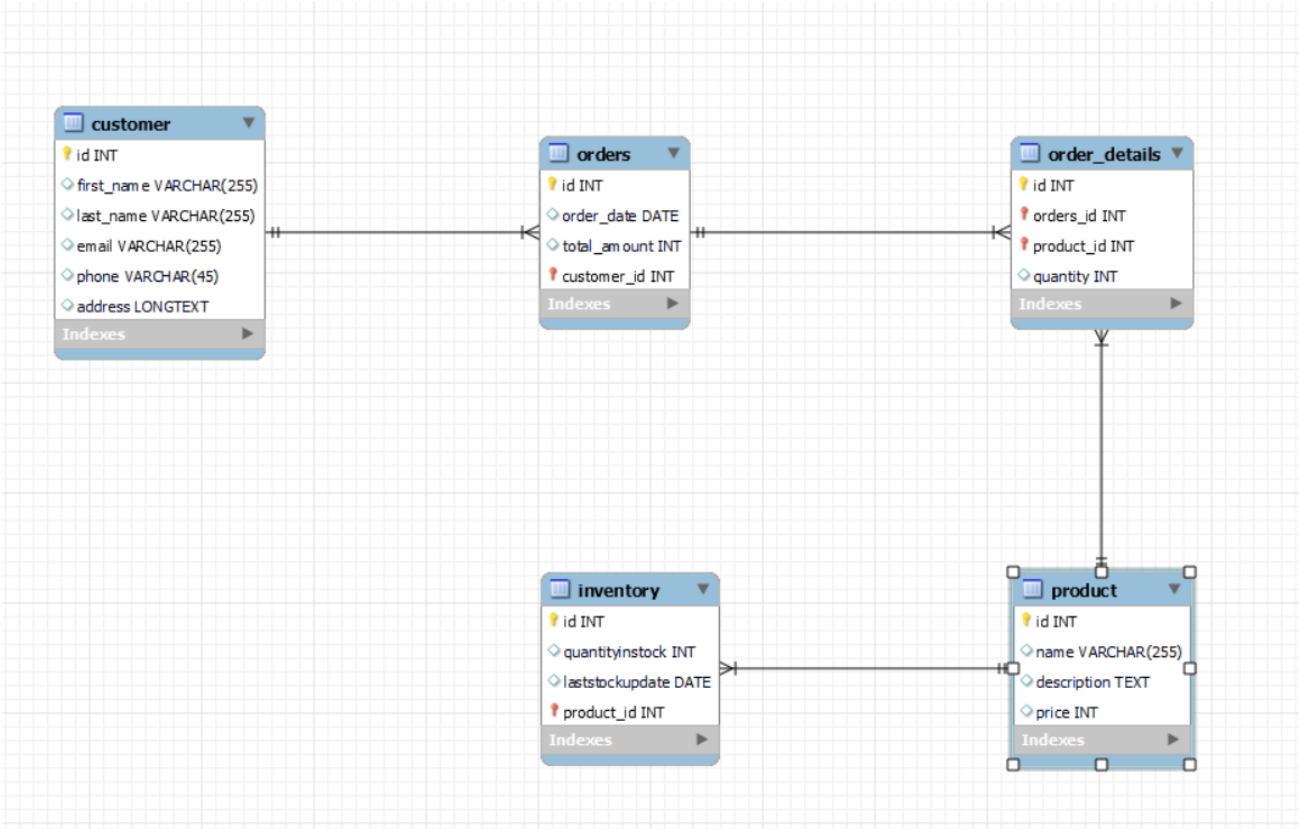
ASSIGNMENT NO : 1

TECHSHOP, AN ELECTRONIC GADGETS SHOP

ER DIAGRAM:



Task:1. Database Design:

-- MySQL Workbench Forward Engineering

-- ---

-- Schema techshop

-- ---

-- ---

-- Schema techshop

-- ---

CREATE SCHEMA IF NOT EXISTS `techshop` DEFAULT CHARACTER SET utf8 ;

