

RESEARCH
COMPUTING CENTER
INFORMATION TECHNOLOGY SERVICES

relational

Introduction to databases and SQL

^ ...

October 6, 2026 ◆ Casey McLaughlin

Overview ◆ Questions that we will cover today

- What is a relational database?
- What is a relational database management system?
- How do I model a basic database?
- What are tables and keys?
- What are primary, surrogate, and foreign keys?
- What is SQL?
- How do I write SQL to query data in a database?
- How do I write SQL to query data from multiple tables?
- How do I write SQL to add and update data in a database?

What is a Relational Database?

- “A database based on the relational model of data, as proposed by E.F. Codd in 1970.” —Wikipedia

- “A relational database is a type of database that stores and provides access to data in tables that can be linked to each through shared columns and rows, called relations, with unique identifiers (keys) that show the different relationships between tables.” – DataBricks*

What is SQL?

- SQL = “Structured Query Language”
- The primary way for querying relational databases

Relational Database Management System

(RDMS)

RDBMS ◆ Relational Database Management System

Servers:

- $$ - Microsoft SQL Server
- $$ - Microsoft Access
- $$ - Oracle Database
- Free - MySQL and MariaDB
- Free - Postgres
- Free - SQLite
- *Many more...*

Clients:

- $$ - Microsoft SQL Server
- $$ - Microsoft Access
- $$ - JetBrains DataGrip
- Free - HeidiSQL (Windows/Linux)
- Free - Beekeeper Studio (All)
- Free - Terminal access
- *Many more...*

Let’s build a relational database!

XYZ Restaurant

“XYZ Restaurant has a need to track customers’ names, email addresses, birthdays, and favorite dishes.”

Spreadsheet vs. Relational Data Terminology

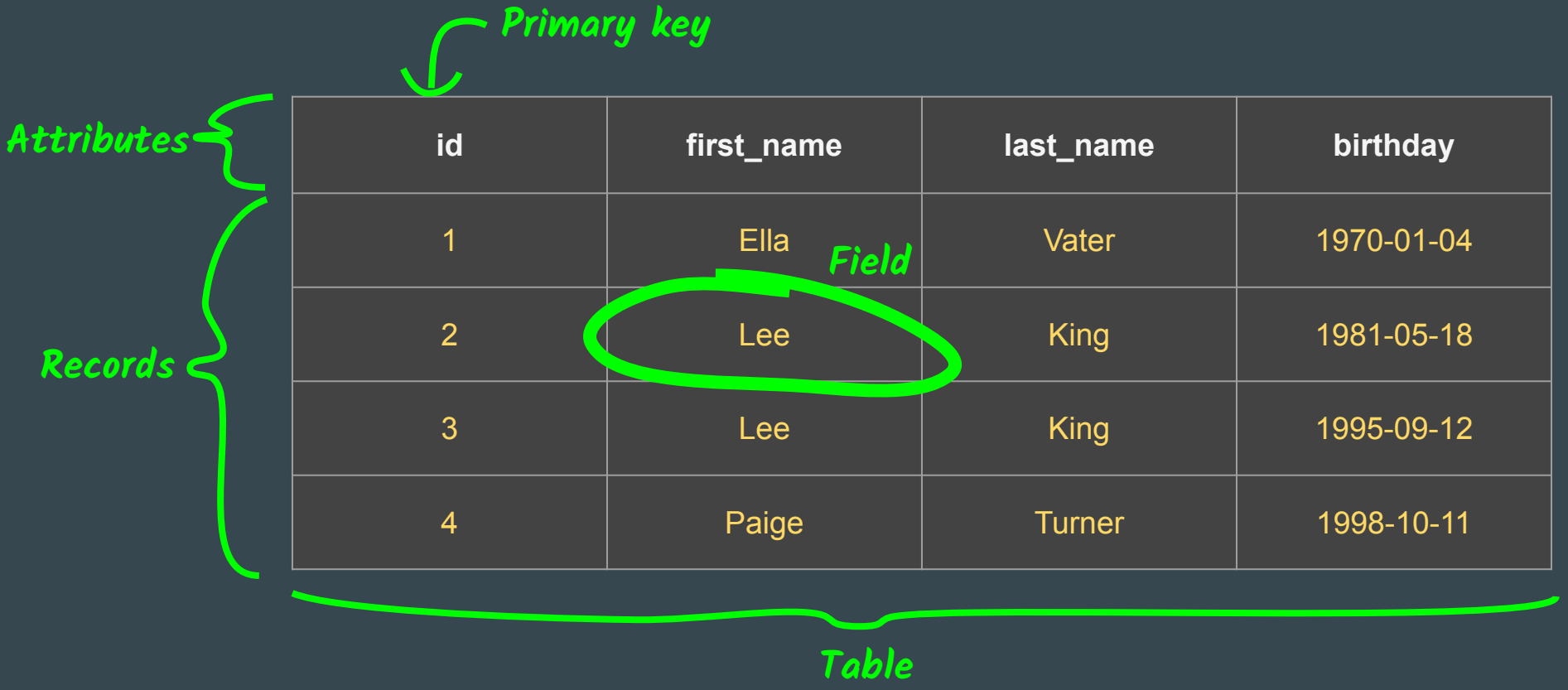
Spreadsheet	RDMS
Workbook	**Schema** or **Database**
Worksheet	**Table**
Row	**Record**, **Tuple**, or **Row**
Column	**Attribute** or **Column**
Cell	**Field**

