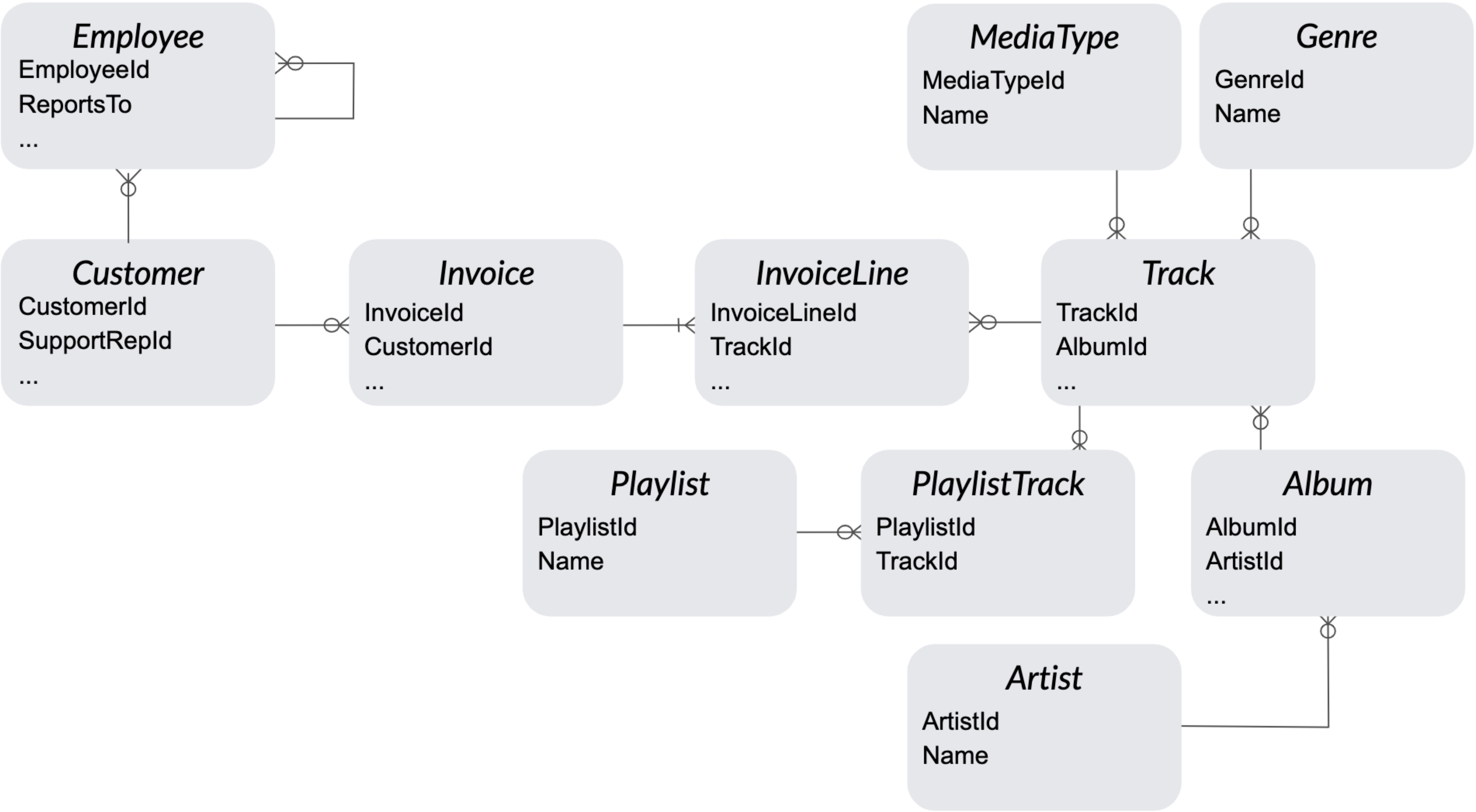
Using joins

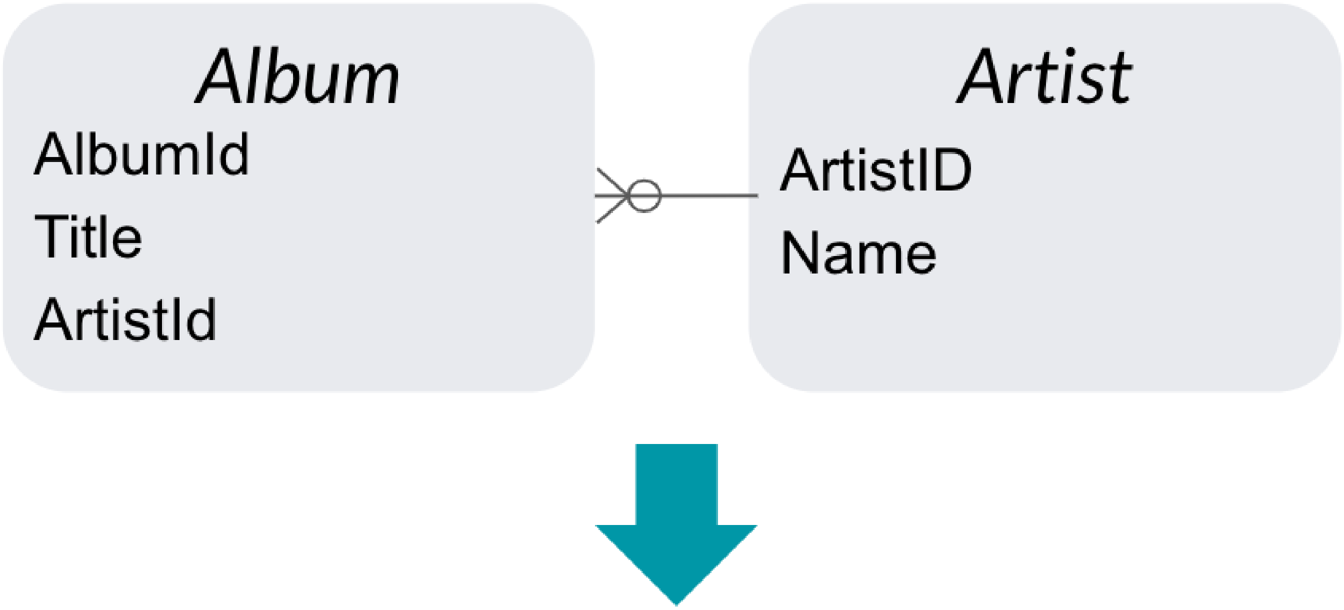
INTRODUCTION TO ORACLE SQL

Sara Billen
Curriculum Manager

Chinook dataset



Combining data from more than one table



Album

AlbumId

Title

ArtistId

Artist

ArtistID

Name

Pearl Jam Albums
Riot Act
Ten
….

SQL joins

Type of joins:

- **Inner Join**
- Outer Joins
- Cross Joins
- Self Joins

Inner join

```sql
SELECT Album.Title, Artist.Name
FROM Album INNER JOIN Artist
ON  Album.ArtistId = Artist.ArtistId
```

```
| Album.Title                     | Artist.Name   |
|---------------------------------|---------------|
| Black Sabbath                   | Black Sabbath |
| Black Sabbath Vol. 4 (Remaster) | Black Sabbath |
| The Cream Of Clapton            | Eric Clapton  |
| Unplugged                       | Eric Clapton  |
| ...                             | ...           |
```

Breakdown of an inner join

1. Select columns to output and include table names
2. In the `FROM` , list the relevant tables separated by `INNER JOIN`
3. Use `ON` to define the column to join on

```
