

SQL Lesson 1

Chi Gao

2/23/2021

Introduction

Introduction

Our Scenario Today

We are running a bookstore collectively (just like the left-wing leaning bookstore on campus **The Groundwork Books!**). How do you manage your book inventory?

- What are some **common operations** that you want to do to your inventory? (CRUD)
- What are some **tool(s)** that you can think of to support your operation on the inventory?
- With these tool(s), what are some **problems** that might emerge in the foreseeable future?

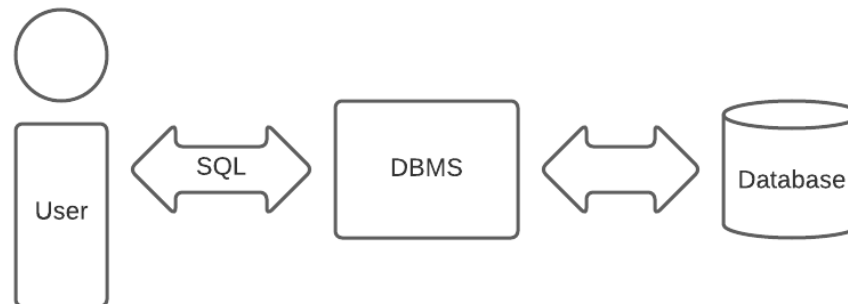
Using database and database management system, many of these concerns will be addressed.

Note: These problems and solutions are generalizable.

Introduction

Database related concepts

- **Database:** Shared collection of related data used to support the activities of a particular organization.
- **Database Management System(DBMS):** Programs that support CRUD operations and control all access to databases.
 - Provide an environment that is both **convenient** and **efficient** for users to retrieve and store information
- **SQL:** Structured Query Language - language used to talk to DBMS



Introduction

Formalized Motivation

- **Automation.** Interface linking to other programs
- **Summary Statistics.** Help you generate the report
- **Enforcement of Integrity constraints.** Rules to constraints that users enter valid information (**data type!**)
- **Multiuser system.** Control for access, data sharing
 - e.g. Read and write, read-only
- **Backup and recovery facilities.**

Data Model

Data model

A bird-eye view of our data

books

```
## # A tibble: 13 x 5
```

```
##   title          author      ISBN13    avg_rating publisher
##   <chr>          <chr>      <chr>      <dbl> <chr>
## 1 Beyond Good and Ev~ Friedrich Ni~ 97806797~    3.99 Vintage
## 2 Sanshiro        Natsume Sose~ 97819292~    3.83 Center for Japanese S~
## 3 The World of Yeste~ Stefan Zweig  97808032~    4.49 University of Nebrask~
## 4 Chronicle of the N~ Alvar Nunez ~ 97801424~    3.66 Penguin Classics
## 5 The Epic of Gilgam~ Anonymous    97801410~    3.7  Penguin Books Limited
## 6 Letter from an Unk~ Stefan Zweig  97819065~    4.23 Pushkin Press
## 7 Gorgias          Plato        97801404~    3.96 Penguin Classics
## 8 Twelfth Night    William Shak~ 97807434~    3.98 Simon Schuster
## 9 Leviathan        Thomas Hobbes 97801404~    3.71 Penguin Books
## 10 The Essays: A Sele~ Michel de Mo~ 97801404~    4.07 Penguin Classics
## 11 The Prince       Niccolo Mach~ 97801404~    3.82 Penguin Group
## 12 Second Treatise of~ John Locke    97809151~    3.77 Hackett Publishing Co~
## 13 Candide and Relate~ Voltaire     97808722~    3.81 Hackett Publishing Co~
```

customers

```
## # A tibble: 5 x 5
```

##	id	first	last	purchase_number	fav_book
##	<dbl>	<chr>	<chr>	<dbl>	<dbl>
## 1	11111	Philip	Fry	2	9780872205468
## 2	11112	Turanga	Leela	1	9780141026282
## 3	11113	Hubert	Farnsworth	3	9780140449150
## 4	11114	Amy	Wong	2	9781929280100
## 5	11115	Hermes	Conrad	1	9780142437070