USE `techshop` ;

```
-- -----------------------------------------------------
-- Table `techshop`.`customer`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `techshop`.`customer` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `first_name` VARCHAR(255) NULL,
  `last_name` VARCHAR(255) NULL,
  `email` VARCHAR(255) NULL,
  `phone` VARCHAR(45) NULL,
  `address` LONGTEXT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `phone_UNIQUE` (`phone` ASC) )
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `techshop`.`product`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `techshop`.`product` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(255) NULL,
  `description` TEXT NULL,
  `price` INT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `techshop`.`inventory`
-- -----------------------------------------------------
```

```
CREATE TABLE IF NOT EXISTS `techshop`.`inventory` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `quantityinstock` INT NULL,
  `laststockupdate` DATE NULL,
  `product_id` INT NOT NULL,
  PRIMARY KEY (`id`, `product_id`),
  INDEX `fk_inventory_product1_idx` (`product_id` ASC) ,
  CONSTRAINT `fk_inventory_product1`
    FOREIGN KEY (`product_id`)
    REFERENCES `techshop`.`product` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `techshop`.`orders`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `techshop`.`orders` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `order_date` DATE NULL,
  `total_amount` INT NULL,
  `customer_id` INT NOT NULL,
  PRIMARY KEY (`id`, `customer_id`),
  INDEX `fk_orders_customer_idx` (`customer_id` ASC) ,
  CONSTRAINT `fk_orders_customer`
    FOREIGN KEY (`customer_id`)
    REFERENCES `techshop`.`customer` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
-- -----------------------------------------------------
-- Table `techshop`.`order_details`

-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `techshop`.`order_details` (

  `id` INT NOT NULL AUTO_INCREMENT,
  `orders_id` INT NOT NULL,
  `product_id` INT NOT NULL,
  `quantity` INT NULL,

  PRIMARY KEY (`id`, `orders_id`, `product_id`),
  INDEX `fk_order_details_orders1_idx` (`orders_id` ASC) ,
  INDEX `fk_order_details_product1_idx` (`product_id` ASC) ,
  CONSTRAINT `fk_order_details_orders1`
    FOREIGN KEY (`orders_id`)
    REFERENCES `techshop`.`orders` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_order_details_product1`
    FOREIGN KEY (`product_id`)
    REFERENCES `techshop`.`product` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

INSERTION:

-- customer insertion

```
insert into customer (first_name,last_name,email,phone,address)
values

('prasanna','prabakaran','prasanna@gmail.com','9360805403','arumparthapuram'),

('pradeep','munusamy','pradeep@gmail.com','9360805405','cuddalore'),

 ('niranjan','kumar','niranjan@gmail.com','9360805404','lawspet'),

 ('naveen','kumar','naveen@gmail.com','9360805433','rainbow nagar'),

('mani','bharathi','mani@gmail.com','9360805444','oldtown');

('gokul','kumar','gokul@gmail.com','9360805422','park'),

('sedhu','prakash','sedhu@gmail.com','9360805400','villianur'),

('selva','vignesh','selva@gmail.com','9360809864','ariyur'),

('ragul','balaji','ragul@gmail.com','9360803334','mudaliayrpet'),

('vinay','kumar','vinay@gmail.com','93608035642','nehru street');
```

```
mysql> select * from customer;
+----+------------+------------+--------------------+-------------+-----------------+
| id | first_name | last_name  | email              | phone       | address         |
+----+------------+------------+--------------------+-------------+-----------------+
| 11 | prasanna   | prabakaran | prasanna@gmail.com | 9360805403  | arumparthapuram |
| 12 | pradeep    | munusamy   | pradeep@gmail.com  | 9360805405  | cuddalore       |
| 13 | niranjan   | kumar      | niranjan@gmail.com | 9360805404  | lawspet         |
| 14 | naveen     | kumar      | naveen@gmail.com   | 9360805433  | rainbow nagar   |
| 15 | mani       | bharathi   | mani@gmail.com     | 9360805444  | oldtown         |
| 19 | gokul      | kumar      | gokul@gmail.com    | 9360805422  | park            |
| 20 | sedhu      | prakash    | sedhu@gmail.com    | 9360805400  | villianur       |
| 21 | selva      | vignesh    | selva@gmail.com    | 9360809864  | ariyur          |
| 22 | ragul      | balaji     | ragul@gmail.com    | 9360803334  | mudaliayrpet    |
| 23 | vinay      | kumar      | vinay@gmail.com    | 93608035642 | nehru street    |
+----+------------+------------+--------------------+-------------+-----------------+
10 rows in set (0.00 sec)
```

-- product insertion

```
INSERT INTO product (name, description, price) VALUES

('monitor', '2TB external hard drive for backup', 2000),

('smartphone', 'Latest smartphone with 6.5-inch display', 30000),

('mouse', 'High-speed wireless router for home network', 800),

('headphones', 'Wireless noise-canceling headphones', 2500),
```

('desktop', 'Powerful desktop computer for gaming', 3500),

('printer', 'Color inkjet printer with wireless capability', 4000),

('smartwatch', 'Fitness tracker and smartwatch combo', 8000),

('laptop', 'High-performance laptop with Intel Core i7', 55000),

('camera', 'Mirrorless camera with 24MP sensor', 20000),

('tablet', '10-inch tablet with high-resolution display', 10000);

