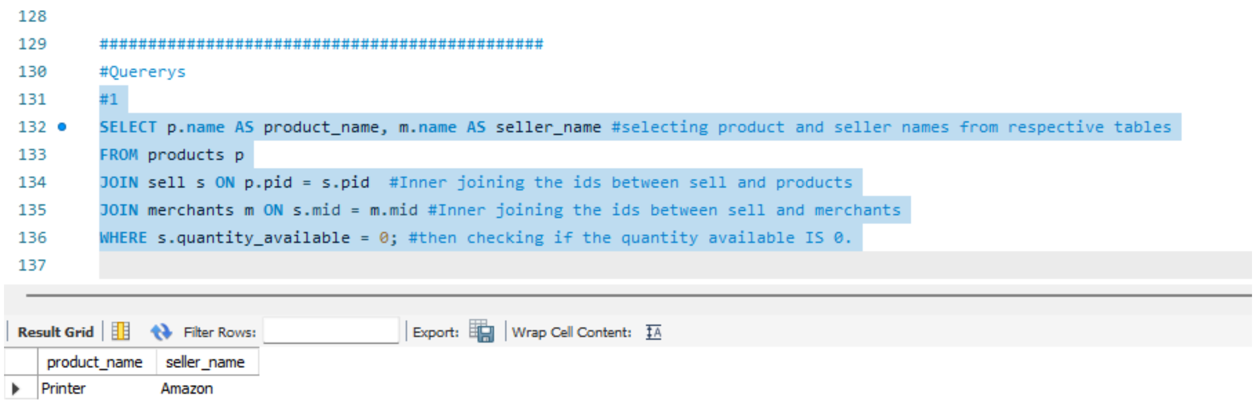
Title Page

- **Title:** DB Assignment 3
- **Your Name:** Antonio Cima
- **Date:** 10/11/2024

Query 1

- We need to find out the product and the seller of products that are NOT in stock (have 0 quantity)
- We do this by joining the ids together to check that the products and the sellers are valid. And then ultimately checking in the sellers table, quantity available if the product_quantity is = 0.

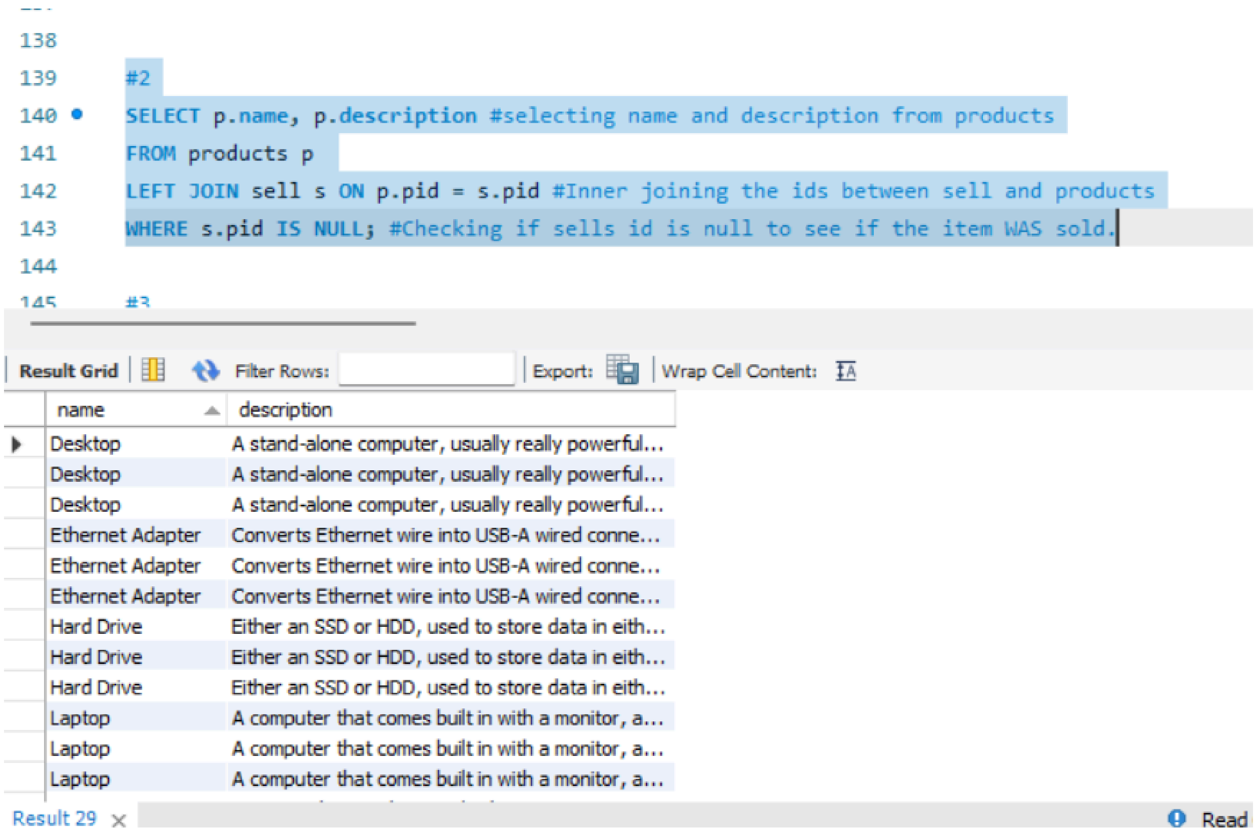


```sql

###############################################
#Quererys
#1
SELECT p.name AS product_name, m.name AS seller_name #selecting product and seller names from respective tables
FROM products p
JOIN sell s ON p.pid = s.pid  #Inner joining the ids between sell and products
JOIN merchants m ON s.mid = m.mid #Inner joining the ids between sell and merchants
WHERE s.quantity_available = 0; #then checking if the quantity available IS 0.

