ASSIGNMENT – 1 SQL - TechShop, an electronic gadgets shop

Name – Anish Kumar **Python Batch -1**

Task 1

1. **Create the database named "TechShop"**

```sql
create database Techshop;
```

100 %

Messages

```
Commands completed successfully.

Completion time: 2024-09-20T11:55:03.5777005+05:30
```

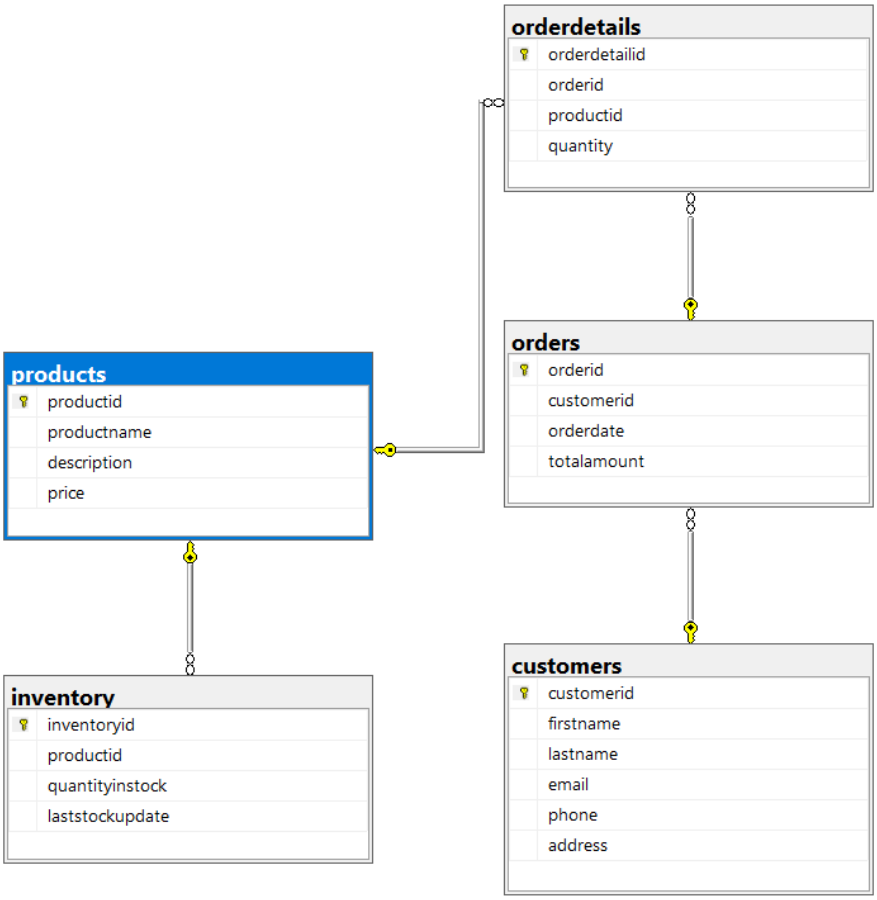
2. **Define the schema for the Customers, Products, Orders, OrderDetails and Inventory tables based on the provided schema.**

```sql
create table customers (
    customerid int identity primary key ,
    firstname varchar(50),
    lastname varchar(50),
    email varchar(100),
    phone varchar(15),
    address varchar(255)
);
create table products (
    productid int identity primary key ,
    productname varchar(100),
    description varchar(255),
    price decimal(10, 2)
);
create table orders (
    orderid int identity primary key ,
    customerid int,
    orderdate date,
    totalamount decimal(10, 2),
    foreign key (customerid) references customers(customerid)
);
create table orderdetails (
    orderdetailid int identity primary key ,
    orderid int,
    productid int,
    quantity int,
    foreign key (orderid) references orders(orderid),
    foreign key (productid) references products(productid)
);
create table inventory (
    inventoryid int identity primary key ,
    productid int,
    quantityinstock int,
    laststockupdate date,
    foreign key (productid) references products(productid)
);
```

```
100 %
Messages
Commands completed successfully.

Completion time: 2024-09-20T12:12:52.3967049+05:30
```

