

# Table of Contents

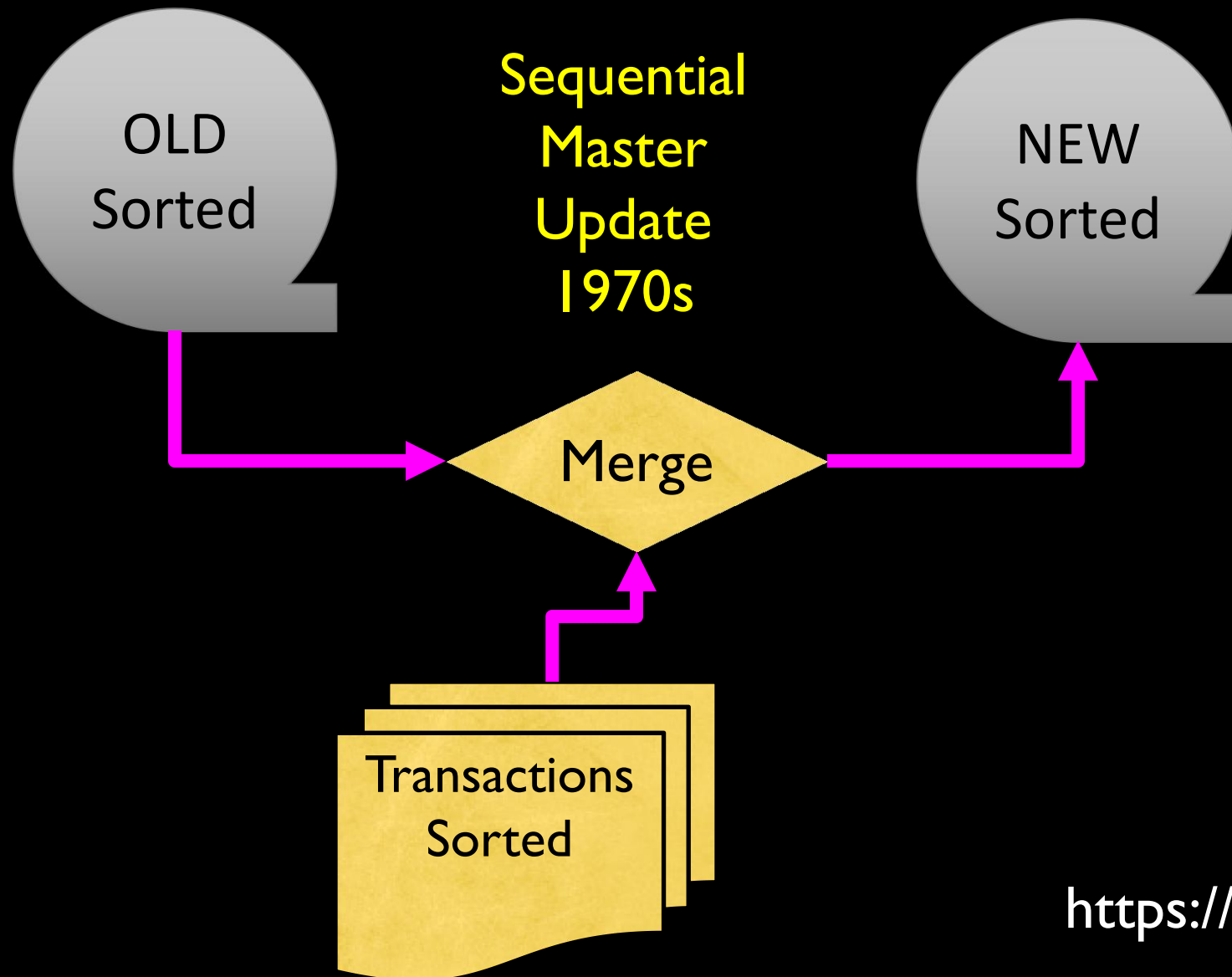
This slide deck consists of slides used in 2 lecture videos in Week 5. Below is a list of shortcut hyperlinks for you to jump into specific sections.

