Back to Basics:
The SELECT Statement

Andrew Karcher

April 13th, 2019

Who are you?

- How many of you have never written a query before?
- How many of you have worked with a database before?
- How many of you have worked with SQL Server before?
- How many of you have done data analysis? With what?

- Regardless, of your answers, we are going to start with the basics
  - If you have no experience, hopefully this will give you the confidence to start
  - If you have experience, hopefully you will gain some nuggets of wisdom

Agenda

- Relational Database Structure
- ANSI Specification
- The SELECT Statement

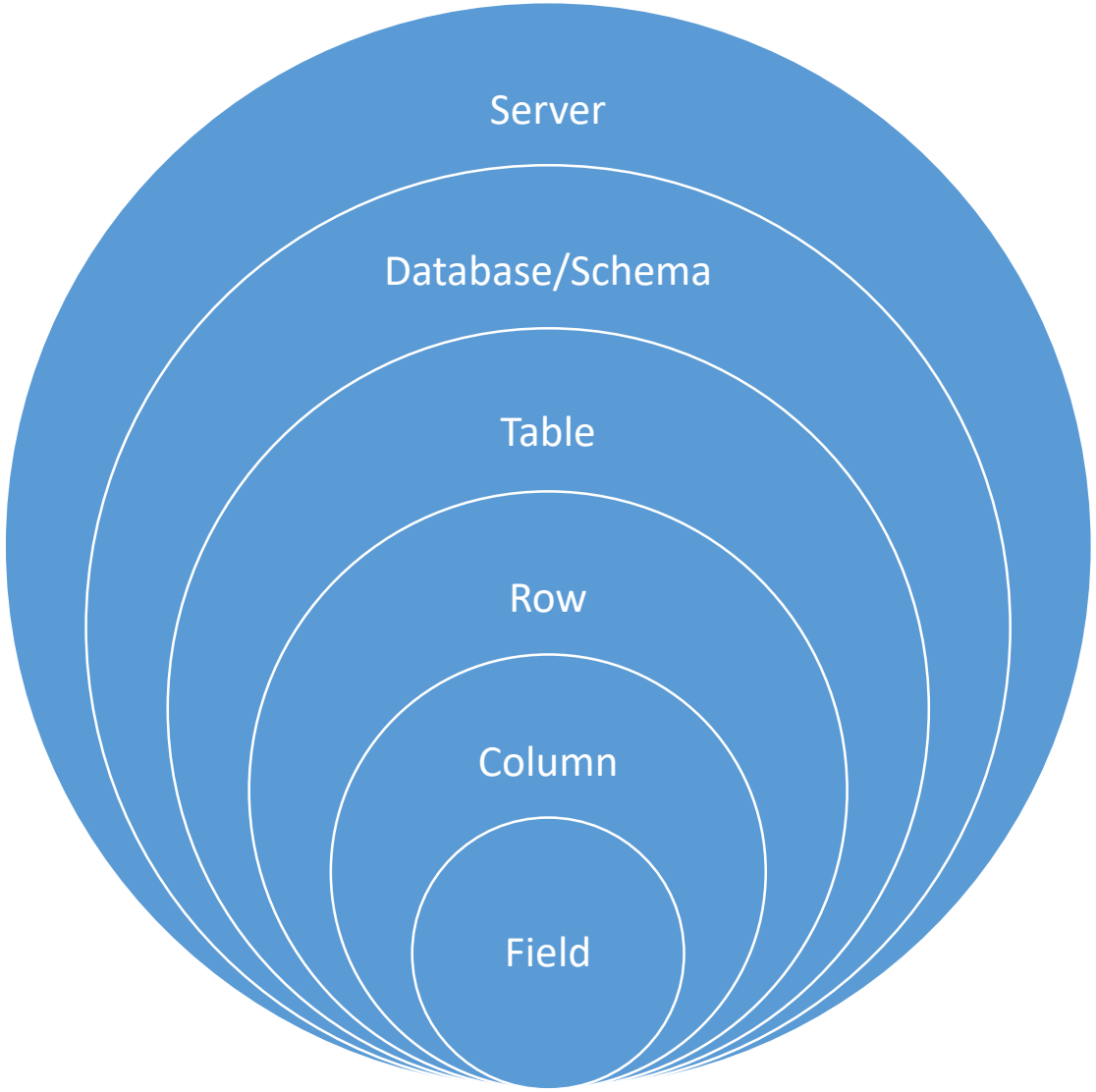
What is a Relational Database?