SELECT Album.Title, Artist.Name
```

```
FROM Album INNER JOIN Artist
```

```
ON Album.ArtistId = Artist.ArtistId
```

Add in other necessary clauses like `WHERE` , `ORDER BY` !

Inner join

```sql
SELECT Album.Title, Artist.Name
FROM Album INNER JOIN Artist
ON  Album.ArtistId = Artist.ArtistId
WHERE Artist.Name = 'Pearl Jam'
```

```
| Album.Title | Artist.Name |
|-------------|-------------|
| Pearl Jam   | Pearl Jam   |
| Riot Act    | Pearl Jam   |
| Ten         | Pearl Jam   |
| ...         | ...         |
```

USING instead of ON

```
SELECT Album.Title, Artist.Name
FROM Album INNER JOIN Artist
ON  Album.ArtistId = Artist.ArtistId
```

same as

```
SELECT Album.Title, Artist.Name
FROM Album INNER JOIN Artist
USING (ArtistId)
```

- Columns need to be identically named in the two tables
- Enclose column name in parentheses

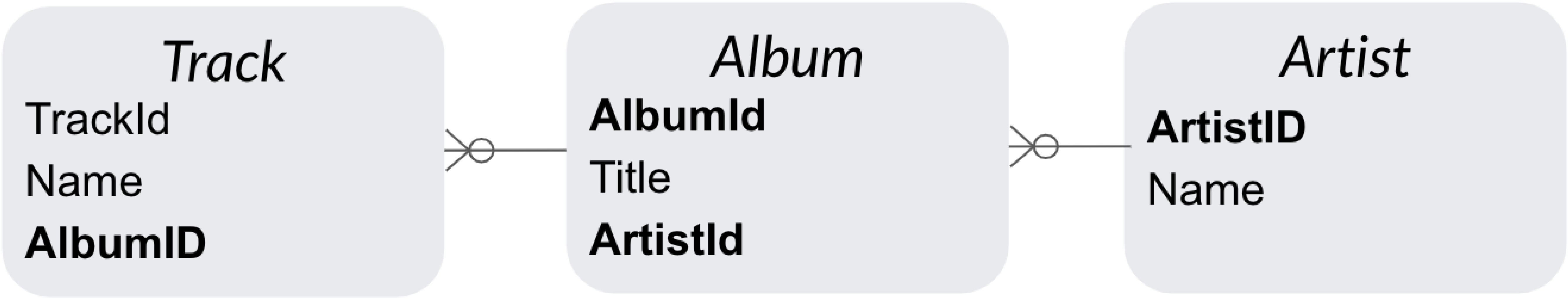
Table aliases

```
SELECT DISTINCT Customer.FirstName, Customer.LastName,
                Employee.FirstName, Employee.LastName
FROM Customer INNER JOIN Employee
ON Customer.SupportRepID = Employee.EmployeeID
```

