Internship Program: Soulvibe.Tech

“Farmer & Market Insights Using SQL”

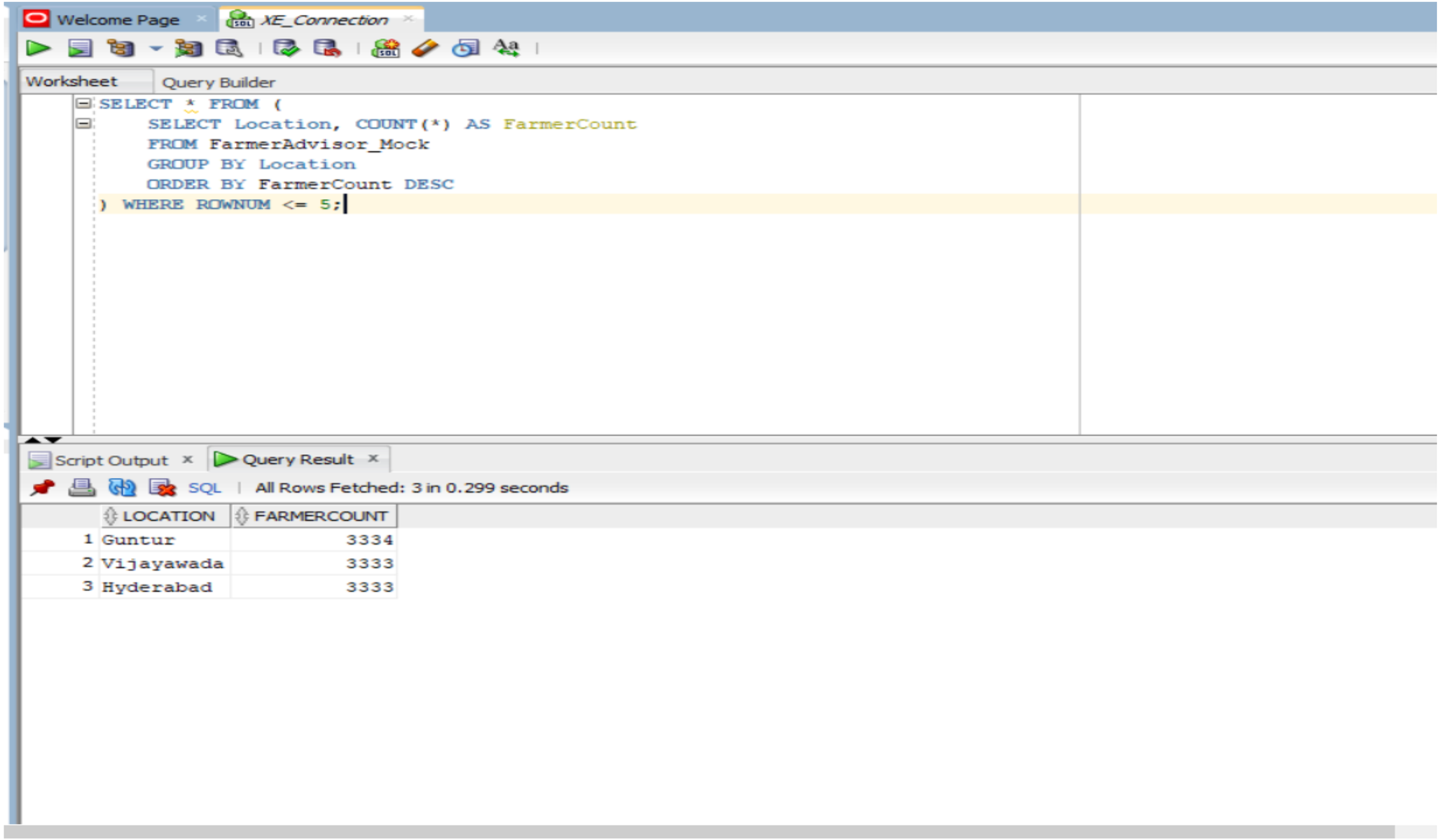
Batch Name: SVT/DAINT/2025/07-B11

Introduction

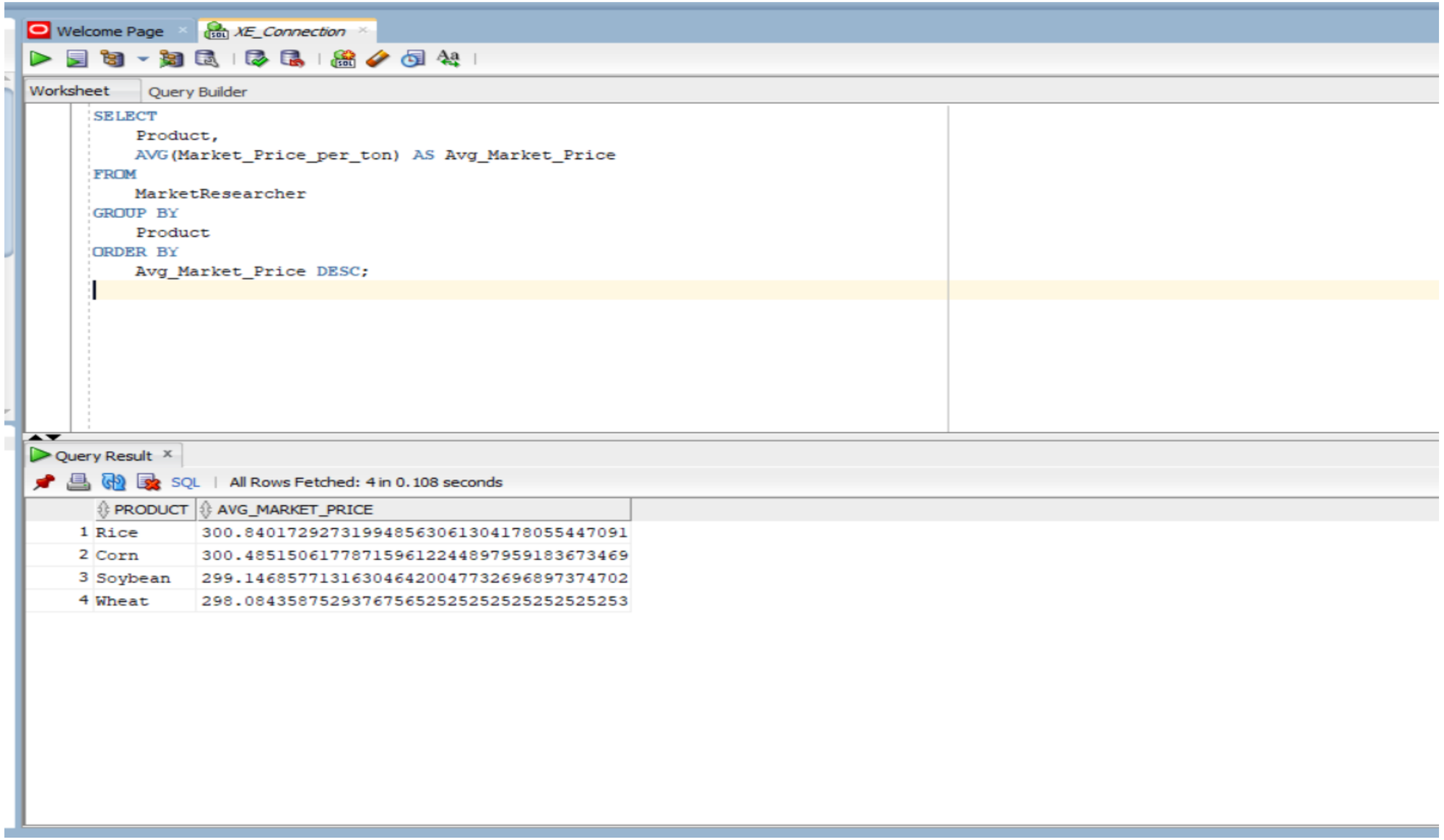
Overview of the main objectives

In this task, I was assigned to analyze agricultural data using SQL with a focus on two key tables: FarmerAdvisor and MarketResearcher. The objective was to extract meaningful insights related to crop performance, farmer behavior, and market dynamics. Using SQL, I performed various operations such as filtering, grouping, joining tables, and applying window functions to answer specific business questions. These included identifying high-growth crops, ranking profitability, tracking market price changes, and understanding patterns in farmer-crop relationships. The goal was to simulate real-world data analysis scenarios to support better decision-making in agriculture and advisory services by leveraging structured query logic and analytical thinking.

