Introduction to RDM and SQL

Day 2

Day 2

- The FROM Clause
  - Table expressions
- The WHERE Clause
  - Ternary Logic
  - Dealing with NULLs
  - Using predicates

```
      6
5 SELECT [DISTINCT] Customer,
                    COUNT(*) AS Count
1 FROM    Orders
2 WHERE   OrderDate > '19650101'
3 GROUP BY    Customer
4 HAVING  COUNT(*) > 1
7 ORDER BY Count DESC
8 OFFSET 0 ROWS FETCH NEXT 10 ROWS ONLY;
```

Order ID	Customer	Order Date
1	Dave	1/1/2018
2	John	2/1/2018
3	Gerald	3/1/2018
4	John	9/1/2018

The FROM Clause

“Hello World!”

```
SELECT * | <Expression> [AS <Alias>];

SELECT * | <Expression> [AS <Alias>]
FROM   <Table Expression>;
```

Table Expressions

The counterpart of the RDM relation. Therefore, they are a set.

What are the properties of a relation?

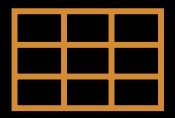
What does it represent in the RDM?

- ✓ A relational data model is a model of reality
- ✓ Every relation represents one thing from the real world
- ✓ A tuple is an instance of that thing **and** is uniquely identifiable
- ✓ Every attribute describes an inherent property of that thing
- ✓ Identifying attributes distinguish one thing from another
- ✓ Every fact is represented in one place in the database, and always as an intersection of an attribute and a tuple
- ✓ A SQL database approximates the relational data model

Join Syntax

```
<Table Expression 1>
{CROSS|INNER|[LEFT|RIGHT|FULL] OUTER} JOIN
<Table Expression 2>
[ON <Qualification Predicate>]
```

Join Processing



1. Cartesian product

2. Qualification

3. Reservation

Cartesian Product

Customers

CROSS JOIN Orders

Customer	Country
Dave	USA
John	USA
Gerald	Canada
Jose	Peru
Tim	*NULL*

X

Order ID	Customer	Order Date
1	Dave	1/1/2018
2	John	2/1/2018
3	Gerald	3/1/2018
4	John	9/1/2018

Customer	Country	Order ID	Customer	Order Date
Dave	USA	1	Dave	1/1/2018
Dave	USA	2	John	2/1/2018
Dave	USA	3	Gerald	3/1/2018
Dave	USA	4	John	9/1/2018
John	USA	1	Dave	1/1/2018
John	USA	2	John	2/1/2018
John	USA	3	Gerald	3/1/2018
John	USA	4	John	9/1/2018
Gerald	Canada	1	Dave	1/1/2018
Gerald	Canada	2	John	2/1/2018
Gerald	Canada	3	Gerald	3/1/2018
Gerald	Canada	4	John	9/1/2018
Jose	Peru	1	Dave	1/1/2018
Jose	Peru	2	John	2/1/2018
Jose	Peru	3	Gerald	3/1/2018
Jose	Peru	4	John	9/1/2018
Tim	*NULL*	1	Dave	1/1/2018
Tim	*NULL*	2	John	2/1/2018
Tim	*NULL*	3	Gerald	3/1/2018
Tim	*NULL*	4	John	9/1/2018

Qualification

```
Customers INNER JOIN Orders
ON   Customers.Customer =
     Orders.Customer
```

Customer	Country	Order ID	Customer	Order Date
Dave	USA	1	Dave	1/1/2018
Dave	USA	2	John	2/1/2018
Dave	USA	3	Gerald	3/1/2018
Dave	USA	4	John	9/1/2018
John	USA	1	Dave	1/1/2018
John	USA	2	John	2/1/2018
John	USA	3	Gerald	3/1/2018
John	USA	4	John	9/1/2018
Gerald	Canada	1	Dave	1/1/2018
Gerald	Canada	2	John	2/1/2018
Gerald	Canada	3	Gerald	3/1/2018
Gerald	Canada	4	John	9/1/2018
Jose	Peru	1	Dave	1/1/2018
Jose	Peru	2	John	2/1/2018
Jose	Peru	3	Gerald	3/1/2018
Jose	Peru	4	John	9/1/2018
Tim	*NULL*	1	Dave	1/1/2018
Tim	*NULL*	2	John	2/1/2018
Tim	*NULL*	3	Gerald	3/1/2018
Tim	*NULL*	4	John	9/1/2018