With aliases:

```
SELECT DISTINCT c.FirstName, c.LastName, e.FirstName, e.LastName
FROM Customer c INNER JOIN Employee e
ON c.SupportRepID = e.EmployeeID
```

Joining more than two tables



```
SELECT t.Name AS Track, al.Title AS Album, ar.Name AS Artist
FROM
    Track t INNER JOIN Album al USING (AlbumId)
    INNER JOIN Artist ar USING (ArtistId)
```

Joining more than two tables

```
SELECT t.Name as Track, al.Title as Album, ar.Name as Artist
FROM
    Track t INNER JOIN Album al USING (AlbumId)
    INNER JOIN Artist ar USING (ArtistId)
```

```
| Track              | Album                          | Artist      |
|--------------------|--------------------------------|-------------|
| The Legacy         | A Matter of Life and Death     | Iron Maiden |
| Lord of Light      | A Matter of Life and Death     | Iron Maiden |
| Out of the Shadows | A Matter of Life and Death     | Iron Maiden |
| ...                | ...                            | ...         |
```

Let's practice!

INTRODUCTION TO ORACLE SQL

Outer joins

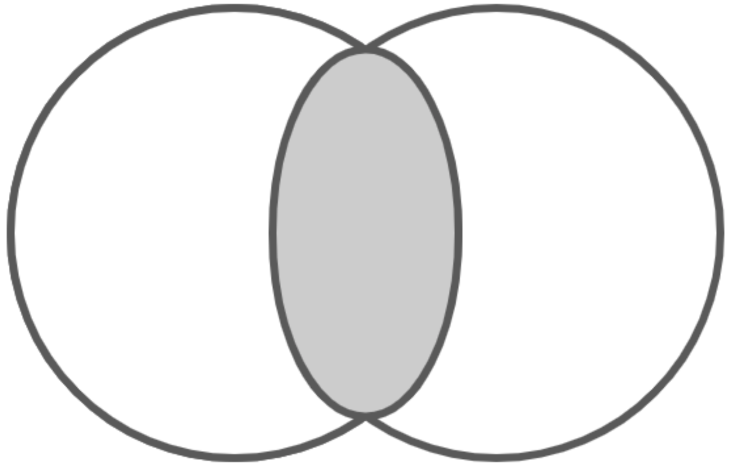
INTRODUCTION TO ORACLE SQL

Sara Billen
Instructor

Three types of outer joins

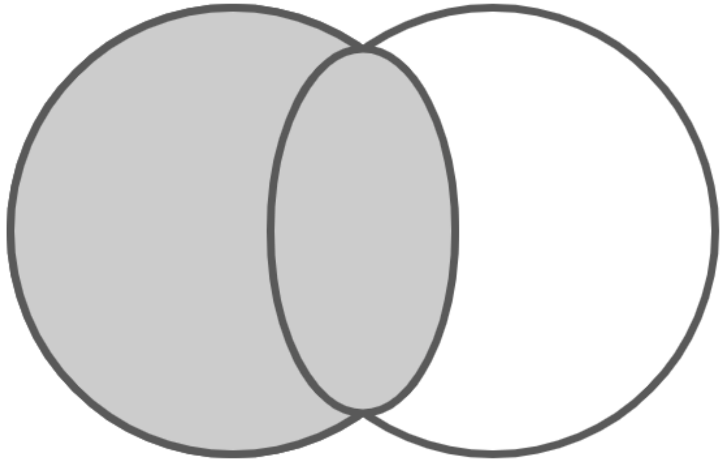
1. `LEFT OUTER JOIN`
2. `RIGHT OUTER JOIN`
3. `FULL OUTER JOIN`

Inner Join



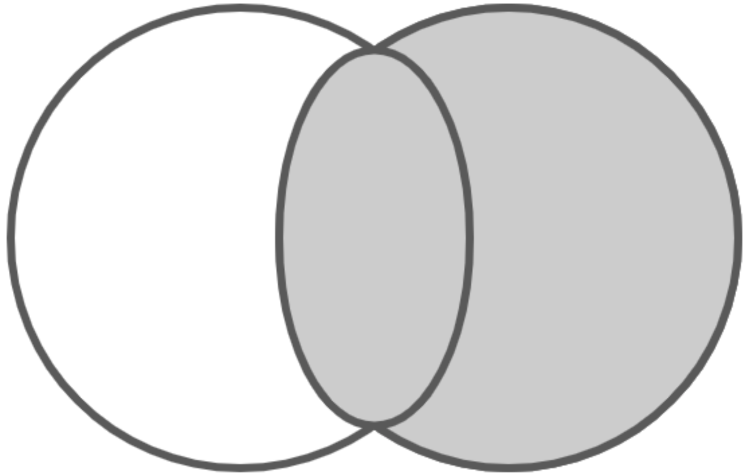
Returns matched rows only

Left Outer Join



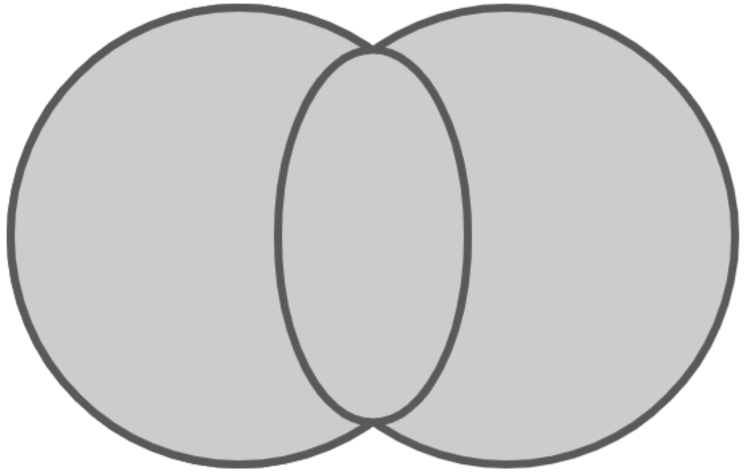
Returns matched rows and left table's rows