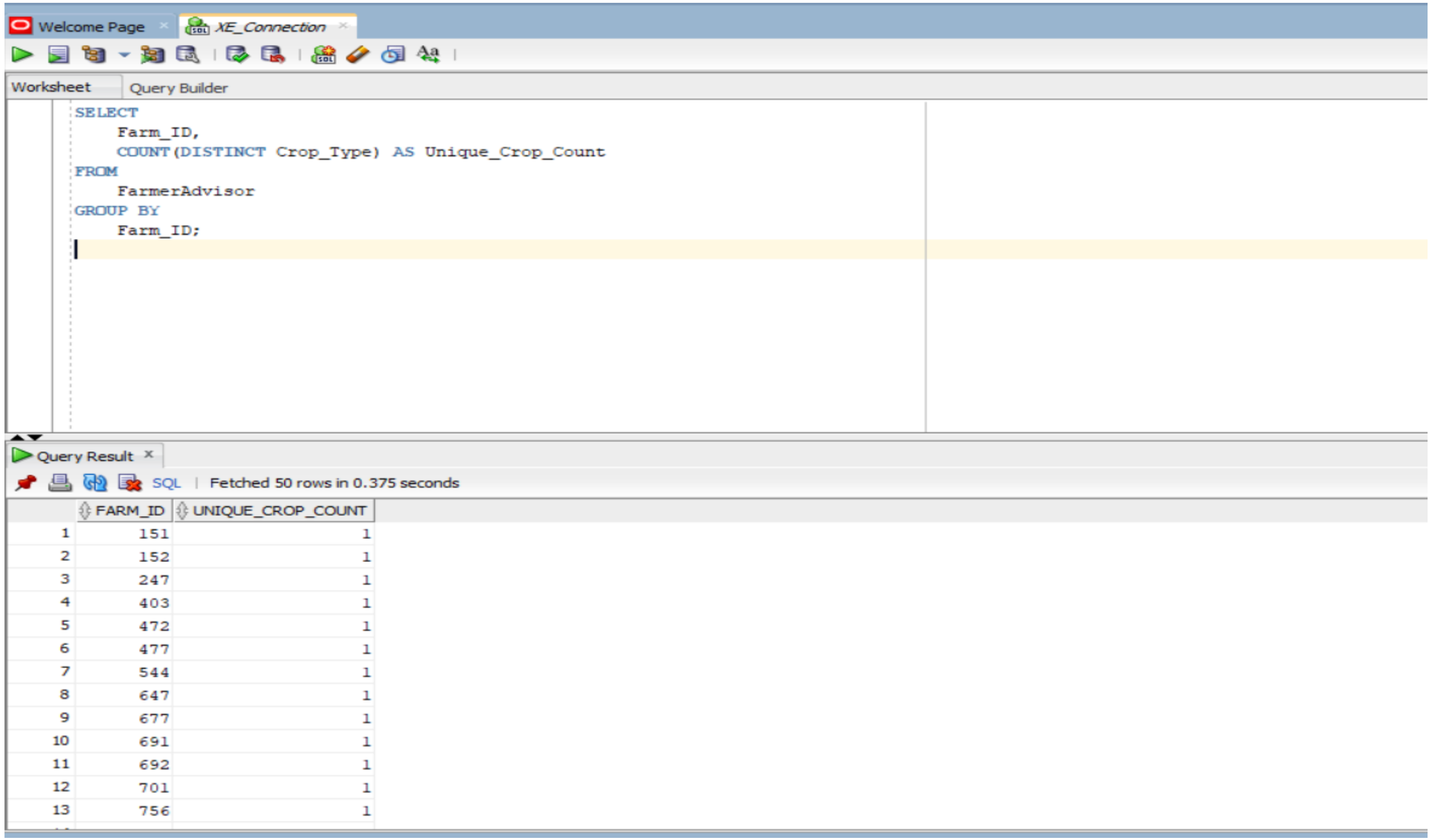
1 Top 5 locations where the maximum number of farmers with advisors.



