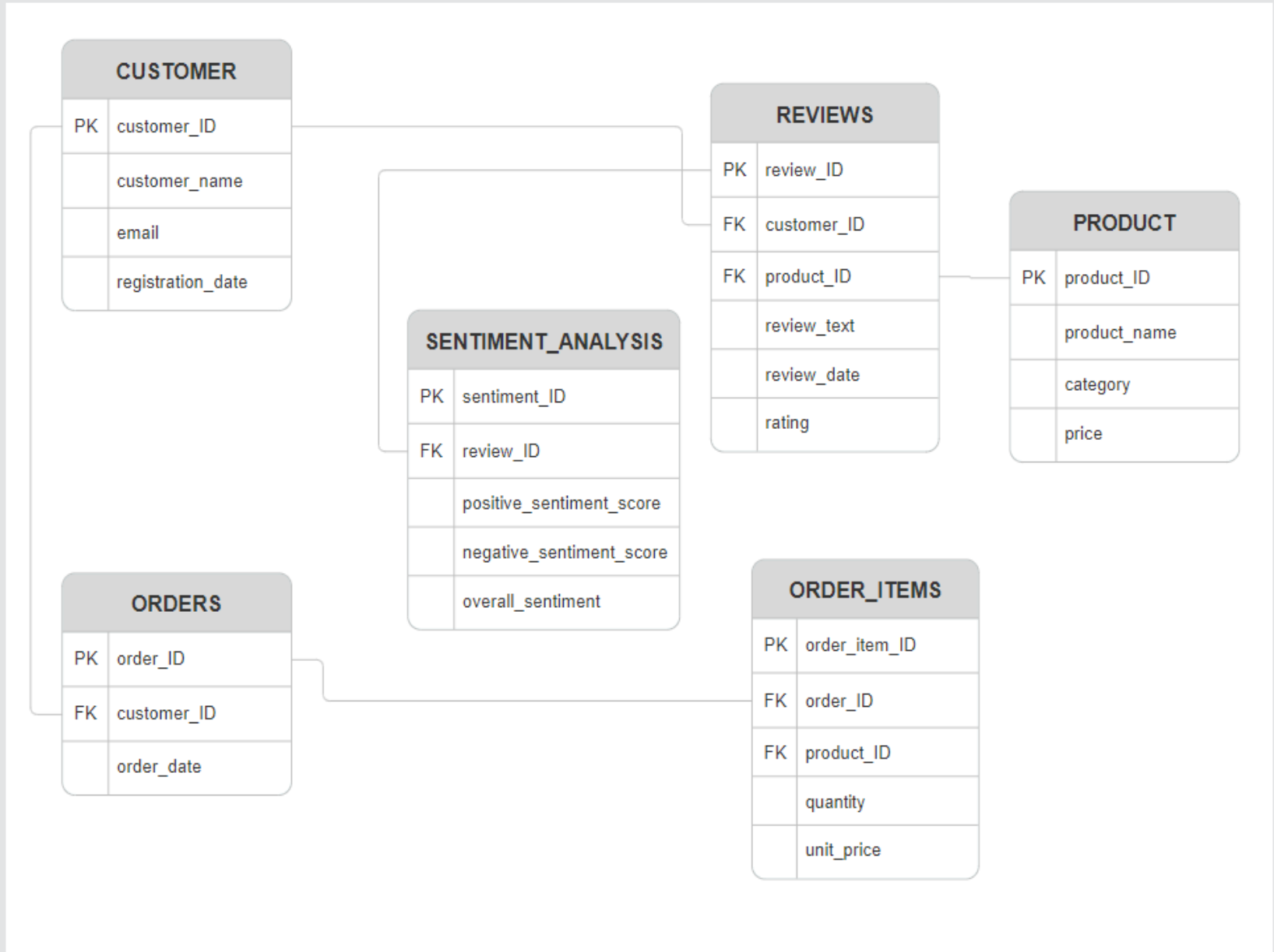
TEEMOOD SENTIMENT INSIGHT (TMSI) ENGINE

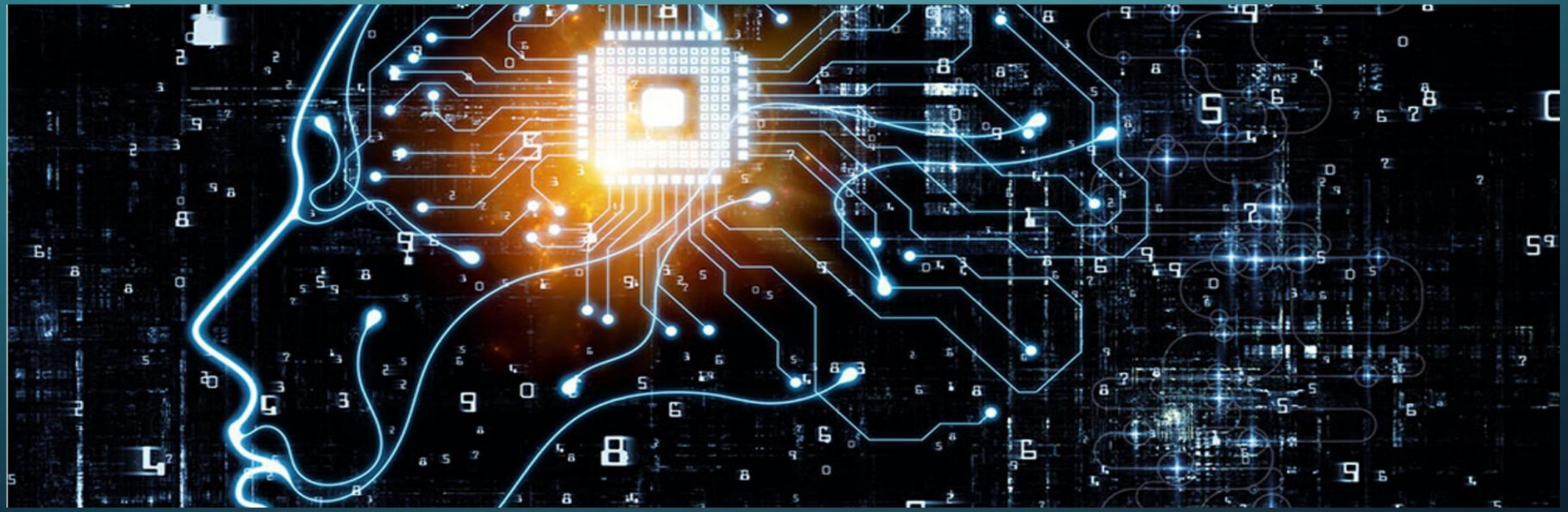
SYNOPSIS

- - The TeeMood Sentiment Insight (TMSI) Engine Is A Specialized SQL Project.
- - It Focuses On Understanding Customer Feelings About T-shirt Products And Services In An E-commerce Business.
- - The Project Analyzes Online Reviews And Feedback From Customers.
- - It Uses Advanced Techniques In Sql Queries To Figure Out The True Emotions And Opinions Expressed By Customers.
- - The Goal Is To Uncover Insights That Can Help Make Strategic Decisions, Improve Products, And Build Lasting Customer Satisfaction And Loyalty.

ER DIAGRAM


CUSTOMER
PK customer_ID
customer_name
email
registration_date
REVIEWS
PK review_ID
FK customer_ID
FK product_ID
review_text
review_date
rating
PRODUCT
PK product_ID
product_name
category
price
SENTIMENT_ANALYSIS
PK sentiment_ID
FK review_ID
positive_sentiment_score
negative_sentiment_score
overall_sentiment
ORDERS
PK order_ID
FK customer_ID
order_date
ORDER_ITEMS
PK order_item_ID
FK order_ID
FK product_ID
quantity
unit_price

STRUCTURE OF TABLES



SYNTAX: DESC CUSTOMERS;

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

Field	Type	Null	Key	Default	Extra
customer_id	int	NO	PRI	NULL	
customer_name	varchar(100)	YES		NULL	
email	varchar(100)	YES		NULL	
registration_date	date	YES		NULL	

SYNTAX: DESC PRODUCTS;

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

Field	Type	Null	Key	Default	Extra
product_id	int	NO	PRI	NULL	
product_name	varchar(100)	YES		NULL	
category	varchar(50)	YES		NULL	
price	decimal(10,2)	YES		NULL	

SYNTAX: DESC ORDERS;

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

Field	Type	Null	Key	Default	Extra
order_id	int	NO	PRI	NULL	
customer_id	int	YES	MUL	NULL	
order_date	date	YES		NULL	

SYNTAX: DESC ORDER_ITEMS;

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

Field	Type	Null	Key	Default	Extra
order_item_id	int	NO	PRI	NULL	
order_id	int	YES	MUL	NULL	
product_id	int	YES	MUL	NULL	
quantity	int	YES		NULL	
unit_price	decimal(10,2)	YES		NULL	

