SQL Bootcamp

By Craig Sakuma

Introductions

Craig Sakuma

- Data Science Consultant
- General Assembly Instructor for Data Science
- Co-founder of Deal Décor
- MBA from Wharton
- B.Eng from Northwestern University

Fun Fact

Developed a novelty BBQ product that was featured in USA Today

Class Introductions

- Name
- What's your job?
- How do you plan to apply skills from today's workshop?
- Fun Fact

Objectives for Class

- Get strong foundation of SQL
- Immediately use skills at work
- Remove barriers/frustration
- Develop skills to be self-sufficient after class
  - Tools and examples for practicing on your own
  - Comfort with SQL to learn on own
  - Ability to troubleshoot problems

HAVE FUN!

Course Structure

- Lectures on topics
  - Interaction is good
  - Feel free to ask questions
  - If there's not enough time to cover questions, we'll put it in a parking lot for after class
- Hands on exercises
  - Pair programming
  - Mix up partners

Schedule

Time	Topic
10:00 – 11:00	Overview of SQL
11:00 – 12:00	Fundamentals of Queries
12:00 – 1:00	Lunch
1:00 – 2:15	Creating Tables
2:15 – 3:30	Joining Tables
3:30 – 4:00	Functions
4:00 – 5:00	Group By

What is SQL?

- Structured Query Language
  - Programming language
  - Structured data (requires some overhead)
  - Relational Database
  - Allows you to share large sets with many users
  - Scalable data storage

