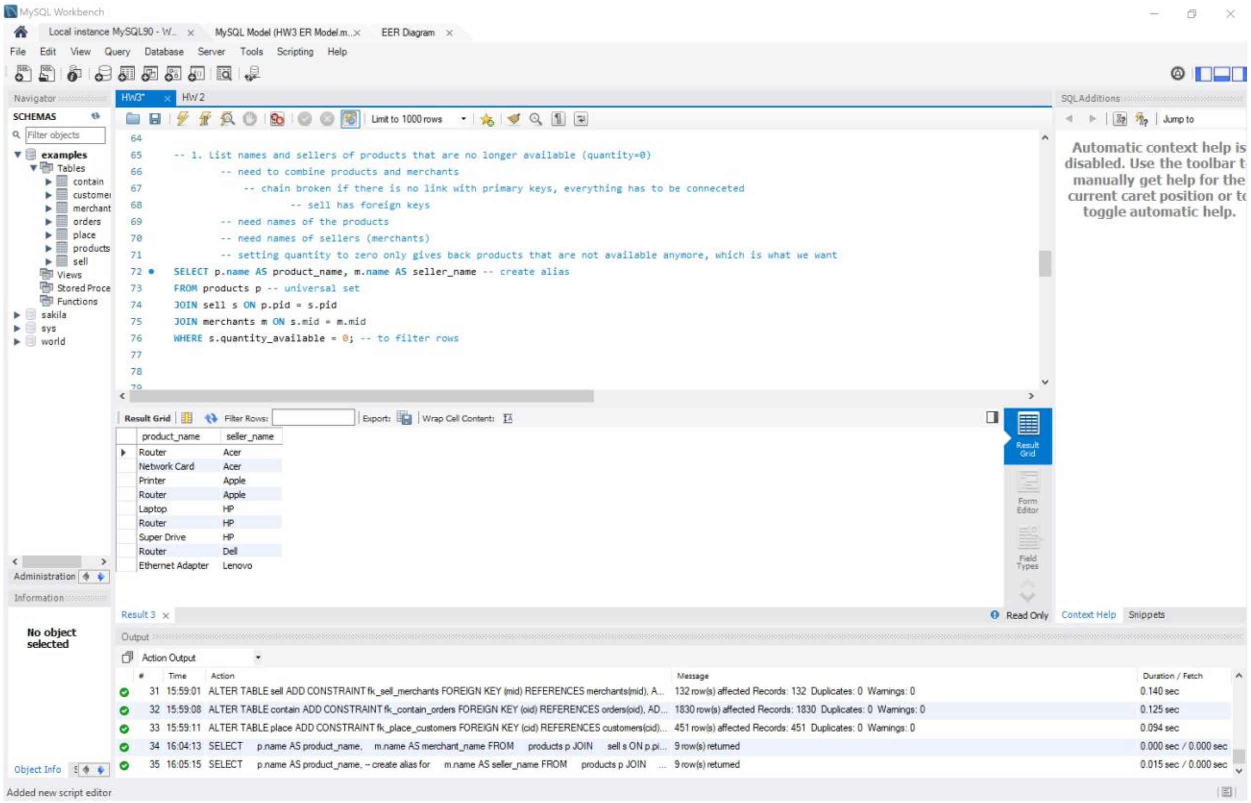
Christina DiMaggio

October 11, 2024

DB Assignment 3

1. List names and sellers of products that are no longer available (quantity=0)



```
-- 1. List names and sellers of products that are no longer available (quantity=0)
        -- need to combine products and merchants
            -- chain broken if there is no link with primary keys, everything has to be conneceted
                    -- sell has foreign keys
        -- need names of the products
        -- need names of sellers (merchants)
        -- setting quantity to zero only gives back products that are not available anymore, which is what we want
SELECT p.name AS product_name, m.name AS seller_name -- create alias
FROM products p -- universal set
JOIN sell s ON p.pid = s.pid
JOIN merchants m ON s.mid = m.mid
WHERE s.quantity_available = 0; -- to filter rows
```

product_name	seller_name
Router	Acer
Network Card	Acer
Printer	Apple
Router	Apple
Laptop	HP
Router	HP
Super Drive	HP
Router	Dell
Ethernet Adapter	Lenovo

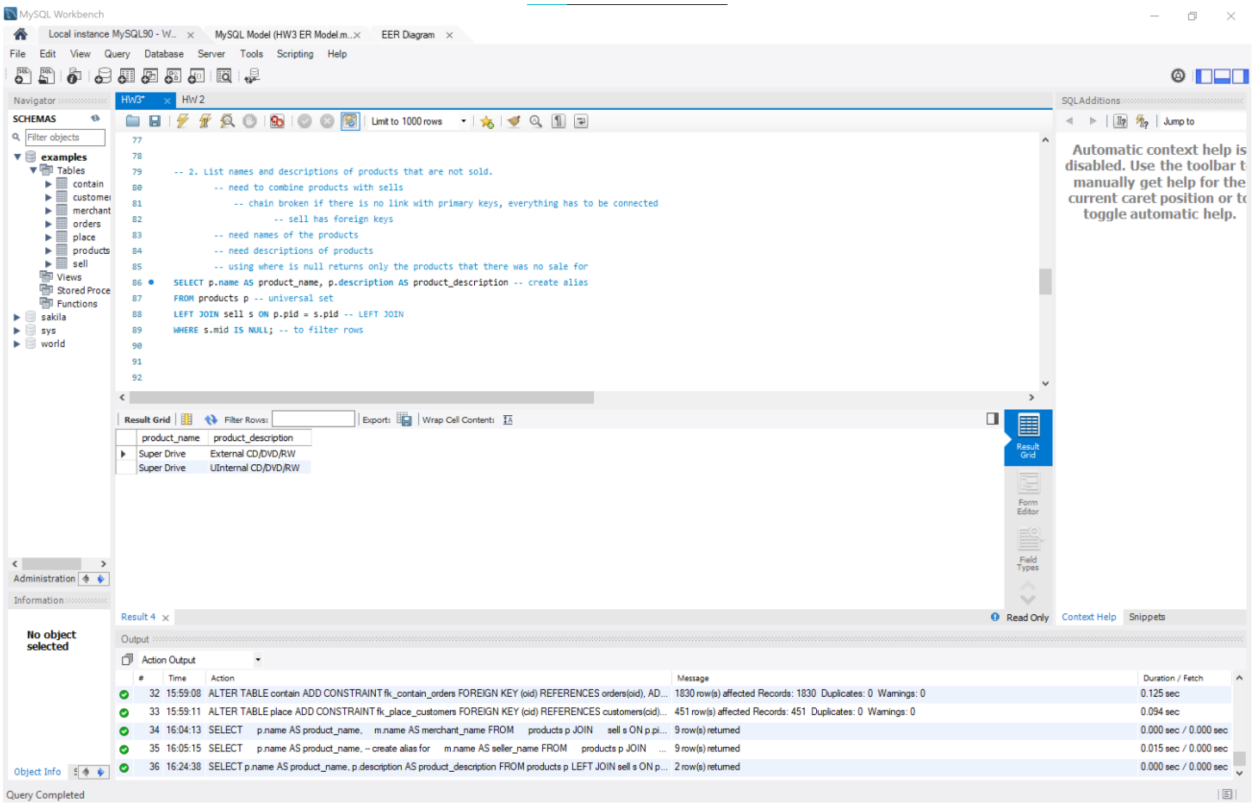
```
#   Time      Action                                                                                   Message                                                       Duration / Fetch
31  15:59:01  ALTER TABLE sell ADD CONSTRAINT fk_sell_merchants FOREIGN KEY (mid) REFERENCES merchants(mid), A...   132 row(s) affected Records: 132  Duplicates: 0  Warnings: 0    0.140 sec
32  15:59:08  ALTER TABLE contain ADD CONSTRAINT fk_contain_orders FOREIGN KEY (oid) REFERENCES orders(oid), AD...  1830 row(s) affected Records: 1830  Duplicates: 0  Warnings: 0  0.125 sec
33  15:59:11  ALTER TABLE place ADD CONSTRAINT fk_place_customers FOREIGN KEY (cid) REFERENCES customers(cid)...    451 row(s) affected Records: 451  Duplicates: 0  Warnings: 0    0.094 sec
34  16:04:13  SELECT    p.name AS product_name,    m.name AS merchant_name FROM    products p JOIN    sell s ON p.pi...   9 row(s) returned    0.000 sec / 0.000 sec
35  16:05:15  SELECT    p.name AS product_name, -- create alias for    m.name AS seller_name FROM    products p JOIN    ...   9 row(s) returned    0.015 sec / 0.000 sec
```