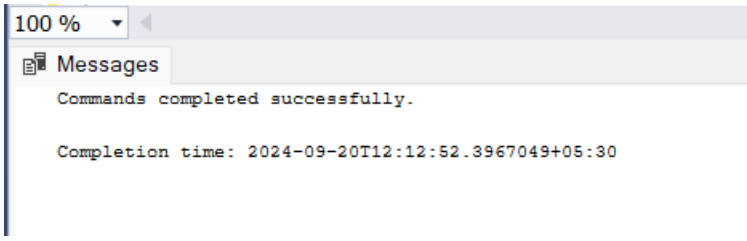
3. **Create an ERD (Entity Relationship Diagram) for the database.**



4. **Create appropriate Primary Key and Foreign Key constraints for referential integrity.**

```
create table products (
    productid int identity primary key ,
    productname varchar(100),
    description varchar(255),
    price decimal(10, 2)
);
create table orders (
    orderid int identity primary key ,
```

```sql
    customerid int,
    orderdate date,
    totalamount decimal(10, 2),
    foreign key (customerid) references customers(customerid)
);
create table orderdetails (
    orderdetailid int identity primary key ,
    orderid int,
    productid int,
    quantity int,
    foreign key (orderid) references orders(orderid),
    foreign key (productid) references products(productid)
);
create table inventory (
    inventoryid int identity primary key ,
    productid int,
    quantityinstock int,
    laststockupdate date,
    foreign key (productid) references products(productid)
       );
```



5. **Insert at least 10 sample records into each of the following tables.**
   a. **Customers**
   b. **Products**
   c. **Orders**
   d. **OrderDetails**
   e. **Inventory**

```sql
insert into customers (firstname, lastname, email, phone, address)
values
('peter', 'smith', 'jane.smith@example.com', '0987654321', 'Avenger Avenue
321'),
('Dan', 'williams', 'e.williams@example.com', '3456789012', '321 maple blvd'),
('daniel', 'brown', 'd.brown@example.com', '4567890123', '654 cedar court'),
('michael', 'johnson', 'm.johnson@example.com', '2345678901', '789 pine road'),
('Aussie', 'davis', 'emma.davis@example.com', '5678901234', '987 birch lane'),
('john', 'doe', 'john.doe@example.com', '1234567890', '123 elm street'),
('olivia', 'taylor', 'o.taylor@example.com', '9012345678', '357 hickory
place'),
('william', 'miller', 'w.miller@example.com', '6789012345', '159 spruce
street'),
('sophia', 'wilson', 'sophia.wilson@example.com', '7890123456', '753 sycamore
drive'),
('james', 'moore', 'j.moore@example.com', '8901234567', '951 redwood circle');

insert into products (productname, description, price)
values
```

```sql
('laptop', 'high-performance laptop', 999.99),
('smartphone', 'latest smartphone model', 799.99),
('tablet', '10-inch display tablet', 499.99),
('smartwatch', 'wearable smart device', 199.99),
('headphones', 'noise-cancelling headphones', 149.99),
('keyboard', 'mechanical keyboard', 99.99),
('mouse', 'wireless mouse', 49.99),
('monitor', '27-inch 4k monitor', 299.99),
('speaker', 'bluetooth speaker', 59.99),
('charger', 'fast charging adapter', 29.99);

insert into orders (customerid, orderdate, totalamount)
values
(1, '2024-09-01', 1299.98),
(2, '2024-09-02', 1049.98),
(3, '2024-09-03', 1499.98),
(4, '2024-09-04', 549.98),
(5, '2024-09-05', 249.98),
(6, '2024-09-06', 849.98),
(7, '2024-09-07', 399.98),
(8, '2024-09-08', 229.98),
(9, '2024-09-09', 399.98),
(10, '2024-09-10', 699.98);

insert into orderdetails (orderid, productid, quantity)
values
(1, 1, 1), (1, 2, 1),
(2, 3, 2),
(3, 4, 3),
(4, 5, 1),
(5, 6, 2),
(6, 7, 1),
(7, 8, 1),
(8, 9, 2),
(9, 10, 3);

insert into inventory (productid, quantityinstock, laststockupdate)
values
(1, 50, '2024-09-01'),
(2, 100, '2024-09-02'),
(3, 75, '2024-09-03'),
(4, 60, '2024-09-04'),
(5, 30, '2024-09-05'),
(6, 40, '2024-09-06'),
(7, 85, '2024-09-07'),
(8, 25, '2024-09-08'),
(9, 70, '2024-09-09'),
(10, 90, '2024-09-10');
```