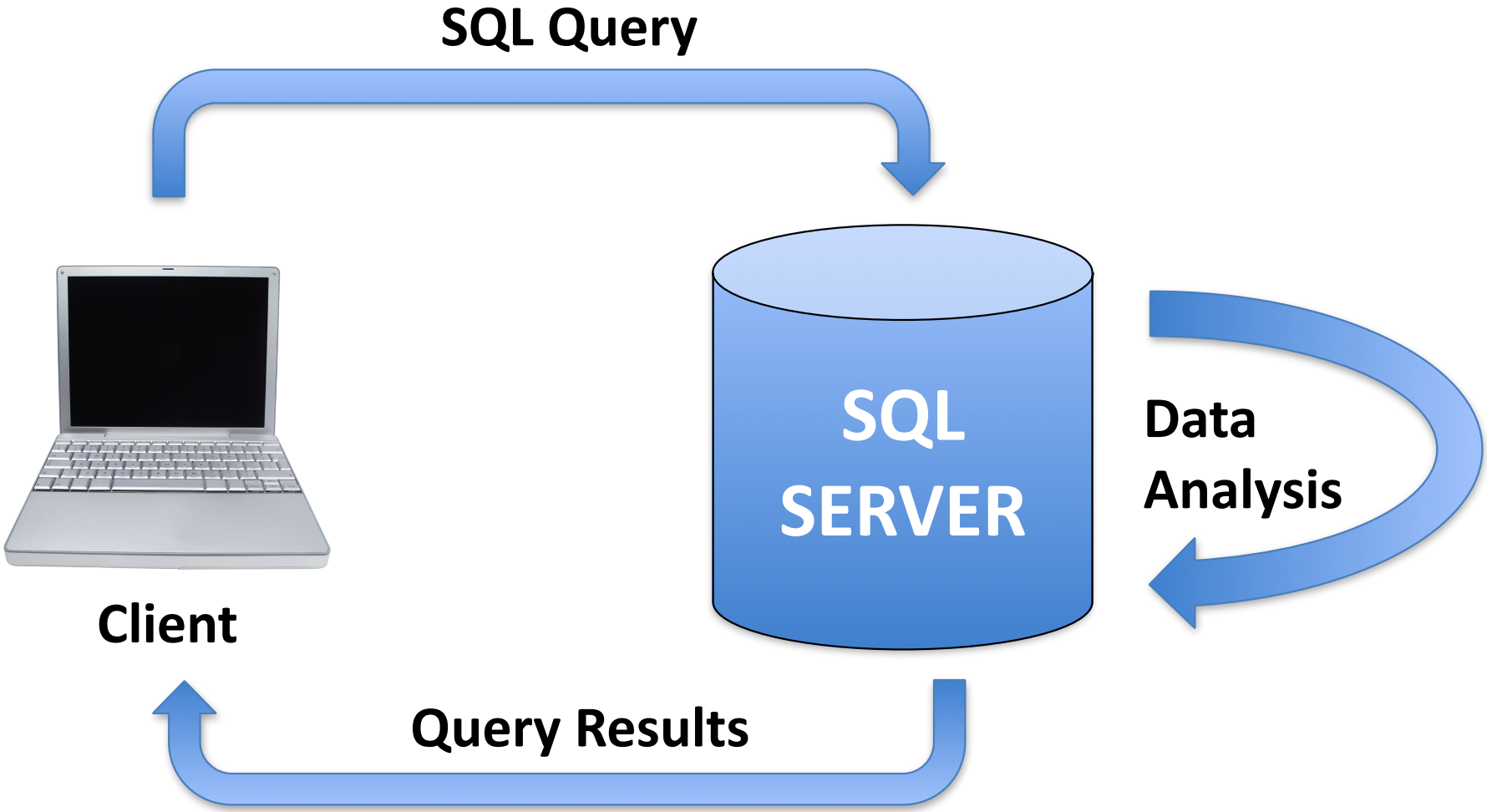
SQL is Searchable

SQL Types

- PostgreSQL
- SQLite
- MySQL
- Amazon Redshift
- Oracle
- Microsoft SQL Server

Functionality is similar
but syntax can be different

SQL – Behind the Scenes



However, SQLite operates without a Server

Tables

- Data containers
- Organized by rows and columns

CustomerId	FirstName	LastName	City
16	Frank	Harris	Mountain View
17	Jack	Smith	Redmond
18	Michelle	Brooks	New York
19	Tim	Goyer	Cupertino
20	Dan	Miller	Mountain View
21	Kathy	Chase	Reno
22	Heather	Leacock	Orlando
23	John	Gordon	Boston
24	Frank	Ralston	Chicago
25	Victor	Stevens	Madison

Tables Are Like Spreadsheets

Tables

Data Fields (like columns)

Database Records (like rows)

CustomerId	FirstName	LastName	City
16	Frank	Harris	Mountain View
17	Jack	Smith	Redmond
18	Michelle	Brooks	New York
19	Tim	Goyer	Cupertino
20	Dan	Miller	Mountain View
21	Kathy	Chase	Reno
22	Heather	Leacock	Orlando
23	John	Gordon	Boston
24	Frank	Ralston	Chicago
25	Victor	Stevens	Madison

- Fields have Data Types
- Primary Keys (unique identifier)
- Foreign Keys (identifier for linking to other tables)

Data Types

- Characters
  - char
  - varchar
- Numbers
  - integers
  - floats
- Boolean (True/False)
- Temporal
  - Dates
  - Times

Sample Database – sql_workshop.db

11 Data Tables

- Album
- Artist
- Customer
- Employee
- Genre
- Invoice
- InvoiceLIne
- MediaType
- Playlist
- PlaylistTrack
- Track

Note: Database example come from ChinookDatabase 1.4

SQLite3 – Start a Session

1. Open terminal
2. For PC Users, change to directory with sqlite3 files
3. Enter “sqlite3”
4. Your session should start

```
~$sqlite3
SQLite version 3.7.13 2012-07-17 17:46:21
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite>
```

To end session, type .quit or .exit

Syntax for Slides

Brackets are placeholders

Example (for change directory):

cd [directory name]

Applied for changing to directory ‘sqlite3’

cd sqlite3

SQLite3 – Open a Database

At terminal launch sqlite:

sqlite3

Open a database:

.open sql_workshop.db

View the database schema:

.schema

Explore the Data Tables and Identify Foreign Keys

Query Types

- SELECT
  - Creates view of records from database
- INSERT
  - Inserts new records into a table
- UPDATE
  - Updates existing records into a table
- DELETE
  - Removes records

