SQL TASK

CUSTOMERS TABLE:

```
create table customers (id int primary key auto_increment ,name char(50), email varchar(255),
address varchar(255) );
```

```
insert into customers (id, name, email, address) values
 ( 1, 'Rajesh Kumar', 'rajeshkumar@example.com', '12 MG Road, Bengaluru, Karnataka'),
(2, 'Priya Sharma', 'priyasharma@example.com', '45 Lajpat Nagar, New Delhi’),
 (3, 'Arun Nair', 'arunnair@example.com', '78 Marine Drive, Kochi, Kerala’),
 (4, 'Sneha Reddy', 'snehareddy@example.com', '23 Jubilee Hills, Hyderabad, Telangana'),
(5, 'Vikram Gupta', 'vikramgupta@example.com', '34 Park Street, Kolkata, West Bengal'),
(6, 'Deepa Mehta', 'deepamehta@example.com', '90 Anna Salai, Chennai, Tamil Nadu'),
(7, 'Kiran Desai', 'kirandesai@example.com', '56 FC Road, Pune, Maharashtra'),
(8, 'Amit Singh', 'amitsingh@example.com', '67 Hazratganj, Lucknow, Uttar Pradesh'),
(9, 'Pooja Joshi', 'poojajoshi@example.com', '14 MG Marg, Dehradun, Uttarakhand'),
(10, 'Ravi Patel', 'ravipatel@example.com', '89 Ring Road, Ahmedabad, Gujarat’);
```

ORDERS TABLE:

```
create table orders (id int primary key auto_increment, customer_id int not null, order_date date,
total_amount int);
```

```
insert into orders (id, customer_id, order_date, total_amount) values
    (1, 1, '2024-01-01', 1500),
   (2, 2, '2024-01-05', 2000),
    (3, 3, '2024-01-10', 1200),
    (4, 4, '2024-01-15', 2500),
    (5, 5, '2024-01-20', 1800);
```

PRODUCTS TABLE:

```
create table products (id int primary key auto_increment,name char(50), price int, description
varchar(255));
```

```
insert into products (id, name, price, description) values
(1, 'Laptop', 500, '15-inch, 8GB RAM, 256GB SSD’),
(2, 'Smartphone', 200, '6.5-inch screen, 128GB storage, dual-camera’),
 (3, 'Headphones', 30, 'Over-ear, noise-canceling, Bluetooth’),
 (4, 'Washing Machine', 150, '7kg capacity, front-load, energy-efficient’),
 (5, 'Air Conditioner', 350, '1.5 Ton, split AC, inverter technology’),
 (6, 'Refrigerator', 250, 'Double-door, 300L capacity, frost-free’),
 (7, 'Microwave Oven', 100, '20L capacity, convection, auto-cook menu’),
 (8, 'Smartwatch', 80, 'Fitness tracker, heart-rate monitor, water-resistant’),
 (9, 'Camera', 400, 'DSLR, 24MP, Wi-Fi-enabled'),
(10, 'Gaming Console', 450, '4K HDR, 1TB storage, wireless controllers’);
```

ORDER_DETAILS TABLE:

create table order_details (id int primary key auto_increment, order_id int not null, product_id int not null, quantity int default 1, foreign key (order_id) references orders(id), foreign key (product_id) references products(id));

INSERT INTO order_details (order_id, product_id, quantity) VALUES (1, 1, 1),
(1, 3, 2),(2, 2, 1),(2, 8, 1),(3, 6, 1),(3, 7, 1),(4, 5, 1),(5, 4, 1),(5, 9, 1);

SOLUTIONS:

- Retrieve all customers who have placed an order in the last 30 days

select name,order_date,total_amount from customers inner join orders on customers.id = orders.customer_id where order_date <= “2024-01-15”;

The above query combines customers and orders table , there by display all customers who placed order in 30 days

- Retrieve the average total of all orders.

select avg(total_amount) as Total_avg from orders;

The above query takes avg of all the order’s amount.

- Get the total amount of all orders placed by each customer.

select Name ,sum(total_amount) as Each_total from customers inner join orders on customers.id = orders.customer_id group by customer_id;

The   above gets the total amount of all orders placed by each customer.

- Update the price of Product C to 45.00.

```
update products set price=45 where id = 3;
```

The above query Update the price of Product headphones to 45.00

- Add a new column discount to the products table.

```
alter table products add discount int default 10;
```

The above query adds a new column discount to the products table.

- Retrieve the top 3 products with the highest price.

```
select name , price from products order by price desc limit 3;
```

The above query retrieves the top 3 products with the highest price.

- Join the orders and customers tables to retrieve the customer's name and order date for each order.

```
select name, order_date from customers join orders on customers.id = orders.customer_id;
```

The above query joins the orders and customers tables to retrieve the customer's name and order date for each order.

- Retrieve the orders with a total amount greater than 150.00.

The below query retrieves the orders with a total amount greater than 150.00.

```
select * from orders where total_amount > 150;
```