CS253: Software Development

Welcome to Lecture 12!

Daniel George

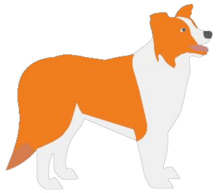
October 5, 2023

Announcements (same as Tuesday)

- Today/this week: deep dive into SQL and data
- Assignment 2 is due on Friday 10/6, midnight CST
- Daniel is here on campus this week in CNS for lectures

Group Game!

SQL (Structured Query Language) II

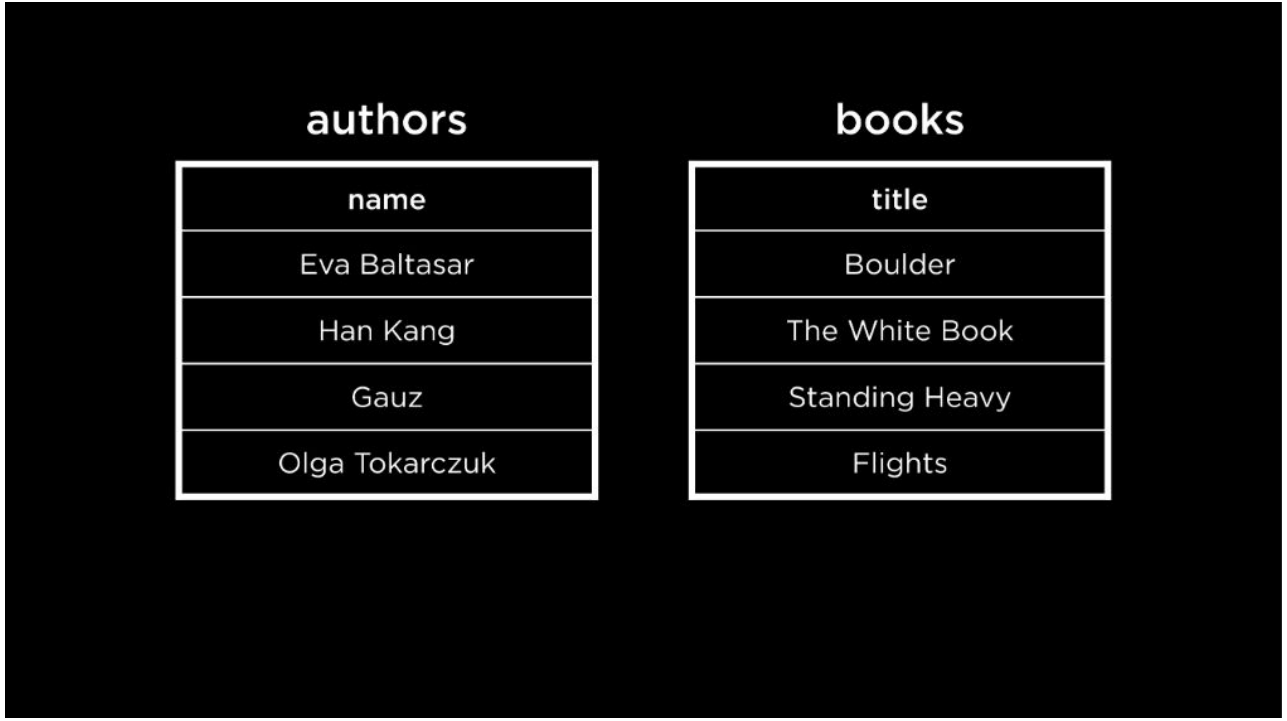


Entity Relationships

- Databases can have multiple tables
- Last class, we saw a table of books and a table of movies, but these didn’t have any relationship with each other
- We could have a database has many different tables inside it — for books, authors, publishers and so on
- In this way, we could have a **relational database** (because the tables have relationships between them). We mentioned last time that SQL was used to query these databases

Relationship Modeling

- Let’s take our “books” example. Suppose we had these tables in our database:



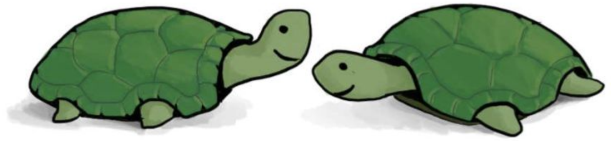
authors

name
Eva Baltasar
Han Kang
Gauz
Olga Tokarczuk

books

title
Boulder
The White Book
Standing Heavy
Flights

Think-Pair-Share Terrapins!



