Restaurant Orders Management System

Final Project for SQL Module

by Aman Tadvi

Description :-

Following database schema is designed to function as a backend storage database for a web application built to manage a restaurant.

A quarter's worth of orders from a fictitious restaurant serving international cuisine, including the date and time of each order, the items ordered, and additional details on the type, name and price of the items.

This database contain 3 tables :-

1. Menu table
2. Order details table
3. Restaurant Sales table

- **Commands :-**

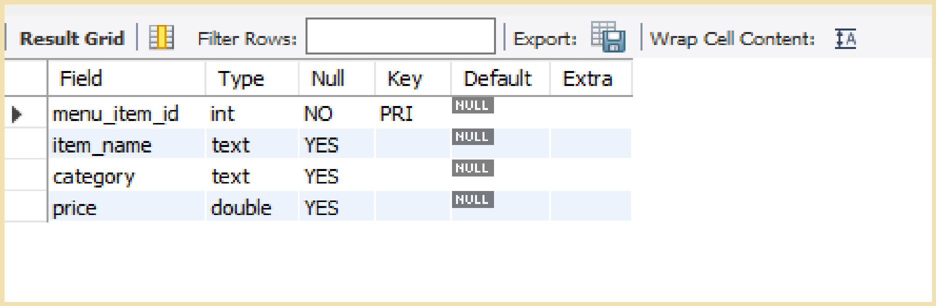
create database foodorder;

show foodorder;

use foodorder;

- **Table description :-**

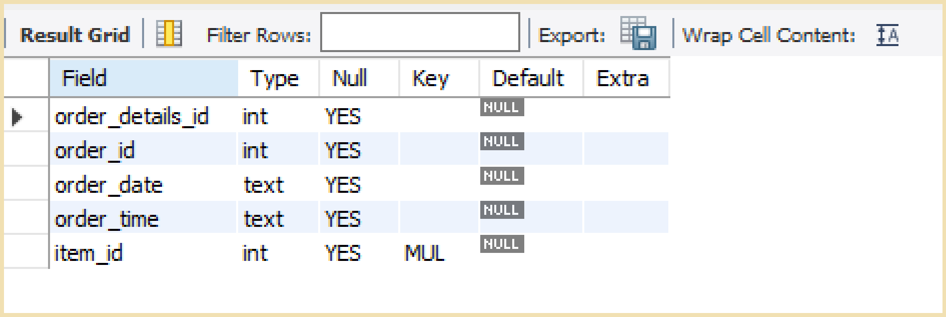
1) **Menu Table**
   desc menu_items;



Field	Type	Null	Key	Default	Extra
menu_item_id	int	NO	PRI	NULL	
item_name	text	YES		NULL	
category	text	YES		NULL	
price	double	YES		NULL	

2) **Order Table**

desc order_details ;

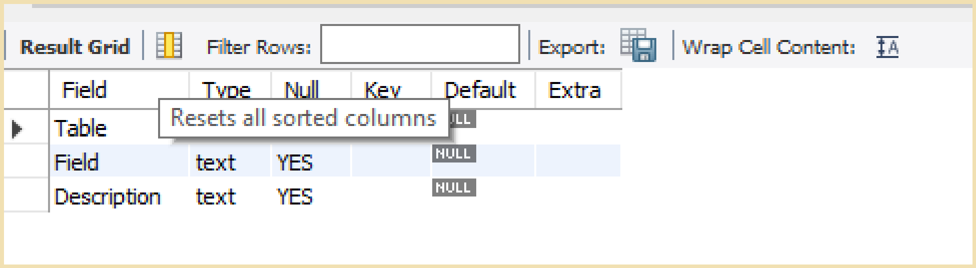


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
order_details_id	int	YES		NULL	
order_id	int	YES		NULL	
order_date	text	YES		NULL	
order_time	text	YES		NULL	
item_id	int	YES	MUL	NULL	

3) Restaurant Sales Table

desc restaurant ;



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

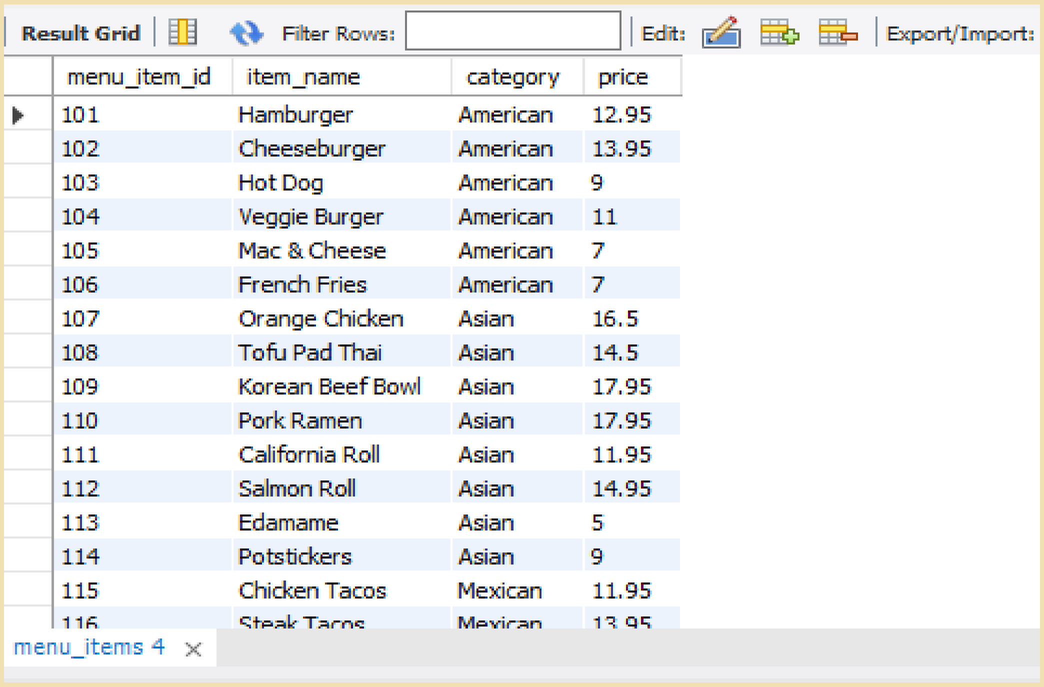
Resets all sorted columns

Field	Type	Null	Key	Default	Extra
Table				NULL	
Field	text	YES		NULL	
Description	text	YES		NULL	

- **Showing Tablets :-**

1) Menu Table

select * from menu_items;

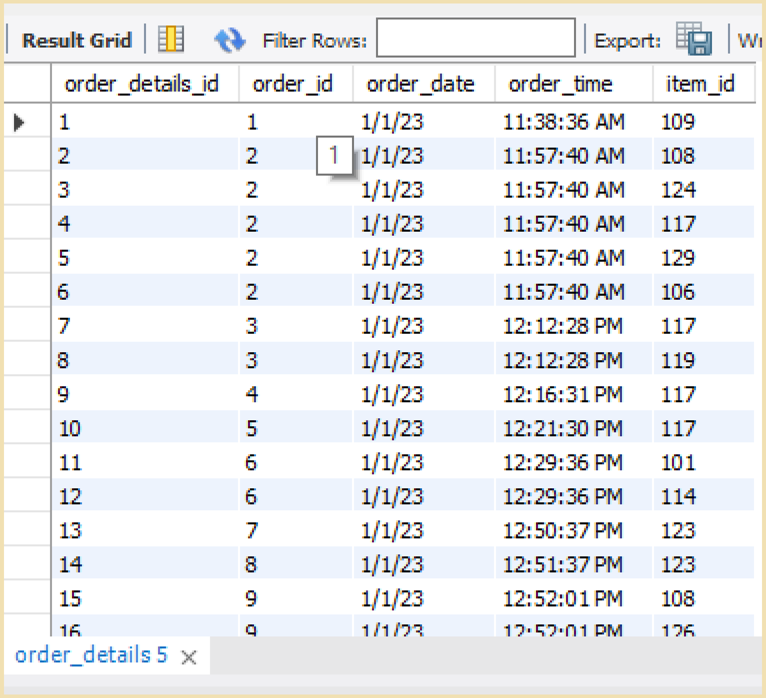

Result Grid | Filter Rows: | Edit: | Export/Import:

menu_item_id	item_name	category	price
101	Hamburger	American	12.95
102	Cheeseburger	American	13.95
103	Hot Dog	American	9
104	Veggie Burger	American	11
105	Mac & Cheese	American	7
106	French Fries	American	7
107	Orange Chicken	Asian	16.5
108	Tofu Pad Thai	Asian	14.5
109	Korean Beef Bowl	Asian	17.95
110	Pork Ramen	Asian	17.95
111	California Roll	Asian	11.95
112	Salmon Roll	Asian	14.95
113	Edamame	Asian	5
114	Potstickers	Asian	9
115	Chicken Tacos	Mexican	11.95
116	Steak Tacos	Mexican	13.95

menu_items 4

2) Order Table

select * from order_details;


