SQL Survival Guide: Joins

Andy Choens

EBCoP

June 17, 2015

Outline

Outline for section 1

SQL Survival Guide

- **Additional Presentation Resources:**
  - `https://github.com/choens/sql-survival-guide`
  - **SQL Queries:** "sql/04-joins/"
  - **Data Provided (SQLite):** "data/04-joins.sqlite"
- **Further Reading:**
  - `https://en.wikipedia.org/wiki/Join_(SQL)`
- Example data in SQLite: data/04-joins.sqlite.

Example Data

- **Two tables of example data:**
  - DEPARTMENTS
  - EMPLOYEES
- **SQL Queries:** "sql/04-joins/"
  - create_tables.sql: Creates the example tables.
  - Other files contain example queries.

Table: DEPARTMENTS

DEPT_ID	DEPT_NAME	DEPT_FLOOR
31	Sales	1
33	Engineering	3
34	Clerical	2
35	Marketing	3

Aliased as 'dept'.

Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER
1	31	Rafferty	Gerry	M
3	33	Jones	Jon	M
5	33	Heisenberg	Werner	M
7	34	Robinson	Elizabeth	F
9	34	Smith	Jefferson	M
11	NULL	Williams	Serena	F

Aliased as 'empl'.

Outline for section 2

Dafynitions

Why:

- We need a common set of terms to discuss SQL Joins.
- If I use a term you don't recognize, stop me.

Dafynition: Relational Data [1]

Relational Data:

- Based on the relational model, as proposed by E.F. Codd in 1970.
- Data is organized into one or more tables (relations) with a unique key for each row.
- Foreign keys make it possible to link rows in one table to rows in another table.
  - In the EMPLOYEES table, DEPT_ID is a foreign key.

[1] https://en.wikipedia.org/wiki/Relational_database

Dafynition: Clause[2]

The following protected words denote a SQL clause:

- SELECT
- FROM
- WHERE
- HAVING

[2] https://en.wikipedia.org/wiki/SQL#Language_elements

Dafynition: Clause[2]

The following protected words denote a SQL clause:

- SELECT
- FROM
- WHERE
- HAVING

Style:

- Align SQL Clauses
- Sub-clauses should be consistently indented.

[2] https://en.wikipedia.org/wiki/SQL#Language_elements

Dafynition: Join Clause[3]

```
1    from EMPLOYEES empl inner join DEPARTMENTS dept
2    on empl.dept_id = dept.dept_id
```

- The FIRST thing you should write.
- Combines records from two or more tables (result set).
- SQL Joins are a difficult skill to master.

[3] https://en.wikipedia.org/wiki/Join_(SQL)

Outline for section 3

Order of Operations: Math

Question: What is the answer to this equation?

$$\sqrt{(2^2 + 2) \cdot 6}$$

Order of Operations: Math

Question: What is the answer to this equation?

$$\sqrt{(2^2 + 2) \cdot 6}$$

Anwer: 6

Order of Operations: Math

Question: What is the answer to this equation?

$$\sqrt{(2^2 + 2) \cdot 6}$$

Anwer: 6

Question: How were we all able to come up with the same answer?

Order of Operations: Math

Question: What is the answer to this equation?

$$\sqrt{(2^2 + 2) \cdot 6}$$

Anwer: 6

Question: How were we all able to come up with the same answer?

Answer: Mathematical Order of Operations

Order of Operations: SQL

SQL Order Of Operations:

1. FROM
2. WHERE
3. GROUP BY
4. SELECT
5. HAVING
6. ORDER BY

Order of Operations: SQL

SQL Order Of Operations — Notes:

- Sub-queries are run before the outer query.
- The optimizer may reorganize query (relational algebra).
- Sometimes writing things out of order can bite you.
- Examples of such a problem is forthcoming.

Result Set

- The FIRST thing you should write.
- Combines records from two or more tables (result set).
- SQL Joins are a difficult skill to master.
- They are necessary for working with normalized, relational data.

Outline for section 4

Order of Operations

What is the answer to this (silly) equation?

$$\sqrt{(2^2 + 2) \cdot 6}$$

Order of Operations

What is the answer to this (silly) equation?

$$\sqrt{(2^2 + 2) \cdot 6}$$

And how were we all able to come up with the same answer?