Normalization

the process of organizing the columns (attributes) and tables of a relational database to minimize data redundancy and improve data integrity

Surrogate/synthetic keys should be arbitrary values

Relational Data Terminology Review



id	first_name	last_name	birthday
1	Ella	Vater	1970-01-04
2	Lee	King	1981-05-18
3	Lee	King	1995-09-12
4	Paige	Turner	1998-10-11

Relationships Review

Foreign key

customers

id	first_name	last_name	birthday	favorite_dish_id
1	Ella	Vater	1970-01-04	2
2	Lee	King	1981-05-18	1
3	Lee	King	1995-09-12	3
4	Paige	Turner	1998-10-11	3

dishes

id	dish_name	price
1	pizza	10.00
2	chicken	5.00
3	fish	7.50

Common RDMS Data Types (non-comprehensive list)

Type	Description
char	Fixed-length string of characters (e.g., “abc”)
varchar	Variable-length string of characters (e.g. “lorum ipsum”)
text	Long string of characters (difficult to index or query)
int	A whole number (positive or negative) (e.g., -235, 100)
float	A number with a decimal (e.g., -240.24, 93.12)
datetime	A point in time (e.g., 2025-01-05 09:03:23)
tinyint	Either a one or a zero; typically used for true/false values

XYZ Restaurant

“XYZ Restaurant has a need to track customers’ names, email addresses, birthdays, and favorite dishes.”

“XYZ Restaurant also hosts events, and would like to keep track of names, date/times, and contacts for events.”

Relationships Review

Entity Relation (ER) Diagram

schema

customers

id	first_name	last_name	favorite_dish
1	Ella	Vater	2
2	Lee	King	1
3	Lee	King	3
4	Paige	Turner	3

one

many

events

id	event_name	date	organizer_customer_id
1	Party Time!	2024-10-11	3
2	Graduation Reception	2024-11-03	2
3	Office Retreat	2024-09-01	2

