

**Name – Ishaan Gupta**

**Batch – c# batch 2**

**Lab Excercise - Data Selection**

***(question+SQL query+output added below)***

## **Table Structure -**

DataSelectionLab.s...SS.master (sa (62))

-- Ishaan Gupta

```
CREATE DATABASE DataSelectionLab;
USE DataSelectionLab;

CREATE TABLE Categories (
    category_id INT PRIMARY KEY,
    category_name VARCHAR(50) NOT NULL
);

CREATE TABLE Products (
    product_id INT PRIMARY KEY IDENTITY(1,1),
    product_name VARCHAR(100) NOT NULL,
    category_id INT,
    list_price DECIMAL(10, 2) NOT NULL,
    model_year INT,
    FOREIGN KEY (category_id) REFERENCES Categories(category_id)
);

CREATE TABLE Customers (
    customer_id INT PRIMARY KEY IDENTITY(1,1),
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    city VARCHAR(50),
    state VARCHAR(50)
);

CREATE TABLE Orders (
    order_id INT PRIMARY KEY IDENTITY(1,1),
    customer_id INT,
    order_date DATE NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

**Q1) Write a query to display customer list by the first name in descending order.**

```
SELECT first_name, last_name FROM Customers ORDER BY first_name DESC;
```

100 %

Results Messages

	first_name	last_name
1	Sunita	Gupta
2	Raj	Sharma
3	Priya	Patel
4	Anand	Kumar
5	Amit	Singh

**Q2) Write a query to display the first name, last name, and city of the customers. It sorts the customer list by the city first and then by the first name.**

SELECT first\_name, last\_name, city FROM Customers ORDER BY city, first\_name;

100 %

Results

Messages

	first_name	last_name	city
1	Priya	Patel	Ahmedabad
2	Anand	Kumar	Chennai
3	Amit	Singh	Delhi
4	Sunita	Gupta	Jaipur
5	Raj	Sharma	Mumbai

**Q3) Write a query to returns the top three most expensive products.**

```
SELECT * FROM products ORDER BY list_price DESC OFFSET 0 ROWS FETCH NEXT 3 ROWS ONLY;
```

100 %

Results Messages

	product_id	product_name	category_id	list_price	model_year
1	6	Diamond Necklace	7	11999.99	2022
2	4	Refrigerator	5	4999.99	2018
3	2	Laptop	1	2599.99	2019

**Q4) Write a query to finds the products whose list price is greater than 300 and model year is 2018.**

SELECT product\_name, list\_price, model\_year FROM Products WHERE list\_price > 300 AND model\_year = 2018;

100 %

Results

Messages

	product_id	product_name	category_id	list_price	model_year
1	6	Diamond Necklace	7	11999.99	2022
2	4	Refrigerator	5	4999.99	2018
3	2	Laptop	1	2599.99	2019

**Q5) Write a query to finds products whose list price is greater than 3,000 or model year is 2018. Any product that meets one of these conditions is included in the result set.**

SELECT product\_name, list\_price, model\_year FROM Products WHERE list\_price > 3000 OR model\_year = 2018;

100 %

Results

Messages

	product_name	list_price	model_year
1	Smartphone	489.99	2018
2	Refrigerator	4999.99	2018
3	Diamond Necklace	11999.99	2022

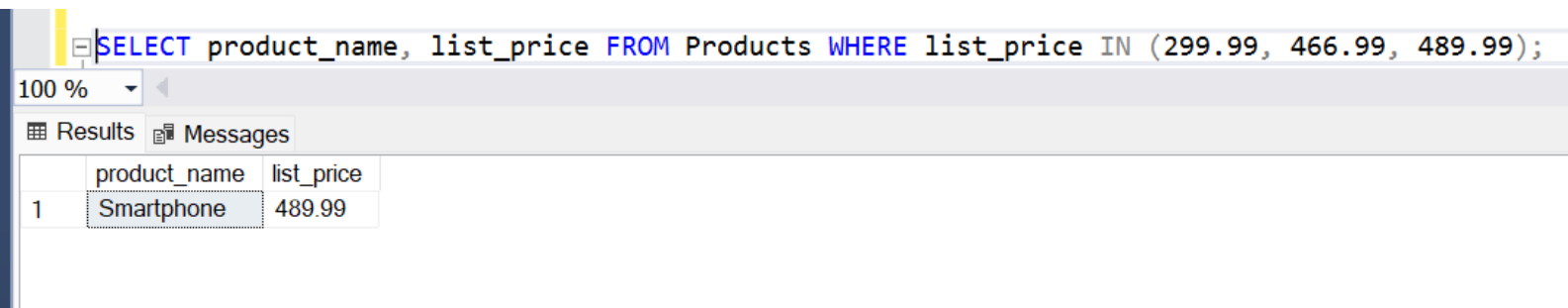
**Q6) Write a query to find the products whose list prices are between 1,899 and 1,999.99.**