Result Grid | Filter Rows: | Export:

order_details_id	order_id	order_date	order_time	item_id
1	1	1/1/23	11:38:36 AM	109
2	2	1/1/23	11:57:40 AM	108
3	2	1/1/23	11:57:40 AM	124
4	2	1/1/23	11:57:40 AM	117
5	2	1/1/23	11:57:40 AM	129
6	2	1/1/23	11:57:40 AM	106
7	3	1/1/23	12:12:28 PM	117
8	3	1/1/23	12:12:28 PM	119
9	4	1/1/23	12:16:31 PM	117
10	5	1/1/23	12:21:30 PM	117
11	6	1/1/23	12:29:36 PM	101
12	6	1/1/23	12:29:36 PM	114
13	7	1/1/23	12:50:37 PM	123
14	8	1/1/23	12:51:37 PM	123
15	9	1/1/23	12:52:01 PM	108
16	9	1/1/23	12:52:01 PM	126

order_details 5

3) Restaurant Sales Table

select * from restaurants;

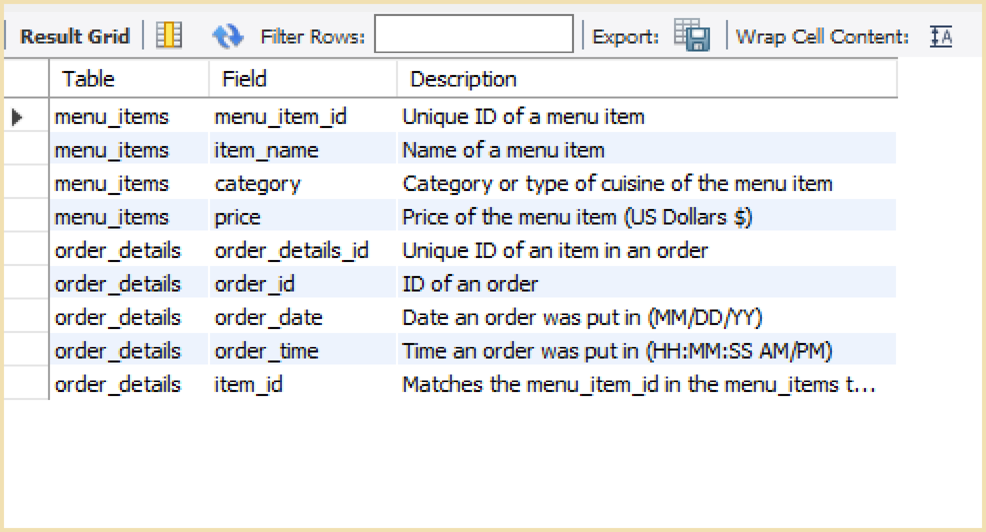


Table	Field	Description
menu_items	menu_item_id	Unique ID of a menu item
menu_items	item_name	Name of a menu item
menu_items	category	Category or type of cuisine of the menu item
menu_items	price	Price of the menu item (US Dollars $)
order_details	order_details_id	Unique ID of an item in an order
order_details	order_id	ID of an order
order_details	order_date	Date an order was put in (MM/DD/YY)
order_details	order_time	Time an order was put in (HH:MM:SS AM/PM)
order_details	item_id	Matches the menu_item_id in the menu_items t...

- **Primary Key :-**

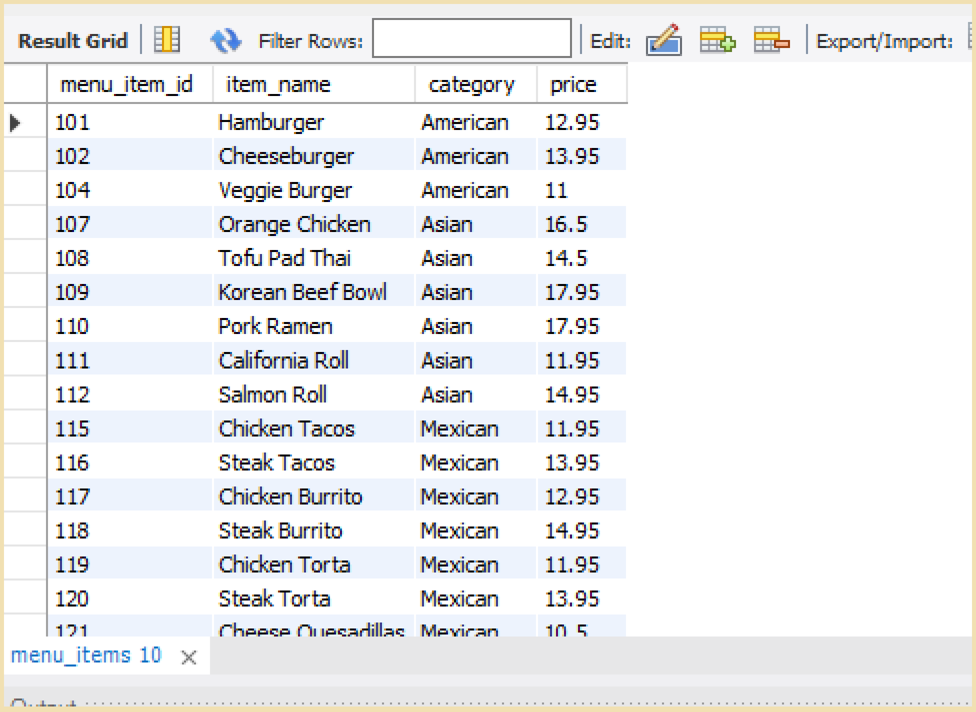
In this project , menu_items table comes with a pre_existing primary key on ' menu_item_id' which was already defined when the data was obtained from external sources.It uniquely identifies each menu item record in menu_items table.

- **Using Alter Tables and Columns Queries for adding foreign key :-**

```
alter table order_details
add
foreign key
(item_id) references menu_items(menu_item_id);
```

- **Using Where Clause to find Price more than $10 :-**

```
SELECT
    *
FROM
menu_items
WHERE
    price > 10;
```



menu_item_id	item_name	category	price
101	Hamburger	American	12.95
102	Cheeseburger	American	13.95
104	Veggie Burger	American	11
107	Orange Chicken	Asian	16.5
108	Tofu Pad Thai	Asian	14.5
109	Korean Beef Bowl	Asian	17.95
110	Pork Ramen	Asian	17.95
111	California Roll	Asian	11.95
112	Salmon Roll	Asian	14.95
115	Chicken Tacos	Mexican	11.95
116	Steak Tacos	Mexican	13.95
117	Chicken Burrito	Mexican	12.95
118	Steak Burrito	Mexican	14.95
119	Chicken Torta	Mexican	11.95
120	Steak Torta	Mexican	13.95
121	Cheese Quesadillas	Mexican	10.5

- **Using AND Clause to find order time in 12 am and 02 pm :-**

SELECT

*

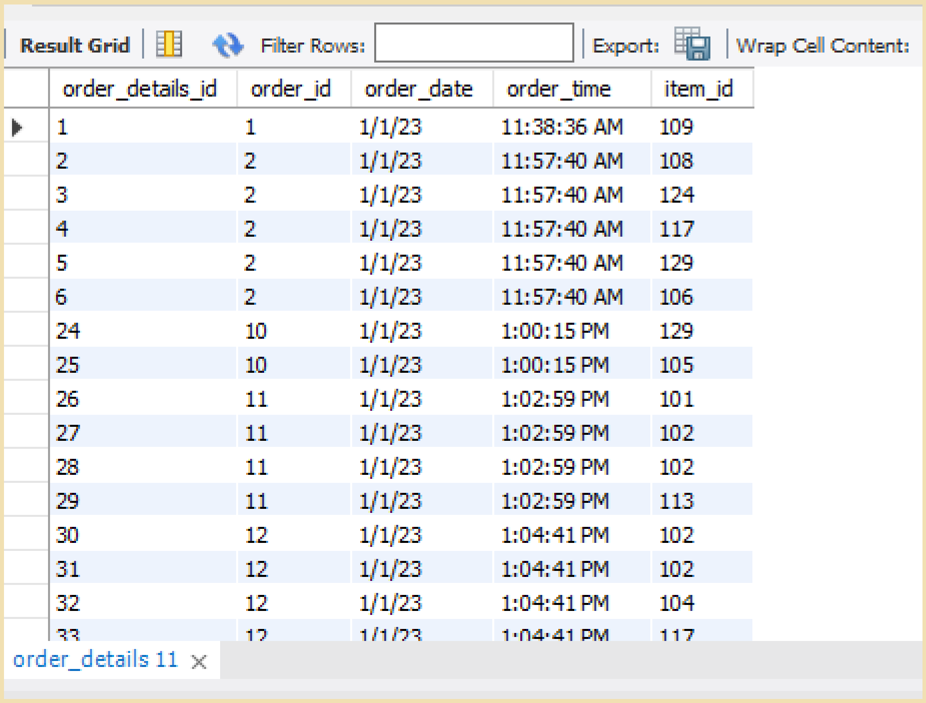
FROM

order_details

WHERE

order_time< '12:00:00 AM'