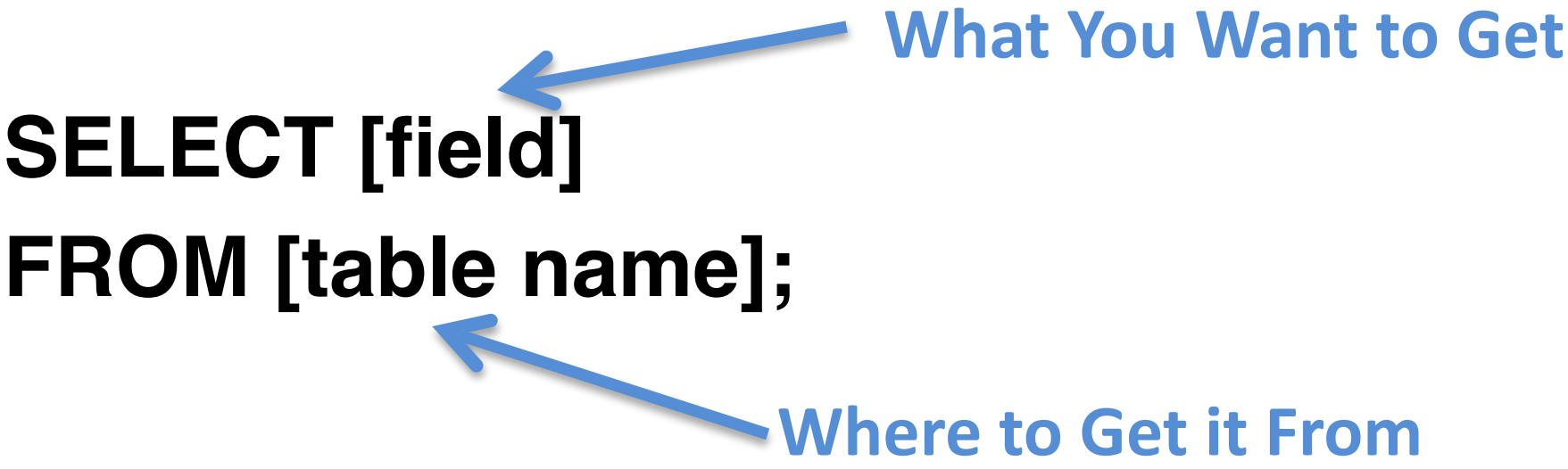
CHANGES DATA IN TABLES

Basic Query Structure

- SELECT (values you want to view)
- FROM (data source – table name)
- WHERE (conditions for results)
- ; (delimiter for end of query)

Simplest Query

You Only Need Two Things for a Query:



Semi-colon is a delimiter used to mark the end of a query

Common Basic Queries

```
SELECT [field1 ], [field 2]
FROM [table name]
WHERE [field 1] = 'value';
```

Select multiple fields

Create conditions for query

You Can Create Multiple Conditions with AND /OR

Queries with Limits

```
SELECT *
FROM [table name]
WHERE [field 1] > 10
LIMIT 10;
```

Wildcard for Selecting All Fields

Limit Results to 10 Records

Avoid Long Queries that Slow Down the Server
…And Keep Your Database Admin Happy

SQLite3 – Using Text Editor to Create SQL Files

1. Open your text editor

2. Create a file named basic_query.sql with:
   SELECT * FROM Album LIMIT 10;

3. Save file

SQLite3 – Read SQL File

Execute SQL from file

.read [filename]

Try it yourself:

.read basic_query.sql

```
sqlite> .read basic_query.sql
1|For Those About To Rock We Salute You|1
2|Balls to the Wall|2
3|Restless and Wild|2
4|Let There Be Rock|1
5|Big Ones|3
6|Jagged Little Pill|4
7|Facelift|5
8|Warner 25 Anos|6
9|Plays Metallica By Four Cellos|7
10|Audioslave|8
```

SQLite3 – Comment

Write comments that aren't executed as code

-- [single line of text]

/*[multiple lines of text]

[multiple lines of text]*/

Use for documenting code

Comment out code that you don't want to run

sublime text shortcut: command + /

SQLite3 Settings

View Column Headers

.header on

Organize Data in Columns:

.mode col

Makes Viewing Results
Easier to Read

Instructions for Exercises

- Pair programming
  - Using only one computer
  - Take turns typing
  - Collaborate on solutions
- Resources
  - Online documentation
  - Stackoverflow / Google

Exercise #1

1. What are the Genres in the database?

2. What are the customer names that are from California?
   (Hint: text strings need to be in single quotes)

What other interesting queries can you create?

SELECT Options

- SELECT COUNT(*)
  Counts the records in the database
- SELECT DISTINCT first_name
  Selects unique first names
  (Note: if multiple fields are provided it will select distinct combinations of those fields)
- SELECT COUNT(DISTINCT first_name)
  Counts the number of unique first names

WHERE Clauses

- Combine multiple conditions with AND / OR

```
SELECT *
FROM films
WHERE state= ‘CA’ OR state= ‘WA’ OR state= ‘OR’

WHERE country!= ‘USA’ AND country!=‘Canada’
```

Apply Any Equality conditions: >, >=, <, <=, =, !=

WHERE Clauses

- IS NULL / IS NOT NULL
  WHERE GenreId Is NULL

- IN (‘item1’,’item2’, etc….)
  WHERE Genre IN (‘Pop’,’Rock’)

- BETWEEN [x] AND [y]
  WHERE Milliseconds BETWEEN 180000 AND 240000

Best Practices

- CAPITALIZE SQL COMMANDS
- Use Indentation to Improve Readability:

Option #1

```
SELECT
      AlbumId,
      Title
FROM
      Album
```

Option #2

```
SELECT AlbumId, Title
FROM Album
WHERE AlbumId =1
```

- Error Tracking
  - Create a text file to keep notes on your errors
- Save Class Examples as Files

Exercise #2

1. How many songs are longer than 10 minutes?
2. How many invoices were there in January 2010?
3. How many tracks have NULL Genre?
4. How many distinct album titles are there? How many distinct album IDs? Why would these have different counts?

