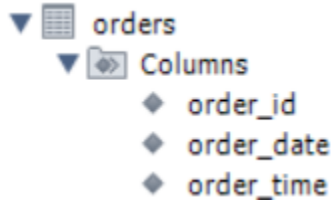


PIZZAHUT SALES - SQL PROJECT .

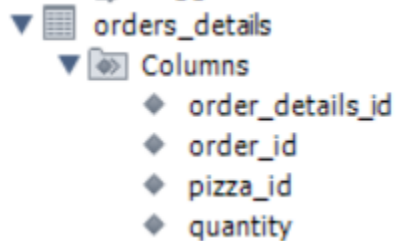
DATABASE NAME - PIZZAHUT

TABLES:

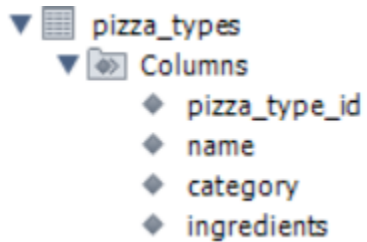
- ORDERS



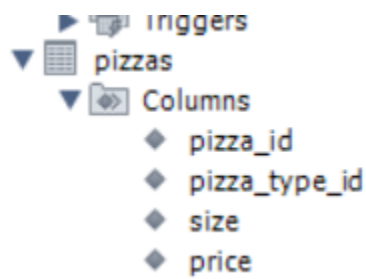
- ORDERS_DETAILS



- PIZZA_TYPES



- PIZZAS



SCREENSHOTS OF QUERIES

1. Retrieving total orders .

The screenshot displays a database management interface. On the left, the 'SCHEMAS' pane shows a tree view with 'pizzahut' expanded, containing tables like 'orders', 'orders_details', 'pizza_types', and 'pizzas'. The main query editor shows a SQL query: `-- Retrieve the total number of orders placed.` followed by `select count(order_id) as total_orders from orders;`. Below the query editor, the 'Result Grid' shows a single column 'total_orders' with the value '21350'. The 'Output' pane at the bottom shows a log of database actions, including table creation, session variable setting, and the execution of the query, which returned 1 row.

Table: orders

Columns:

- order_id int PK
- order_date date
- order_time time

Result Grid

total_orders
21350

Output

#	Time	Action	Message
56	11:39:19	create table orders_details(order_details_id int not null, order_id int not null, pizza_id text not null, quantity int not null)	0 row(s) affected
57	11:39:39	SHOW SESSION VARIABLES LIKE 'lower_case_table_names'	OK
58	11:39:39	SHOW DATABASES	OK
59	11:39:40	SHOW SESSION VARIABLES LIKE 'lower_case_table_names'	OK
60	11:39:40	SHOW COLUMNS FROM 'pizzahut'. 'orders_details'	OK
61	11:39:44	PREPARE stmt FROM 'INSERT INTO 'pizzahut'. 'orders_details' ('order_details_id','order_id','pizza_id','quantity')	OK
62	11:44:03	DEALLOCATE PREPARE stmt	OK
63	11:52:23	select count(order_id) as total_orders from orders LIMIT 0, 1000	1 row(s) returned

2. Calculate the total revenue generated from pizza sales.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

Filter objects

SCHEMAS

pizzahut

Tables

orders

orders_details

Columns

Indexes

Foreign Keys

Triggers

pizza_types

pizzas

Views

Stored Procedures

Functions

project

sys

Query 1 SQL File 1* SQL File 2*

Limit to 1000 rows

```

1 -- Calculate the total revenue generated from pizza sales.
2
3 • select round (sum(orders_details.quantity * pizzas.price),2) as total_sales
4   from orders_details join pizzas
5   on pizzas.pizza_id = orders_details.pizza_id

```

Result Grid

total_sales
817860.05

Filter Rows:

Export:

Wrap Cell Content:

Result 8 x

Output

Action Output

#	Time	Action	Message
75	12:39:34	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned
76	12:40:00	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned
77	12:40:00	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned
78	12:40:01	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned
79	12:40:01	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned
80	12:40:02	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned
81	12:40:02	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned
82	12:40:03	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizz...	1 row(s) returned

Schema: pizzahut

3. To calculate highest pricing of pizzas

File Edit View Query Database Server Tools Scripting Help

Navigator: Filter objects

SCHEMAS

- pizzahut
 - Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
 - Views
 - Stored Procedures
 - Functions
 - project
 - sys

