SQL Reinforcement Project

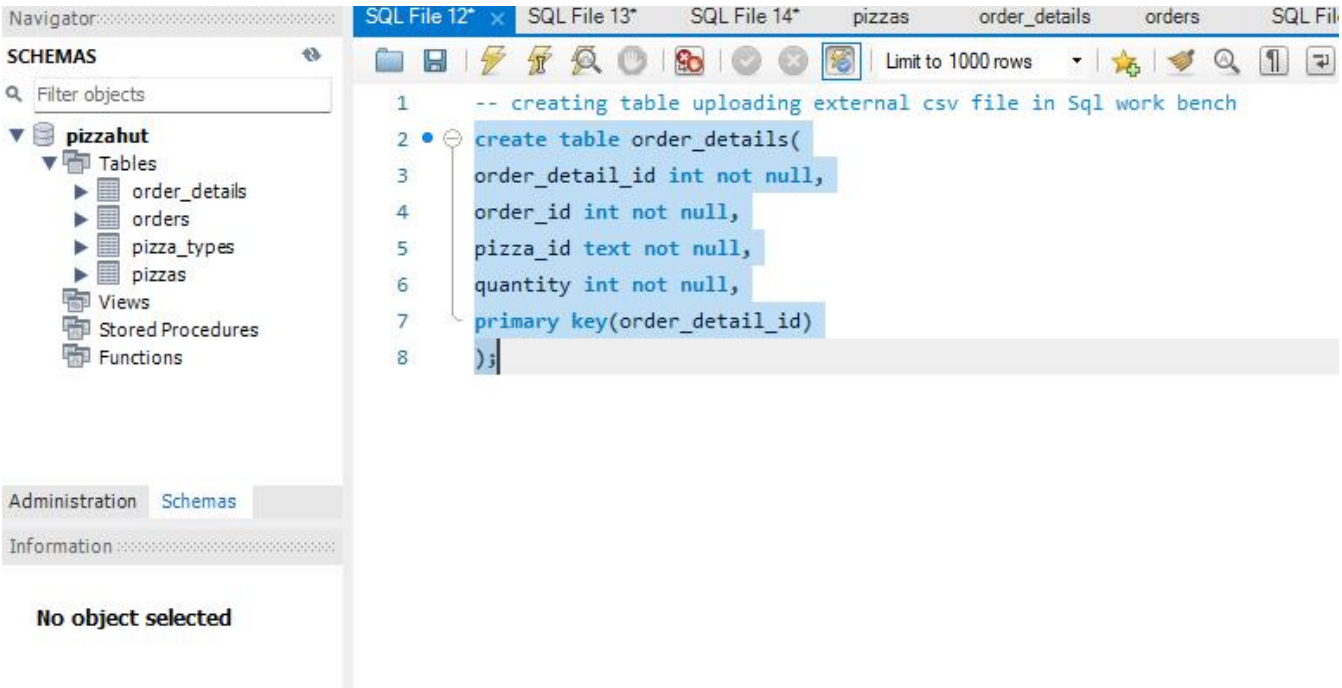
Question 1:

creating table uploading external csv file in Sql work bench

Query:

```
create table order_details(
order_detail_id int not null,
order_id int not null,
pizza_id text not null,
quantity int not null,
primary key(order_detail_id)
);
```

Output:



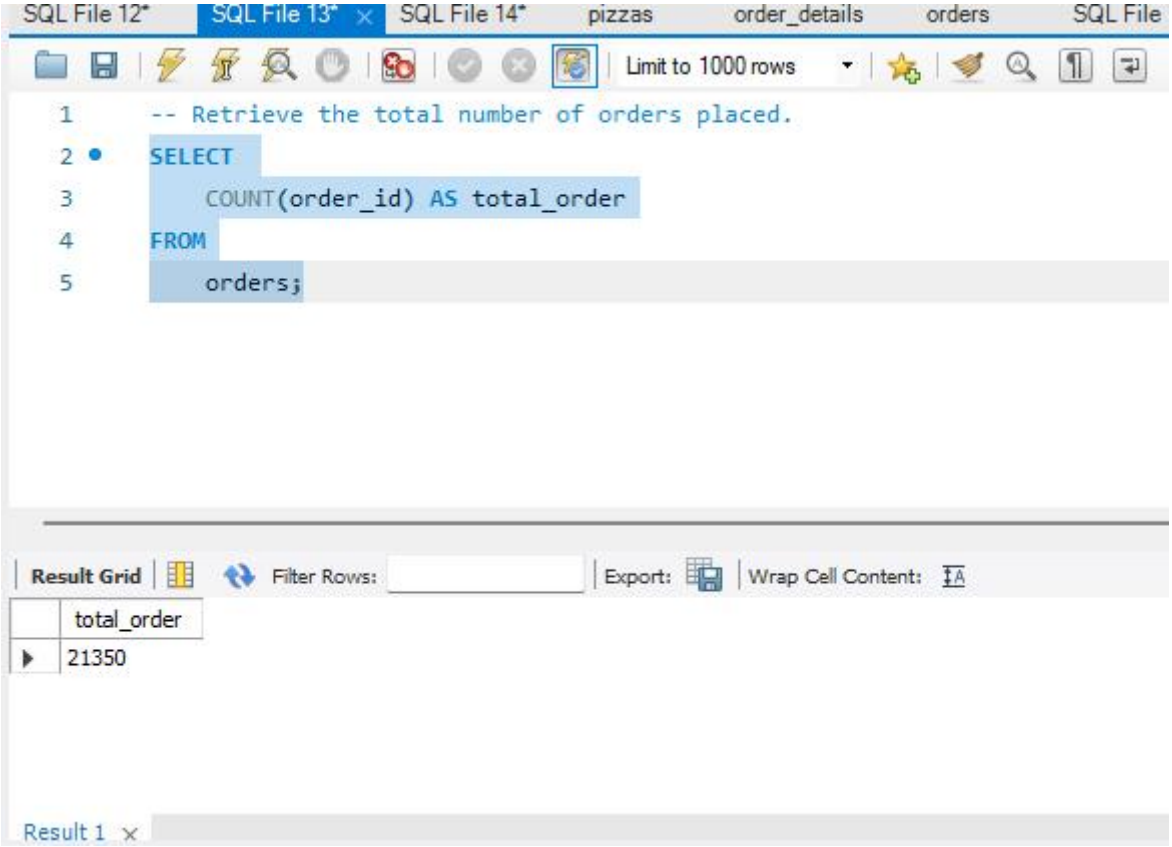
Question 2:

Retrieve the total number of orders placed.

Query:

```
SELECT
  COUNT(order_id) AS total_order
FROM
  orders;
```

Output:



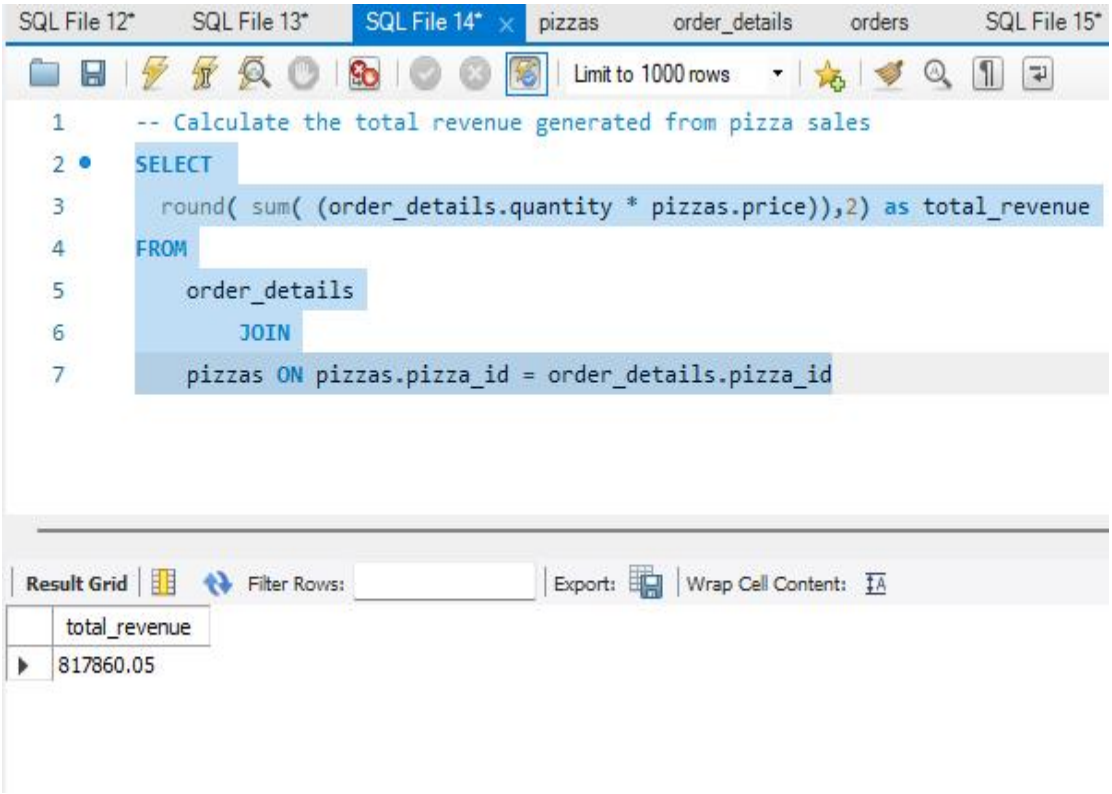
Question 3:

Calculate the total revenue generated from pizza sales

Query:

```
SELECT
 round( sum( (order_details.quantity * pizzas.price)),2) as total_revenue
FROM
  order_details
    JOIN
  pizzas ON pizzas.pizza_id = order_details.pizza_id
```

Output:



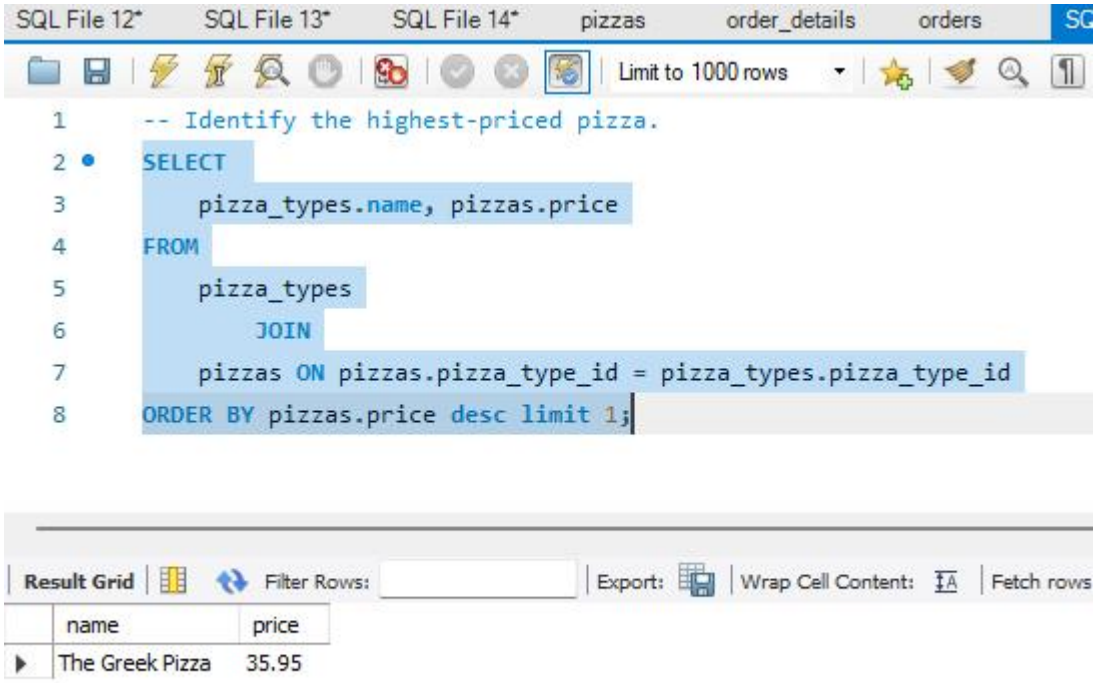
Question 4:

Identify the highest-priced pizza.