ORDER BY

- Sorts data
  - ASC for ascending
  - DESC for descending
- Can have multiple fields
  - First field listed takes precedence
- Example:

```
SELECT *
FROM Track
ORDER BY GenreId ASC, Milliseconds DESC
```

LIKE / NOT LIKE

Used in WHERE Clauses

- 'abc' LIKE 'abc' TRUE
- 'abc' LIKE 'a%' TRUE
- 'abc' LIKE '_b_' TRUE
- 'abc' LIKE 'c' FALSE

Example:

WHERE customer.email LIKE ‘%@%._ _ _’

LIKE is not case sensitive

Exercise #3

1. What are the 5 longest songs?
2. R.E.M. has collaborated with a couple artists, can you find which artists they've collaborated with?
   (Hint: Use Like)
3. How many 'Love' songs are there?

Table Commands

- CREATE TABLE
  - Creates new table
  - Requires fields to be defined by data type
- DROP TABLE
  - Deletes table from database
- ALTER TABLE
  - Change database column name
  - Add or remove columns
  - Change table name

Create Table

Creating new tables requires definitions for fields and data types:

```
CREATE TABLE [table name] (
    [field name1] [data type1],
    [filed name2] [data type2],
    etc…
);
```

Create Table Example

```
CREATE TABLE FavoriteSongs (
    id INTEGER PRIMARY KEY,
    Title TEXT,
    Length INTEGER
);
```

Create Table Using Query

```
CREATE TABLE FavoriteTracks AS
    SELECT *
    FROM Track
    WHERE GenreId=1;
```

Commit Changes

Many database require confirmation after changes are made

- Create, Alter or Drop Tables
- Insert, Update or Delete Values

Enter “COMMIT;” after changes are made

However, SQLite automatically commits

Insert Query

Use Insert Queries to add records to your tables

INSERT INTO [table name]
([field 1], [field 2])
VALUES ([value 1], [value2]);

Insert Query Example

```
INSERT INTO FavoriteSongs
(id, Title, Length)
VALUES (0, "Call Me Maybe", 260);
```

Drop Table

Use DROP TABLE to completely remove a table:

DROP TABLE [table name];

DROP TABLE FavoriteSongs;

Exercise #5

1. Create a new table from scratch
2. Insert values into the table
3. Create table from a query
4. Drop the tables you created

JOINS

- Connect data tables together
- Requires foreign key

```
SELECT [field]
FROM [table1]
JOIN [table2]
ON [table1 foreign key] = [table2 foreign key]
```

JOIN Example

```
SELECT *
FROM Track
JOIN Genre
ON Track.GenreId = Genre.GenreId
LIMIT 10;
```

Need to use table name with field name since the same field can be in multiple tables

Multiple Joins

- Queries can include multiple joins
- Order of tables is very important
- Count the results to validate the query operated correctly
- Each join adds extra work for the server to execute (slows down query performance)