What does query do? How does it solve the problem?

This query can INNER join products with sells and merchants by utilizing their primary keys and or foreign keys. Inner joins are for exactly which columns we need and are using. Comments on SQL queries also better define what codes represent.

The query nicely lays out the name of the product on the left and its respective merchant (seller) name on the right of the table where there is no more of the product available.

2. List names and descriptions of products that are not sold.



```sql
-- 2. List names and descriptions of products that are not sold.
        -- need to combine products with sells
            -- chain broken if there is no link with primary keys, everything has to be connected
                    -- sell has foreign keys
        -- need names of the products
        -- need descriptions of products
        -- using where is null returns only the products that there was no sale for
SELECT p.name AS product_name, p.description AS product_description -- create alias
FROM products p -- universal set
LEFT JOIN sell s ON p.pid = s.pid -- LEFT JOIN
WHERE s.mid IS NULL; -- to filter rows
```

product_name	product_description
Super Drive	External CD/DVD/RW
Super Drive	UInternal CD/DVD/RW

Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or tc toggle automatic help.

No object selected

```
#   Time      Action                                                                                            Message                                                          Duration / Fetch
32  15:59:08  ALTER TABLE contain ADD CONSTRAINT fk_contain_orders FOREIGN KEY (oid) REFERENCES orders(oid), AD...  1830 row(s) affected Records: 1830  Duplicates: 0  Warnings: 0  0.125 sec
33  15:59:11  ALTER TABLE place ADD CONSTRAINT fk_place_customers FOREIGN KEY (cid) REFERENCES customers(cid)...     451 row(s) affected Records: 451  Duplicates: 0  Warnings: 0    0.094 sec
34  16:04:13  SELECT    p.name AS product_name,    m.name AS merchant_name FROM    products p JOIN    sell s ON p.pi...  9 row(s) returned                                     0.000 sec / 0.000 sec
35  16:05:15  SELECT    p.name AS product_name, -- create alias for    m.name AS seller_name FROM    products p JOIN    ...  9 row(s) returned                                 0.015 sec / 0.000 sec
36  16:24:38  SELECT p.name AS product_name, p.description AS product_description FROM products p LEFT JOIN sell s ON p...  2 row(s) returned                             0.000 sec / 0.000 sec
```

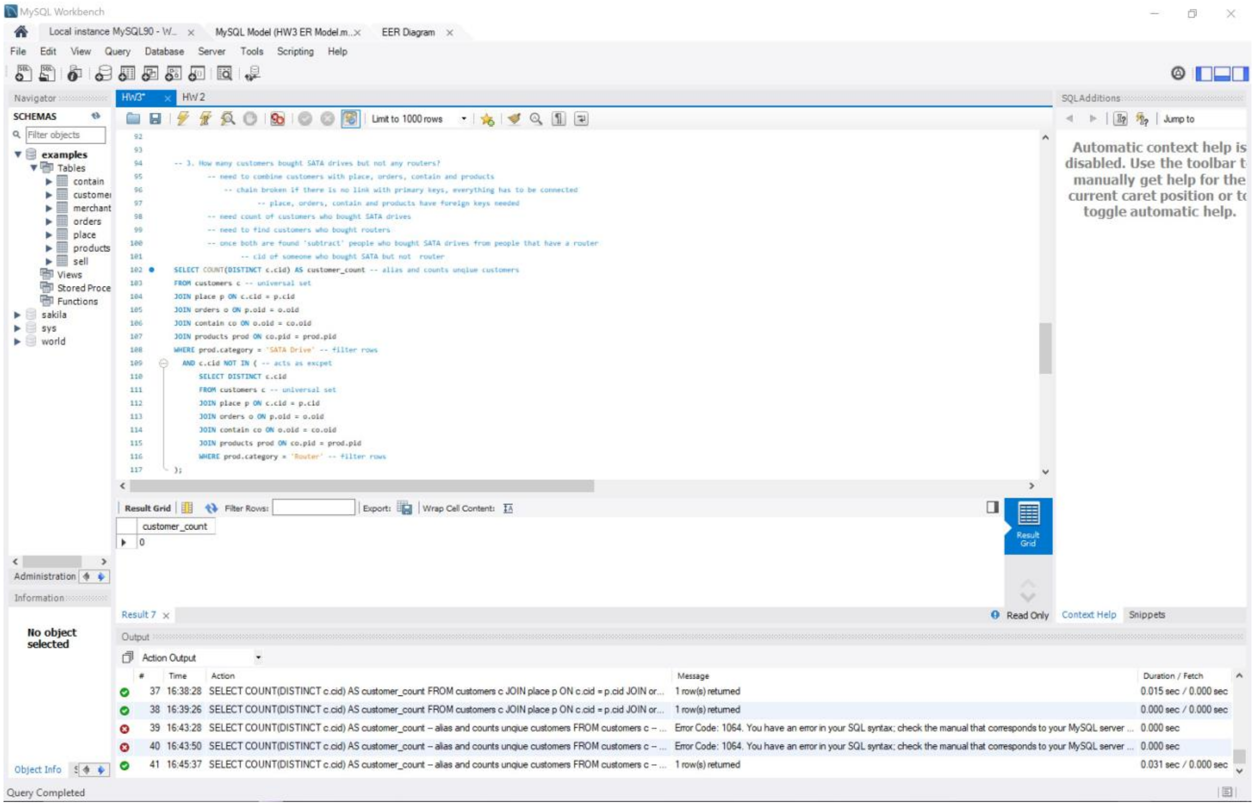
Query Completed

What does query do? How does it solve the problem?

This query can LEFT join products with sells by utilizing their primary keys and or foreign keys. Left joins will return all records from the left table, and the matched records from the right table. It will report all products whether they sold or not. Comments on SQL queries also better define what codes represent.

The query nicely lays out the name of the product on the left and its respective product description on the right of the table where there were no sales made for the product.

3. **How many customers bought SATA drives but not any routers?**



```
-- 3. How many customers bought SATA drives but not any routers?
        -- need to combine customers with place, orders, contain and products
            -- chain broken if there is no link with primary keys, everything has to be connected
                    -- place, orders, contain and products have foreign keys needed
        -- need count of customers who bought SATA drives
        -- need to find customers who bought routers
        -- once both are found 'subtract' people who bought SATA drives from people that have a router
                -- cid of someone who bought SATA but not  router
SELECT COUNT(DISTINCT c.cid) AS customer_count -- alias and counts unqiue customers
FROM customers c -- universal set
JOIN place p ON c.cid = p.cid
JOIN orders o ON p.oid = o.oid
JOIN contain co ON o.oid = co.oid
JOIN products prod ON co.pid = prod.pid
WHERE prod.category = 'SATA Drive' -- filter rows
  AND c.cid NOT IN ( -- acts as excpet
      SELECT DISTINCT c.cid
      FROM customers c -- universal set
      JOIN place p ON c.cid = p.cid
      JOIN orders o ON p.oid = o.oid
      JOIN contain co ON o.oid = co.oid
      JOIN products prod ON co.pid = prod.pid
      WHERE prod.category = 'Router' -- filter rows
);
```

customer_count
0

Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or tc toggle automatic help.

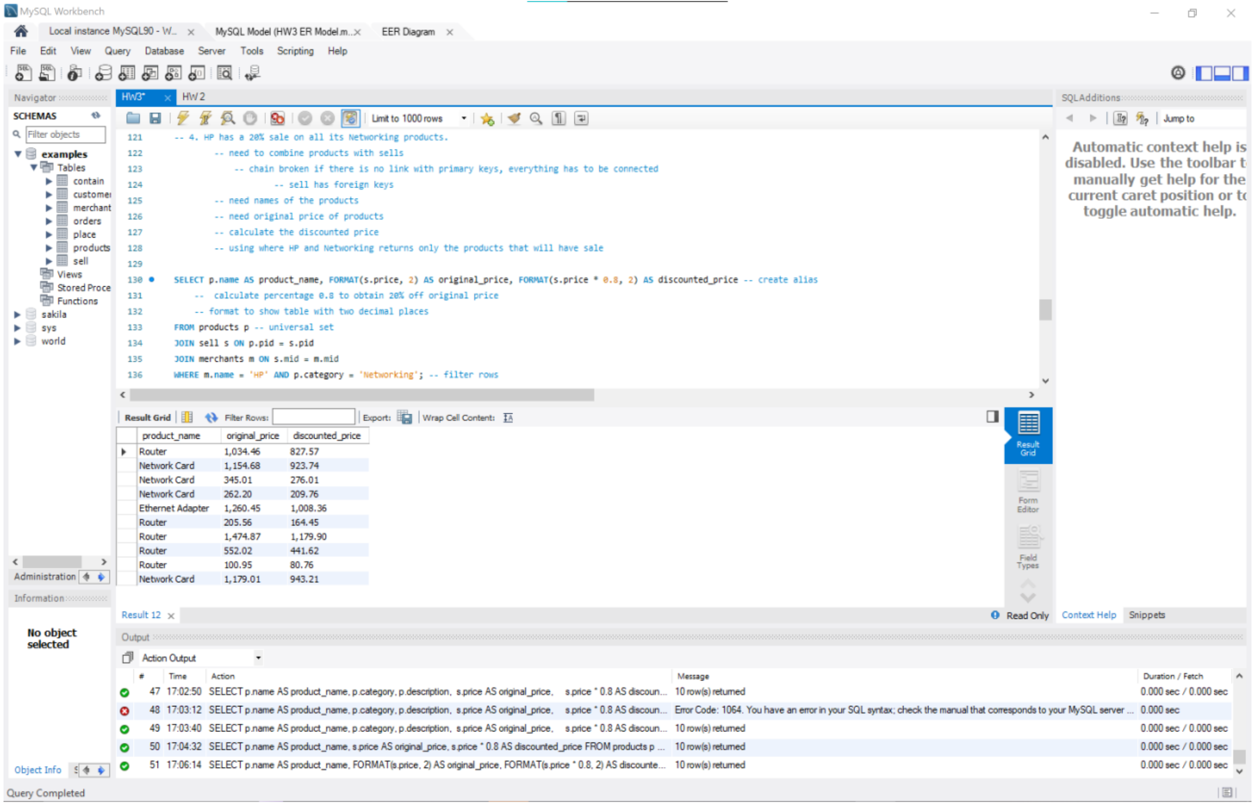
```
#   Time      Action                                                                                      Message                                                                           Duration / Fetch
37  16:38:28  SELECT COUNT(DISTINCT c.cid) AS customer_count FROM customers c JOIN place p ON c.cid = p.cid JOIN or...   1 row(s) returned                                                                 0.015 sec / 0.000 sec
38  16:39:26  SELECT COUNT(DISTINCT c.cid) AS customer_count FROM customers c JOIN place p ON c.cid = p.cid JOIN or...   1 row(s) returned                                                                 0.000 sec / 0.000 sec
39  16:43:28  SELECT COUNT(DISTINCT c.cid) AS customer_count -- alias and counts unqiue customers FROM customers c -- ...   Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server ...   0.000 sec
40  16:43:50  SELECT COUNT(DISTINCT c.cid) AS customer_count -- alias and counts unqiue customers FROM customers c -- ...   Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server ...   0.000 sec
41  16:45:37  SELECT COUNT(DISTINCT c.cid) AS customer_count -- alias and counts unqiue customers FROM customers c -- ...   1 row(s) returned                                                                 0.031 sec / 0.000 sec
```

What does query do? How does it solve the problem?

This query can join customers with place, orders, contains, and products by utilizing their primary keys and or foreign keys. INNER join is completed twice, once with where clause SATA Drive, the other with where clause Router. Command 'AND c.cid NOT IN' takes the cid of someone who bought a SATA Drive and compares it to the table of people who bought Routers. It will report customers who did not buy Router Comments on SQL queries also better define what codes represent.

The query nicely lays out the customer count of someone who bought a SATA Drive but not a Router.

4. **HP has a 20% sale on all its Networking products.**



```sql
-- 4. HP has a 20% sale on all its Networking products.
        -- need to combine products with sells
            -- chain broken if there is no link with primary keys, everything has to be connected
                    -- sell has foreign keys
        -- need names of the products
        -- need original price of products
        -- calculate the discounted price
        -- using where HP and Networking returns only the products that will have sale

SELECT p.name AS product_name, FORMAT(s.price, 2) AS original_price, FORMAT(s.price * 0.8, 2) AS discounted_price -- create alias
    --  calculate percentage 0.8 to obtain 20% off original price
    -- format to show table with two decimal places
FROM products p -- universal set
JOIN sell s ON p.pid = s.pid
JOIN merchants m ON s.mid = m.mid
WHERE m.name = 'HP' AND p.category = 'Networking'; -- filter rows
```

product_name	original_price	discounted_price
Router	1,034.46	827.57
Network Card	1,154.68	923.74
Network Card	345.01	276.01
Network Card	262.20	209.76
Ethernet Adapter	1,260.45	1,008.36
Router	205.56	164.45
Router	1,474.87	1,179.90
Router	552.02	441.62
Router	100.95	80.76
Network Card	1,179.01	943.21

```
#   Time      Action                                                                                                Message                                                                                          Duration / Fetch
47  17:02:50  SELECT p.name AS product_name, p.category, p.description,  s.price AS original_price,   s.price * 0.8 AS discoun...  10 row(s) returned                                                                            0.000 sec / 0.000 sec
48  17:03:12  SELECT p.name AS product_name, p.category, p.description,  s.price AS original_price,   s.price * 0.8 AS discoun...  Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server ...  0.000 sec
49  17:03:40  SELECT p.name AS product_name, p.category, p.description,  s.price AS original_price,   s.price * 0.8 AS discoun...  10 row(s) returned                                                                            0.000 sec / 0.000 sec
50  17:04:32  SELECT p.name AS product_name, s.price AS original_price, s.price * 0.8 AS discounted_price FROM products p ...  10 row(s) returned                                                                   0.000 sec / 0.000 sec
51  17:06:14  SELECT p.name AS product_name, FORMAT(s.price, 2) AS original_price, FORMAT(s.price * 0.8, 2) AS discounte...  10 row(s) returned                                                                 0.000 sec / 0.000 sec
```

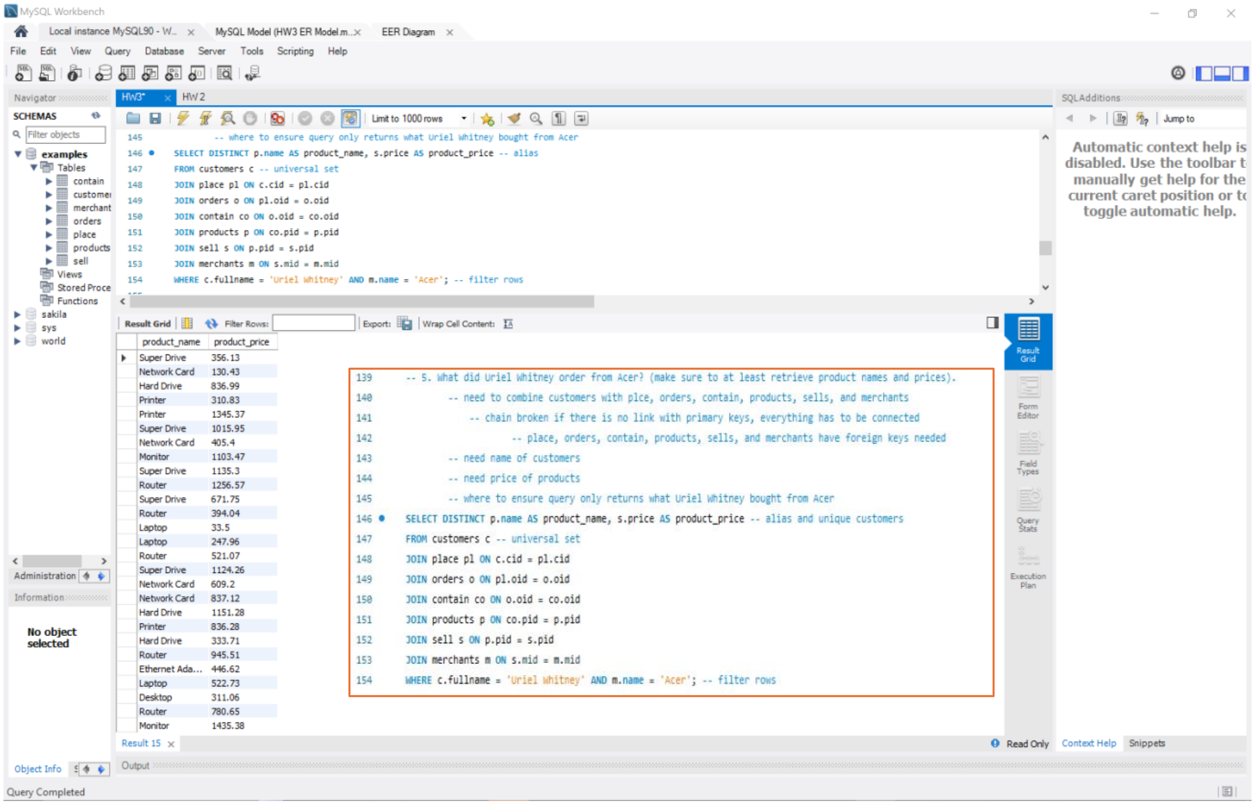
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or tc toggle automatic help.

What does query do? How does it solve the problem?

This query can INNER join products with sells by utilizing their primary keys and or foreign keys. Inner joins are for exactly which columns we need and are using. Comments on SQL queries also better define what codes represent.

The query nicely lays out the name of the HP Networking product on the far left, and its respective original price then discounted price of 20% off.

5. **What did Uriel Whitney order from Acer? (make sure to at least retrieve product names and prices).**



```
        -- where to ensure query only returns what Uriel Whitney bought from Acer
SELECT DISTINCT p.name AS product_name, s.price AS product_price -- alias
FROM customers c -- universal set
JOIN place pl ON c.cid = pl.cid
JOIN orders o ON pl.oid = o.oid
JOIN contain co ON o.oid = co.oid
JOIN products p ON co.pid = p.pid
JOIN sell s ON p.pid = s.pid
JOIN merchants m ON s.mid = m.mid
WHERE c.fullname = 'Uriel Whitney' AND m.name = 'Acer'; -- filter rows
```

product_name	product_price
Super Drive	356.13
Network Card	130.43
Hard Drive	836.99
Printer	310.83
Printer	1345.37
Super Drive	1015.95
Network Card	405.4
Monitor	1103.47
Super Drive	1135.3
Router	1256.57
Super Drive	671.75
Router	394.04
Laptop	33.5
Laptop	247.96
Router	521.07
Super Drive	1124.26
Network Card	609.2
Network Card	837.12
Hard Drive	1151.28
Printer	836.28
Hard Drive	333.71
Router	945.51
Ethernet Ada...	446.62
Laptop	522.73
Desktop	311.06
Router	780.65
Monitor	1435.38

```
-- 5. What did Uriel Whitney order from Acer? (make sure to at least retrieve product names and prices).
        -- need to combine customers with plce, orders, contain, products, sells, and merchants
            -- chain broken if there is no link with primary keys, everything has to be connected
                    -- place, orders, contain, products, sells, and merchants have foreign keys needed
        -- need name of customers
        -- need price of products
        -- where to ensure query only returns what Uriel Whitney bought from Acer
SELECT DISTINCT p.name AS product_name, s.price AS product_price -- alias and unique customers
FROM customers c -- universal set
JOIN place pl ON c.cid = pl.cid
JOIN orders o ON pl.oid = o.oid
JOIN contain co ON o.oid = co.oid
JOIN products p ON co.pid = p.pid
JOIN sell s ON p.pid = s.pid
JOIN merchants m ON s.mid = m.mid
WHERE c.fullname = 'Uriel Whitney' AND m.name = 'Acer'; -- filter rows
```