```

product_name	seller_name
Printer	Amazon

Query 2

- We need to list products and also list their descriptions of products that were not sold.
- We do this by doing a very similar approach to query1 by comparing IDs with product and sell then checking if s.pid is null, for if it is, that means the product was never sold.

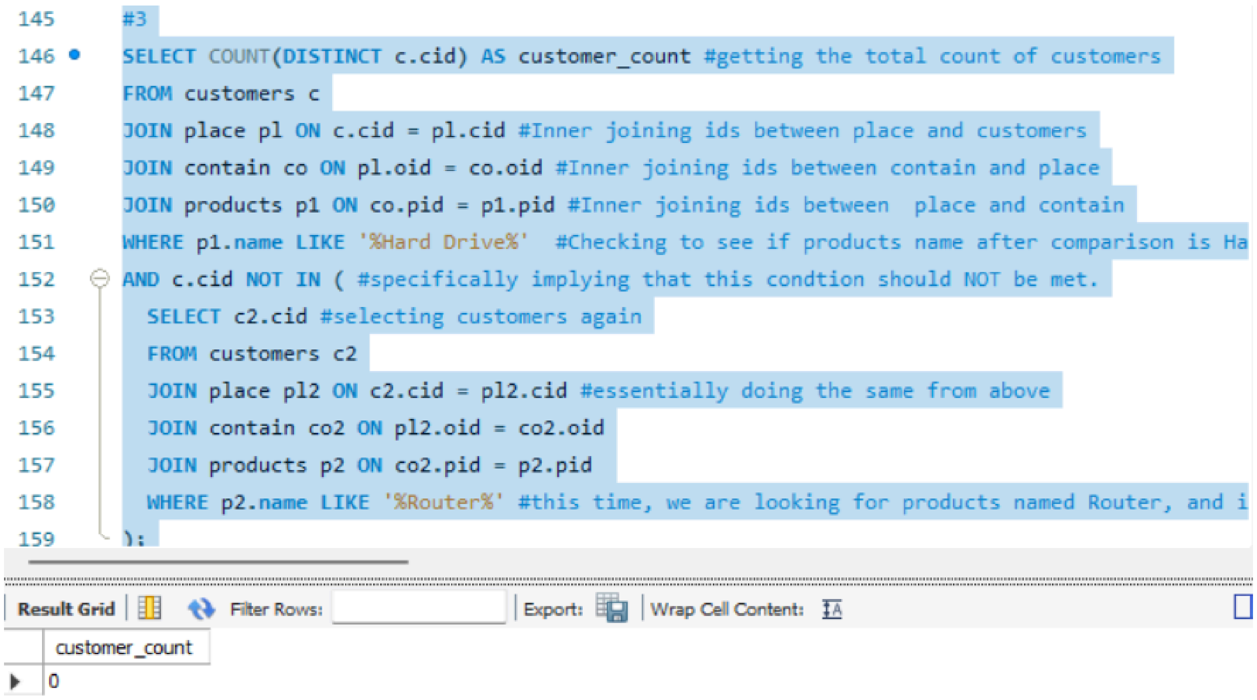


```sql

#2
SELECT p.name, p.description #selecting name and description from products
FROM products p
LEFT JOIN sell s ON p.pid = s.pid #Inner joining the ids between sell and products
WHERE s.pid IS NULL; #Checking if sells id is null to see if the item WAS sold.

#3
```

name	description
Desktop	A stand-alone computer, usually really powerful...
Desktop	A stand-alone computer, usually really powerful...
Desktop	A stand-alone computer, usually really powerful...
Ethernet Adapter	Converts Ethernet wire into USB-A wired conne...
Ethernet Adapter	Converts Ethernet wire into USB-A wired conne...
Ethernet Adapter	Converts Ethernet wire into USB-A wired conne...
Hard Drive	Either an SSD or HDD, used to store data in eith...
Hard Drive	Either an SSD or HDD, used to store data in eith...
Hard Drive	Either an SSD or HDD, used to store data in eith...
Laptop	A computer that comes built in with a monitor, a...
Laptop	A computer that comes built in with a monitor, a...
Laptop	A computer that comes built in with a monitor, a...