100 %

Messages

```
(10 rows affected)

(10 rows affected)

(10 rows affected)

(10 rows affected)

(10 rows affected)

Completion time: 2024-09-20T12:24:23.3311103+05:30
```

Task 2

1. Write an SQL query to retrieve the names and emails of all customers.

```
select firstname, lastname, email
from customers;
```

100 %

Results | Messages

	firstname	lastname	email
1	peter	smith	jane.smith@example.com
2	Dan	williams	e.williams@example.com
3	daniel	brown	d.brown@example.com
4	michael	johnson	m.johnson@example.com
5	Aussie	davis	emma.davis@example.com
6	john	doe	john.doe@example.com
7	olivia	taylor	o.taylor@example.com
8	william	miller	w.miller@example.com
9	sophia	wilson	sophia.wilson@example.com
10	james	moore	j.moore@example.com

2. Write an SQL query to list all orders with their order dates and corresponding customer names.

```
select orders.orderid, orders.orderdate, customers.firstname,
customers.lastname
from orders
inner join customers
on orders.customerid = customers.customerid;
```

100 %

Results | Messages

	orderid	orderdate	firstname	lastname
1	1	2024-09-01	peter	smith
2	2	2024-09-02	Dan	williams
3	3	2024-09-03	daniel	brown
4	4	2024-09-04	michael	johnson
5	5	2024-09-05	Aussie	davis
6	6	2024-09-06	john	doe
7	7	2024-09-07	olivia	taylor
8	8	2024-09-08	william	miller
9	9	2024-09-09	sophia	wilson
10	10	2024-09-10	james	moore

3. **Write an SQL query to insert a new customer record into the "Customers" table. Include customer information such as name, email, and address.**

```sql
insert into customers (firstname, lastname, email, phone, address)
values ('alex', 'johnson', 'alex.johnson@example.com', '1234567890', '456
oak street');
```

100 %

Messages

```
(1 row affected)

Completion time: 2024-09-20T12:31:03.5724586+05:30
```

4. **Write an SQL query to update the prices of all electronic gadgets in the "Products" table by increasing them by 10%.**

```sql
update products
set price = price * 1.10 ;
select * from products;
```

100 %

Results | Messages

	productid	productname	description	price
1	1	laptop	high-performance laptop	1209.99
2	2	smartphone	latest smartphone model	967.99
3	3	tablet	10-inch display tablet	604.99
4	4	smartwatch	wearable smart device	241.99
5	5	headphones	noise-cancelling headphones	181.49
6	6	keyboard	mechanical keyboard	120.99
7	7	mouse	wireless mouse	60.49
8	8	monitor	27-inch 4k monitor	362.99
9	9	speaker	bluetooth speaker	72.59
10	10	charger	fast charging adapter	36.29

5. **Write an SQL query to delete a specific order and its associated order details from the "Orders" and "OrderDetails" tables. Allow users to input the order ID as a parameter.**

```sql
declare @orderid int = 1;

delete from orderdetails
where orderid = @orderid;

delete from orders
where orderid = @orderid;
```

100 %

Messages

```
(2 rows affected)

(1 row affected)

Completion time: 2024-09-20T12:38:10.4146018+05:30
```