Order of Operations

Just like math, SQL has an order of operations:

1. FROM
2. WHERE
3. GROUP BY
4. SELECT
5. HAVING
6. ORDER BY

Order of Operations

Some last order of operation notes:

- Sub-queries are run before the outer query.
- The optimizer may reorganize query (relational algebra).
- Sometimes writing things out of order can bite you.

Dafynition: Result Set[4]

SQL Output (Result Set):

- A set of rows from a database.
- Effectively a table.

[4] https://en.wikipedia.org/wiki/Result_set

Dafynition: Result Set[4]

SQL Output (Result Set):

- A set of rows from a database.
- Effectively a table.

Pro Tip: Imagine that each step in the order of operations produces a result set, and passes it to the next next clause.

[4] https://en.wikipedia.org/wiki/Result_set

Outline for section 5

Types of Joins

ANSI joins:

1. Cross Join
2. **Inner Join**
3. **Left Outer Join**
4. Right Outer Join
5. Full Outer Join

1. DO NOT USE!
2. **Very Useful.**
3. **Very Useful.**
4. Most of you should not use this.
5. You will rarely use this.

- Some of these are more useful than others.

Result Set Order

The No Particular Order Guarantee:

- Example queries and result sets provided for each join type.
- Result sets shown ordered by EID or DEPT_ID for readability.
- ANSI SQL does not guarantee the order of the result set, unless specified by an Order By clause.

Cross Join: Discussion

Cross Join:

- Returns the Cartesian product of the tables in the FROM clause.
- Incorrectly referred to as a Cartesian Join.
- Can be written explicitly and implicitly.
- Writing a Cross Join is **ALMOST ALWAYS** a bad idea.

Explicit Cross Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from DEPARTMENTS dept cross join EMPLOYEES empl
3    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/cross-join.sql

Explicit Cross Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from DEPARTMENTS dept cross join EMPLOYEES empl
3    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/cross-join.sql

Answer: 24 rows!

Cross Join: Example Result Set

Left Table: DEPARTMENTS

DEPT_ID	DEPT_NAME	DEPT_FLOOR	
31	Sales	1	→
31	Sales	1	→
31	Sales	1	→
31	Sales	1	→
31	Sales	1	→
31	Sales	1	→

Right Table: EMPLOYEES

	EID	DEPT_ID	LAST_NAME	FIRST_NAME
←	1	31	Rafferty	Gerry
←	3	33	Jones	Jon
←	5	33	Heisenberg	Werner
←	7	34	Robinson	Elizabeth
←	9	34	Smith	Jefferson
←	11	NULL	Williams	Serena

N Rows Returned: (N Rows Left Table) * (N Rows Right Table)

$$4 \cdot 6 = 24$$

Cross Join: Result Set

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER	DEPT_ID	DEPT_NAME	DEPT_FLOOR
1	31	Rafferty	Gerry	M	31	Sales	1
1	31	Rafferty	Gerry	M	33	Engineering	3
1	31	Rafferty	Gerry	M	34	Clerical	2
1	31	Rafferty	Gerry	M	35	Marketing	3
3	33	Jones	Jon	M	31	Sales	1
3	33	Jones	Jon	M	33	Engineering	3
3	33	Jones	Jon	M	34	Clerical	2
3	33	Jones	Jon	M	35	Marketing	3
5	33	Heisenberg	Werner	M	31	Sales	1
5	33	Heisenberg	Werner	M	33	Engineering	3
5	33	Heisenberg	Werner	M	34	Clerical	2
5	33	Heisenberg	Werner	M	35	Marketing	3
7	34	Robinson	Elizabeth	F	31	Sales	1
7	34	Robinson	Elizabeth	F	33	Engineering	3
7	34	Robinson	Elizabeth	F	34	Clerical	2
7	34	Robinson	Elizabeth	F	35	Marketing	3
9	34	Smith	Jefferson	M	31	Sales	1
9	34	Smith	Jefferson	M	33	Engineering	3
9	34	Smith	Jefferson	M	34	Clerical	2
9	34	Smith	Jefferson	M	35	Marketing	3
11	[NULL]	Williams	Serena	F	31	Sales	1
11	[NULL]	Williams	Serena	F	33	Engineering	3
11	[NULL]	Williams	Serena	F	34	Clerical	2
11	[NULL]	Williams	Serena	F	35	Marketing	3