Query 1 SQL File 1* SQL File 2* SQL File 5*

```

1  -- Identify the highest-priced pizza.
2  •  SELECT
3      pizza_types.name, pizzas.price
4  FROM
5      pizza_types
6      JOIN
7      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
8  ORDER BY pizzas.price DESC
9  LIMIT 1

```

Result Grid

name	price
The Greek Pizza	35.95

Result 1 x

Output

Action Output

#	Time	Action	Message
1	08:04:25	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizza...	1 row(s) returned
2	08:10:30	select pizza_types.name , pizzas price from pizza_types join pizzas on pizza_types pizza_type_id = pizzas pizza_t...	1 row(s) returned

No object selected

4. Identify the most common pizza size ordered.

Navigator: CHEMAS

- Filter objects
- pizzahut
 - Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
 - Views
 - Stored Procedures
 - Functions
- project
- sys

Query 1 SQL File 1* SQL File 2* SQL File 5* SQL File 6* x

```

1  -- Identify the most common pizza size ordered.
2
3  SELECT
4      pizzas.size,
5      COUNT(orders_details.order_details_id) AS order_count
6  FROM
7      pizzas
8      JOIN
9      orders_details ON pizzas.pizza_id = orders_details.pizza_id
10 GROUP BY pizzas.size
11 ORDER BY order_count DESC;
12

```

Result Grid

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

Result 1 x

Output

Action Output

#	Time	Action	Message
1	08:04:25	select round (sum(orders_details.quantity * pizzas.price),2) as total_sales from orders_details join pizzas on pizza...	1 row(s) returned
2	08:10:30	select pizza_types.name , pizzas price from pizza_types join pizzas on pizza_types.pizza_type_id = pizzas.pizza_t...	1 row(s) returned
3	08:42:37	select pizzas.size, count(orders_details.order_details_id) as order_count from pizzas join orders_details on pizzas....	5 row(s) returned

No object selected

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

Navigator: pizzahut

- Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
- Views
- Stored Procedures
- Functions

Query 1 SQL File 1* SQL File 2* SQL File 5* SQL File 6* SQL File 7* x

```

1  -- List the top 5 most ordered pizza types along with their quantities.
2
3  SELECT
4      pizza_types.name, SUM(orders_details.quantity) AS quantity
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9      JOIN
10     orders_details ON orders_details.pizza_id = pizzas.pizza_id
11 GROUP BY pizza_types.name
12 ORDER BY quantity DESC
13 LIMIT 5;

