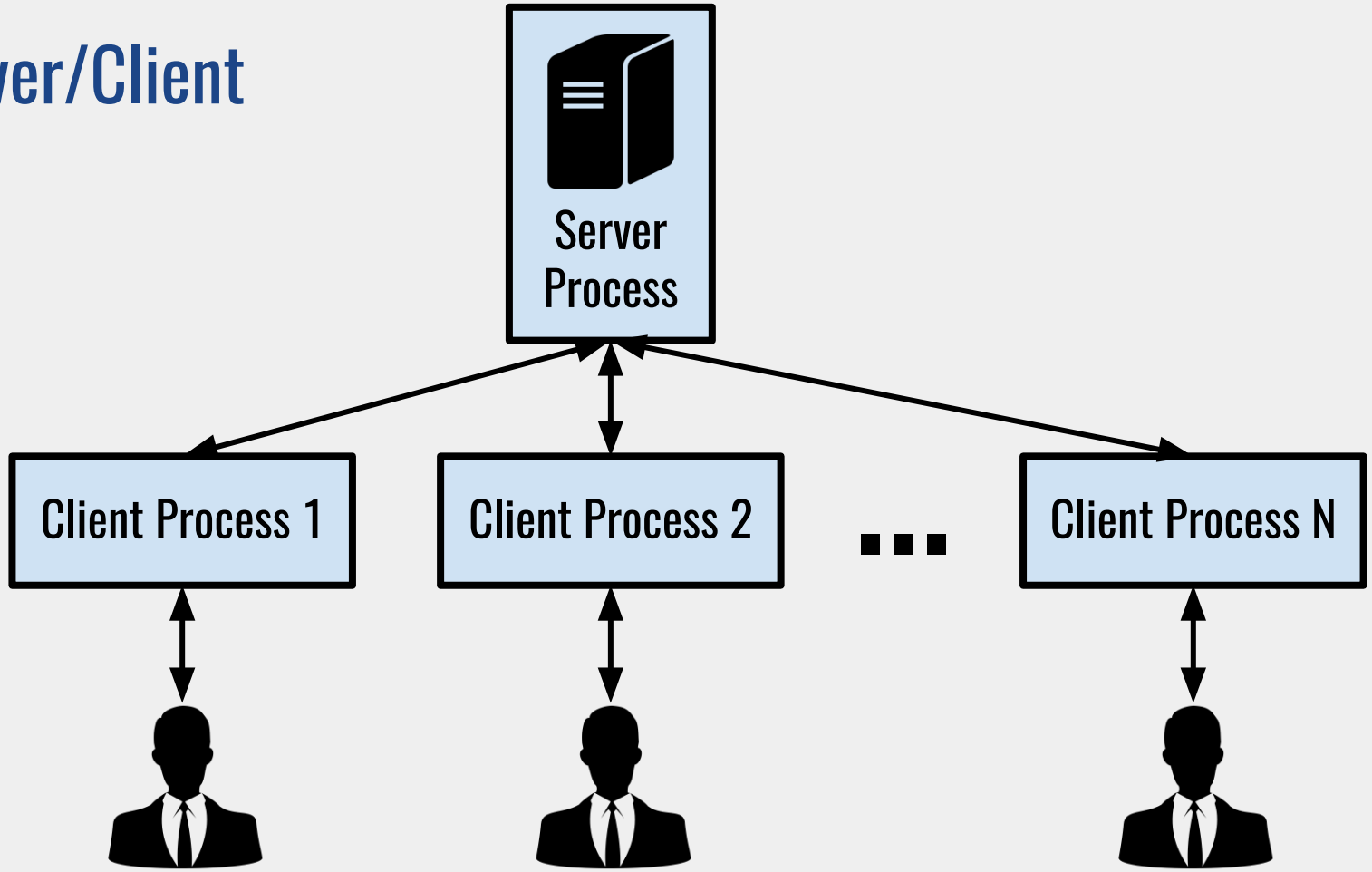
Databases and DBMSs

- Popular **Relational DBMSs**
  - MySQL, SQLite, PostgreSQL, Oracle
- Non-relational DBMSs (which we won’t discuss much in this class)
  - MongoDB, NoSQL, CouchDB

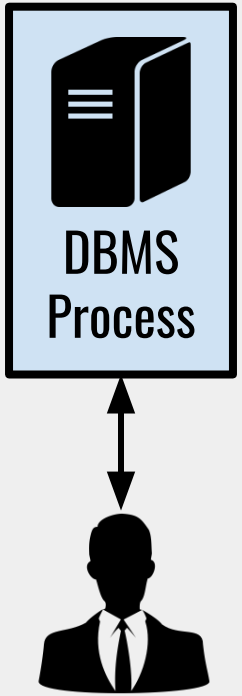
Databases and DBMSs

- There are two main types of DBMS **architectures**
  - **Server/Client** architecture
    - PostgreSQL, MySQL, Oracle
  - **Standalone** architecture
    - SQLite

Server/Client



Standalone



- In-class Exercise
  - Model Ford Motor Company's manufacturing database
  - Ford manufactures cars at multiple locations in the US
    - Flat Rock Assembly Plant 1, Michigan
    - Chicago Assembly, Illinois
    - Dearborn Truck, Michigan
    - Kansas City Assembly, Missouri
  - Ford manufactures many types of cars
    - F-150, Mustang, Focus, Explorer, Flex, …
  - Ford sells to many customers
    - Dealerships, Companies, Gov't, Individuals, …

- In-class Exercise
  - Tables: **Facility Item Purchase Customer**
  - Relationships to model
    - Facility <-> Item
    - Item <-> Purchase
    - Purchase <-> Customer

- In-class Exercise
  - Design the Schema
  - Enter a few rows into each table
  - Show the relationships between tables with arrows

Databases and DBMSs

- Most **relational DBMSs** use **SQL (Structured Query Language)** as the primary method of interacting with (adding, removing, accessing) the **DB** managed by the **DBMS**
- **SQL** is a programming language used in relational database management systems
- Used by MySQL, SQLite, PostgreSQL, Oracle, and many more
- Many different DBMSs “unified” by one common language!

Databases and DBMSs

- The SQL query language is used to:
  - Define the relational database structure
  - Load data into them
  - Remove data
  - Access the data
- We'll learn about this soon

Databases and DBMSs

- In particular, we will be learning the **SQLite** DBMS because:
  - It is easy to install
    - And is pre-installed on Macs!
  - Has a simple, easy to understand interface
  - Python has a built-in module for connecting to a sqlite database
    - Python does not have a built-in module for connecting to other commonly-used relational DBMSs such as MySQL and Postgres
  - SQLite is the most used database engine in the world (they claim)

Databases and DBMSs

- **Reading Materials**
  - en.wikipedia.org/wiki/Database
  - en.wikipedia.org/wiki/Relational_database
  - cl.cam.ac.uk/~fms27/db/tr-98-2.pdf
  - coding-geek.com/how-databases-work (very technical)