Implicit Cross Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl, DEPARTMENTS dept
3    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/cross-join.sql

Cross Join: Results

Cross Join:

- Cross Join == Danger!
- Returns the maximum number of rows possible.
- I have never written a Cross Join outside of a training environment.
- You probably won't either.

Inner Join: Discussion

Inner Join:

- Returns all records which have matching records in both tables, according to the Join clause or WHERE clause.
- Can be written explicitly or implicitly.
- Returns the least number of rows.

Explicit Inner Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl inner join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/cross-join.sql

Explicit Inner Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl inner join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/cross-join.sql

Answer: 5 rows

Inner Join: Example Result Set (1)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	
1	31	Rafferty	Gerry	→

Right Table: DEPARTMENTS

	DEPT_ID	DEPT_NAME	DEPT_FLOOR
←	31	Sales	1

Result Set:

- Includes all rows in both tables which match via the 'on' statement.

Inner Join: Example Result Set (2)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	
3	33	Jones	Jon	→
5	33	Heisenberg	Werner	→

Right Table: DEPARTMENTS

	DEPT_ID	DEPT_NAME	DEPT_FLOOR
←	33	Engineering	1
←	33	Engineering	1

Result Set:

- Dept 33 returned twice, because it matches two employees.

Inner Join: Result Set

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER	DEPT_ID	DEPT_NAME	DEPT_FLOOR
1	31	Rafferty	Gerry	M	31	Sales	1
3	33	Jones	Jon	M	33	Engineering	3
5	33	Heisenberg	Werner	M	33	Engineering	3
7	34	Robinson	Elizabeth	F	34	Clerical	2
9	34	Smith	Jefferson	M	34	Clerical	2

Question: What happened to Dept 35?

Inner Join: Result Set

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER	DEPT_ID	DEPT_NAME	DEPT_FLOOR
1	31	Rafferty	Gerry	M	31	Sales	1
3	33	Jones	Jon	M	33	Engineering	3
5	33	Heisenberg	Werner	M	33	Engineering	3
7	34	Robinson	Elizabeth	F	34	Clerical	2
9	34	Smith	Jefferson	M	34	Clerical	2

Question: What happened to Dept 35?

Answer: Dropped because there aren't any employees in Marketing.

Implicit Inner Join: Example

This should look familiar:

```
1    select *
2    from EMPLOYEES empl, DEPARTMENTS dept
3    where empl.dept_id = dept.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/cross-join.sql

Result Set:

- Also returns 5 rows.

Why Not Use Implicit Join Syntax?

Question: What's the difference?

- Implicit Cross Join
- Implicit Inner Join

Why Not Use Implicit Join Syntax?

Question: What's the difference?

- Implicit Cross Join
- Implicit Inner Join

Answer: The WHERE clause.

- An Implicit Inner Join is just an optimized Cross Join!
- SQL optimizers convert it to explicit Inner Join.
- If followed explicitly it could be S.L.O.W.
- Implicit Join syntax is deprecated. (Geek-speak for don't do it.)
- Too easy to write a Cross Join.

Other Inner Joins

We have to move on but:

- Inner Joins are really useful.
- See inner-join.sql for more examples and ways to write Inner Joins.

Outer Join

Outer Join:

- The result set from Outer Join retains more records than Inner Join.
- Types of Outer Join:
  - Left Outer Join (Left Join)
  - Right Outer Join (Right Join)
  - Full Outer Join
- Each has different rules for which rows are part of the result set.

Left Outer Join: Discussion

Left Outer Join:

- Result set includes all members of the 'left' table.
- When a row in the left table does not match any row in the right table, there will be NULLS in all columns from the right table.
- Left Join is the same as Left Outer Join

Left Outer Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl left join DEPARTMENTS dept
3    on empl.department_id = dept_dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

Note:

- EMPLOYEES is the Left Table.
- DEPARTMENTS is the Right Table.
- I really hope you can see why.

Left Outer Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl left join DEPARTMENTS dept
3    on empl.department_id = dept_dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

Answer: 6 rows

Left Outer Join: Example Result Set (1)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	
1	31	Rafferty	Gerry	→

Right Table: DEPARTMENTS

DEPT_ID	DEPT_NAME	DEPT_FLOOR
31	Sales	1

Result: For matched rows, the result set is identical to Inner Join.

Left Outer Join: Example Result Set (2)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	
11	NULL	Williams	Serena	→

Right Table: DEPARTMENTS

DEPT_ID	DEPT_NAME	DEPT_FLOOR
NULL	NULL	NULL

Result: An Inner Join would have dropped this row.

Left Outer Join: Result Set

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER	DEPT_ID	DEPT_NAME	DEPT_FLOOR
1	31	Rafferty	Gerry	M	31	Sales	1
3	33	Jones	Jon	M	33	Engineering	3
5	33	Heisenberg	Werner	M	33	Engineering	3
7	34	Robinson	Elizabeth	F	34	Clerical	2
9	34	Smith	Jefferson	M	34	Clerical	2
11	NULL	Williams	Serena	NULL	NULL	NULL	NULL

Left Outer Join: DANGER!

Question: How many rows will the following queries return?
Hint: They return the same number of rows.