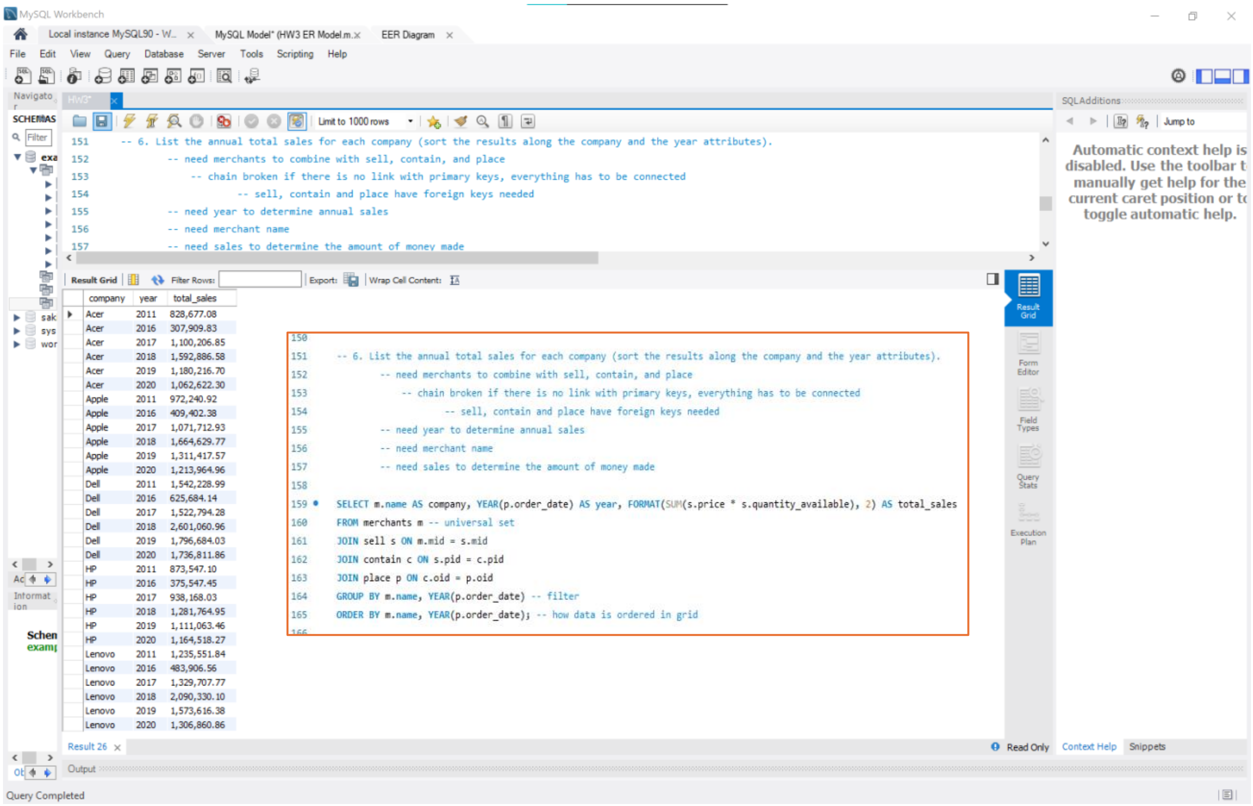
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or tc toggle automatic help.

What does query do? How does it solve the problem?

This query can join customers with place, orders, contains, products, sell, and merchants by utilizing their primary keys and or foreign keys. Inner joins are for exactly which columns we need and are using. Comments on SQL queries also better define what codes represent.

The query nicely lays out the name of the product on the left and its respective price on the right of the table that Uriel Whitney bought from Acer.

6. **List the annual total sales for each company (sort the results along the company and the year attributes).**


MySQL Workbench
Local instance MySQL90 - W...
MySQL Model* (HW3 ER Model.m...
EER Diagram
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
HW3*
Limit to 1000 rows
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or to toggle automatic help.
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Result Grid
Form Editor
Field Types
Query Stats
Execution Plan
Result 26
Read Only
Context Help
Snippets
Output
Query Completed

```sql
-- 6. List the annual total sales for each company (sort the results along the company and the year attributes).
        -- need merchants to combine with sell, contain, and place
            -- chain broken if there is no link with primary keys, everything has to be connected
                    -- sell, contain and place have foreign keys needed
        -- need year to determine annual sales
        -- need merchant name
        -- need sales to determine the amount of money made
```

```sql

-- 6. List the annual total sales for each company (sort the results along the company and the year attributes).
        -- need merchants to combine with sell, contain, and place
            -- chain broken if there is no link with primary keys, everything has to be connected
                    -- sell, contain and place have foreign keys needed
        -- need year to determine annual sales
        -- need merchant name
        -- need sales to determine the amount of money made

SELECT m.name AS company, YEAR(p.order_date) AS year, FORMAT(SUM(s.price * s.quantity_available), 2) AS total_sales
FROM merchants m -- universal set
JOIN sell s ON m.mid = s.mid
JOIN contain c ON s.pid = c.pid
JOIN place p ON c.oid = p.oid
GROUP BY m.name, YEAR(p.order_date) -- filter
ORDER BY m.name, YEAR(p.order_date); -- how data is ordered in grid
```