Right Outer Join



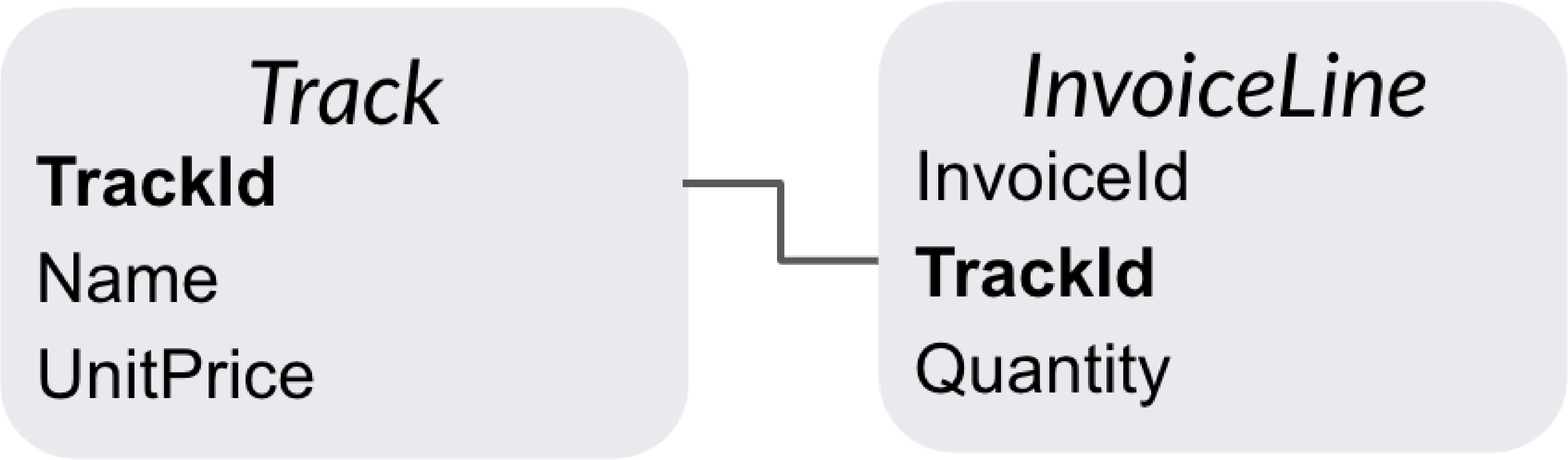
Returns matched rows and right table's rows

Full Outer Join



Returns matched rows and both tables' rows

Track and invoice tables



Track

TrackId

Name

UnitPrice

InvoiceLine

InvoiceId

TrackId

Quantity

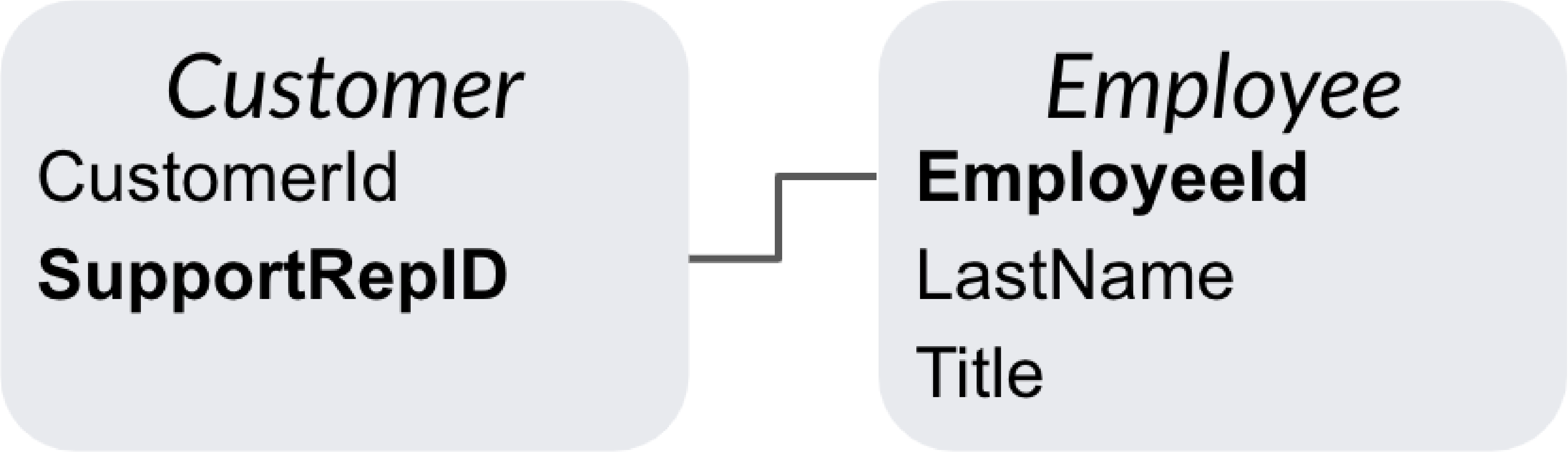
- Every invoice refers to at least one track
- Not all tracks are in an invoice