```
mysql> select * from product;
+----+------------+-------------------------------------------------+-------+
| id | name       | description                                     | price |
+----+------------+-------------------------------------------------+-------+
|  1 | monitor    | 2TB external hard drive for backup              |  2000 |
|  2 | smartphone | Latest smartphone with 6.5-inch display         | 30000 |
|  3 | mouse      | High-speed wireless router for home network     |   800 |
|  4 | headphones | Wireless noise-canceling headphones             |  2500 |
|  5 | desktop    | Powerful desktop computer for gaming            |  3500 |
|  6 | printer    | Color inkjet printer with wireless capability |  4000 |
|  7 | smartwatch | Fitness tracker and smartwatch combo            |  8000 |
|  8 | laptop     | High-performance laptop with Intel Core i7      | 55000 |
|  9 | camera     | Mirrorless camera with 24MP sensor              | 20000 |
| 10 | tablet     | 10-inch tablet with high-resolution display     | 10000 |
+----+------------+-------------------------------------------------+-------+
```

-- order insertion

INSERT INTO orders (order_date,total_amount,customer_id) values

('2020/04/30',54000,11),

('2020/05/31',36000,12),

('2024/02/30',20000,13),

('2021/03/20',9000,14),

('2022/11/25',10000,15),

('2024/01/12',7500,19),

('2020/02/14',1000,20),

('2024/08/11',800,21),

('2023/02/04',54000,19),

('2021/03/30',36000,21),

('2023/03/30',14000,11),

('2023/10/15',81000,13),

('2020/05/29',3000,15),

('2021/06/29',7000,23),

('2022/11/22',10000,22),

('2023/12/25',10000,12),

('2020/09/30',15000,21),

('2020/06/30',400,11),

('2022/06/11',10000,23),

('2022/07/15',50000,1);

```
mysql> select * from orders;
+----+------------+--------------+-------------+
| id | order_date | total_amount | customer_id |
+----+------------+--------------+-------------+
| 21 | 2020-04-30 |        54000 |          11 |
| 22 | 2020-05-31 |        36000 |          12 |
| 23 | 0000-00-00 |        20000 |          13 |
| 24 | 2021-03-20 |         9000 |          14 |
| 25 | 2022-11-25 |        10000 |          15 |
| 26 | 2024-01-12 |         7500 |          19 |
| 27 | 2020-02-14 |         1000 |          20 |
| 28 | 2024-08-11 |          800 |          21 |
| 29 | 2023-02-04 |        54000 |          19 |
| 30 | 2021-03-30 |        36000 |          21 |
| 31 | 2023-03-30 |        14000 |          11 |
| 32 | 2023-10-15 |        81000 |          13 |
| 33 | 2020-05-29 |         3000 |          15 |
| 34 | 2021-06-29 |         7000 |          23 |
| 35 | 2022-11-22 |        10000 |          22 |
| 36 | 2023-12-25 |        10000 |          12 |
| 37 | 2020-09-30 |        15000 |          21 |
| 38 | 2020-06-30 |          400 |          11 |
| 39 | 2022-06-11 |        10000 |          23 |
| 40 | 2022-07-15 |        50000 |          14 |
+----+------------+--------------+-------------+
```

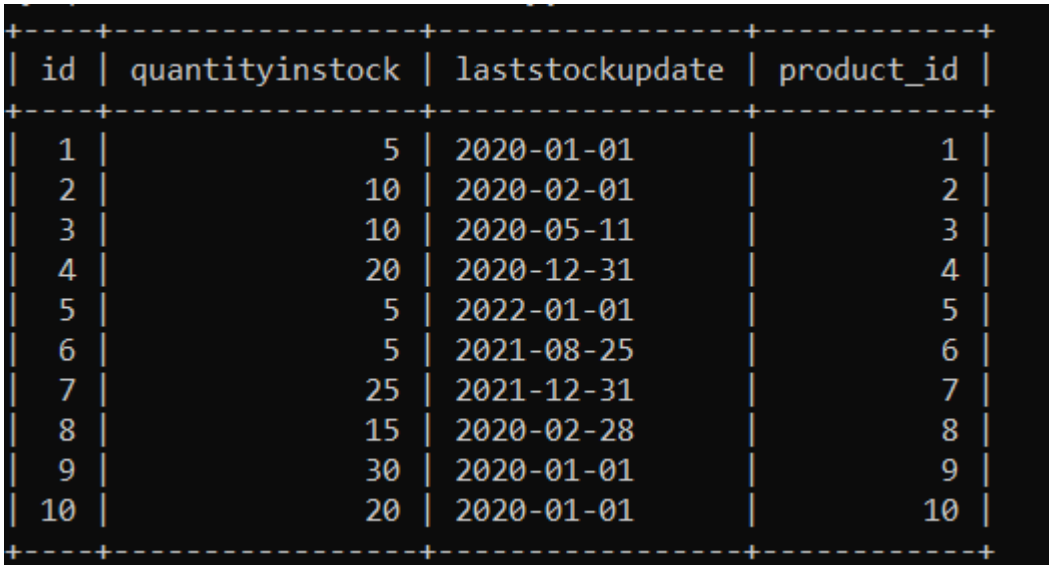
-- order detail insertion