Query 3

- We need to figure out how many customers bought SATA drives but not any routers
- We do this by tri-comparing place, contain and products' id to guarantee the customer has purchased this router. Then we do roughly the same process of checking the customers' purchases for Router by using the "NOT IN" command, which is essentially a big NOT condition.

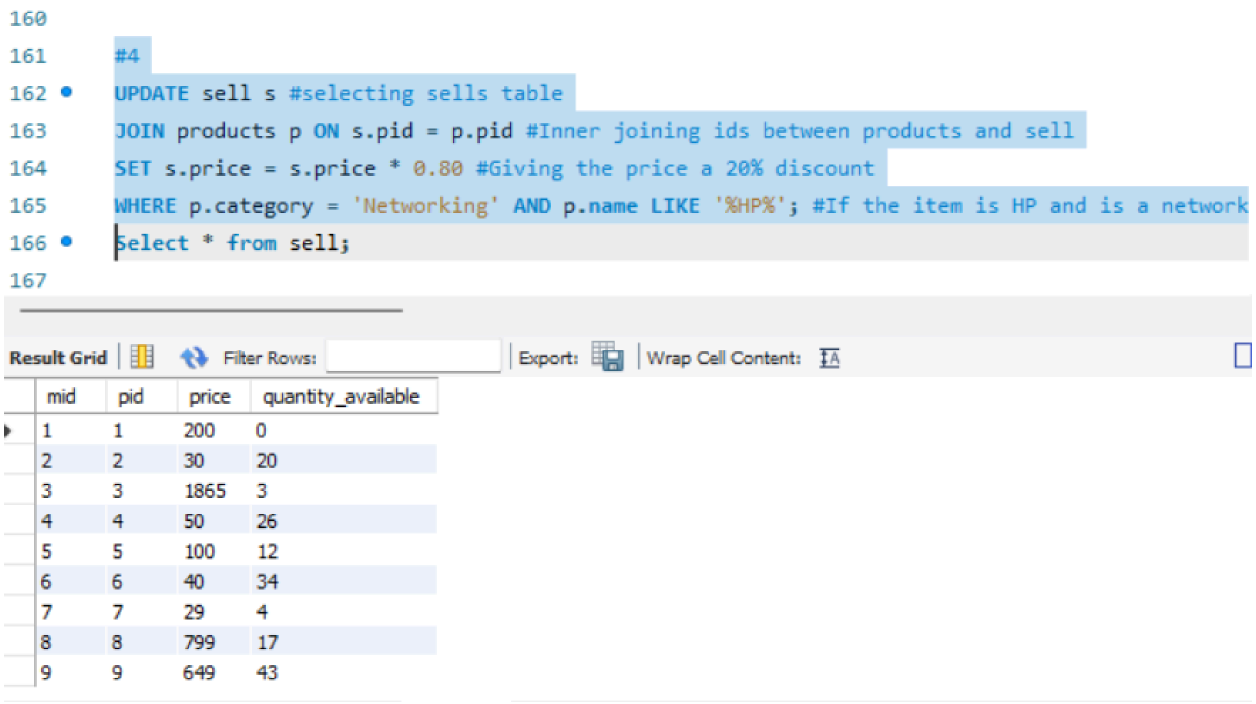


```
#3
SELECT COUNT(DISTINCT c.cid) AS customer_count #getting the total count of customers
FROM customers c
JOIN place pl ON c.cid = pl.cid #Inner joining ids between place and customers
JOIN contain co ON pl.oid = co.oid #Inner joining ids between contain and place
JOIN products p1 ON co.pid = p1.pid #Inner joining ids between  place and contain
WHERE p1.name LIKE '%Hard Drive%'  #Checking to see if products name after comparison is Ha
AND c.cid NOT IN ( #specifically implying that this condtion should NOT be met.
  SELECT c2.cid #selecting customers again
  FROM customers c2
  JOIN place pl2 ON c2.cid = pl2.cid #essentially doing the same from above
  JOIN contain co2 ON pl2.oid = co2.oid
  JOIN products p2 ON co2.pid = p2.pid
  WHERE p2.name LIKE '%Router%' #this time, we are looking for products named Router, and i
);
```

customer_count
0

Query 4

- We just need to update HP products that are networking products with a 20% discount.
- We do this by checking the ids between sell and product, then essentially giving the price a 20% discount by looking at the category and the merchants name.



