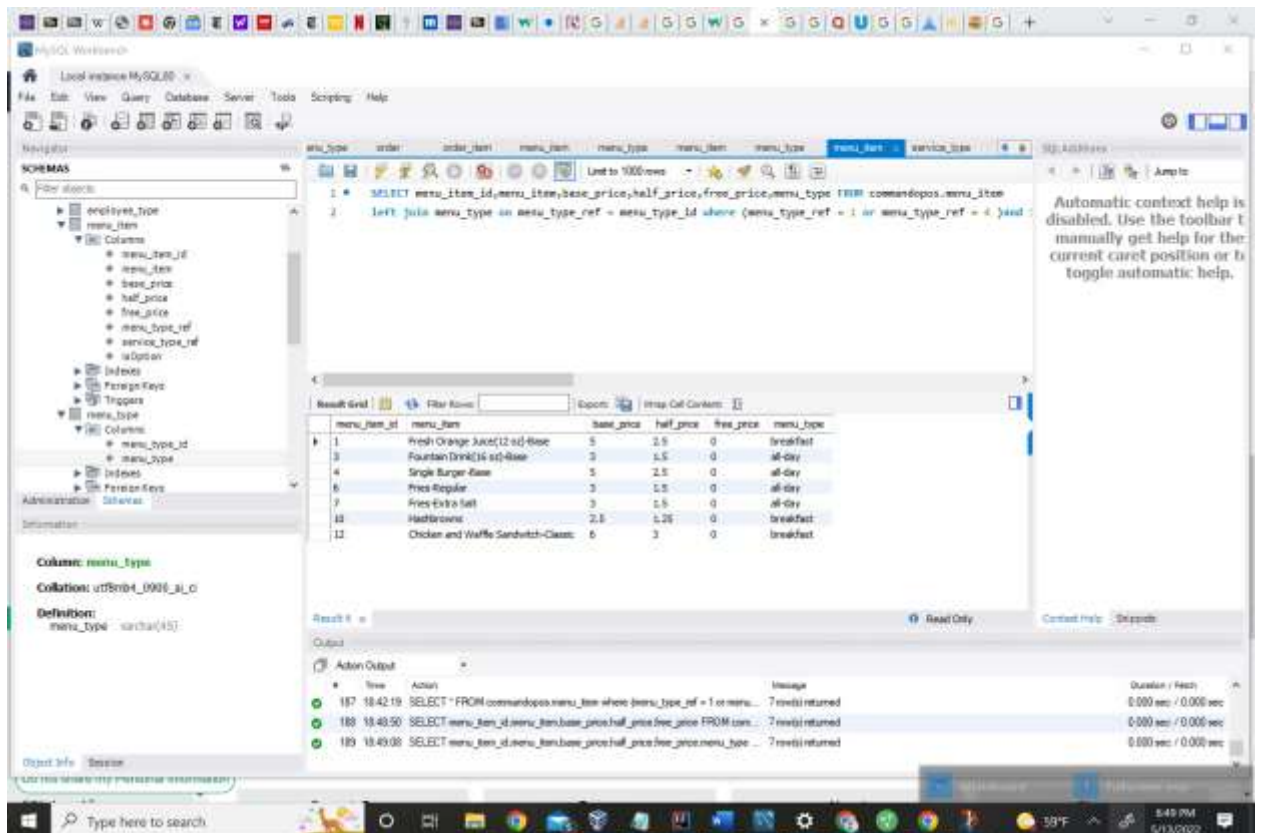


```
SELECT menu_item_id,menu_item,base_price,half_price,free_price,menu_type FROM
commandopos.menu_item
left join menu_type on menu_type_ref = menu_type_id where (menu_type_ref = 1 or
menu_type_ref = 4 )and isOption = 0;
```



Query Items served for Lunch

SELECT menu\_item\_id,menu\_item,base\_price,half\_price,free\_price,menu\_type FROM commandopos.menu\_item

left join menu\_type on menu\_type\_ref = menu\_type\_id where (menu\_type\_ref = 2 or menu\_type\_ref = 4) and isOption = 0;

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL:

```

1.  menu_id,menu_item,base_price,half_price,free_price,menu_type FROM commandopos.menu_item
2.  left join menu_type on menu_type_ref = menu_type_id where (menu_type_ref = 3 or menu_type_ref = 4)and isOption = 0;

```

The result set displays the following data:

menu_item_id	menu_item	base_price	half_price	free_price	menu_type
5	Fountain Drink(16 oz)Base	3	1.5	0	all-day
4	Single Burger-Base	5	2.5	0	all-day
6	Price-Rugular	2	1.5	0	all-day
8	Strawberry Lemonade(16 oz)	4	2	0	lunch

The Action Output pane shows the execution of three queries:

- 193 18:53:17 SELECT menu\_item\_id,menu\_item,base\_price,half\_price,free\_price,menu\_type ... 5 rows(s) returned
- 194 18:53:44 SELECT - FROM commandopos.menu\_item LIMIT 0, 1000 13 rows(s) returned
- 195 18:54:19 SELECT menu\_item\_id,menu\_item,base\_price,half\_price,free\_price,menu\_type ... 4 rows(s) returned

Query Items served for Dinner

SELECT menu\_item\_id,menu\_item,base\_price,half\_price,free\_price,menu\_type FROM commandopos.menu\_item

left join menu\_type on menu\_type\_ref = menu\_type\_id where (menu\_type\_ref = 3 or menu\_type\_ref = 4)and isOption = 0;

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigation

SCHEMAS

Filter schemas

- commandos
- commandos.menu\_item
  - menu\_item\_id
  - menu\_item
  - base\_price
  - half\_price
  - free\_price
  - menu\_type\_ref
  - service\_type\_ref
  - is\_option
- commandos.menu\_type
  - menu\_type\_id
  - menu\_type

Administration Schemas

Information

Column: menu\_type

Collation: utf8mb4\_0900\_ai\_ci

Definition: menu\_type varchar(45)

SQL Editor

1 SELECT menu\_item\_id, menu\_item, base\_price, half\_price, free\_price, menu\_type FROM commandos.menu\_item

2 LEFT JOIN menu\_type ON menu\_type\_ref = menu\_item\_id WHERE (menu\_type\_ref = 3 OR menu\_type\_ref = 4) AND

Result Grid

menu_item_id	menu_item	base_price	half_price	free_price	menu_type
5	Fountain Drink (16 oz) Base	2	1.5	0	all-day
4	Single Burger Base	1	2.5	0	all-day
6	Free Regular	1	1.5	0	all-day
8	Barometer Dressed Beer - 18 oz	8	2	0	drink

Result 10

Output

Action Output

Time	Action	Message	Duration / Pct
194 18:53:44	SELECT * FROM commandos.menu_item LIMIT 0, 1000	13 rows returned	0.000 sec / 0.000 sec
195 18:54:19	SELECT menu_item_id, menu_item, base_price, half_price, free_price, menu_type	4 rows returned	0.000 sec / 0.000 sec
196 18:55:42	SELECT menu_item_id, menu_item, base_price, half_price, free_price, menu_type	4 rows returned	0.000 sec / 0.000 sec