```

Result Grid

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

Result 1 x

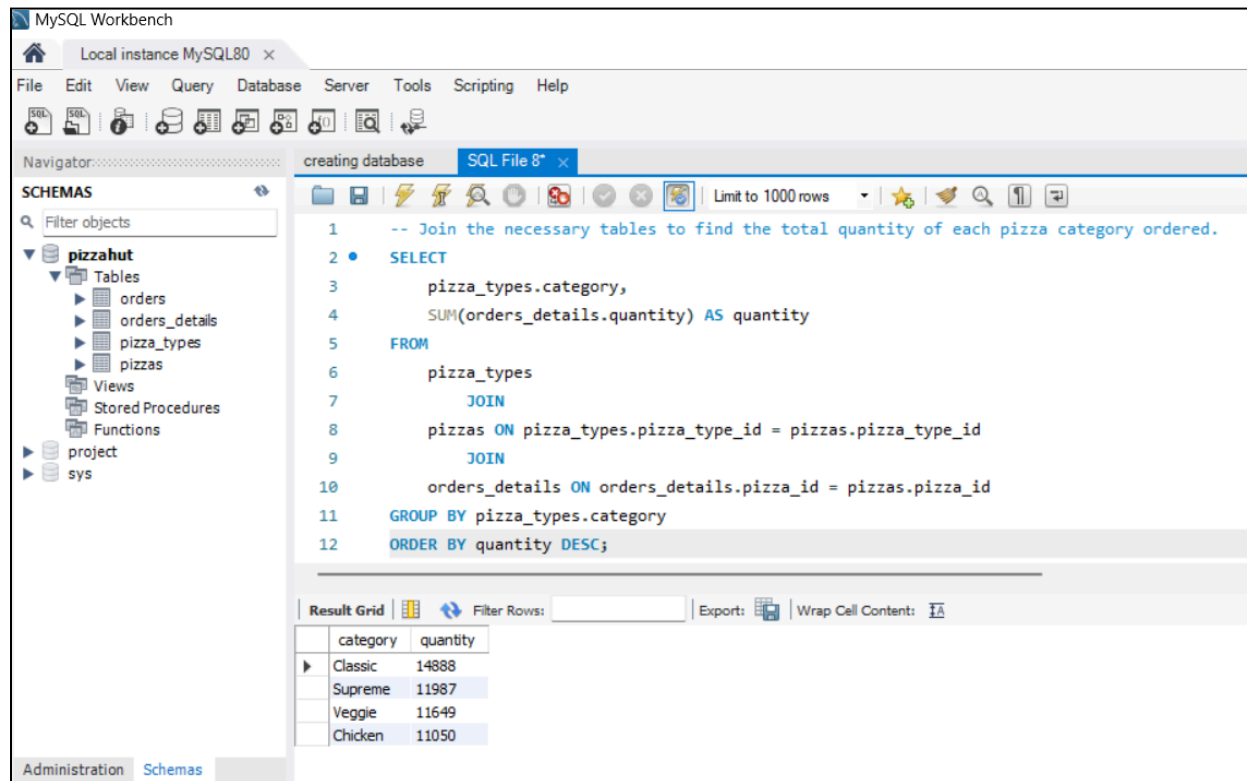
Output

Action Output

#	Time	Action	Message
3	08:42:37	select pizzas.size, count(orders_details.order_details_id) as order_count from pizzas join orders_details on pizzas....	5 row(s) returned

Object selected

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



The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays the 'pizzahut' database structure, including tables like 'orders', 'orders_details', 'pizza_types', and 'pizzas'. The main editor window shows a SQL query designed to find the total quantity of each pizza category ordered. The query uses a series of JOINs to connect the 'pizza_types' table with the 'pizzas' and 'orders_details' tables. The results are displayed in a 'Result Grid' at the bottom, showing four categories: Classic, Supreme, Veggie, and Chicken, with their respective total quantities.

```
1  -- Join the necessary tables to find the total quantity of each pizza category ordered.
2  •  SELECT
3      pizza_types.category,
4      SUM(orders_details.quantity) AS quantity
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9      JOIN
10     orders_details ON orders_details.pizza_id = pizzas.pizza_id
11  GROUP BY pizza_types.category
12  ORDER BY quantity DESC;
```

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

7. Determine the distribution of orders by hour of the day.

HEMAS

Filter objects

pizzahut

- Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
- Views
- Stored Procedures
- Functions

project

sys

creating database Join the necessary tables to find... SQL File 10*

Limit to 1000 rows

```

1 -- Determine the distribution of orders by hour of the day.
2
3 • SELECT
4     HOUR(order_time) AS hour, COUNT(order_id) AS order_count
5 FROM
6     orders
7 GROUP BY HOUR(order_time)
8 ;

```

Result Grid

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920

Result 1 x

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

HEMAS

Filter objects

pizzahut

- Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
- Views
- Stored Procedures
- Functions

project

sys

creating database SQL File 11*

Limit to 1000 rows

```

1 -- Join relevant tables to find the category-wise distribution of pizzas.
2 • SELECT
3     category, COUNT(name)
4 FROM
5     pizza_types
6 GROUP BY category;

```

Result Grid

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

Result 1 x

Output

Action Output

#	Time	Action	Message
8	08:52:29	select pizza_types.name, sum(orders_details.quantity) as quantity from pizza_types join pizzas on pizza_types....	5 row(s) returned

No object selected

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

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: creating database SQL File 11* SQL File 12* x

Limit to 1000 rows

1 -- Group the orders by date and calculate the average number of pizzas ordered per day.
 2
 3 • SELECT
 4 ROUND(AVG(QUANTITY), 0)
 5 FROM
 6 (SELECT
 7 orders.order_date, SUM(orders_details.quantity) AS quantity
 8 FROM
 9 orders
 10 JOIN orders_details ON orders.order_id = orders_details.order_id
 11 GROUP BY orders.order_date) AS order_quantity;

Result Grid Filter Rows: Export: Wrap Cell Content: [...](#)

ROUND(AVG(QUANTITY), 0)
 138

Administration Schemas

Information

No object selected

Result 5 x

Output

Action Output

#	Time	Action	Message
13	09:51:41	select category , count(name) from pizza_types group by category LIMIT 0, 1000	4 row(s) returned