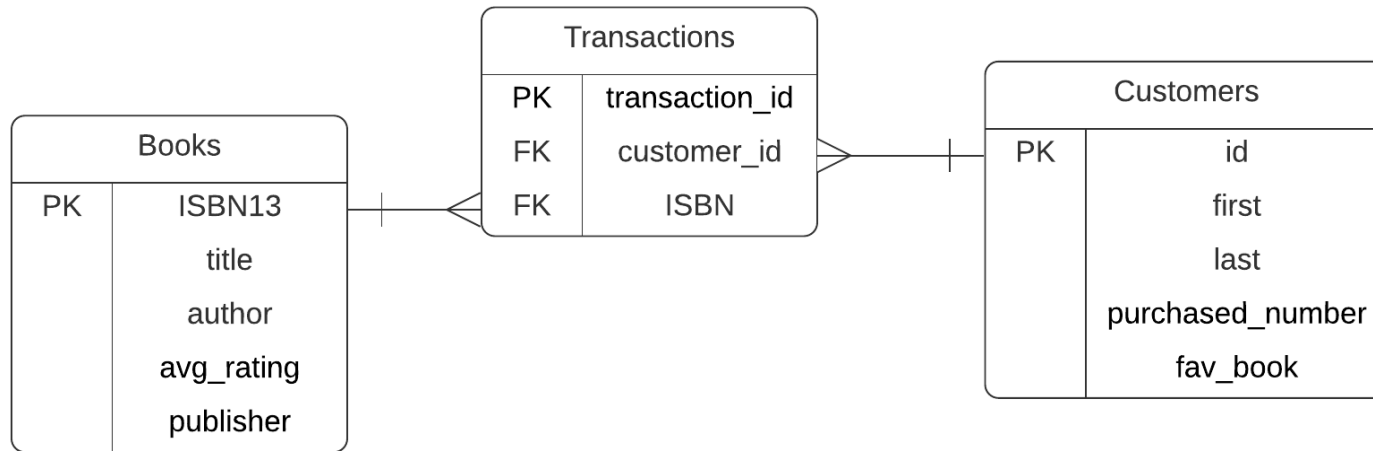
transactions

A tibble: 8 x 3

##	transaction_id	customer_id	ISBN
##	<dbl>	<dbl>	<dbl>
## 1	1	11111	9780872205468
## 2	2	11111	9780140449044
## 3	3	11112	9780141026282
## 4	4	11113	9780140449150
## 5	5	11113	9780872205468
## 6	6	11114	9781929280100
## 7	7	11114	9780743482776
## 8	8	11115	9780142437070

Data Model

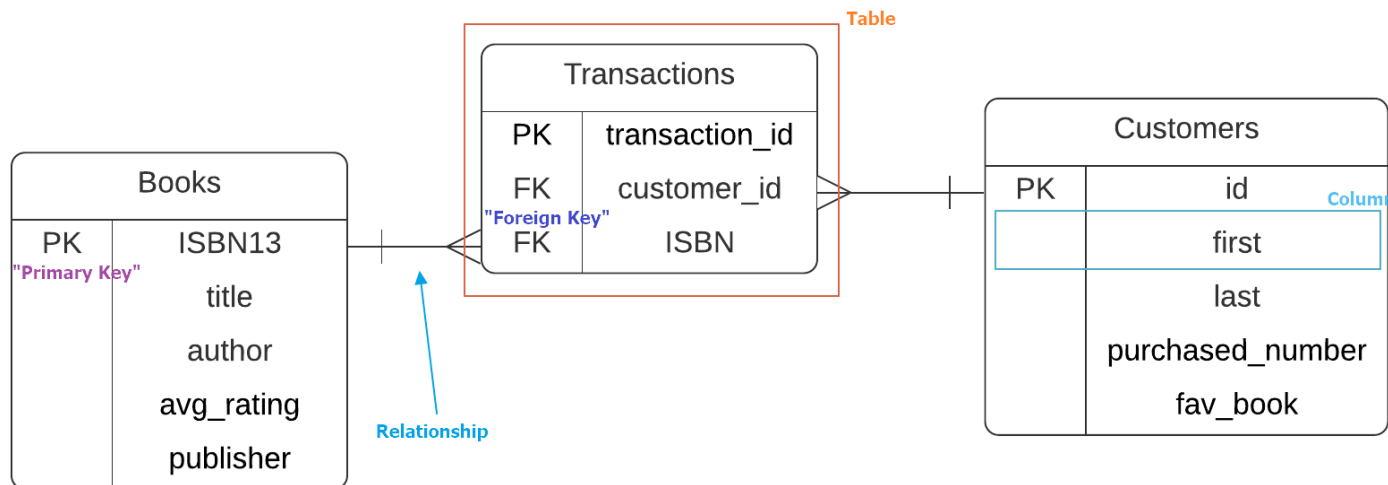
Entity-Relationship Diagram



Data Model

Anatomy of a Database

- **Primary Key:** The column that uniquely identify each row in a table
- **Foreign Key:** A column in a table that references the primary key in another table



Question: There is a missing relation. Can you spot it?

