

# Sub-queries

IMPROVING QUERY PERFORMANCE IN SQL SERVER



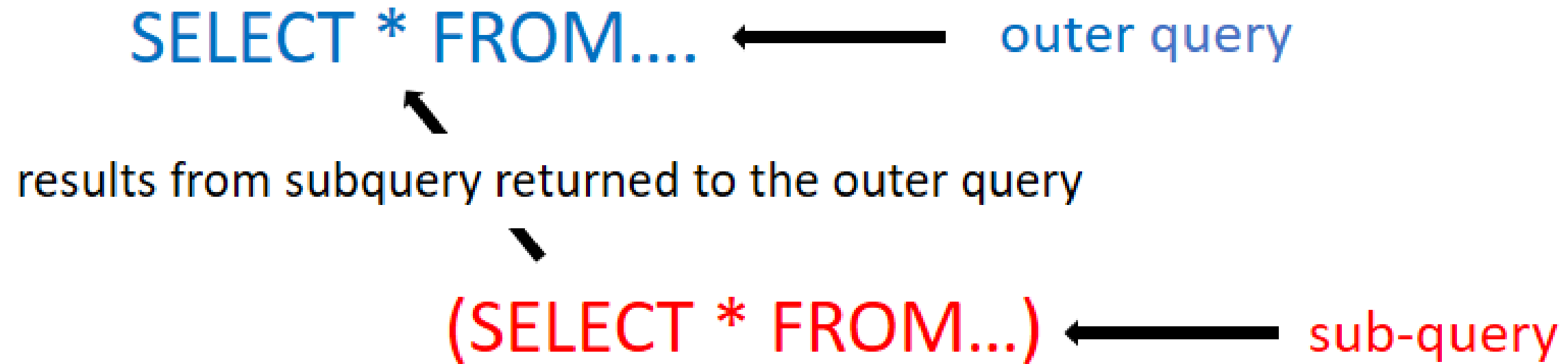
**Dean Smith**

Founder, Atamai Analytics

# How do sub-queries look?

(SELECT \* FROM...) ← sub-query

# How do sub-queries look?



# Sub-query with FROM

```
SELECT OrderID,  
       CustomerID,  
       NumDays  
FROM  
      (SELECT *,  
       DATEDIFF(DAY,OrderDate,ShippedDate) AS NumDays  
       FROM Orders) AS o  
WHERE NumDays >= 35;
```

# Sub-query with FROM

```
SELECT OrderID,  
       CustomerID,  
       NumDays  
FROM  
  (SELECT *,  
   DATEDIFF(DAY, OrderDate, ShippedDate) AS NumDays  
   FROM Orders) AS o  
WHERE NumDays >= 35;
```

OrderID	CustomerID	NumDays
10380	HUNGO	35
10427	PICCO	35
10545	LAZYK	35
10593	LEHMS	35
10660	HUNGC	37
10777	GOURL	37
10924	BERGS	35

# Sub-query with WHERE

```
SELECT CustomerID  
       ,CompanyName  
FROM Customers  
WHERE CustomerID  
       IN (SELECT CustomerID  
           FROM Orders  
           WHERE Freight > 800);
```

# Sub-query with WHERE

```
SELECT CustomerID
       ,CompanyName
FROM Customers
WHERE CustomerID
      IN (SELECT CustomerID
          FROM Orders
          WHERE Freight > 800);
```

CustomerID	CompanyName
QUEEN	Queen Cozinha
QUICK	QUICK-Stop
SAVEA	Save-a-lot Markets

# Sub-query with SELECT

```
SELECT CustomerID,  
       CompanyName,  
       (SELECT AVG(Freight)  
        FROM Orders o  
        WHERE c.CustomerID = o.CustomerID) AS AvgFreight  
FROM Customers c;
```



# Sub-query with SELECT

```
SELECT CustomerID,  
       CompanyName,  
       (SELECT AVG(Freight)  
        FROM Orders o  
        WHERE c.CustomerID = o.CustomerID) AS AvgFreight  
FROM Customers c;
```

CustomerID	CompanyName	AvgFreight
ALFKI	Alfreds Futterkiste	37.6
ANATR	Ana Trujillo Emparedados y helados	24.4
ANTON	Antonio Moreno Taquería	38.4
...	...	...

