

ORDER MANAGEMENT SYSTEM

Here in the order management system firstly, I have created a database in mysql with the schema name "ordermanagement_system" and then created a table and named: "products" and added (inserted) few products (values) which should be in the inventory. Later on I have connected database into the python and that code is shown below source code. Later on using sql in python I have created a orderline (table which have product name, product_id and status) added orders in the orderline and verified whether they are present in the inventory (products table) or not. And followed the problem statement where commands of multiple types created and executed as per the requirement

ORDERMANAGEMENT SYSTEM source code:

```
from contextlib import nullcontext
#database connection
import os
import mysql.connector
database_connection= mysql.connector.connect(
    host="localhost",
    user="root",
    password="Srav123@",
    database="ordermanagement_system"
)
cursor = database_connection.cursor()

import sys

def create_order():
    get_id = """select order_id from orderline order by order_id desc limit 1"""
    cursor.execute(get_id)
    order_id_res = cursor.fetchall()
    if order_id_res:
        order_id = int(order_id_res[0][0])
        order_id = (str(order_id + 1), )
    else:
        order_id = ('1', )
    get_id = """insert into orderline(order_id) values (%s)"""
    cursor.execute(get_id, order_id)
    database_connection.commit()
    print("Order created with id ",order_id[0])

def add_order_line(ord_id, prd_name, product_qnt):
```

```
product_name = (prd_name, )
quantity = str(product_qnt)
order_id = (str(ord_id), )
```

```
check_product = """select product_name from products where product_name=%s"""
cursor.execute(check_product, product_name)
check_product_res = cursor.fetchall()
if not check_product_res:
    return "Product is not available"
```

```
check_order_id = """select order_id from orderline where order_id=%s"""
cursor.execute(check_order_id, order_id)
order_id_res = cursor.fetchall()
if not order_id_res:
    return "Order id is not available"
update_product = "UPDATE orderline SET product_name = '{}', product_quantity = '{}', status =
'Draft' WHERE order_id = '{}".format(product_name[0], quantity, order_id[0])
cursor.execute(update_product)
database_connection.commit()

return "{} {} added to order {}".format(quantity, product_name[0], order_id[0])
```

```
def show_order_id(order_id):
    get_order = """select * from orderline where order_id=%s"""
    cursor.execute(get_order, (str(order_id), ))
    get_order_res = cursor.fetchall()
    for x in get_order_res:
        print("Order", x[0], "Draft", x[1])
        print(x[2], x[1], "Draft")
```

```
def show_orders():
    get_order = """select * from orderline"""
    cursor.execute(get_order)
    get_order_res = cursor.fetchall()
    for x in get_order_res:
        if x[2]:
            print("Order", x[0], "Draft", x[1])
            print(x[2], x[1], "Draft")
```

```
given_command = sys.argv[0]
if given_command == "CREATE_ORDER":
    create_order()
elif given_command == "ADD_ORDERLINE":
```

```

order_id = sys.argv[2]
product_name = sys.argv[3]
product_qnt = sys.argv[4]
response = add_order_line(order_id, product_name, product_qnt)
print(response)
elif given_command == "SHOW_ORDER":
    show_order_id(sys.argv[2])
elif given_command == "SHOW_ORDERS":
    show_orders()

select=input("Please select any one option below \n 1.CREATE_ORDER\n 2.ADD_PRODUCT\n
3.SHOW_ORDER \n 4.SHOW_ORDERS\n")
if select=="CREATE_ORDER":
    create_order()
elif select=="ADD_PRODUCT":
    add_order_line()
elif select=="SHOW_ORDER":
    show_order_id(order_id)
elif select=="SHOW_ORDERS":
    show_orders()
else :
    print("please select valid option")

```

SQL QUERIES :

----->

First I have created a database and tables then inserted values into it(this is inventory)

```

CREATE DATABASE ordermanagement_system
USE ordermanagement_system;
CREATE TABLE products (
    product_name VARCHAR(20),
    product_quatity INT
);
INSERT INTO products(product_name,product_quatity )
VALUES
('apple','24'),
('mango','20'),
('kiwi','30'),
('avacado','28');

```

SCREENSHOTS:

ordermanagementsys.py - Visual Studio Code

```
1 from contextlib import nullcontext
2 import os
3 import mysql.connector
4
5 database_connection= mysql.connector.connect(
6     host="localhost",
7     user="root",
8     password="srav123@",
9     database="ordermanagement_system"
10 )
11 cursor = database_connection.cursor()
12
13 import sys
14
15 def create_order():
16     get_id = """select order_id from orderline order by order_id desc limit 1"""
17     cursor.execute(get_id)
18     order_id_res = cursor.fetchall()
19     if order_id_res:
20         order_id = int(order_id_res[0][0])
21         order_id = (str(order_id + 1), )
22     else:
```

Traceback (most recent call last):
File "c:/Users/srava/PycharmProjects/pythonProject1/test.py", line 6, in <module>
test_result = ordermanagementsys.create_order(test_input)
NameError: name 'test_input' is not defined
PS C:\Users\srava\pycharmprojects\pythonProject1> & c:/Python310/python.exe c:/Users/srava/PycharmProjects/pythonProject1/ordermanagementsys.py
Please select any one option below
1.CREATE_ORDER
2.ADD_PRODUCT
3.SHOW_ORDER
4.SHOW_ORDERS
SHOW_ORDERS
Order 3 Draft 2
mango 2 Draft
Order 4 Draft 5
mango 5 Draft
PS C:\Users\srava\pycharmprojects\pythonProject1>