AND order_time> '02:00:00 PM'



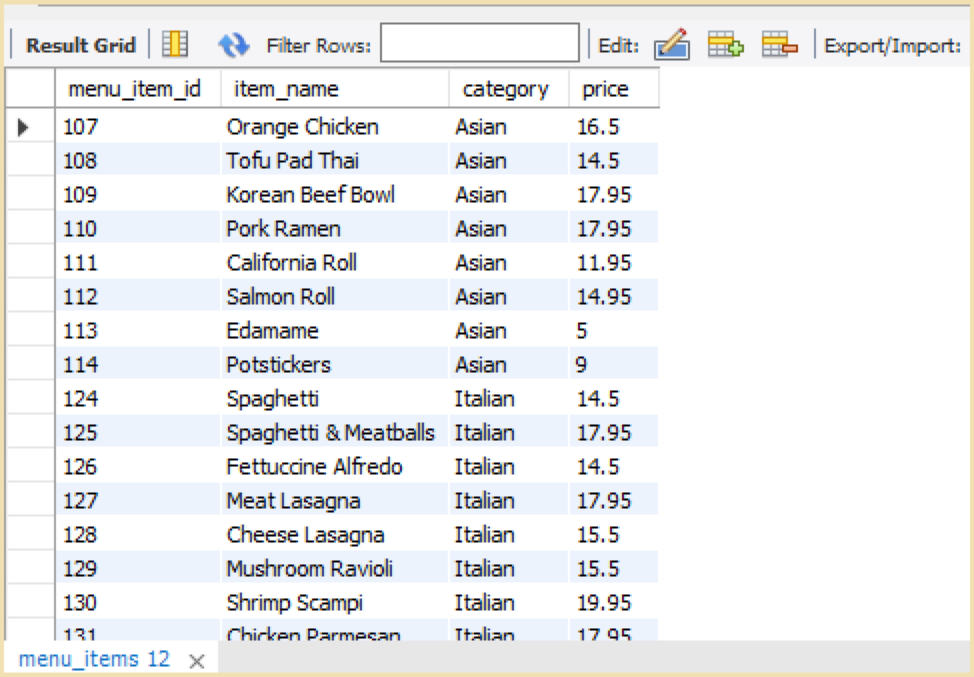
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_details_id	order_id	order_date	order_time	item_id
1	1	1/1/23	11:38:36 AM	109
2	2	1/1/23	11:57:40 AM	108
3	2	1/1/23	11:57:40 AM	124
4	2	1/1/23	11:57:40 AM	117
5	2	1/1/23	11:57:40 AM	129
6	2	1/1/23	11:57:40 AM	106
24	10	1/1/23	1:00:15 PM	129
25	10	1/1/23	1:00:15 PM	105
26	11	1/1/23	1:02:59 PM	101
27	11	1/1/23	1:02:59 PM	102
28	11	1/1/23	1:02:59 PM	102
29	11	1/1/23	1:02:59 PM	113
30	12	1/1/23	1:04:41 PM	102
31	12	1/1/23	1:04:41 PM	102
32	12	1/1/23	1:04:41 PM	104
33	12	1/1/23	1:04:41 PM	117

order_details 11

- **Using OR Clause to find the category of food Asian Or Italian :-**

```
SELECT
    *
FROM
menu_items
WHERE
    category = 'Asian'
        OR category = 'Italian';
```

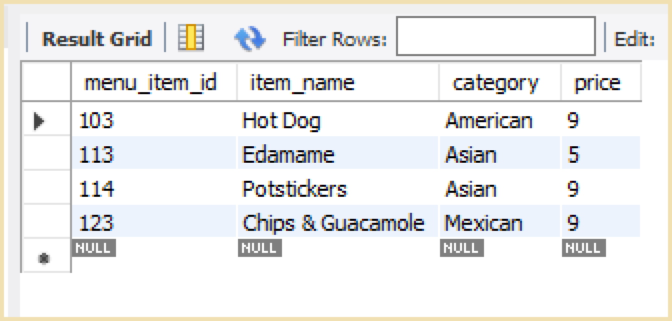

Result Grid | Filter Rows: | Edit: | Export/Import:

menu_item_id	item_name	category	price
107	Orange Chicken	Asian	16.5
108	Tofu Pad Thai	Asian	14.5
109	Korean Beef Bowl	Asian	17.95
110	Pork Ramen	Asian	17.95
111	California Roll	Asian	11.95
112	Salmon Roll	Asian	14.95
113	Edamame	Asian	5
114	Potstickers	Asian	9
124	Spaghetti	Italian	14.5
125	Spaghetti & Meatballs	Italian	17.95
126	Fettuccine Alfredo	Italian	14.5
127	Meat Lasagna	Italian	17.95
128	Cheese Lasagna	Italian	15.5
129	Mushroom Ravioli	Italian	15.5
130	Shrimp Scampi	Italian	19.95
131	Chicken Parmesan	Italian	17.95

menu_items 12

- **Using In Clause to find the price in $5 - $9 :-**

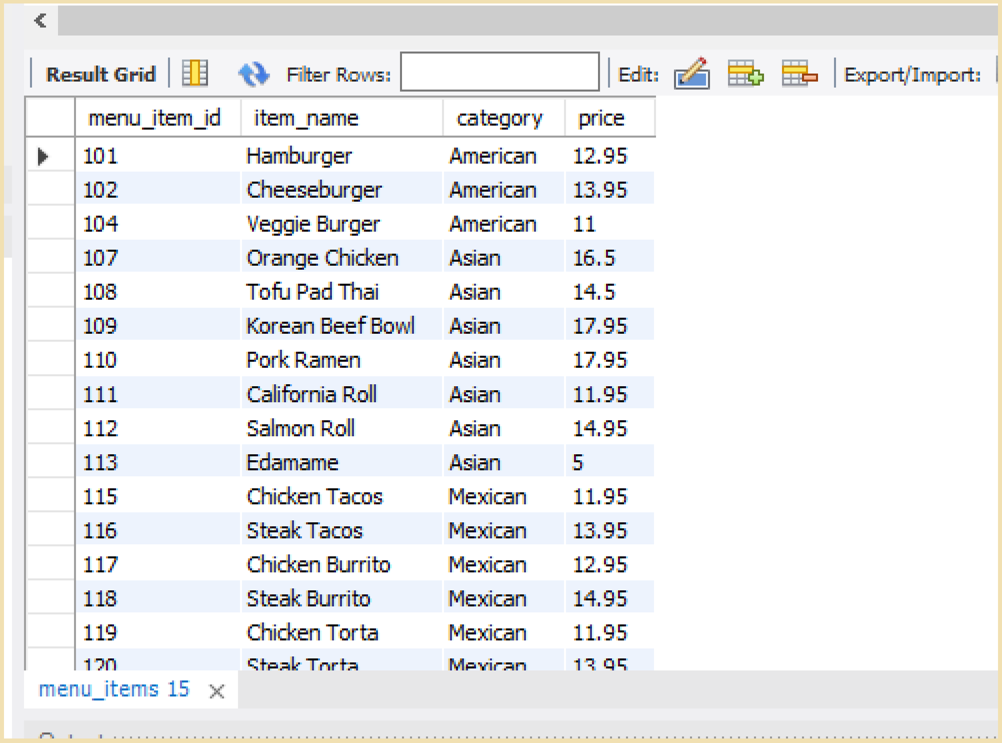
```
SELECT
  *
FROM
menu_itemsjj
WHERE
  price IN (5 , 9);
```


Result Grid | Filter Rows: | Edit:

menu_item_id	item_name	category	price
103	Hot Dog	American	9
113	Edamame	Asian	5
114	Potstickers	Asian	9
123	Chips & Guacamole	Mexican	9
NULL	NULL	NULL	NULL

- **Using Not IN Clause to find the price except $7 - $9 :-**

```
SELECT
    *
FROM
menu_items
WHERE
    price NOT IN (7 , 9);
```