Just looking at these two columns, how can we tell who wrote which book? What ideas do you have?

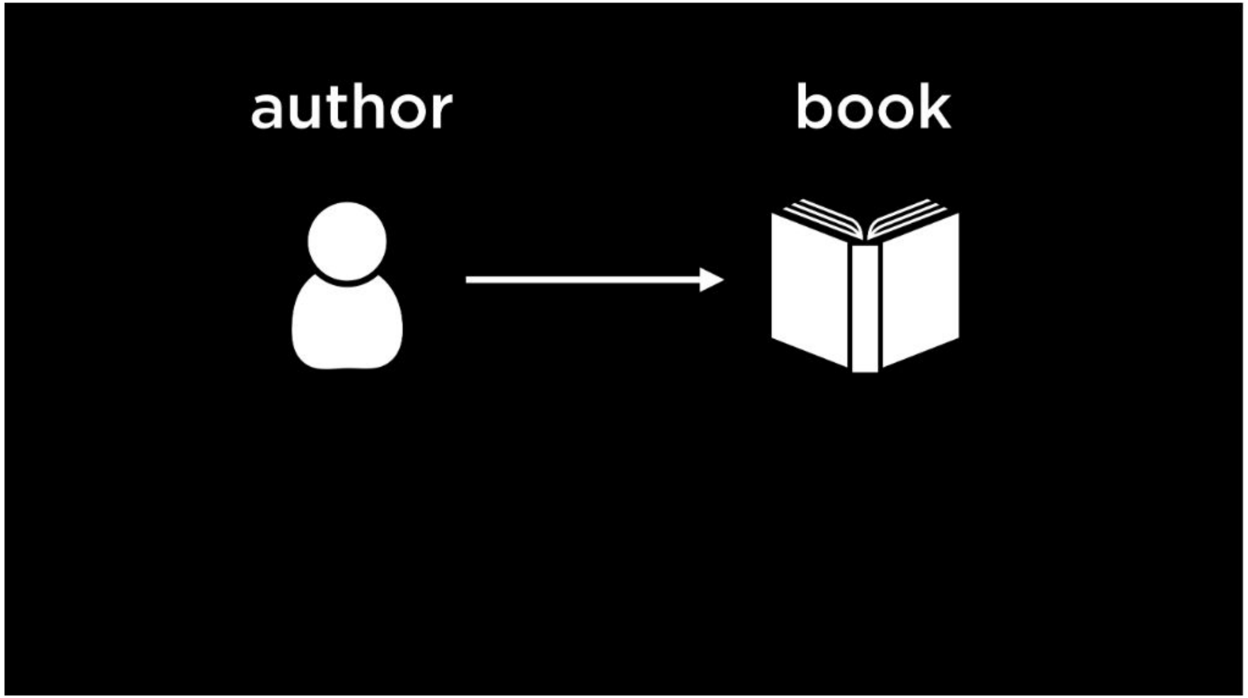
Image credit: Prof. Mary Wootters, Stanford

Two Possible Ways

- **The “Honor System”:** the first row in the authors table will always correspond to the first row in the books table. The problem with this system is that one may make a mistake (add a book but forget to add its corresponding author, or vice versa). Also, an author may have written more than one book or a book may be co-written by multiple authors.
- **Going back to a one-table approach:** This approach could result in redundancy (duplication of data) if one author writes multiple books or if a book is co-written by multiple authors. Below is a snapshot of the one-table approach with some redundant data.

