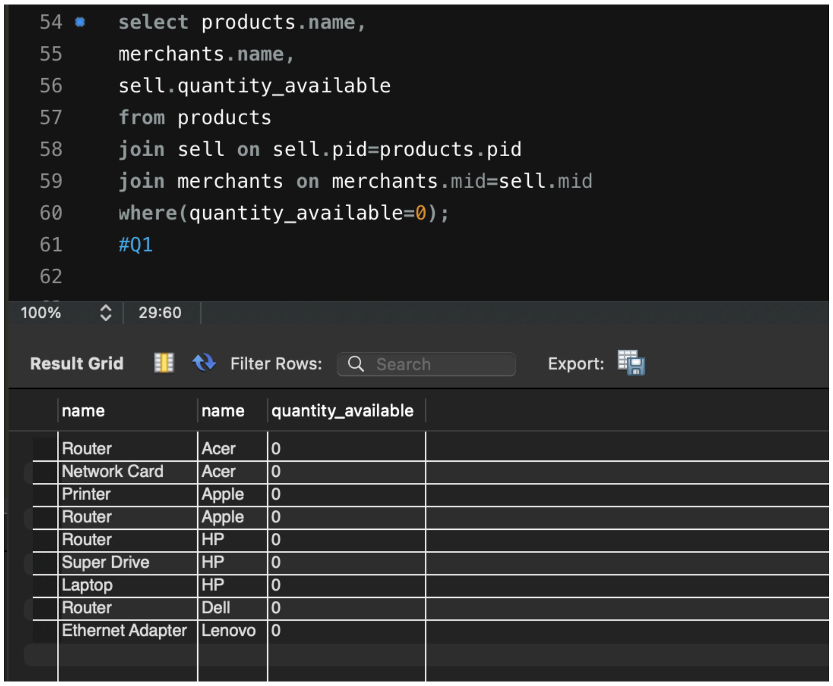
SQL Documentation



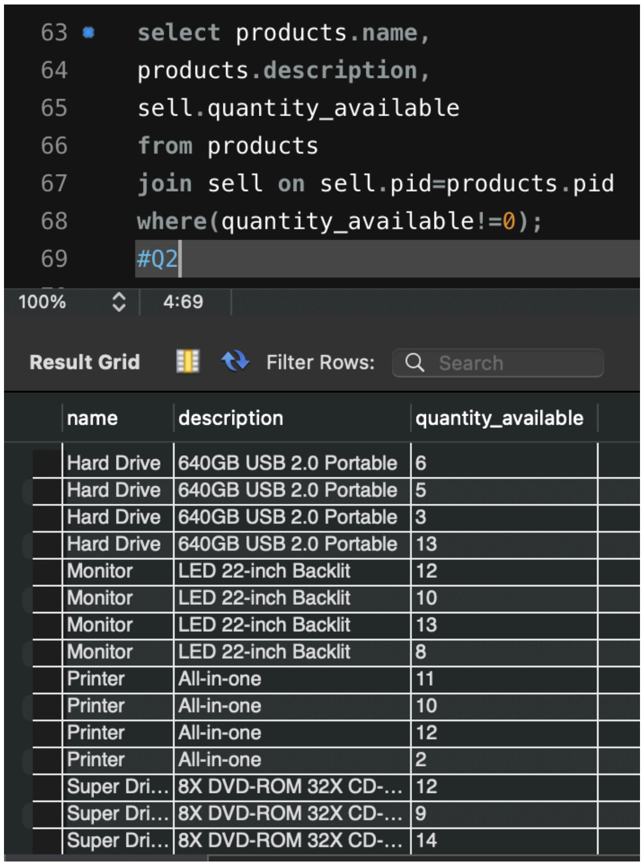
```
select products.name,
merchants.name,
sell.quantity_available
from products
join sell on sell.pid=products.pid
join merchants on merchants.mid=sell.mid
where(quantity_available=0);
#Q1
```

Result Grid

name	name	quantity_available
Router	Acer	0
Network Card	Acer	0
Printer	Apple	0
Router	Apple	0
Router	HP	0
Super Drive	HP	0
Laptop	HP	0
Router	Dell	0
Ethernet Adapter	Lenovo	0

Q1:
This query selects the key column names and joins together the tables (sell & merchants) using **ON** on the foreign keys. I used where to filter what rows have a 0 quantity.

Q2: This query selects names and the description of the products after joining together the sell table. I joined sell in order to get the quantity available column. I used where to set the join condition to anything other than 0. There are 123 rows.