SYNTAX: DESC REVIEWS;

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

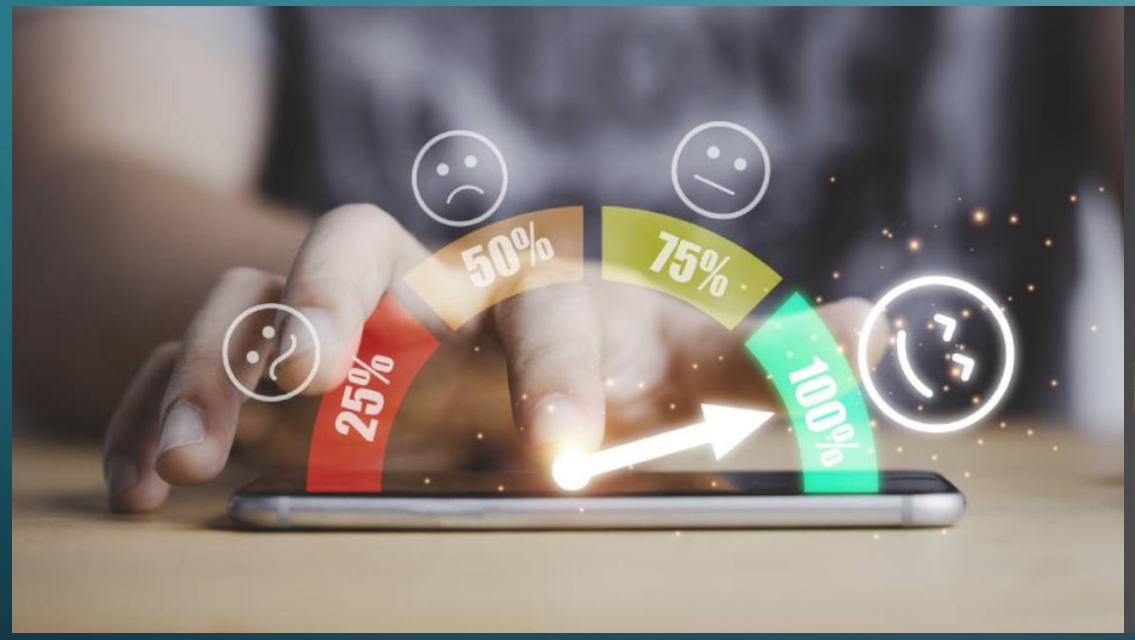
Field	Type	Null	Key	Default	Extra
review_id	int	NO	PRI	NULL	
customer_id	int	YES	MUL	NULL	
product_id	int	YES	MUL	NULL	
review_text	varchar(50)	YES		NULL	
review_date	date	YES		NULL	
rating	int	YES		NULL	

SYNTAX:
DESC SENTIMENT_ANALYSIS;

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

Field	Type	Null	Key	Default	Extra
sentiment_id	int	NO	PRI	NULL	
review_id	int	YES	MUL	NULL	
positive_sentiment_score	float	YES		NULL	
negative_sentiment_score	float	YES		NULL	
overall_sentiment	varchar(20)	YES		NULL	

CONTENTS OF TABLES



SYNTAX:
SELECT * FROM CUSTOMERS;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Conten

customer_id	customer_name	email	registration_date
1	John Doe	john@example.com	2024-01-01
2	Jane Smith	jane@example.com	2024-01-05
3	David Johnson	david@example.com	2024-01-10
4	Emily Brown	emily@example.com	2024-01-15
5	Michael Wilson	michael@example.com	2024-01-20
6	Jennifer Martinez	jennifer@example.com	2024-01-25
7	Christopher Anderson	christopher@example.com	2024-01-30
8	Amanda Taylor	amanda@example.com	2024-02-05
9	Daniel Thomas	daniel@example.com	2024-02-10
10	Sarah Garcia	sarah@example.com	2024-02-15
11	Matthew Rodriguez	matthew@example.com	2024-02-20
12	Laura Martinez	laura@example.com	2024-02-25
13	Justin Hernandez	justin@example.com	2024-03-01
14	Rebecca Garcia	rebecca@example.com	2024-03-05
15	Nathan Martinez	nathan@example.com	2024-03-10
16	Kimberly Lopez	kimberly@example.com	2024-03-15
17	Andrew Perez	andrew@example.com	2024-03-20
18	Stephanie Gonzalez	stephanie@example.com	2024-03-25
19	Jonathan Hernandez	jonathan@example.com	2024-03-30
20	Melissa Gonzalez	melissa@example.com	2024-04-05
NULL	NULL	NULL	NULL