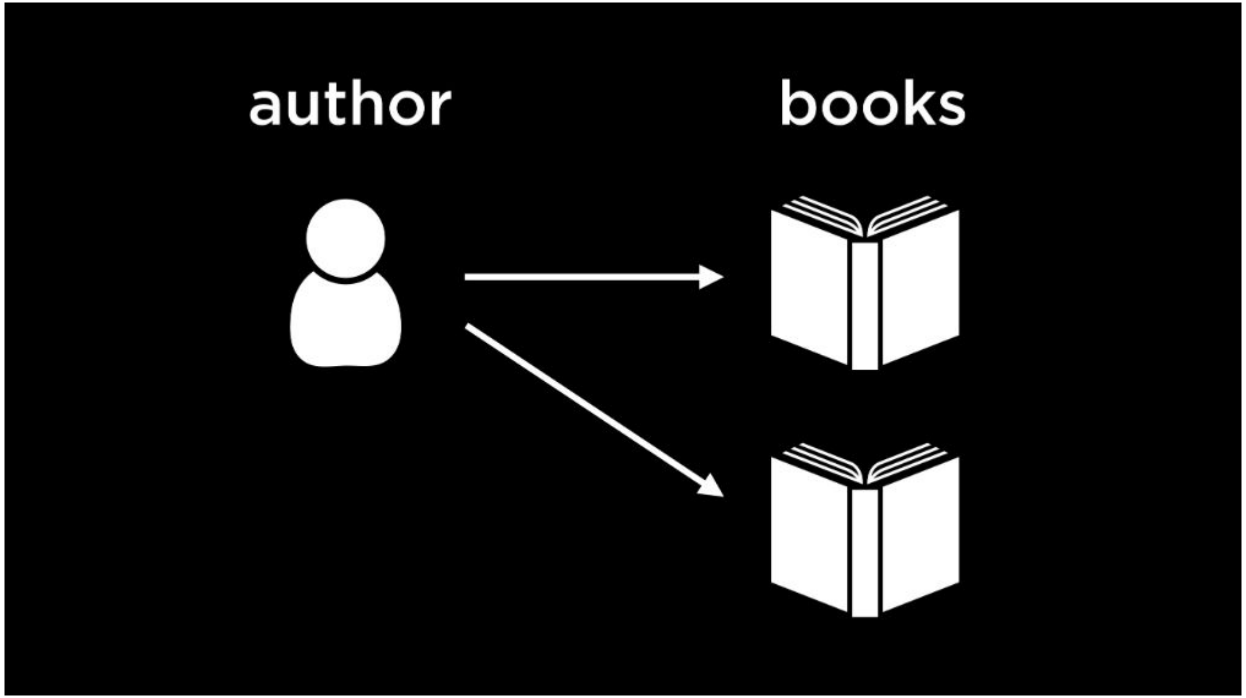
One-to-Many Relationship

- Consider this case, where each author writes only one book and each book is written by one author