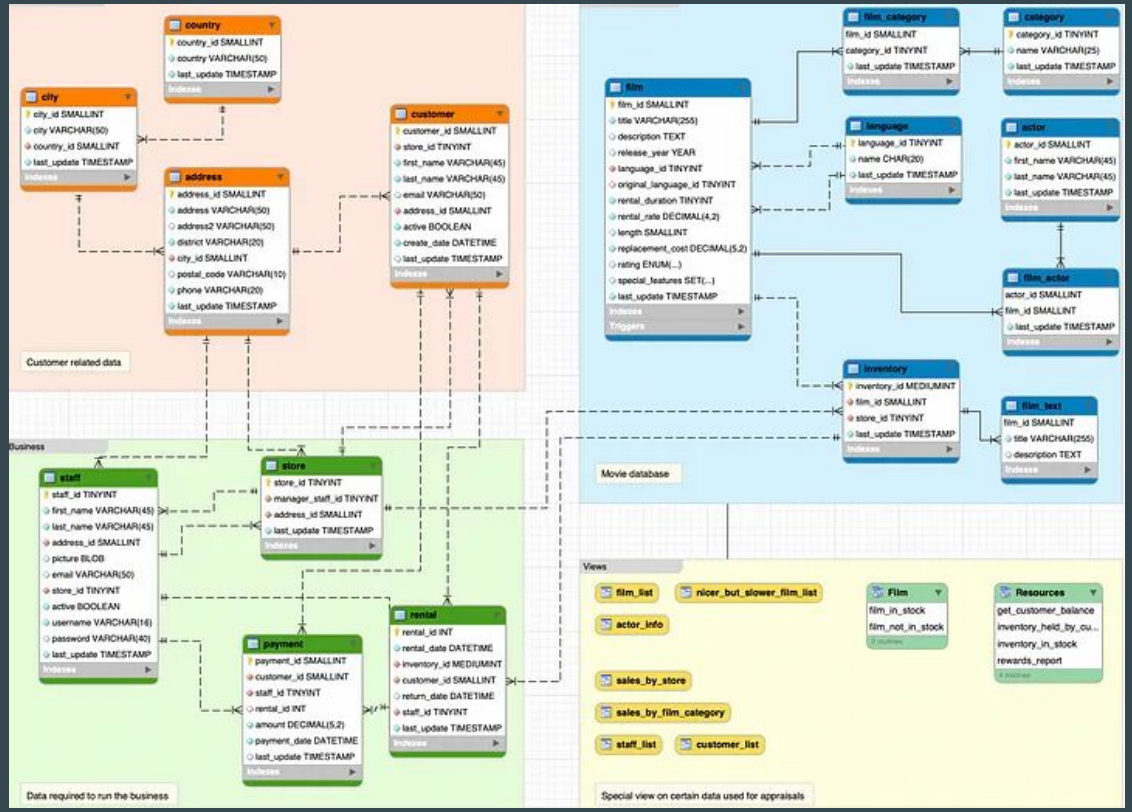
many

one

dishes

id	dish_name	price
1	pizza	10.00
2	chicken	5.00
3	fish	7.50

Example Entity Relation Diagram



Let’s write some SQL!

SQL Query Syntax Review



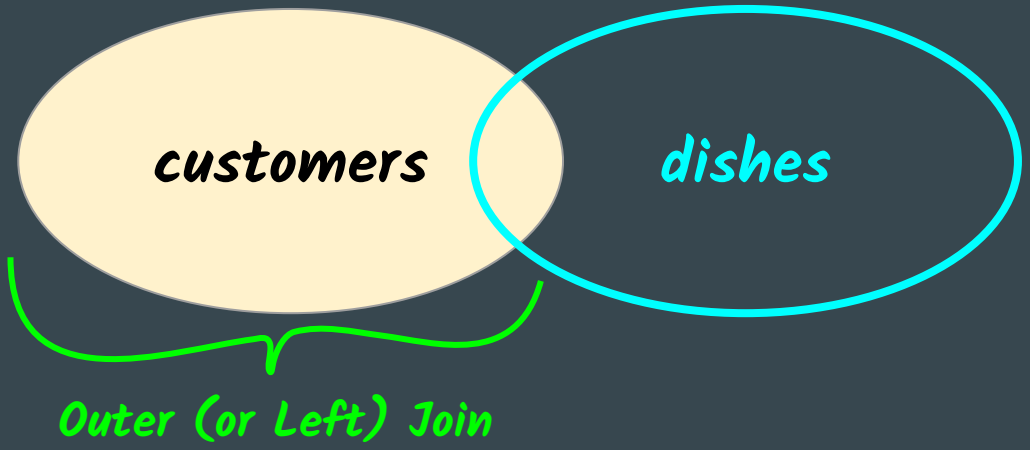

```
SELECT first_name, last_name FROM customers WHERE state = 'FL';
```