- (page 2) [Week 5: How Databases Work](#)
- (page 11) [Week 5: Introduction to Structured Query Language \(SQL\)](#)

Charles Severance  
[www.dj4e.com](http://www.dj4e.com)

# Single Table SQL





[https://en.wikipedia.org/wiki/IBM\\_729](https://en.wikipedia.org/wiki/IBM_729)

# Random Access

- When you can randomly access data...
- How can you layout data to be most efficient?
- Sorting might not be the best idea



[https://en.wikipedia.org/wiki/Hard\\_disk\\_drive\\_platter](https://en.wikipedia.org/wiki/Hard_disk_drive_platter)

# Structured Query Language

- Structured Query Language (SQL) came out of a government / industry partnership
- National Institute of Standards and Technology (NIST)

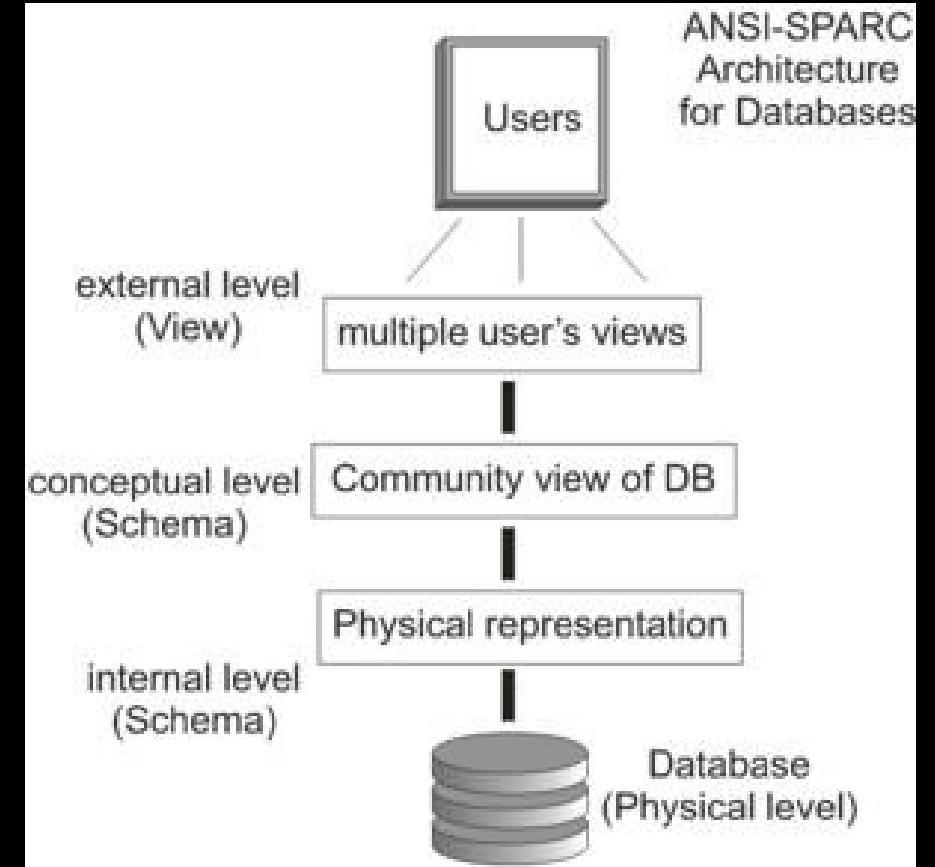


<https://youtu.be/rLUm3vst87g>

# SQL

Structured Query Language is the language we use to issue commands to the database

- Create/Insert data
- Read/Select some data
- Update data
- Delete data



<http://en.wikipedia.org/wiki/SQL>

[https://en.wikipedia.org/wiki/ANSI-SPARC\\_Architecture](https://en.wikipedia.org/wiki/ANSI-SPARC_Architecture)

# Relational Databases

Relational databases model data by storing rows and columns in tables. The power of the relational database lies in its ability to efficiently retrieve data from those tables and in particular where there are multiple tables and the relationships between those tables involved in the query.

