Retrieving data from multiple tables

Database Design

UMKC

School of Medicine

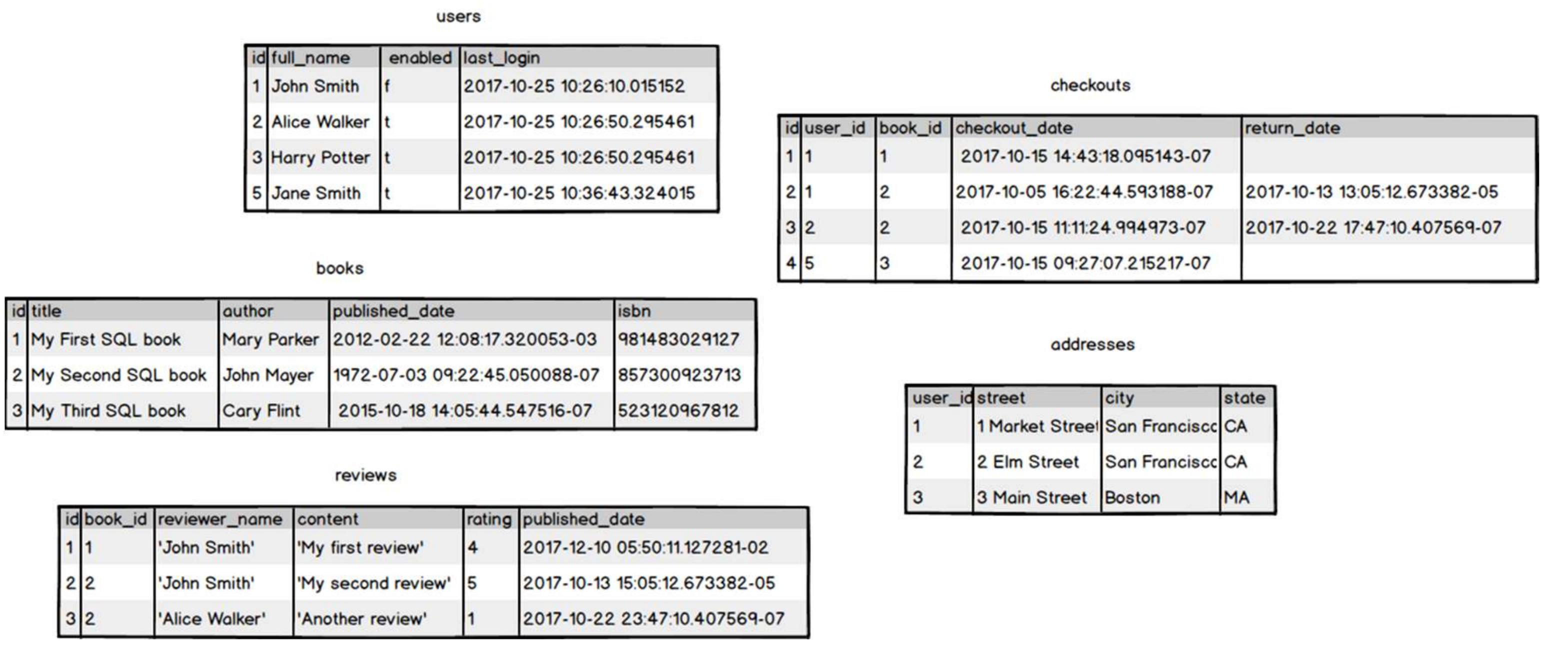
Database Design – Database Schema

- At a high level, the process of database design involves defining entities to represent different sorts of data and designing relationships between those entities.
    - An entity represents a real world object, or a set of data that we want to model within our database
    - We can often identify these as the major nouns of the system we're modeling
    - We will draw a direct correlation between an entity and a single table of data. Knowing how foreign key constraints work allow us to understand how tables relate to one another.