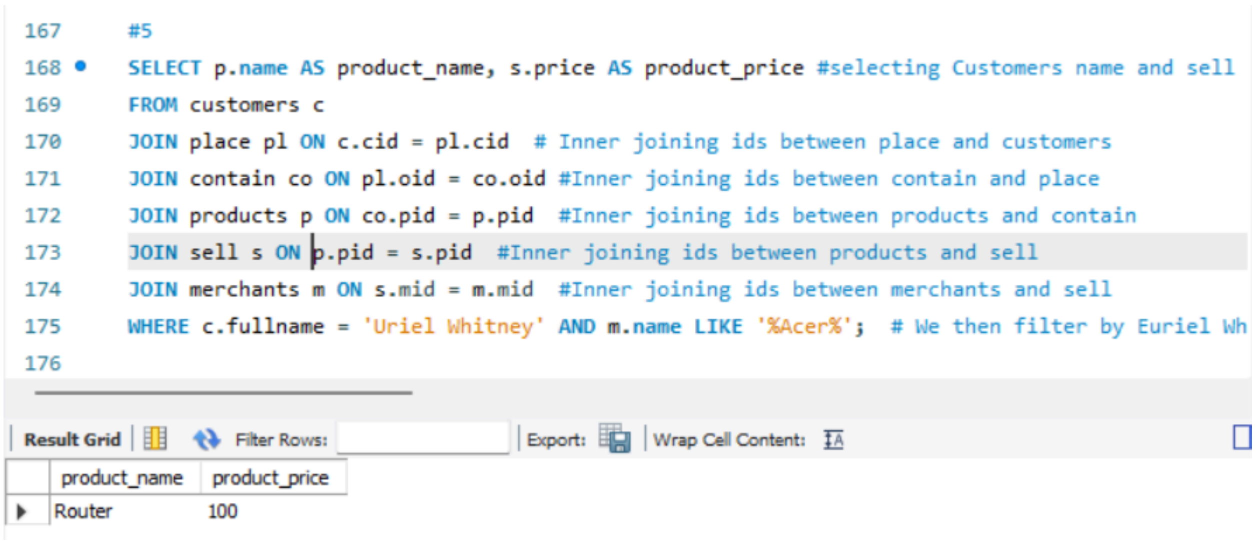
```

#4
UPDATE sell s #selecting sells table
JOIN products p ON s.pid = p.pid #Inner joining ids between products and sell
SET s.price = s.price * 0.80 #Giving the price a 20% discount
WHERE p.category = 'Networking' AND p.name LIKE '%HP%'; #If the item is HP and is a network
select * from sell;

```

mid	pid	price	quantity_available
1	1	200	0
2	2	30	20
3	3	1865	3
4	4	50	26
5	5	100	12
6	6	40	34
7	7	29	4
8	8	799	17
9	9	649	43

-

Query 5

- We need to find out what Uriel Whitney ordered from Acer (if anything)
- We do this by doing a lot of joining together, we need to do this in order to match the product, the price, the merchant, and the customer who bought the product together, then at the end, we filter by Uriel and the Acer merchant



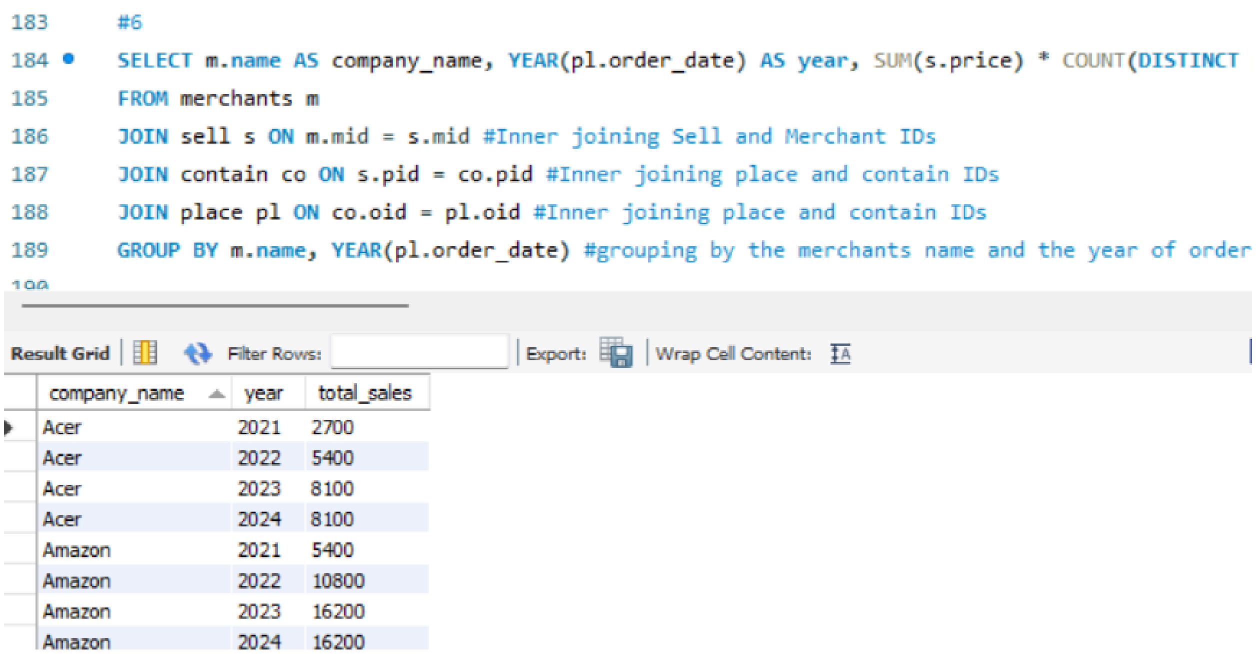
```sql
#5
SELECT p.name AS product_name, s.price AS product_price #selecting Customers name and sell
FROM customers c
JOIN place pl ON c.cid = pl.cid  # Inner joining ids between place and customers
JOIN contain co ON pl.oid = co.oid #Inner joining ids between contain and place
JOIN products p ON co.pid = p.pid  #Inner joining ids between products and contain
JOIN sell s ON p.pid = s.pid  #Inner joining ids between products and sell
JOIN merchants m ON s.mid = m.mid  #Inner joining ids between merchants and sell
WHERE c.fullname = 'Uriel Whitney' AND m.name LIKE '%Acer%';  # We then filter by Euriel Wh
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

product_name	product_price
Router	100

Query 6

- We need to figure out the yearly salary of these companies
- We start by getting the name, the year, and the price per unique order, then we join to verify, and then we group merchants and years together by the date.



