PIZZA SALES SQL QUERIES

Q1. Retrieve the total number of orders placed.

```sql
Create database pizzahut;

-- Q1: Retrieve the total number of orders placed.

SELECT
    COUNT(order_id)
FROM
    orders;
```

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

count(order_id)
21350

Q2: Calculate the total revenue generated from pizza sales.

```sql
-- Q2: Calculate the total revenue generated from pizza sales.

SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_revenue
FROM
    order_details
        JOIN
    pizzas ON order_details.pizza_id = pizzas.pizza_id;
```

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

total_revenue
817860.05

Q3: Identify the highest-priced pizza

```sql
-- Q3: Identify the highest-priced pizza.

SELECT
    pizza_types.name, pizzas.price
FROM
    pizza_types
        LEFT JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
ORDER BY pizzas.price DESC
LIMIT 1;
```

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

name	price
The Greek Pizza	35.95

Q4: Identify the most common pizza size ordered.

```sql
## Q4: Identify the most common pizza size ordered.

SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
        LEFT JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;
```

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

size	order_count
L	18526

Q5: List the top 5 most ordered pizza types along with their quantities.

```sql
-- Q5: List the top 5 most ordered pizza types along with their quantities.

SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
        LEFT JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        LEFT JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;
```

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

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

Q6: Join the necessary tables to find the total quantity of each pizza category ordered.

```sql
-- Q6: Join the necessary tables to find the total quantity of each pizza category ordered.

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

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

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Q7: Determine the distribution of orders by hour of the day.

```sql
-- Q7: Determine the distribution of orders by hour of the day.


SELECT
    HOUR(order_time) AS order_hour, COUNT(order_id)
FROM
    orders
GROUP BY order_hour;
```

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

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

Q8: Join relevant tables to find the category-wise distribution of pizzas(count pizzas).

```sql
--  Q8: Join relevant tables to find the category-wise
--      distribution of pizzas(count pizzas).

SELECT
    category, COUNT(name)
FROM
    pizza_types
GROUP BY category;
```

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

category	COUNT(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

Q9: Group the orders by date and calculate the average number of pizzas ordered per day.

```sql
-- Q9: 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
    LEFT JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS avg_order;
```

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

ROUND(AVG(quantity),0)
138

Q10: Determine the top 3 most ordered pizza types based on revenue.

```sql
-- Q10: Determine the top 3 most ordered pizza types based on revenue.

SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        LEFT JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        LEFT JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

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

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

Q11: Calculate the percentage contribution of each pizza type(category) to total revenue.

```sql
-- Q11: Calculate the percentage contribution of each pizza type(category) to total revenue.

SELECT
    pizza_types.category,
    ROUND(ROUND(SUM(order_details.quantity * pizzas.price),
                    2) / (SELECT
                    ROUND(SUM(order_details.quantity * pizzas.price),
                                2)
                FROM
                    pizzas
                        LEFT JOIN
                    order_details ON pizzas.pizza_id = order_details.pizza_id) * 100,
            2) AS revenue
FROM
    pizza_types
        LEFT JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        LEFT JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

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

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Q12: Analyse the cumulative revenue generated over time.

```sql
-- Q12: Analyze the cumulative revenue generated over time.
SELECT order_date,
Round(sum(revenue) OVER(ORDER BY order_date),2) AS cumulative_revenue
FROM
    (SELECT
        orders.order_date,
        ROUND(SUM(order_details.quantity * pizzas.price),
                2) AS revenue
    FROM
        pizzas
            LEFT JOIN
        order_details ON pizzas.pizza_id = order_details.pizza_id
            LEFT JOIN
        orders ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) As revenue_by_date ;
```

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

order_date	cumulative_revenue
NULL	NULL
2015-01-01	2713.85
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	23990.35
2015-01-11	25862.65

Q13: Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```sql
-- Q13: Determine the top 3 most ordered pizza types based on revenue for each pizza category.
SELECT category, 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,
    ROUND(SUM(order_details.quantity * pizzas.price),2) AS revenue
FROM
    pizza_types
        LEFT JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        LEFT JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.category,pizza_types.name) as sq) as sq
WHERE rn<=3;
```

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

category	name	revenue
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.5
Veggie	The Four Cheese Pizza	32265.7
Veggie	The Mexicana Pizza	26780.75