Multiple Joins - Example

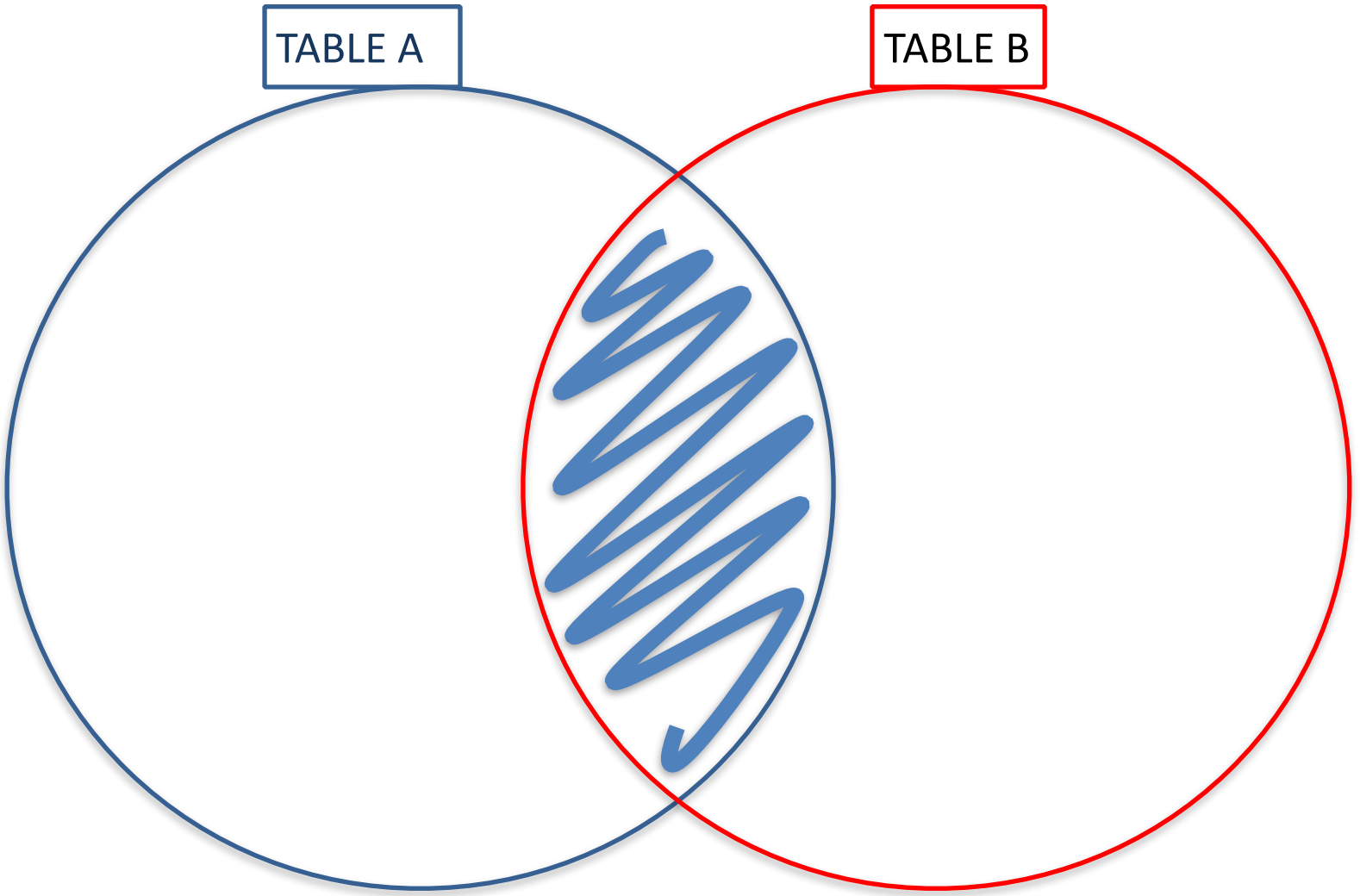
```
SELECT Artist.Name, COUNT(*)
FROM Track
JOIN Album
ON Track.AlbumId = Album.AlbumId
JOIN Artist
ON Album.ArtistId = Artist.ArtistID
LIMIT 10;
```

Order of Tables is EXTREMELY important

Exercise #4

1. How many tracks are in the Rock genre?
2. How many tracks are performed by R.E.M.?
3. How many tracks are performed by R.E.M. with other artists as collaborators?
4. What other interesting queries can you create that join 2 tables?

INNER JOIN



This is the default type for ‘JOIN’

