MODULE 2 DATABASE PROGRAMMING

# Keys, JOIN, and UNION

# YESTERDAY...

How do we specify the order of query results?

What's the syntax for only returning the first 8 records?

What is an aggregate function?

How do you get records into summary rows?

How do you use IN or NOT IN during the where clause?

amazon

# Tables (maybe)

**Users**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| FirstName | varchar(50) | ☐ |
| LastName | varchar(50) | ☐ |
| MemberSince | datetime | ☑ |
| isActive | bit | ☐ |
| | | ☐ |

**Purchases**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| userID | int | ☐ |
| productID | int | ☐ |
| PurchaseDate | datetime | ☐ |
| isActive | bit | ☐ |
| | | ☐ |

**Shipping_Addresses**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| Street_1 | varchar(50) | ☐ |
| Street_2 | varchar(50) | ☑ |
| City | varchar(50) | ☐ |
| State | varchar(50) | ☐ |
| Zip | varchar(50) | ☐ |
| userID | int | ☐ |
| isActive | bit | ☐ |
| | | ☐ |

**Products**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| Description | varchar(MAX) | ☐ |
| Price | float | ☐ |
| Location | int | ☐ |
| isActive | bit | ☐ |
| | | ☐ |

**Users**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| FirstName | varchar(50) | ☐ |
| LastName | varchar(50) | ☐ |
| MemberSince | datetime | ☑ |
| isActive | bit | ☐ |
| | | ☐ |

**Shipping_Addresses**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| Street_1 | varchar(50) | ☐ |
| Street_2 | varchar(50) | ☑ |
| City | varchar(50) | ☐ |
| State | varchar(50) | ☐ |
| Zip | varchar(50) | ☐ |
| userID | int | ☐ |
| isActive | bit | ☐ |
| | | ☐ |

**Purchases**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| userID | int | ☐ |
| productID | int | ☐ |
| PurchaseDate | datetime | ☐ |
| isActive | bit | ☐ |
| | | ☐ |

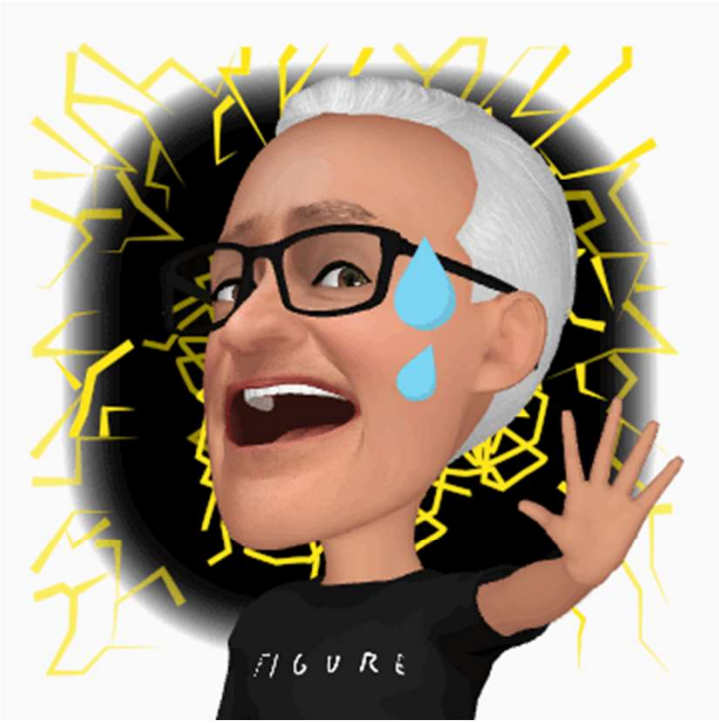
**Products**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| Description | varchar(MAX) | ☐ |
| Price | float | ☐ |
| Location | int | ☐ |
| isActive | bit | ☐ |
| | | ☐ |

select * from Shipping_Addresses where userID in (select id from Users where FirstName='Henry' and LastName='Edwards')

select * from Purchases where userID in (select id from Users where FirstName='Henry' and LastName='Edwards')

select * from Products where id in (select productID from Purchases where userID in (select id from Users where FirstName='Henry' and LastName='Edwards'))