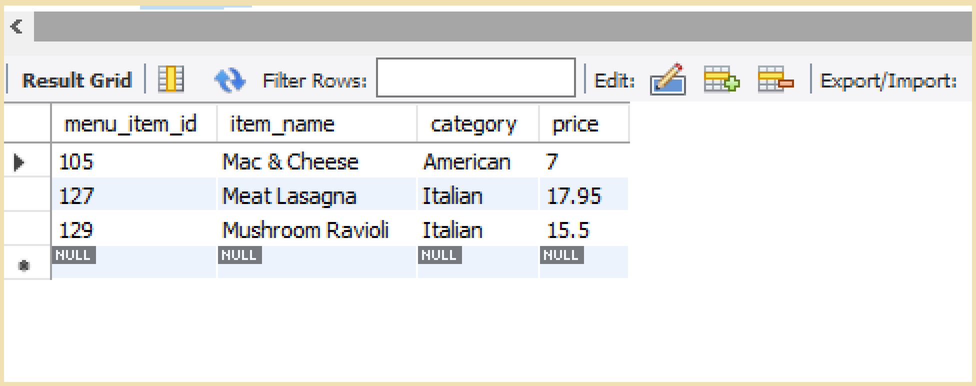
Result Grid | Filter Rows: | Edit: | Export/Import:

menu_item_id	item_name	category	price
101	Hamburger	American	12.95
102	Cheeseburger	American	13.95
104	Veggie Burger	American	11
107	Orange Chicken	Asian	16.5
108	Tofu Pad Thai	Asian	14.5
109	Korean Beef Bowl	Asian	17.95
110	Pork Ramen	Asian	17.95
111	California Roll	Asian	11.95
112	Salmon Roll	Asian	14.95
113	Edamame	Asian	5
115	Chicken Tacos	Mexican	11.95
116	Steak Tacos	Mexican	13.95
117	Chicken Burrito	Mexican	12.95
118	Steak Burrito	Mexican	14.95
119	Chicken Torta	Mexican	11.95
120	Steak Torta	Mexican	13.95

menu_items 15

- **Using Like Clause to find the item name starting form "M" :-**

```
SELECT
    *
FROM
menu_items
WHERE
item_name LIKE 'm%';
```

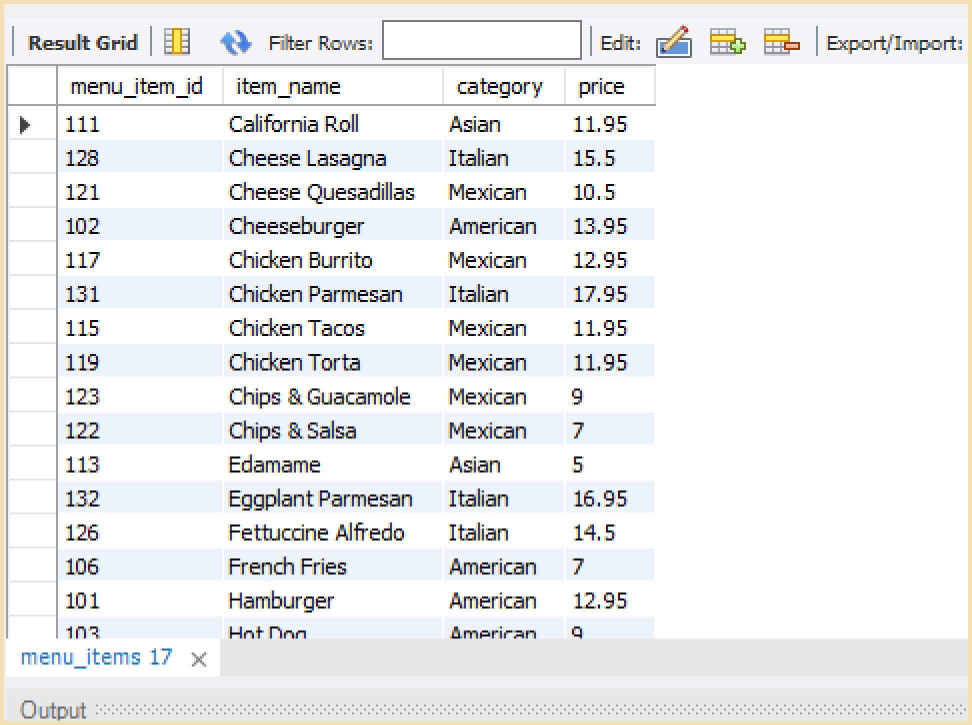


Result Grid | Filter Rows: | Edit: | Export/Import:

menu_item_id	item_name	category	price
105	Mac & Cheese	American	7
127	Meat Lasagna	Italian	17.95
129	Mushroom Ravioli	Italian	15.5
NULL	NULL	NULL	NULL

- **Using Order By Clause to get item name in ascending order :-**

```
SELECT
   *
FROM
menu_items
ORDER BY item_name;
```

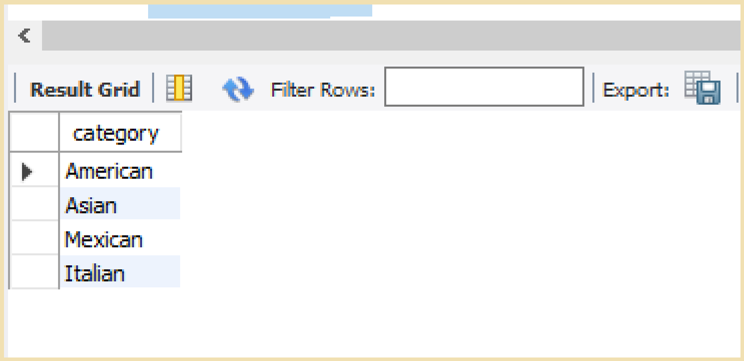


Result Grid | Filter Rows: | Edit: | Export/Import:

menu_item_id	item_name	category	price
111	California Roll	Asian	11.95
128	Cheese Lasagna	Italian	15.5
121	Cheese Quesadillas	Mexican	10.5
102	Cheeseburger	American	13.95
117	Chicken Burrito	Mexican	12.95
131	Chicken Parmesan	Italian	17.95
115	Chicken Tacos	Mexican	11.95
119	Chicken Torta	Mexican	11.95
123	Chips & Guacamole	Mexican	9
122	Chips & Salsa	Mexican	7
113	Edamame	Asian	5
132	Eggplant Parmesan	Italian	16.95
126	Fettuccine Alfredo	Italian	14.5
106	French Fries	American	7
101	Hamburger	American	12.95
103	Hot Dog	American	9

- **Using Distinct Clause to get category names :-**

```
SELECT DISTINCT
    category
FROM
menu_items;
```

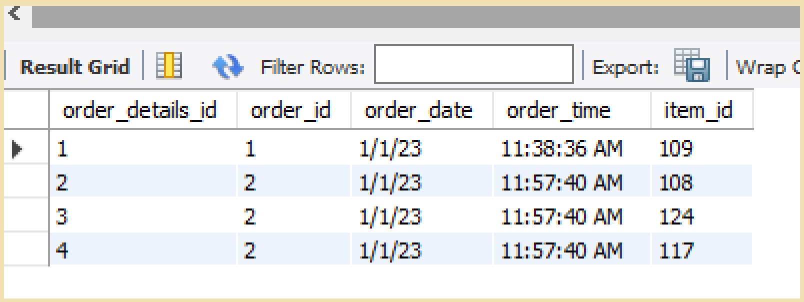


Result Grid | Filter Rows: | Export:

	category
▶	American
	Asian
	Mexican
	Italian

- **Using Limit Clause to get only 4 order details :-**

```
SELECT
    *
FROM
order_details
LIMIT 4;
```



Result Grid | Filter Rows: | Export: | Wrap C

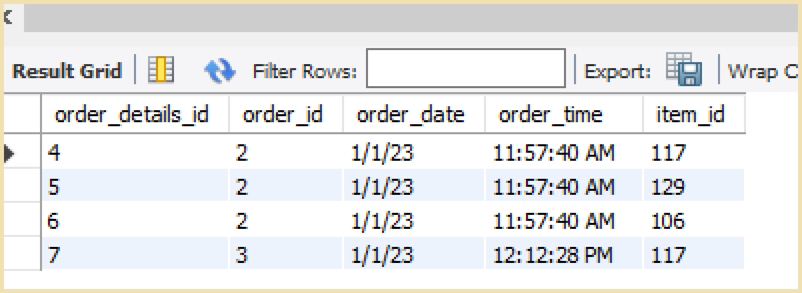
	order_details_id	order_id	order_date	order_time	item_id
▶	1	1	1/1/23	11:38:36 AM	109
	2	2	1/1/23	11:57:40 AM	108
	3	2	1/1/23	11:57:40 AM	124
	4	2	1/1/23	11:57:40 AM	117

- **Using OFFSET Clause to get to get specific row details :-**

```
SELECT
    *
FROM
```

order_details

LIMIT 3 , 4;



order_details_id	order_id	order_date	order_time	item_id
4	2	1/1/23	11:57:40 AM	117
5	2	1/1/23	11:57:40 AM	129
6	2	1/1/23	11:57:40 AM	106
7	3	1/1/23	12:12:28 PM	117

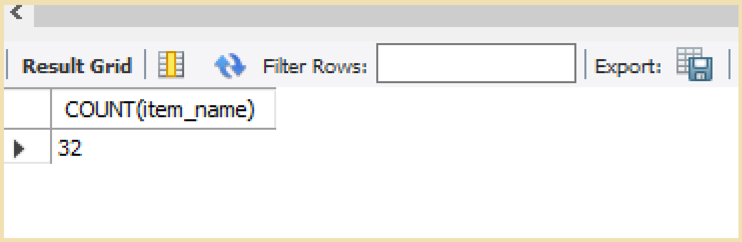
- **Using Count Clause to get total count of item name :-**