SYNTAX:
SELECT * FROM PRODUCTS;

Result Grid | Filter Rows: | Edit:

product_id	product_name	category	price
1	Graphic Tee	Men	19.99
2	V-Neck Tee	Women	14.99
3	Long Sleeve Tee	Men	24.99
4	Tank Top	Women	12.99
5	Polo Shirt	Men	29.99
6	Crop Top	Women	16.99
7	Hoodie	Men	34.99
8	Sweatshirt	Women	27.99
9	Ringer Tee	Men	22.99
10	Tie-Dye Tee	Women	18.99
11	Cargo Shorts	Men	39.99
12	Denim Skirt	Women	32.99
13	Chino Pants	Men	49.99
14	Maxi Dress	Women	54.99
15	Track Pants	Men	29.99
16	Leggings	Women	24.99
17	Button-Down Shirt	Men	44.99
18	Blouse	Women	34.99
19	Cargo Pants	Men	42.99
20	Sundress	Women	39.99
NULL	NULL	NULL	NULL

SYNTAX:
SELECT * FROM ORDERS;

Result Grid | Filter Rows: | Edit:

order_id	customer_id	order_date
1	1	2024-01-01
2	2	2024-01-05
3	3	2024-01-10
4	4	2024-01-15
5	5	2024-01-20
6	6	2024-01-25
7	7	2024-01-30
8	8	2024-02-05
9	9	2024-02-10
10	10	2024-02-15
11	11	2024-02-20
12	12	2024-02-25
13	13	2024-03-01
14	14	2024-03-05
15	15	2024-03-10
16	16	2024-03-15
17	17	2024-03-20
18	18	2024-03-25
19	19	2024-03-30
20	20	2024-04-05
NULL	NULL	NULL

SYNTAX:
SELECT * FROM ORDER_ITEMS;

Result Grid | Filter Rows: | Edit:

order_item_id	order_id	product_id	quantity	unit_price
1	1	1	2	19.99
2	2	2	1	14.99
3	3	3	3	24.99
4	4	4	2	12.99
5	5	5	1	29.99
6	6	6	1	16.99
7	7	7	2	34.99
8	8	8	1	27.99
9	9	9	3	22.99
10	10	10	2	18.99
11	11	11	1	39.99
12	12	12	2	32.99
13	13	13	1	49.99
14	14	14	1	54.99
15	15	15	2	29.99
16	16	16	2	24.99
17	17	17	1	44.99
18	18	18	1	34.99
19	19	19	2	42.99
20	20	20	1	39.99
NULL	NULL	NULL	NULL	NULL

SYNTAX:
SELECT * FROM REVIEWS;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap C

