MySQL Assignment

Instructions:

Answer all questions using MySQL.
Use appropriate subqueries, joins, and aggregate functions wherever applicable.
Make sure to use proper aliasing, GROUP BY, HAVING, DISTINCT, etc., as needed.
Data

```
-- Customers Table
CREATE TABLE Customers (
   CustomerID INT PRIMARY KEY,
   Name VARCHAR(100),
   City VARCHAR(100)
);

-- Orders Table
CREATE TABLE Orders (
   OrderID INT PRIMARY KEY,
   CustomerID INT,
   OrderDate DATE,
   Amount DECIMAL(10,2),
   FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
);

-- Products Table
CREATE TABLE Products (
   ProductID INT PRIMARY KEY,
   ProductName VARCHAR(100),
   Price DECIMAL(10,2)
);


-- OrderDetails Table
CREATE TABLE OrderDetails (
   OrderDetailID INT PRIMARY KEY,
   OrderID INT,
   ProductID INT,
   Quantity INT,
   FOREIGN KEY (OrderID) REFERENCES Orders(OrderID),
   FOREIGN KEY (ProductID) REFERENCES Products(ProductID)
);
```

Part A – Subqueries (20 marks)

1. Write a query to find customers who have placed orders in every month of the current year.

```
select c.name
from customers c
where not exists (
  select m.month
  from (
    select 1 as month union all select 2 union all select 3 union all
    select 4 union all select 5 union all select 6 union all
    select 7 union all select 8 union all select 9 union all
    select 10 union all select 11 union all select 12
  ) as m
  where not exists (
    select 1 from orders o
    where o.customerid = c.customerid
      and year(o.orderdate) = year(curdate())
      and month(o.orderdate) = m.month
  )
);
```

2. Retrieve the names of products that have been ordered more than the average quantity across all products.

```
select p.productname
from products p
join orderdetails od on p.productid = od.productid
group by p.productid
having sum(od.quantity) > (
  select avg(total_qty) from (
    select sum(quantity) as total_qty
    from orderdetails
    group by productid
  ) as avg_table
);
```

3. Find customers who have never ordered a product priced above ₹1000.

```
select name
from customers
where customerid not in (
  select distinct o.customerid
  from orders o
  join orderdetails od on o.orderid = od.orderid
  join products p on od.productid = p.productid
  where p.price > 1000
);
```

4. List the top 3 products by total revenue using a subquery.

```
select productname
from products
where productid in (
  select productid
  from (
    select od.productid, sum(od.quantity * p.price) as revenue
    from orderdetails od
    join products p on od.productid = p.productid
    group by od.productid
    order by revenue desc
    limit 3
  ) as top_products
);
```

5. Find orders that contain only one product using a correlated subquery.

```
select orderid
from orders o
where 1 = (
  select count(*) from orderdetails od
  where od.orderid = o.orderid
);
```