# Types of sub-queries

## Uncorrelated sub-query

```
SELECT CustomerID
       ,CompanyName
FROM Customers
WHERE CustomerID IN
      (SELECT CustomerID
       FROM Orders
       WHERE Freight > 800);
```

- Sub-query *does not* contain a reference to the outer query
- Sub-query *can* run independently of the outer query
- Used with `WHERE` and `FROM`

## Correlated sub-query

```
SELECT CustomerID,
       CompanyName,
       (SELECT AVG(Freight)
        FROM Orders o
        WHERE c.CustomerID = o.CustomerID) AS AvgFreight
FROM Customers c;
```

- Sub-query *contains* a reference to the outer query
- Sub-query *cannot* run independently of the outer query
- Used with `WHERE` and `SELECT`

# Sub-query performance

## Correlated

- Sub-query executes for each row in the outer query

## Uncorrelated

- Sub-query executes only once and returns the results to the outer query

# Sub-query vs. INNER JOIN

## Correlated sub-query

```
SELECT CustomerID,  
       CompanyName,  
       (SELECT AVG(Freight)  
        FROM Orders o  
        WHERE c.CustomerID = o.CustomerID) AS AvgFreight  
FROM Customers c;
```

## INNER JOIN

```
SELECT c.CustomerID,  
       c.CompanyName,  
       AVG(o.Freight)  
FROM Customers c  
INNER JOIN Orders o  
      ON c.CustomerID = o.CustomerID  
GROUP BY c.CustomerID,  
         c.CompanyName;
```