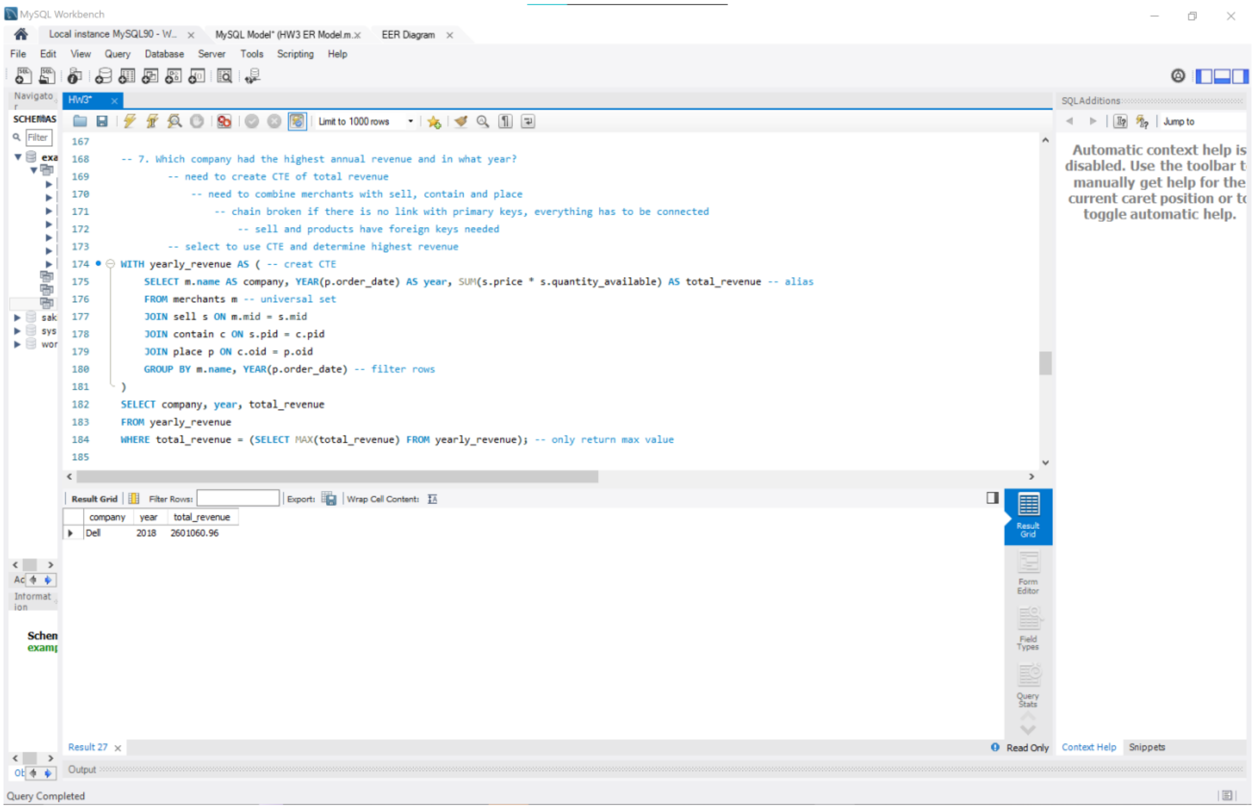
company	year	total_sales
Acer	2011	828,677.08
Acer	2016	307,909.83
Acer	2017	1,100,206.85
Acer	2018	1,592,886.58
Acer	2019	1,180,216.70
Acer	2020	1,062,622.30
Apple	2011	972,240.92
Apple	2016	409,402.38
Apple	2017	1,071,712.93
Apple	2018	1,664,629.77
Apple	2019	1,311,417.57
Apple	2020	1,213,964.96
Dell	2011	1,542,228.99
Dell	2016	625,684.14
Dell	2017	1,522,794.28
Dell	2018	2,601,060.96
Dell	2019	1,796,684.03
Dell	2020	1,736,811.86
HP	2011	873,547.10
HP	2016	375,547.45
HP	2017	938,168.03
HP	2018	1,281,764.95
HP	2019	1,111,063.46
HP	2020	1,164,518.27
Lenovo	2011	1,235,551.84
Lenovo	2016	483,906.56
Lenovo	2017	1,329,707.77
Lenovo	2018	2,090,330.10
Lenovo	2019	1,573,616.38
Lenovo	2020	1,306,860.86

What does query do? How does it solve the problem?

This query can join merchants with sell, contain, and place by utilizing their primary keys and or foreign keys. Inner joins are for exactly which columns we need and are using. Comments on SQL queries also better define what codes represent.

The query nicely lays out the name of the company by groups with their total sales amount for different years.

7. Which company had the highest annual revenue and in what year?



Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```sql
-- 7. Which company had the highest annual revenue and in what year?
        -- need to create CTE of total revenue
            -- need to combine merchants with sell, contain and place
                -- chain broken if there is no link with primary keys, everything has to be connected
                    -- sell and products have foreign keys needed
        -- select to use CTE and determine highest revenue
WITH yearly_revenue AS ( -- creat CTE
    SELECT m.name AS company, YEAR(p.order_date) AS year, SUM(s.price * s.quantity_available) AS total_revenue -- alias
    FROM merchants m -- universal set
    JOIN sell s ON m.mid = s.mid
    JOIN contain c ON s.pid = c.pid
    JOIN place p ON c.oid = p.oid
    GROUP BY m.name, YEAR(p.order_date) -- filter rows
)
SELECT company, year, total_revenue
FROM yearly_revenue
WHERE total_revenue = (SELECT MAX(total_revenue) FROM yearly_revenue); -- only return max value
```

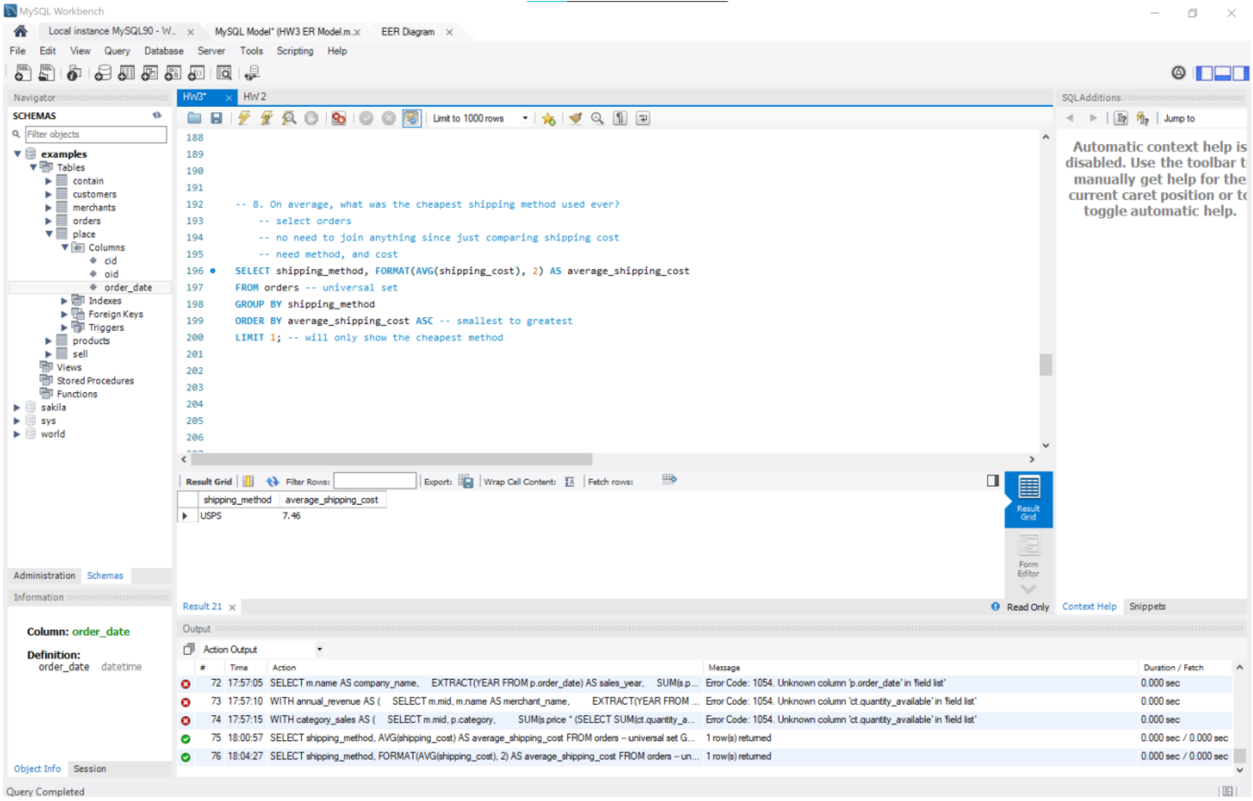
company	year	total_revenue
Dell	2018	2601060.96

What does query do? How does it solve the problem?

This query creates a CTEs to be able to calculate the yearly revenue from each company by joining merchants with sell, contain, and place. Select is important for determining which company, year, and total revenue has the highest max value. Comments on SQL queries also better define what codes represent.

The query nicely lays out the company, year, and total revenue of the top company.

8. On average, what was the cheapest shipping method used ever?