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

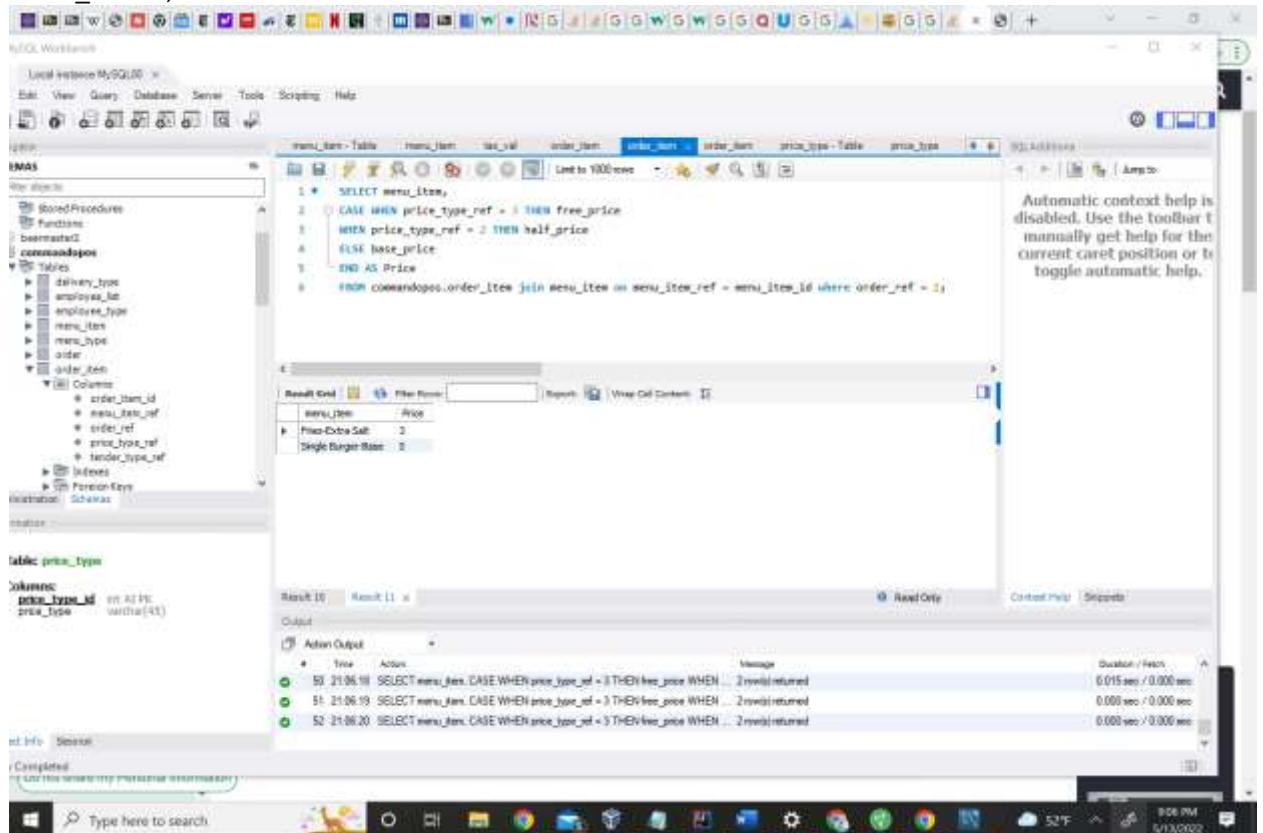
Type here to search

5:55 PM 6/13/2022

Make sure we can see each order in its entirety (items, price, options) –

FYI options are just another menu Item

```
SELECT menu_item,  
CASE WHEN price_type_ref = 3 THEN free_price  
WHEN price_type_ref = 2 THEN half_price  
ELSE base_price  
END AS Price  
FROM commandpos.order_item join menu_item on menu_item_ref = menu_item_id where  
order_ref = 2;
```



Make sure we can separate the items in an order by service type (grill, sides, drinks, etc)

```
SELECT menu_item,service_type FROM commandpos.order_item join menu_item on  
menu_item_ref = menu_item_id join service_type on service_type_id = service_type_ref where  
order_ref = 1;
```

The screenshot shows the HeidiSQL interface with a SQL query executed. The query is: `SELECT menu_item,service_type FROM commandpos.order_item join menu_item on menu_item_ref = menu_item_id join service_type on service_type_id = service_type_ref where order_ref = 1;`

The results are displayed in a table with two columns: `menu_item` and `service_type`.

menu_item	service_type
Chicken and Buffalo Sandwich-Classic	grill
Hashbrowns	flair
Fresh Orange Juice(12 oz)-base	beverages(wm-alc-hld)

The bottom panel shows the execution log with three rows of results, each taking 0.000 seconds to execute.

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.



Make sure we have at least 3 prices for each item (regular, halfoff, free)

SELECT menu\_item, base\_price,half\_price,free\_price FROM commandopos.menu\_item;

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays the 'commandopos' database with a list of tables including 'menu\_item'. The 'Columns' pane for 'menu\_item' shows columns: 'menu\_item\_id', 'menu\_item', 'base\_price', 'half\_price', 'free\_price', 'menu\_type\_ref', 'service\_type\_ref', and 'isOption'. The main query editor contains the SQL statement: `SELECT menu_item, base_price,half_price,free_price FROM commandopos.menu_item;`. The 'Result Grid' displays 13 rows of data. The 'Output' pane at the bottom shows the execution log with three successful queries.

menu_item	base_price	half_price	free_price
Fresh Orange Juice(12 oz)-Base	8	2.5	0
Fresh Orange Juice(12 oz)-Strawled	8	2.5	0
Fountain Drink(16 oz)-Base	3	1.3	0
Single Burger-Base	3	2.5	0
Single Burger-with Cheese	5.5	2.75	0
Pizza-Regular	3	1.5	0
Pizza-Extra-Small	3	1.5	0
Beer-master Brewed Beer-16 oz	6	3	0
Strawberry Lemonade(16 oz)	4	2	0
Hashbrowns	2.5	1.25	0
Single Burger-Fix Sun	5	2.5	0
Chicken and Waffle Sandwich-Chi...	6	3	0
Chicken and Waffle Sandwich-Chi...	6.5	3.25	0

Output Log:

Time	Action	Message	Duration / Rows
198 19:02:01	SELECT * FROM commandopos.tbl LIMIT 0, 1000	3 rows returned	0.016 sec / 0.000 sec
199 19:04:47	SELECT * FROM commandopos.menu_item LIMIT 0, 1000	13 rows returned	0.000 sec / 0.000 sec
200 19:09:23	SELECT menu_item, base_price,half_price,free_price FROM commandopos.me...	13 rows returned	0.000 sec / 0.000 sec

Make sure we can view all tickets by server

SELECT order\_id,date,time,employee\_first\_name,employee\_last\_name FROM  
commandpos.order left join employee\_list on employee\_list\_id = server\_ref where server\_ref  
= employee\_list\_id and (delivery\_type\_ref = 1 or delivery\_type\_ref = 2);

The screenshot shows the SQL Developer interface with a query executed against a local MySQL database. The query is: `SELECT order_id,date,time,employee_first_name,employee_last_name FROM commandpos.order left join employee_list on employee_list_id = server_ref where server_ref = employee_list_id and (delivery_type_ref = 1 or delivery_type_ref = 2);`

The result set displays 10 rows of data:

order_id	date	time	employee_first_name	employee_last_name
1	2022-12-31	09:39:00	5	O'Phalen
2	2022-12-31	09:15:00	Doug	Deacon
4	2022-12-31	11:38:45	Pepper	Jack
5	2022-12-31	11:55:27	Doug	Deacon
7	2022-12-31	23:01:00	8	O'Phalen
8	2022-12-31	23:02:00	Doug	Deacon
9	2022-12-31	23:04:00	Pepper	Jack
10	2022-12-31	23:05:00	Pepper	Jack

The bottom pane shows the execution log with three entries:

Time	Action	Message	Duration / Fetch
14 19:52:04	SELECT order_id,date,time,employee_first_name,employee_last_name FROM c...	8 rows(s) returned	0.000 sec / 0.000 sec
15 19:52:46	SELECT * FROM commandpos.order LIMIT 0, 1000	10 rows(s) returned	0.000 sec / 0.000 sec
16 19:53:08	SELECT order_id,date,time,employee_first_name,employee_last_name FROM c...	8 rows(s) returned	0.016 sec / 0.000 sec