- What entities might we define for SQL Book application?
    - Well, we can have a users table, and we can think of a user as a specific entity within our app; a 'user' is someone who uses our app.
    - The purpose of our SQL Book app is to allow users to use books about SQL, so in this context we can think of books as an entity within our system.
    - One of the things our users can do is to checkout books, so we could have a third entity called checkouts that exists between users and books.
    - We also want users to be able to leave reviews of books they've read, so we might have another entity called reviews.
    - Finally, we want to store address information for each user. Since this address data will only be used occasionally and not for every user interaction, we decide to store it in a separate table.

Database Design – Database Schema

- We have defined the entities, we can plan tables to store the data for each entity.



users

id	full_name	enabled	last_login
1	John Smith	f	2017-10-25 10:26:10.015152
2	Alice Walker	t	2017-10-25 10:26:50.295461
3	Harry Potter	t	2017-10-25 10:26:50.295461
5	Jane Smith	t	2017-10-25 10:36:43.324015

checkouts

id	user_id	book_id	checkout_date	return_date
1	1	1	2017-10-15 14:43:18.095143-07	
2	1	2	2017-10-05 16:22:44.593188-07	2017-10-13 13:05:12.673382-05
3	2	2	2017-10-15 11:11:24.994973-07	2017-10-22 17:47:10.407569-07
4	5	3	2017-10-15 09:27:07.215217-07	

books

id	title	author	published_date	isbn
1	My First SQL book	Mary Parker	2012-02-22 12:08:17.320053-03	981483029127
2	My Second SQL book	John Mayer	1972-07-03 09:22:45.050088-07	857300923713
3	My Third SQL book	Cary Flint	2015-10-18 14:05:44.547516-07	523120967812

addresses

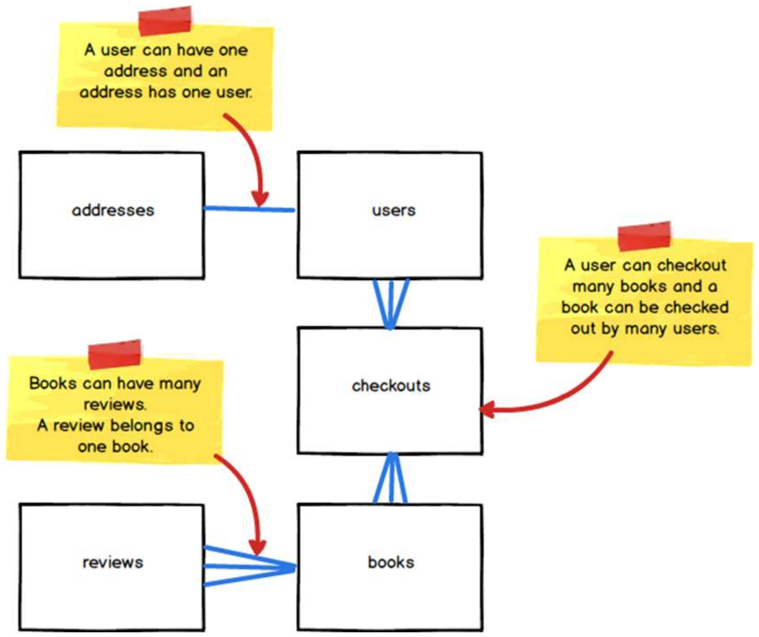
user_id	street	city	state
1	1 Market Stree	San Franciscc	CA
2	2 Elm Street	San Franciscc	CA
3	3 Main Street	Boston	MA

reviews

id	book_id	reviewer_name	content	rating	published_date
1	1	'John Smith'	'My first review'	4	2017-12-10 05:50:11.127281-02
2	2	'John Smith'	'My second review'	5	2017-10-13 15:05:12.673382-05
3	2	'Alice Walker'	'Another review'	1	2017-10-22 23:47:10.407569-07

Database Design – Database Schema

- Relationships
  - We've decided on the entities we want. There's something missing though, and that's the relationships between our entities.
  - If we look at the diagram of our five tables, the tables are all isolated and it's not obvious how these tables should relate to each other. Let's simplify our tables a bit and explicitly define some relationships between them.