```sql
#6
SELECT m.name AS company_name, YEAR(pl.order_date) AS year, SUM(s.price) * COUNT(DISTINCT
FROM merchants m
JOIN sell s ON m.mid = s.mid #Inner joining Sell and Merchant IDs
JOIN contain co ON s.pid = co.pid #Inner joining place and contain IDs
JOIN place pl ON co.oid = pl.oid #Inner joining place and contain IDs
GROUP BY m.name, YEAR(pl.order_date) #grouping by the merchants name and the year of order
```

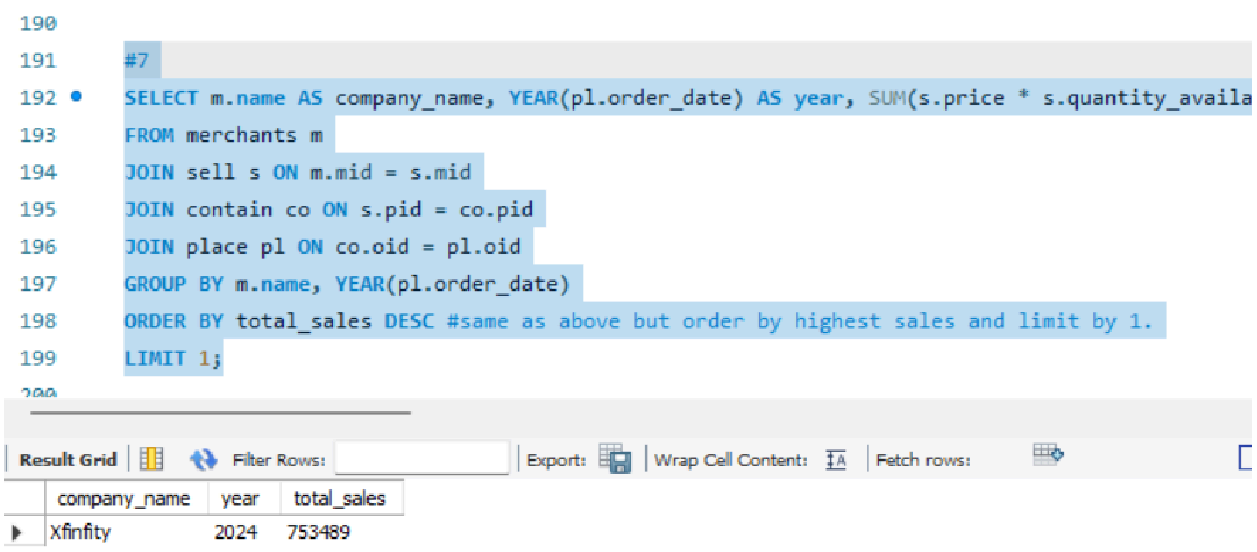
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

company_name	year	total_sales
Acer	2021	2700
Acer	2022	5400
Acer	2023	8100
Acer	2024	8100
Amazon	2021	5400
Amazon	2022	10800
Amazon	2023	16200
Amazon	2024	16200

-

Query 7

- We need to figure out the company with the highest sales and what year, easy.
- Same as Querery 6, but now we order by total sales and limit by 1 to get our answer.



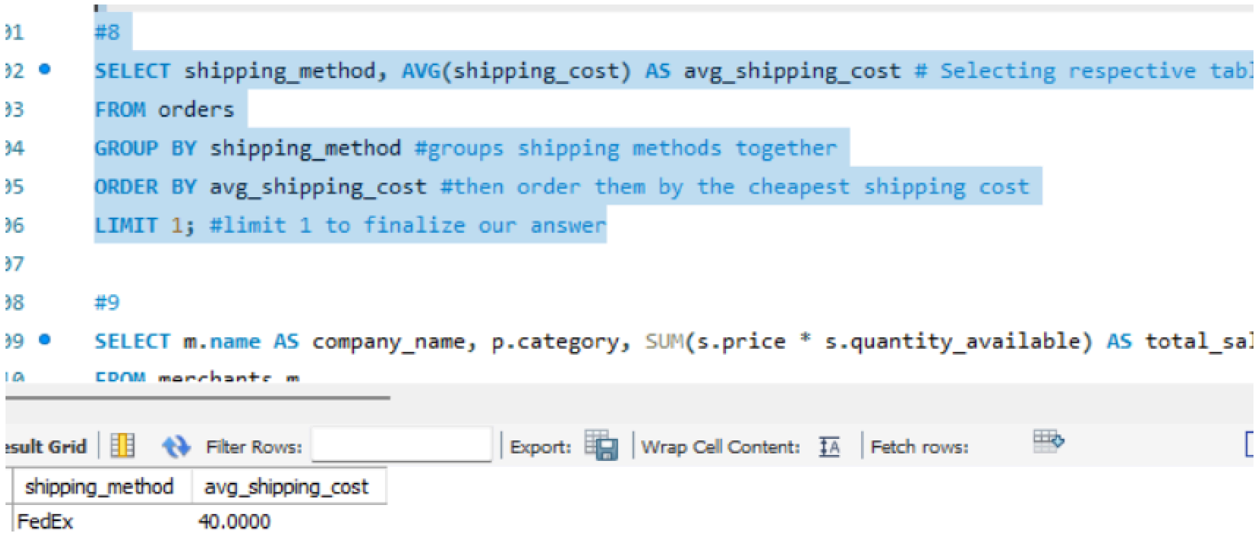
```
#7
SELECT m.name AS company_name, YEAR(pl.order_date) AS year, SUM(s.price * s.quantity_availa
FROM merchants m
JOIN sell s ON m.mid = s.mid
JOIN contain co ON s.pid = co.pid
JOIN place pl ON co.oid = pl.oid
GROUP BY m.name, YEAR(pl.order_date)
ORDER BY total_sales DESC #same as above but order by highest sales and limit by 1.
LIMIT 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

```
company_name  year  total_sales
Xfinfity      2024  753489
```