- Invented by E.F. Codd in 1970 at IBM
- Based on a set of 12 rules. However, most databases today do not follow all twelve rules
- The better known rules are the normalization rules also defined by Codd.
  - A high level the rules are designed to reduce data redundancy in the database
- Commercial vendors of relational databases include Microsoft, Oracle, DB2, MySQL, etc.
- SQL (Structured Query Language) is the language that is used to interface with the data in the database
  - SQL is the gateway to all relational databases and increasing most NoSQL databases as well.
  - Knowing good SQL is a very valuable skill

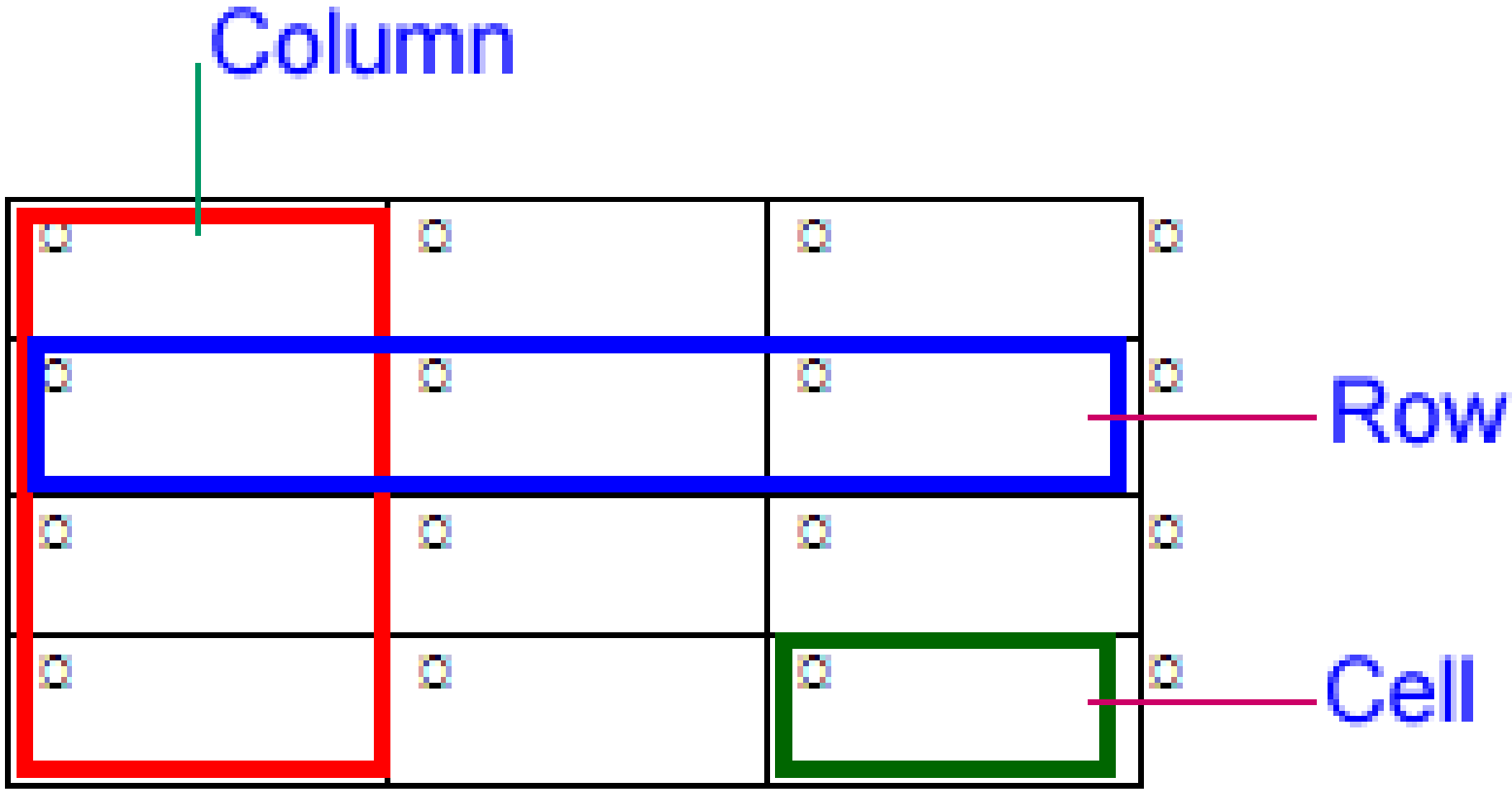
What is ANSI-SQL?