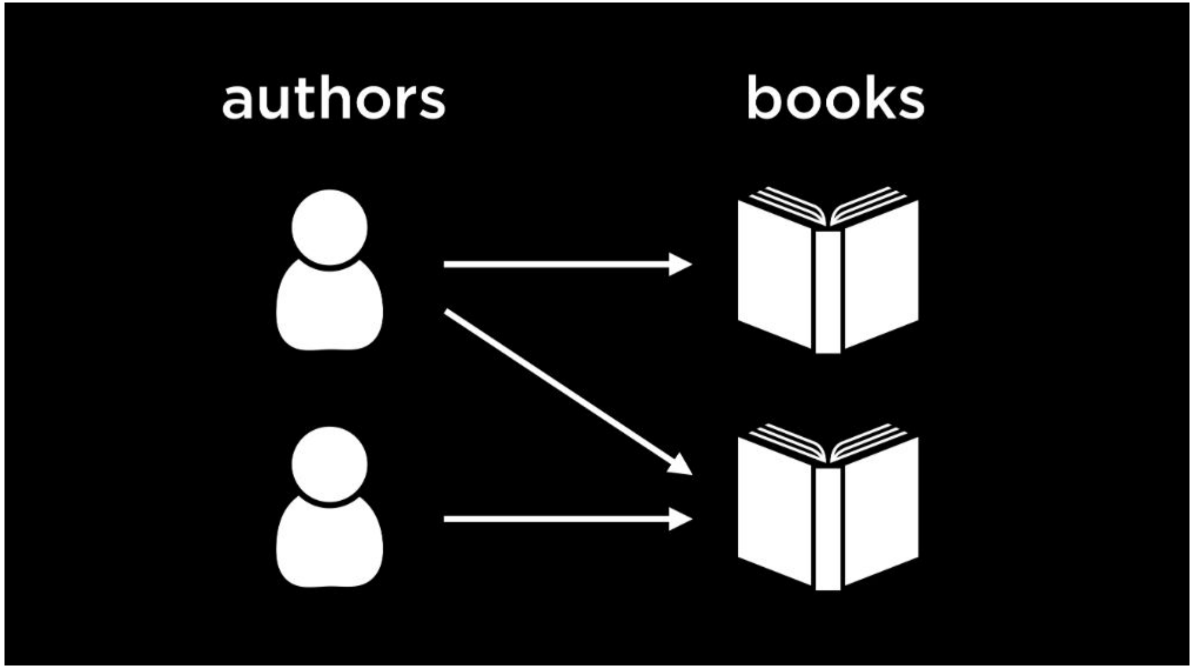
One-to-Many Relationship

- On the other hand, if an author can write multiple books…



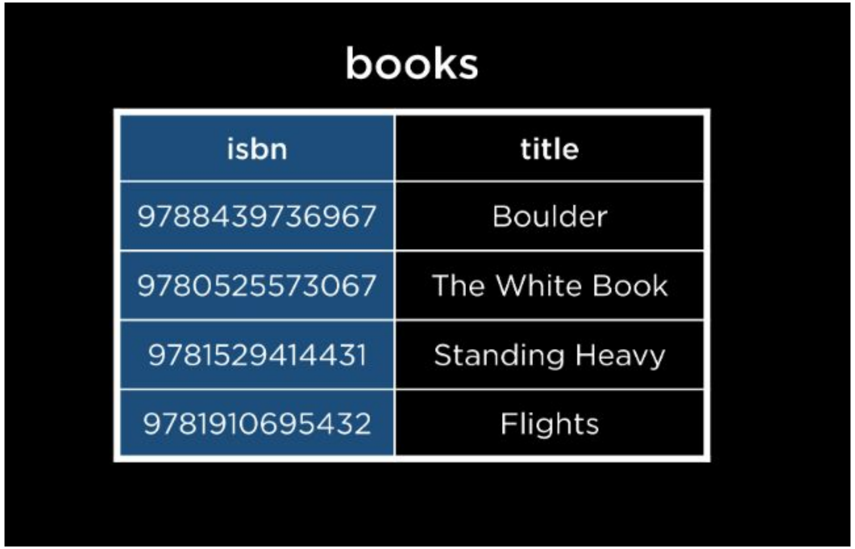
Many-to-Many Relationship

- Finally, we see a situation where not only can one author write multiple books, but books can also be co-written by multiple authors



PRIMARY KEYS

- A primary key is an identifier which is unique for every item in a table
- In the case of books, every book has a unique identifier called an ISBN

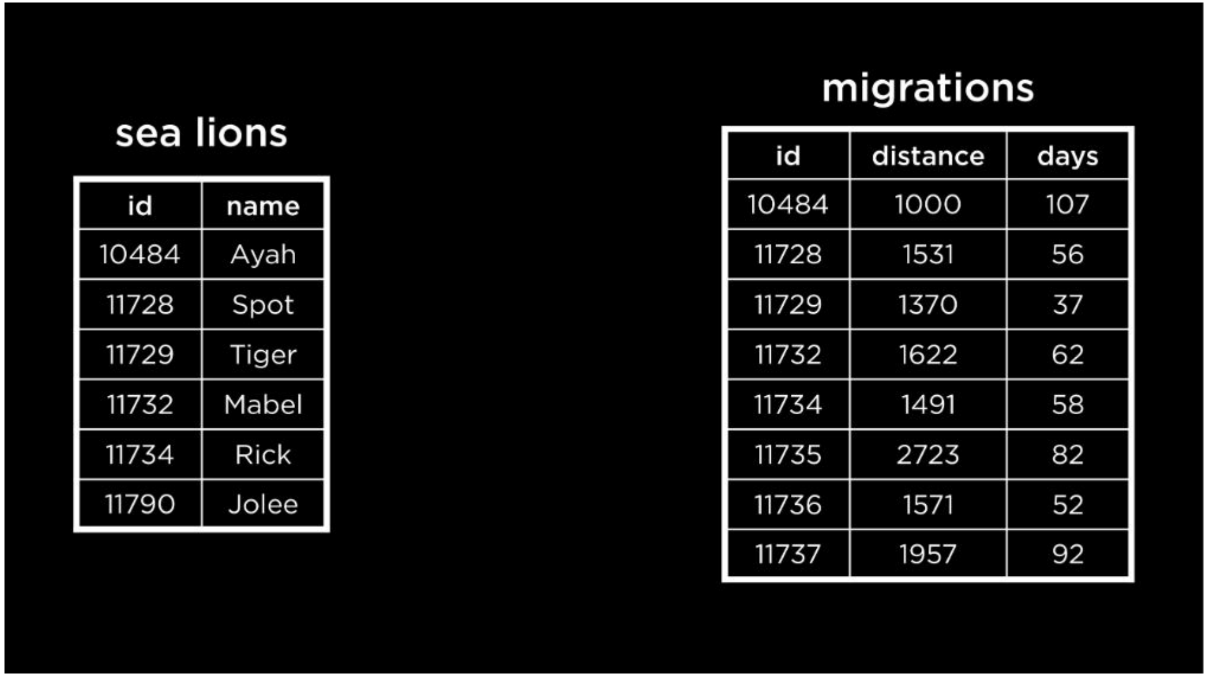


books

isbn	title
9788439736967	Boulder
9780525573067	The White Book
9781529414431	Standing Heavy
9781910695432	Flights