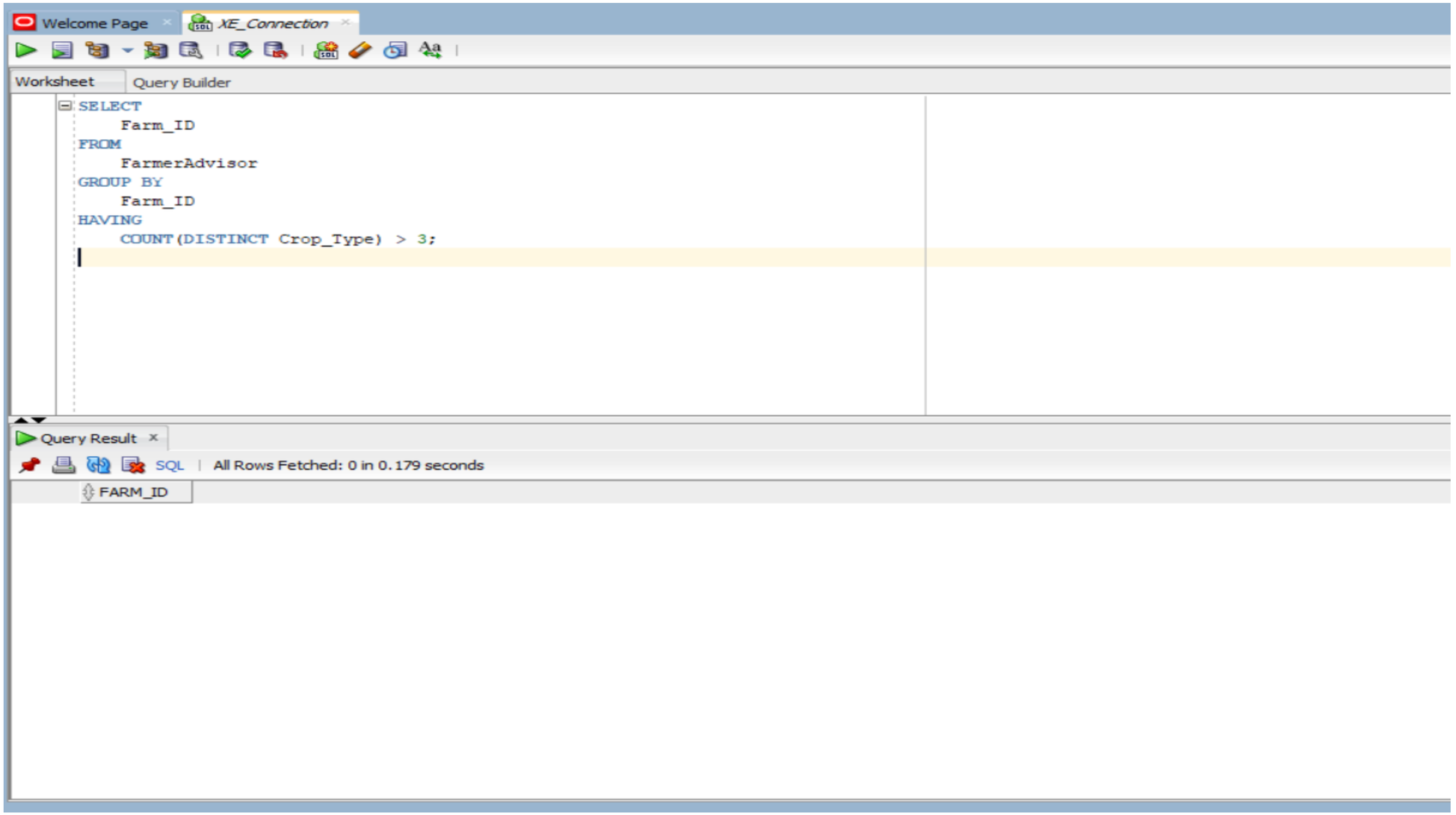
2 The average market price and sort in descending order.



