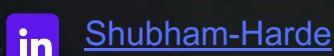


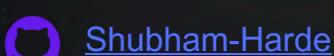
This SQL project on pizza sales reports aims to utilize data analysis methods to extract valuable insights from a database, empowering stakeholders to make informed decisions and propel business growth in the competitive pizza industry.

PIZZA SALES ANALYSIS WITH SQL

Hello, my name is Shubham Harde. In this project, I utilized SQL queries to address various questions related to pizza sales.



[Shubham-Harde](#)



[Shubham-Harde](#)



shubham.harde@yahoo.com

Business Analyst | Data Analyst

PREVIEWING CSV FILES: FIRST 5 ROWS

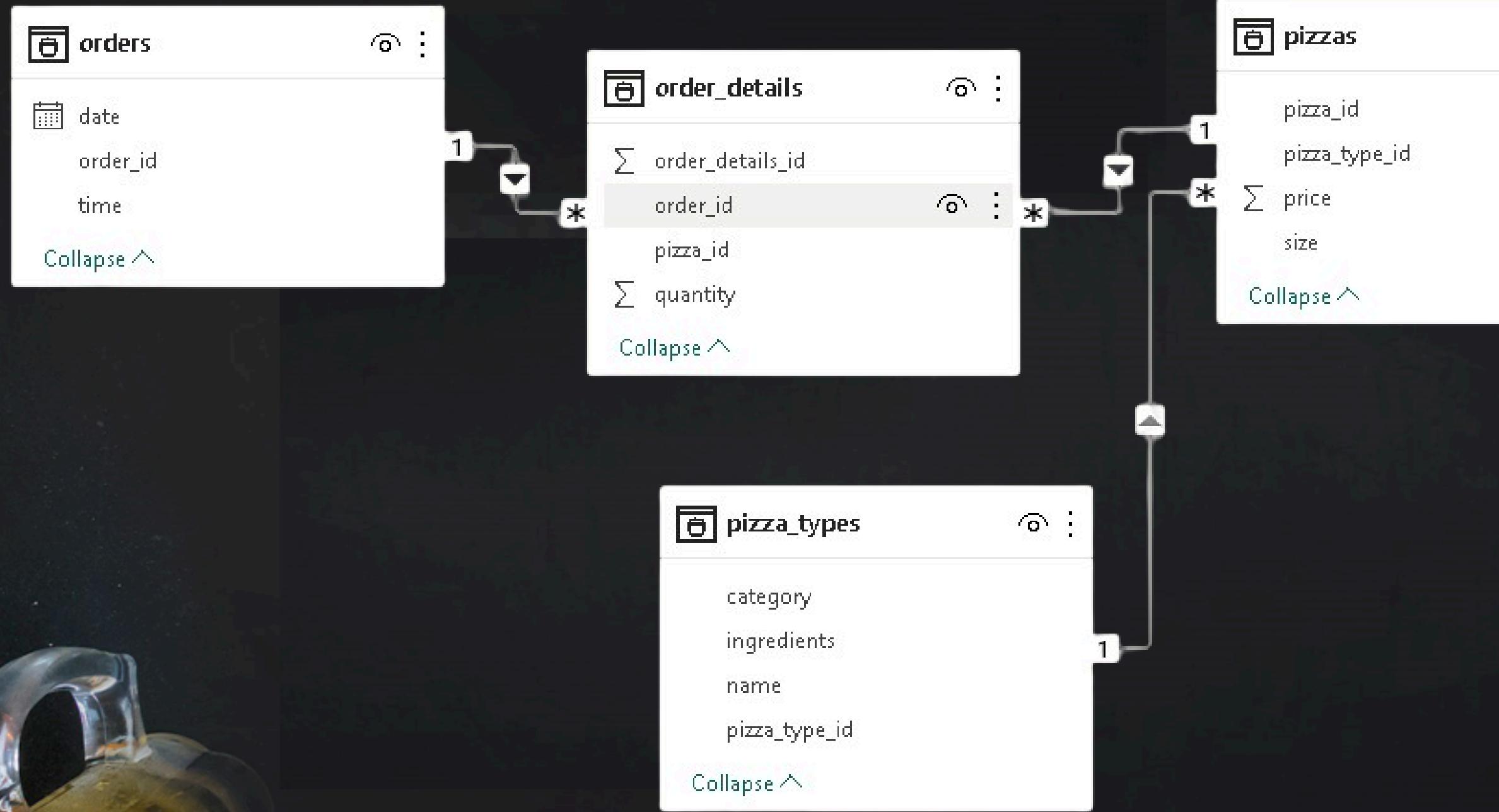
order_id	date	time
19402	Friday, 27 November, 2015	11:21:54 AM
19403	Friday, 27 November, 2015	11:29:51 AM
19404	Friday, 27 November, 2015	11:32:40 AM
19405	Friday, 27 November, 2015	11:36:39 AM
19406	Friday, 27 November, 2015	11:46:40 AM