JOIN

- This operation allows us to combine two or more tables together. To see this, let's look at this example database



sea lions

id	name
10484	Ayah
11728	Spot
11729	Tiger
11732	Mabel
11734	Rick
11790	Jolee

migrations

id	distance	days
10484	1000	107
11728	1531	56
11729	1370	37
11732	1622	62
11734	1491	58
11735	2723	82
11736	1571	52
11737	1957	92

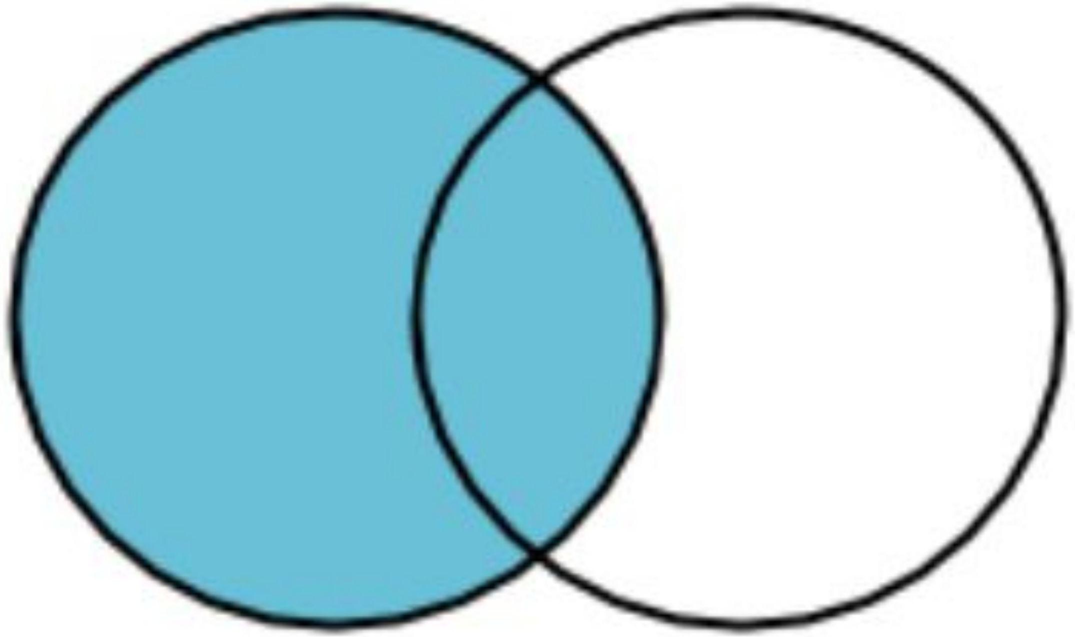
JOIN

- To find out how far the sea lion Spot travelled, or answer similar questions about each sea lion, we could use nested queries. Alternately, we could join the tables sea lions and migrations together such that each sea lion also has its corresponding information as an extension of the same row
- We can join the tables on the sea lion ID (the common factor between the two tables) to ensure that the correct rows are lined up against each other