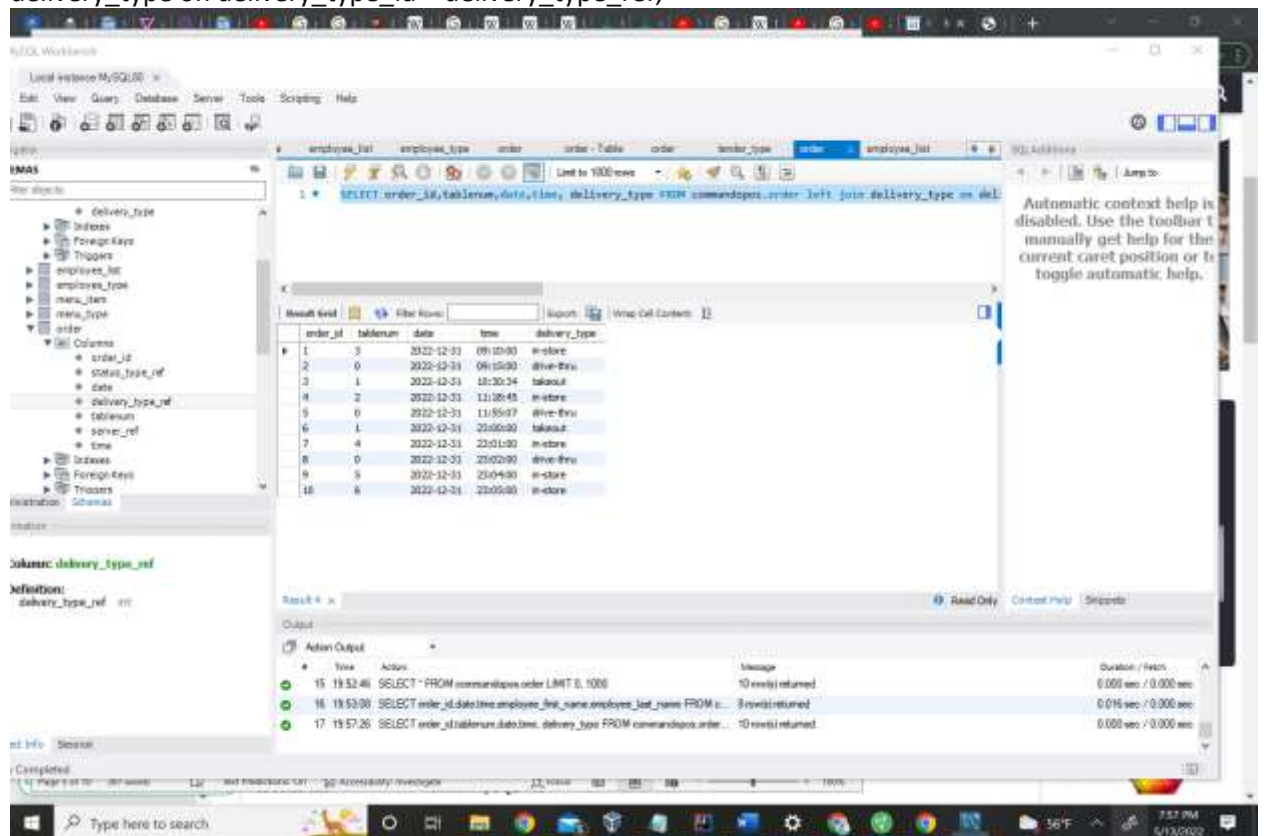
Make sure we can list all tickets by table noting which orders are in the kitchen

SELECT order\_id,tablename,date,time, status\_type FROM commandpos.order left join status\_type on status\_type\_id = status\_type\_ref;

The screenshot shows the MySQL Workbench interface. On the left, the 'Schemas' pane displays the 'commandpos' database with tables: delivery\_type, employees\_list, employees\_type, menu\_item, menu\_type, order, order\_id, status\_type\_ref, tablename, time, delivery\_type\_ref, and some\_ref. The 'Columns' pane for the 'order' table is expanded, showing columns: order\_id, status\_type\_ref, date, delivery\_type\_ref, tablename, and some\_ref. The 'Tables' pane shows the 'order' table selected. The 'Query' pane contains the SQL query: `SELECT order_id,tablename,date,time, status_type FROM commandpos.order left join status_type on status_type_id = status_type_ref;`. The 'Result Grid' shows 10 rows of data. The 'Output' pane shows the execution of the query, with a message: '10 rows returned'. The 'Status' pane shows the query execution status: 'Completed'.

order_id	tablename	date	time	status_type
1	3	2022-12-31	08:00:00	delivered
2	0	2022-12-31	08:25:00	delivered
3	1	2022-12-31	08:30:34	delivered
4	3	2022-12-31	11:38:45	delivered
5	0	2022-12-31	11:50:07	in queue
6	1	2022-12-31	23:00:30	in queue
7	4	2022-12-31	23:01:30	in kitchen
8	0	2022-12-31	23:02:00	in kitchen
9	3	2022-12-31	23:04:30	in kitchen
10	6	2022-12-31	23:05:30	in kitchen

Make sure we can organize tickets by delivery type (sit down, take out, drive thru)  
 SELECT order\_id,tablename,date,time, delivery\_type FROM commandpos.order left join  
 delivery\_type on delivery\_type\_id = delivery\_type\_ref;



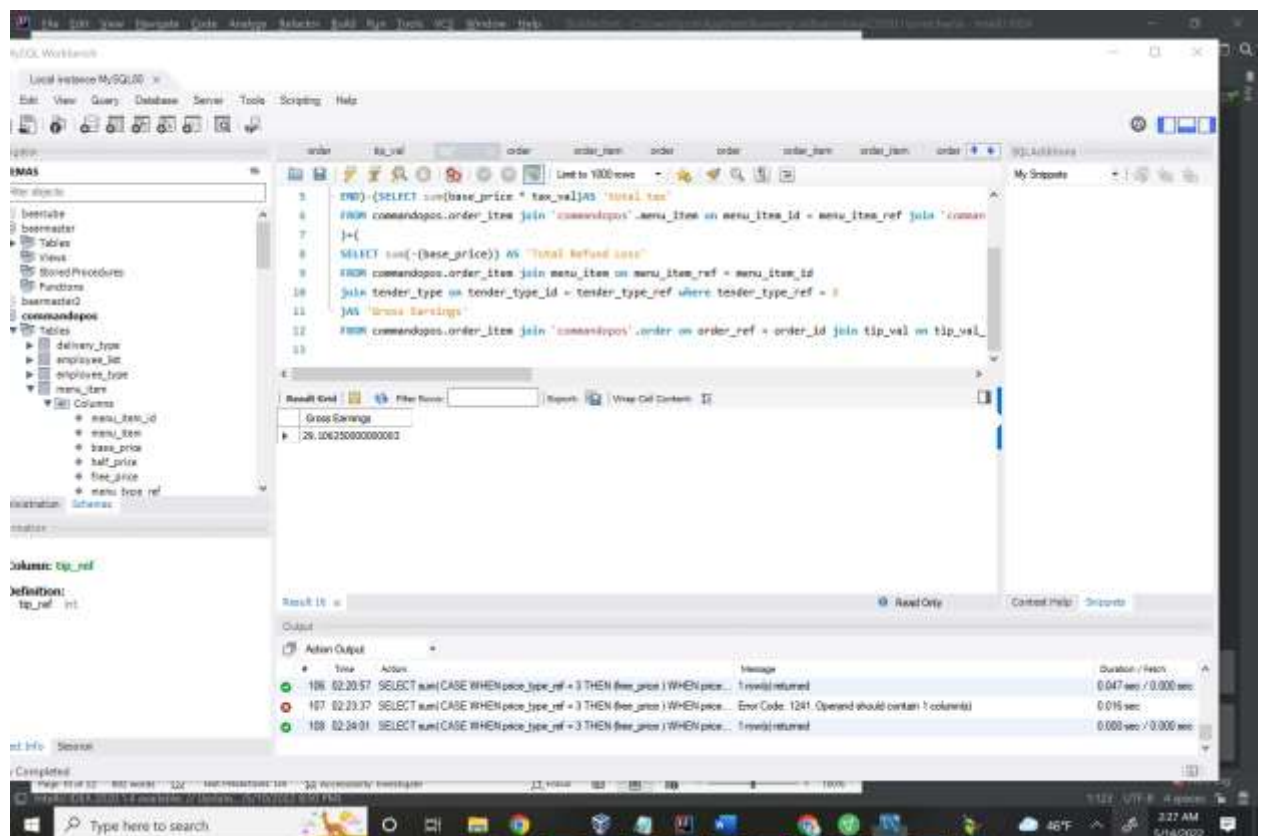
Gross Earnings(Net Earnings from sales(full and discounted)+refund loss-taxes)Gratuities aren't counted