SELECT

COUNT(item_name)

FROM

menu_items;



COUNT(item_name)
32

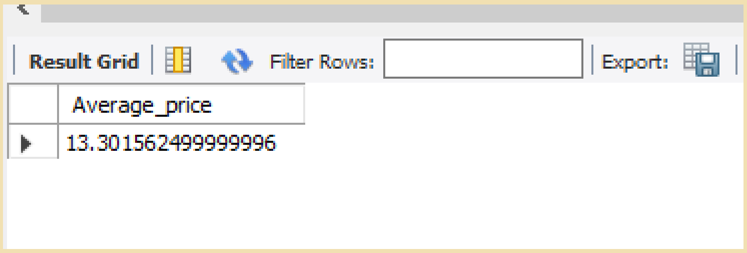
- **Using Average Clause to get average price from Menu items :-**

SELECT

AVG(price) AS Average_price

FROM

menu_items;


Result Grid | Filter Rows: | Export:

	Average_price
▶	13.301562499999996

- **Using Is Null Clause to find the null values in item_id :-**

SELECT

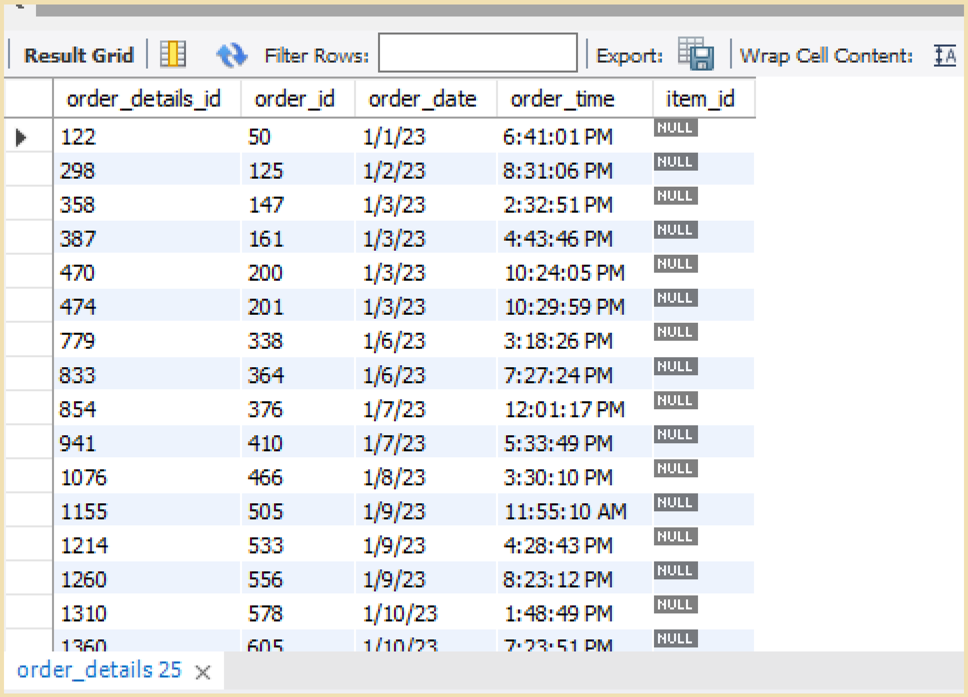
*

FROM

order_details

WHERE

item_id IS NULL;

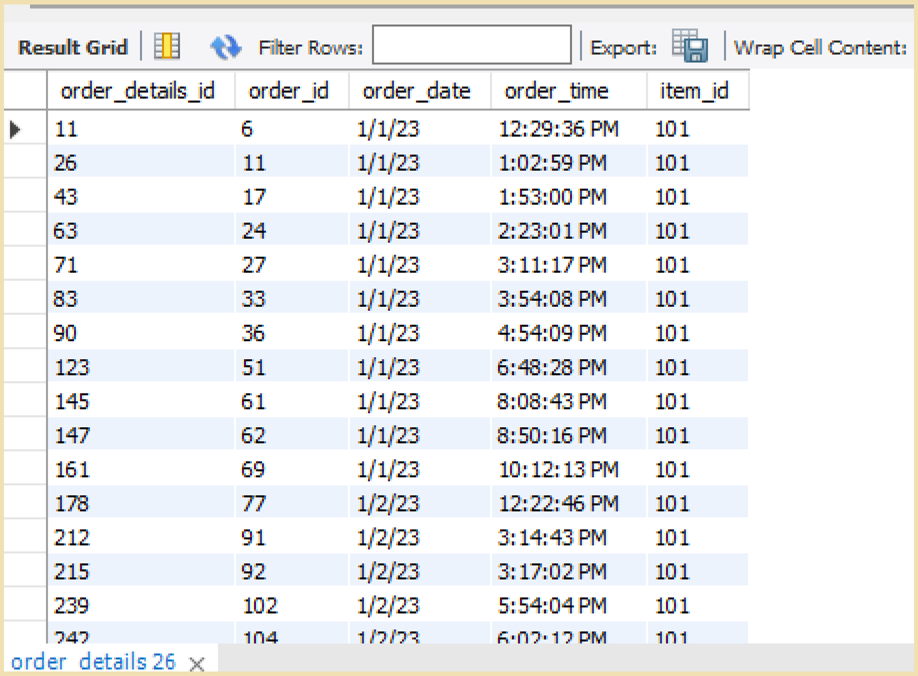

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	order_details_id	order_id	order_date	order_time	item_id
▶	122	50	1/1/23	6:41:01 PM	NULL
	298	125	1/2/23	8:31:06 PM	NULL
	358	147	1/3/23	2:32:51 PM	NULL
	387	161	1/3/23	4:43:46 PM	NULL
	470	200	1/3/23	10:24:05 PM	NULL
	474	201	1/3/23	10:29:59 PM	NULL
	779	338	1/6/23	3:18:26 PM	NULL
	833	364	1/6/23	7:27:24 PM	NULL
	854	376	1/7/23	12:01:17 PM	NULL
	941	410	1/7/23	5:33:49 PM	NULL
	1076	466	1/8/23	3:30:10 PM	NULL
	1155	505	1/9/23	11:55:10 AM	NULL
	1214	533	1/9/23	4:28:43 PM	NULL
	1260	556	1/9/23	8:23:12 PM	NULL
	1310	578	1/10/23	1:48:49 PM	NULL
	1360	605	1/10/23	7:23:51 PM	NULL

order_details 25

- **Using Not Null clause to get values from except Null values :-**

```
SELECT
    *
FROM
order_details
WHERE
item_id IS NOT NULL;
```



order_details_id	order_id	order_date	order_time	item_id
11	6	1/1/23	12:29:36 PM	101
26	11	1/1/23	1:02:59 PM	101
43	17	1/1/23	1:53:00 PM	101
63	24	1/1/23	2:23:01 PM	101
71	27	1/1/23	3:11:17 PM	101
83	33	1/1/23	3:54:08 PM	101
90	36	1/1/23	4:54:09 PM	101
123	51	1/1/23	6:48:28 PM	101
145	61	1/1/23	8:08:43 PM	101
147	62	1/1/23	8:50:16 PM	101
161	69	1/1/23	10:12:13 PM	101
178	77	1/2/23	12:22:46 PM	101
212	91	1/2/23	3:14:43 PM	101
215	92	1/2/23	3:17:02 PM	101
239	102	1/2/23	5:54:04 PM	101
242	104	1/2/23	6:02:12 PM	101

- **Using Update Clause to Update the price $11 in menu_item_id in Menu_items table :-**

```
UPDATE menu_items
SET
    price = '11'
WHERE
menu_item_id = 104;
```