Reservation

```
Customers LEFT OUTER JOIN
Orders
ON   Customers.Customer =
     Orders.Customer
```

Customer	Country
Dave	USA
John	USA
Gerald	Canada
Jose	Peru
Tim	*NULL*

Customer	Country	Order ID	Customer	Order Date
Dave	USA	1	Dave	1/1/2018
John	USA	2	John	2/1/2018
John	USA	4	John	9/1/2018
Gerald	Canada	3	Gerald	3/1/2018
Jose	Peru	*NULL*	*NULL*	*NULL*
Tim	*NULL*	*NULL*	*NULL*	*NULL*

Chiastic Order

“When the going gets tough, the tough gets going”

The FROM Clause

The WHERE Clause

```
                6
5 SELECT [DISTINCT] Customer,
                    COUNT(*) AS Count
1 FROM    Orders
2 WHERE   OrderDate > '19650101'
3 GROUP BY    Customer
4 HAVING  COUNT(*) > 1
7 ORDER BY NumOrders DESC
8 OFFSET 0 ROWS FETCH NEXT 10 ROWS ONLY;
```

Order ID	Customer	Order Date
1	Dave	1/1/2018
2	John	2/1/2018
3	Gerald	3/1/2018
4	John	9/1/2018

ANSI 89 JOIN Syntax

```
SELECT  <Expressions>
FROM    <Table Expression 1>,
        <Table  Expression 2>,
        <Table Expression 3>
WHERE   <Qualification Predicates>
```

```
Customers LEFT OUTER JOIN Orders
ON   Customers.Customer = Orders.Customer
     AND Orders.OrderDate > ‘20180101’

Customers LEFT OUTER JOIN Orders
ON   Customers.Customer = Orders.Customer
WHERE Orders.OrderDate > ‘20180101’
```

“All at Once”

```
SELECT  Customer AS Client,
        Client+‘@’+Country AS ExtendedName
FROM    Customers;

UPDATE  Table
SET     Column1 = Column2,
        Column2 = Column1;
```


NULL

NULL

NULL is a special marker used in Structured Query Language to indicate that a data value does not exist in the database. Introduced by … E. F. Codd, SQL NULL serves to fulfil the requirement that all true relational database management systems (RDBMS) support a representation of "missing information and inapplicable information".

https://en.wikipedia.org/wiki/Null_(SQL)

Ternary Logic Evaluation

a	b	a OR b	a AND b	a = b	NOT a
True	True	True	True	True	False
True	False	True	False	False	False
True	Unknown	True	Unknown	Unknown	False
False	True	True	False	False	True
False	False	False	False	True	True
False	Unknown	Unknown	False	Unknown	True
Unknown	True	True	Unknown	Unknown	Unknown
Unknown	False	Unknown	False	Unknown	Unknown
Unknown	Unknown	Unknown	Unknown	Unknown	Unknown

NULL Infestations

- Default when not explicitly specified
- “De-normalization”
- Generalizations
- NULL alternatives

The Controversies…

“The SQL implementation of NULL is flawed.”

- Dr. Edgar F. Codd

“SQL Null implementation is so inherently flawed and should be eliminated altogether.”

- C. Date & H. Darwen

A NULLess Schema

- Darwen, “How To Handle Missing Information Without Using Nulls”
- F. Pascal, “The Final NULL in the Coffin”

NULL Types

A-Values

"The value exists, we just don't know it."

I-Values

"This attribute is irrelevant for this tuple."

NULL is an indicator to the absence of a value.

It is a state, It is not a value!

WHERE

- Ternary Logic - TRUE, FALSE, UNKNOWN
- WHERE eliminates rows for which the predicate does not evaluate to TRUE
- NULL comparison is always UNKNOWN
- IS NULL is a state predicate, not a comparison

Common Predicates

IS [NOT] NULL

IS [NOT] DISTINCT FROM

IS [TRUE|FALSE|UNKNOWN]

IN

BETWEEN

LIKE

The FROM Clause

Next week:

```
GROUP BY
HAVING
SELECT
ORDER BY
```