INNER JOIN Example

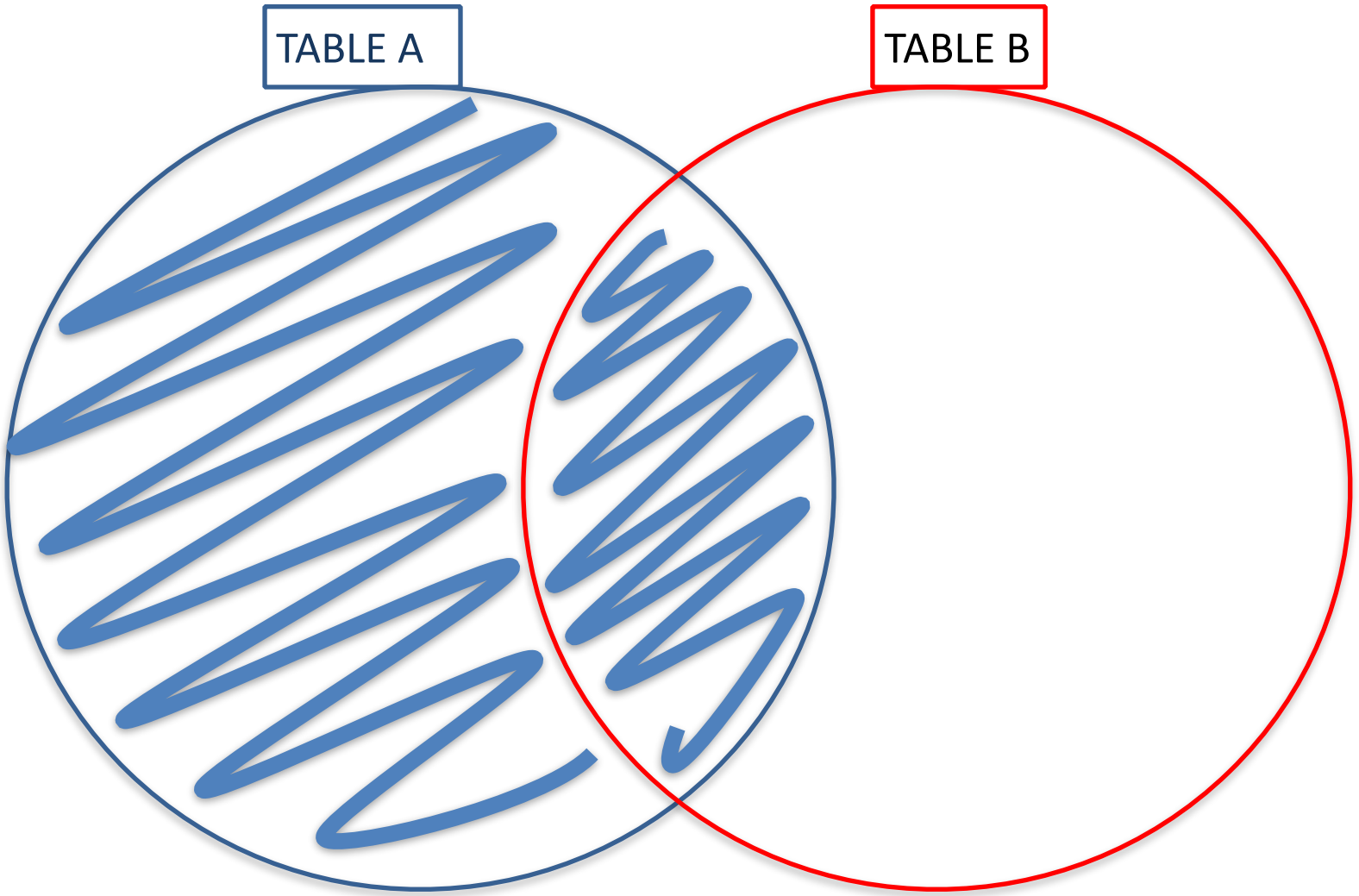
State	*City*
CA	San Francisco
WA	Tacoma

City	*Mascot*
San Francisco	49ers
New York	Jets

Results

State	*City*	*Mascot*
CA	San Francisco	49ers

LEFT JOIN



LEFT JOIN Example

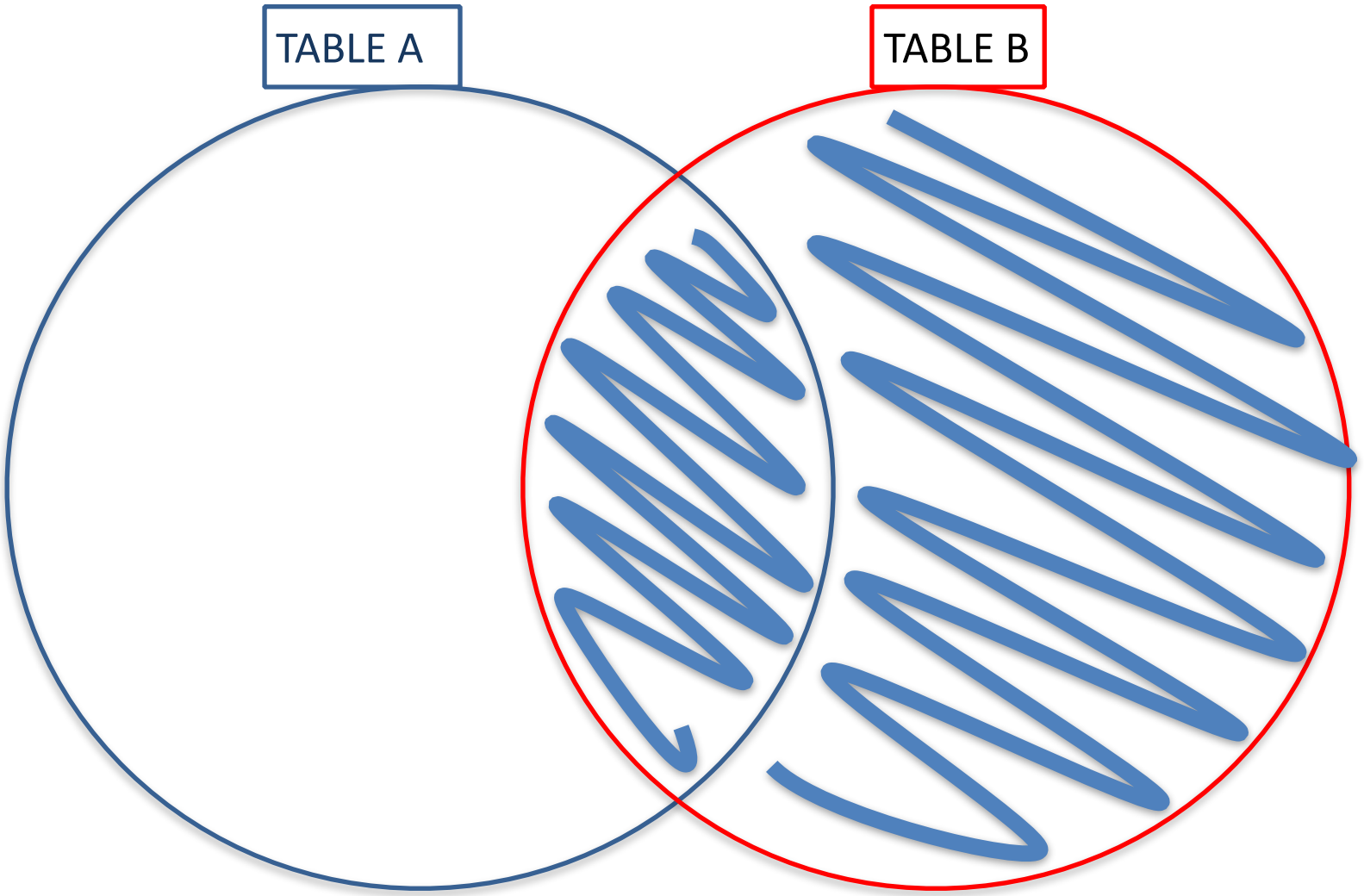
State	*City*
CA	San Francisco
WA	Tacoma

City	*Mascot*
San Francisco	49ers
New York	Jets

Results

State	*City*	*Mascot*
CA	San Francisco	49ers
WA	Tacoma	

RIGHT JOIN



RIGHT JOIN Example

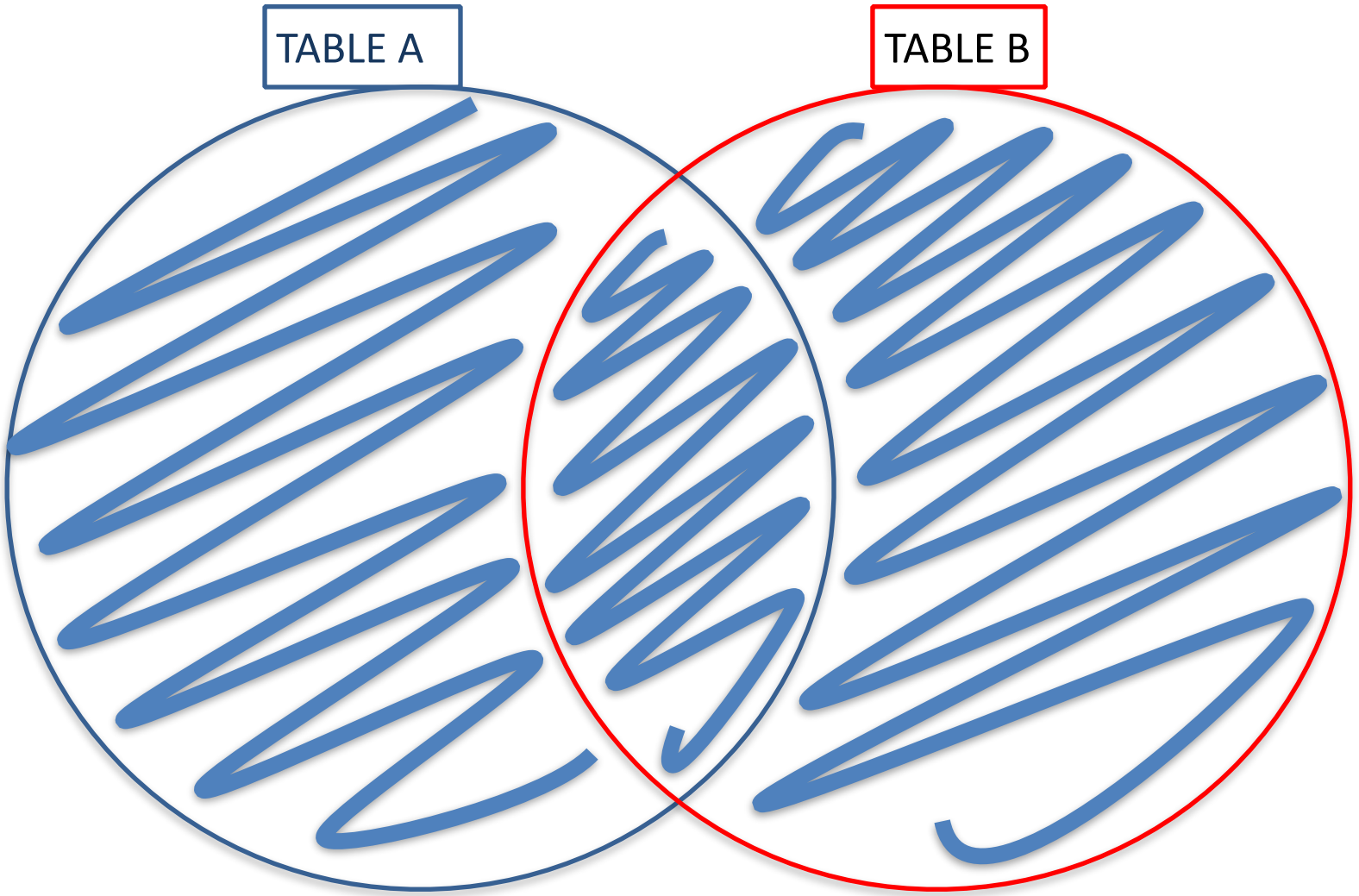
State	*City*
CA	San Francisco
WA	Tacoma

City	*Mascot*
San Francisco	49ers
New York	Jets

Results

State	*City*	*Mascot*
CA	San Francisco	49ers
	New York	Jets

OUTER JOIN



OUTER JOIN Example

State	*City*
CA	San Francisco
WA	Tacoma

City	*Mascot*
San Francisco	49ers
New York	Jets

Results

State	*City*	*Mascot*
CA	San Francisco	49ers
WA	Tacoma	
	New York	Jets

FUNCTIONS

- SELECT SUM(amount)
- SELECT MAX(release_year)
- SELECT MIN(length)
- SELECT AVG(price)