- Keyword
- Attributes
- Table
- Operator
- Literal

Outer/Left Joins

```
SELECT first_name, last_name

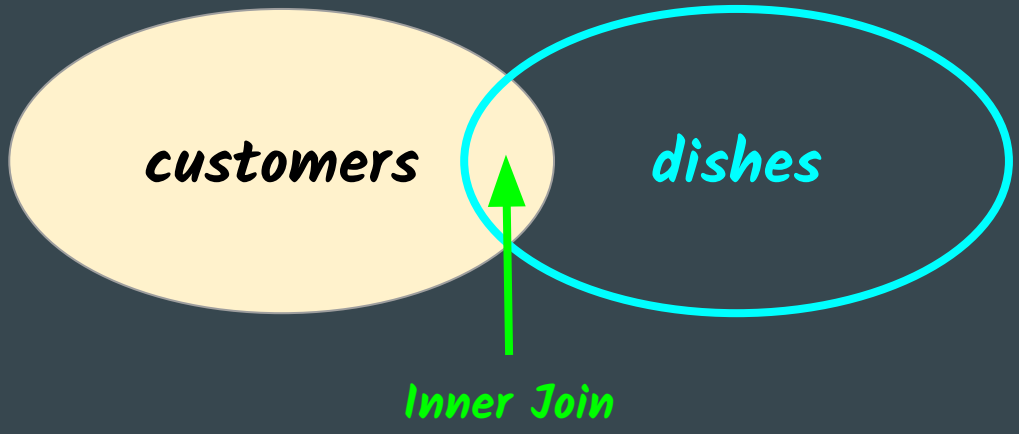
FROM customers c LEFT JOIN dishes d ON c.favorite_dish_id = d.id;
```



Inner Joins

```
SELECT first_name, last_name

FROM customers c INNER JOIN dishes d ON c.favorite_dish_id = d.id;
```



Aggregate Functions

```sql
SELECT COUNT(id) FROM customers;
```

```sql
SELECT SUM(price) AS total FROM dishes;
```

```sql
SELECT CONCAT(first_name, ' ', last_name) AS full_name
FROM customers;
```

Modifying Data: Adding a record

```
INSERT INTO customers (first_name, last_name, email)
VALUES ('Bob', 'Jones', 'bjones@example.org');
```

Modifying Data: Updating a record

```
UPDATE customers SET first_name = 'Robert'
WHERE first_name = 'Bob' AND last_name = 'Jones';
```

```
UPDATE customers SET first_name = 'Robert' WHERE id = 26;
```

Modifying Data: Deleting a record

```
DELETE FROM customers WHERE id = 26;
```

Important things we didn't cover in-full today...

- Normal forms (1st, 2nd, 3rd)
- Many-to-many and one-to-one relationships
- Recursive relationships
- Indexes
- Multi-table joins
- Advanced aggregate functions (GROUP BY, etc...)
- All data types (each RDMS differs slightly in these)
- Access control (user accounts and permissions)
- Transactions and stored procedures

How to continue your RDMS and SQL journey

- LinkedIn Learning video series
  - Programming Foundations: Databases
  - https://bit.ly/3XNMhjQ

- Notes:
  - I would stick with all lowercase field and table names and use *snake_case*
  - Whatever you choose, be consistent!

- Me:
  - Casey McLaughlin
  - cmclaughlin@fsu.edu

SURVEY: