Rotman

INTRO TO SQL

ROMA & RBAC SQL Workshop

https://tdmdal.github.io/sql-roma-rbac-2023/

January 31, 2023 Prepared by Jay / TDMDAL

Rotman School of Management
UNIVERSITY OF TORONTO

Goal (2 x 2 hrs)

- Use SQL for your case competition (???), or…

- Understand what's SQL and related concepts such as
  - Database (DB) and relational DB (RDB)
  - DB management system (DBMS) and RDBMS

- Learn the basics SQL coding
  - Simple column and row operations
  - Simple aggregations
  - simple join operations

- Know what to learn next and where to find free learning resources

What is SQL (Structured Query Language)

- A language to organize/query/manipulate data
  - What kind of language?
  - What kind of data?

- Before getting into the details, let’s have a taste of SQL
  - https://www.programiz.com/sql/online-compiler/

A Taste of SQL (1)

```
SELECT first_name, last_name, age
FROM Customers
WHERE age >= 25
ORDER BY age DESC;
```

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Table: Customers

$\rightarrow$

Try it here: https://www.programiz.com/sql/online-compiler/

A Taste of SQL (1)

```
SELECT first_name, last_name, age
FROM Customers
WHERE age >= 25
ORDER BY age DESC;
```

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Table: Customers

→

first_name	last_name	age
John	Doe	31
Betty	Doe	28
John	Reinhardt	25

Try it here: https://www.programiz.com/sql/online-compiler/

A Taste of SQL (2)

```
SELECT country, AVG(age) AS average_age
FROM Customers
GROUP BY country
HAVING average_age < 28;
```

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Table: Customers

$\rightarrow$

Try it here: https://www.programiz.com/sql/online-compiler/

A Taste of SQL (2)

```
SELECT country, AVG(age) AS average_age
FROM Customers
GROUP BY country
HAVING average_age < 28;
```

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Table: Customers

$\rightarrow$

country	average_age
UK	23.5
USA	26.5

Try it here: https://www.programiz.com/sql/online-compiler/

A Taste of SQL (3)

```
SELECT Customers.customer_id, last_name, amount
FROM Customers
INNER JOIN Orders
  ON Customers.customer_id = Orders.customer_id
ORDER BY amount;
```

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Table: Customers

order_id	item	amount	customer_id
1	Keyboard	400	4
2	Mouse	300	4
3	Monitor	12000	**3**
4	Keyboard	400	1
5	Mousepad	250	2

Table: Orders

$\longrightarrow$

Try it here: https://www.programiz.com/sql/online-compiler/

A Taste of SQL (3)

```
SELECT Customers.customer_id, last_name, amount
FROM Customers
INNER JOIN Orders
  ON Customers.customer_id = Orders.customer_id
ORDER BY amount;
```

customer_id	first_name	last_name	age	country
1	John	Doe	31	USA
2	Robert	Luna	22	USA
3	David	Robinson	22	UK
4	John	Reinhardt	25	UK
5	Betty	Doe	28	UAE

Table: Customers

order_id	item	amount	customer_id
1	Keyboard	400	4
2	Mouse	300	4
3	Monitor	12000	**3**
4	Keyboard	400	1
5	Mousepad	250	2

Table: Orders

customer_id	last_name	amount
1	Doe	400
2	Luna	250
3	Robinson	12000
4	Reinhardt	400
4	Reinhardt	300

Try it here: https://www.programiz.com/sql/online-compiler/

So far, how do you like SQL?

- What’s not hard, in my opinion
  - Learning the syntax of SQL
    - SQL reads like English
    - SQL is well documented, and has a large online community

- What’s hard, perhaps for beginners
  - Setup an environment so you can query data using SQL
    - Setup a database, design the tables, inject the data, etc.
    - Usually the job of a database engineer / administrator (not you, the business analyst)
  - **Solve a business question with a series SQL statements (our focus)**
    - A business question $\rightarrow$ SQL statements (you, the data/business analyst)

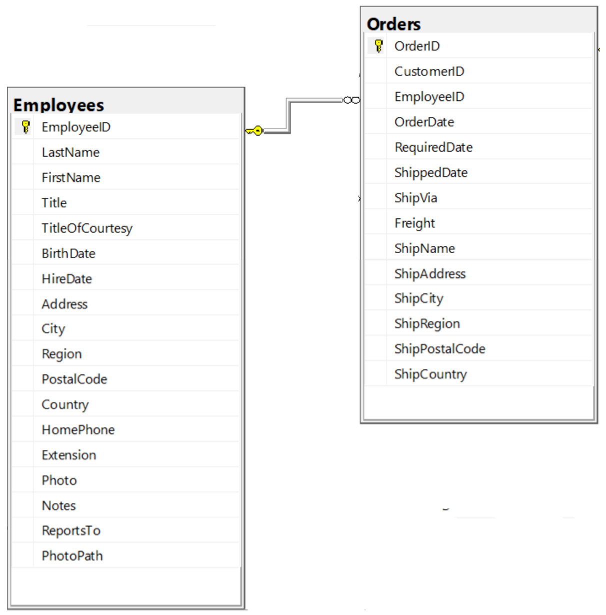
Back track a bit, what’s SQL

- Most widely used database (DB) language
    - a domain specific language: managing data stored in relational DBs
- Not a proprietary language
    - Open specifications/standards (ANSI & ISO)
    - All major DBMS (DB Mgmt. System) vendors implement Standard SQL
    - However, SQL Extensions are usually DB specific (SQL dialects)
- Powerful despite simplicity

ANSI - American National Standards Institute; ISO – International Organization for Standardization

What’s a DB and a Relational DB

- What’s a database: A collection of data in an organized way
- *Relational DB (RDB)*
    - tables
        - columns/fields/variables, and a datatype per column
        - rows/records/observations
    - primary key, foreign key, constraints and relationships
    - other objects: indices, views, triggers and many more

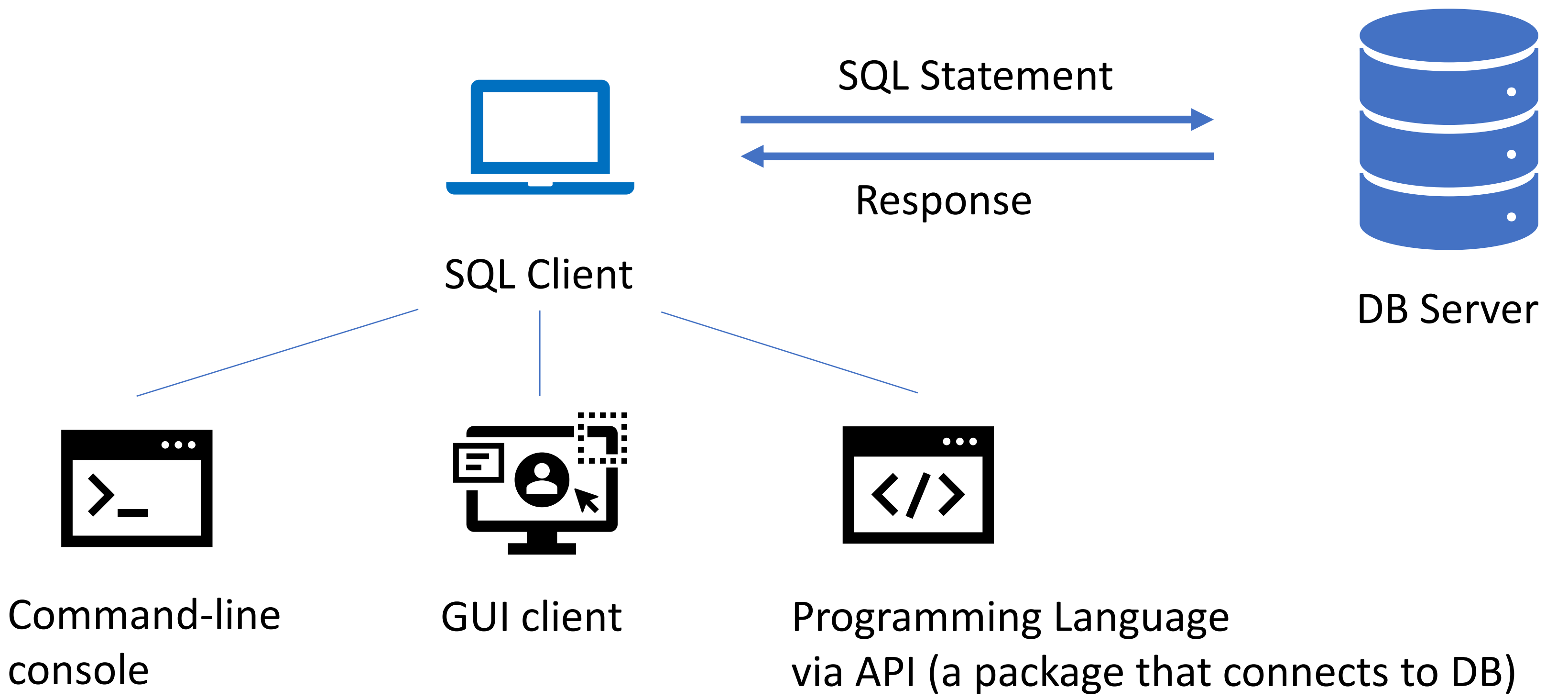


What is a DB Management System

- A software system that manages/maintains DBs

- A few examples of Relational DBMS (RDBMS)
  - Open source: SQLite, DuckDB, MariaDB, PostgreSQL
  - Commercial: MySQL, Microsoft SQL Server, Oracle, etc.

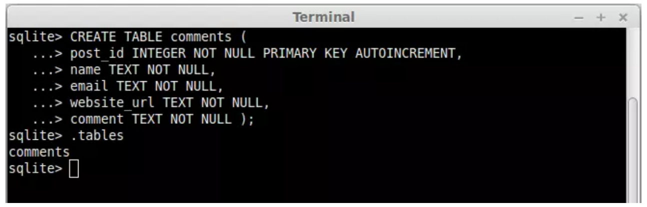
Connect to a DB and write SQL – Architecture



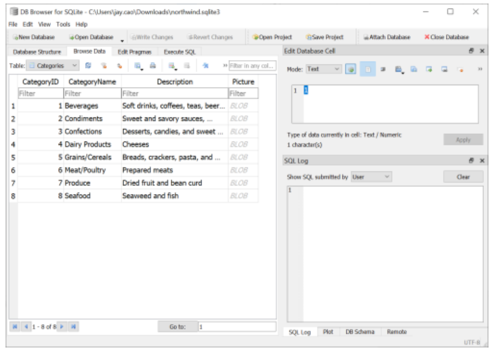
Note: SQL client and DB server can be on the same computer

SQL Clients - A Few Examples

- DB specific management client
  - command-line console
  - GUI (Graphic User Interface) client
    - e.g., [DB Browser for SQLite](), [MySQL Workbench](), [pgAdmin for PostgreSQL](), [MS SSMS]()



```
sqlite> CREATE TABLE comments (
   ...> post_id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
   ...> name TEXT NOT NULL,
   ...> email TEXT NOT NULL,
   ...> website_url TEXT NOT NULL,
   ...> comment TEXT NOT NULL );
sqlite> .tables
comments
sqlite> 
```



DB Browser for SQLite - C:\Users\jay.cao\Downloads\northwind.sqlite3

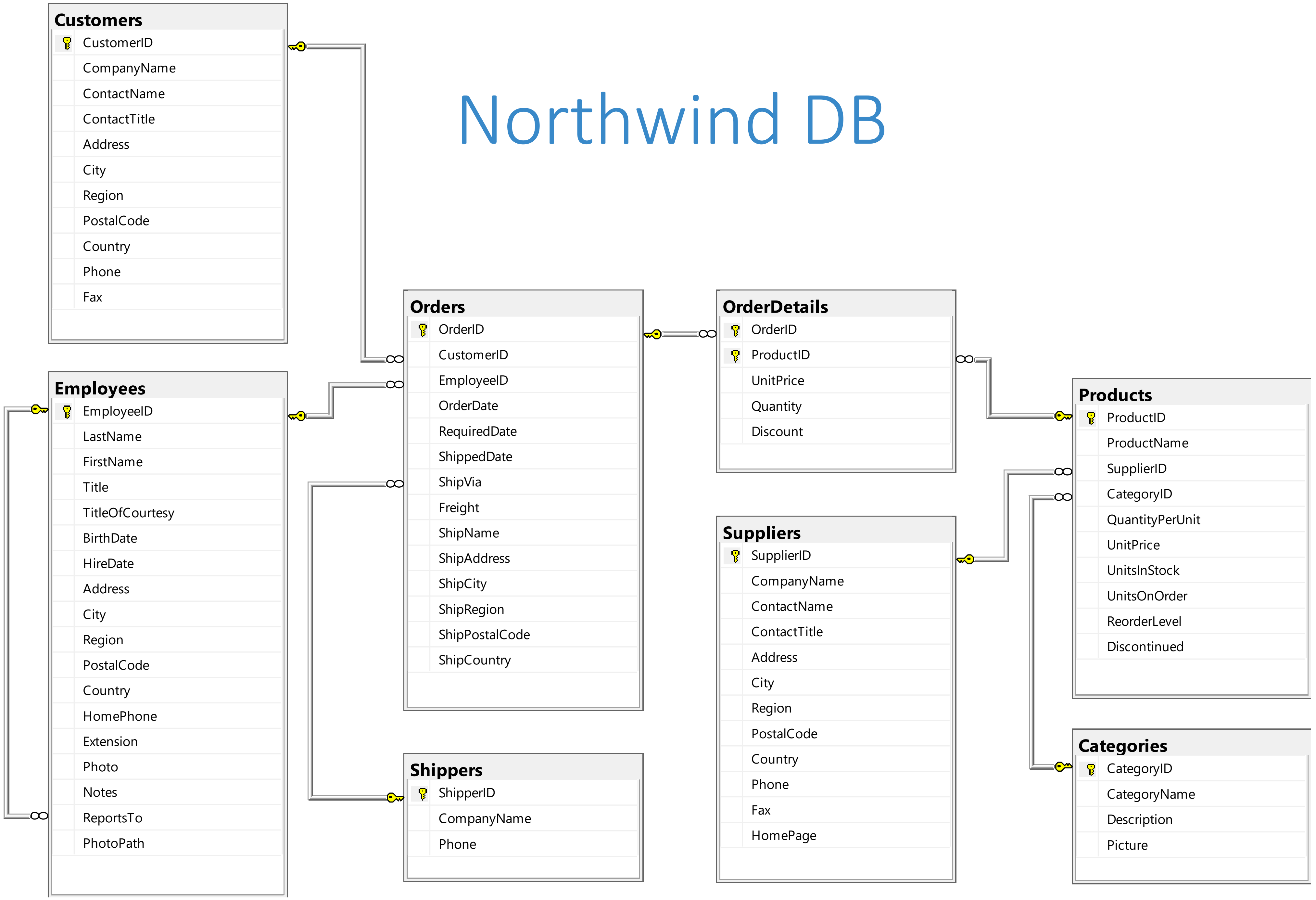
	CategoryID	CategoryName	Description	Picture
1	1	Beverages	Soft drinks, coffees, teas, beer...	BLOB
2	2	Condiments	Sweet and savory sauces, ...	BLOB
3	3	Confections	Desserts, candies, and sweet ...	BLOB
4	4	Dairy Products	Cheeses	BLOB
5	5	Grains/Cereals	Breads, crackers, pasta, and ...	BLOB
6	6	Meat/Poultry	Prepared meats	BLOB
7	7	Produce	Dried fruit and bean curd	BLOB
8	8	Seafood	Seaweed and fish	BLOB

- Generic DB client can connect to different DBs through connectors
  - GUI client (e.g. [DBeaver](), [Beekeeper Studio](), [Navicat]())
  - Programming language
    - e.g., Python + [SQLAlchemy]() + DBAPI (e.g. [SQLite](), [MySQL](), [PostgreSQL](), etc.), R + [dbplyr]()
    - **In this workshop**: Python + [JupySQL]() + [SQLAlchemy]() --> [DuckDB]()

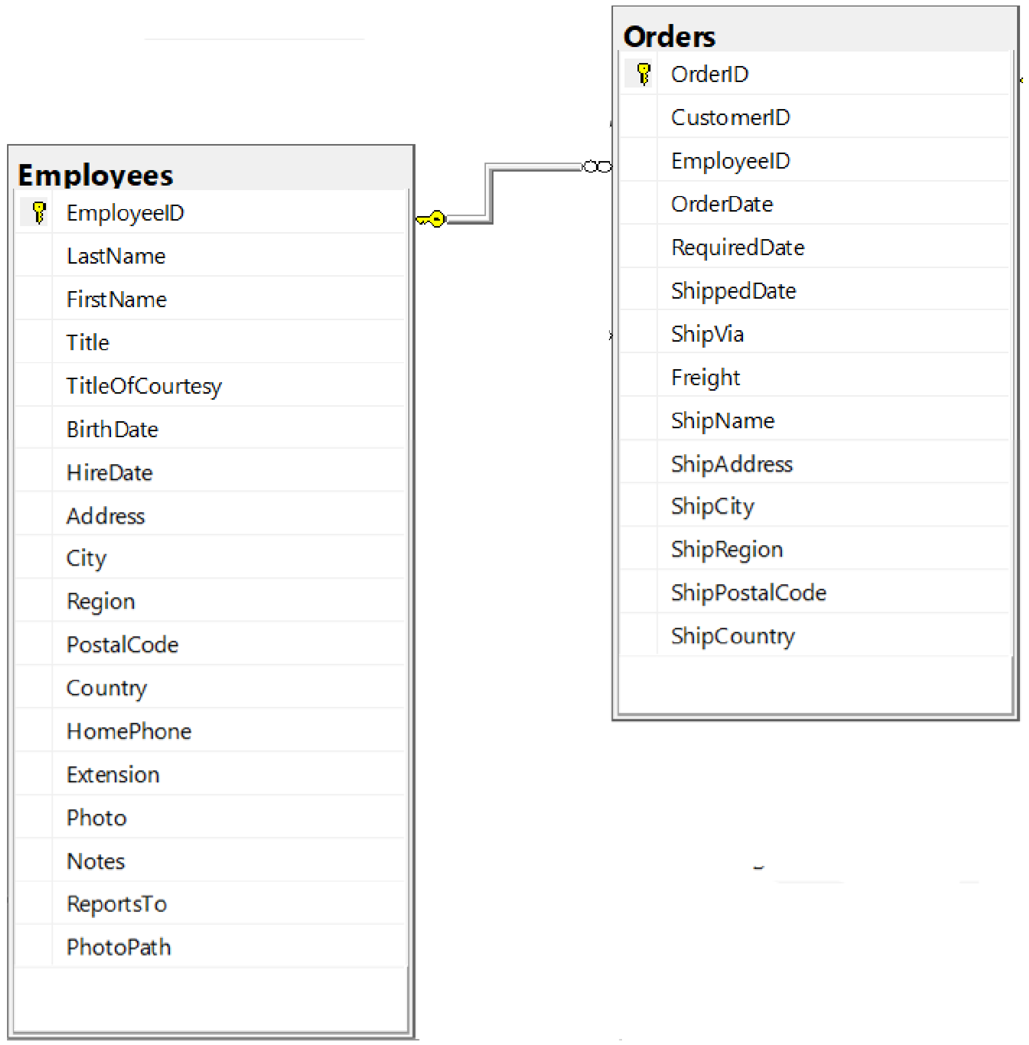
SQL Hands-on Learning (Learning-by-doing)

- Workshop website: https://tdmdal.github.io/sql-roma-rbac-2023/
- Google Colab
  - Google's Jupyter Notebook
  - A notebook can contain live code, equations, visualizations and narrative text
- Why DuckDB?
  - Light-weight, feature rich, and fast
  - Perfect for stand-alone data manipulation/analysis tasks on your laptop
  - perfect for learning SQL

Northwind DB



Primary key (PK), foreign key (FK), constraints & relationships - 1



EmployeeID	LastName	FirstName	Title	…
1	Davolio	Nancy	Sales Representative	…
2	Fuller	Andrew	Vice President, Sales	…
3	Leverling	Janet	Sales Representative	…
4	Peacock	Margaret	Sales Representative	…
…	…	…	…	…

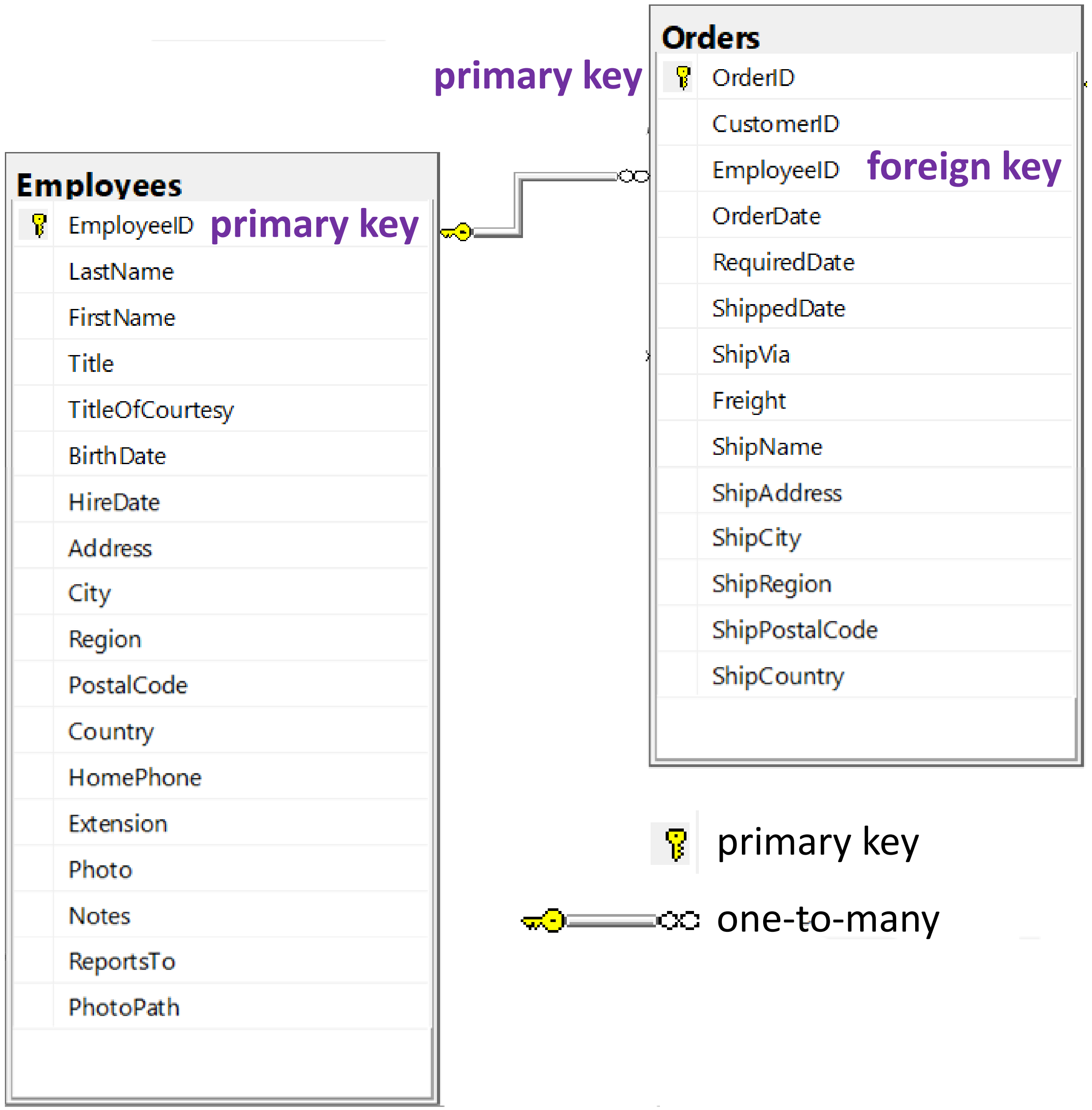
OrderID	CustomerID	EmployeeID	…
10248	VINET	5	…
10249	TOMSP	6	…
10250	HANAR	4	…
…	…	…	…

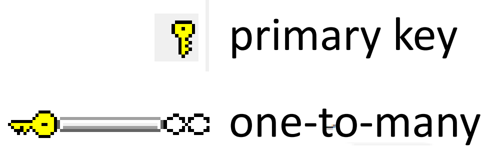
PK, FK, constraints & relationships - 2

- Two keys
  - **primary key**: uniquely identifies an observation in its own table
  - **foreign key**: uniquely identifies an observation in another table

- Relationship between tables
  - one-to-one
  - **one-to-many**
  - many-to-many

- FK constraints





Hands-on Part 1: Basics

- Retrieve data: `SELECT...FROM...`
- Filter data: `SELECT...FROM...WHERE...`
  - `IN`, `NOT`, `LIKE` and % wildcard
- Sort retrieved data: `SELECT...FROM...ORDER BY...`
- Create calculated fields
  - mathematical calculations (e.g. `+, -, *, /`)
  - data manipulation functions (e.g. `year()`, ||)

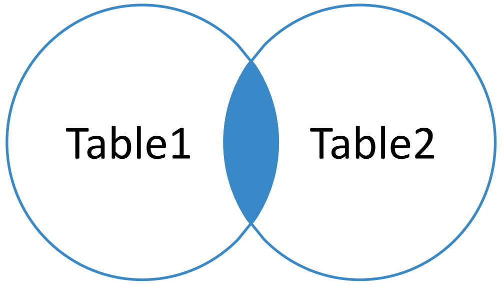
Hands-on Part 2: Summarize and Group Data

- Summarize data using aggregate functions (e.g. `COUNT()`, `MIN()`, `MAX()`, and `AVG()`).
- Group data and filter groups: `SELECT...FROM...GROUP BY...HAVING...`
- SELECT statement syntax ordering
  - `SELECT...FROM...WHERE...GROUP BY...HAVING...ORDER BY...`

Hands-on Part 3: Join Tables

- Inner join: `SELECT...FROM...INNER JOIN...ON...`
- Left join: `SELECT...FROM...LEFT JOIN...ON...`
- Other join variations (see appendix)

Join – Inner Join



```
SELECT *
FROM Table1
   INNER JOIN Table2
   ON Table1.pk = Table2.fk;
```

Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

pk	t1c1	fk	t2c1
1	a	1	c
1	a	1	d

Join – Left (Outer) Join

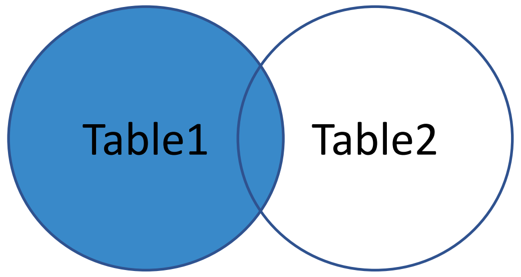


Table1

pk	t1c1
1	a
2	b

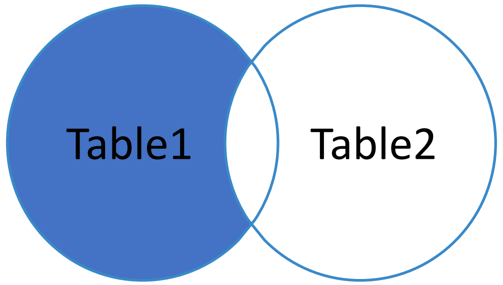
Table2

fk	t2c1
1	c
1	d
3	e

```
SELECT *
FROM Table1
  LEFT JOIN Table2
  ON Table1.pk = Table2.fk;
```

pk	t1c1	fk	t2c1
1	a	1	c
1	a	1	d
2	b	null	null

Join - Left (Outer) Join With Exclusion



```
SELECT *
FROM Table1
  LEFT JOIN Table2
  ON Table1.pk = Table2.fk
WHERE Table2.fk is NULL;
```

Table1

pk	t1c1
1	a
2	b

Table2

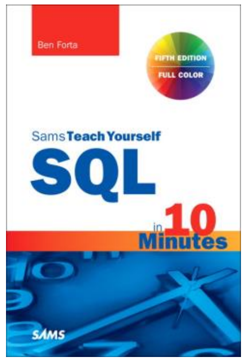
fk	t2c1
1	c
1	d
3	e

pk	t1c1	fk	t2c1
2	b	null	null

Learning Resources

- Online free resources
  - Learn SQL: SQL Tutorial for Beginners by programiz (from Nepal)
  - Introduction to DBs and SQL by programiz
  - SQL tutorial by W3 School
  - SQL for Data Analysis at Udacity
  - Learning SQL Programming by Scott Simpson (1h 27m) on LinkedIn Learning

- A little book
  - SQL in 10mins a Day (5th edition) by Ben Forta



Appendix

- Many join operation variations
- SQL is much more…

Join – Right (Outer) Join*

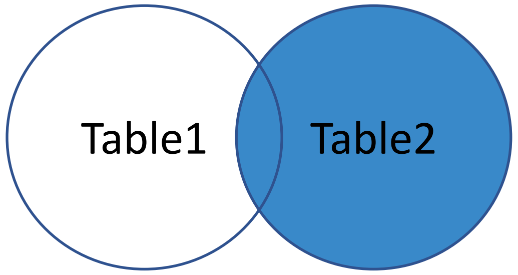


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```
SELECT *
FROM Table2
  LEFT JOIN Table1
  ON Table2.fk = Table1.pk
```

Use LEFT JOIN if a DBMS doesn't support RIGHT JOIN keyword.

```
---------------------------
SELECT *
FROM Table1
  RIGHT JOIN Table2
  ON Table1.pk = Table2.fk;
```

DuckDB supports RIGHT JOIN keyword.

pk	t1c1	fk	t2c1
1	a	1	c
1	a	1	d
null	null	3	e

Join - Right (Outer) Join With Exclusion*

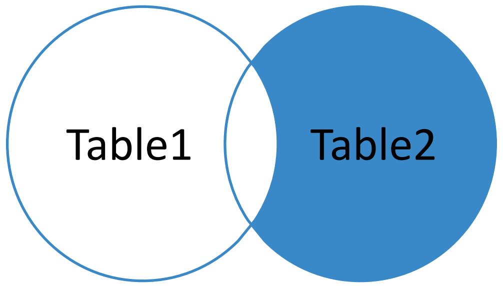


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```
SELECT *
FROM Table2
  LEFT JOIN Table1
  ON Table2.fk = Table1.pk
WHERE Table1.pk is NULL;
```

Use LEFT JOIN and WHERE if a DBMS doesn't support RIGHT JOIN keyword

```
---------------------------
SELECT *
FROM Table1
  RIGHT JOIN Table2
  ON Table1.pk = Table2.fk
WHERE Table1.pk is NULL;
```

DuckDB supports RIGHT JOIN keyword.

pk	t1c1	fk	t2c1
null	null	3	e

Join – Full (Outer) Join

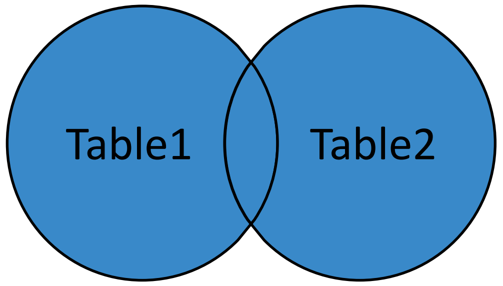


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```
SELECT pk, t1c1, fk, t2c1
FROM Table1
  FULL JOIN Table2
  ON Table1.pk = Table2.fk;
```

DuckDB supports FULL (OUTER) JOIN keyword.

pk	t1c1	fk	t2c1
1	a	1	c
1	a	1	d
2	b	null	null
null	null	3	e

Join – Full (Outer) Join (using LEFT JOIN & UNION)

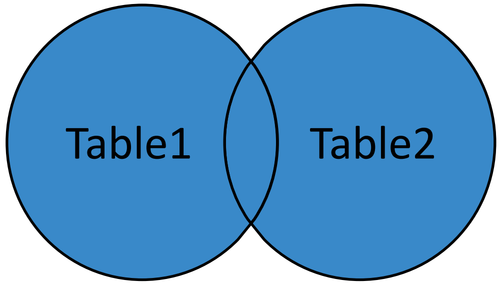


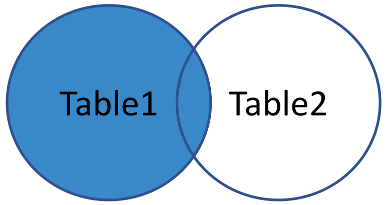
Table1

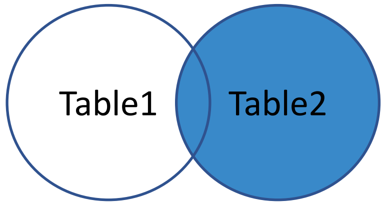
pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```
SELECT pk, t1c1, fk, t2c1
FROM Table1
  LEFT JOIN Table2
  ON Table1.pk = Table2.fk
UNION
SELECT pk, t1c1, fk, t2c1
FROM Table2
  LEFT JOIN Table1
  ON Table2.fk = Table1.pk;
```





pk	t1c1	fk	t2c1
1	a	1	c
1	a	1	d
2	b	null	null
null	null	3	e

Note: Do it the above way if a DBMS doesn’t support FULL (OUTER) JOIN.

Join – Full (Outer) Join With Exclusion*

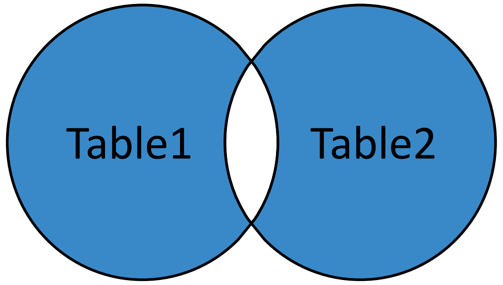


Table1

pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```
SELECT pk, t1c1, fk, t2c1
FROM Table1
  FULL JOIN Table2
  ON Table1.pk = Table2.fk
WHERE Table2.fk is NULL
  OR table2.fk is NULL;;
```

DuckDB supports FULL (OUTER) JOIN key word.

pk	t1c1	fk	t2c1
2	b	null	null
null	null	3	e

Join – Full (Outer) Join With Exclusion*

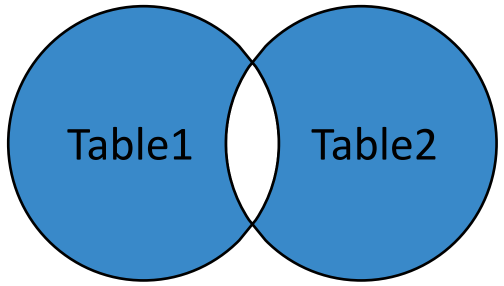


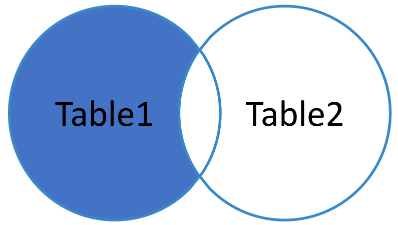
Table1

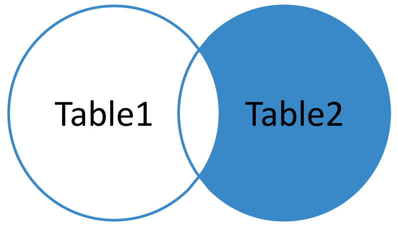
pk	t1c1
1	a
2	b

Table2

fk	t2c1
1	c
1	d
3	e

```
SELECT pk, t1c1, fk, t2c1
FROM Table1
  LEFT JOIN Table2
  ON Table1.pk = Table2.fk
WHERE Table2.fk is NULL
UNION
SELECT pk, t1c1, fk, t2c1
FROM Table2
  LEFT JOIN Table1
  ON Table2.fk = Table1.pk
WHERE Table1.pk is NULL;
```





pk	t1c1	fk	t2c1
2	b	null	null
null	null	3	e

Note: Do it the above way if a DBMS doesn't support FULL (OUTER) JOIN.

SQL is much more - 1

- Sub-query
- CTE and temporary table
- Self-join
- CASE keyword
- UNION keyword

SQL is much more - 2

- Insert data (`INSERT INTO…VALUES…; INSERT INTO…SELECT…FROM…`)
- Update data (`UPDATE…SET…WHERE…`)
- Delete data (`DELETE FROM…WHERE…`)
- Manipulate tables (`CREATE TABLE…; ALTER TABLE…; DROP TABLE…`)
- Views (`CREATE VIEW…AS…`)

The list goes on and on

- Stored procedures
- Functions
- Transaction processing
- Cursors (going through table row by row)
- WINDOW function
- Query optimization
- DB permissions & security
- …

Ref. A stack overflow discussion on What is “Advanced” SQL.