MODULE 2 DATABASE PROGRAMMING

Keys, JOIN, and UNION

TECH ELEVATOR

A Stride Company

YESTERDAY…

How do we specify the order of query results?

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

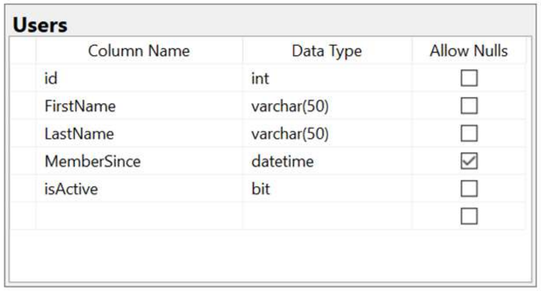
What is an aggregate function?

How do you get records into summary rows?

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

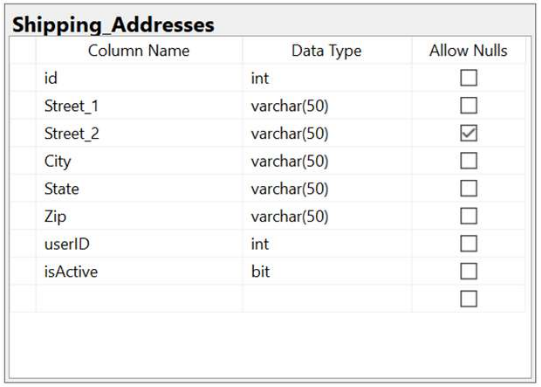
amazon

Tables (maybe)



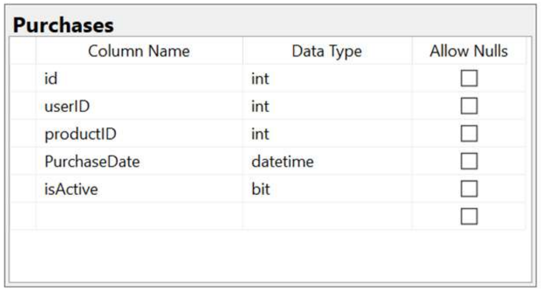
Users

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



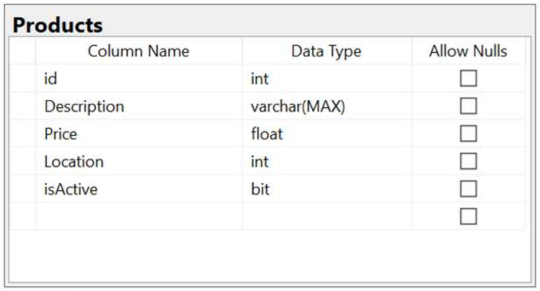
Shipping_Addresses

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	□
		□



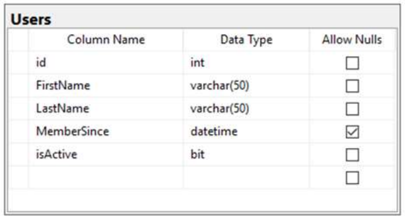
Purchases

Column Name	Data Type	Allow Nulls
id	int	□
userID	int	□
productID	int	□
PurchaseDate	datetime	□
isActive	bit	□
		□