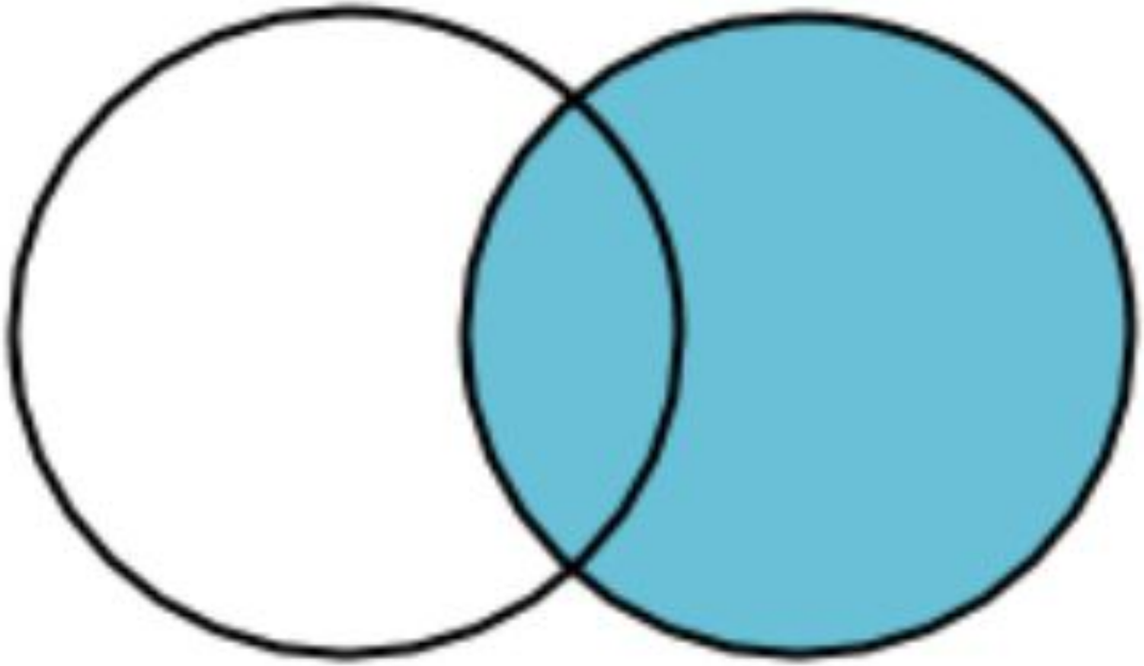
JOIN

```
SELECT *
FROM “sea_lions”
JOIN “migrations”
ON “migrations”.”id” = “sea_lions”.”id”;
```

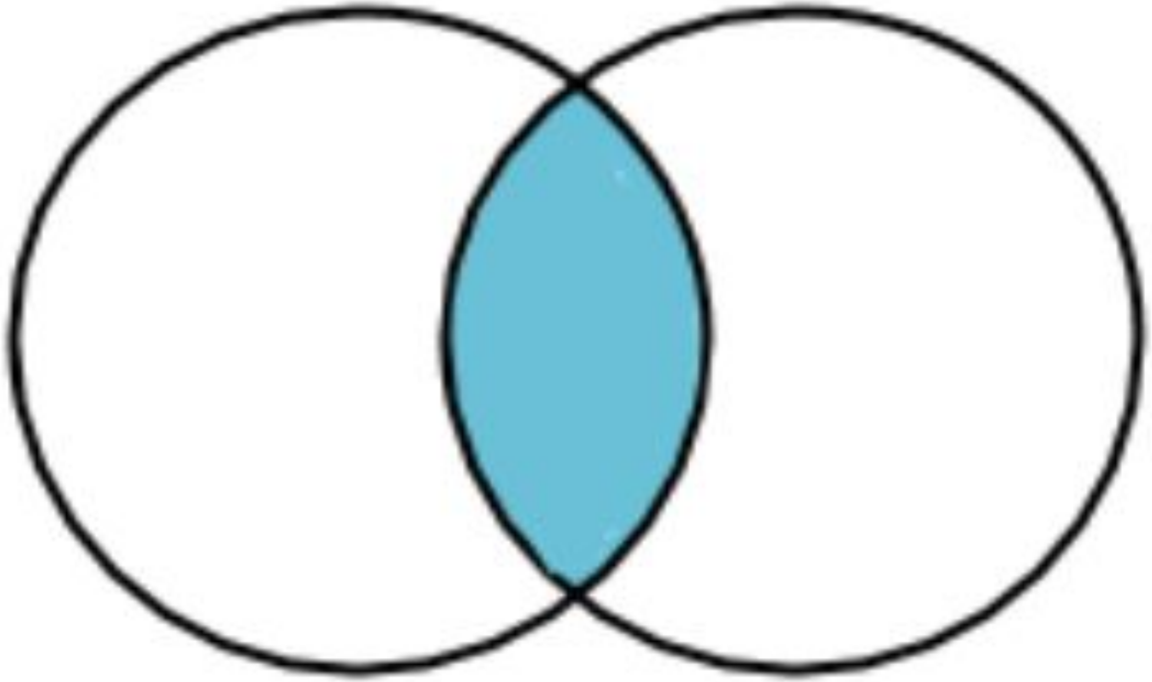
LEFT JOIN



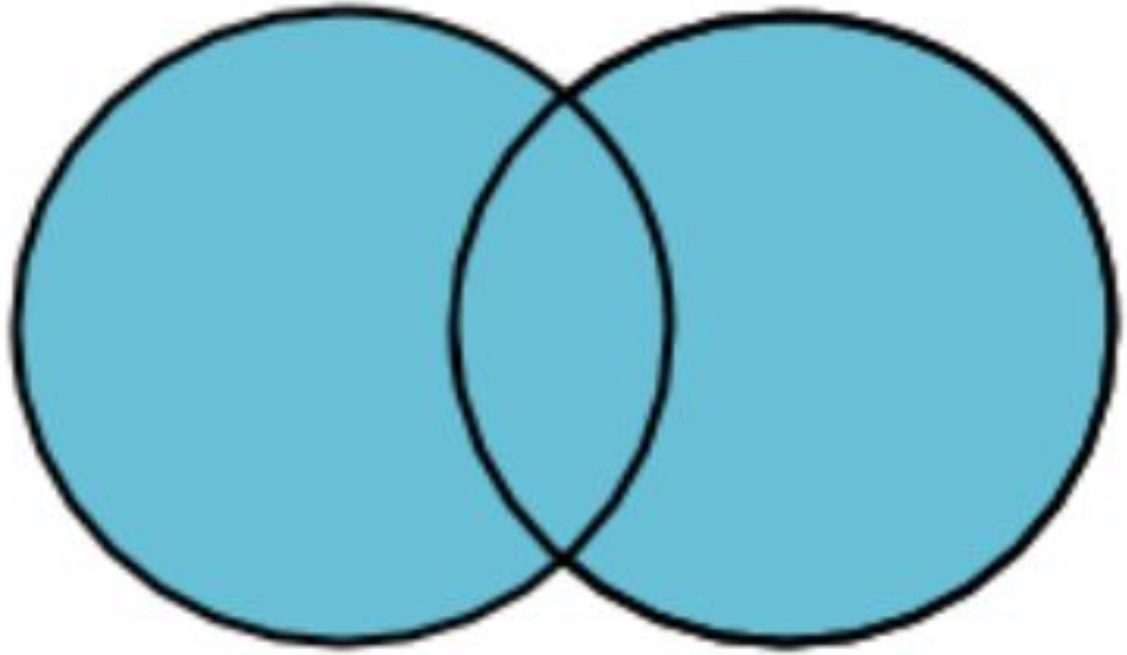
RIGHT JOIN



INNER JOIN



FULL OUTER JOIN



INDEX

```
SELECT *
FROM “table”
WHERE “year” = 2023;
```

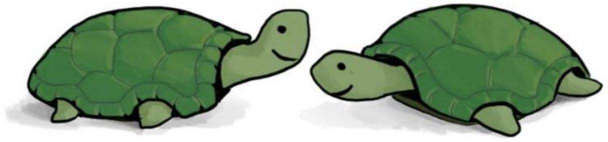
Here is a query we have seen before. What happens when we run this query? Under the hood, the table is scanned top-to-bottom, one row at a time.

CREATE INDEX

- We can create an index on a column to make queries run much faster

```
CREATE INDEX “title_index”
ON “movies” (“title”);
```