Left outer join

```
SELECT t.TrackId, t.Name, i.InvoiceId, i.Quantity
  FROM Track t LEFT OUTER JOIN InvoiceLine i
  USING (TrackId)
```

trackid	name	composer	invoiceid	quantity
20	Overdose	AC/DC	214	1
20	Overdose	AC/DC	3	1
21	A Bad Place To Be	AC/DC	319	1
22	Whole Lotta Rosie	AC/DC	null	null
23	Walk on Water	Steven Tyler	null	null

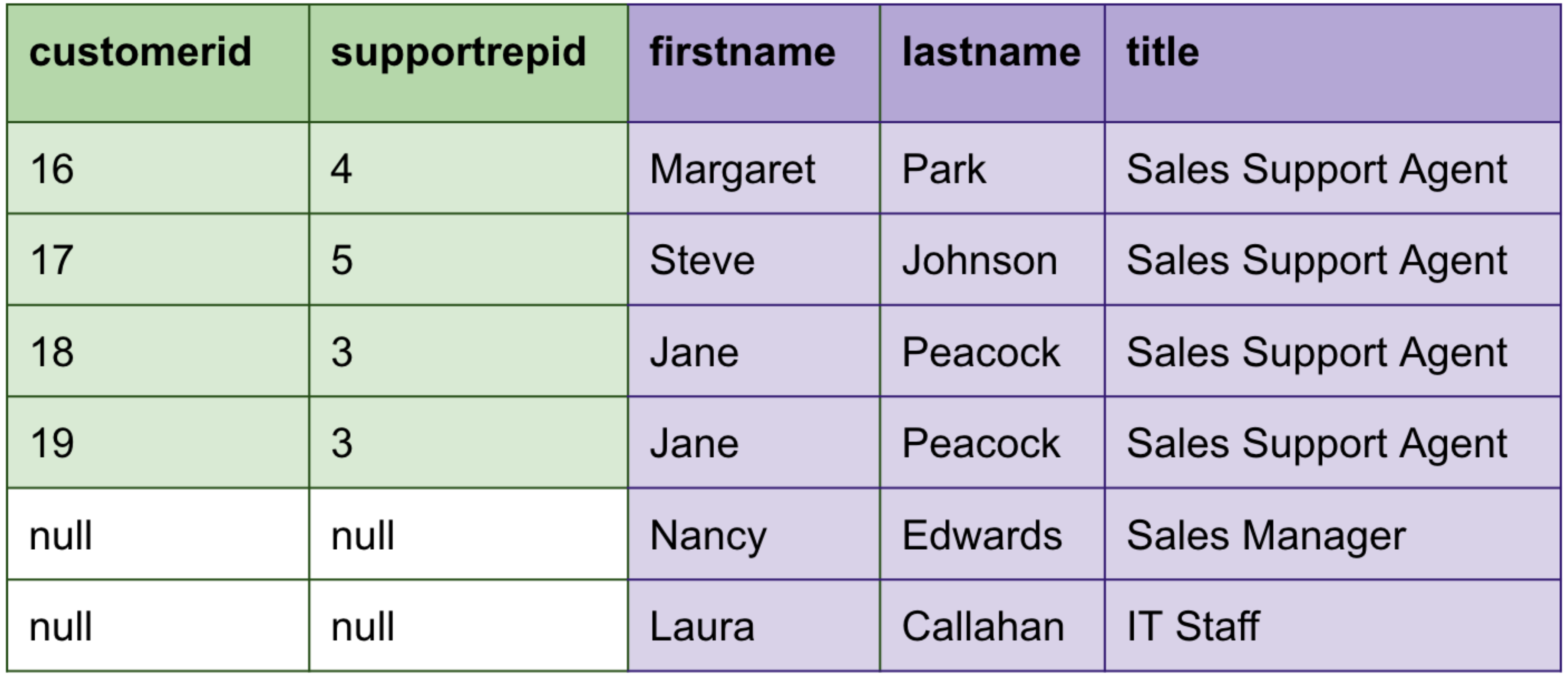
Customer and employee tables



- Every customer has a support rep
- Not every employee is a support rep

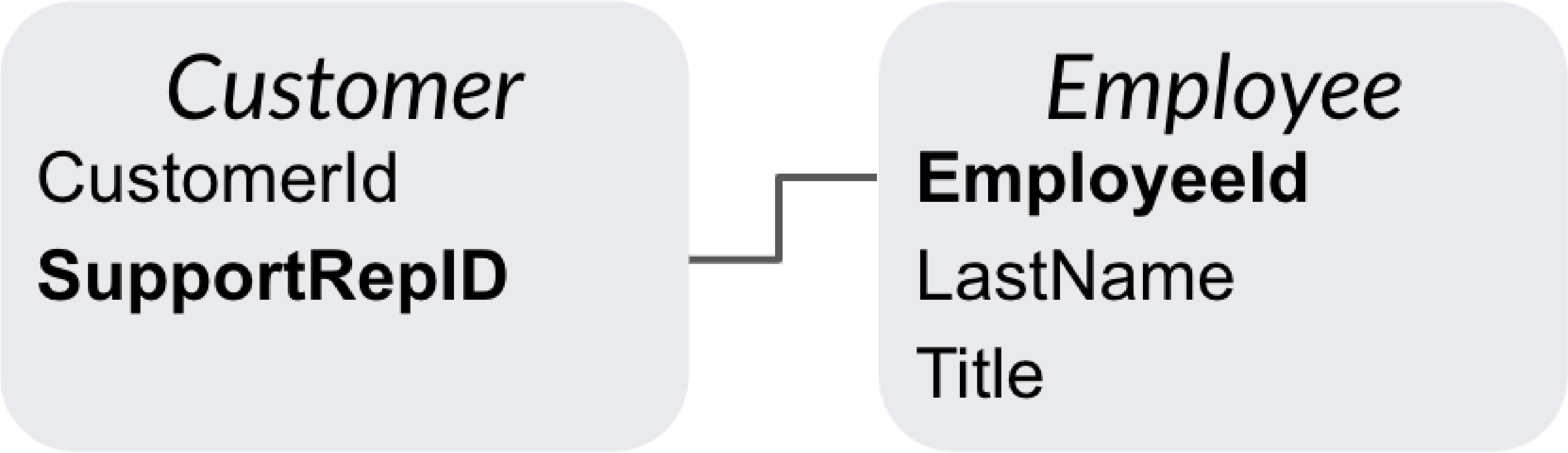
Right outer join

```
SELECT c.CustomerId, c.SupportRepId, e.FirstName, e.LastName, e.Title
FROM Customer c RIGHT OUTER JOIN Employee e
    ON c.SupportRepId = EmployeeId
```



customerid	supportrepid	firstname	lastname	title
16	4	Margaret	Park	Sales Support Agent
17	5	Steve	Johnson	Sales Support Agent
18	3	Jane	Peacock	Sales Support Agent
19	3	Jane	Peacock	Sales Support Agent
null	null	Nancy	Edwards	Sales Manager
null	null	Laura	Callahan	IT Staff

Customer and employee tables



Customer

CustomerId

SupportRepID

Employee

EmployeeId

LastName

Title

- **Not** every customer has a support rep
- Not every employee is a support rep

Full outer join

```
SELECT c.CustomerId, c.SupportRepId,e.FirstName, e.LastName, e.Title
  FROM Customer c FULL OUTER JOIN Employee e
  ON c.SupportRepId = EmployeeId
```

customerid	supportrepid	firstname	lastname	title
17	5	Steve	Johnson	Sales Support Agent
18	3	Jane	Peacock	Sales Support Agent
19	null	null	null	null
null	null	Nancy	Edwards	Sales Manager
null	null	Laura	Callahan	IT Staff

Let's practice!