**SELECT sum(  
 CASE WHEN price\_type\_ref = 3 THEN (free\_price )  
 WHEN price\_type\_ref = 2 THEN (half\_price)  
 ELSE (base\_price)  
 END)-(SELECT sum(base\_price \* tax\_val)AS 'total tax'**

```

FROM commandpos.order_item join `commandpos`.menu_item on menu_item_id =
menu_item_ref join `commandpos`.order on order_ref = order_id join tax_val on tax_val_id
= tax_ref where date = '2022-12-31'
)+(
SELECT sum(-(base_price)) AS 'Total Refund Loss'
FROM commandpos.order_item join menu_item on menu_item_ref = menu_item_id
join tender_type on tender_type_id = tender_type_ref where tender_type_ref = 3
)AS 'Gross Earnings'
FROM commandpos.order_item join `commandpos`.order on order_ref = order_id join
tip_val on tip_val_id = tip_ref join menu_item on menu_item_id = menu_item_ref where date
= '2022-12-31';

```

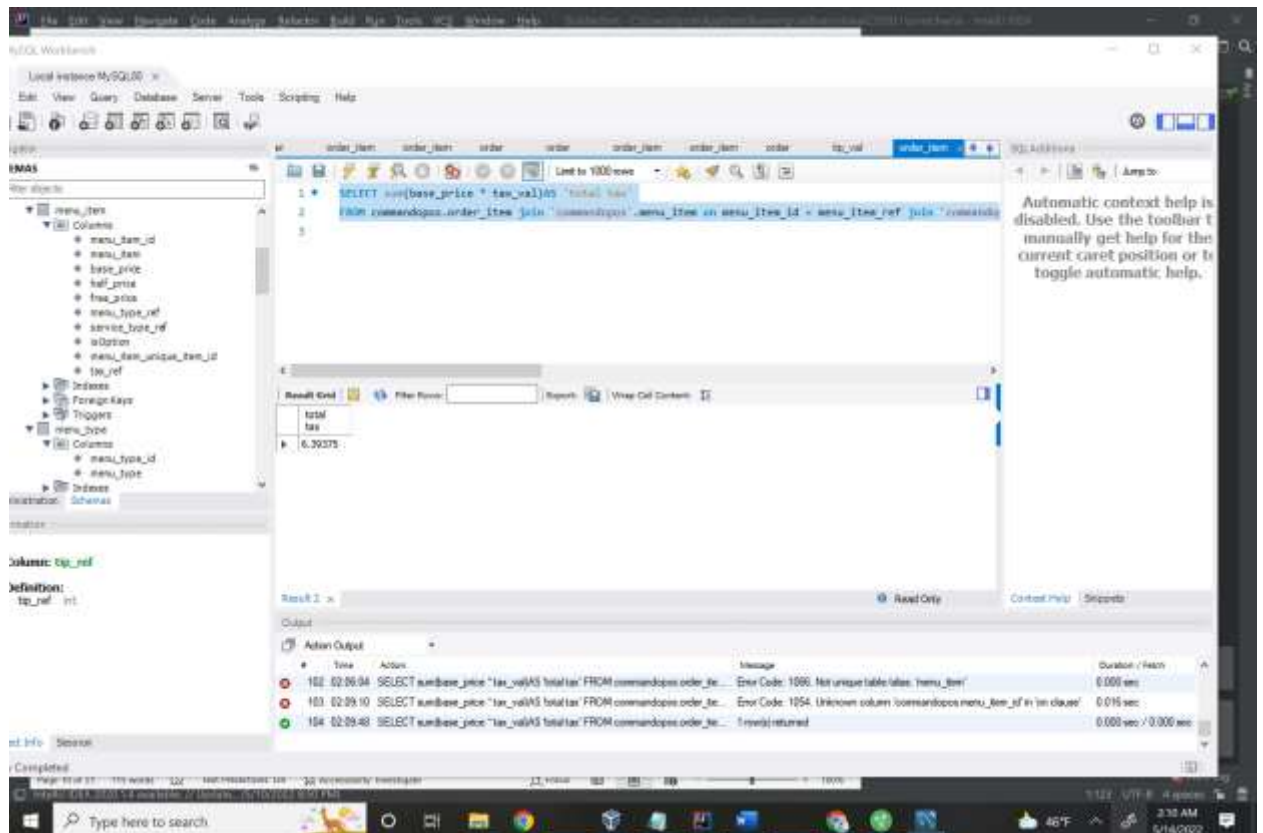


### Total Tax

```

SELECT sum(base_price * tax_val)AS 'total tax'
FROM commandpos.order_item join `commandpos`.menu_item on menu_item_id =
menu_item_ref join `commandpos`.order on order_ref = order_id join tax_val on tax_val_id
= tax_ref where date = '2022-12-31';