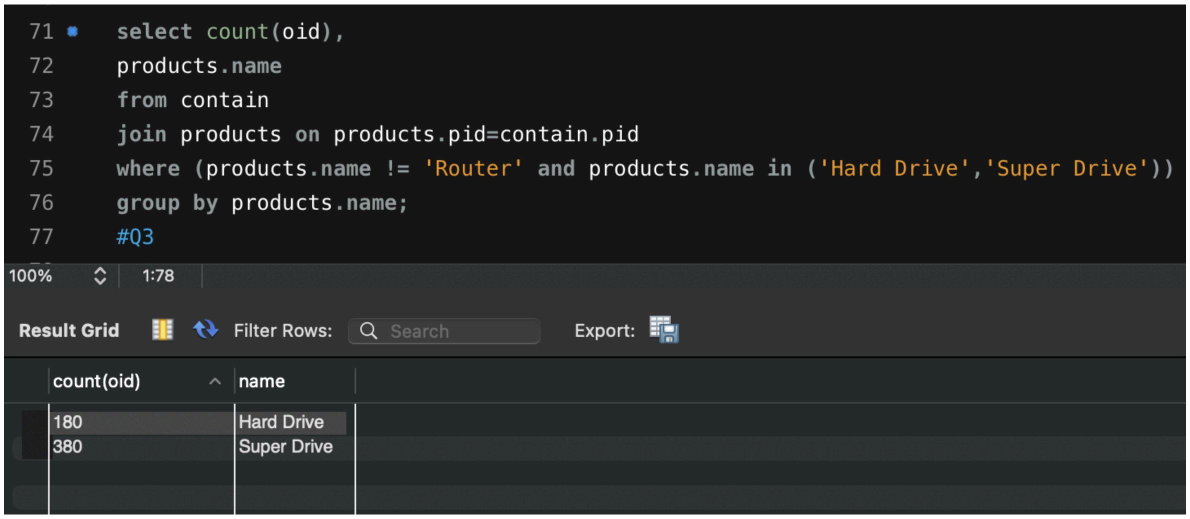
```
select products.name,
products.description,
sell.quantity_available
from products
join sell on sell.pid=products.pid
where(quantity_available!=0);
#Q2
```

```
100%    4:69
```

Result Grid   Filter Rows: Search

name	description	quantity_available
Hard Drive	640GB USB 2.0 Portable	6
Hard Drive	640GB USB 2.0 Portable	5
Hard Drive	640GB USB 2.0 Portable	3
Hard Drive	640GB USB 2.0 Portable	13
Monitor	LED 22-inch Backlit	12
Monitor	LED 22-inch Backlit	10
Monitor	LED 22-inch Backlit	13
Monitor	LED 22-inch Backlit	8
Printer	All-in-one	11
Printer	All-in-one	10
Printer	All-in-one	12
Printer	All-in-one	2
Super Dri…	8X DVD-ROM 32X CD-…	12
Super Dri…	8X DVD-ROM 32X CD-…	9
Super Dri…	8X DVD-ROM 32X CD-…	14

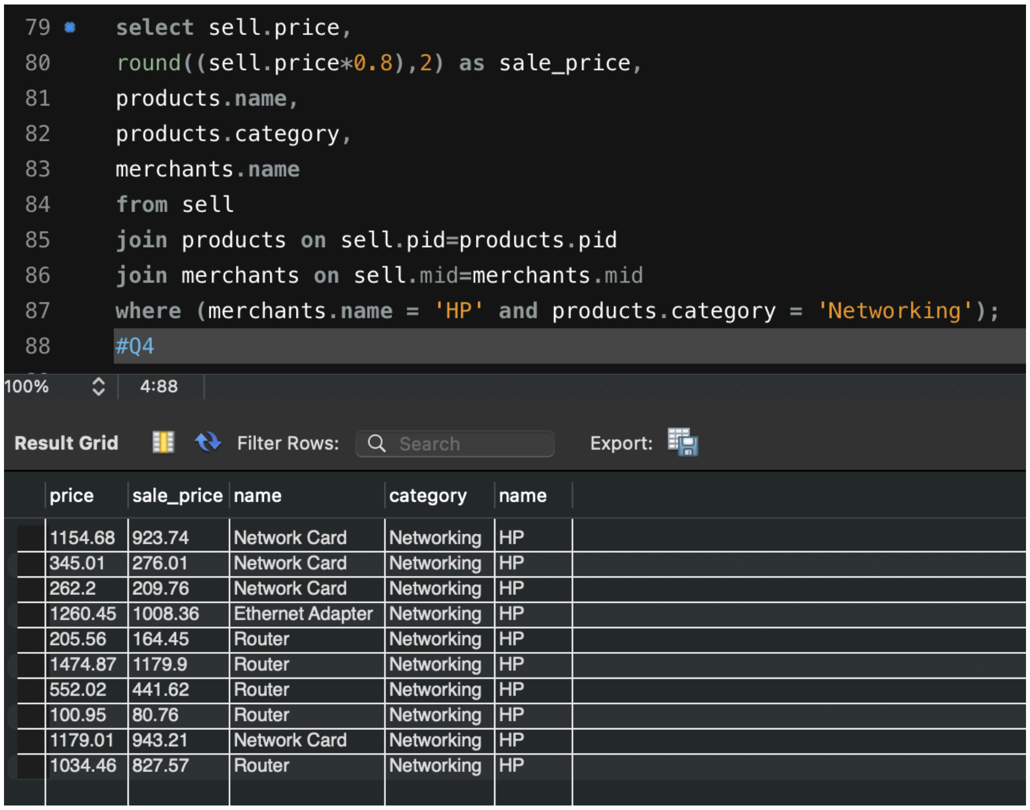
Q3: This query selects the order count (oid) of customers who bought SATA drives && did not buy a router. I joined the products table using **on** and did products.pid=contain.pid. My filter is in the where statement. I set the names to grab the SATA drives and not the Router product.



```sql
select count(oid),
products.name
from contain
join products on products.pid=contain.pid
where (products.name != 'Router' and products.name in ('Hard Drive','Super Drive'))
group by products.name;
#Q3
```