- SQL was originally developed at IBM in the 1970s
- First commercial product was sold in 1979
- First standard “Database Language SQL” was developed in 1986
- New versions in ’89. ‘91, ‘96, ’99, ‘03, ’06, ’08, ‘11, ’16
- Most major database engines strive for ANSI compliance but are not 100% due to factors such as keywords, existing functions, etc.
- Many of the statements and concepts are similar across data base engines.
- SQL is the primary language for querying data

What are the pieces of a Relational Database?



Table\Row\Column\Cell



Data Types

- Every column in a table has a defined data type
- A Data Type defines what type of data is allowed within a particular column.
- If you provide the wrong data type an insert or update will return an error
- SQL Server does do a lot of work for you when converting between two different data types.
- Choosing the right data type is important.
  - Choose too big and you could be using more storage than needed
  - Choose too small and you could run out of values or have errors trying to store values that are too large

Most Common SQL Server Data Types

- Numeric
  - Integer (TinyInt, SmallInt, Int, BigInt)
  - Decimal (precision, scale)
  - Float
- Character
  - Char(size)
  - Varchar(size -> Max)
  - Text
  - Nvarchar (size -> Max)
- Dates
  - Datetime
  - Datetime2
- Others
  - Bit
  - UniqueIdentifier

Working with NULLs

- NULL represents a missing or unknown value
- ANSI behaviour for NULL values:
  - The result of any expression containing a NULL value is NULL
    - 2 + NULL = NULL
    - ‘MyString: ‘ + NULL = NULL
  - Equality comparisons always return false for NULL values
    - NULL = NULL returns *false*
    - NULL **IS NULL** returns *true*

Working with NULLs
NULL Functions

- ISNULL(*column/variable, value*)
  - Returns *value* if the column or variable is NULL
- *NULLIF(column/variable*, *value*)
  - Returns NULL if the column or variable is *value*
- COALESCE (*column/variable1*, *column/variable2*,…)
  - Returns the value of the first non-NULL column or variable in the list

The SELECT Statement

Basic SELECT Statement

- Where are you getting your data from?
  - FROM
  - What Table is you data in?
- What fields do you need?
  - SELECT
  - What Columns do you want?
- Select statement returns the Rows

Adding the WHERE Clause

- The WHERE clause allows you to limit the number of rows returned by filtering on a condition.
- Results in a True\False that is evaluated on a row level basis
- WHERE clause is super powerful and is generally used in almost every query in some way
- Most typical way to use it is to filter by a specific value in a column
  - Can use arithmetic operators on numeric values
  - Can use functions on Text Values and other Data Types.

GROUP BY

- Used to Aggregate rows into a combined resultset
- Answers questions like:
  - How many products am I currently selling by SubCategoryID?
  - What is the average price by SubCategoryID
- Each Column returned needs to be an aggregate or a column in the GROUP BY clause
- Aggregate Functions
  - Min, Max, Count, Sum, Avg
- For even more functionality, check out the OVER Clause or search for Window Functions

ORDER BY

- Used to Order the returned rows in a set order
- Sorts columns in the Resultset in either Ascending (ASC) or Descending Order (DESC)
- ORDER BY is processed after GROUP BY