INTRODUCTION TO ORACLE SQL

More joins

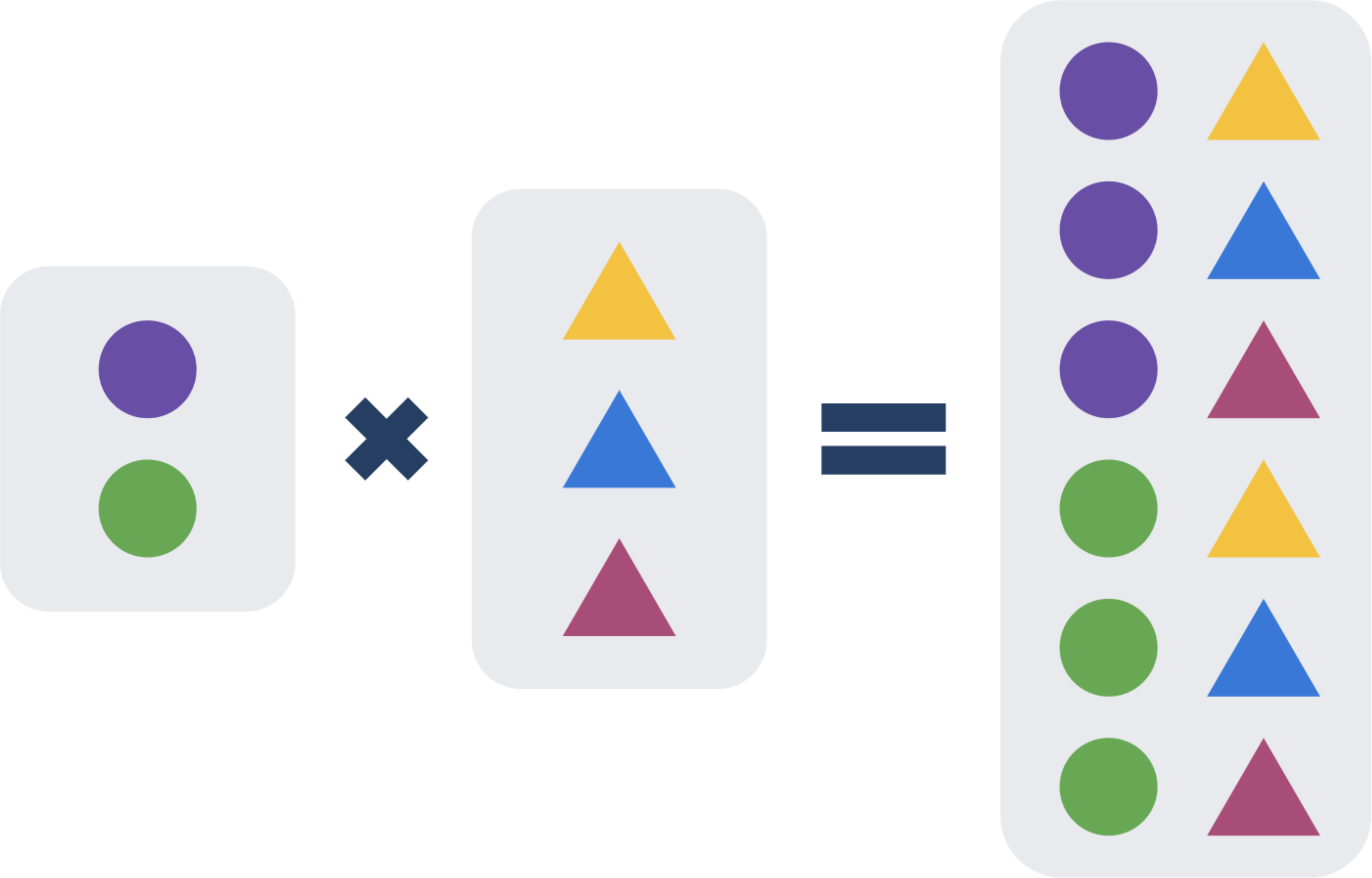
INTRODUCTION TO ORACLE SQL

Sara Billen
Curriculum Manager

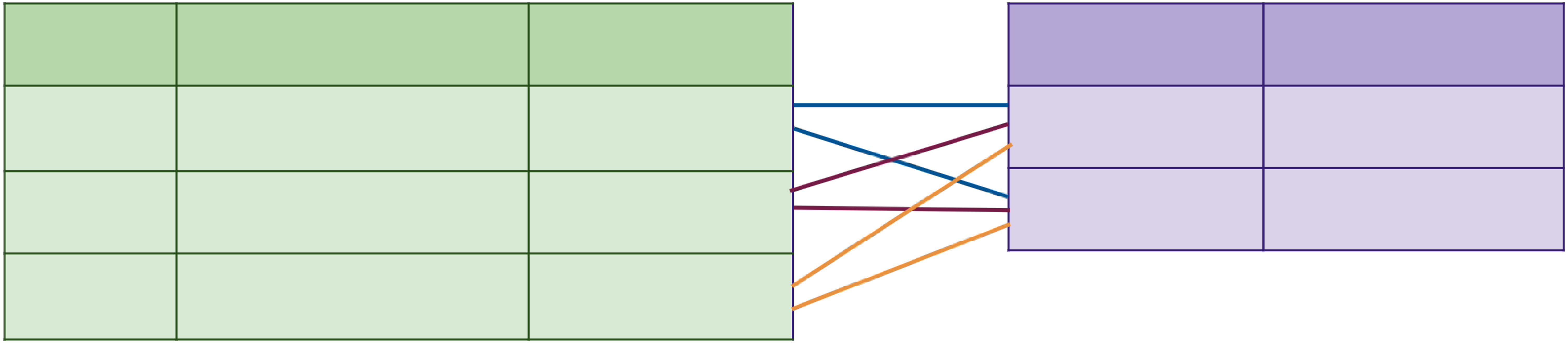
Other join types

- `CROSS JOIN`
- Self `JOIN`

Cartesian product (AKA cross product)



Cross product on tables



Cross product example

```
SELECT *
FROM MediaType
```

```
| MediaTypeId | Name                          |
|-------------|-------------------------------|
| 1           | MPEG audio file               |
| 2           | Protected AAC audio file      |
| 3           | Protected MPEG-4 video file |
| 4           | Purchased AAC audio file      |
| 5           | AAC audio file                |
```

Cross product example

```
SELECT m.MediaTypeId, m.Name, t.TrackId, t.Name, t.MediaTypeId
FROM MediaType m CROSS JOIN Track t
```

```
| MediaTypeId | Name                        | TrackId | Name        | MediaTypeId       |
|-------------|-----------------------------|---------|-------------|-------------------|
| 1           | MPEG audio file             | 3021    | Forty       | 1                 |
| 2           | Protected AAC audio file    | 3021    | Forty       | 1                 |
| 3           | Protected MPEG-4 video file | 3021    | Forty       | 1                 |
| 4           | Purchased AAC audio file    | 3021    | Forty       | 1                 |
| 5           | AAC audio file              | 3021    | Forty       | 1                 |
```

Self join

- Join a table to itself

Use case:

```
SELECT * FROM Employee
```

```
| EmployeeId | LastName | Title               | ... | ReportsTo |
|------------|----------|---------------------|-----|-----------|
| 1          | Adames   | General Manager     | ... | null      |
| 2          | Edwards  | Sales Manager       | ... | 1         |
| 3          | Peacock  | Sales Support Agent | ... | 2         |
| 4          | Park     | Sales Support Agent | ... | 2         |
```

Self join

```sql
SELECT e.LastName Employee, m.LastName ReportsTo
FROM Employee e JOIN Employee m
ON (e.ReportsTo = m.EmployeeId)
```

```
| Employee | ReportsTo |
|----------|-----------|
| Edwards  | Adams     |
| Peacock  | Edwards   |
| Mitchell | Adams     |
| Park       | Edwards   |
| ...      | ...       |
```

Let's practice!