count(oid)	name
180	Hard Drive
380	Super Drive

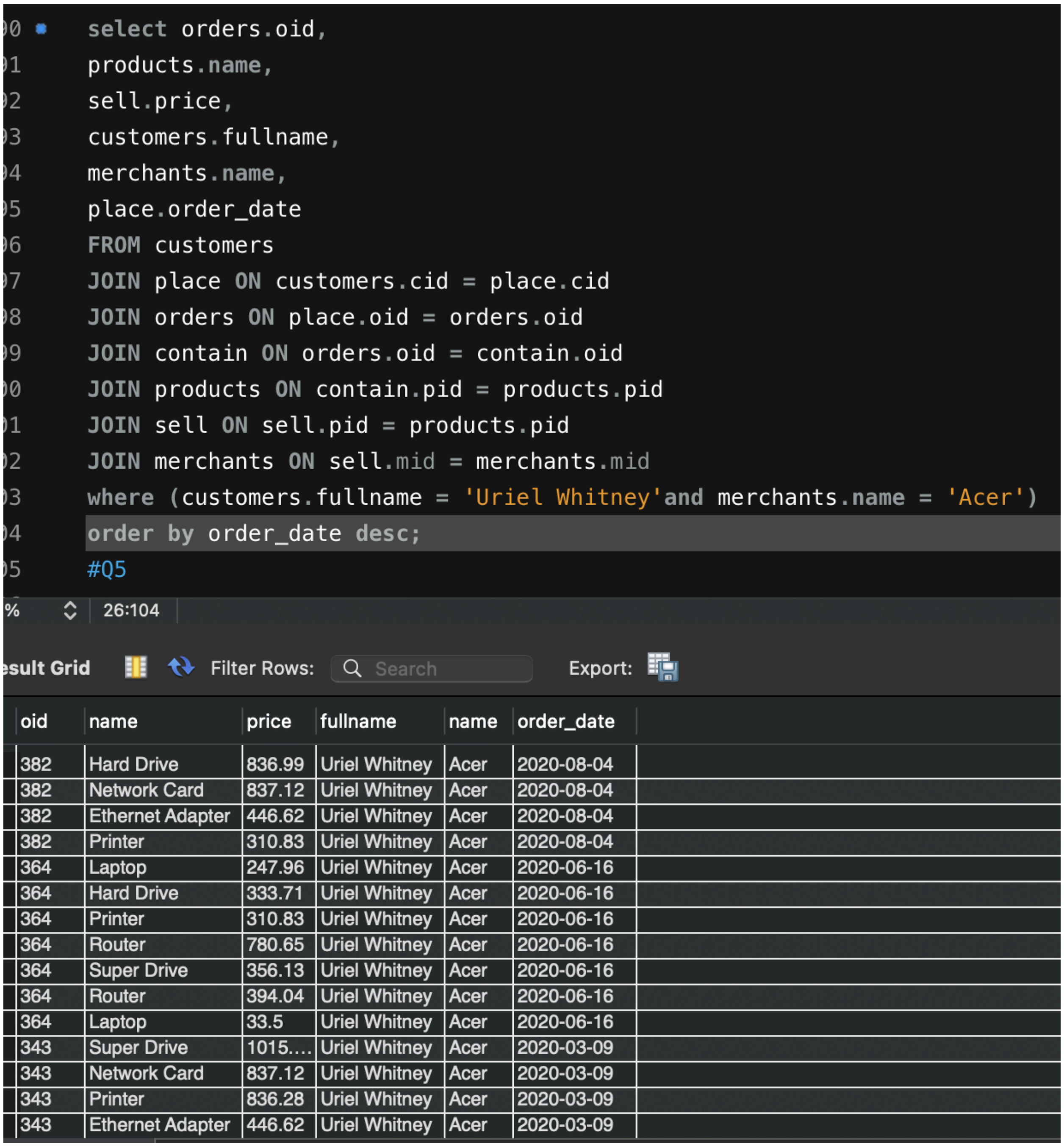
Q4: In this query I used the sell table joined by the products and merchants tables. The sell table has the price attribute, the merchants table has the seller name, and the products table has the actual products tied to the price. To get the new price I just multiplied the sell.price attribute by 0.8 and put it inside a round function also. I renamed the column sale_price.



```sql
select sell.price,
round((sell.price*0.8),2) as sale_price,
products.name,
products.category,
merchants.name
from sell
join products on sell.pid=products.pid
join merchants on sell.mid=merchants.mid
where (merchants.name = 'HP' and products.category = 'Networking');
#Q4
```

price	sale_price	name	category	name
1154.68	923.74	Network Card	Networking	HP
345.01	276.01	Network Card	Networking	HP
262.2	209.76	Network Card	Networking	HP
1260.45	1008.36	Ethernet Adapter	Networking	HP
205.56	164.45	Router	Networking	HP
1474.87	1179.9	Router	Networking	HP
552.02	441.62	Router	Networking	HP
100.95	80.76	Router	Networking	HP
1179.01	943.21	Network Card	Networking	HP
1034.46	827.57	Router	Networking	HP