Query:

```
SELECT
   pizza_types.name, pizzas.price
FROM
   pizza_types
      JOIN
   pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
ORDER BY pizzas.price desc limit 1;
```

Output:



Question 5:

Identify the most common pizza size ordered.

Query:

```
SELECT
   pizzas.size,
   COUNT(order_details.order_detail_id) AS order_count
FROM
   order_details
      JOIN
   pizzas ON pizzas.pizza_id = order_details.pizza_id
GROUP BY size
ORDER BY order_count DESC;
```

Output:

```
-- Identify the most common pizza size ordered.
    pizzas.size,
    COUNT(order_details.order_detail_id) AS order_count
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
GROUP BY size
ORDER BY order_count DESC;
```

size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

Question 6:

List the top 5 most ordered pizza types along with their quantities.

Query:

```
SELECT
   pizza_types.name, SUM(order_details.quantity) as top_5_orders
FROM
   pizza_types
      JOIN
   pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
      JOIN
   order_details ON order_details.pizza_id = pizzas.pizza_id
group by pizza_types.name order by top_5_orders desc limit 5
;
```

Output:

```
-- List the top 5 most ordered pizza types along with their quantitie
SELECT
    pizza_types.name, SUM(order_details.quantity) as top_5_orders
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON order_details.pizza_id = pizzas.pizza_id
group by pizza_types.name order by top_5_orders desc limit 5
;
```

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

name	top_5_orders
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

Question 7:

Join the necessary tables to find the total quantity of each pizza category ordered.

Query:

```
SELECT
  pizza_types.category,
  SUM(order_details.quantity) AS total_quantity_of_each_pizza_category_ordered
FROM
  pizza_types
    JOIN
  pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
    JOIN
  order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category;
```

Output:

```
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS total_quantity_of_each_pizza_category_ordered
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category;
```

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

category	total_quantity_of_each_pizza_category_ordered
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

Question 8:

Determine the distribution of orders by hour of the day.

Query:

```
SELECT
   HOUR(order_time) as Every_Hour, COUNT(order_id) as Order_by_hour
FROM
   orders
GROUP BY HOUR(order_time) order by Order_by_hour desc ;
```

Output:

```sql
-- Determine the distribution of orders by hour of the day.
SELECT
    HOUR(order_time) as Every_Hour, COUNT(order_id) as Order_by_hour
FROM
    orders
GROUP BY HOUR(order_time) order by Order_by_hour desc ;
```

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

Every_Hour	Order_by_hour
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642
14	1472
15	1468
11	1231
21	1198
22	663
23	28
10	8
9	1

Question 9:

Join relevant tables to find the category-wise distribution of pizzas.

Query:

```sql
SELECT
    category, COUNT(name) AS number_of_pizzas
FROM
    pizza_types
GROUP BY category
ORDER BY number_of_pizzas DESC;
```

Output:

```sql
-- Join relevant tables to find the category

SELECT
    category,
    COUNT(name) AS number_of_pizzas
FROM
    pizza_types
GROUP BY
    category
ORDER BY
    number_of_pizzas DESC;
```

Result Grid | Filter Rows: | Export: | W

category	number_of_pizzas
Supreme	9
Veggie	9
Classic	8
Chicken	6

Question 10:

Group the orders by date and calculate the average number of pizzas ordered per day.