- **Using Inner Join Clause to join Menu_items table and Order_details table:-**

SELECT

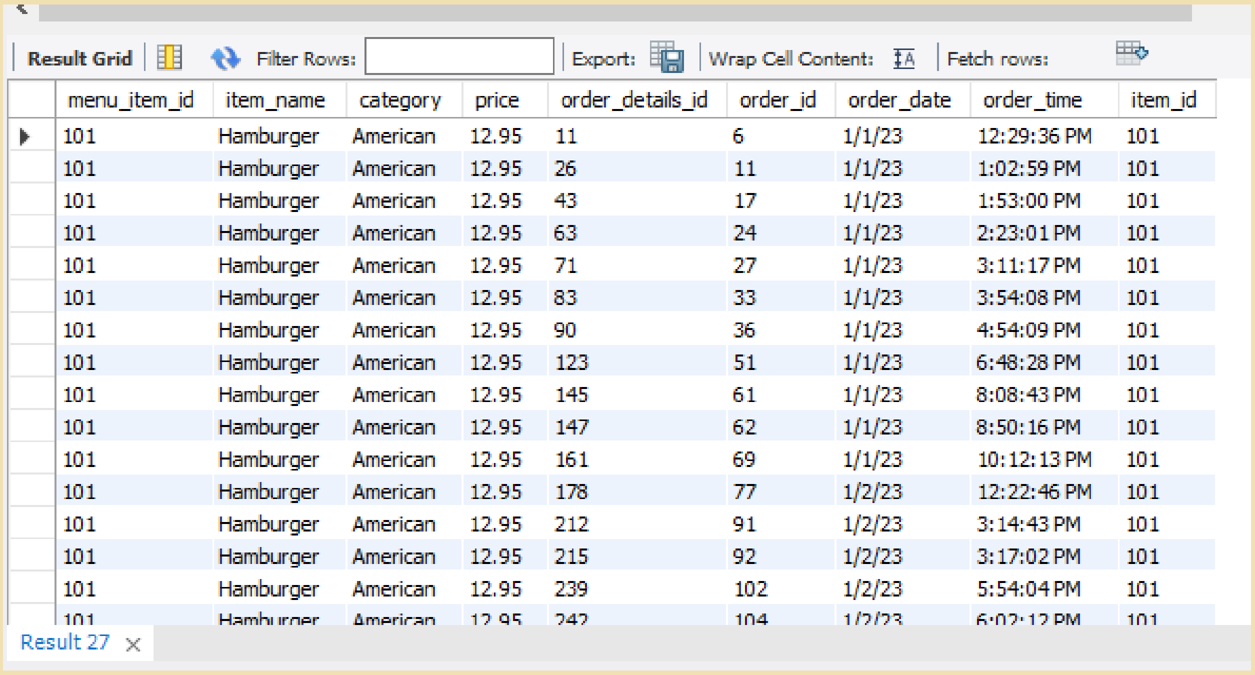
*

FROM

menu_items M

INNER JOIN

order_details O ON M.menu_item_id = O.item_id;



menu_item_id	item_name	category	price	order_details_id	order_id	order_date	order_time	item_id
101	Hamburger	American	12.95	11	6	1/1/23	12:29:36 PM	101
101	Hamburger	American	12.95	26	11	1/1/23	1:02:59 PM	101
101	Hamburger	American	12.95	43	17	1/1/23	1:53:00 PM	101
101	Hamburger	American	12.95	63	24	1/1/23	2:23:01 PM	101
101	Hamburger	American	12.95	71	27	1/1/23	3:11:17 PM	101
101	Hamburger	American	12.95	83	33	1/1/23	3:54:08 PM	101
101	Hamburger	American	12.95	90	36	1/1/23	4:54:09 PM	101
101	Hamburger	American	12.95	123	51	1/1/23	6:48:28 PM	101
101	Hamburger	American	12.95	145	61	1/1/23	8:08:43 PM	101
101	Hamburger	American	12.95	147	62	1/1/23	8:50:16 PM	101
101	Hamburger	American	12.95	161	69	1/1/23	10:12:13 PM	101
101	Hamburger	American	12.95	178	77	1/2/23	12:22:46 PM	101
101	Hamburger	American	12.95	212	91	1/2/23	3:14:43 PM	101
101	Hamburger	American	12.95	215	92	1/2/23	3:17:02 PM	101
101	Hamburger	American	12.95	239	102	1/2/23	5:54:04 PM	101
101	Hamburger	American	12.95	242	104	1/2/23	6:02:12 PM	101

- **Using Right join to join the Menu_items table and Order_details table :-**

SELECT

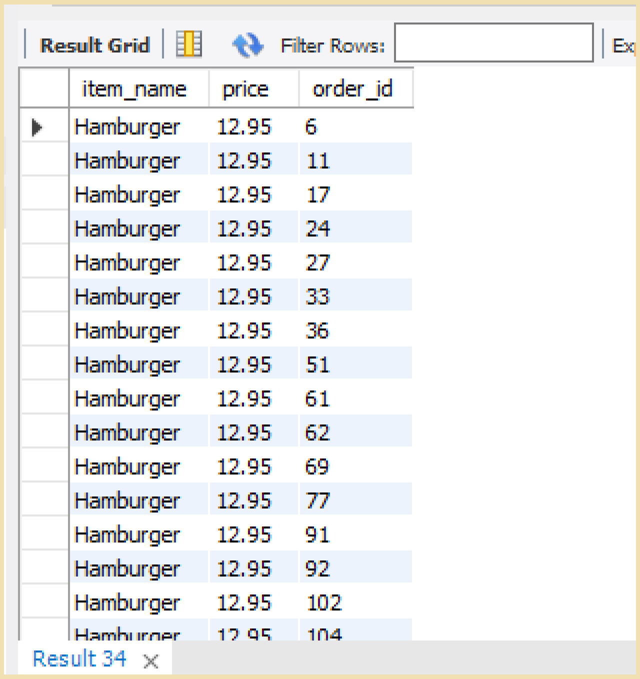
item_name,price,order_id

FROM

menu_items M

right join

order_details O ON M.menu_item_id = O.item_id;



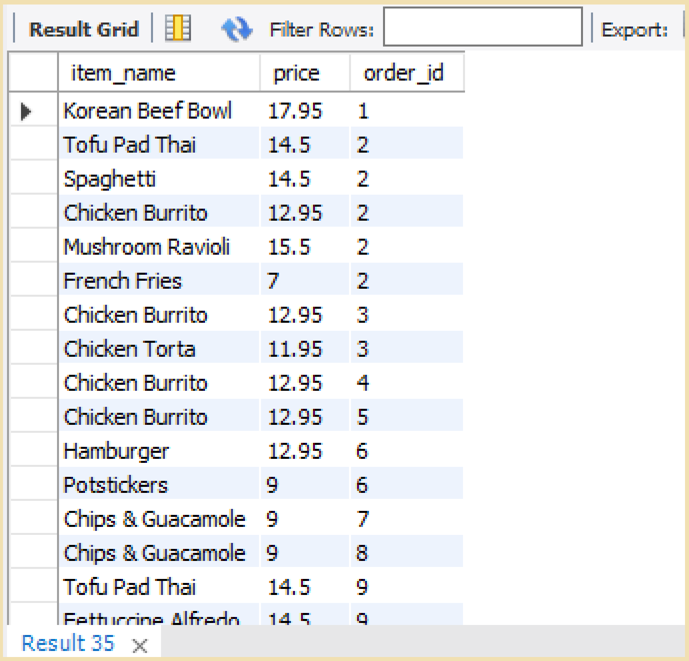
Result Grid | Filter Rows: | Ex

item_name	price	order_id
Hamburger	12.95	6
Hamburger	12.95	11
Hamburger	12.95	17
Hamburger	12.95	24
Hamburger	12.95	27
Hamburger	12.95	33
Hamburger	12.95	36
Hamburger	12.95	51
Hamburger	12.95	61
Hamburger	12.95	62
Hamburger	12.95	69
Hamburger	12.95	77
Hamburger	12.95	91
Hamburger	12.95	92
Hamburger	12.95	102
Hamburger	12.95	104

Result 34

- **Using Left join to join the Menu_items table and Order_details table :-**

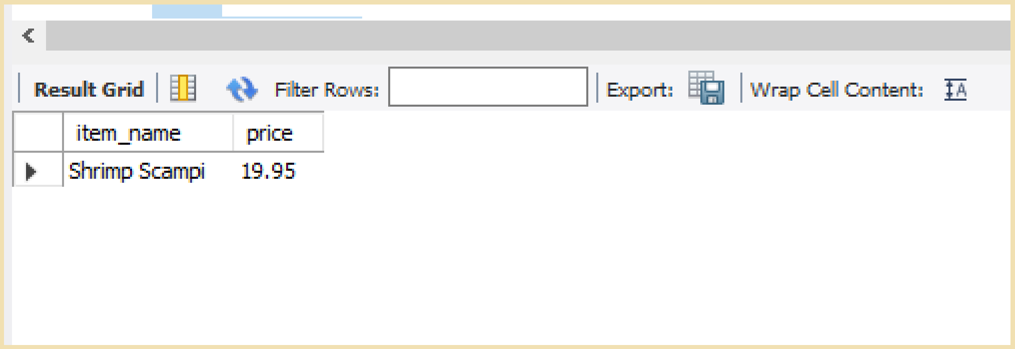
```
    SELECT
item_name,price,order_id
FROM
menu_items M
        left join
order_details O ON M.menu_item_id = O.item_id;
```



item_name	price	order_id
Korean Beef Bowl	17.95	1
Tofu Pad Thai	14.5	2
Spaghetti	14.5	2
Chicken Burrito	12.95	2
Mushroom Ravioli	15.5	2
French Fries	7	2
Chicken Burrito	12.95	3
Chicken Torta	11.95	3
Chicken Burrito	12.95	4
Chicken Burrito	12.95	5
Hamburger	12.95	6
Potstickers	9	6
Chips & Guacamole	9	7
Chips & Guacamole	9	8
Tofu Pad Thai	14.5	9
Fettuccine Alfredo	14.5	9