6. **Write an SQL query to insert a new order into the "Orders" table. Include the customer ID, order date, and any other necessary information.**

```sql
insert into orders (customerid, orderdate, totalamount)
values (11, '2024-09-19', 289.99);
```

100 %

Messages

```
(1 row affected)

Completion time: 2024-09-20T12:39:49.4535912+05:30
```

7. **Write an SQL query to update the contact information (e.g., email and address) of a specific customer in the "Customers" table. Allow users to input the customer ID and new contact information.**

```sql
update customers
set email = 'new.email',
address = '777 Los Angeles'
where CustomerID = 7;
```

```
100 %
Messages

(1 row affected)

Completion time: 2024-09-20T12:31:03.5724586+05:30
```

8. **Write an sql query to recalculate and update the total cost of each order in the "orders" table based on the prices and quantities in the "orderdetails" table.**

```sql
update orders
set totalamount = (
    select
        sum(od.quantity * p.price)
    from
        orderdetails od
    join
        products p on od.productid = p.productid
    where
        od.orderid = orders.orderid );
```

```
100 %
Messages

(10 rows affected)

Completion time: 2024-09-20T12:51:20.0209858+05:30
```

9. **Write an SQL query to delete all orders and their associated order details for a specific customer from the "Orders" and "OrderDetails" tables. Allow users to input the customer ID as a parameter.**

```sql
declare @customerid int = 6;

delete from orderdetails
```

```sql
where orderid in (
    select orderid
    from orders
    where customerid = @customerid
);

delete from orders
where customerid = @customerid;
```

```
100 %
Messages

(1 row affected)

(1 row affected)

Completion time: 2024-09-20T12:55:17.7503934+05:30
```

10. Write an SQL query to insert a new electronic gadget product into the "Products" table, including product name, category, price, and any other relevant details.

```sql
insert into products (productname, description, price)
values ('smartphone f11', 'latest model with snapdragon 8 gen 4', 899.99);
```

```
100 %
Messages

(1 row affected)

Completion time: 2024-09-20T12:57:47.4714760+05:30
```

11. Write an SQL query to update the status of a specific order in the "Orders" table (e.g., from "Pending" to "Shipped"). Allow users to input the order ID and the new status.

```sql
alter table orders
add status varchar(50);

declare @orderid int = 4;
declare @newstatus varchar(20) = 'shipped';

update orders
set status = @newstatus
where orderid = @orderid;
```

```
100 %
Messages

(1 row affected)

Completion time: 2024-09-20T13:01:28.6001902+05:30
```

12. **Write an SQL query to calculate and update the number of orders placed by each customer in the "Customers" table based on the data in the "Orders" table.**

```sql
alter table customers
add ordercount int default 0;

update customers
set ordercount = (
    select count(*)
    from orders
    where orders.customerid = customers.customerid
          );
```

```
100 %
Messages

(1 row affected)

Completion time: 2024-09-20T13:01:28.6001902+05:30
```

TASK 3

1. **Write an SQL query to retrieve a list of all orders along with customer information (e.g., customer name) for each order.**

```sql
select o.orderid,
    o.orderdate,
    o.totalamount,
    c.customerid,
      concat(c.firstname, ' ', c.lastname) as customername from orders as
o
inner join
      customers as c
      on o.customerid=c.customerid;
```

100 %

Results | Messages

	OrderID	OrderDate	TotalAmount	CustomerID	CustomerName
1	1	2024-09-01	1299.98	1	John Doe
2	2	2024-09-02	1049.98	2	Jane Smith
3	3	2024-09-03	1499.98	3	Michael Johnson
4	4	2024-09-04	549.98	4	Emily Williams
5	5	2024-09-05	249.98	5	Daniel Brown
6	7	2024-09-07	399.98	7	William Miller
7	8	2024-09-08	229.98	8	Sophia Wilson
8	9	2024-09-09	399.98	9	James Moore
9	10	2024-09-10	699.98	10	Olivia Taylor
10	11	2024-09-19	289.99	11	Alex Johnson

2. **Write an SQL query to find the total revenue generated by each electronic gadget product. Include the product name and the total revenue.**

```sql
select
    p.productname,
    sum(od.quantity * p.price) as totalrevenue
from
    orderdetails od
inner join
    products p on od.productid = p.productid
group by
          p.productname;
```

100 %

Results | Messages

	ProductName	TotalRevenue
1	Charger	98.97
2	Headphones	164.99
3	Keyboard	219.98
4	Laptop	1099.99
5	Monitor	329.99
6	Smartphone	879.99
7	Smartwatch	659.97
8	Speaker	131.98
9	Tablet	1099.98

3. **Write an SQL query to list all customers who have made at least one purchase. Include their names and contact information.**

```sql
select
    c.customerid,
    concat(c.firstname, ' ', c.lastname) as customername,
    c.email,
    c.phone,
    c.address
from
    customers c
inner join
```

```sql
    orders o on c.customerid = o.customerid
group by
    c.customerid, c.firstname, c.lastname, c.email, c.phone, c.address
order by
          customername;
```

100 %

Results | Messages

	CustomerID	CustomerName	Email	Phone	Address
1	11	Alex Johnson	alex.johnson@example.com	1234567890	456 Oak Street
2	5	Daniel Brown	d.brown@example.com	4567890123	654 Cedar Court
3	4	Emily Williams	e.williams@example.com	3456789012	321 Maple Blvd
4	9	James Moore	j.moore@example.com	8901234567	951 Redwood Circle
5	2	Jane Smith	jane.smith@example.com	0987654321	456 Oak Avenue
6	1	John Doe	john.doe@example.com	1234567890	123 Elm Street
7	3	Michael Johnson	m.johnson@example.com	2345678901	789 Pine Road
8	10	Olivia Taylor	o.taylor@example.com	9012345678	357 Hickory Place
9	8	Sophia Wilson	sophia.wilson@example.com	7890123456	753 Sycamore Drive
10	7	William Miller	new.email	6789012345	777 tokyo

4. **Write an SQL query to find the most popular electronic gadget, which is the one with the highest total quantity ordered. Include the product name and the total quantity ordered.**

```sql
select top 1
    p.productname,
    sum(od.quantity) as totalquantityordered
from
    orderdetails od
inner join
    products p on od.productid = p.productid
group by
    p.productname
order by
      totalquantityordered desc;
```

100 %

Results | Messages

	ProductName	TotalQuantityOrdered
1	Charger	3

5. **Write an SQL query to retrieve a list of electronic gadgets along with their corresponding categories.**

```sql
select
    productname,
    category
from
    products
where
       category = 'electronics';
```

100 %

Results | Messages

	ProductName	Category
1	Laptop	Electronics
2	Smartphone	Electronics
3	Tablet	Electronics
4	Smartwatch	Electronics
5	Headphones	Electronics
6	Monitor	Electronics
7	Speaker	Electronics
8	Smartphone X10	Electronics

6. **Write an SQL query to calculate the average order value for each customer. Include the customer's name and their average order value.**

```sql
select
    concat(c.firstname, ' ', c.lastname) as customername,
    avg(o.totalamount) as averageordervalue
from
    customers c
inner join
    orders o on c.customerid = o.customerid
group by
          c.firstname, c.lastname;
```

100 %

Results | Messages

	CustomerName	AverageOrderValue
1	Daniel Brown	249.980000
2	John Doe	1299.980000
3	Alex Johnson	289.990000
4	Michael Johnson	1499.980000
5	William Miller	399.980000
6	James Moore	399.980000
7	Jane Smith	1049.980000
8	Olivia Taylor	699.980000
9	Emily Williams	549.980000
10	Sophia Wilson	229.980000

7. **Write an SQL query to find the order with the highest total revenue. Include the order ID, customer information, and the total revenue.**

```sql
with ordertotals as (
    select
        o.orderid,
        o.customerid,
        o.totalamount,
        concat(c.firstname, ' ', c.lastname) as customername,
        c.email,
```

```sql
        c.phone,
        c.address
    from
        orders o
    inner join
        customers c on o.customerid = c.customerid
)
select
    orderid,
    customername,
    email,
    phone,
    address,
    totalamount as totalrevenue
from
    ordertotals
where
            totalamount = (select max(totalamount) from orders);
```