Query 8

- We need to find out on average which shipping method is the cheapest
- Very simple, we just get our shipping methods and we just order by low to high and limit to 1 answer, then we have our answer!



```
#8
SELECT shipping_method, AVG(shipping_cost) AS avg_shipping_cost # Selecting respective tab
FROM orders
GROUP BY shipping_method #groups shipping methods together
ORDER BY avg_shipping_cost #then order them by the cheapest shipping cost
LIMIT 1; #limit 1 to finalize our answer

#9
SELECT m.name AS company_name, p.category, SUM(s.price * s.quantity_available) AS total_sa
FROM merchants m
```

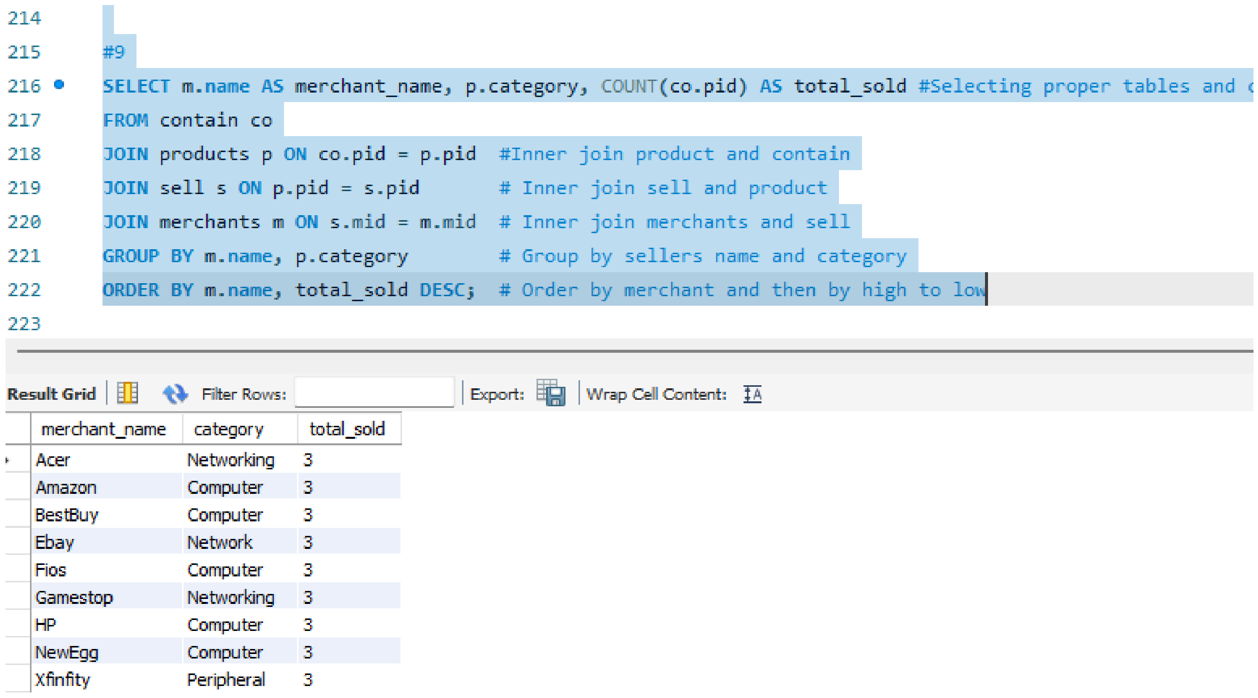
sult Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

```
shipping_method  avg_shipping_cost
FedEx            40.0000
```