- **Using Sub query to find the Max price form menu_items :-**

```
SELECT
item_name, price
FROM
menu_items
WHERE
   (price) IN (SELECT
MAX(price)
      FROM
menu_items
      ORDER BY price);
```



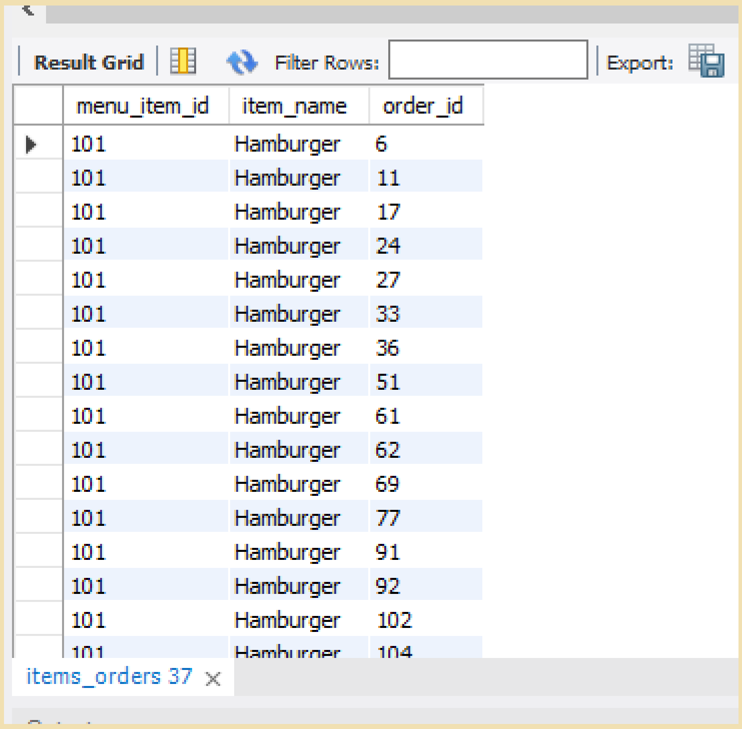
item_name	price
Shrimp Scampi	19.95

- **Using view Clause :-**

```
CREATE VIEW items_orders AS
   SELECT
menu_item_id, item_name, order_id
   FROM
menu_items
         JOIN
order_details ON menu_items.menu_item_id = order_details.item_id;
```

- **Select Command to see view :-**

```
SELECT
   *
FROM
items_orders;
```

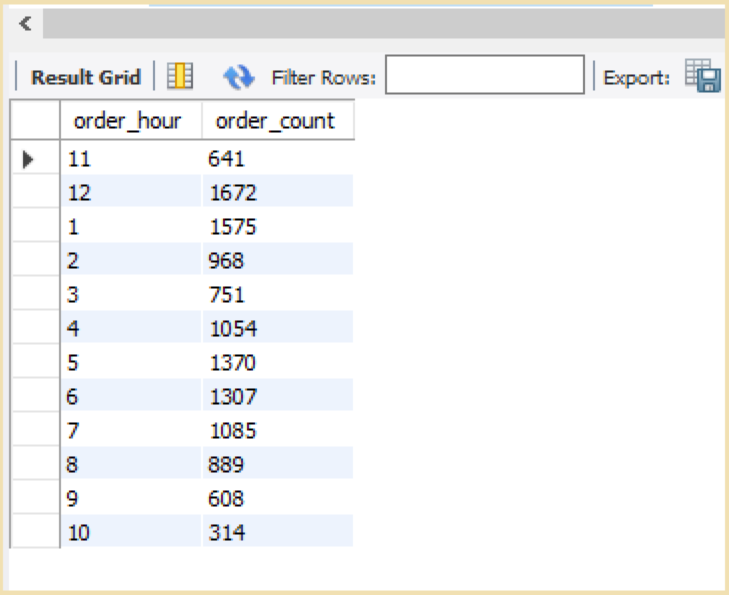


menu_item_id	item_name	order_id
101	Hamburger	6
101	Hamburger	11
101	Hamburger	17
101	Hamburger	24
101	Hamburger	27
101	Hamburger	33
101	Hamburger	36
101	Hamburger	51
101	Hamburger	61
101	Hamburger	62
101	Hamburger	69
101	Hamburger	77
101	Hamburger	91
101	Hamburger	92
101	Hamburger	102
101	Hamburger	104

items_orders 37

- **Using count clause to get order Id according t order time :-**

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



order_hour	order_count
11	641
12	1672
1	1575
2	968
3	751
4	1054
5	1370
6	1307
7	1085
8	889
9	608
10	314

- **Using Case Clause to give discount on food according price limit :-**

select category, price,

case when price >= 15 then "7% discount"

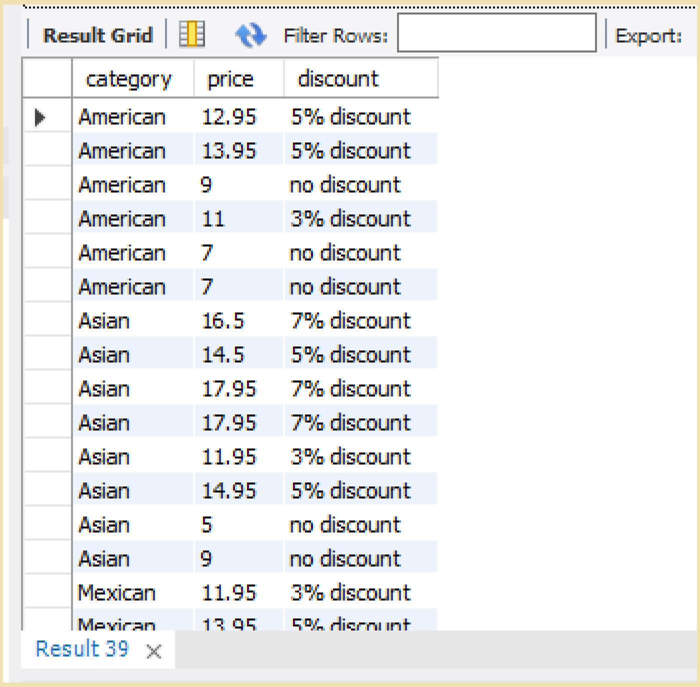
when price >=12 then "5% discount"

when price >10 and price <12 then "3% discount"

when price <10 then "no discount"

end as discount

from menu_items;

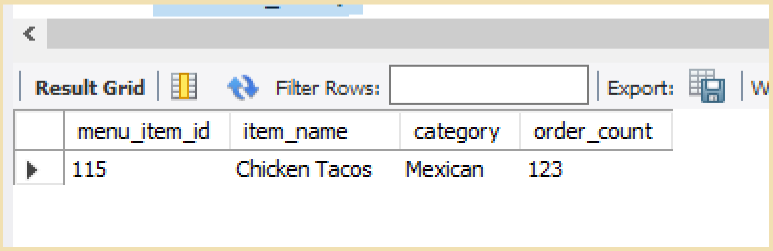


category	price	discount
American	12.95	5% discount
American	13.95	5% discount
American	9	no discount
American	11	3% discount
American	7	no discount
American	7	no discount
Asian	16.5	7% discount
Asian	14.5	5% discount
Asian	17.95	7% discount
Asian	17.95	7% discount
Asian	11.95	3% discount
Asian	14.95	5% discount
Asian	5	no discount
Asian	9	no discount
Mexican	11.95	3% discount
Mexican	13.95	5% discount

Questions:-

1) What were the least ordered items? What categories were they in?

```
SELECT
menu_item_id,
item_name,
   category,
COUNT(item_id) AS order_count
FROM
order_details od
      JOIN
menu_items m ON item_id = menu_item_id
GROUP BY menu_item_id ,item_name , category
ORDER BY order_count ASC
LIMIT 1;
```



menu_item_id	item_name	category	order_count
115	Chicken Tacos	Mexican	123

2) **What were the most ordered items? What categories were they in?**

SELECT

menu_item_id,

item_name,

category,

COUNT(item_id) AS order_count

FROM

order_details od

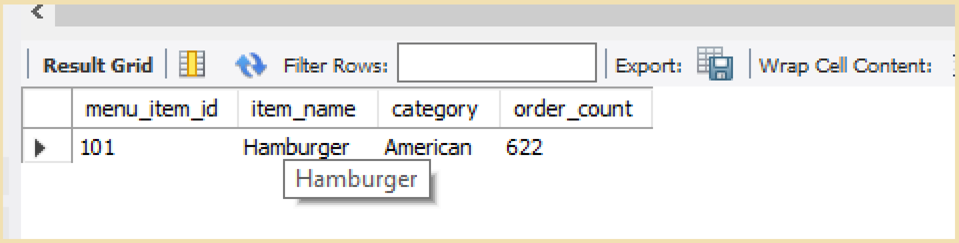
JOIN

menu_items m ON item_id = menu_item_id

GROUP BY menu_item_id ,item_name , category

ORDER BY order_count DESC

LIMIT 1;