100 %

Results | Messages

	OrderID	CustomerName	Email	Phone	Address	TotalRevenue
1	3	Michael Johnson	m.johnson@example.com	2345678901	789 Pine Road	1499.98

8. Write an SQL query to list electronic gadgets and the number of times each product has been ordered.

```sql
select
    p.productname,
    count(od.orderid) as timesordered
from
    products p
inner join
    orderdetails od on p.productid = od.productid
where
    p.productname in ('laptop', 'smartphone', 'tablet', 'smartwatch',
'headphones', 'monitor', 'speaker')
group by
    p.productname
order by
            timesordered desc;
```

100 %

Results | Messages

	ProductName	TimesOrdered
1	headphones	1
2	monitor	1
3	smartwatch	1
4	speaker	1
5	tablet	1

9. **Write an SQL query to find customers who have purchased a specific electronic gadget product. Allow users to input the product name as a parameter.**

```
declare @productname varchar(100) = 'tablet';

select distinct
    c.customerid,
    c.firstname,
    c.lastname,
    c.email,
    c.phone,
    c.address
from
    orderdetails od
join
    orders o on od.orderid = o.orderid
join
    products p on od.productid = p.productid
join
    customers c on o.customerid = c.customerid
where
    p.productname = @productname;
```

100 %

Results | Messages

	CustomerID	FirstName	LastName	Email	Phone	Address
1	2	Dan	williams	e.williams@example.com	3456789012	321 maple blvd

10. **Write an SQL query to calculate the total revenue generated by all orders placed within a specific time period. Allow users to input the start and end dates as parameters.**

```
declare @startdate varchar(100) = '2024-09-01';
declare @enddate varchar(100) = '2024-09-07';
```

```sql
select
    sum(totalamount) as totalrevenue
from
    orders
where
            orderdate between @startdate and @enddate;
```

100 %

Results | Messages

	TotalRevenue
1	2722.41

Task 4:

1. **Write an SQL query to find out which customers have not placed any orders.**

```sql
select
    c.customerid,
    c.firstname,
    c.lastname,
    c.email,
    c.phone,
    c.address
from
    customers c
where
    c.customerid not in (
        select o.customerid
        from orders o
            );
```

110 %

Results | Messages

	customerid	firstname	lastname	email	phone	address
1	11	Alex	Johnson	alex.johnson@example.com	1234567890	456 Oak Street

2. **Write an SQL query to find the total number of products available for sale.**

```sql
SELECT
    SUM(QuantityInStock) AS TotalProductsAvailable
FROM
            Inventory;
```

100 %

Results | Messages

	TotalProductsAvailable
1	625

3. **Write an SQL query to calculate the total revenue generated by TechShop.**

```
SELECT
    SUM(OD.Quantity * P.Price) AS TotalRevenue
FROM
    OrderDetails OD
JOIN
            Products P ON OD.ProductID = P.ProductID;
```

100 %

Results | Messages

	TotalRevenue
1	5214.93

4. **Write an SQL query to calculate the average quantity ordered for products in a specific category. Allow users to input the category name as a parameter.**

```
DECLARE @CategoryName VARCHAR(50)= 'Electronics';

SELECT
    AVG(OD.Quantity) AS AverageQuantityOrdered
FROM
    OrderDetails OD
WHERE
    OD.ProductID IN (
        SELECT P.ProductID
        FROM Products P
        WHERE P.Category = @CategoryName
           );
```

100 %

Results | Messages

	AverageQuantityOrdered
1	1

5. **Write an SQL query to calculate the total revenue generated by a specific customer. Allow users to input the customer ID as a parameter.**

```sql
DECLARE @CustomerID INT= 4;

SELECT
    SUM(OD.Quantity * P.Price) AS TotalRevenue
FROM
    Orders O
JOIN
    OrderDetails OD ON O.OrderID = OD.OrderID
JOIN
    Products P ON OD.ProductID = P.ProductID
WHERE
            O.CustomerID = @CustomerID;
```