```
1    select *
2    from EMPLOYEES empl left join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    where dept.dept_id < 34
5    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

```
1    select *
2    from EMPLOYEES empl, DEPARTMENTS dept
3    where
4        empl.dept_id = dept.dept_id
5        and dept.dept_id < 34
6    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

Left Outer Join: DANGER!

Answer: Only 3 rows.

Left Outer Join: DANGER!

Question: Why doesn't this return 6 rows?

```
1    select *
2    from EMPLOYEES empl left join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    where dept.dept_id < 34
5    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

Left Outer Join: DANGER!

Question: Why doesn't this return 6 rows?

```
1    select *
2    from EMPLOYEES empl left join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    where dept.dept_id < 34
5    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

Answer: SQL Order of Operations!

- The FROM clause returns 6 rows.
- The WHERE clause runs AFTER the FROM clause, and drops 3 rows.
- End Result: 3 rows returned.

Left Outer Join: Danger!

```
1   select *
2   from (
3         -- This sub-query returns 6 rows.
4         select *
5         from EMPLOYEES empl left join DEPARTMENTS dept
6         on empl.dept_id = dept.dept_id
7       ) src0
8   -- But then ommits three of those rows from the final
        result set.
9   where src0.dept_id < 34
10  ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

This highlights what the former query is doing.
It is easier to understand why this only returns 3 rows.

Left Outer Join: DANGER!

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl left join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4       and dept.dept_id < 34
5    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

Left Outer Join: DANGER!

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl left join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4       and dept.dept_id < 34
5    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/left-join.sql

Answer: 6 rows

- Moved the filter statement to the FROM clause.
- It is OK to have a WHERE clause that refers to the columns in the Left Table (EMPLOYEES).

Right Outer Join: Discussion

Right Outer Join:

- Basically a left join with the table order reversed.
- When a row in the right table does not match any row in the left table, there will be NULLS in all columns from the left table.
- Right Join is the same as Right Outer Join
- Speakers of LTR languages tend to prefer Left Joins.
- Speakers of RTL languages often prefer Right Joins.

Right Outer Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl right join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/right-join.sql

Note: Not supported by some SQL implementations such as SQLite.

Right Outer Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl right join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/right-join.sql

Note: Not supported by some SQL implementations such as SQLite.

Answer: 6 rows

Right Outer Join: Example Result Set (1)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME
1	31	Rafferty	Gerry

Right Table: DEPARTMENTS

	DEPT_ID	DEPT_NAME	DEPT_FLOOR
←	31	Sales	1

Result: For matched rows, the result set is the same as Inner Join.