INTRODUCTION TO ORACLE SQL

Set operators

INTRODUCTION TO ORACLE SQL

Sara Billen
Curriculum Manager

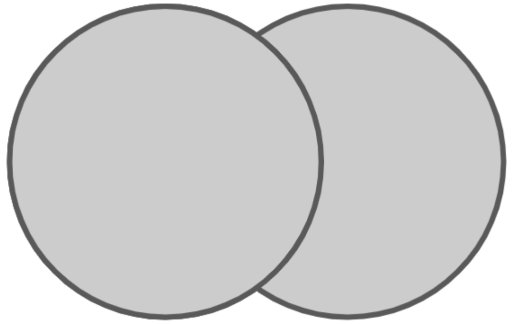
What are set operators?

Set operators take the output of two or more `SELECT` queries and combine them into one result.

- Join clauses combines tables
  - Column-oriented
- Set operators combines queries
  - Row-oriented

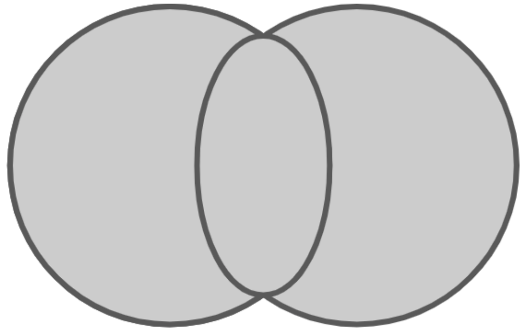
Types of set operators

Union



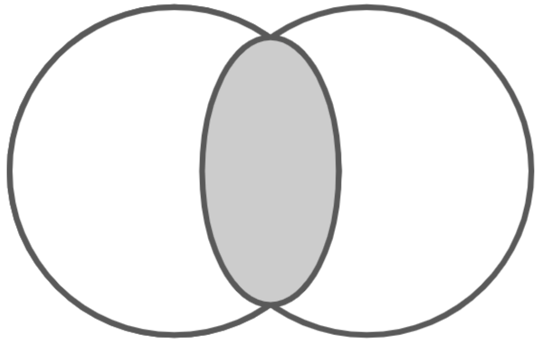
All rows with no duplicates

Union All



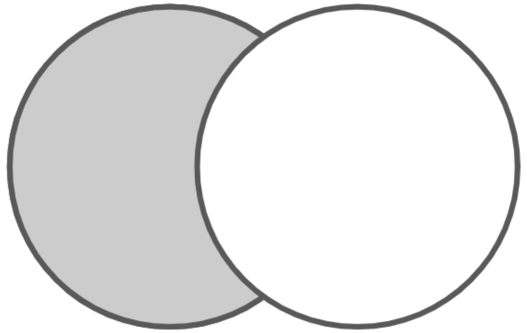
All rows with duplicates

Intersect



Rows outputted by both queries

Minus



Distinct rows in 1st query that are not in the 2nd

Union

All rows with no duplicates

What are all the cities associated with our clients?

```
SELECT City FROM Customer
UNION
SELECT BillingCity FROM Invoice
```

```
| City           |
|----------------|
| Lyon           |
| Fort Worth     |
| Vienne         |
| Brussels       |
| Orlando        |
| Copenhagen     |
| Oslo           |
| Rio de Janeiro |
| Boston         |
| ...            |
```

Union all

All rows with duplicates

What are all the cities associated with our clients and with what frequency?

```sql
SELECT City from Customer
UNION ALL
SELECT BillingCity from Invoice
```

```
| City          |
|---------------|
| Oslo          |
| Prague        |
| Prague        |
| Vienee        |
| Brussels      |
| Copenhagen    |
| Mountain View |
| Mountain View |
| Mountain View |
| ...           |
```

Intersect

Rows outputted by both queries

Which tracks by Miles Davis are in a playlist?

```
(SELECT TrackId from PlaylistTrack)
INTERSECT
(SELECT TrackId from Track
WHERE Composer = 'Miles Davis')
```

```
| TrackId |
|---------|
| 612     |
| 600     |
| 614     |
| 604     |
| 605     |
| 598     |
| 617     |
```

Minus

Distinct rows in 1st query that aren't in the 2nd query

Who are artists that don't compose music?

```
(SELECT Name FROM Artist)
MINUS
(SELECT Composer FROM Track)
ORDER BY 1 DESC
```

```
| Name                     |
|--------------------------|
| Zeca Pagodinho           |
| Youssou N'Dour           |
| Yo-Yo Ma                 |
| Yehudi Menuhin           |
| Xis                      |
| Wilhelm Kempff           |
| Whitesnake               |
| Vinícius E Qurteto Em Cy |
| Vinícius E Odette Lara   |
| ...                      |
```

Let's practice!

INTRODUCTION TO ORACLE SQL