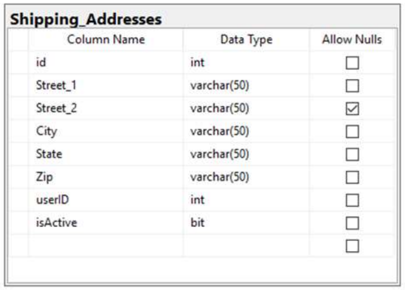
Products

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



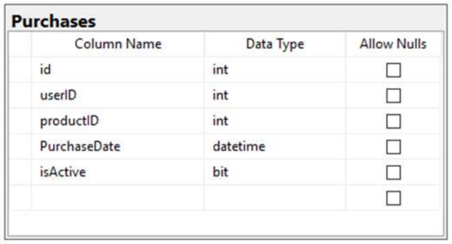
Users

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



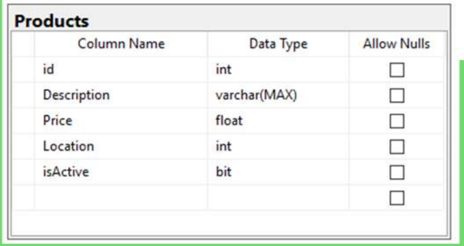
Shipping_Addresses

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	□
		□



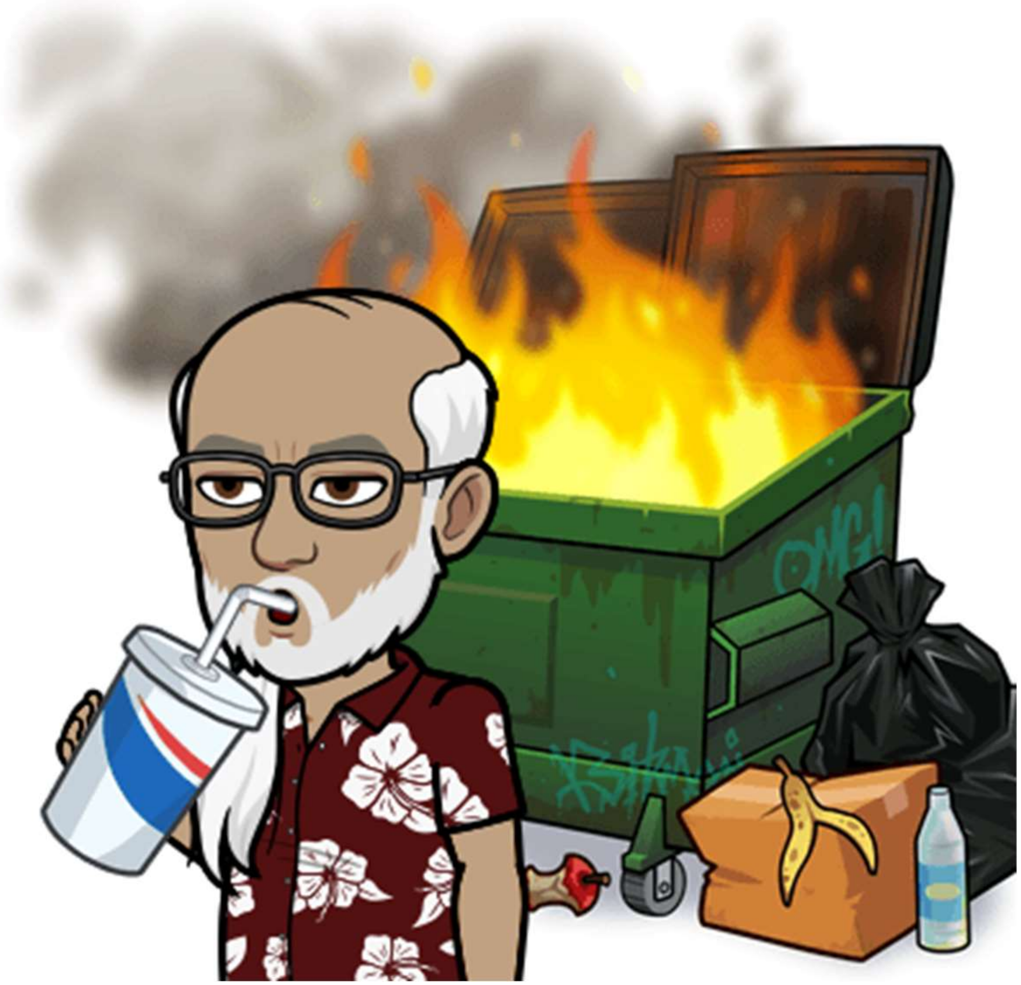
Products

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

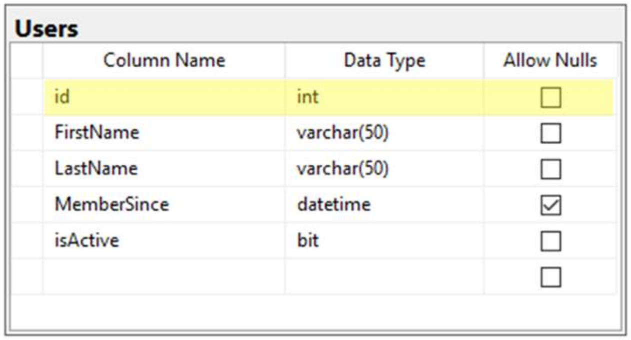
```
select * from Shipping_Addresses where userID in (select id from
Users where FirstName=’Henry’ and LastName=’Edwards’)
```

```
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’))
```