[http://en.wikipedia.org/wiki/Relational\\_database](http://en.wikipedia.org/wiki/Relational_database)



# Common Database Systems

- Three major Database Management Systems in wide use
  - **Postgres** – Open source, enterprise-scale, very tweakable
  - **Oracle** - Large, commercial, enterprise-scale, very tweakable
  - **MySql** - Simpler but very fast and scalable - commercial open source
  - **SqlServer** - Very nice - from Microsoft (also Access)
- Many other smaller projects, free and open source
  - HSQL, **SQLite**, ...



# Database Model

A **database model** or **database schema** is the **structure or format of a database**, described in a formal language supported by the database management system. In other words, a “database model” is the application of a data model when used in conjunction with a database management system.

[http://en.wikipedia.org/wiki/Database\\_model](http://en.wikipedia.org/wiki/Database_model)

# SQL

**Structured Query Language** is the language we use to issue commands to the database

- Create data (a.k.a Insert)
- Retrieve data
- Update data
- Delete data

<http://en.wikipedia.org/wiki/SQL>

# Lets Make a Database

<https://www.dj4e.com/lectures/SQL-01-Basics.txt>

# Start Simple - A Single Table

```
$ sqlite3 zip.sqlite3
SQLite version 3.11.0 2016-02-15 17:29:24
Enter ".help" for usage hints.
sqlite> .tables
sqlite> CREATE TABLE Users(
...>     id INTEGER NOT NULL
...>         PRIMARY KEY AUTOINCREMENT,
...>     name VARCHAR(128),
...>     email VARCHAR(128)
...> ) ;
sqlite> .tables
Users
sqlite> .schema Users
CREATE TABLE Users(
    id INTEGER NOT NULL
        PRIMARY KEY AUTOINCREMENT,
    name VARCHAR(128),
    email VARCHAR(128)
);
sqlite>
```

```
CREATE TABLE Users(
    id integer NOT NULL
        PRIMARY KEY
        AUTOINCREMENT,
    name VARCHAR(128),
    email VARCHAR(128)
);
```

# SQL: Insert

The Insert statement inserts a row into a table

```
INSERT INTO Users (name, email) VALUES ('Kristin', 'kf@umich.edu')
```

# SQL: Delete

Deletes a row in a table based on selection criteria

```
DELETE FROM Users WHERE email='ted@umich.edu'
```

# SQL: Update

Allows the updating of a field with a where clause

```
UPDATE Users SET name='Charles' WHERE email='csev@umich.edu'
```



# Retrieving Records: Select

The select statement retrieves a group of records - you can either retrieve all the records or a subset of the records with a WHERE clause

```
SELECT * FROM Users
```

```
SELECT * FROM Users WHERE email='csev@umich.edu'
```

# Sorting with ORDER BY

You can add an **ORDER BY** clause to **SELECT** statements to get the results sorted in ascending or descending order

```
SELECT * FROM Users ORDER BY email
```

```
SELECT * FROM Users ORDER BY name DESC
```