```

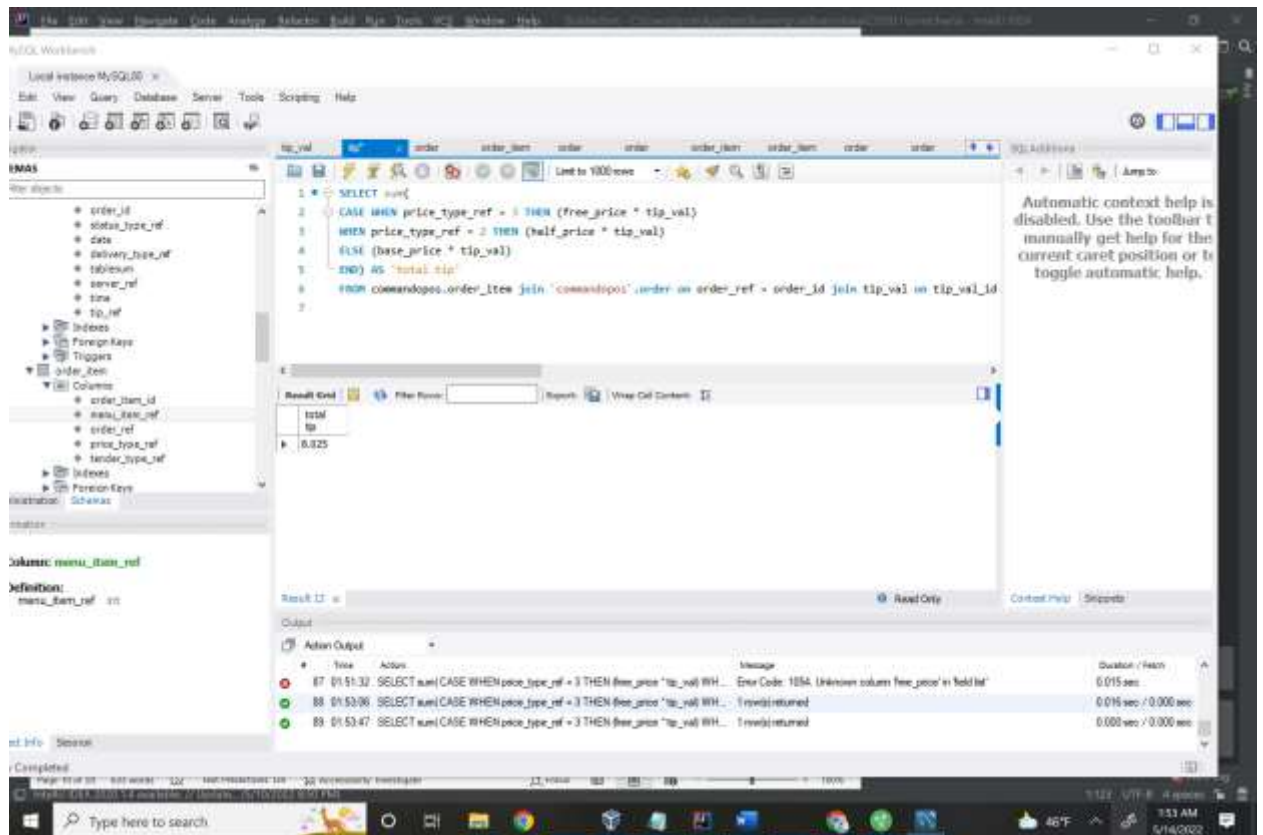


## Total Gratuity

```

SELECT sum(
CASE WHEN price_type_ref = 3 THEN (free_price * tip_val)
WHEN price_type_ref = 2 THEN (half_price * tip_val)
ELSE (base_price * tip_val)
END) AS 'total tip'
FROM commandopos.order_item join `commandopos`.order on order_ref = order_id join
tip_val on tip_val_id = tip_ref join menu_item on menu_item_id = menu_item_ref where date
= '2022-12-31';

```

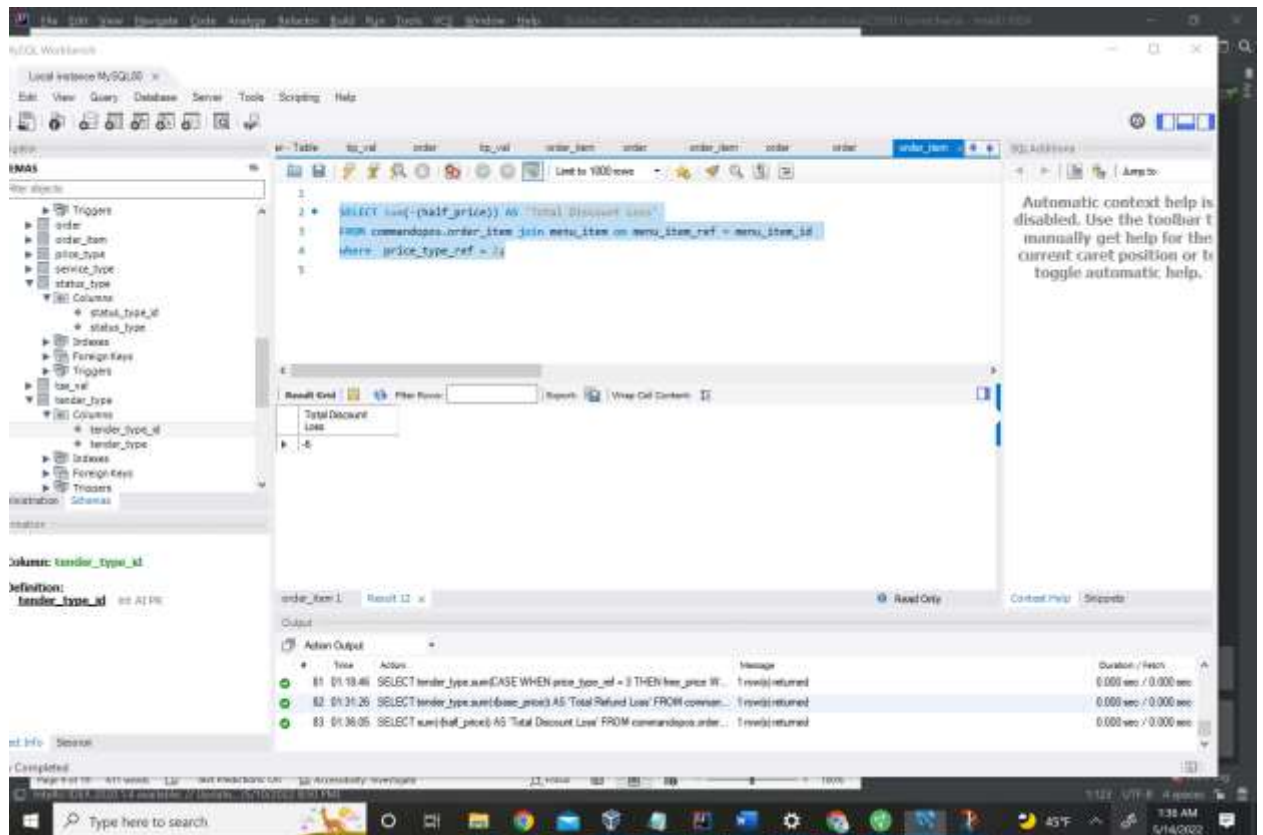


Total discount

SELECT sum(-(half\_price)) AS 'Total Discount Loss'

FROM commandopos.order\_item join menu\_item on menu\_item\_ref = menu\_item\_id

where price\_type\_ref = 2;



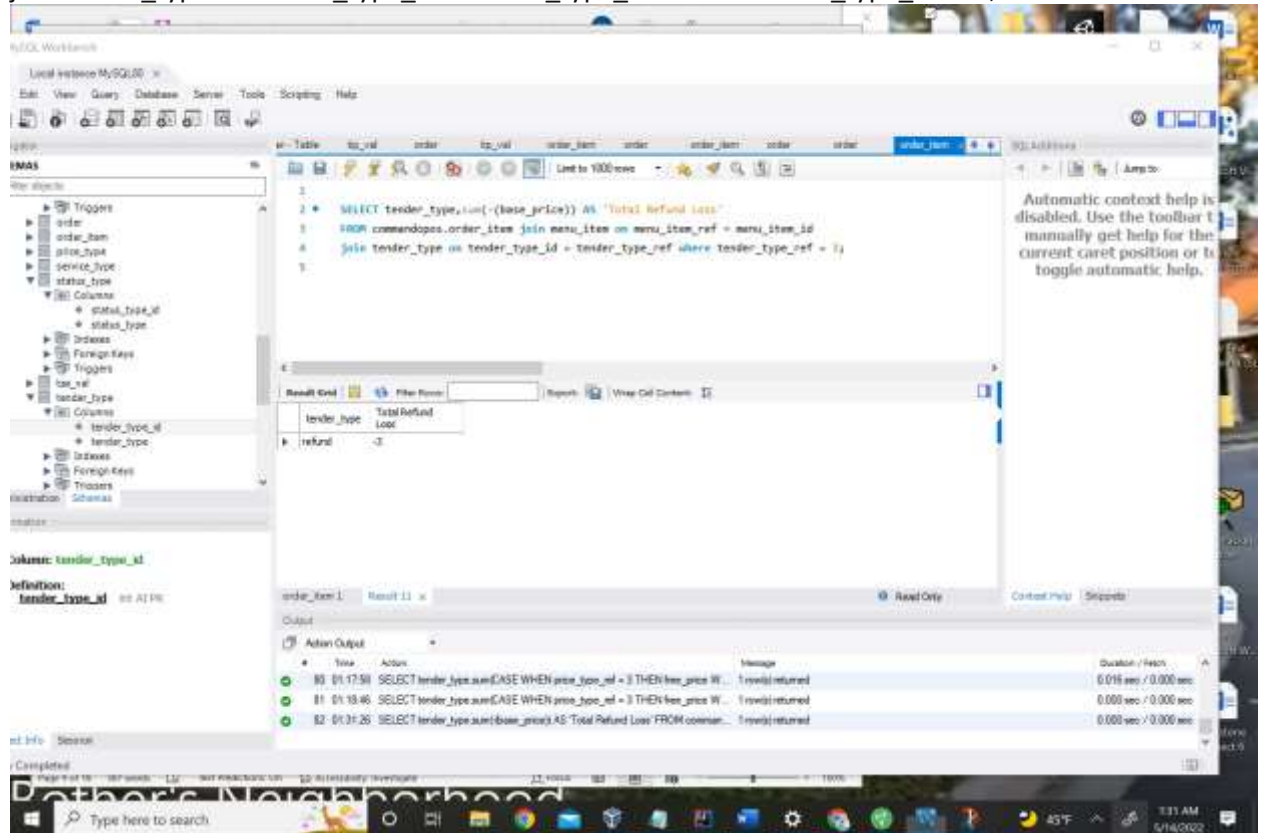
Total Refund

SELECT tender\_type,sum(-(base\_price)) AS 'Total Refund Loss'