-

Query 9

- What is the best sold category for each company?
- We can do this by getting a total count on all items purchased by customers in contain, then we inner join the ids together to then group the merchants and category, to then combine and show the total amount of items sold.



```
#9
SELECT m.name AS merchant_name, p.category, COUNT(co.pid) AS total_sold #Selecting proper tables and c
FROM contain co
JOIN products p ON co.pid = p.pid  #Inner join product and contain
JOIN sell s ON p.pid = s.pid       # Inner join sell and product
JOIN merchants m ON s.mid = m.mid  # Inner join merchants and sell
GROUP BY m.name, p.category        # Group by sellers name and category
ORDER BY m.name, total_sold DESC;  # Order by merchant and then by high to low
```

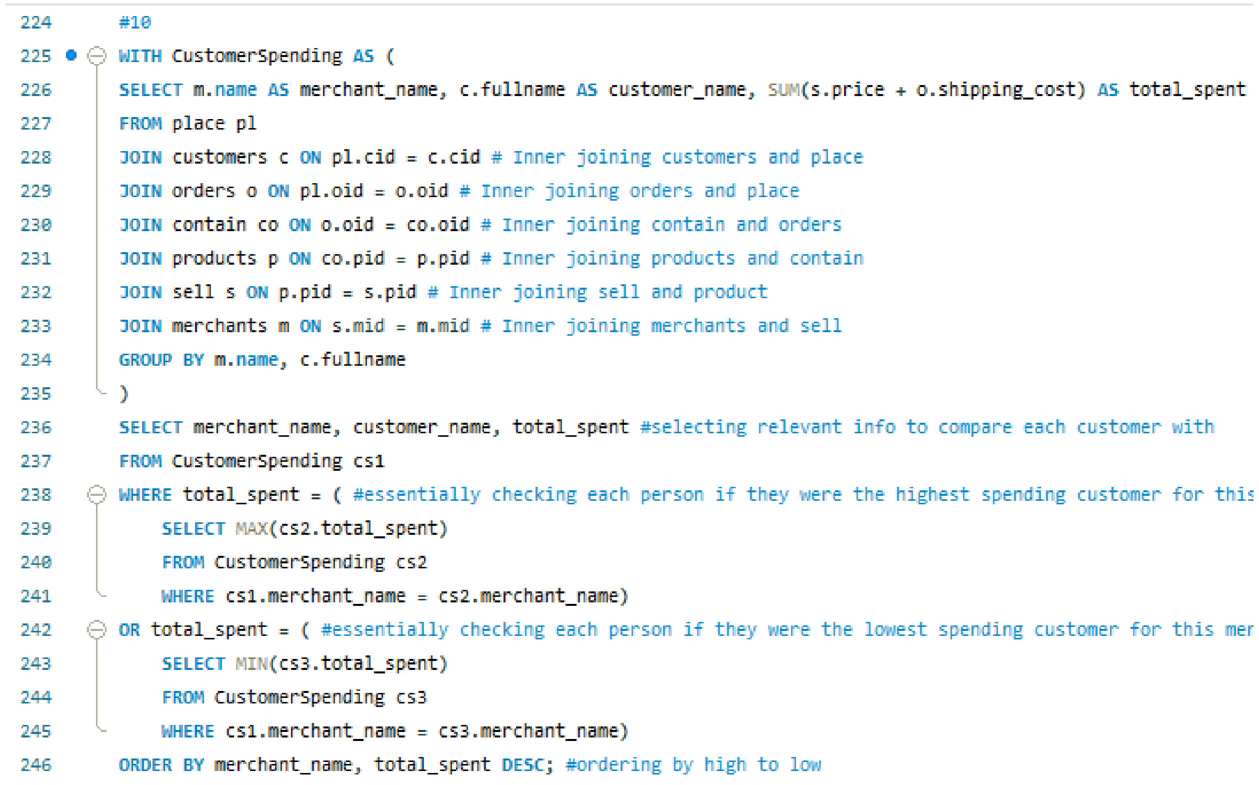
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

merchant_name	category	total_sold
Acer	Networking	3
Amazon	Computer	3
BestBuy	Computer	3
Ebay	Network	3
Fios	Computer	3
Gamestop	Networking	3
HP	Computer	3
NewEgg	Computer	3
Xfinfity	Peripheral	3

