```
#Here we first create tables as per requirment


-- Create the customers table
CREATE TABLE
customers (
  customer_id INT PRIMARY KEY,
  name VARCHAR(255),
  age INT,
  gender
CHAR(1)
);

-- Insert 10 entries into the customers table
INSERT INTO customers (customer_id,
name, age, gender)
VALUES
  (1, 'John Doe', 30, 'M'),
  (2, 'Jane Doe', 25, 'F'),
  (3, 'Peter
Smith', 40, 'M'),
  (4, 'Mary Jones', 35, 'F'),
  (5, 'David Brown', 20, 'M'),
  (6, 'Susan
Green', 25, 'F'),
  (7, 'Michael Williams', 30, 'M'),
  (8, 'Sarah Johnson', 35, 'F'),
  (9,
'William Thomas', 20, 'M'),
  (10, 'Elizabeth White', 25, 'F');



-- Create the products
table
CREATE TABLE products (
  product_id INT PRIMARY KEY,
  name VARCHAR(255),
  category
VARCHAR(255),
  price INT
);

-- Insert 10 entries into the products table
INSERT INTO products
(product_id, name, category, price)
VALUES
  (1, 'Product 1', 'Category 1', 100),
  (2,
'Product 2', 'Category 2', 200),
  (3, 'Product 3', 'Category 3', 300),
  (4, 'Product 4',
'Category 4', 400),
  (5, 'Product 5', 'Category 5', 500),
  (6, 'Product 6', 'Category 6',
600),
  (7, 'Product 7', 'Category 7', 700),
  (8, 'Product 8', 'Category 8', 800),
  (9,
'Product 9', 'Category 9', 900),
  (10, 'Product 10', 'Category 10', 1000);

#Check if there is
any null value

select * from customers
where customer_id is null
or name is null or age is
null or gender is null

select * from products
where product_id is null
or name is null or
category is null or price is null
```

```
select * from inventory
where product_id is null
or
stock_count is null

select * from sales
where product_id is null
or transaction_id is null or
customer_id is null or product_id is null
or date is null or quantity is null

-- As we can
see there are no null values present in table

-- Check is there invalid value present
select *
from customers
where age>100 or gender !='M' or gender != 'F' or
customer_id<0

-- As we
can see, in customers table there are no unusal values

select * from products
where price<0
or product_id<0

-- As we can see, in products table there are no unusal values

select *
from inventory
where product_id<0 or stock_count<0

-- As we can see, in inventory table
there are no unusal values

select * from sales
where transaction_id<0 or customer_id<0
or product_id<0
or  CONVERT(date, date) IS NULL;

-- Here we can see that there are no
unusal values in this column.


-- 1. Calculate the total revenue generated by the company for
each product category.

select products.category, sum(amount) as total_revenue  from sales
join
products on products.product_id=sales.product_id
group by category
order by total_revenue
desc

--from results we can see that category 9 has highest revenue.

/*2. Determine the top 5
customers who have made the highest total purchases, considering
the customer's age and
gender.*/

select  distinct customers.customer_id,
customers.name,customers.age,customers.gender,
sum(amount) over (partition by customers.name)
as total_purchase from sales
join customers on customers.customer_id=sales.customer_id
order by
total_purchase desc
limit 5

#from above data we find top 5 customers who have made highest
purchases.
```

```sql
/*3. Analyze the inventory data and identify products that need restocking (stock
count less than a specified threshold).
 for our convenience, we consider that minimum stock
limit is 30% of total inventory of products */

with cte as(select product_id, stock_count,
round(stock_count*0.3) as minimum_stock_limit from inventory)

select * from cte
where
stock_count < minimum_stock_limit

#from above results, we can see that all products are
available in sufficient quantity

#4. Write a SQL query to calculate the average age of
customers for each product category.

select round(avg(age),1) as average_age , p.category
from
customers c join sales s on
c.customer_id=s.customer_id
left join  products        p on

p.product_id=s.product_id
group by p.category

#5. Write a SQL query to retrieve the top 3
product categories that have the highest average transaction amount.

select round(avg(age),1)
as average_age , p.category
from customers c join sales s on
c.customer_id=s.customer_id
left
join  products        p on
p.product_id=s.product_id
group by p.category

#6. Identify the most
profitable product category by calculating the average revenue per unit sold.

select
round(sum(amount)/sum(quantity)) as average_revenue   , p.category
from sales s
join products p
on p.product_id=s.product_id
group by p.category
order by average_revenue desc
```