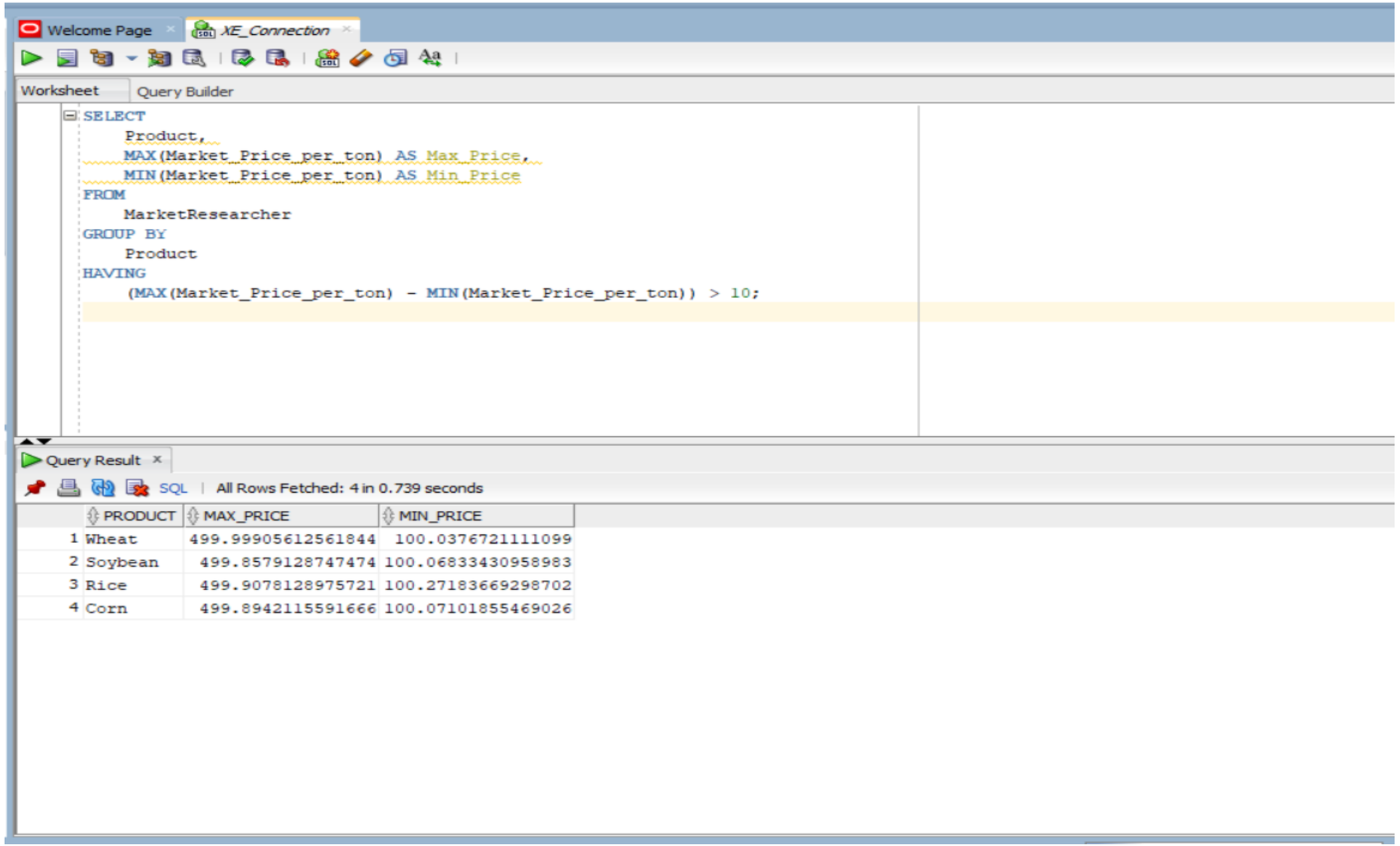
3 Count how many unique crops each farmer is associated with.