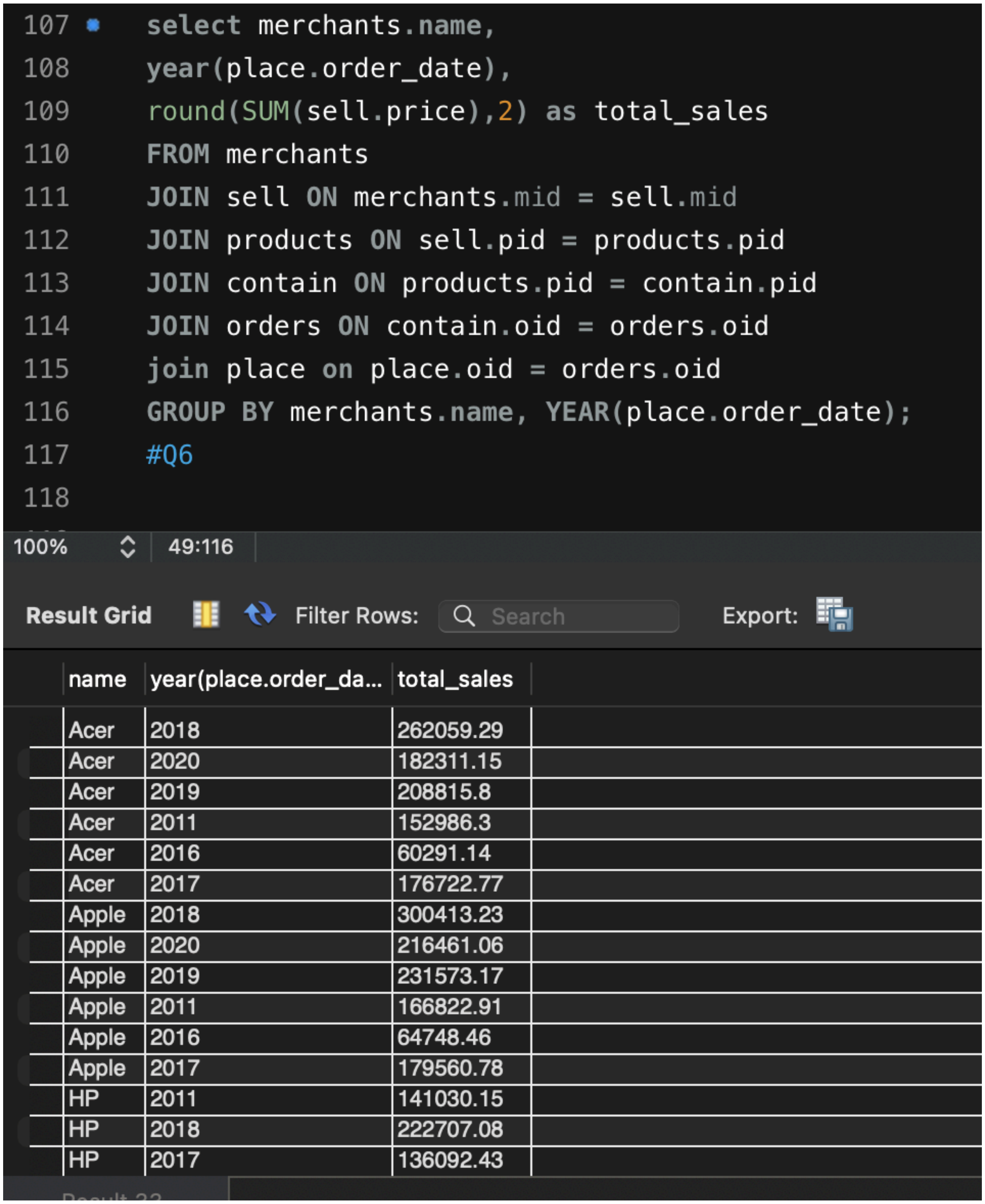
Q5: This query was quite simple. I selected the product names, prices, customer, names, and order_dates. I then joined all the necessary tables together and used to a where condition to filter on Uriel Whitney's names and the Acer company. This got me the history of her purchases with the Acer company.



```sql
select orders.oid,
products.name,
sell.price,
customers.fullname,
merchants.name,
place.order_date
FROM customers
JOIN place ON customers.cid = place.cid
JOIN orders ON place.oid = orders.oid
JOIN contain ON orders.oid = contain.oid
JOIN products ON contain.pid = products.pid
JOIN sell ON sell.pid = products.pid
JOIN merchants ON sell.mid = merchants.mid
where (customers.fullname = 'Uriel Whitney'and merchants.name = 'Acer')
order by order_date desc;
#Q5
```