# Let's practice!

IMPROVING QUERY PERFORMANCE IN SQL SERVER

# Presence and absence

IMPROVING QUERY PERFORMANCE IN SQL SERVER

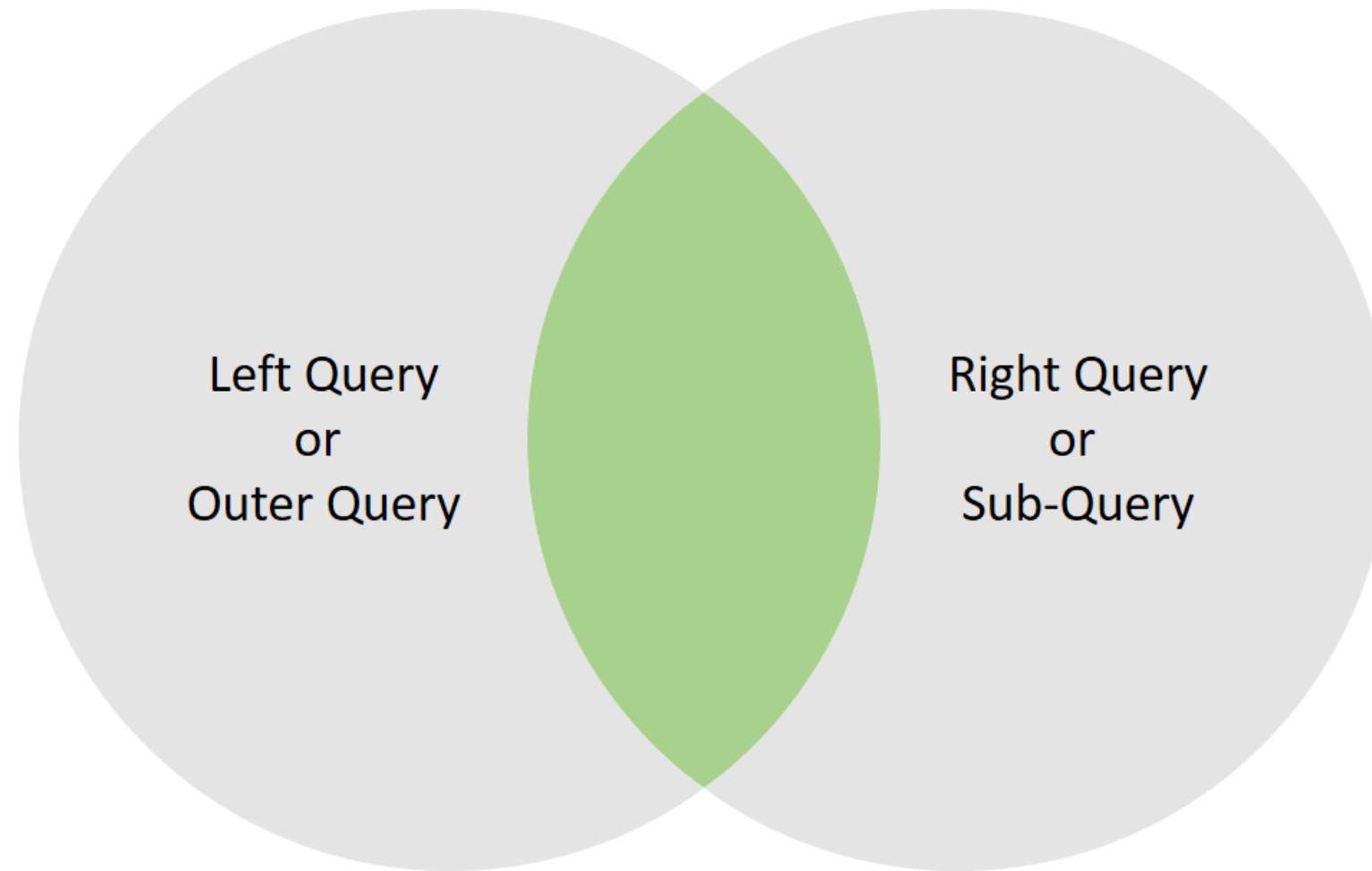