```
insert into order_details(orders_id,product_id,quantity) values
(21,2,2),
(22,1,1),
(23,5,2),
(24,6,3),
(25,7,2),
(26,3,5),
(27,10,1),
(28,4,10),
(29,9,10),
(30,1,1),
(31,7,1),
(32,2,3),
(33,3,2),
(34,4,1),
(35,5,1),
(36,10,10),
(37,8,10),
(38,9,5),
(39,5,1),
(40,7,10);
```

```
mysql> select * from order_details;
+----+-----------+------------+----------+
| id | orders_id | product_id | quantity |
+----+-----------+------------+----------+
|  1 |        21 |          2 |        2 |
|  2 |        22 |          1 |        1 |
|  3 |        23 |          5 |        2 |
|  4 |        24 |          6 |        3 |
|  5 |        25 |          7 |        2 |
|  6 |        26 |          3 |        5 |
|  7 |        27 |         10 |        1 |
|  8 |        28 |          4 |       10 |
|  9 |        29 |          9 |       10 |
| 10 |        30 |          1 |        1 |
| 11 |        31 |          7 |        1 |
| 12 |        32 |          2 |        3 |
| 13 |        33 |          3 |        2 |
| 14 |        34 |          4 |        1 |
| 15 |        35 |          5 |        1 |
| 16 |        36 |         10 |       10 |
| 17 |        37 |          8 |       10 |
| 18 |        38 |          9 |        5 |
| 19 |        39 |          5 |        1 |
| 20 |        40 |          7 |       10 |
+----+-----------+------------+----------+
```

-- inventory insertion

```
insert into inventory(quantityinstock,laststockupdate,product_id) values
(5,'2020/01/01',1),
(10,'2020/02/01',2),
(10,'2020/05/11',3),
(20,'2020/12/31',4),
(5,'2022/01/01',5),
(5,'2021/08/25',6),
(25,'2021/12/31',7),
(15,'2020/02/28',8),
(30,'2020/01/01',9),
(20,'2020/01/01',10);
```



id	quantityinstock	laststockupdate	product_id
1	5	2020-01-01	1
2	10	2020-02-01	2
3	10	2020-05-11	3
4	20	2020-12-31	4
5	5	2022-01-01	5
6	5	2021-08-25	6
7	25	2021-12-31	7
8	15	2020-02-28	8
9	30	2020-01-01	9
10	20	2020-01-01	10

-- Task 2: Select, Where, Between, AND, LIKE:

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

```
select concat(first_name," ",last_name) as name,email from customer;
```

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

```
select o.id,concat(c.first_name," ",c.last_name) as customer_name, o.order_date
from customer c,orders o
where c.id=o.customer_id;
```

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

```
insert into customer (first_name,last_name,email,phone,address)
values
('praveen','kumar','praveen@gmail.com','9360805402','porur');
```

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

```
update product
set price=price+(0.1*price);
```

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.

```
delete o,s
from order_details o,orders s
where s.id=o.orders_id and s.id=12;
```

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.

```
INSERT INTO orders (order_date,total_amount,customer_id) values
('2024/02/19',9000,11);
```

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.

```
update customer  set email='pradap@gmail.com',address='gandhi nagar' where id=12;
```

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.

```
update orders o
set total_amount=(select p.price*od.quantity from
product p join order_details od on p.id=od.product_id
where o.id=od.orders_id);
```

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.