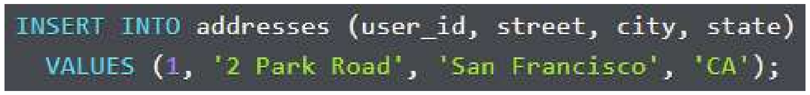

A user can have one address and an address has one user.
addresses
users
A user can checkout many books and a book can be checked out by many users.
checkouts
Books can have many reviews. A review belongs to one book.
reviews
books

Database Design – Database Schema

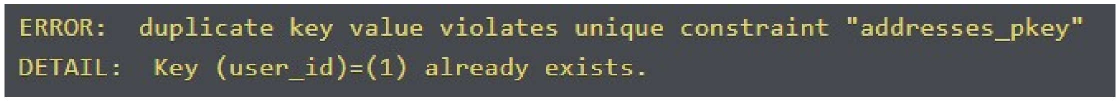
- The diagram shows an abstract representation of our various entities and also the relationships between them
    - This is simple Entity Relationship Diagram, or ERD.
    - ERD as any diagram models relationships between entities.
    - We know the tables that we need and we've also defined the relationships that should exist between those tables in our ERD.
- How do we actually implement those relationships in terms of our table schema?
    - The answer to that is to use keys.
    - Earlier we looked at an aspect of schema called constraints, and explored how constraints act on and work with the data.
- Keys are a special type of constraint used to establish relationships and uniqueness.
    - They can be used to identify a specific row in the current table, or to refer to a specific row in another table.
    - Two types of keys that fulfil these particular roles: Primary Keys, and Foreign Keys.

Database Design – Database Schema

- Referential Integrity
    - This is a concept used when discussing relational data which states that table relationships must always be consistent.
    - Different RDBMSes might enforce referential integrity rules differently, but the concept is the same.
    - The constraints we've defined for our addresses table enforce the one to one relationship we want between it and our users table, whereby a user can only have one address and an address must have one, and only one, user. This is an example of referential integrity.
- What happens if we try to add another address for a user who already has one?



```
INSERT INTO addresses (user_id, street, city, state)
  VALUES (1, '2 Park Road', 'San Francisco', 'CA');
```

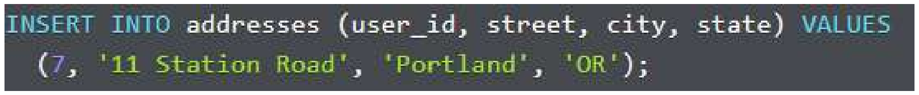


```
ERROR:  duplicate key value violates unique constraint "addresses_pkey"
DETAIL:  Key (user_id)=(1) already exists.
```

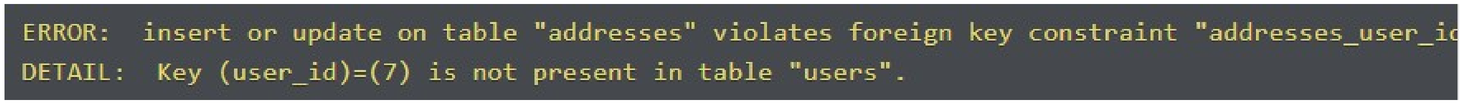
- The error above occurs because we are trying to insert a value 1 into the user_id column when such a value already exists in that column. The UNIQUE constraint on the column prevents us from doing so.

Database Design – Database Schema

- How about if we try to add an address for a user who doesn't exist?



```
INSERT INTO addresses (user_id, street, city, state) VALUES
  (7, '11 Station Road', 'Portland', 'OR');
```



```
ERROR:  insert or update on table "addresses" violates foreign key constraint "addresses_user_i
DETAIL:  Key (user_id)=(7) is not present in table "users".
```

- Here we get a different error. The FOREIGN KEY constraint on the user_id column prevents us from adding the value 7 to that column because that value is not present in the id column of the users table.

Normalization : database design technique

Normalization reduces redundancy and dependency of data. It divides larger tables to smaller tables and links them using relationships.

- Database Normal Forms
- 1NF Rules
- What is a KEY?
- What is Composite Key

- 2NF Rules
- Database - Foreign Key
- What are transitive functional dependencies?

- 3NF Rules
- Boyce-Codd Normal Form (BCNF)

Retrieving data from multiple tables

- Questions