**Dean Smith**

Founder, Atamai Analytics

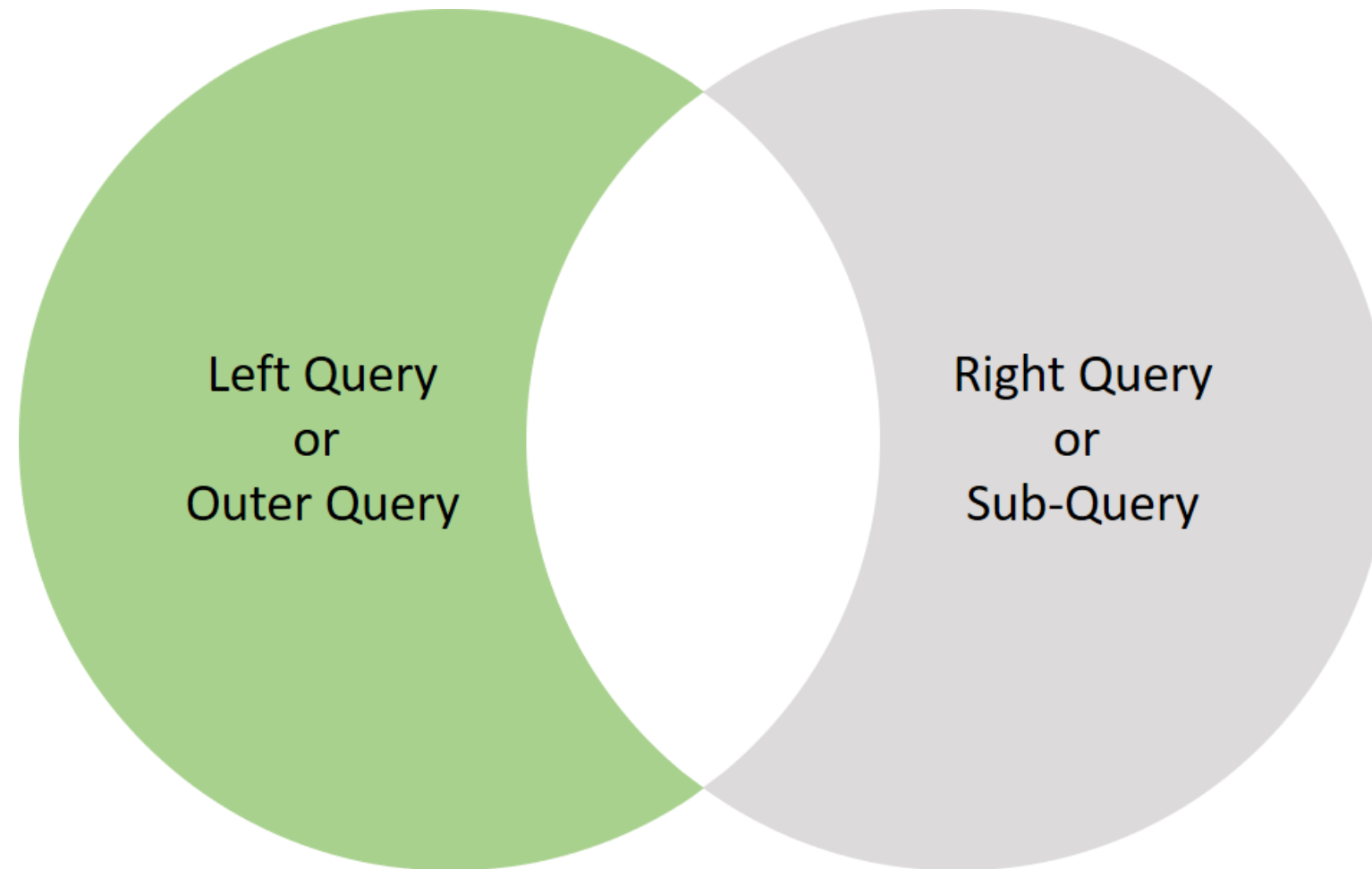
# Venn diagram - presence

Data present in both tables.