SQL

SQL

Introduction

- SQL stands for **Structured Query Language**
- Use SQL to talk to a DBMS
- Pronounced “SQL” or “Sequel”
- It is the amalgamation of
 - a data query Language (R)
Our Focus
 - a data definition language (C)
 - a data control language (Access)
 - a data manipulation language (UD)

SQL

Introduction

- SQL is formally defined, but implemented differently
 - **Incompatibility**, but very similar. You only need to learn it once.
 - Popular Extensions: MySQL, PostgreSQL,
 - We will be using **SQLite**
- Run from
 1. command line prompt
 2. Dedicated Program like DB Browser
 3. Other programming languages such as Python and R, with dedicated libraries

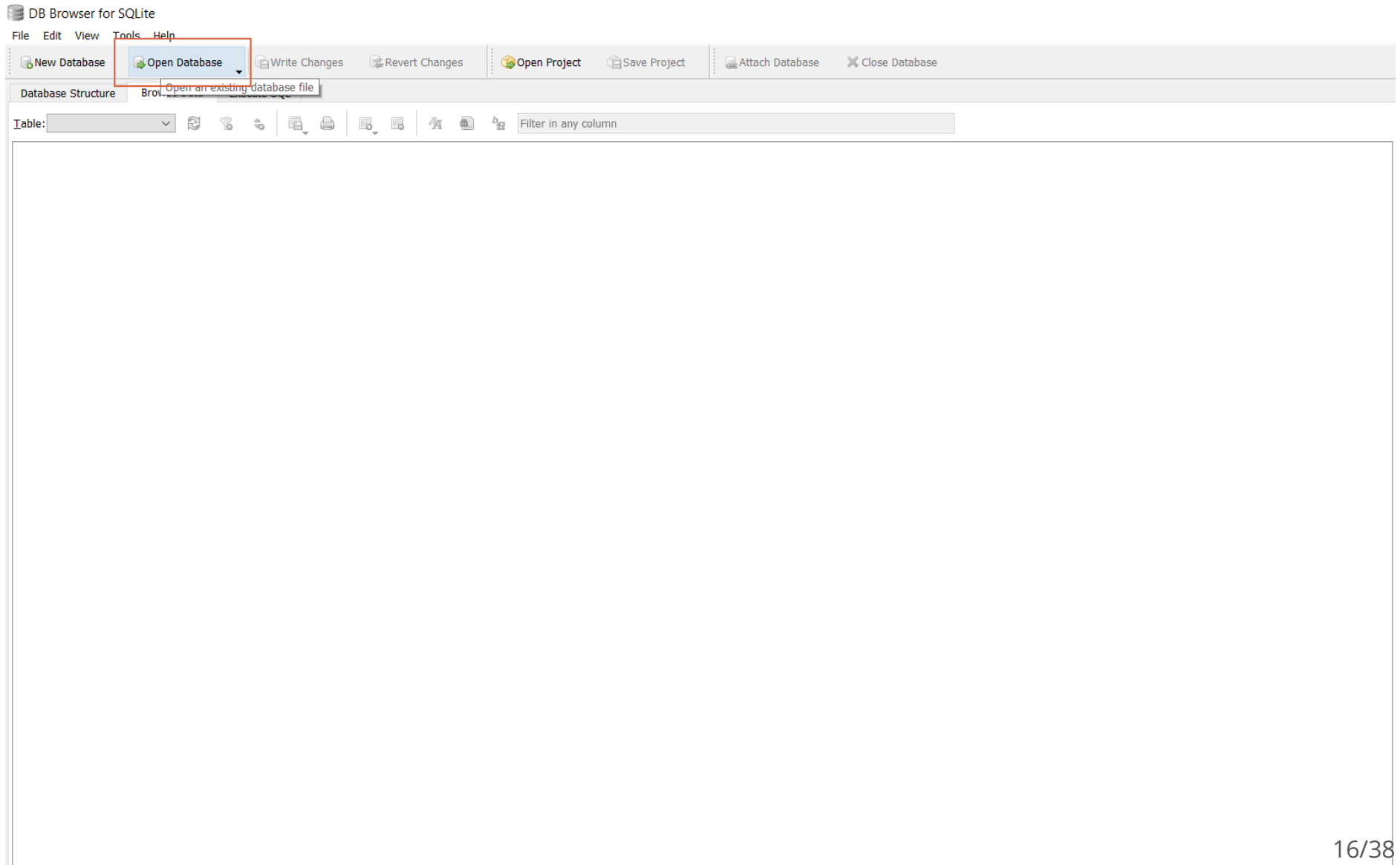
SQL

Language & Syntax

- **Case Insensitive!**
 - UPPERCASE FOR KEYWORDS
 - lowercase for anything else
- End your statement with ';'
 - empowers nice formatting

A Tour of DB Browser

Open Existing Database



A Tour of DB Browser

Database Structure

Database Structure

Browse Data

Execute SQL

Create Table

Create Index

Print

Name	Type	Schema
▼ Tables (3)		
> books		CREATE TABLE "books" ("title" TEXT, "author" TEXT, "ISBN13" TEXT, "avg_rating" REAL, "publisher" TEXT)
> customers		CREATE TABLE "customers" ("id" INTEGER, "first" TEXT, "last" TEXT, "purchase_number" INTEGER, "fav_book" TEXT)
> transactions		CREATE TABLE "transactions" ("transaction_id" INTEGER, "customer_id" INTEGER, "ISBN" TEXT)
Indices (0)		
Views (0)		
Triggers (0)		

A Tour of DB Browser

Browse Data

Database Structure Browse Data Execute SQL

Table: books Filter in any column

Switch a table

	title	author	ISBN13	avg_rating	publisher
	Filter	Filter	Filter	Filter	Filter
1	Beyond Good and Evil	Friedrich Nietzsche	9780679724650	3.99	Vintage
2	Sanshiro	Natsume Soseki	9781929280100	3.83	Center for Japanese Studies/Universi...
3	The World of Yesterday	Stefan Zweig	9780803252240	4.49	University of Nebraska Press