FROM commandops.order\_item join menu\_item on menu\_item\_ref = menu\_item\_id



join tender\_type on tender\_type\_id = tender\_type\_ref where tender\_type\_ref = 3;



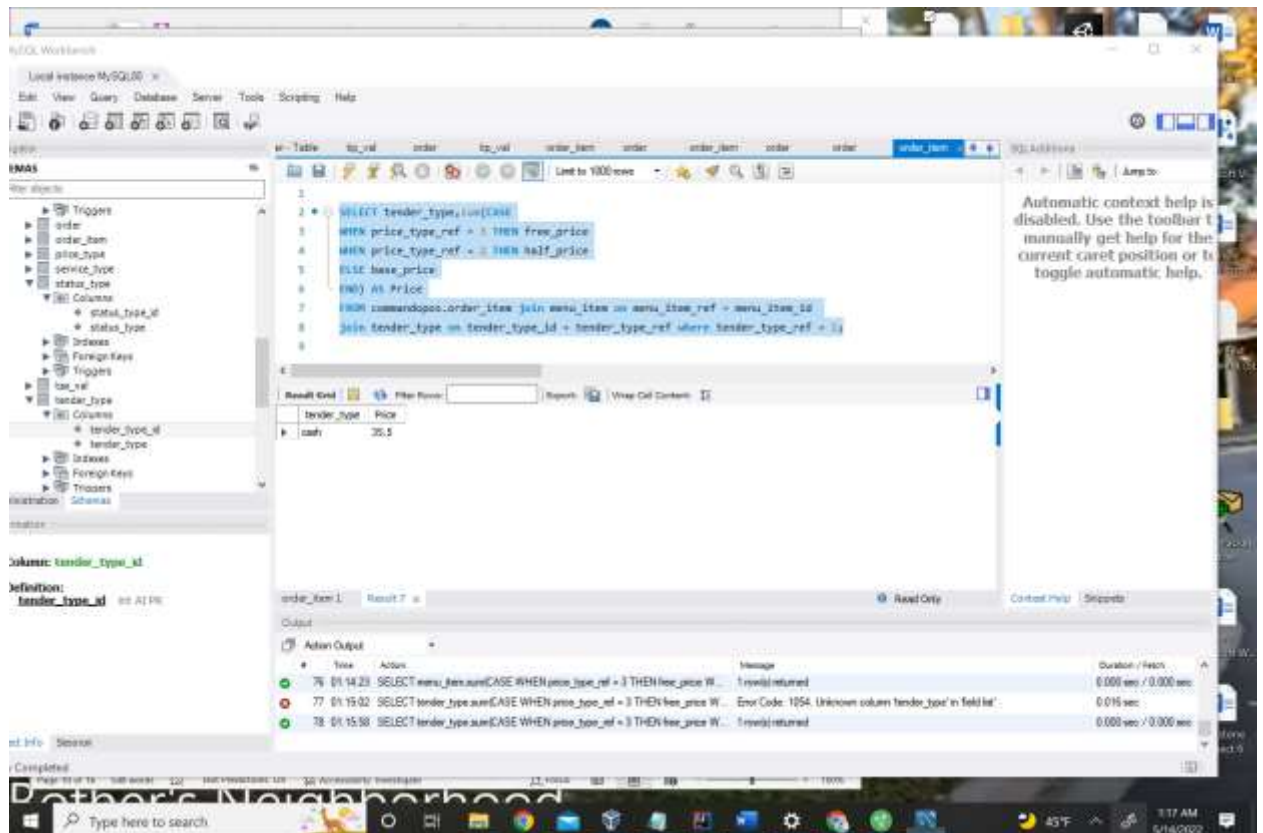
Make sure we can generate a listing of total sales by tender type per day

### Cash Query

```

SELECT tender_type,sum(CASE
WHEN price_type_ref = 3 THEN free_price
WHEN price_type_ref = 2 THEN half_price
ELSE base_price
END) AS Price
FROM commandpos.order_item join menu_item on menu_item_ref = menu_item_id
join tender_type on tender_type_id = tender_type_ref where tender_type_ref = 1;

```



Debit Query

SELECT tender\_type,sum(CASE

WHEN price\_type\_ref = 3 THEN free\_price

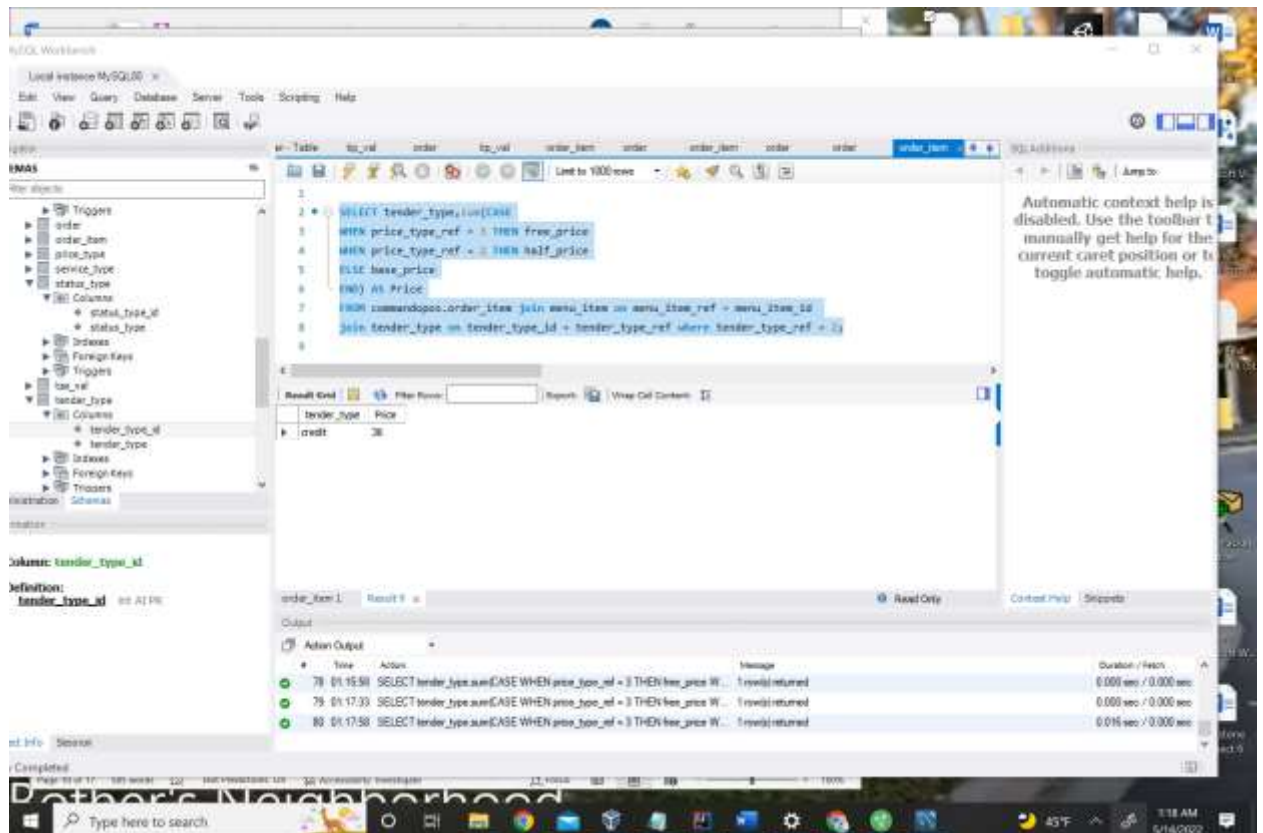
WHEN price\_type\_ref = 2 THEN half\_price