# SQL Summary

```
INSERT INTO Users (name, email) VALUES ('Kristin', 'kf@umich.edu')
```

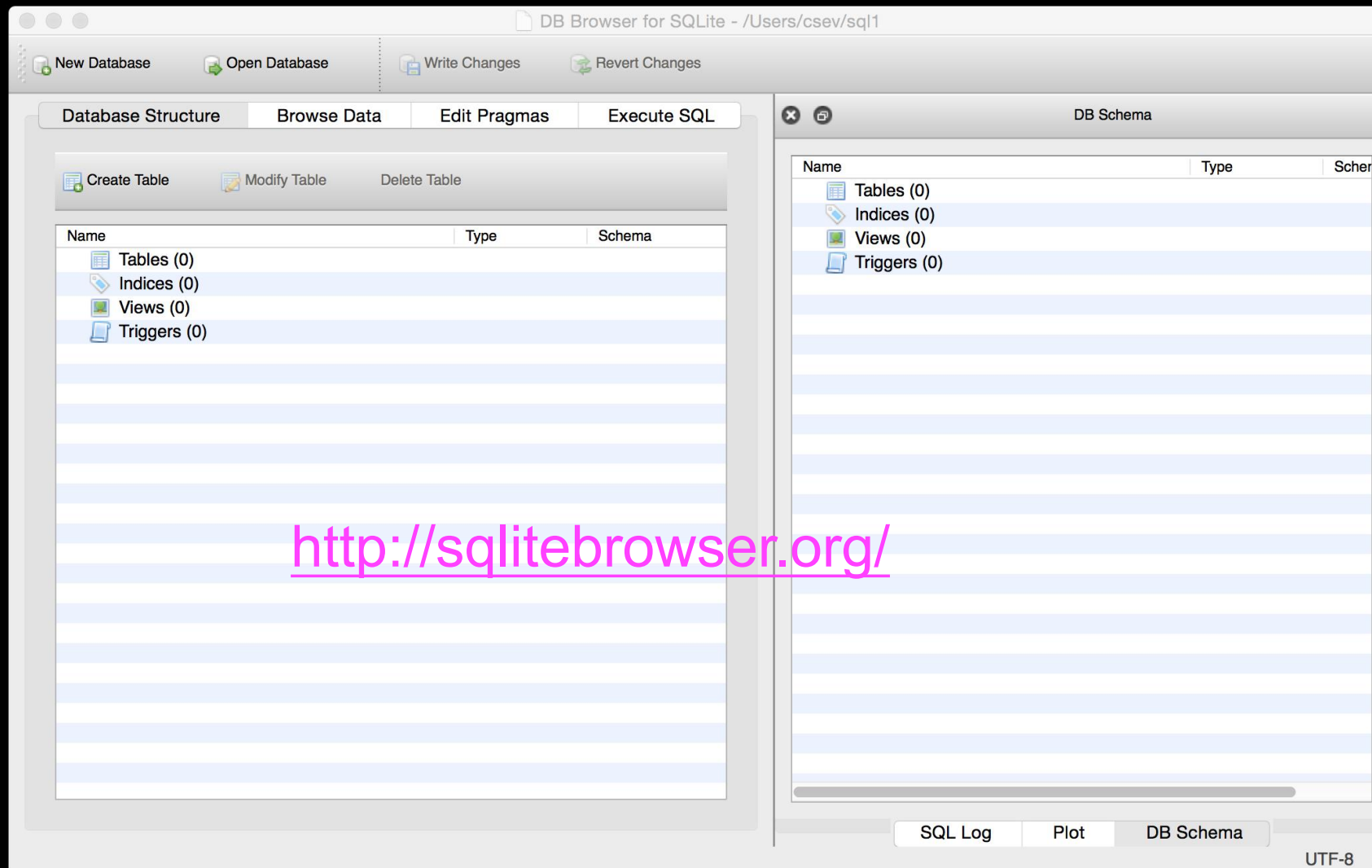
```
DELETE FROM Users WHERE email='ted@umich.edu'
```

```
UPDATE Users SET name="Charles" WHERE email='csev@umich.edu'
```

```
SELECT * FROM Users
```

```
SELECT * FROM Users WHERE email='csev@umich.edu'
```

```
SELECT * FROM Users ORDER BY email
```



# Acknowledgements / Contributions

These slides are Copyright 2019- Charles R. Severance ([www.dr-chuck.com](http://www.dr-chuck.com)) as part of [www.dj4e.com](http://www.dj4e.com) and made available under a Creative Commons Attribution 4.0 License. Please maintain this last slide in all copies of the document to comply with the attribution requirements of the license. If you make a change, feel free to add your name and organization to the list of contributors on this page as you republish the materials.

Initial Development: Charles Severance, University of Michigan School of Information

Insert new Contributors and Translators here including names and dates

Continue new Contributors and Translators here