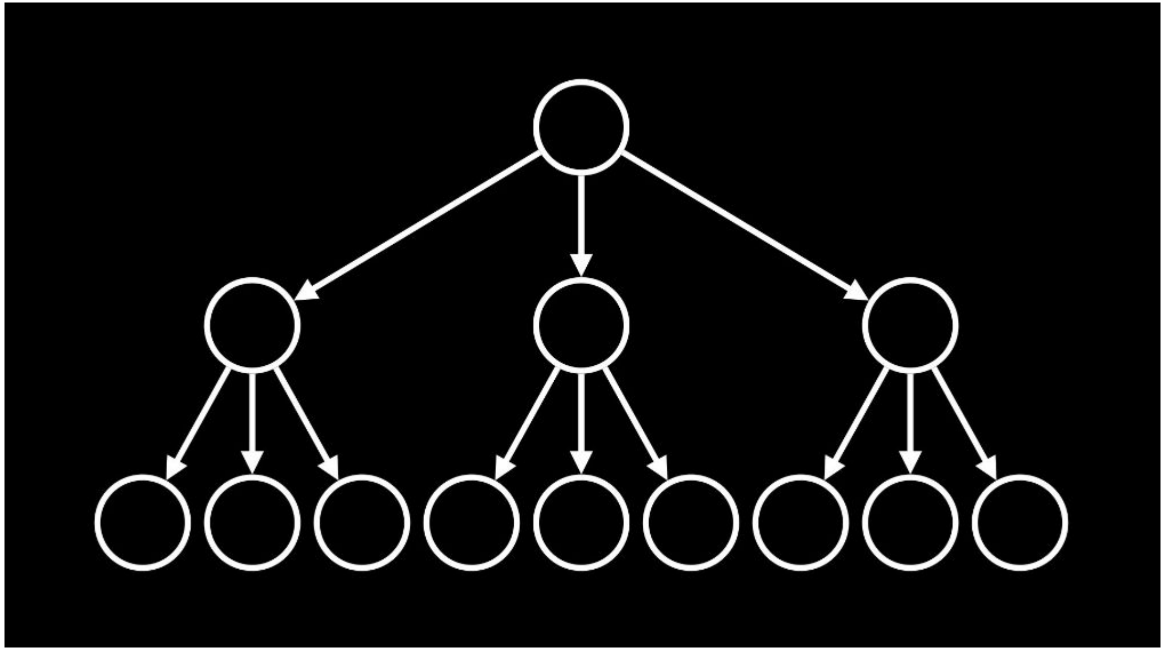
Think-Pair-Share Terrapins!



Suppose I told you that there are tradeoffs involved with indexes (it seems we can’t ever win!). What questions would you have about indexes to help understand this? If indexes make queries faster, what could that tradeoff include?

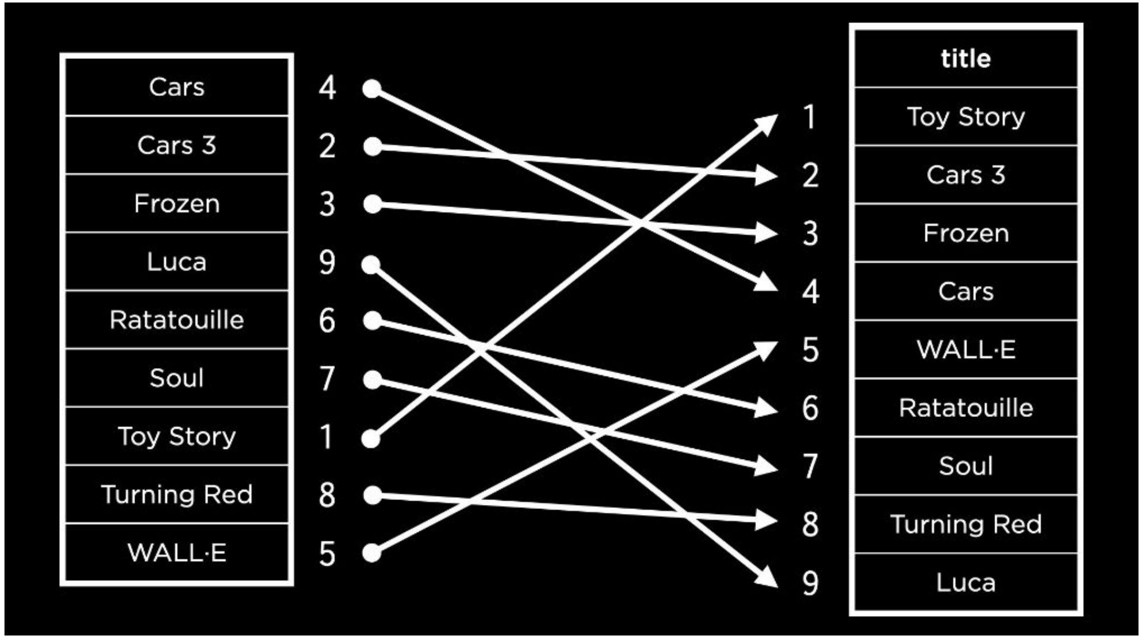
How Indexes are Implemented [extra info]

Indexes are implemented using a data structure called a **B-tree**



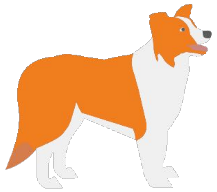
How Indexes are Implemented [extra info]

If the movie titles were sorted alphabetically, it would be a lot easier to find a particular movie by using **binary search** (for more, take CS354!)



Practice

What are your questions?



Thank you!