```
alter table order_details
add constraint fk_deletion
foreign key (orders_id)
references orders(id)
on delete cascade;
delete from orders where id=14;
```

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.

```
INSERT INTO product (name, description, price) VALUES
('mac book', 'High-performance laptop with mac os', 150000);
alter table product
add category varchar(255);
update product
set category=case
when id=1 then 'gadget'
when id=2 then 'gadget'
```

```
when id=3 then 'i/o device'
when id=4 then 'gadget'
when id=5 then 'gadget'
when id=6 then 'i/o device'
when id=7 then 'gadget'
when id=8 then 'i/o device'
when id=9 then 'i/o device'
when id=10 then 'i/o device'
else 'gadget' end;
```

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.

```
alter table orders
add status varchar(255);
update orders
set status= case when id%2=0 then 'shipped' else 'pending' end;
update orders
set status='shipped' where id=2;
```

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.

```
alter table customer
add number_of_orders int;
update customer c
set number_of_orders=(select count(*)
from orders o
where c.id=o.customer_id);
```

-- Task 3. Aggregate functions, Having, Order By, GroupBy and Joins:

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

```
select concat(c.first_name," ",c.last_name) as
name,c.phone,c.email,o.order_date,o.total_amount,o.status
from customer c join orders o on
c.id=o.customer_id;
```

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

```
select p.name, sum(o.total_amount) as revenue from product p join order_details od on
p.id=od.product_id  join orders o on o.id=od.orders_id
group by p.id;
```

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

```
select concat(c.first_name," ",c.last_name),c.phone,c.email from customer c join orders o
on c.id=o.customer_id  group by c.id having count(c.id)>=1;
```

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.

```
select p.name,sum(od.quantity) as popular_gadget from product p join order_details od
on p.id=od.product_id  group by p.id order by popular_gadget desc limit 0,1;
```

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

```
select category,group_concat(name) as devices from product group by category;
```

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

```
select c.first_name,avg(o.total_amount) from customer c join orders o on
c.id=o.customer_id  group by c.id;
```

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

```
select o.id,c.*,o.total_amount from customer c join orders o on c.id=o.customer_id

having max(o.total_amount);
```

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

```
select p.name,count(p.id) as number_of_times_ordered from product p join order_details od on p.id=od.product_id  group by p.id;
```

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.

```
select p.name,group_concat(concat(c.first_name," ",c.last_name)) as customers from customer c join orders o on c.id=o.customer_id join order_details od on o.id=od.orders_id
join product p on p.id=od.product_id  group by p.id;
```

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.

```
select sum(total_amount) as total_revenue from orders where order_date between '2024-01-01' and '2024-12-31';
```

-- Task 4. Subquery and its type:

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

```
select concat(first_name," ",last_name) as customer from customer where id not in(select customer_id from orders);
```

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

```
select i.product_id,(i.quantityinstock- (select sum(od.quantity) from order_details od
where od.product_id=i.product_id)) as number_of_products_available_for_sale from inventory i;
```

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

```
select sum(total_amount) as toatal_revenue from (select total_amount from orders) as
revenue_by_techshop;
```

4. Write an SQL query to calculate the average quantity ordered for products in a specific category.

```
select p.category,(select avg(quantity) from order_details od where od.id in (select id from
product where category=p.category) as average_quantity_category from product p group by
p.category;
```

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.

```
select concat(c.first_name," ",c.last_name) as name , (select sum(o.total_amount) from
orders o where o.customer_id=c.id group by o.customer_id)as total_revenue
from customer c;
```

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.

```
select concat(c.first_name," ",c.last_name) as name , (select count(o.customer_id) from
orders o where o.customer_id=c.id group by o.customer_id)as order_count from customer c
order by order_count desc;
```

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.

```
select p.name , (select sum(od.quantity) from order_details od where p.id=od.product_id
group by od.product_id) as popular_product ,p.category from product p
order by popular_product desc limit 0,1;
```

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.

```
select concat(c.first_name," ",c.last_name) as most_money_spender, (select sum(o.total_amount) from orders o where c.id=o.customer_id group by o.customer_id) as money_spent from customer c order by money_spent desc limit 0,1;
```

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

```
select concat(c.first_name," ",c.last_name) as name, (select avg(o.total_amount) from orders o where o.customer_id=c.id group by o.customer_id) as average_order_value from customer c order by average_order_value desc;
```

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 concat(c.first_name," ",c.last_name) as name, (select count(o.customer_id) from orders o where o.customer_id=c.id group by o.customer_id) as total_number_of_orders from customer c order by total_number;
```