oid	name	price	fullname	name	order_date
382	Hard Drive	836.99	Uriel Whitney	Acer	2020-08-04
382	Network Card	837.12	Uriel Whitney	Acer	2020-08-04
382	Ethernet Adapter	446.62	Uriel Whitney	Acer	2020-08-04
382	Printer	310.83	Uriel Whitney	Acer	2020-08-04
364	Laptop	247.96	Uriel Whitney	Acer	2020-06-16
364	Hard Drive	333.71	Uriel Whitney	Acer	2020-06-16
364	Printer	310.83	Uriel Whitney	Acer	2020-06-16
364	Router	780.65	Uriel Whitney	Acer	2020-06-16
364	Super Drive	356.13	Uriel Whitney	Acer	2020-06-16
364	Router	394.04	Uriel Whitney	Acer	2020-06-16
364	Laptop	33.5	Uriel Whitney	Acer	2020-06-16
343	Super Drive	1015.…	Uriel Whitney	Acer	2020-03-09
343	Network Card	837.12	Uriel Whitney	Acer	2020-03-09
343	Printer	836.28	Uriel Whitney	Acer	2020-03-09
343	Ethernet Adapter	446.62	Uriel Whitney	Acer	2020-03-09

Q6: In this query, I connected many tables using joins and utilized the year function to isolate the year associated with the order_date. I took the sum of all the prices of their respective merchant.The table showed the total_sales for each year for each company.



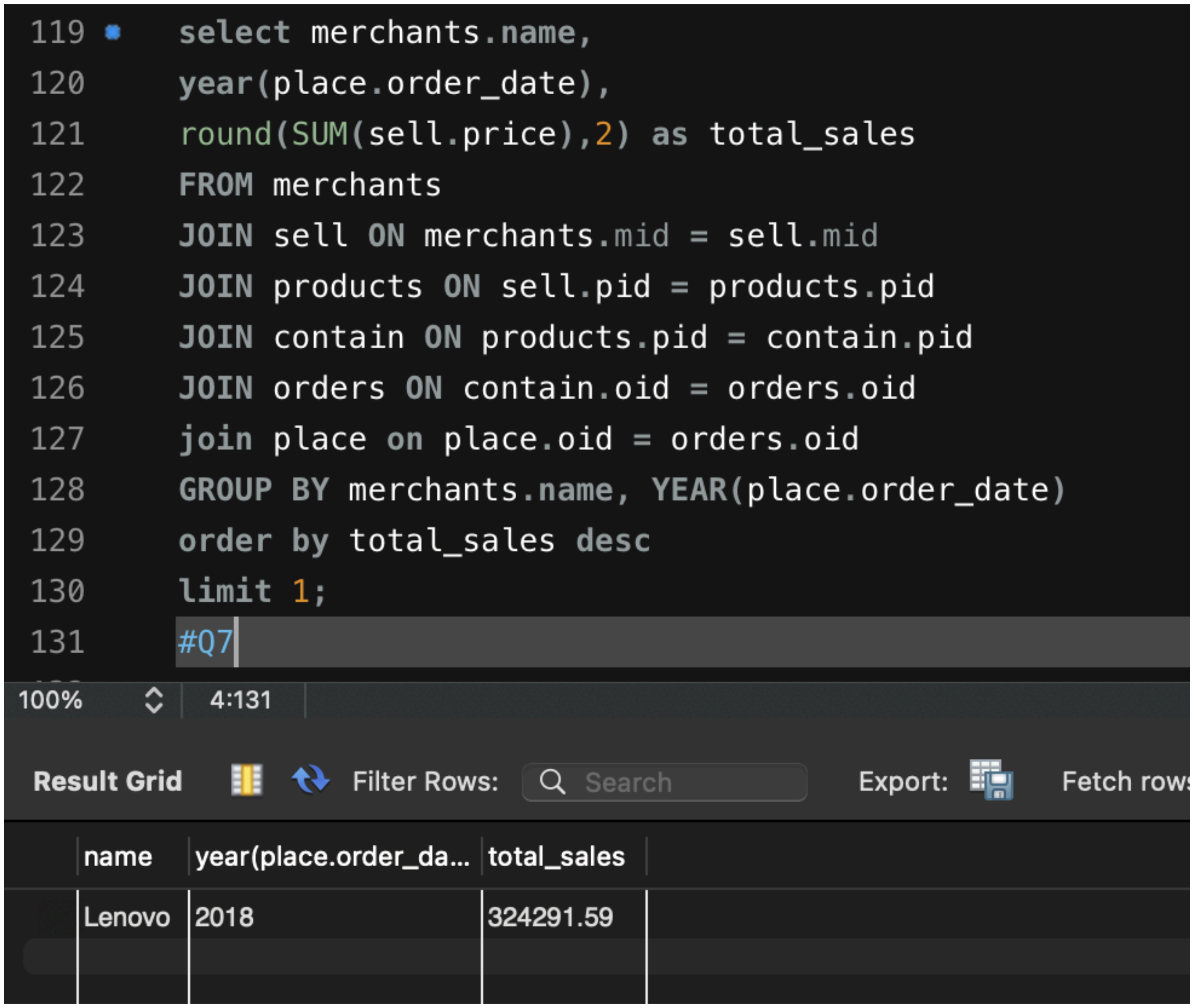
```
select merchants.name,
year(place.order_date),
round(SUM(sell.price),2) as total_sales
FROM merchants
JOIN sell ON merchants.mid = sell.mid
JOIN products ON sell.pid = products.pid
JOIN contain ON products.pid = contain.pid
JOIN orders ON contain.oid = orders.oid
join place on place.oid = orders.oid
GROUP BY merchants.name, YEAR(place.order_date);
#Q6
```