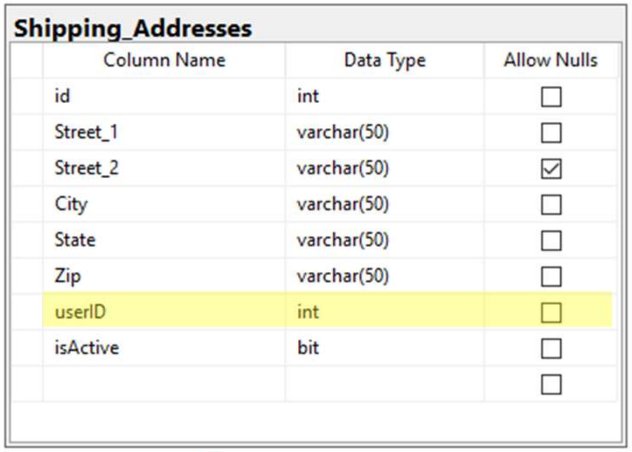

OMG!

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	□
		□

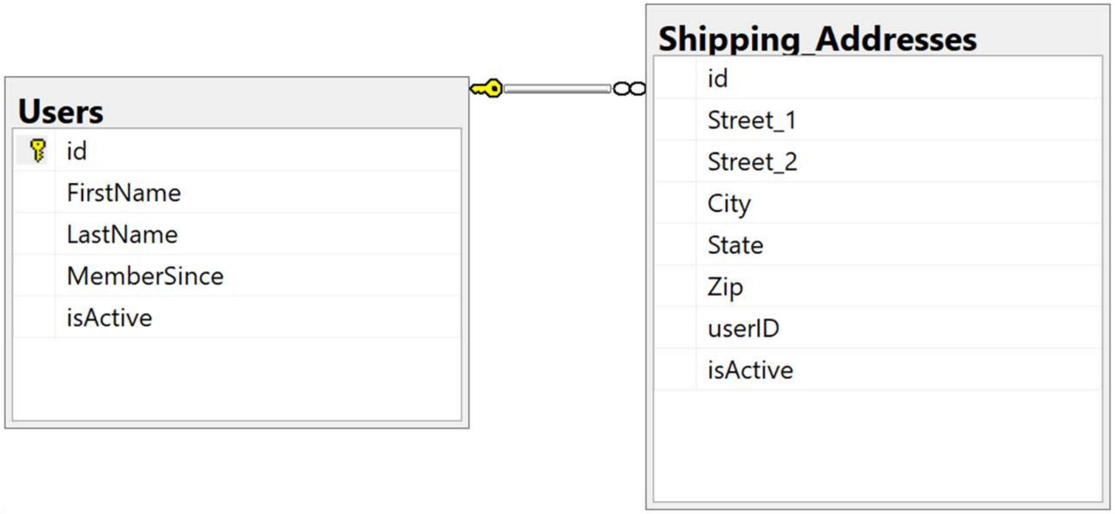
Primary key columns are columns that hold a value that is unique for every row in that table.

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

ELEVATE YOURSELF

ELEVATE YOURSELF

CARDINALITY



Users

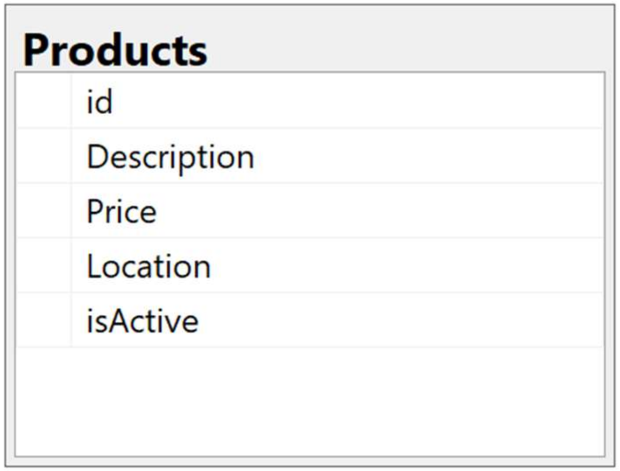
id
FirstName
LastName
MemberSince
isActive