```
SELECT product_name, list_price FROM Products WHERE list_price BETWEEN 1899 AND 1999.99;
```

100 %	◀
Results	Messages
product_name	list_price



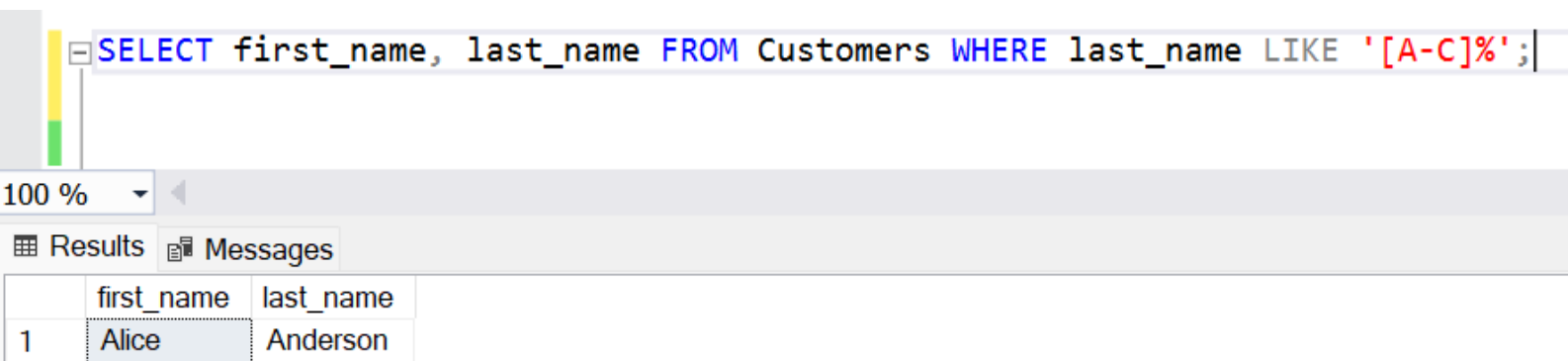
**Q7. Write a query uses the [IN](#) operator to find products whose list price is 299.99 or 466.99 or 489.99**



The screenshot shows a SQL query execution interface. At the top, a text box contains the query: `SELECT product_name, list_price FROM Products WHERE list_price IN (299.99, 466.99, 489.99);`. Below the text box is a toolbar with a '100 %' zoom level and a left arrow. Underneath the toolbar are two tabs: 'Results' (active) and 'Messages'. The 'Results' tab displays a table with two columns: 'product\_name' and 'list\_price'. The table contains one row with the values 'Smartphone' and '489.99'.

	product_name	list_price
1	Smartphone	489.99

**Q8. Write a query to the customers where the first character in the last name is the letter in the range A through C**



The screenshot shows a SQL query editor with the following query: `SELECT first_name, last_name FROM Customers WHERE last_name LIKE '[A-C]%' ;`. Below the editor, there is a toolbar with a zoom level of 100% and a left arrow. Two tabs are visible: "Results" and "Messages". The "Results" tab is active, displaying a table with two columns: "first\_name" and "last\_name". The first row of the table contains the values "Alice" and "Anderson".

	first_name	last_name
1	Alice	Anderson

**Q9) Write a query using NOT LIKE operator to find customers where the first character in the first name is not the letter A**

```
SELECT first_name, last_name FROM Customers  
WHERE first_name NOT LIKE 'A%';
```

100 %

Results Messages

	first_name	last_name
1	Raj	Sharma
2	Priya	Patel
3	Sunita	Gupta

**Q10) Write a query to return the number of customers by state and city group state and city.**

```
SELECT state, city, COUNT(*) as customer_count  
FROM Customers  
GROUP BY state, city;
```

100 %

Results Messages

	state	city	customer_count
1	Gujarat	Ahmedabad	1
2	Tamil Nadu	Chennai	1
3	Delhi	Delhi	1
4	Rajasthan	Jaipur	1
5	Maharashtra	Mumbai	1
6	NY	New York	1

**Q11. Write a query to return the number of orders placed by the customer group by customer id and year.**

```
SELECT customer_id, YEAR(order_date) as order_year, COUNT(*) as order_count
FROM Orders
GROUP BY customer_id, YEAR(order_date);
```

100 %

Results Messages

	customer_id	order_year	order_count
1	1	2023	1
2	2	2023	1
3	3	2023	1
4	4	2023	1
5	5	2023	1

**Q12) Write query to finds the maximum and minimum list group by category id. Then, it filters out the category which has the maximum list price greater than 4,000 or the minimum list price less than 500.**

```
SELECT c.category_id,  
       MAX(p.list_price) as max_price,  
       MIN(p.list_price) as min_price  
FROM Categories c  
JOIN Products p ON c.category_id = p.category_id  
GROUP BY c.category_id, c.category_name  
HAVING MAX(p.list_price) >= 4000 OR MIN(p.list_price) <= 500;
```

100 %

Results Messages

	category_id	max_price	min_price
1	1	2599.99	489.99
2	3	250.99	250.99
3	5	4999.99	4999.99
4	6	379.99	379.99
5	7	11999.99	11999.99