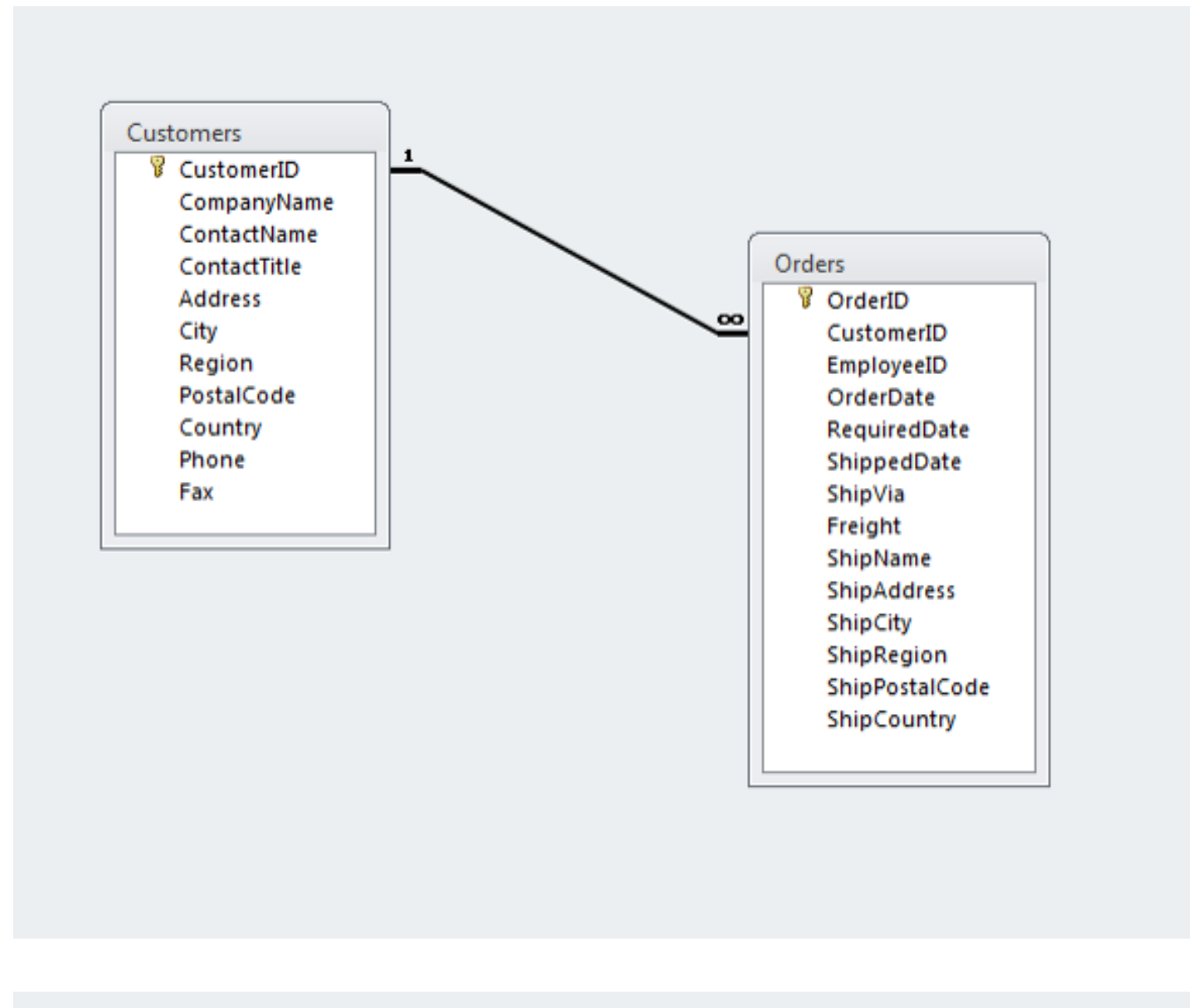
# Venn diagram - absence

Data present in the left table but absent in the right table.





# Customer Orders database



# INTERSECT

```
SELECT CustomerID  
FROM Customers
```

```
SELECT CustomerID  
FROM Orders;
```

# INTERSECT

```
SELECT CustomerID  
FROM Customers
```

INTERSECT

```
SELECT CustomerID  
FROM Orders;
```

CustomerID
ALFKI
LAUGB
QUICK
REGGC
SPLIR
CHOPS
...

# EXCEPT

```
SELECT CustomerID  
FROM Customers
```

```
SELECT CustomerID  
FROM Orders;
```

# EXCEPT

```
SELECT CustomerID  
FROM Customers
```

EXCEPT

```
SELECT CustomerID  
FROM Orders;
```

CustomerID
FISSA
PARIS

# INTERSECT and EXCEPT

## Advantages

- Great for data interrogation
- Remove duplicates from the returned results

## Disadvantages

- The number and order of columns in the `SELECT` statement must be the same between queries

# Let's practice!

IMPROVING QUERY PERFORMANCE IN SQL SERVER

# Alternative methods

# 1

IMPROVING QUERY PERFORMANCE IN SQL SERVER



**Dean Smith**

Founder, Atamai Analytics



# EXISTS

```
SELECT CustomerID,  
        CompanyName,  
        ContactName  
FROM Customers c  
WHERE EXISTS  
        (SELECT 1  
         FROM Orders o  
         WHERE c.CustomerID = o.CustomerID);
```

CustomerID	CompanyName	ContactName
ALFKI	Alfreds Futterkiste	Maria Anders
LAUGB	Laughing Bacchus Wine Cellars	Yoshi Tannamuri
QUICK	QUICK-Stop	Horst Kloss
...	...	...

# IN

```
SELECT CustomerID,  
       CompanyName,  
       ContactName  
FROM Customers  
WHERE CustomerID IN  
       (SELECT CustomerID  
        FROM Orders);
```

CustomerID	CompanyName	ContactName
ALFKI	Alfreds Futterkiste	Maria Anders
LAUGB	Laughing Bacchus Wine Cellars	Yoshi Tannamuri
QUICK	QUICK-Stop	Horst Kloss
...	...	...