4	Chronicle of the Narvaez Expedition	Alvar Nunez Cabeza de Vaca	9780142437070	3.66	Penguin Classics
5	The Epic of Gilgamesh	Anonymous	9780141026282	3.7	Penguin Books Limited
6	Letter from an Unknown Woman and ...	Stefan Zweig	9781906548933	4.23	Pushkin Press
7	Gorgias	Plato	9780140449044	3.96	Penguin Classics
8	Twelfth Night	William Shakespeare	9780743482776	3.98	Simon Schuster
9	Leviathan	Thomas Hobbes	9780140431957	3.71	Penguin Books
10	The Essays: A Selection	Michel de Montaigne	9780140446029	4.07	Penguin Classics
11	The Prince	Niccolo Machiavelli	9780140449150	3.82	Penguin Group
12	Second Treatise of Government	John Locke	9780915144860	3.77	Hackett Publishing Company ...
13	Candide and Related Texts	Voltaire	9780872205468	3.81	Hackett Publishing Company, Inc.

1 - 13 of 13

Go to: 1

A Tour of DB Browser

Execute SQL

The screenshot shows the 'Execute SQL' tab in a database browser. The 'Code Space' contains a SQL query. A purple box highlights the 'Execute Code (ctrl + enter)' button in the toolbar. Below the code space, the 'Output Table' displays the results of the query. The table has three columns: 'first', 'last', and 'title'. It contains five rows of data. Below the table, a status bar indicates that the execution was successful and returned 5 rows in 7ms.

Database Structure Browse Data Execute SQL

Execute Code (ctrl + enter)

```
1 SELECT Customers.first, Customers.last, Books.title
2 FROM Customers JOIN books
3 on Customers.fav_book == Books.ISBN13;
```

	first	last	title
1	Philip	Fry	Candide and Related Texts
2	Turanga	Leela	The Epic of Gilgamesh
3	Hubert	Farnsworth	The Prince
4	Amy	Wong	Sanshiro
5	Hermes	Conrad	Chronicle of the Narvaez Expedition

Execution finished without errors.
Result: 5 rows returned in 7ms
At line 1:
SELECT Customers.first, Customers.last, Books.title
FROM Customers JOIN books