FIGURE

# KEYS

**Users**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| FirstName | varchar(50) | ☐ |
| LastName | varchar(50) | ☐ |
| MemberSince | datetime | ☑ |
| isActive | bit | ☐ |
| | | ☐ |

**Shipping_Addresses**

| Column Name | Data Type | Allow Nulls |
|---|---|---|
| id | int | ☐ |
| Street_1 | varchar(50) | ☐ |
| Street_2 | varchar(50) | ☑ |
| City | varchar(50) | ☐ |
| State | varchar(50) | ☐ |
| Zip | varchar(50) | ☐ |
| userID | int | ☐ |
| isActive | bit | ☐ |
| | | ☐ |

**Primary key** columns are columns that hold a value that is unique for every row in that table.

**Foreign key** is a field in one table that uniquely identifies a row of another table

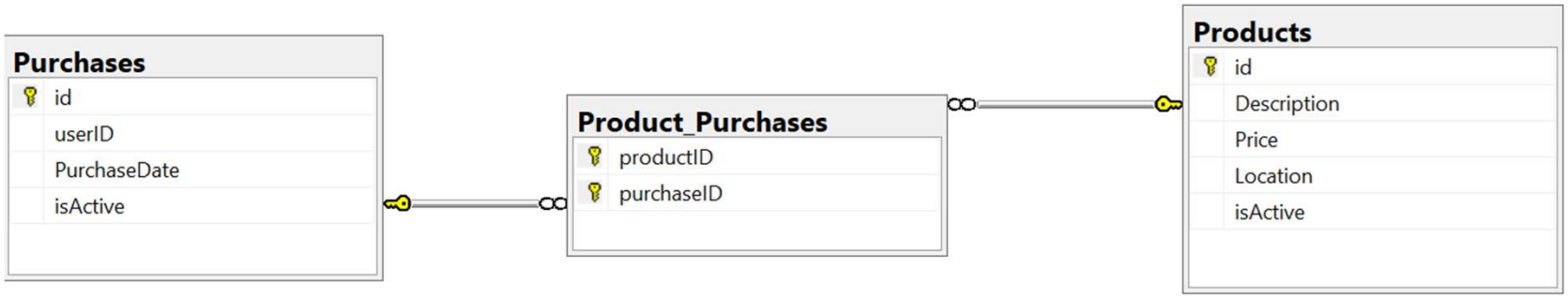
ELEVATE YOURSELF

# CARDINALITY

# ONE TO MANY

# CARDINALITY

# CARDINALITY




# MANY TO MANY

# JOINS

- **SQL JOIN**s allow us to create queries that produce data from one or more tables.

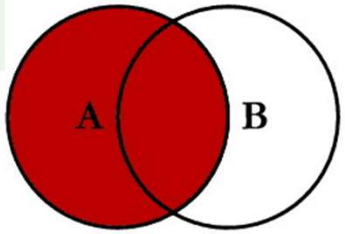
| | code | name | continent | region | surfacearea | indepyear | population | lifeexpectancy | gnp | gnpold | localname | governmentform | headofstate | capital | code2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | ABW | Aruba | North America | Caribbean | 193 | NULL | 103000 | 78.4 | 828.00 | 793.00 | Aruba | Nonmetropolitan Territory of The Netherlands | Beatrix | 129 | AW |
| 2 | AFG | Afghanistan | Asia | Souther... | 652090 | 1919 | 22720000 | 45.9 | 5976 | NULL | Afganist... | Islamic Emirate | Mohamm... | 1 | AF |
| 3 | AFG | Afghanistan | Asia | Souther... | 652090 | 1919 | 22720000 | 45.9 | 5976 | NULL | Afganist... | Islamic Emirate | Mohamm... | 1 | AF |
| 4 | AFG | Afghanistan | Asia | Souther... | 652090 | 1919 | 22720000 | 45.9 | 5976 | NULL | Afganist... | Islamic Emirate | Mohamm... | 1 | AF |
| 5 | AFG | Afghanistan | Asia | Souther... | 652090 | 1919 | 22720000 | 45.9 | 5976 | NULL | Afganist... | Islamic Emirate | Mohamm... | 1 | AF |
| 6 | AGO | Angola | Africa | Central ... | 1246700 | 1975 | 12878000 | 38.3 | 6648 | 7984 | Angola | Republic | Jos? Edu... | 56 | AO |
| 7 | AGO | Angola | Africa | Central ... | 1246700 | 1975 | 12878000 | 38.3 | 6648 | 7984 | Angola | Republic | Jos? Edu... | 56 | AO |
| 8 | AGO | Angola | Africa | Central ... | 1246700 | 1975 | 12878000 | 38.3 | 6648 | 7984 | Angola | Republic | Jos? Edu... | 56 | AO |