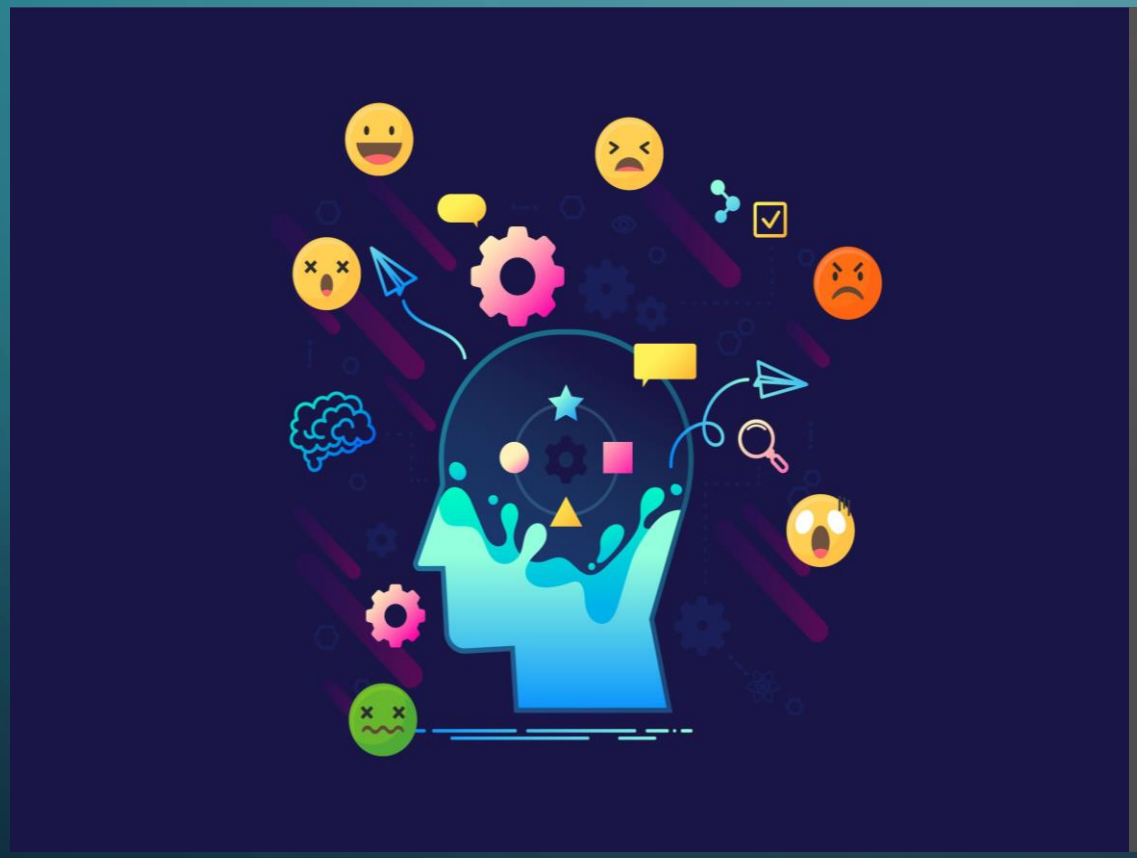
review_id	customer_id	product_id	review_text	review_date	rating
1	1	1	Great quality shirt!	2024-01-02	5
2	2	2	Love the fit!	2024-01-06	4
3	3	3	Sleeves are too long.	2024-01-11	3
4	4	4	Comfortable and stylish.	2024-01-16	5
5	5	5	Runs a bit large.	2024-01-21	4
6	6	6	Perfect for summer!	2024-01-26	5
7	7	7	Hoodie is too thin.	2024-01-31	2
8	8	8	Warm and cozy.	2024-02-06	4
9	9	9	Great vintage look.	2024-02-11	5
10	10	10	Colors are vibrant!	2024-02-16	4
11	11	11	Very comfortable shorts.	2024-02-21	4
12	12	12	Nice quality denim skirt.	2024-02-26	5
13	13	13	Good fit, but fabric wrin...	2024-03-02	3
14	14	14	Beautiful dress, perfect...	2024-03-06	5
15	15	15	Great for workouts.	2024-03-11	4
16	16	16	Love these leggings!	2024-03-16	5
17	17	17	Nice shirt for casual wear.	2024-03-21	4
18	18	18	Comfortable blouse.	2024-03-26	4
19	19	19	Lots of pockets, very p...	2024-03-31	4
20	20	20	Beautiful sundress.	2024-04-06	5
NULL	NULL	NULL	NULL	NULL	NULL

SYNTAX:
SELECT * FROM SENTIMENT_ANALYSIS;

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell

sentiment_id	review_id	positive_sentiment_score	negative_sentiment_score	overall_sentiment
1	1	0.8	0.2	Positive
2	2	0.7	0.3	Positive
3	3	0.4	0.6	Negative
4	4	0.9	0.1	Positive
5	5	0.6	0.4	Positive
6	6	0.85	0.15	Positive
7	7	0.3	0.7	Negative
8	8	0.75	0.25	Positive
9	9	0.8	0.2	Positive
10	10	0.7	0.3	Positive
11	11	0.75	0.25	Positive
12	12	0.9	0.1	Positive
13	13	0.5	0.5	Neutral
14	14	0.85	0.15	Positive
15	15	0.8	0.2	Positive
16	16	0.9	0.1	Positive
17	17	0.7	0.3	Positive
18	18	0.75	0.25	Positive
19	19	0.6	0.4	Positive
20	20	0.85	0.15	Positive
NULL	NULL	NULL	NULL	NULL

SUB-QUERIES



-To Calculate The Total Revenue Generated From Sales Of Products In Each Category.