pizza_id	pizza_type_id	size	price
bbq_ckn_s	bbq_ckn	S	12.75
bbq_ckn_m	bbq_ckn	M	16.75
bbq_ckn_l	bbq_ckn	L	20.75
cali_ckn_s	cali_ckn	S	12.75
cali_ckn_m	cali_ckn	M	16.75

order_details_id	order_id	pizza_id	quantity
36	15	big_meat_s	1
56	20	big_meat_s	1
79	32	big_meat_s	1
101	42	big_meat_s	1
150	64	big_meat_s	1

pizza_type_id	name	category	ingredients
bbq_ckn	The Barbecue Chicken Pizza	Chicken	Barbecued Chicken, Red Peppers, Green Peppers, Tomatoes, Red Onions, Barbecue Sauce
cali_ckn	The California Chicken Pizza	Chicken	Chicken, Artichoke, Spinach, Garlic, Jalapeno Peppers, Fontina Cheese, Gouda Cheese
ckn_alfredo	The Chicken Alfredo Pizza	Chicken	Chicken, Red Onions, Red Peppers, Mushrooms, Asiago Cheese, Alfredo Sauce
ckn_pesto	The Chicken Pesto Pizza	Chicken	Chicken, Tomatoes, Red Peppers, Spinach, Garlic, Pesto Sauce
southw_ckn	The Southwest Chicken Pizza	Chicken	Chicken, Tomatoes, Red Peppers, Red Onions, Jalapeno Peppers, Corn, Cilantro, Chipotle Sauce

MODEL VIEW



SQL QUESTIONS FOR ANALYSIS

Basic:

- Retrieve the total number of orders placed.
- Calculate the total revenue generated from pizza sales.
- Identify the highest-priced pizza.
- Identify the most common pizza size ordered.
- List the top 5 most ordered pizza types along with their quantities.

Intermediate:

- Join the necessary tables to find the total quantity of each pizza category ordered.
- Determine the distribution of orders by hour of the day.
- Join relevant tables to find the category-wise distribution of pizzas.
- Group the orders by date and calculate the average number of pizzas ordered per day.
- Determine the top 3 most ordered pizza types based on revenue.

Advanced:

- Calculate the percentage contribution of each pizza type to total revenue.
- Analyze the cumulative revenue generated over time.
- Determine the top 3 most ordered pizza types based on revenue for each pizza category.

DATABASE INITIALIZATION AND DATA IMPORT



The image shows a close-up of a pizza slice with various toppings, including olives, cheese, and vegetables, set against a dark background.

```
SCHEMAS
Filter objects
pizzahut
Tables
orders
    Columns
        order_id
        order_date
        order_time
    Indexes
    Foreign Keys
    Triggers
orders_details
    Columns
        order_details_
        order_id
        pizza_id
        quantity
    Indexes
    Foreign Keys
    Triggers
pizza_types
    Columns
        pizza_type_id
        name
        category
        ingredients
    Indexes
    Foreign Keys
    Triggers
pizzas
    Columns
        pizza_id
        pizza_type_id
        size
        price

1 • CREATE DATABASE IF NOT EXISTS pizzahut;
2
3 • CREATE TABLE orders(
4     order_id INT NOT NULL,
5     order_date DATE NOT NULL,
6     order_time TIME NOT NULL,
7     PRIMARY KEY(order_id) );
8
9 • CREATE TABLE orders_details(
10    order_details_id INT NOT NULL,
11    order_id INT NOT NULL,
12    pizza_id TEXT NOT NULL,
13    quantity INT NOT NULL,
14    PRIMARY KEY(order_details_id) );
```

- Created "pizzahut" database.
- Imported "pizzas" and "pizza_types" CSV files into MySQL using the data import wizard.
- Created "orders" and "order_details" tables due to incorrect field types for specific columns during CSV file import in MySQL.
- Imported the same CSV files into the newly created tables in MySQL using the table data import wizard.

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

Query:

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

Output:

Result Grid	
	total_orders
▶	21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

Query:

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

Output:

Result Grid	
	total_sales
▶	817860.05



IDENTIFY THE HIGHEST-PRICED PIZZA.