ELSE base\_price

END) AS Price

FROM commandopos.order\_item join menu\_item on menu\_item\_ref = menu\_item\_id

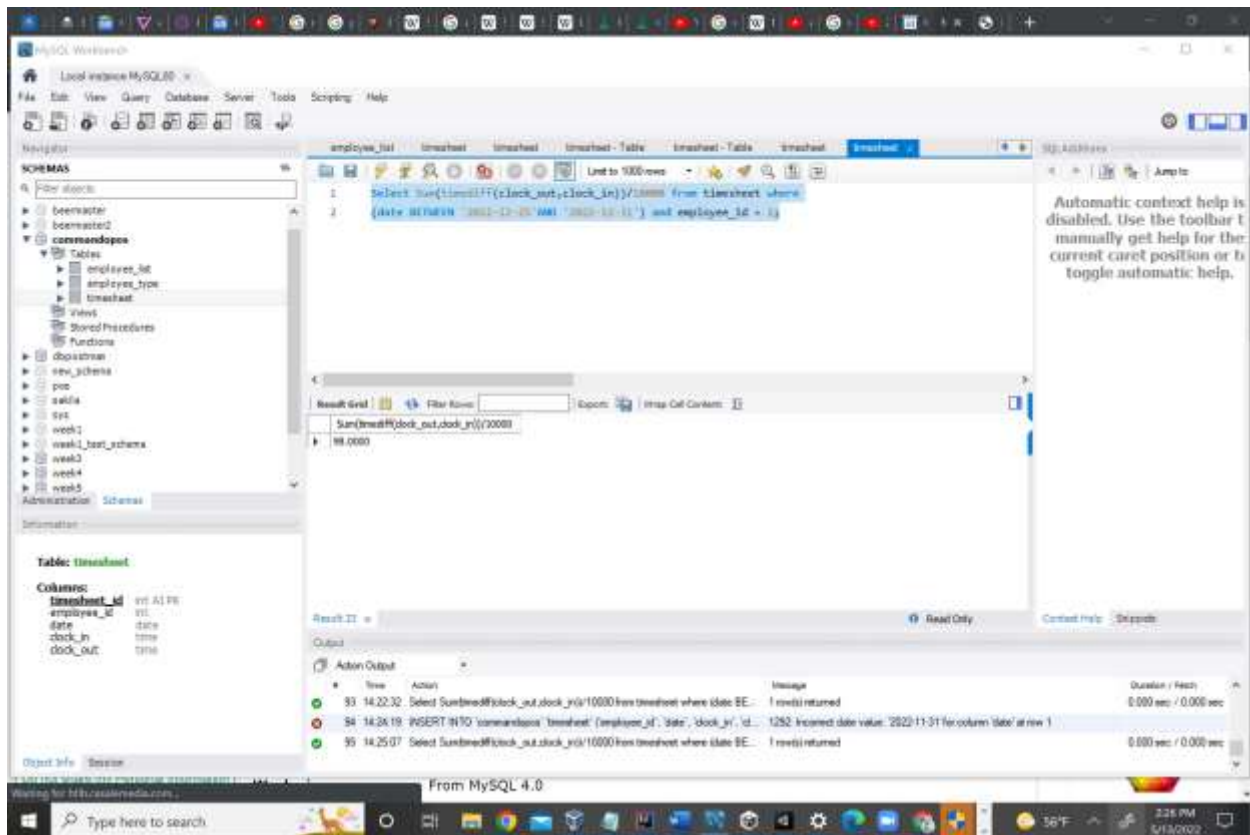
join tender\_type on tender\_type\_id = tender\_type\_ref where tender\_type\_ref = 2;



Make sure we can identify all the options for any menu item

SELECT menu\_item,base\_price,half\_price,free\_price,menu\_item\_id FROM  
commandopos.menu\_item where menu\_item\_unique\_item\_id = 3;





Make sure we can list all employees by role

```
SELECT employee_list_id, employee_first_name, employee_last_name, employee_type
FROM commandopos.employee_list
inner join employee_type on
employee_role_id = employee_type_id;
```

MySQL Workbench

Local instance MySQL80 - x

File Edit View Query Database Server Tools Scripting Help

Navigation

SCHEMAS

- Filter schemas
- beermaster
- beermaster2
- commandopos
  - Tables
    - employees\_list
      - Columns
        - employee\_list\_id
        - employee\_first\_name
        - employee\_last\_name
        - employee\_role\_id
      - Indexes
      - Foreign Keys
      - Triggers
    - employees\_type
      - Columns
        - employee\_type\_id
        - employee\_type
      - Indexes
      - Foreign Keys
      - Triggers
- Administration
  - Schemas
- Information

Column: employee\_type

Collation: utf8mb4\_0900\_ai\_ci

Definition: employee\_type varchar(45)

SQL Editor

```
1 SELECT employee_list_id, employee_first_name, employee_last_name, employee_type
2 INNER JOIN employees_type ON
3 employee_role_id = employee_type_id;
```

Result Grid

employee_list_id	employee_first_name	employee_last_name	employee_type
1	Alexis	Beer	Manager
2	Al	O'Riordan	Server
3	Pepper	Jack	Server
4	Hugh	Joe	Barista
5	Thomas	Twinkl	Barista
6	Donald	Dimadome	Host
7	Justin	Kane	Head Chef
8	Will	Power	Busser
9	Nick	Waller	Dishwasher

Result 1

Read Only

Context Help

Drop

Output

Admin Output

#	Time	Action	Message	Duration / Pacing
89	14:34:24	SELECT - FROM commandopos.employees_list join employees_type on employee...	3 rows returned	0.000 sec / 0.000 sec
100	14:35:16	SELECT - FROM commandopos.employees_list inner join employees_type on emp...	3 rows returned	0.000 sec / 0.000 sec
101	14:36:14	SELECT employee_list_id, employee_first_name, employee_last_name, employee...	3 rows returned	0.000 sec / 0.000 sec

Output Info

Page 2 of 2

Type here to search

2:36 PM 5/13/2022

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.