Shipping_Addresses

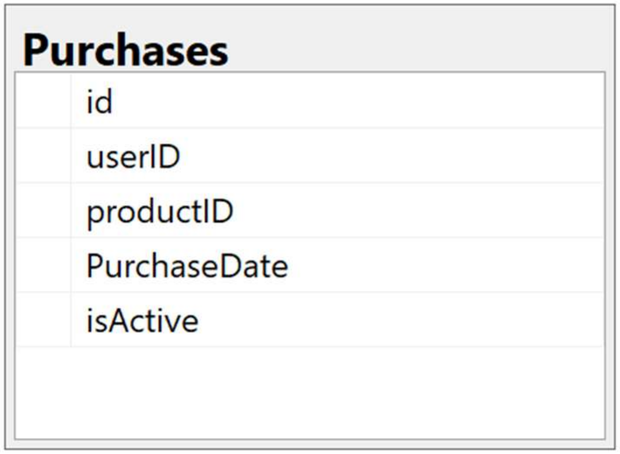
id
Street_1
Street_2
City
State
Zip
userID
isActive

ONE TO MANY

CARDINALITY

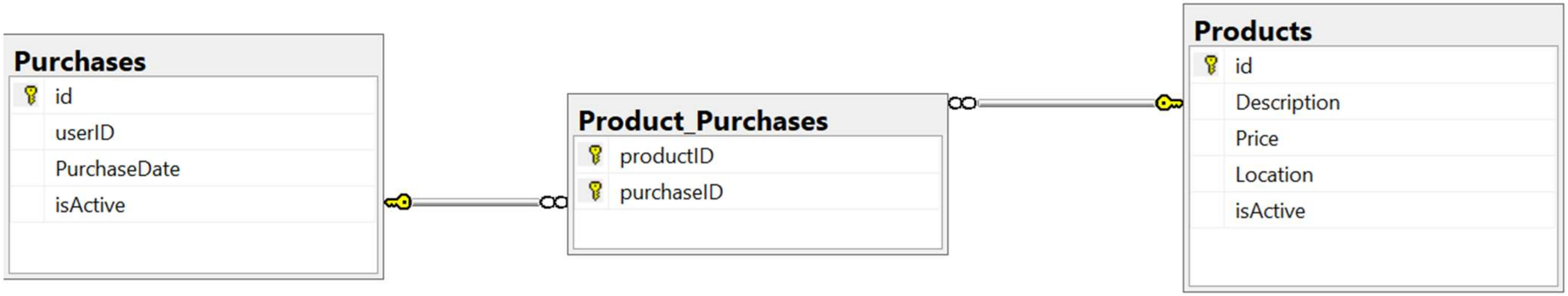


Products
id
Description
Price
Location
isActive



Purchases
id
userID
productID
PurchaseDate
isActive

CARDINALITY



Purchases

- id
- userID
- PurchaseDate
- isActive

Product_Purchases

- productID
- purchaseID

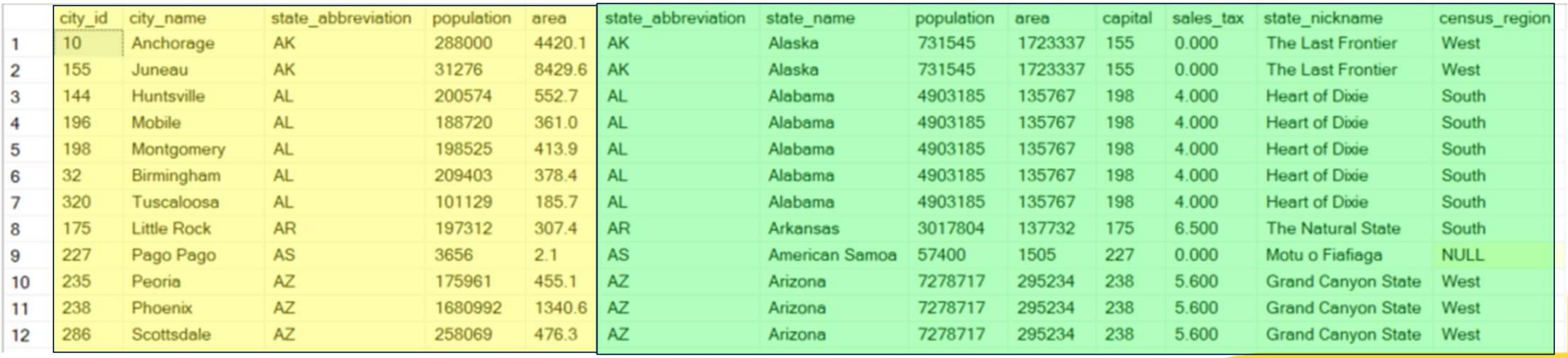
Products

- id
- Description
- Price
- Location
- isActive

MANY TO MANY

JOINS

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

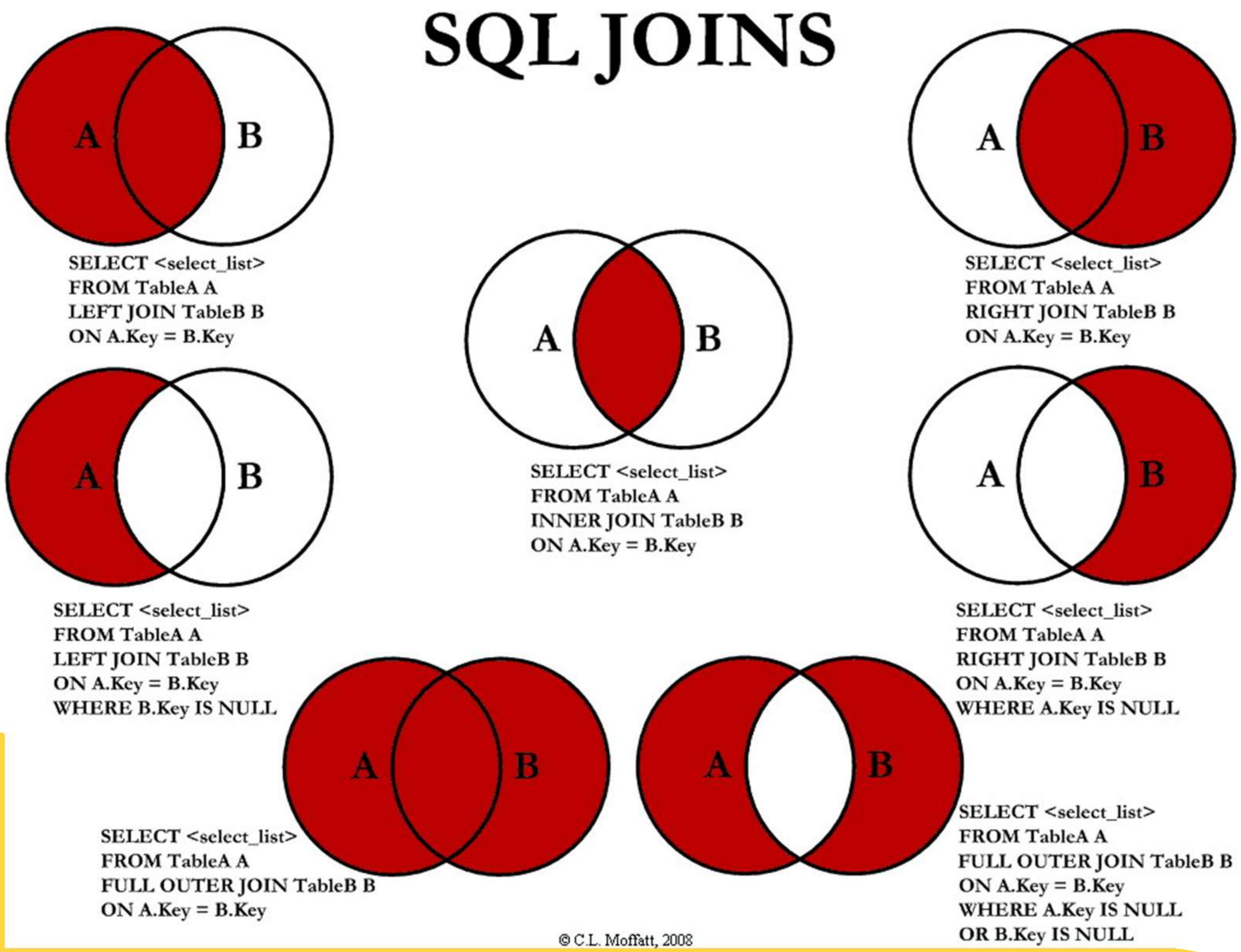