100% 49:116

Result Grid Filter Rows: Search Export:

name	year(place.order_da...	total_sales
Acer	2018	262059.29
Acer	2020	182311.15
Acer	2019	208815.8
Acer	2011	152986.3
Acer	2016	60291.14
Acer	2017	176722.77
Apple	2018	300413.23
Apple	2020	216461.06
Apple	2019	231573.17
Apple	2011	166822.91
Apple	2016	64748.46
Apple	2017	179560.78
HP	2011	141030.15
HP	2018	222707.08
HP	2017	136092.43

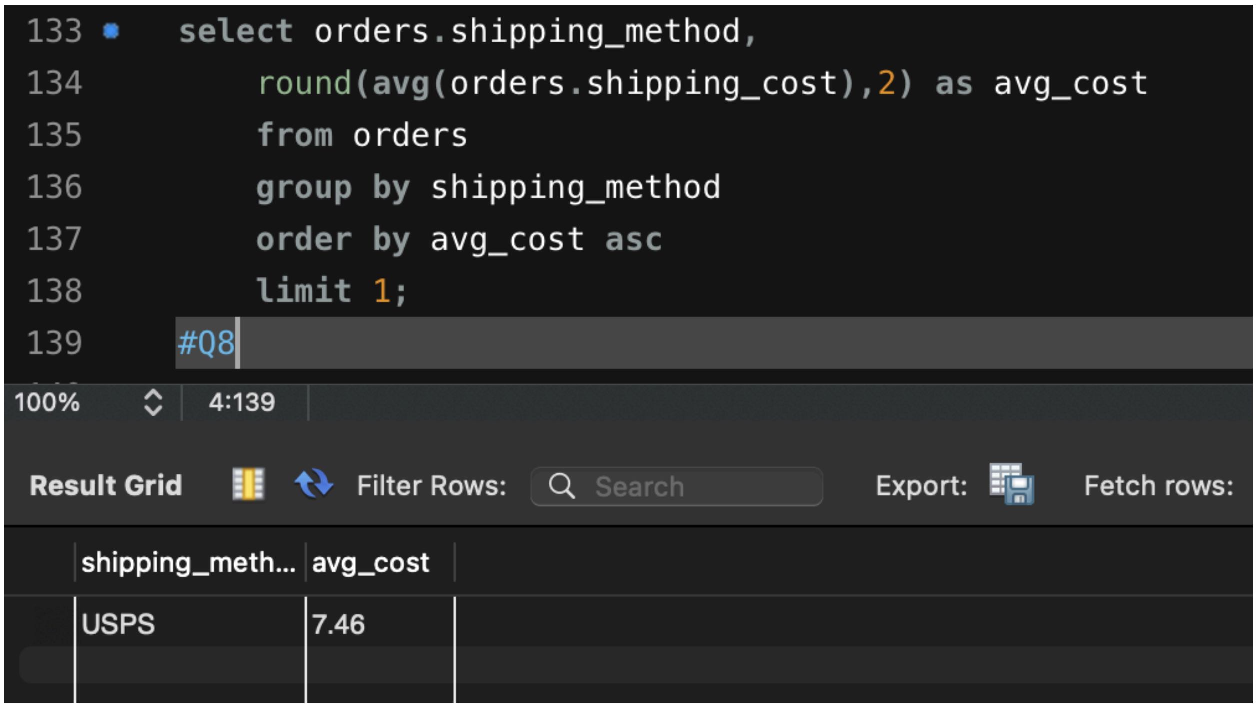
Q7:This query sums up all the order revenue as total_sales by year and orders by desc. I used limit to get the top 1 result. The YEAR function is used here because it isolates the year portion of a date or datetime.



```
select merchants.name,
year(place.order_date),
round(SUM(sell.price),2) as total_sales
FROM merchants
JOIN sell ON merchants.mid = sell.mid
JOIN products ON sell.pid = products.pid
JOIN contain ON products.pid = contain.pid
JOIN orders ON contain.oid = orders.oid
join place on place.oid = orders.oid
GROUP BY merchants.name, YEAR(place.order_date)
order by total_sales desc
limit 1;
#Q7
```