Non-relational Operations

Overview

- If you have taken the R module, you know them all already

**WHEN YOU REALIZED
YOUR R SKILL HELPS YOUR SQL**



Non-relational Operations

Overview

- Select columns: `SELECT column(s) FROM table`
- Select unique rows: `SELECT distinct rows from your table`
- Sort by a column: `ORDER BY a column`
- Filter rows by a condition: `WHERE [criteria]`
- Calculate new values on the fly
- Calculate summary statistics with functions
- Aggregate over groups: `GROUP BY`

SELECT

-- syntax

```
SELECT [column(s)] FROM table;
```

Select 'title' and 'author' column from table 'books'

```
SELECT title, author FROM books;
```

In R:

```
books %>% select(title, author)
```

SELECT

SELECT *

- Select all columns in a table
- Note '*' is called wild card

-- syntax

```
SELECT * FROM [table]
```

Select all columns in table books

```
SELECT * FROM books;
```

```
books %>% select_all()
```


SELECT

SELECT DISTINCT

Select distinct rows from selected column(s) in a table

```
SELECT DISTINCT [column(s)] FROM [table];
```

Question: How many books are sold (not how many total copies of books)?

```
SELECT DISTINCT ISBN FROM transactions;
```

In R:

```
transactions %>% select(ISBN) %>% unique()
```

SELECT

ORDER BY

- Select out columns from table, ordered by column(s), in ascending order (or descending order)

```
SELECT [column(s)] FROM [table] ORDER BY [column(s)] [DESC];
```

Question: Explain this statement in plain English

```
SELECT title, author FROM books ORDER BY author DESC;
```

```
books %>% select(title, author) %>% arrange(author) # ascending order
```

```
books %>% select(title, author) %>% arrange(desc(author)) # descending order
```

Question

Using SELECT and ORDER BY, find out who purchased the most books.

Hint: Sort the customers table by what column? In which order?

Boolean Algebra

- Two values: TRUE, FALSE
- Operators: AND(&), OR(|), NOT(!)

AND

AND	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

OR

OR

TRUE

FALSE

TRUE

TRUE

TRUE

FALSE

TRUE

FALSE

NOT

NOT

TRUE

FALSE

FALSE

TRUE

Boolean Algebra

Practice

- $(1 < -1) \text{ OR } (0 == 0)$
- $(1 < -1) \text{ AND } (-1 < 2)$

WHERE

- Use WHERE clause to keep rows according to a certain criteria

-- syntax

```
SELECT [column(s)] FROM [table]  
WHERE [criteria return True];
```

Who purchased more than 1 books?

-- notice how purchase_number **is** NOT **in** the outcome

```
SELECT first, last FROM customers  
WHERE purchase_number >= 2;
```

In R:

```
customers %>% filter(purchase_number >= 2) %>% select(first, last)
```

WHERE

compound criteria, LIKE, IN

Whose First name started with letter A or L?

```
SELECT first, last FROM customers  
WHERE (first LIKE 'A%') OR (last LIKE 'L%');
```

What books did customers whose id is 11111 and 11112 purchase?

```
SELECT * FROM transactions  
WHERE customer_id IN (11111,11112);
```

Question (manual join): I also want to know who these customers and what books are. Using WHERE, how can I find out about these information?

Question: I want to check who purchased between 1 and 3 books. What is wrong with the following code?

```
SELECT first, last, purchase_number FROM customers  
WHERE (purchase_number > 1) OR (purchase_number < 3);
```

Or in R:

```
customers %>% filter(purchase_number > 1 & purchase_number < 3)
```

Calculation on the fly

'||', Renaming using AS

Column-wise operations:

```
SELECT first || ' ' || last AS full_name  
FROM customers;
```

Aggregate Function

-- syntax

```
SELECT [function(column(s))] FROM [table]
```

What is the average rating of the books?

```
SELECT avg(avg_rating) FROM books;
```

- List of aggregate functions
 - avg(X)
 - count(X)
 - min(X)
 - max(X)
 - sum(X)

Aggregate Function

Practice

Question: What is the total number of purchases in *transactions* table? What is the total number of purchases in *customers* table? Do they match? This illustrates the principle of **single source of truth**

Group By

```
-- syntax  
SELECT [function(column(s))] FROM table  
GROUP BY [column(s)];
```

Example: What is the number of purchases by each person in transaction table?

```
SELECT customer_id, count(*) FROM transactions  
GROUP BY customer_id;
```

In R:

```
transactions %>% group_by(customer_id) %>% summarize(n = n())
```

whose record did we miss?