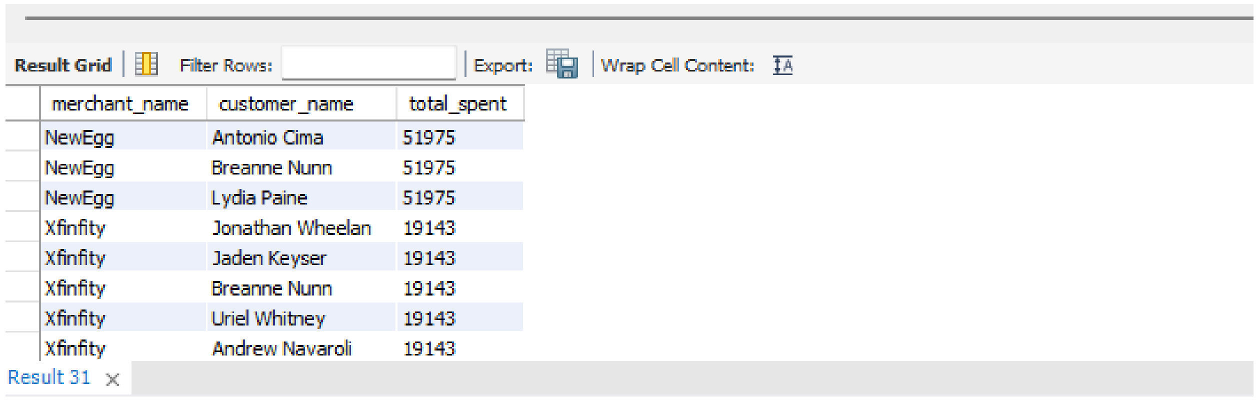
-

Query 10

- For each company, find out which customers have spent the most and the least amounts.
- We do this by first getting the relevant information from the tables, along with getting the sum of product price and shipping cost combined. We then join a bunch to verify the ids, then we group each person by their name. We then put that all into a temporary table known as Customer Spending to use later. NEXT, we then grab that information FROM customer spending to then identify the highest and lowest spender of each company, then we cycle through and we did it! (Note, mines is a tiny bit buggy just due to how the data is inserted, I had a really hard time trying to use place table to insert 2 foreign keys, so I did a lot of Unioning and that seemed to work, but every customer had bought the same amount of products, so that's why there's a bunch of customers there)



```sql
#10
WITH CustomerSpending AS (
SELECT m.name AS merchant_name, c.fullname AS customer_name, SUM(s.price + o.shipping_cost) AS total_spent
FROM place pl
JOIN customers c ON pl.cid = c.cid # Inner joining customers and place
JOIN orders o ON pl.oid = o.oid # Inner joining orders and place
JOIN contain co ON o.oid = co.oid # Inner joining contain and orders
JOIN products p ON co.pid = p.pid # Inner joining products and contain
JOIN sell s ON p.pid = s.pid # Inner joining sell and product
JOIN merchants m ON s.mid = m.mid # Inner joining merchants and sell
GROUP BY m.name, c.fullname
)
SELECT merchant_name, customer_name, total_spent #selecting relevant info to compare each customer with
FROM CustomerSpending cs1
WHERE total_spent = ( #essentially checking each person if they were the highest spending customer for this
    SELECT MAX(cs2.total_spent)
    FROM CustomerSpending cs2
    WHERE cs1.merchant_name = cs2.merchant_name)
OR total_spent = ( #essentially checking each person if they were the lowest spending customer for this mer
    SELECT MIN(cs3.total_spent)
    FROM CustomerSpending cs3
    WHERE cs1.merchant_name = cs3.merchant_name)
ORDER BY merchant_name, total_spent DESC; #ordering by high to low
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

merchant_name	customer_name	total_spent
NewEgg	Antonio Cima	51975
NewEgg	Breanne Nunn	51975
NewEgg	Lydia Paine	51975
Xfinfity	Jonathan Wheelan	19143
Xfinfity	Jaden Keyser	19143
Xfinfity	Breanne Nunn	19143
Xfinfity	Uriel Whitney	19143
Xfinfity	Andrew Navaroli	19143

Result 31 ×

-