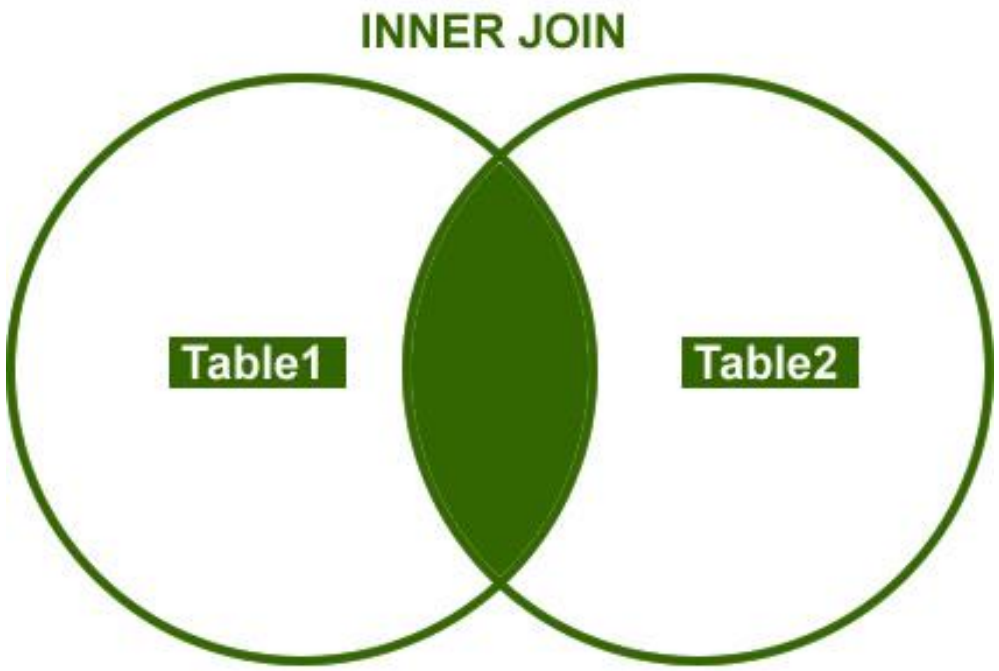
HAVING

- HAVING is typically used to filter the results of a GROUP BY statement.
- Can use it without a GROUP BY, but it functions as a WHERE clause
- Answers questions like:
  - Give me the ProductSubCategorys with a Average List Price over $500

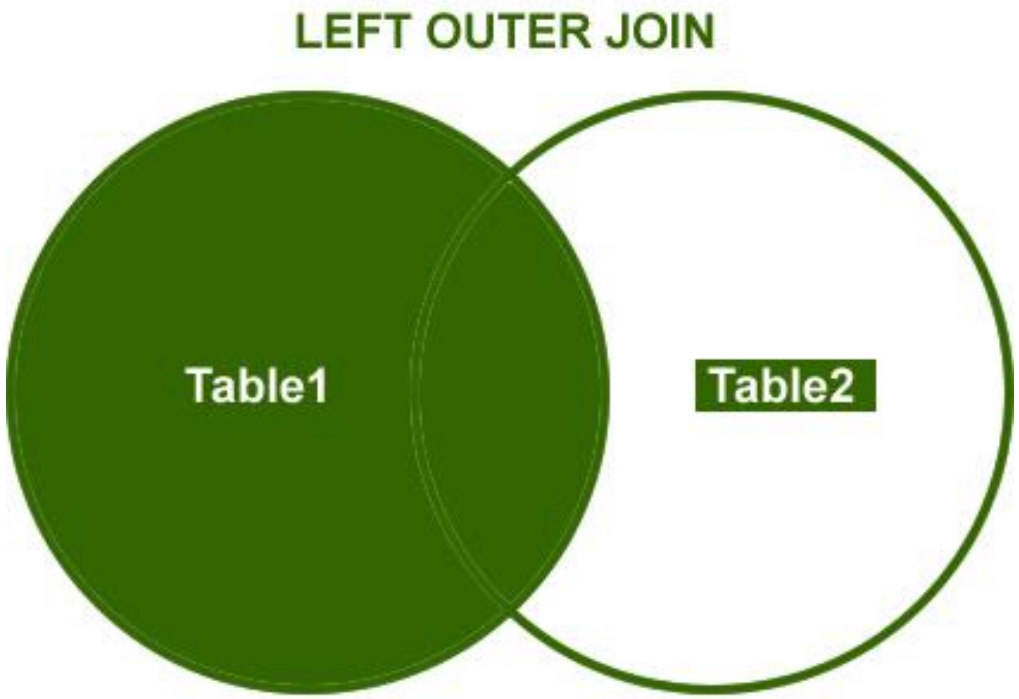
Joins

- In a relational database you are always going to have to Join data from one table to another
- Types of Joins
  - INNER JOIN
  - LEFT OUTER JOIN
  - RIGHT OUTER JOIN
  - FULL JOIN
  - CROSS JOIN