SYNTAX:

```
 SELECT  P.CATEGORY,
SUM(OI.QUANTITY * OI.UNIT_PRICE) AS TOTAL_REVENUE
FROM     PRODUCTS P
INNER  JOIN
ORDER_ITEMS OI ON P.PRODUCT_ID = OI.PRODUCT_ID
GROUP BY
P.CATEGORY;
```

Result Grid | Filter Rows: | Export:

category	total_revenue
Men	564.82
Women	369.86

-Retrieves Customers Who Have Ordered The Most Expensive Product, Which Can Provide Insights Into High-value Customers

SYNTAX:

```
SELECT C.CUSTOMER_ID,C.CUSTOMER_NAME
FROM CUSTOMERS C
WHERE C.CUSTOMER_ID IN (SELECT O.CUSTOMER_ID
 FROM ORDERS O
INNER JOIN
 ORDER_ITEMS OI ON O.ORDER_ID = OI.ORDER_ID
 WHERE  OI.UNIT_PRICE = (SELECT MAX(UNIT_PRICE)
FROM  ORDER_ITEMS));
```

Result Grid | Filter Rows: | Edit:

customer_id	customer_name
14	Rebecca Garcia
NULL	NULL

- Calculates The Total Number Of Orders Placed By Each Customer, Which Can Be Used To Identify Loyal Customers Or Segment Customers Based On Their Purchasing Behavior:

SYNTAX:

```
 SELECT C.CUSTOMER_ID, C.CUSTOMER_NAME,
(SELECT COUNT(*)
FROM      ORDERS O
WHERE O.CUSTOMER_ID = C.CUSTOMER_ID)
AS TOTAL_ORDERSFROM     CUSTOMERS C;
```

Result Grid | Filter Rows: | Export:

customer_id	customer_name	total_orders
1	John Doe	1
2	Jane Smith	1
3	David Johnson	1
4	Emily Brown	1
5	Michael Wilson	1
6	Jennifer Martinez	1
7	Christopher Anderson	1
8	Amanda Taylor	1
9	Daniel Thomas	1
10	Sarah Garcia	1
11	Matthew Rodriguez	1
12	Laura Martinez	1
13	Justin Hernandez	1
14	Rebecca Garcia	1
15	Nathan Martinez	1
16	Kimberly Lopez	1
17	Andrew Perez	1
18	Stephanie Gonzalez	1
19	Jonathan Hernandez	1
20	Melissa Gonzalez	1

- To Calculate The Total Revenue Generated By Each Customer By Summing Up The Total Amount Spent On All Orders Placed By That Customer:

SYNTAX:

```
SELECT C.CUSTOMER_ID, C.CUSTOMER_NAME,
(SELECT SUM(OI.QUANTITY * OI.UNIT_PRICE)
FROM ORDERS O
INNER JOIN ORDER_ITEMS OI ON O.ORDER_ID =
OI.ORDER_ID
WHERE   O.CUSTOMER_ID = C.CUSTOMER_ID)
AS TOTAL_REVENUEFROM     CUSTOMERS C;
```

Result Grid | Filter Rows: | Export:

customer_id	customer_name	total_revenue
1	John Doe	39.98
2	Jane Smith	14.99
3	David Johnson	74.97
4	Emily Brown	25.98
5	Michael Wilson	29.99
6	Jennifer Martinez	16.99
7	Christopher Anderson	69.98
8	Amanda Taylor	27.99
9	Daniel Thomas	68.97
10	Sarah Garcia	37.98
11	Matthew Rodriguez	39.99
12	Laura Martinez	65.98
13	Justin Hernandez	49.99
14	Rebecca Garcia	54.99
15	Nathan Martinez	59.98
16	Kimberly Lopez	49.98
17	Andrew Perez	44.99
18	Stephanie Gonzalez	34.99
19	Jonathan Hernandez	85.98
20	Melissa Gonzalez	39.99

- To Find Customers Who Placed Orders Worth More Than Average:

```
SYNTAX: SELECT C.CUSTOMER_ID, C.CUSTOMER_NAME
FROM CUSTOMERS C
WHERE C.CUSTOMER_ID IN
(SELECT O.CUSTOMER_ID
 FROM ORDERS O
WHERE O.ORDER_ID IN
(SELECT  ORDER_ID
FROM  ORDER_ITEMS
GROUP BY
ORDER_ID
 HAVING  SUM(QUANTITY * UNIT_PRICE) >
(SELECT AVG(ORDER_TOTAL)
FROM (SELECT SUM(QUANTITY * UNIT_PRICE)
AS ORDER_TOTAL
FROM
ORDER_ITEMS
GROUP BY
ORDER_ID)
AS AVG_ORDERS)));
```