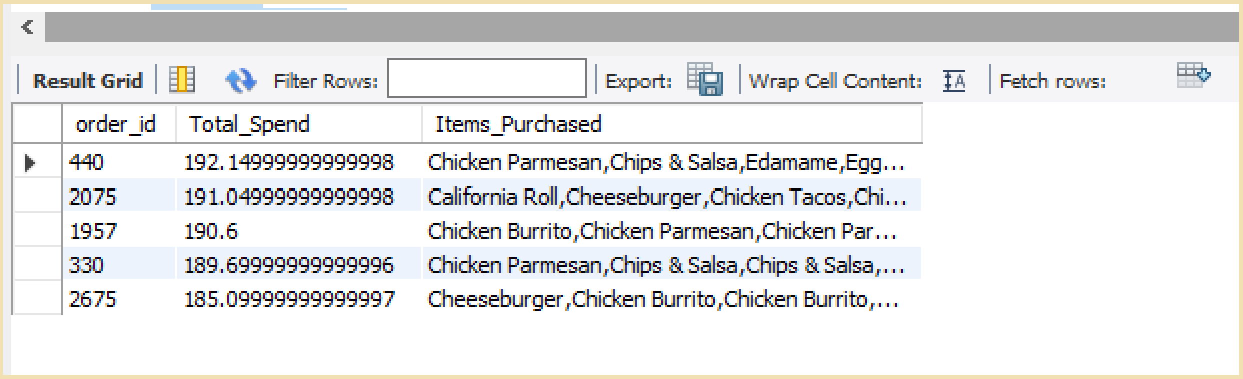


menu_item_id	item_name	category	order_count
101	Hamburger	American	622

3) **What do the highest spend orders look like? Which items did they buy and how much did they spend?**

SELECT

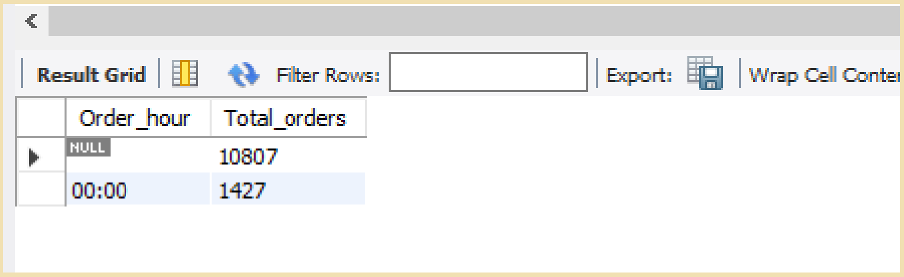
```
order_id,
SUM(price) AS Total_Spend,
   GROUP_CONCAT(item_name
      ORDER BY item_name ASC) AS Items_Purchased
FROM
order_details
      JOIN
menu_items ON item_id = menu_item_id
GROUP BY order_id
ORDER BY Total_Spend DESC
LIMIT 5;
```



```
order_id  Total_Spend         Items_Purchased
440       192.14999999999998  Chicken Parmesan,Chips & Salsa,Edamame,Egg...
2075      191.04999999999998  California Roll,Cheeseburger,Chicken Tacos,Chi...
1957      190.6               Chicken Burrito,Chicken Parmesan,Chicken Par...
330       189.69999999999996  Chicken Parmesan,Chips & Salsa,Chips & Salsa,...
2675      185.09999999999997  Cheeseburger,Chicken Burrito,Chicken Burrito,...
```

4) Were there certain times that had more or less orders?

```
SELECT
   DATE_FORMAT(order_time, '%H:00') AS Order_hour,
COUNT(*) AS Total_orders
FROM
order_details
GROUP BY Order_hour
ORDER BY order_hour;
```



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

	Order_hour	Total_orders
▶	NULL	10807
	00:00	1427

5) Which cuisines should we focus on developing more menu items for based on the data?

1. Total Orders by Category :-

SELECT

category, COUNT(item_id) AS Total_orders

FROM

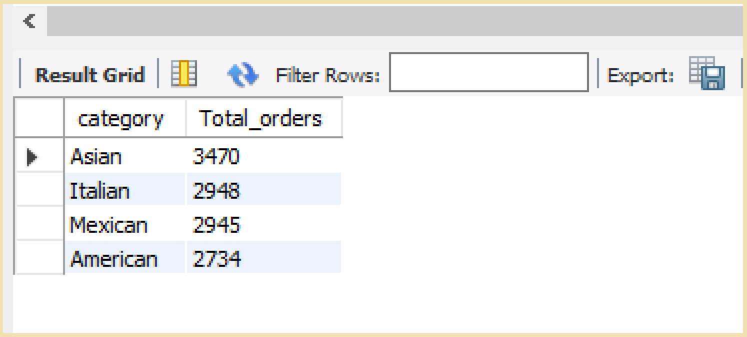
Order_details

JOIN

Menu_items ON item_id = menu_item_id

GROUP BY category

ORDER BY Total_orders DESC;



Result Grid | Filter Rows: | Export:

	category	Total_orders
▶	Asian	3470
	Italian	2948
	Mexican	2945
	American	2734

2. Revenue By Category :-

SELECT

category, SUM(price * item_id) AS Total_revenue

FROM

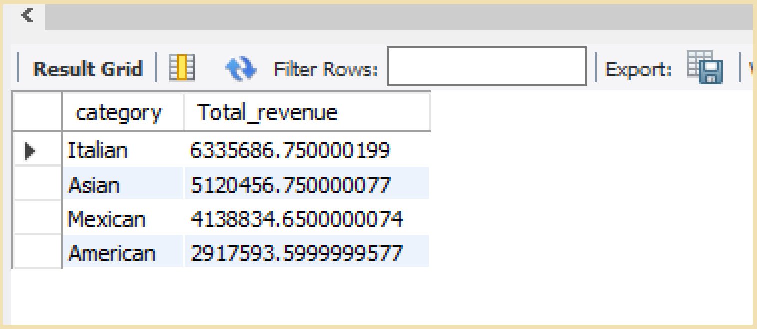
Order_details

JOIN

menu_items ON item_id = menu_item_id

GROUP BY category

ORDER BY Total_revenue DESC;



category	Total_revenue
Italian	6335686.750000199
Asian	5120456.750000077
Mexican	4138834.6500000074
American	2917593.5999999577

3. Time-Based Analysis :-

SELECT

category,

HOUR(order_time) AS order_hour,

COUNT(item_id) AS Total_orders

FROM

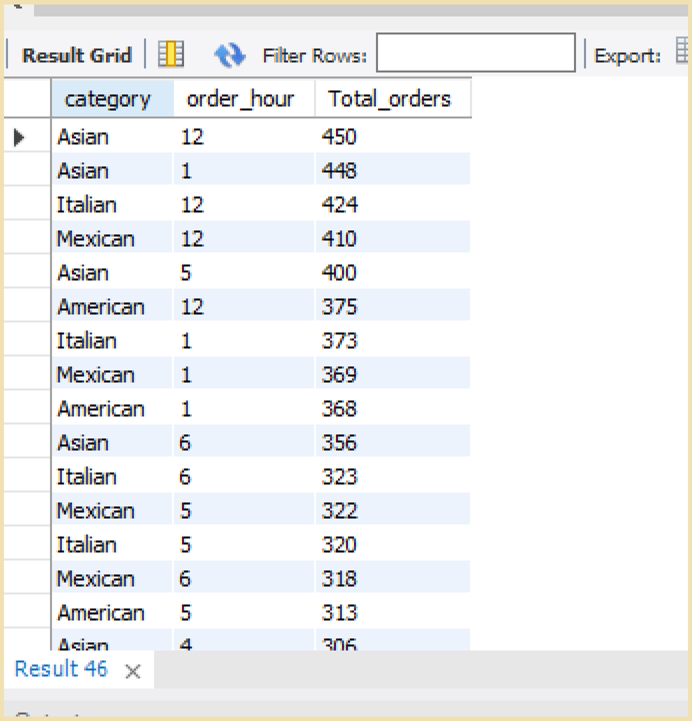
order_details

JOIN

menu_items ON item_id = menu_item_id

GROUP BY category ,order_hour

ORDER BY Total_orders DESC;



Result Grid | Filter Rows: | Export:

category	order_hour	Total_orders
Asian	12	450
Asian	1	448
Italian	12	424
Mexican	12	410
Asian	5	400
American	12	375
Italian	1	373
Mexican	1	369
American	1	368
Asian	6	356
Italian	6	323
Mexican	5	322
Italian	5	320
Mexican	6	318
American	5	313
Asian	4	306

Result 46