Executing Multiple Functions

```
SELECT
    MAX(Total),
    MIN(Total),
    MAX(Total) – MIN(Total) as range
FROM
    invoice
```

Exercise #6

1. What was the sales total for January 2010?

2. What is the average length of a song by R.E.M.?

Group By

- Technique for Aggregating Data
- Usually requires aggregation function in SELECT statement
- Similar to Pivot Tables

Group Example

Name	City
Bob	SF
Terry	SF
Joe	LA
Tina	NYC
Jen	NYC
John	NYC

What would you do if you wanted to Group by City

Group By

SF	LA	NYC
Bob	Joe	Tina
Terry		Jen
		John

Now how would you describe the amount for each City?
Sum, Count, Max, Minimum, Average?

GROUP BY Example

- For example:

```
SELECT Composer, COUNT(*)
FROM Track
WHERE Composer NOT NULL
GROUP BY Composer
ORDER BY COUNT(*) DESC
LIMIT 10;
```

Exercise #7

1. Which Artists have the most Tracks?
2. What is the Artist and Album Name for the album with the longest playing time?

ADDITIONAL TECHNIQUES

SUB QUERIES

- Use results of one query as an input to another query
- Treats the results of a query like a table
- Powerful, but can also add complexity
  - Less intuitive to read
  - Harder to trouble shoot when errors occur
- Build and test the subquery first

EXTRACT from Date

- Components of Dates can be Extracted
  - day, month, year
  - hour, minute, second
- SELECT EXTRACT(month from InvoiceDate) FROM Invoice;

CAST AS TYPE

- Function to convert data types
- Examples of uses:
  - Convert numbers to strings
  - Convert strings to dates
- Syntax:
  - SELECT CAST(Date AS text);

CASE STATEMENTS

- Similar to IF statements in Excel
- Create new values from existing data
- For example:
  - You have customer age data
  - Customers behave in age segments (e.g., kids, teens, adults, seniors)
  - CASE statements can be used to create categories for age ranges

Resources for Future Reference

- W3 Schools SQL Tutorials
  www.w3schools.com/sql/default.asp
- Tutorials Point
  tutorialspoint.com/sql/index.html