Right Outer Join: Example Result Set (2)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME
NULL	NULL	NULL	NULL

Right Table: DEPARTMENTS

	DEPT_ID	DEPT_NAME	DEPT_FLOOR
←	35	Marketing	3

Result:

- The Right Join keeps Dept 35 in the result set.
- The same thing can be written as a Left Join:

```
1    select *
2    from DEPARTMENTS dept left join EMPLOYEES empl
3    on dept.dept_id = empl.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/right-join.sql

Right Outer Join: Result Set

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER	DEPT_ID	DEPT_NAME	DEPT_FLOOR
1	31	Rafferty	Gerry	M	31	Sales	1
3	33	Jones	Jon	M	33	Engineering	3
5	33	Heisenberg	Werner	M	33	Engineering	3
7	34	Robinson	Elizabeth	F	34	Clerical	2
9	34	Smith	Jefferson	M	34	Clerical	2
NULL	NULL	NULL	NULL	NULL	35	Marketing	3

Question: What happened to Serena Williams?

Right Outer Join: Result Set

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER	DEPT_ID	DEPT_NAME	DEPT_FLOOR
1	31	Rafferty	Gerry	M	31	Sales	1
3	33	Jones	Jon	M	33	Engineering	3
5	33	Heisenberg	Werner	M	33	Engineering	3
7	34	Robinson	Elizabeth	F	34	Clerical	2
9	34	Smith	Jefferson	M	34	Clerical	2
NULL	NULL	NULL	NULL	NULL	35	Marketing	3

Question: What happened to Serena Williams?

Answer: Because DEPT_ID is NULL, she is not in the result set.

Right Outer Join: Danger!

Danger Will Robinson! Danger!

- Right Outer Joins are subject to the same dangers as Left Outer Joins.
- Conceptualizing Right Joins can be hard for native English speakers.
- Using left and right joins in the same query is TROUBLE.

Full Outer Join: Discussion

Full Outer Join:

- Combines the effect of applying both left and right outer joins.
- Result set includes all rows from both tables, at least once.
- Where rows do not match, the result set will have NULL values for every column of the table that lacks a matching row
- Not frequently used.
- Can be used to understand why a result set is smaller than expected.

Full Outer Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl full outer join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/full-outer-join.sql

Note: Not supported by some SQL implementations such as SQLite.

Full Outer Join: Example

Question: How many rows will the following query return?

```
1    select *
2    from EMPLOYEES empl full outer join DEPARTMENTS dept
3    on empl.dept_id = dept.dept_id
4    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/full-outer-join.sql

Note: Not supported by some SQL implementations such as SQLite.

Answer: 7 rows.

Full Outer Join: Example Result Set (1)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	
1	31	Rafferty	Gerry	→

Right Table: DEPARTMENTS

	DEPT_ID	DEPT_NAME	DEPT_FLOOR
←	31	Sales	1

Result:

- For matched rows, result set is same as Inner Join.

Full Outer Join: Example Result Set (2)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	
11	NULL	Williams	Serena	→

Right Table: DEPARTMENTS

	DEPT_ID	DEPT_NAME	DEPT_FLOOR
←	NULL	NULL	NULL

Result:

- For unmatched rows: KEEP THEM ALL!

Full Outer Join: Example Result Set (3)

Left Table: EMPLOYEES

EID	DEPT_ID	LAST_NAME	FIRST_NAME	
NULL	NULL	NULL	NULL	→

Right Table: DEPARTMENTS

	DEPT_ID	DEPT_NAME	DEPT_FLOOR
←	35	Marketing	3

Result:

- For unmatched rows: KEEP THEM ALL!

Full Outer Join: Result Set

EID	DEPT_ID	LAST_NAME	FIRST_NAME	GENDER	DEPT_ID	DEPT_NAME	DEPT_FLOOR
1	31	Rafferty	Gerry	M	31	Sales	1
3	33	Jones	Jon	M	33	Engineering	3
5	33	Heisenberg	Werner	M	33	Engineering	3
7	34	Robinson	Elizabeth	F	34	Clerical	2
9	34	Smith	Jefferson	M	34	Clerical	2
11	NULL	Williams	Serena	NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL	NULL	35	Marketing	3

Outline for section 6

Self Join: Discussion

Self Join:

- Joins a table to itself.
- To do so, use one of the Joins we have already discussed.
- This is more useful than you might think.
- Often used in conjunction with a subquery.
- To demo, we will need a new table.