name	year(place.order_da...	total_sales
Lenovo	2018	324291.59

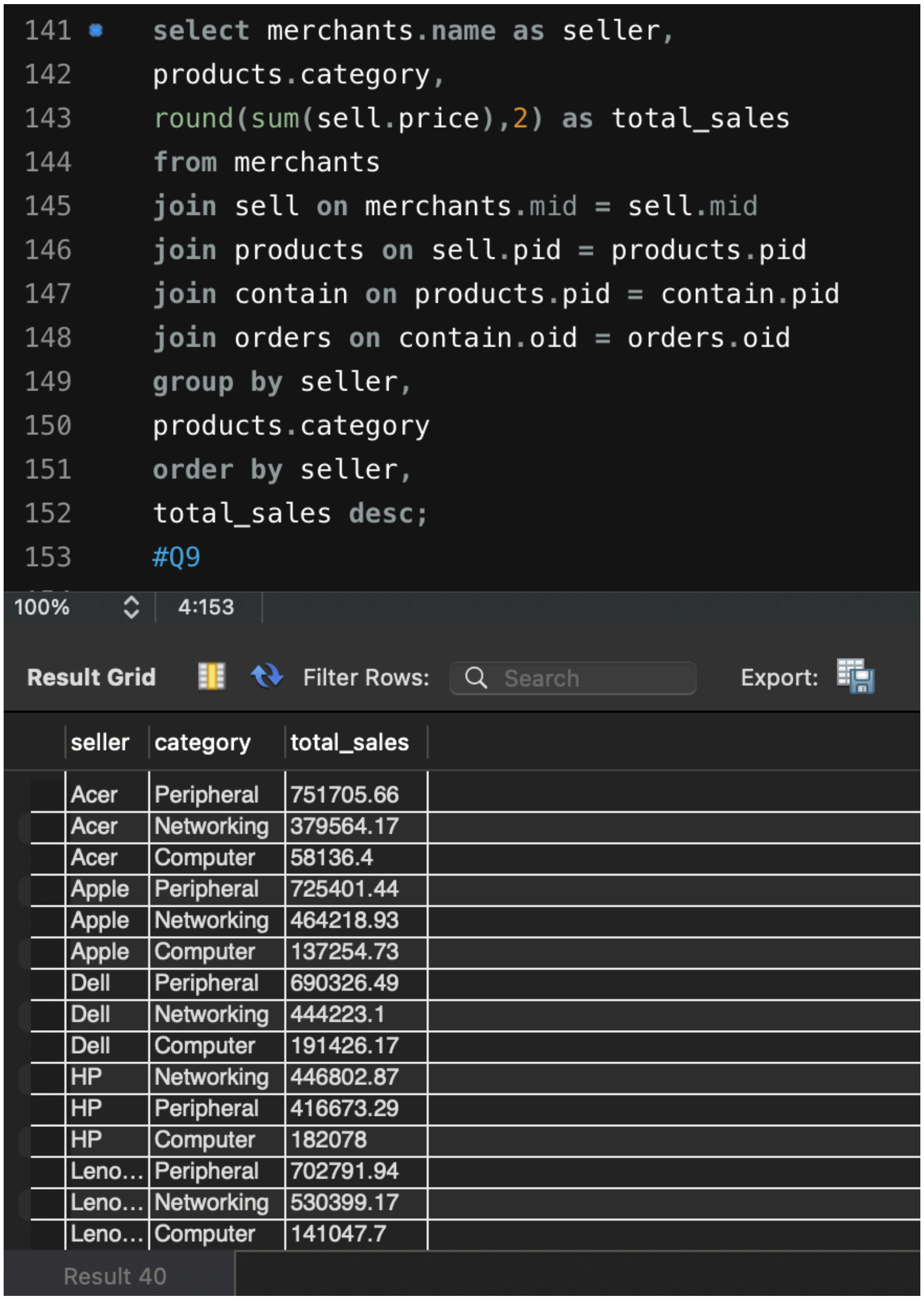
Q8: This query averages up all the shippings costs and differentiates from the three different companies. After I got the averages, I ordered by least to greatest and then limited by 1 to get the cheapest cost



```
select orders.shipping_method,
    round(avg(orders.shipping_cost),2) as avg_cost
    from orders
    group by shipping_method
    order by avg_cost asc
    limit 1;
#Q8
```

```
shipping_meth...  avg_cost
USPS              7.46
```

Q9: In this query I had to get the sum of the prices for all the orders and sort that into categories for each company. I pulled the company name, product category, and rounded sum of all the orders. I then joined all the necessary tables based on their PKs and FKs. I finally added the group by seller and ordered the row to show total_sales in descending order. The final result shows the sales for each category in each company.



```
select merchants.name as seller,
products.category,
round(sum(sell.price),2) as total_sales
from merchants
join sell on merchants.mid = sell.mid
join products on sell.pid = products.pid
join contain on products.pid = contain.pid
join orders on contain.oid = orders.oid
group by seller,
products.category
order by seller,
total_sales desc;
#Q9
```