Query:

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

Output:

Result Grid | Filter Rows:

	name	price
▶	The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

Query:

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

Output:

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

LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

Query:

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

Output:

	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

JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

Query:

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

Output:

	category	quantity_ordered
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

Query:

```
• SELECT  
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count  
FROM  
    orders  
GROUP BY HOUR(order_time);
```

Output:

hour	order_count
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

JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

Query:

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

Output:

	category	pizza_count
▶	Chicken	6
	Classic	6
	Supreme	9
	Veggie	9

GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

Query:

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

Output:

Result Grid		Filter Rows:
	avg_pizza_ordered_per_day	
▶	138	

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

Query:

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

Output:

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

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

Query:

```
SELECT
    pizza_types.category,
    CONCAT(ROUND((SUM(orders_details.quantity * pizzas.price) / (SELECT
        SUM(orders_details.quantity * pizzas.price)
    FROM
        orders_details
        JOIN
            pizzas ON pizzas.pizza_id = orders_details.pizza_id)) * 100,
    2), ' %') AS revenue_percentage
FROM
    pizza_types
    JOIN
        pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN
        orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue_percentage DESC;
```

Output:

	category	revenue_percentage
▶	Classic	26.91 %
	Supreme	25.46 %
	Chicken	23.96 %
	Veggie	23.68 %

 Shubham-Harde

 Shubham-Harde

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(orders_details.quantity*pizzas.price) AS revenue  
FROM orders_details  
JOIN pizzas  
ON orders_details.pizza_id = pizzas.pizza_id  
JOIN orders  
ON orders.order_id = orders_details.order_id  
GROUP BY orders.order_date) AS sales;
```

Output:

	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

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

Query:

```
SELECT name, category, 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((orders_details.quantity)*pizzas.price) AS revenue
FROM pizza_types
JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN orders_details
ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category, pizza_types.name) AS a) AS b
WHERE rn<=3;
```

Output:

	name	category	revenue
▶	The Thai Chicken Pizza	Chicken	43434.25
	The Barbecue Chicken Pizza	Chicken	42768
	The California Chicken Pizza	Chicken	41409.5
	The Classic Deluxe Pizza	Classic	38180.5
	The Hawaiian Pizza	Classic	32273.25
	The Pepperoni Pizza	Classic	30161.75
	The Spicy Italian Pizza	Supreme	34831.25
	The Italian Supreme Pizza	Supreme	33476.75
	The Sicilian Pizza	Supreme	30940.5
	The Four Cheese Pizza	Veggie	32265.70000000065
	The Mexicana Pizza	Veggie	26780.75
	The Five Cheese Pizza	Veggie	26066.5

PIZZA SALES ANALYSIS: SQL-GENERATED INSIGHTS AND SKILLS DEMONSTRATE

- Total Orders: 21,350
- Revenue Generated: 817,860.05
- Highest-Priced Pizza: The Greek Pizza at 35.95
- Most Common Pizza Size: Large - 18,526 orders
- Top 5 Most Ordered Pizza Types:
 - Classic Deluxe: 2,453 orders
 - Barbecue Chicken: 2,432 orders
 - Hawaiian: 2,422 orders
 - Pepperoni: 2,418 orders
 - Thai Chicken: 2,371 orders
- Pizza Category Quantities:
 - Classic: 14,888 orders
 - Supreme: 11,987 orders
 - Veggie: 11,649 orders
 - Chicken: 11,050 orders
- Distribution of Orders by Hour
 - 12 PM: 2,520 orders
 - 1 PM: 2,455 orders
- Category-wise Pizza Distribution
 - Chicken: 6 pizzas
 - Classic: 8 pizzas
 - Supreme: 9 pizzas
 - Veggies: 9 pizzas

- Average Pizzas Ordered per Day: 138
- Top 3 Most Ordered Pizza Types by Revenue for Each Category
 - Chicken:
 - Thai Chicken Pizza: 43,434.25
 - Barbecue Chicken Pizza: 42,768
 - California Chicken Pizza: 41,409.5
 - Percentage Contribution of Pizza Types to Total Revenue
 - Classic: 26.91%
 - Supreme: 25.46%
 - Chicken: 23.96%
 - Veggie: 23.68%
 - Cumulative Revenue Over Time
 - 2015-01-01: 2,713.85
 - 2015-01-02: 5,445.75
 - Top 3 Most Ordered Pizza Types by Revenue
 - Thai Chicken Pizza: 43,434.25
 - Barbecue Chicken Pizza: 42,768
 - California Chicken Pizza: 41,409.5

SKILLS DEMONSTRATE

- 1) Data Retrieval:
 - SELECT
 - COUNT, SUM, ROUND, AVG
- 2) Data Filtering and Sorting:
 - ORDER BY, DESC
 - LIMIT
- 3) Data Aggregation:
 - GROUP BY
- 4) Joining Tables:
 - JOIN, ON
- 5) Subqueries
- 6) Window Functions:
 - OVER
 - PARTITION BY
 - RANK
- 7) Mathematical Functions:
 - COUNT
 - SUM
 - ROUND
 - AVG
- 8) Date/Time Functions:
 - HOUR
- 9) String Functions:
 - CONCAT
- 10) Aliases Functions:
 - AS



THANK YOU



[Shubham-Harde](#)



[Shubham-Harde](#)



shubham.harde@yahoo.com

Business Analyst | Data Analyst