Table: SALES

SALE_ID	EID	SALE_VAL	SALE_DT
1	1	15.00	2015-01-01
2	1	30.12	2015-06-15
3	1	45.79	2015-03-02

Aliased as 'sale'.

Note: Query to create this table is not in create-tables.sql (yet).

Self Join: Example

Question: How many rows will the following query return?

```
1     select sale.eid, sale.eid, sale.sale_val, sale_dt
2     from SALES sale
3     inner join (
4                 -- Last Sale (ls)
5                 -- This query runs BEFORE the Join Clause.
6                 select eid, max(sale_dt) max_dt
7                 from SALES
8                 group by eid
9               ) ls
10    on sale.eid = ls.eid and sale.sale_dt = ls.max_dt
11    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/self-join.sql

Self Join: Example

Question: How many rows will the following query return?

```
1     select sale.eid, sale.eid, sale.sale_val, sale_dt
2     from SALES sale
3     inner join (
4                 -- Last Sale (ls)
5                 -- This query runs BEFORE the Join Clause.
6                 select eid, max(sale_dt) max_dt
7                 from SALES
8                 group by eid
9               ) ls
10    on sale.eid = ls.eid and sale.sale_dt = ls.max_dt
11    ;
```

Source: https://github.com/Choens/sql-survival-guide/blob/master/sql/04-joins/self-join.sql

Answer: Only 1 row.

Self Join: Example Result Set

Left Table: SALES

SALE_ID	EID	SALE_VAL	SALE_DT	
1	1	15.00	2015-01-01	→

Right Table: LAST SALE

	EID	MAX_DT
←	1	2015-06-15

Result:

- The sub-query (LAST SALE) is run first. It returns 1 row.
- Inner Join returns matching rows between SALE and LAST SALE.
- Each copy of a table must have a unique alias.
- A subquery is not required, but is common.

Self Outer Join: Result Set

SALE_ID	EID	SALE_VAL	SALE_DT
2	1	30.12	2015-06-15

Outline for section 7

Aggregation: How To

Separate your thoughts into two steps.:

1. FROM
2. GROUP BY

Pro Tip: Treat each step in the SQL Order of Operations as though it has an independent result set.

Outline for section 8

Relational Data: How To

Working with relational data is harder than working with flat data.

Some things to consider:

- What is the 'unit' of your table(s)?
- Valid data must have a unique identifier for each row in a table.
  - People / Employees / Medicaid Recipients
  - Departments / Providers
  - Events / MDC Codes / Medications
- The unique identifier changes across tables.
  - One to One
  - One to Many
  - To use, you must understand how the tables are related.

Relational Data: Practical How To

Assume you are joining two tables (Table A, Table B):

Table A:

- Count the number of total records in Table A.
- Count the number of distinct unique identifiers in Table A.
- If they don't match. That isn't the unique identifier.
- Look for Nulls in any column you are going to filter, sort or join by.

Table B:

- Count the number of total records in Table B.
- Count the number of distinct unique identifiers in Table B.
- If they don't match. That isn't the unique identifier.
- Look for Nulls in any column you are going to filter, sort or join by.

Approach Relational Data Like a Scientist (1)

Don't Assume, Prove:

- You believe Table A has a one-to-many-relationship with Table B.
- Don't assume you know what is going on.
- We are scientists.
- We have tools for this.

Approach Relational Data Like a Scientist (2)

Science Tools For SQL:

- Hypothesis: Every row in Table A matches one (+) rows in Table B.
- Test your hypothesis. Is it true?
- Are you dropping any rows from A?
- Are you duplicating unexpectedly?

Pro Tip: Build a complex query piece by piece! (Test each piece.)

Outline for section 9

Additional Information

- https://en.wikipedia.org/wiki/Join_(SQL)

Questions?

That was a lot of information. And there is a lot of text silly text on this slide. If you are still reading this, congrats. You must be wide awake by now. And maybe, you are wondering why I wrote all of this.

Who has questions?