select * from country order by code

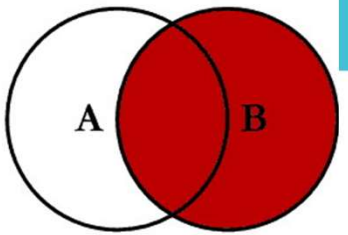
| | id | name | countrycode | district | population |
|---|---|---|---|---|---|
| 1 | 129 | Oranjestad | ABW | ? | 29034 |
| 2 | 1 | Kabul | AFG | Kabol | 1780000 |
| 3 | 2 | Qandahar | AFG | Qan... | 237500 |
| 4 | 3 | Herat | AFG | Herat | 186800 |
| 5 | 4 | Mazar-e-... | AFG | Balkh | 127800 |
| 6 | 56 | Luanda | AGO | Lua | 2022000 |
| 7 | 57 | Huambo | AGO | Hua | 163100 |
| 8 | 58 | Lobito | AGO | Ben... | 130000 |

select * from city order by countrycode

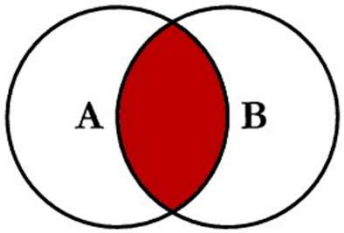
# SQL JOINS



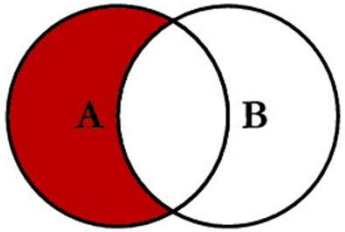

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



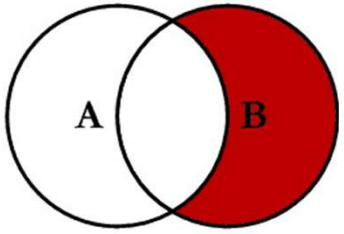

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



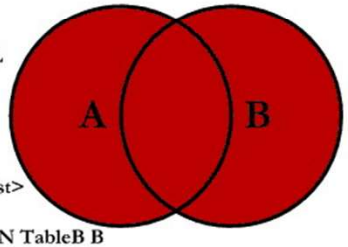

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



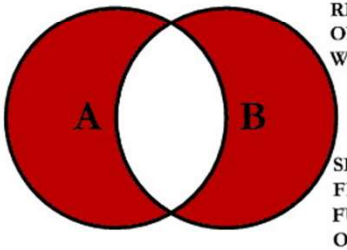

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```

© C.L. Moffatt, 2008

# UNIONS

- A SQL UNION combines the results of two or more queries into a single result set.
- The number of columns involved **must match exactly** and data types must be identical.
- Duplicate rows are removed.

**SYNTAX:**

```
SELECT expression1, expression2, ... expression_n FROM tables [WHERE conditions]
UNION
SELECT expression1, expression2, ... expression_n FROM tables [WHERE conditions]
```

# LET'S CODE!

# WHAT QUESTIONS DO YOU HAVE?

Reading for tonight:

**Inserting, Updating, and Deleting**