```sql
-- 8. On average, what was the cheapest shipping method used ever?
    -- select orders
    -- no need to join anything since just comparing shipping cost
    -- need method, and cost
SELECT shipping_method, FORMAT(AVG(shipping_cost), 2) AS average_shipping_cost
FROM orders -- universal set
GROUP BY shipping_method
ORDER BY average_shipping_cost ASC -- smallest to greatest
LIMIT 1; -- will only show the cheapest method
```

shipping_method	average_shipping_cost
USPS	7.46

```
#   Time      Action                                                                                  Message                                                          Duration / Fetch
72  17:57:05  SELECT m.name AS company_name,    EXTRACT(YEAR FROM p.order_date) AS sales_year,    SUM(s.p...   Error Code: 1054. Unknown column 'p.order_date' in 'field list'          0.000 sec
73  17:57:10  WITH annual_revenue AS (    SELECT m.mid, m.name AS merchant_name,       EXTRACT(YEAR FROM ...   Error Code: 1054. Unknown column 'ct.quantity_available' in 'field list'  0.000 sec
74  17:57:15  WITH category_sales AS (    SELECT m.mid, p.category,       SUM(s.price * (SELECT SUM(ct.quantity_a...   Error Code: 1054. Unknown column 'ct.quantity_available' in 'field list'  0.000 sec
75  18:00:57  SELECT shipping_method, AVG(shipping_cost) AS average_shipping_cost FROM orders -- universal set G...   1 row(s) returned   0.000 sec / 0.000 sec
76  18:04:27  SELECT shipping_method, FORMAT(AVG(shipping_cost), 2) AS average_shipping_cost FROM orders -- un...   1 row(s) returned   0.000 sec / 0.000 sec
```

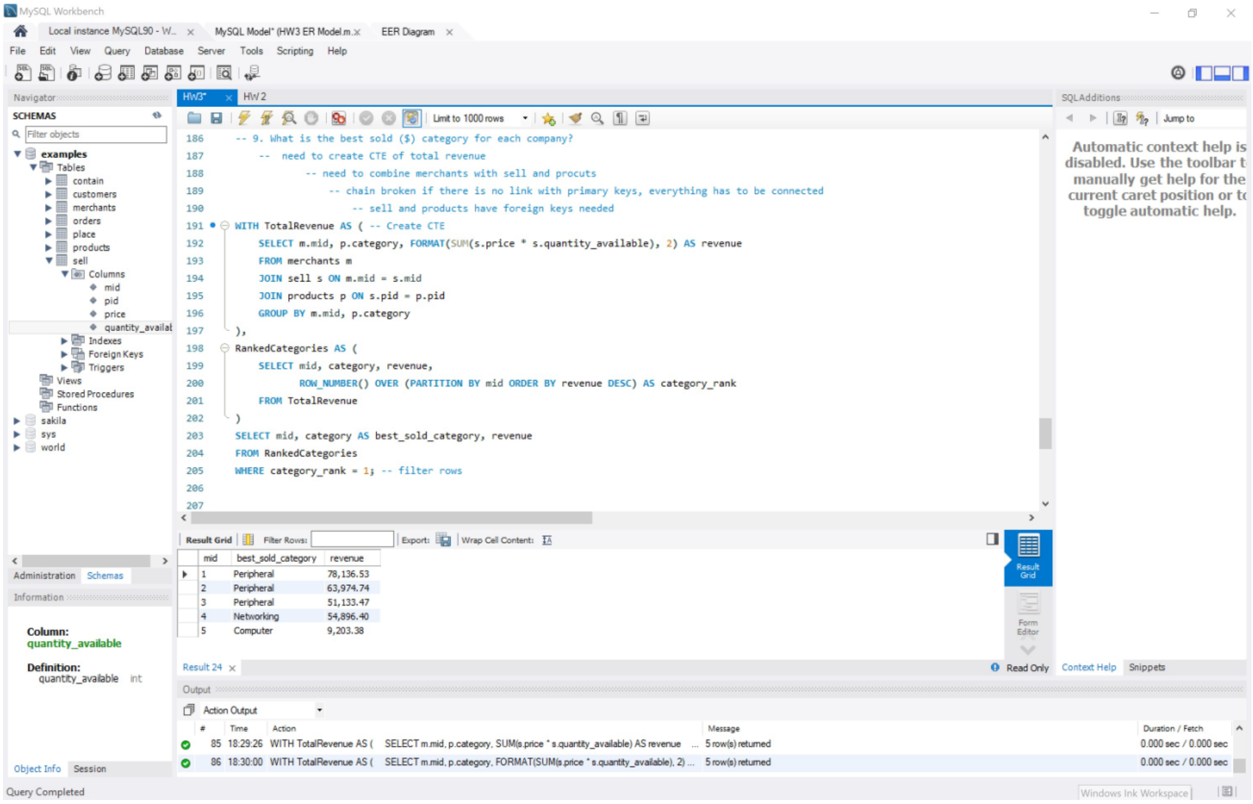
Query Completed

What does query do? How does it solve the problem?

This query is a simple query that utilizes just orders to determine the cheapest shipping method. Comments on SQL queries also better define what codes represent.

The query nicely lays out the cheapest shipping method on the left and its respective average shipping cost on the right.

9. What is the best sold ($) category for each company?