100 %

Results | Messages

	TotalRevenue
1	181.49

6. **Write an SQL query to find the customers who have placed the most orders. List their names and the number of orders they've placed.**

```sql
SELECT
    C.FirstName,
    C.LastName,
    COUNT(O.OrderID) AS NumberOfOrders
FROM
    Customers C
JOIN
    Orders O ON C.CustomerID = O.CustomerID
GROUP BY
    C.CustomerID, C.FirstName, C.LastName
ORDER BY
            NumberOfOrders DESC;
```

100 %

Results | Messages

	FirstName	LastName	NumberOfOrders
1	peter	smith	1
2	Dan	williams	1
3	daniel	brown	1
4	michael	johnson	1
5	Aussie	davis	1
6	john	doe	1
7	olivia	taylor	1
8	william	miller	1
9	sophia	wilson	1
10	james	moore	1

7. **Write an SQL query to find the most popular product category, which is the one with the highest total quantity ordered across all orders.**

```sql
WITH CategoryQuantities AS (
    SELECT
        P.Category,
        SUM(OD.Quantity) AS TotalQuantityOrdered
    FROM
        OrderDetails OD
    JOIN
        Products P ON OD.ProductID = P.ProductID
    GROUP BY
        P.Category
)

SELECT
    Category,
    TotalQuantityOrdered
FROM
    CategoryQuantities
WHERE
    TotalQuantityOrdered = (SELECT MAX(TotalQuantityOrdered) FROM
CategoryQuantities);
```

100 %

Results | Messages

	Category	TotalQuantityOrdered
1	Electronics	11

8. **Write an SQL query to find the customer who has spent the most money (highest total revenue) on electronic gadgets. List their name and total spending.**

```sql
SELECT
    C.FirstName,
    C.LastName,
    SUM(OD.Quantity * P.Price) AS TotalSpending
FROM
    Customers C
JOIN
    Orders O ON C.CustomerID = O.CustomerID
JOIN
    OrderDetails OD ON O.OrderID = OD.OrderID
JOIN
    Products P ON OD.ProductID = P.ProductID
GROUP BY
    C.CustomerID, C.FirstName, C.LastName
ORDER BY
    TotalSpending DESC
       OFFSET 0 ROWS FETCH NEXT 1 ROW ONLY;
```

100 %

Results | Messages

	FirstName	LastName	TotalSpending
1	peter	smith	2177.98

9. **Write an SQL query to calculate the average order value (total revenue divided by the number of orders) for all customers.**

```
SELECT
    SUM(OD.Quantity * P.Price) / NULLIF(COUNT(DISTINCT O.OrderID), 0) AS
AverageOrderValue
FROM
    Orders O
JOIN
    OrderDetails OD ON O.OrderID = OD.OrderID
JOIN
            Products P ON OD.ProductID = P.ProductID;
```

100 %

Results | Messages

	AverageOrderValue
1	579.436666

10. **Write an SQL query to find the total number of orders placed by each customer and list their names along with the order count.**

```
SELECT
    C.FirstName,
    C.LastName,
    COUNT(O.OrderID) AS TotalOrders
FROM
    Customers C
LEFT JOIN
    Orders O ON C.CustomerID = O.CustomerID
GROUP BY
    C.CustomerID, C.FirstName, C.LastName
ORDER BY
            TotalOrders DESC;
```

100 %

Results | Messages

	FirstName	LastName	TotalOrders
1	peter	smith	1
2	Dan	williams	1
3	daniel	brown	1
4	michael	johnson	1
5	Aussie	davis	1
6	john	doe	1
7	olivia	taylor	1
8	william	miller	1
9	sophia	wilson	1
10	james	moore	1
11	Alex	Johnson	0

----------Complete----------