1) Which product has the highest price? Only return a single row.

Query #1

```
SELECT *
FROM products
ORDER BY price DESC
LIMIT 1;
```

product_id	product_name	price
13	Product M	70.00

2) Which customer has made the most orders?

Query #2

```
SELECT COUNT(customers.customer_id) AS counts, customers.customer_id, first_name, last_name
FROM orders
INNER JOIN customers
ON orders.customer_id=customers.customer_id
GROUP BY customers.customer_id, first_name, last_name
ORDER BY counts DESC
LIMIT 5;
```

counts	customer_id	first_name	last_name
2	3	Bob	Johnson
2	1	John	Doe
2	2	Jane	Smith
1	10	Lily	Nelson
1	13	Sophia	Thomas

3) What’s the total revenue per product?

Query #3

```
SELECT products.product_id, SUM(price * quantity) as total_revenue
FROM order_items
INNER JOIN products
ON order_items.product_id=products.product_id
GROUP BY products.product_id
ORDER BY total_revenue DESC;
```

product_id	total_revenue
13	420.00
10	330.00
6	210.00
12	195.00
11	180.00
3	160.00
9	150.00
8	135.00
2	135.00
7	120.00
5	90.00
4	75.00
1	50.00

4) Find the day with the highest revenue.

Query #4

```
SELECT orders.order_date, SUM(products.price * order_items.quantity) AS total_revenue
FROM orders
INNER JOIN order_items
ON orders.order_id=order_items.order_id
INNER JOIN products
ON order_items.product_id=products.product_id
GROUP BY orders.order_date
ORDER BY total_revenue DESC
LIMIT 1;
```

order_date	total_revenue
2023-05-16T00:00:00.000Z	340.00

5) Find the first order (by date) for each customer.

Query #5

```
SELECT customers.customer_id, first_name, last_name, MIN(order_date) AS first_order
FROM customers
JOIN orders
ON customers.customer_id=orders.customer_id
GROUP BY customers.customer_id, first_name, last_name;
```

customer_id	first_name	last_name	first_order
5	Charlie	Davis	2023-05-08T00:00:00.000Z
4	Alice	Brown	2023-05-07T00:00:00.000Z
10	Lily	Nelson	2023-05-13T00:00:00.000Z
6	Eva	Fisher	2023-05-09T00:00:00.000Z
13	Sophia	Thomas	2023-05-16T00:00:00.000Z
2	Jane	Smith	2023-05-02T00:00:00.000Z
7	George	Harris	2023-05-10T00:00:00.000Z
1	John	Doe	2023-05-01T00:00:00.000Z
8	Ivy	Jones	2023-05-11T00:00:00.000Z
11	Oliver	Patterson	2023-05-14T00:00:00.000Z
9	Kevin	Miller	2023-05-12T00:00:00.000Z
3	Bob	Johnson	2023-05-03T00:00:00.000Z
12	Quinn	Roberts	2023-05-15T00:00:00.000Z

6) Find the top 3 customers who have ordered the most distinct products

Query #6

```
SELECT customers.customer_id, first_name, last_name, COUNT(DISTINCT order_items.product_id) as distinct_product_count
FROM customers
INNER JOIN orders
ON customers.customer_id=orders.customer_id
INNER JOIN order_items
ON orders.order_id=order_items.order_id
GROUP BY customers.customer_id
ORDER BY distinct_product_count DESC
LIMIT 3;
```

customer_id	first_name	last_name	distinct_product_count
2	Jane	Smith	3
3	Bob	Johnson	3
1	John	Doe	3

7) Which product has been bought the least in terms of quantity?

Query #7

```
SELECT order_items.product_id, product_name, SUM(quantity) as sum_quant
FROM order_items
INNER JOIN products
ON order_items.product_id=products.product_id
GROUP BY order_items.product_id, product_name
ORDER BY sum_quant ASC
LIMIT 7;
```

product_id	product_name	sum_quant
7	Product G	3
5	Product E	3
4	Product D	3
11	Product K	3
12	Product L	3
9	Product I	3
8	Product H	3

8) What is the median order total?

Query #8

```
    SELECT PERCENTILE_CONT(0.5) WITHIN GROUP(ORDER BY price*quantity)
AS median_price_of_all_orders
    FROM order_items
    INNER JOIN products USING(product_id)
    INNER JOIN orders USING(order_id);
```

median_price_of_all_orders
55

9) For each order, determine if it was ‘Expensive’ (total over 300), ‘Affordable’ (total over 100), or ‘Cheap’.

Query #9

```
   SELECT orders.order_id,
SUM(price*quantity) AS total,
   CASE
                    WHEN
SUM(price*quantity) > 300 then
'Expensive'
                    WHEN
SUM(price*quantity) > 100 then
'Affordable'
                    ELSE 'Cheap'
   END AS order_description
   FROM products
   INNER JOIN order_items
USING(product_id)
   INNER JOIN orders USING(order_id)
   GROUP BY orders.order_id
   ORDER BY orders.order_id ASC;
```

order_id	total	order_description
1	35.00	Cheap
2	75.00	Cheap
3	50.00	Cheap
4	80.00	Cheap
5	50.00	Cheap
6	55.00	Cheap
7	85.00	Cheap
8	145.00	Affordable
9	140.00	Affordable
10	285.00	Affordable
11	275.00	Affordable
12	80.00	Cheap
13	185.00	Affordable
14	145.00	Affordable
15	225.00	Affordable
16	340.00	Expensive

10) Find customers who have ordered the product with the highest price.

Query #10

```
WITH highest_priced_item as (
 SELECT *
 FROM products
 ORDER BY price DESC
 LIMIT 1
)
SELECT CONCAT(customers.first_name, ' ', customers.last_name) AS full_name, products.product_name, price
FROM customers
INNER JOIN orders USING(customer_id)
INNER JOIN order_items USING(order_id)
INNER JOIN products USING(product_id)
WHERE product_name = (SELECT product_name FROM highest_priced_item);
```

full_name	product_name	price
Ivy Jones	Product M	70.00
Sophia Thomas	Product M	70.00