Result Grid Filter Rows:

customer_id	customer_name
3	David Johnson
7	Christopher Anderson
9	Daniel Thomas
12	Laura Martinez
13	Justin Hernandez
14	Rebecca Garcia
15	Nathan Martinez
16	Kimberly Lopez
19	Jonathan Hernandez
NULL	NULL

JOINS

-To Display Customers Who Have Ordered The Most Expensive Product Along With Their Details:

SYNTAX:

```
SELECT  C.CUSTOMER_ID, C.CUSTOMER_NAME
FROM
CUSTOMERS C
JOIN
(SELECT O.CUSTOMER_ID
FROM ORDERS O
INNER JOIN
ORDER_ITEMS OI
ON O.ORDER_ID = OI.ORDER_ID
WHERE OI.UNIT_PRICE =
(SELECT  MAX(UNIT_PRICE)
FROM ORDER_ITEMS)) AS T;
```

Result Grid | Filter Rows:

customer_id	customer_name
1	John Doe
2	Jane Smith
3	David Johnson
4	Emily Brown
5	Michael Wilson
6	Jennifer Martinez
7	Christopher Anderson
8	Amanda Taylor
9	Daniel Thomas
10	Sarah Garcia
11	Matthew Rodriguez
12	Laura Martinez
13	Justin Hernandez
14	Rebecca Garcia
15	Nathan Martinez
16	Kimberly Lopez
17	Andrew Perez
18	Stephanie Gonzalez
19	Jonathan Hernandez
20	Melissa Gonzalez

- To Display Total Number Of Orders Placed By Each Customer Along With Their Details:

SYNTAX:

```
SELECT  C.CUSTOMER_ID, C.CUSTOMER_NAME,
T.TOTAL_ORDERS
FROM CUSTOMERS C
JOIN
(SELECT O.CUSTOMER_ID, COUNT(*)
AS TOTAL_ORDERS
FROM ORDERS O
GROUP BY O.CUSTOMER_ID) AS T
ON C.CUSTOMER_ID = T.CUSTOMER_ID;
```

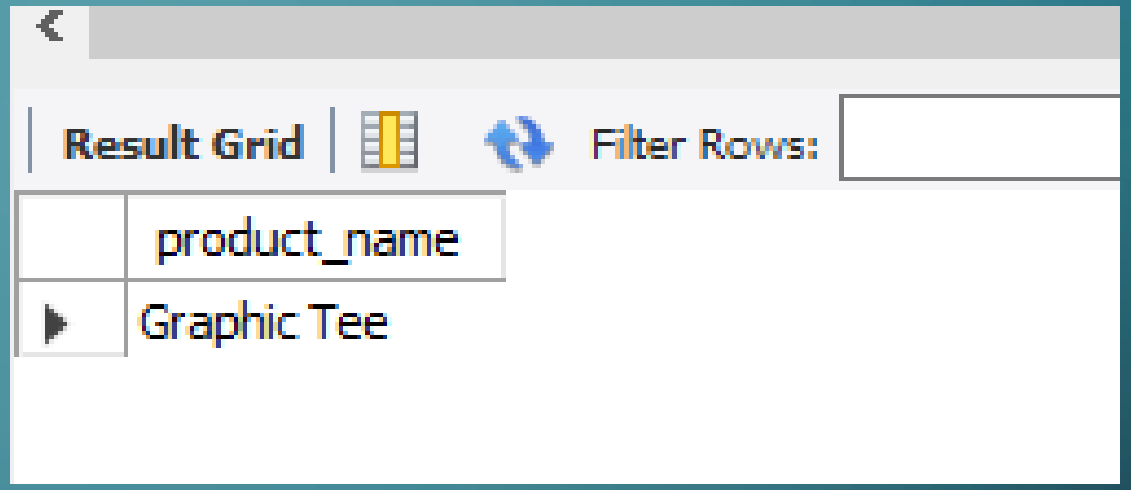
Result Grid | Filter Rows: | Export

customer_id	customer_name	total_orders
1	John Doe	1
2	Jane Smith	1
3	David Johnson	1
4	Emily Brown	1
5	Michael Wilson	1
6	Jennifer Martinez	1
7	Christopher Anderson	1
8	Amanda Taylor	1
9	Daniel Thomas	1
10	Sarah Garcia	1
11	Matthew Rodriguez	1
12	Laura Martinez	1
13	Justin Hernandez	1
14	Rebecca Garcia	1
15	Nathan Martinez	1
16	Kimberly Lopez	1
17	Andrew Perez	1
18	Stephanie Gonzalez	1
19	Jonathan Hernandez	1
20	Melissa Gonzalez	1