# EXISTS vs. IN

- `EXISTS` will stop searching the sub-query when the condition is TRUE
- `IN` collects all the results from a sub-query before passing to the outer query
- Consider using `EXISTS` instead of `IN` with a sub-query

# NOT EXISTS

```
SELECT CustomerID,  
       CompanyName,  
       ContactName  
FROM Customers c  
WHERE NOT EXISTS  
      (SELECT 1  
       FROM Orders o  
       WHERE c.CustomerID = o.CustomerID);
```

CustomerID	CompanyName	ContactName
FISSA	FISSA Fabrica Inter. Salchichas S.A.	Diego Roel
PARIS	Paris spécialités	Marie Bertrand

# NOT IN

```
SELECT CustomerID,  
       CompanyName,  
       ContactName  
FROM Customers  
WHERE CustomerID NOT IN  
      (SELECT CustomerID  
       FROM Orders);
```

CustomerID	CompanyName	ContactName
FISSA	FISSA Fabrica Inter. Salchichas S.A.	Diego Roel
PARIS	Paris spécialités	Marie Bertrand

# NOT IN and NULLs

```
SELECT UNStatisticalRegion AS UN_Region
      ,CountryName
      ,Capital
FROM Nations
WHERE Capital NOT IN
      (SELECT NearestPop
       FROM Earthquakes);
```

UN_Region	CountryName	Capital

# Handling NOT IN NULLs

```
SELECT UNStatisticalRegion AS UN_Region
      ,CountryName
      ,Capital
FROM Nations
WHERE Capital NOT IN
      (SELECT NearestPop
       FROM Earthquakes
       WHERE NearestPop IS NOT NULL);
```

UN_Region	CountryName	Capital
South Asia	India	New Delhi
East Asia and Pacific	Indonesia	Jakarta
East Asia and Pacific	East Timor	Dili
Sahara Africa	Comoros	Moroni
...	...	...

# EXISTS, NOT EXISTS, IN and NOT IN

## Advantages

- Results can contain any column from the outer query, and in any order

## Disadvantages

- The way NOT IN handles NULL values in the sub-query



# Let's practice!

IMPROVING QUERY PERFORMANCE IN SQL SERVER

# Alternative methods

## 2

IMPROVING QUERY PERFORMANCE IN SQL SERVER



**Dean Smith**  
Founder, Atamai Analytics

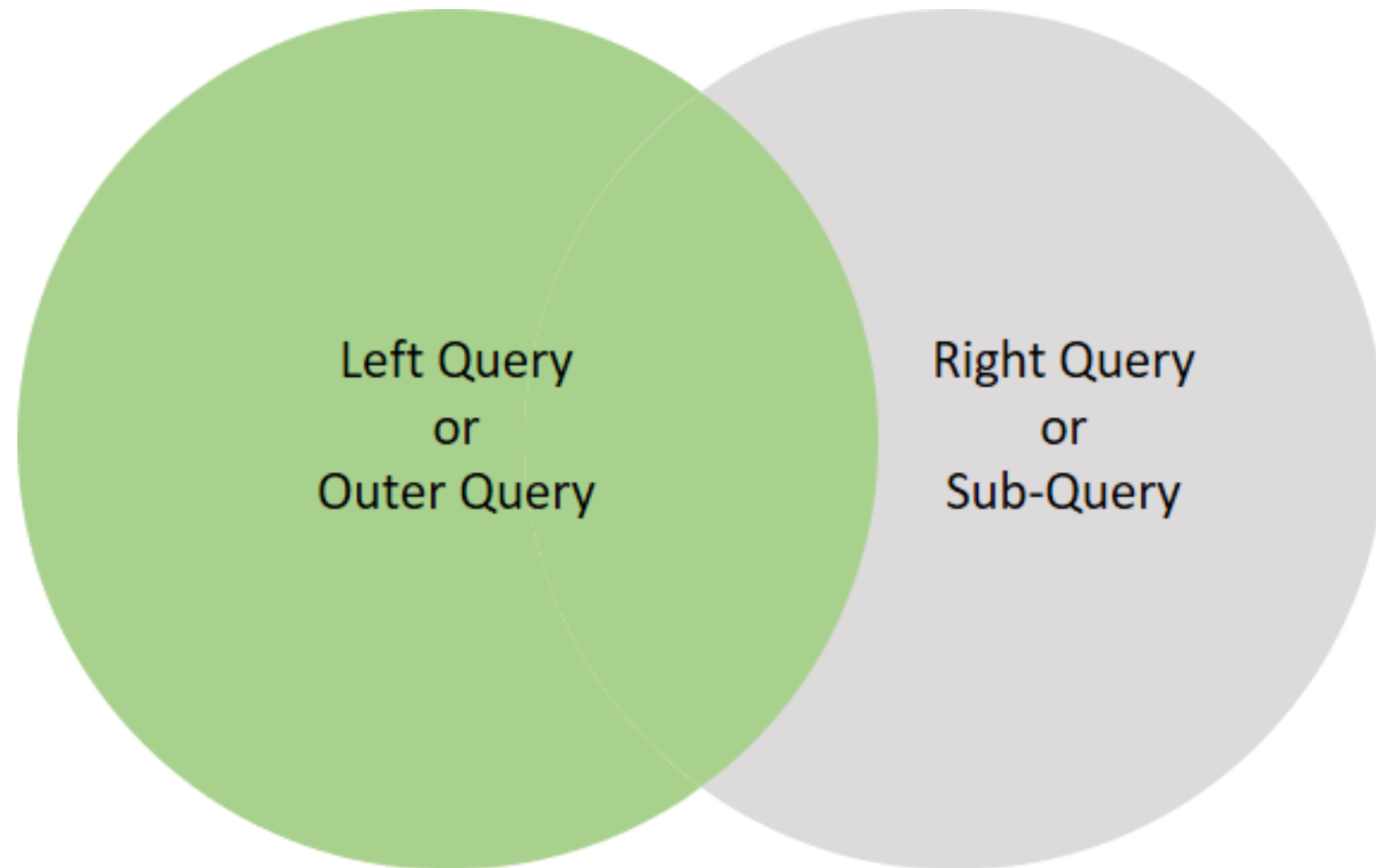
# INNER JOIN

```
SELECT c.CustomerID
        , c.CompanyName
        , o.OrderID
        , o.OrderDate
        , o.ShippedDate
        , o.Freight
FROM Customers c
INNER JOIN Orders o
        ON c.CustomerID = o.CustomerID
```