Make sure we have 2 calculated values for taxes

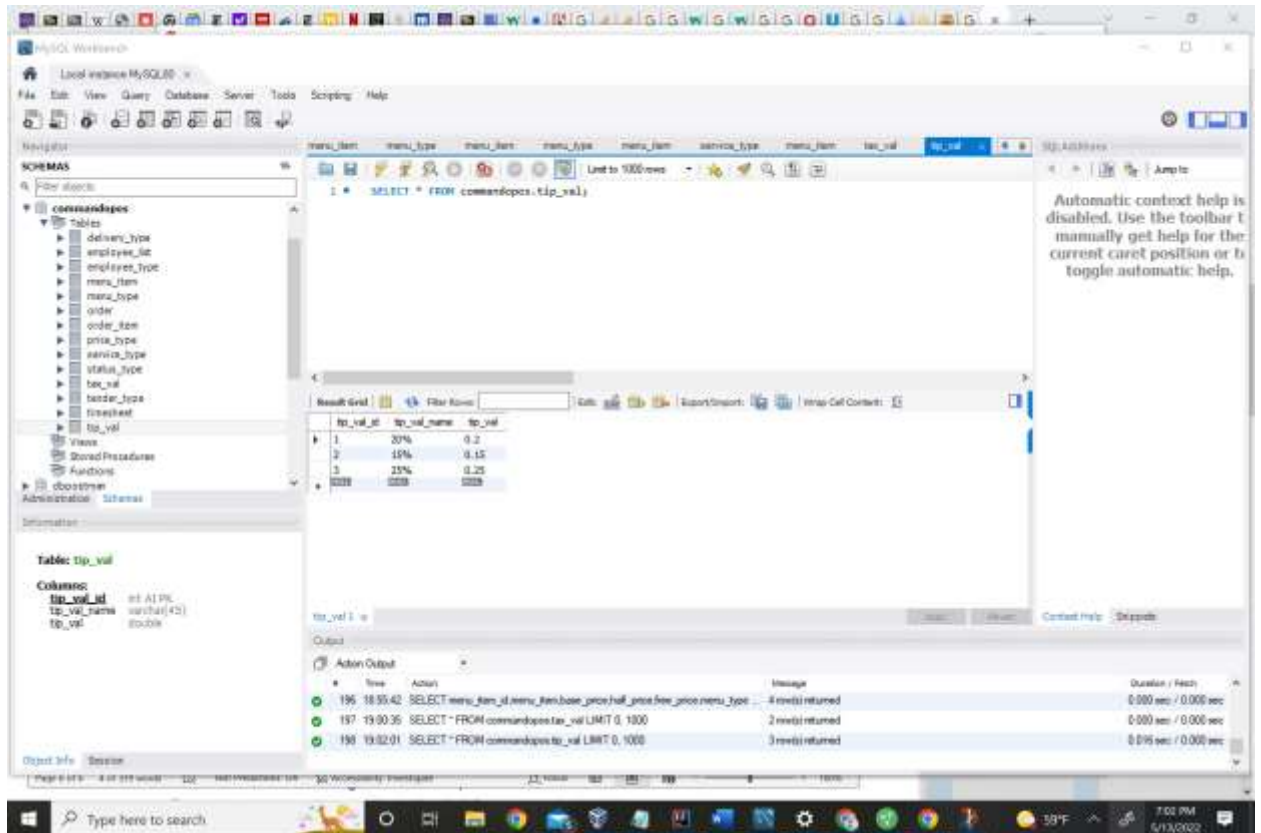
SELECT \* FROM commandpos.tax\_val;

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane displays the 'commandpos' database with various tables. The 'tax\_val' table is selected, and its structure is shown in the 'Table: tax\_val' pane. The table has three columns: 'tax\_val\_id' (int, 42 PK), 'tax\_val\_label' (varchar(45)), and 'tax\_val' (double). The main editor shows the SQL query 'SELECT \* FROM commandpos.tax\_val;'. The 'Result Grid' pane displays the query results, showing two rows of data. The 'Output' pane at the bottom shows the execution log, indicating that the query was executed successfully and returned 2 rows.

tax_val_id	tax_val_label	tax_val
1	alcohol	0.1
2	prepared foods	0.0735

Make sure we have 2 calculated values for gratuities

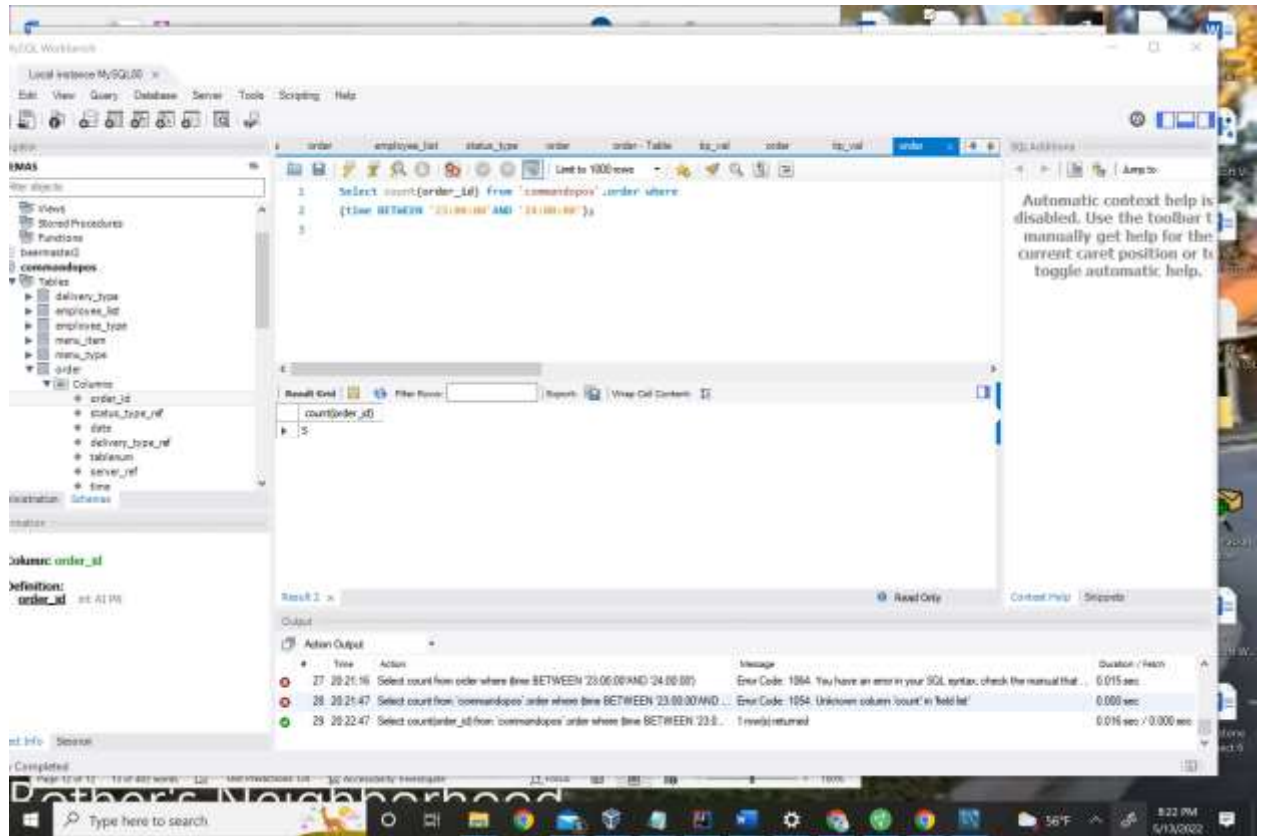
SELECT \* FROM commandpos.tip\_val;



Make sure we can generate the number of orders, average items per order for each hour in a given day.

Number of orders between 11:00PM -12:00AM

Select count(order\_id) from `commandopos`.order where  
(time BETWEEN '23:00:00'AND '24:00:00');



Average items per order for each between 11:00PM -12:00AM

select count(order\_item\_id)/(Select count(order\_id) from `commandopos`.order where  
(time BETWEEN '23:00:00'AND '24:00:00')

) from `commandopos`.order\_item where order\_ref IN (SELECT order\_id from  
'commandopos'.order where (time BETWEEN '23:00:00'AND '24:00:00'))

MySQL Workbench

Local instance MySQL80

File View Query Database Server Tools Scripting Help

Server: localhost

Database: commandops

Table: order\_items

Query:

```
1 select count(order_item_id)/(select count(order_id) from 'commandops'.order where  
2 {time BETWEEN '23:00:00' AND '23:00:00'})  
3 } from 'commandops'.order_items where order_ref IN (SELECT order_id from 'commandops'.order where {time  
4  
5
```

Result Grid

count(order_item_id)/(select count(order_id) from 'commandops'.order where {time BETWEEN '23:00:00' AND '23:00:00'})
1.0000

Schema: commandops

Result 1

Output

Action Output

Time	Action	Message	Duration / Pct
64 21:45:40	select count(distinct order_item_id) from 'commandops'.order_items where order...	Error Code: 1342. Subquery returns more than 1 row	0.016 sec
65 21:49:34	select count(order_item_id) from 'commandops'.order_items where order_ref IN (...)	1 row(s) returned	0.015 sec / 0.000 sec
66 21:50:03	select count(order_item_id)/(select count(order_id) from 'commandops'.order ...)	1 row(s) returned	0.031 sec / 0.000 sec

MySQL Workbench

Type here to search

9:55 PM 4/13/2022