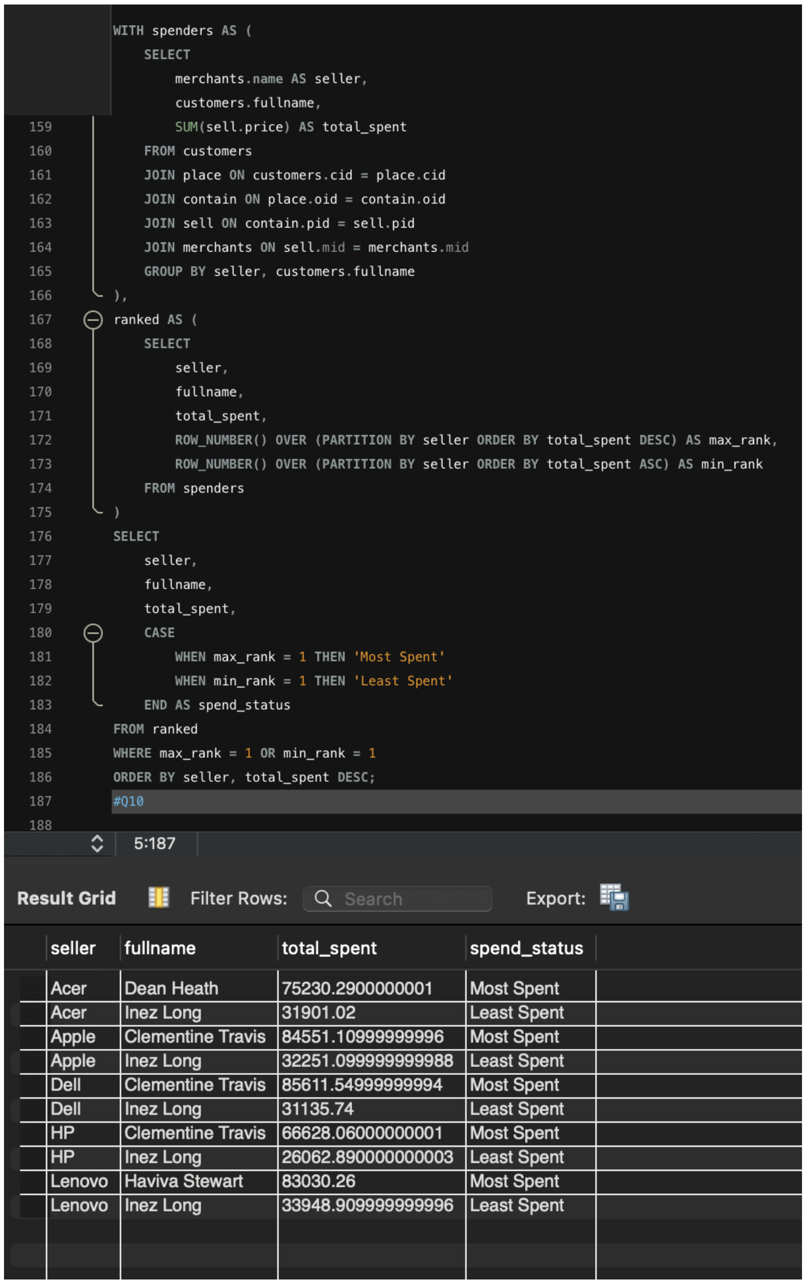
100% 4:153

Result Grid Filter Rows: Search Export:

seller	category	total_sales
Acer	Peripheral	751705.66
Acer	Networking	379564.17
Acer	Computer	58136.4
Apple	Peripheral	725401.44
Apple	Networking	464218.93
Apple	Computer	137254.73
Dell	Peripheral	690326.49
Dell	Networking	444223.1
Dell	Computer	191426.17
HP	Networking	446802.87
HP	Peripheral	416673.29
HP	Computer	182078
Leno...	Peripheral	702791.94
Leno...	Networking	530399.17
Leno...	Computer	141047.7

Result 40

Q10: In this query I started with a temp table (subquery) to get the total price spent by each customer. This subquery was named spenders. I then made another subquery to apply ranking to the spenders subquery. This row_number() function in conjunction with over partition allowed me to create variables that tags the highest or lowest spender. I then moved on to my main query where I pull from the ranked query that ranks/ tags the highest and lowest spenders.
I used case to create if clauses for a column spend_status. My conditions set would change the name value in the spend_status column depending on whether it was the highest or lowest.



```
WITH spenders AS (
    SELECT
        merchants.name AS seller,
        customers.fullname,
        SUM(sell.price) AS total_spent
    FROM customers
    JOIN place ON customers.cid = place.cid
    JOIN contain ON place.oid = contain.oid
    JOIN sell ON contain.pid = sell.pid
    JOIN merchants ON sell.mid = merchants.mid
    GROUP BY seller, customers.fullname
),
ranked AS (
    SELECT
        seller,
        fullname,
        total_spent,
        ROW_NUMBER() OVER (PARTITION BY seller ORDER BY total_spent DESC) AS max_rank,
        ROW_NUMBER() OVER (PARTITION BY seller ORDER BY total_spent ASC) AS min_rank
    FROM spenders
)
SELECT
    seller,
    fullname,
    total_spent,
    CASE
        WHEN max_rank = 1 THEN 'Most Spent'
        WHEN min_rank = 1 THEN 'Least Spent'
    END AS spend_status
FROM ranked
WHERE max_rank = 1 OR min_rank = 1
ORDER BY seller, total_spent DESC;
#Q10
```

5:187

Result Grid Filter Rows: Search Export:

seller	fullname	total_spent	spend_status
Acer	Dean Heath	75230.2900000001	Most Spent
Acer	Inez Long	31901.02	Least Spent
Apple	Clementine Travis	84551.10999999996	Most Spent
Apple	Inez Long	32251.099999999988	Least Spent
Dell	Clementine Travis	85611.54999999994	Most Spent
Dell	Inez Long	31135.74	Least Spent
HP	Clementine Travis	66628.06000000001	Most Spent
HP	Inez Long	26062.890000000003	Least Spent
Lenovo	Haviva Stewart	83030.26	Most Spent
Lenovo	Inez Long	33948.909999999996	Least Spent