12°C Cloudy

main.py - Visual Studio Code

```
1 from contextlib import nullcontext
2 import os
3 import mysql.connector
4
5 database_connection= mysql.connector.connect(
6     host="localhost",
7     user="root",
8     password="srav123@",
9     database="ordermanagement_system"
10 )
11 cursor = database_connection.cursor()
12
13 import sys
14
15 def create_order():
16     get_id = """select order_id from orderline order by order_id desc limit 1"""
17     cursor.execute(get_id)
18     order_id_res = cursor.fetchall()
19     if order_id_res:
20         order_id = int(order_id_res[0][0])
21         order_id = (str(order_id + 1), )
22     else:
```

PS C:\Users\srava\pycharmprojects\pythonProject1> & c:/Python310/python.exe c:/Users/srava/PycharmProjects/pythonProject1/main.py
Please select any one option below 1.CREATE_ORDER
2.ADD_PRODUCT
3.SHOW_ORDER
4.SHOW_ORDERS
& c:/Python310/python.exe c:/Users/srava/PycharmProjects/pythonProject1/main.py
please select valid option
PS C:\Users\srava\pycharmprojects\pythonProject1> & c:/Python310/python.exe c:/Users/srava/PycharmProjects/pythonProject1/main.py
Please select any one option below
1.CREATE_ORDER
2.ADD_PRODUCT
3.SHOW_ORDER
4.SHOW_ORDERS
CREATE_ORDER
Order created with id 6
PS C:\Users\srava\pycharmprojects\pythonProject1>

16°C Mostly cloudy

The screenshot shows the Visual Studio Code editor with a Python file named `main.py`. The script connects to a MySQL database named `ordermanagement_system` and defines a `create_order()` function. The function uses a cursor to execute a SQL query that selects the top order ID from the `orderline` table. It then checks if the order ID exists in the `orderline` table. If it does, it increments the order ID by 1. If not, it inserts a new order line with a product name and quantity.

```
1 from contextlib import nullcontext
2 import os
3 import mysql.connector
4
5 database_connection= mysql.connector.connect(
6     host="localhost",
7     user="root",
8     password="sraav123@",
9     database="ordermanagement_system"
10 )
11 cursor = database_connection.cursor()
12
13 import sys
14
15 def create_order():
16     get_id = """select order_id from orderline order by order_id desc limit 1"""
17     cursor.execute(get_id)
18     order_id_res = cursor.fetchall()
19     if order_id_res:
20         order_id = int(order_id_res[0][0])
21         order_id = (str(order_id + 1), )
22     else:
```

The terminal window shows the execution of the script using PowerShell. The script is run from the directory `C:\Users\sraav\PycharmProjects\pythonProject1`. The output shows that the script successfully added three orders: apple, mango, and mango.

```
PS C:\Users\sraav\PycharmProjects\pythonProject1> python .\main.py ADD_ORDERLINE 3 apple 2
2 apple added to order 3
PS C:\Users\sraav\PycharmProjects\pythonProject1> python .\main.py ADD_ORDERLINE 4 mango 5
5 mango added to order 4
PS C:\Users\sraav\PycharmProjects\pythonProject1>
```

DATABASE DETAILS:

The screenshot shows the MySQL Workbench interface. The left sidebar displays the database schema, including the `products` table. The main window shows a query result grid for the `products` table. The query executed is `select * from PRODUCTS;`. The result grid shows the following data:

product_name	product_quantity
apple	24
mango	20
lowi	30
avocado	28

The bottom panel shows the output of the script, including the execution time and the number of rows affected for each query.

#	Time	Action	Message	Duration / Fetch
2	00:17:44	DELETE from orderline where order_id=1	2 row(s) affected	0.000 sec
3	00:17:58	select * from orderline LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
4	01:05:26	select * from orderline LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
5	01:08:45	select * from orderline LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
6	01:17:03	select * from orderline LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
7	19:22:33	select * from PRODUCTS LIMIT 0, 1000	4 row(s) returned	0.016 sec / 0.000 sec

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

- bakery
- bank
- bookstore
- democracy
- ordermanagement_system
 - Tables
 - orderline
 - products
 - Columns
 - product_name
 - product_quantity
 - Indexes
 - Foreign Keys
 - Triggers
 - Views

Administration Schemas

Information

Table: products

Columns:

- product_name varchar(20)
- product_quantity int

Object Info Session

12°C Cloudy

19:06 17/10/2022

SQL Editor

```
1 USE ordermanagement_system;
2 select * from orderline;
3
4
5
```

Limit to 1000 rows

Result Grid

order_id	product_quantity	product_name	status
1	1	apple	OK
2	2	mango	Draft
4	5	mango	Draft
5	1	apple	OK

Filter Rows: Export: Wrap Cell Contents

Read Only Context Help Snippets

Output

Action Output

#	Time	Action	Message	Duration / Fetch
1	00:14:08	select * from orderline LIMIT 0, 1000	2 row(s) returned	0.000 sec / 0.000 sec
2	00:17:44	DELETE from orderline where order_id=1	2 row(s) affected	0.000 sec
3	00:17:58	select * from orderline LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
4	01:05:26	select * from orderline LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
5	01:08:45	select * from orderline LIMIT 0, 1000	5 row(s) returned	0.000 sec / 0.000 sec
6	01:17:03	select * from orderline LIMIT 0, 1000	5 row(s) 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.