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

Navigator

creating database SQL File 11* SQL File 12* SQL File 13* x

Limit to 1000 rows

1 -- Determine the top 3 most ordered pizza types based on revenue.
2
3 • SELECT
4 pizza_types.name,
5 SUM(orders_details.quantity * pizzas.price) AS revenue
6 FROM
7 pizza_types
8 JOIN
9 pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
10 JOIN
11 orders_details ON orders_details.pizza_id = pizzas.pizza_id
12 GROUP BY pizza_types.name
13 ORDER BY revenue DESC
14 LIMIT 3;

Administration Schemas

Information

No object selected

Result Grid

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

Result 1 x

Output

Action Output

#	Time	Action	Message
14	10:16:18	SELECT round(AVG(QUANTITY),0)FROM (SELECT orders.order_date,sum(orders_details.quantity) AS quanti...	1 row(s) returned

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

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: creating database SQL File 14*

SCHEMAS

Filter objects

- pizzahut
 - Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
 - Views
 - Stored Procedures
 - Functions
- project
- sys

SQL File 14*

```

3 • SELECT
4     pizza_types.category,
5     ROUND(SUM(orders_details.quantity * pizzas.price) / (SELECT
6         ROUND(SUM(orders_details.quantity * pizzas.price),
7             2) AS total_sales
8     FROM
9         orders_details
10    JOIN
11        pizzas ON pizzas.pizza_id = orders_details.pizza_id) * 100,
12    2) AS revenue
13 FROM
14     pizza_types
15    JOIN
16     pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
17    JOIN
18     orders_details ON orders_details.pizza_id = pizzas.pizza_id
19 GROUP BY pizza_types.category
20 ORDER BY revenue DESC;

```

Administration Schemas

Information

No object selected

Result Grid

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Result 1 x

Output

Action Output

#	Time	Action	Message
19	10:21:20	select pizza_types.name, sum(orders_details.quantity * pizzas.price) as revenue from pizza_types join pizzas on ...	3 row(s) returned

12. Analyze the cumulative revenue generated over time.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator: creating database percentage contribution of each... SQL File 15* x

SCHEMAS

Filter objects

- pizzahut
 - Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
 - Views
 - Stored Procedures
 - Functions
 - project
 - sys

Administration Schemas

Information

No object selected

```

1  -- Analyze the cumulative revenue generated over time.
2
3  • select order_date,
4     sum(revenue)over(order by order_date) as cum_revenue
5  from
6  ( select orders.order_date,
7     sum(orders_details.quantity*pizzas.price) as revenue
8  from orders_details join pizzas
9   on orders_details.pizza_id = pizzas.pizza_id
10 join orders
11  on orders.order_id = orders_details.order_id
12  group by orders.order_date) as sales;

```

Result Grid

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

Result 1 x

Output

Action Output

#	Time	Action	Message
20	10:35:39	select pizza_types.category, round(sum(orders_details.quantity*pizzas price) / (select round(sum(orders_details...	4 row(s) returned

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

Local instance mySQL 8.0.17

File Edit View Query Database Server Tools Scripting Help

Navigator: creating database percentage contribution of each... SQL File 15* SQL File 16*

Limit to 1000 rows

SCHEMAS

Filter objects

pizzahut

- Tables
 - orders
 - orders_details
 - pizza_types
 - pizzas
- Views
- Stored Procedures
- Functions
- project
- sys

Administration Schemas

Information: No object selected

```

1  -- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
2  • select name , revenue from
3  ( select category , name , revenue , rank() over(partition by category order by revenue desc) as rn
4  from
5  (select pizza_types.category, pizza_types.name,
6  sum((orders_details.quantity) * pizzas.price) as revenue
7  from pizza_types join pizzas
8  on pizza_types.pizza_type_id = pizzas.pizza_type_id
9  join orders_details
10 on orders_details.pizza_id = pizzas.pizza_id
11 group by pizza_types.category , pizza_types.name )as a) as b
12 where rn <= 3;

```

Result Grid

	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	32265.7000000065

Result 1

Output

Action Output

#	Time	Action	Message
21	10:46:26	select order_date, sum(revenue)over(order by order_date) as cum_revenue from (select orders.order_date, su...	358 row(s) returned

Object Info Session

THANK YOU