- To Display Product Name with the Highest Average Rating:

SYNTAX:

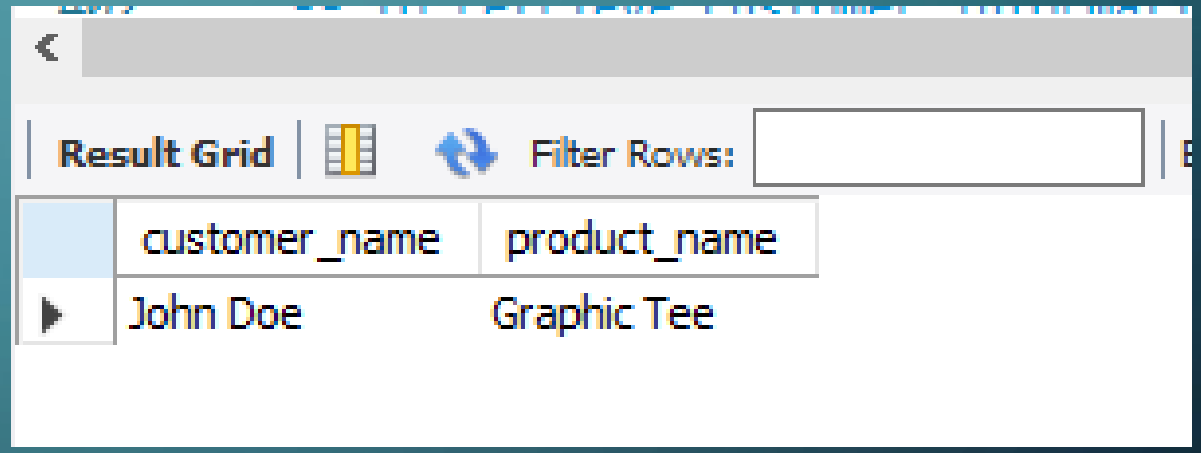
```
SELECT P.PRODUCT_NAME
FROM  PRODUCTS P
JOIN
(SELECT  R.PRODUCT_ID
FROM REVIEWS R
GROUP BY    R.PRODUCT_ID
ORDER BY    AVG(R.RATING)
DESC LIMIT 1) AS T
ON P.PRODUCT_ID = T.PRODUCT_ID;
```



- To Retrieve Customer Names with the Highest Rated Product They Ordered:

SYNTAX:

```
SELECT C.CUSTOMER_NAME, P.PRODUCT_NAME
FROM  CUSTOMERS C
JOIN
ORDERS O ON C.CUSTOMER_ID = O.CUSTOMER_ID
JOIN
ORDER_ITEMS OI ON O.ORDER_ID = OI.ORDER_IDJOIN
(SELECT R.PRODUCT_ID
FROM  REVIEWS R
GROUP BY  R.PRODUCT_ID
ORDER BY  AVG(R.RATING)
DESC    LIMIT 1)
AS TOP_PRODUCT ON OI.PRODUCT_ID =
TOP_PRODUCT.PRODUCT_ID
JOIN     PRODUCTS P
ON OI.PRODUCT_ID = P.PRODUCT_ID;
```



- To Retrieve Customer Information Along With Their Order Details:

SYNTAX:

```
SELECT O.ORDER_ID, C.CUSTOMER_NAME,
O.ORDER_DATE
FROM ORDERS O
JOIN CUSTOMERS C
ON O.CUSTOMER_ID = C.CUSTOMER_ID;
```

Result Grid | Filter Rows:

order_id	customer_name	order_date
1	John Doe	2024-01-01
2	Jane Smith	2024-01-05
3	David Johnson	2024-01-10
4	Emily Brown	2024-01-15
5	Michael Wilson	2024-01-20
6	Jennifer Martinez	2024-01-25
7	Christopher Anderson	2024-01-30
8	Amanda Taylor	2024-02-05
9	Daniel Thomas	2024-02-10
10	Sarah Garcia	2024-02-15
11	Matthew Rodriguez	2024-02-20
12	Laura Martinez	2024-02-25
13	Justin Hernandez	2024-03-01
14	Rebecca Garcia	2024-03-05
15	Nathan Martinez	2024-03-10
16	Kimberly Lopez	2024-03-15
17	Andrew Perez	2024-03-20
18	Stephanie Gonzalez	2024-03-25
19	Jonathan Hernandez	2024-03-30
20	Melissa Gonzalez	2024-04-05

- Prepared By ANTHONY ALBERT