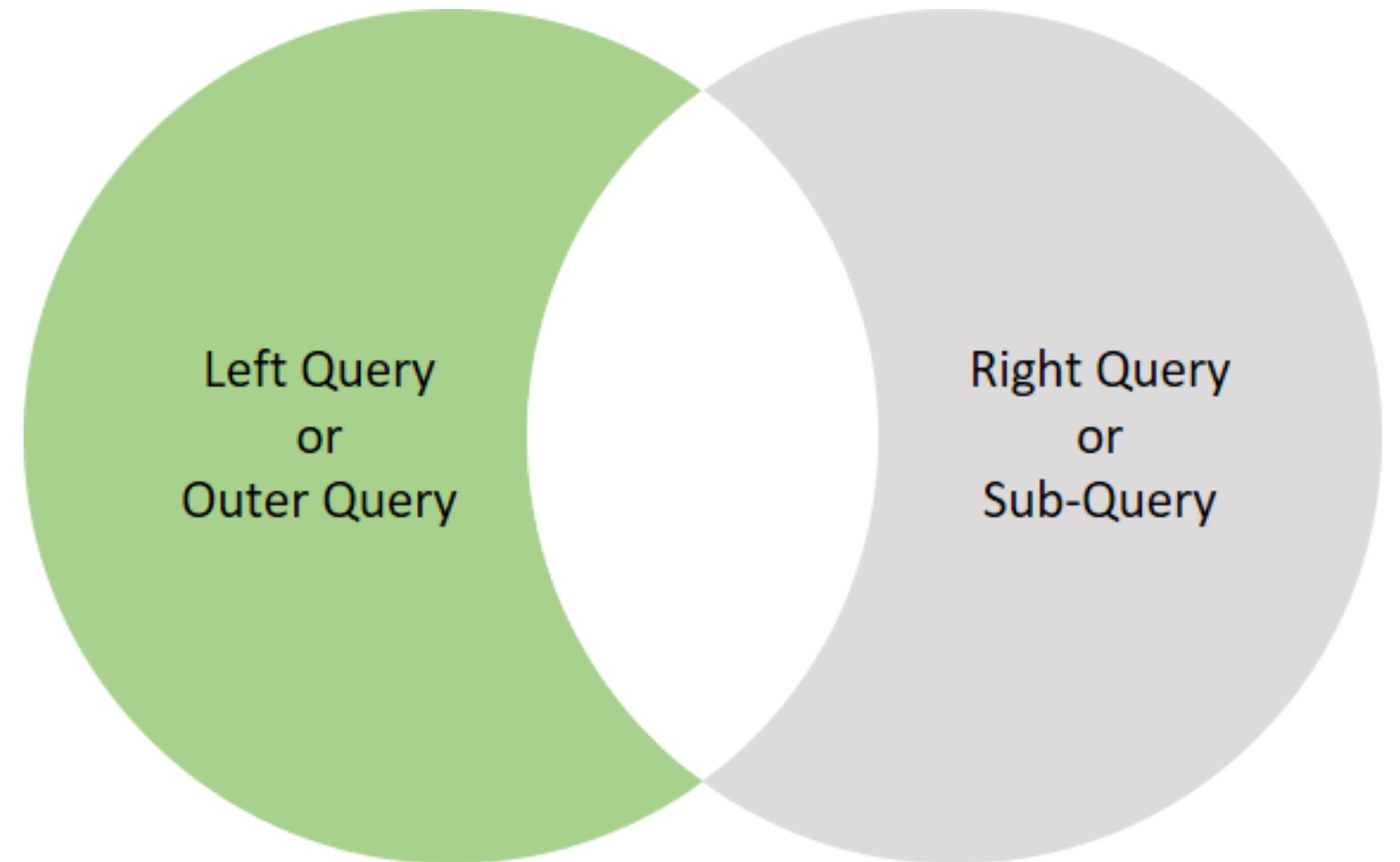
CustomerID	CompanyName	OrderID	...
VINET	Vins et alcools Chevalier	10248	...
HANAR	Hanari Carnes	10250	...
VICTE	Victuailles en stock	10251	...
SUPRD	Suprêmes délices	10252	...
...	...	...	...