	city_id	city_name	state_abbreviation	population	area	state_abbreviation	state_name	population	area	capital	sales_tax	state_nickname	census_region
1	10	Anchorage	AK	288000	4420.1	AK	Alaska	731545	1723337	155	0.000	The Last Frontier	West
2	155	Juneau	AK	31276	8429.6	AK	Alaska	731545	1723337	155	0.000	The Last Frontier	West
3	144	Huntsville	AL	200574	552.7	AL	Alabama	4903185	135767	198	4.000	Heart of Dixie	South
4	196	Mobile	AL	188720	361.0	AL	Alabama	4903185	135767	198	4.000	Heart of Dixie	South
5	198	Montgomery	AL	198525	413.9	AL	Alabama	4903185	135767	198	4.000	Heart of Dixie	South
6	32	Birmingham	AL	209403	378.4	AL	Alabama	4903185	135767	198	4.000	Heart of Dixie	South
7	320	Tuscaloosa	AL	101129	185.7	AL	Alabama	4903185	135767	198	4.000	Heart of Dixie	South
8	175	Little Rock	AR	197312	307.4	AR	Arkansas	3017804	137732	175	6.500	The Natural State	South
9	227	Pago Pago	AS	3656	2.1	AS	American Samoa	57400	1505	227	0.000	Motu o Fiafiaga	NULL
10	235	Peoria	AZ	175961	455.1	AZ	Arizona	7278717	295234	238	5.600	Grand Canyon State	West
11	238	Phoenix	AZ	1680992	1340.6	AZ	Arizona	7278717	295234	238	5.600	Grand Canyon State	West
12	286	Scottsdale	AZ	258069	476.3	AZ	Arizona	7278717	295234	238	5.600	Grand Canyon State	West

select * from city

select * from state

ELEVATE YOURSELF



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

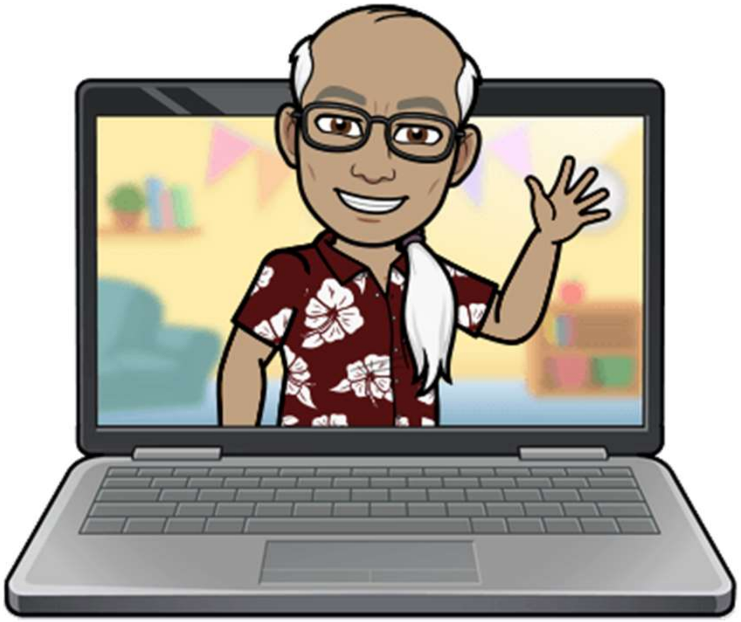
UNIONS

- 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