```sql
-- 9. What is the best sold ($) category for each company?
    --  need to create CTE of total revenue
            -- need to combine merchants with sell and procuts
                -- chain broken if there is no link with primary keys, everything has to be connected
                    -- sell and products have foreign keys needed
WITH TotalRevenue AS ( -- Create CTE
    SELECT m.mid, p.category, FORMAT(SUM(s.price * s.quantity_available), 2) AS revenue
    FROM merchants m
    JOIN sell s ON m.mid = s.mid
    JOIN products p ON s.pid = p.pid
    GROUP BY m.mid, p.category
),
RankedCategories AS (
    SELECT mid, category, revenue,
           ROW_NUMBER() OVER (PARTITION BY mid ORDER BY revenue DESC) AS category_rank
    FROM TotalRevenue
)
SELECT mid, category AS best_sold_category, revenue
FROM RankedCategories
WHERE category_rank = 1; -- filter rows
```

mid	best_sold_category	revenue
1	Peripheral	78,136.53
2	Peripheral	63,974.74
3	Peripheral	51,133.47
4	Networking	54,896.40
5	Computer	9,203.38

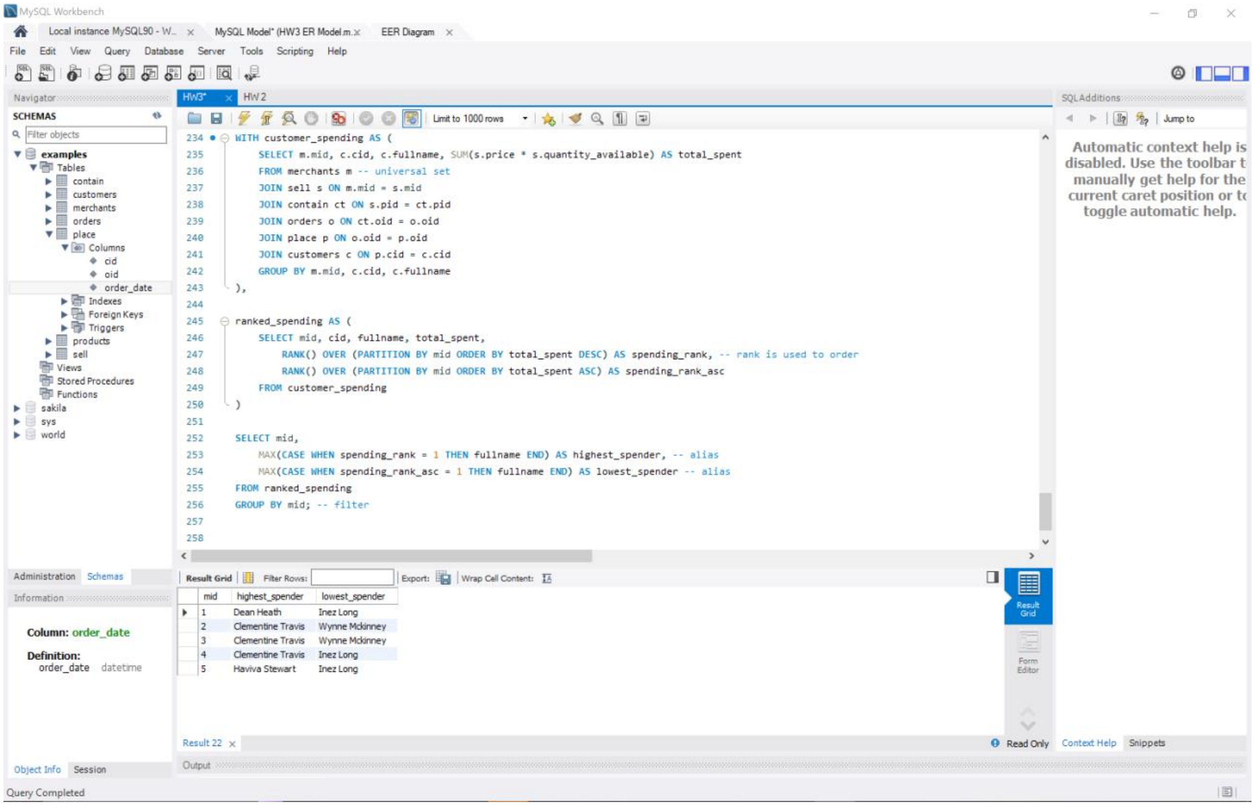
```
#   Time      Action                                                                                         Message            Duration / Fetch
85  18:29:26  WITH TotalRevenue AS (   SELECT m.mid, p.category, SUM(s.price * s.quantity_available) AS revenue   ...  5 row(s) returned  0.000 sec / 0.000 sec
86  18:30:00  WITH TotalRevenue AS (   SELECT m.mid, p.category, FORMAT(SUM(s.price * s.quantity_available), 2) ...  5 row(s) returned  0.000 sec / 0.000 sec
```

What does query do? How does it solve the problem?

This query creates two CTEs one able to calculate the total revenue for all the categories the company sells and two to rank categories within the companies based on their revenue. Select is important for determining the best-sold category for each company based on its rank. Mid can relate to a specific merchant name. Comments on SQL queries also better define what codes represent.

The query nicely lays out the merchant id and their respective best-sold category and the revenue it made.

10. For each company find out which customers have spent the most and the least amounts.



```sql
WITH customer_spending AS (
    SELECT m.mid, c.cid, c.fullname, SUM(s.price * s.quantity_available) AS total_spent
    FROM merchants m -- universal set
    JOIN sell s ON m.mid = s.mid
    JOIN contain ct ON s.pid = ct.pid
    JOIN orders o ON ct.oid = o.oid
    JOIN place p ON o.oid = p.oid
    JOIN customers c ON p.cid = c.cid
    GROUP BY m.mid, c.cid, c.fullname
),

ranked_spending AS (
    SELECT mid, cid, fullname, total_spent,
        RANK() OVER (PARTITION BY mid ORDER BY total_spent DESC) AS spending_rank, -- rank is used to order
        RANK() OVER (PARTITION BY mid ORDER BY total_spent ASC) AS spending_rank_asc
    FROM customer_spending
)

SELECT mid,
    MAX(CASE WHEN spending_rank = 1 THEN fullname END) AS highest_spender, -- alias
    MAX(CASE WHEN spending_rank_asc = 1 THEN fullname END) AS lowest_spender -- alias
FROM ranked_spending
GROUP BY mid; -- filter
```

mid	highest_spender	lowest_spender
1	Dean Heath	Inez Long
2	Clementine Travis	Wynne Mckinney
3	Clementine Travis	Wynne Mckinney
4	Clementine Travis	Inez Long
5	Haviva Stewart	Inez Long

Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or tc toggle automatic help.

What does query do? How does it solve the problem?

This query creates two CTEs one able to calculate the total amount spent by the customer from each merchant and two to rank customers based on spending in ascending and descending order. Select is important for determining which customer is the highest and lowest spender for each merchant. Mid can relate to a specific merchant name. Comments on SQL queries also better define what codes represent.

The query nicely lays out the merchant id and their respective highest and lowest spender.