# LEFT OUTER JOIN

*Inclusive* LEFT OUTER JOIN



*Exclusive* LEFT OUTER JOIN



# Exclusive LEFT OUTER JOIN

```
SELECT c.CustomerID
      , c.CompanyName
      , o.OrderID
      , o.OrderDate
      , o.ShippedDate
      , o.Freight
FROM Customers c
LEFT OUTER JOIN Orders o
  ON c.CustomerID = o.CustomerID
WHERE o.CustomerID IS NULL
```

CustomerID	CompanyName	OrderID	...
FISSA	FISSA Fabrica Inter. Salchichas S.A.	NULL	...
PARIS	Paris spécialités	NULL	...

# Review: INTERSECT and EXCEPT

`INTERSECT` : checks for presence

`EXCEPT` : checks for absence

## Advantages

- Great for data interrogation
- Remove duplicates from the returned results

## Disadvantage

- The number and order of columns in the `SELECT` statement must be the same between queries

# Review: EXISTS and NOT EXISTS

**EXISTS** : checks for presence

**NOT EXISTS** : checks for absence

## Advantages

- The sub-query will stop searching as soon as it evaluates to TRUE
- Results can contain any column from the outer query, and in any order

## Disadvantage

- Results can only contain columns from the outer query

# Review: IN and NOT IN

`IN` : checks for presence

`NOT IN` : checks for absence

## Advantage

- Results can contain any column from the outer query, and in any order

## Disadvantages

- Results can only contain columns from the outer query
- No results returned because of the way `NOT IN` handles nulls in the sub-query



# Review: INNER JOIN and exclusive L.O.J

`INNER JOIN` : checks for presence

exclusive `LEFT OUTER JOIN` : checks for absence

## Advantage

- Results can contain any column, from all joined queries, in any order

## Disadvantage

- Requirement to add the `IS NULL` `WHERE` filter condition with the exclusive `LEFT OUTER JOIN`

# Let's practice!

IMPROVING QUERY PERFORMANCE IN SQL SERVER