Inner Join

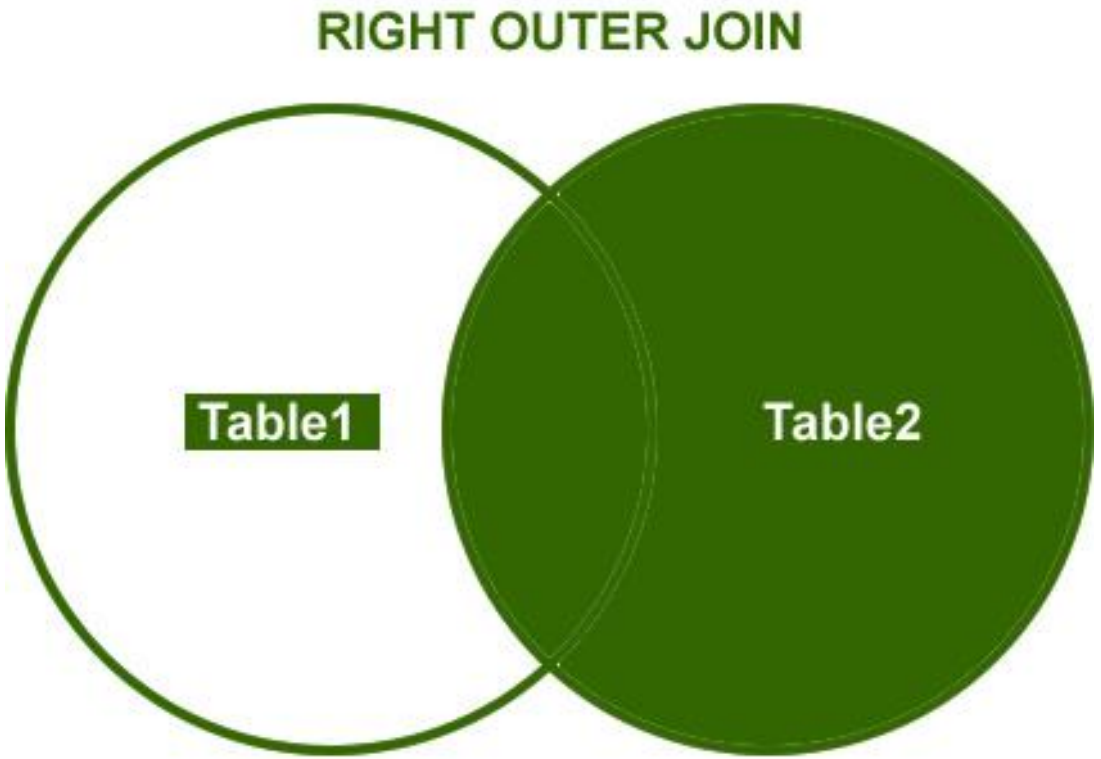

INNER JOIN
Table1
Table2
SELECT *
FROM Table1 t1
INNER JOIN Table2 t2
ON t1.Col1 = t2.Col1
(C) http://blog.SQLAuthority.com