Query:

```sql
SELECT
  ROUND(AVG(quantity), 0)
FROM
  (SELECT
    orders.order_date, SUM(order_details.quantity) AS quantity
  FROM
    orders
  JOIN order_details ON orders.order_id = order_details.order_id
  GROUP BY orders.order_date) AS order_quantity;
```

Output:

```sql
-- Group the orders by date and calculate the
-- average number of pizzas ordered per day.

SELECT
    ROUND(AVG(quantity), 0)
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

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

round(avg(quantity),0)
138

Question 11:

Determine the top 3 most ordered pizza types based on revenue.

Query:

```sql
SELECT
  pizza_types.name,
  ROUND(SUM(order_details.quantity * pizzas.price),
      2) AS Quantity_of_orders
FROM
  pizza_types
    JOIN
  pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN
  order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY Quantity_of_orders DESC
```

LIMIT 3;

Output:

```
-- Determine the top 3 most ordered pizza types based on revenue.
SELECT
    pizza_types.name,
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS Quantity_of_orders
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY Quantity_of_orders DESC
LIMIT 3;
```

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

name	Quantity_of_orders
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

Question 12:

Calculate the percentage contribution of each pizza type to total revenue.

Query:

```
SELECT
  pizza_types.category,
  ROUND(SUM(pizzas.price * order_details.quantity) / (SELECT
          ROUND(SUM((order_details.quantity * pizzas.price)),
                2) AS total_revenue
        FROM
          order_details
            JOIN
```

```
                pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,
            1) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC
LIMIT 5;
```

Output:

```
-- Calculate the percentage contribution of each pizza type to total revenue.
SELECT
    pizza_types.category,
    ROUND(SUM(pizzas.price * order_details.quantity) / (SELECT
                    ROUND(SUM((order_details.quantity * pizzas.price)),
                                2) AS total_revenue
                FROM
                    order_details
                        JOIN
                    pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,
            1) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC
LIMIT 5;
```

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

category	revenue
Classic	26.9
Supreme	25.5
Chicken	24
Veggie	23.7

Question 13:

Analyze the cumulative revenue generated over time.

Query:

```
select order_date,
sum(revenue) over (order by order_date) as cum_revenue
from
(SELECT
   orders.order_date,
   sum(order_details.quantity * pizzas.price) AS revenue
FROM
   order_details
      JOIN
   pizzas ON order_details.pizza_id = pizzas.pizza_id
   join orders on
   orders.order_id=order_details.order_id group by orders.order_date)as sales;
```

Output:

```
-- Analyze the cumulative revenue generated over time
select order_date,
sum(revenue) over (order by order_date) as cum_revenue
from
(SELECT
    orders.order_date,
    sum(order_details.quantity * pizzas.price) AS revenue
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id
    join orders on
    orders.order_id=order_details.order_id group by orders.order_date)as sales;
```

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

order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7
2015-01-08	19399.05
2015-01-09	21526.4
2015-01-10	23990.350000000002

Question 14:

Determine the top 3 most ordered pizza types based on revenue for each pizza category.

Query:

select name, revenue from

(select category,name,revenue,

rank()over (partition by category order by revenue desc) as rn

from

(select pizza_types.category,pizza_types.name,sum((order_details.quantity)*pizzas.price)

as revenue

from pizza_types

join pizzas

on pizza_types.pizza_type_id = pizzas.pizza_type_id

join order_details

on order_details.pizza_id = pizzas.pizza_id

group by pizza_types.category,pizza_types.name) as a) as b

where rn <=3;

Output:

```
select name, revenue from
(select category,name,revenue,
rank()over (partition by category order by revenue desc) as rn
from
(select pizza_types.category,pizza_types.name,sum((order_details.quantity)*pizzas.price)
as revenue
from pizza_types
join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join order_details
on order_details.pizza_id = pizzas.pizza_id
group by  pizza_types.category,pizza_types.name) as a) as b
where rn <=3;
```

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

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Hawaiian Pizza	32273.25
The Pepperoni Pizza	30161.75
The Spicy Italian Pizza	34831.25
The Italian Supreme Pizza	33476.75
The Sicilian Pizza	30940.5
The Four Cheese Pizza	32265.70000000065