LEFT OUTER JOIN



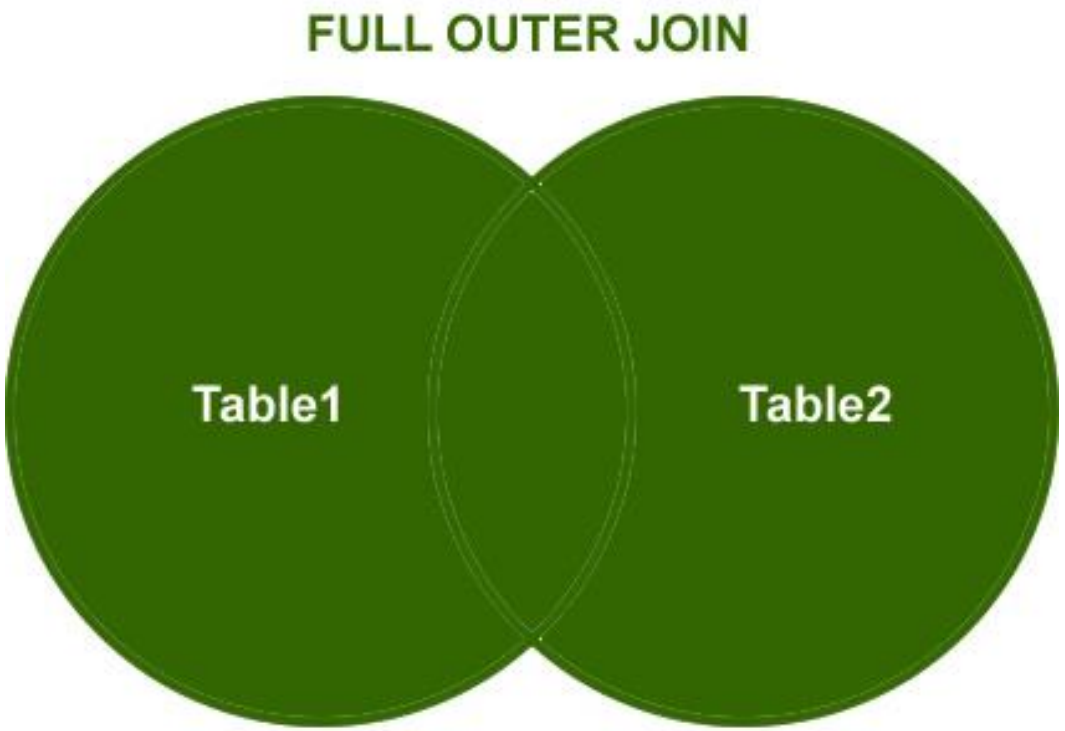

```
SELECT *
FROM Table1 t1
LEFT OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
```

RIGHT OUTER JOIN




```
SELECT *
FROM Table1 t1
RIGHT OUTER JOIN Table2 t2
        ON t1.Col1 = t2.Col1
```

FULL JOIN


FULL OUTER JOIN
Table1
Table2
SELECT *
FROM Table1 t1
FULL OUTER JOIN Table2 t2
ON t1.Col1 = t2.Col1
(C) http://blog.SQLAuthority.com

CROSS JOIN

	ID	Value
1	1	First
2	2	Second
3	3	Third
4	4	Fourth
5	5	Fifth

	ID	Value
1	1	First
2	2	Second
3	3	Third
4	6	Sixth
5	7	Seventh
6	8	Eighth

	ID	Value	ID	Value
1	1	First	1	First
2	1	First	2	Second
3	1	First	3	Third
4	1	First	6	Sixth
5	1	First	7	Seventh
6	1	First	8	Eighth
7	2	Second	1	First
8	2	Second	2	Second
9	2	Second	3	Third
10	2	Second	6	Sixth
11	2	Second	7	Seventh
12	2	Second	8	Eighth

Some Resources

- Code School – Try SQL – Free (https://www.pluralsight.com/courses/code-school-try-sql)
- T-SQL Fundamentals by Itzik Ben-Gan
- Pluralsight – Introduction to SQL (along with 50+ other SQL courses)
- Lots of Blogs
  - Pinal Dave – SQL Authority
  - Adam Machanic
  - Lots of others

About Me

- http://www.andrewkarcher.com
- Twitter: @akarcher
- LinkedIn, Twitter
- Email: akarcher@gmail.com
- Company – Data Architect, Becton Dickinson (BD)
- Organizer for SQL Saturday OC, President San Diego .Net User Group, Organizer – Data Engineering San Diego

Any Questions?

Questions?

akarcher@gmail.com

@akarcher

http://www.andrewkarcher.com