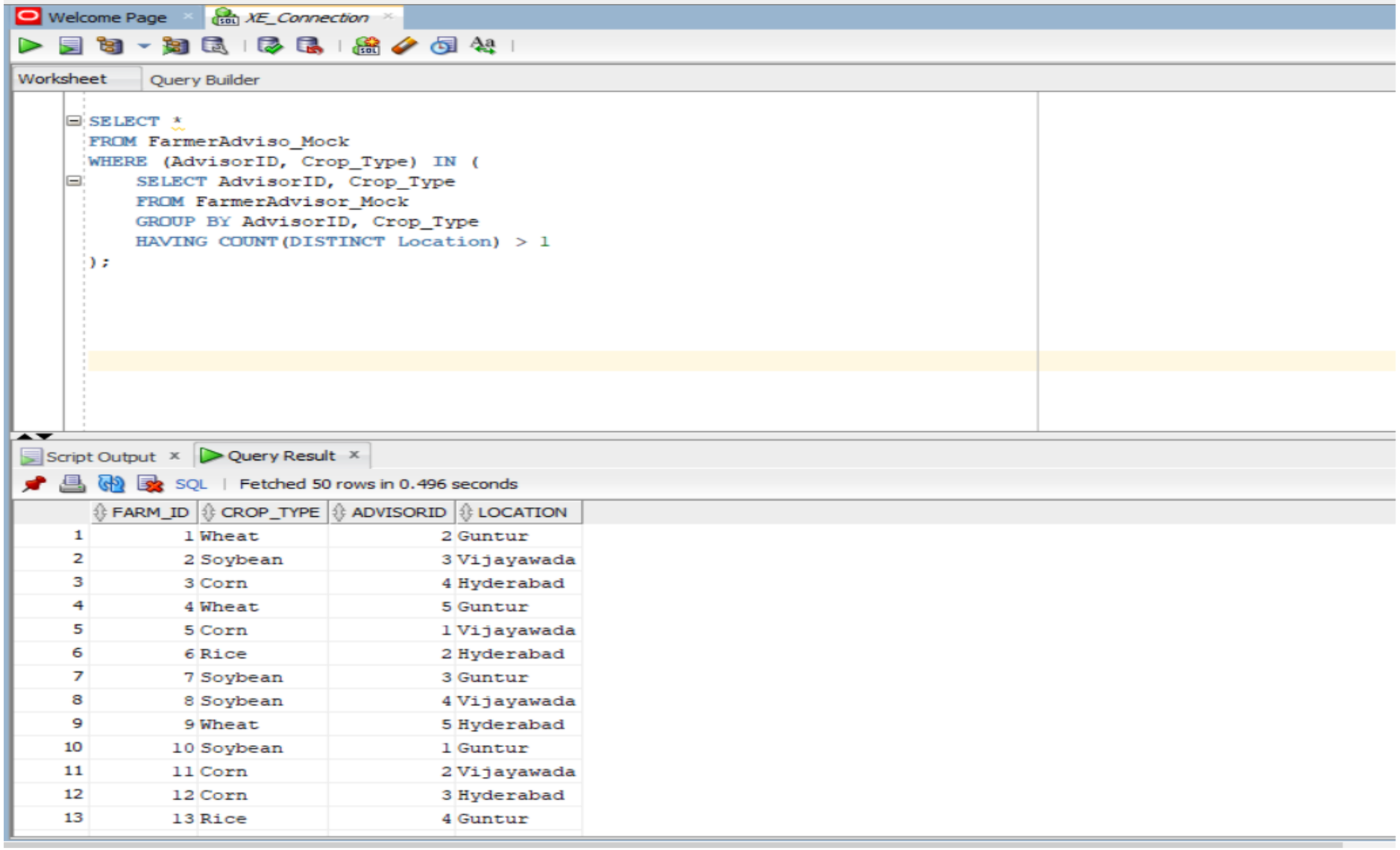
4 Farmers who are growing more than 3 different types of crops.



5 Crops where the max and min market prices differ by more than ₹10 per kg.



6 All advisors who guide farmers growing the same crop in different districts.



	FARM_ID	CROP_TYPE	ADVISORID	LOCATION
1	1	Wheat	2	Guntur
2	2	Soybean	3	Vijayawada
3	3	Corn	4	Hyderabad
4	4	Wheat	5	Guntur
5	5	Corn	1	Vijayawada
6	6	Rice	2	Hyderabad
7	7	Soybean	3	Guntur
8	8	Soybean	4	Vijayawada
9	9	Wheat	5	Hyderabad
10	10	Soybean	1	Guntur
11	11	Corn	2	Vijayawada
12	12	Corn	3	Hyderabad
13	13	Rice	4	Guntur

7 Rank crops by profit per unit using RANK().

Welcome Page | XE_Connection

Worksheet | Query Builder

```
SELECT
    MR.PRODUCT AS CROP_NAME,
    ROUND(MR.MARKET_PRICE_PER_TON - FA.AVG_COST, 2) AS PROFIT_PER_UNIT,
    RANK() OVER (ORDER BY (MR.MARKET_PRICE_PER_TON - FA.AVG_COST) DESC) AS PROFIT_RANK
FROM
    MARKETRESEARCHER MR
JOIN (
    SELECT
        CROP_TYPE,
        ROUND(AVG(NVL(FERTILIZER_USAGE_KG, 0) + NVL(PESTICIDE_USAGE_KG, 0)), 2) AS AVG_COST
    FROM
        FARMERADVISOR
    GROUP BY
        CROP_TYPE
) FA
ON
    MR.PRODUCT = FA.CROP_TYPE;
```

Script Output | Query Result

SQL | Fetched 50 rows in 40.931 seconds

	CROP_NAME	PROFIT_PER_UNIT	PROFIT_RANK
1	Soybean	365.11	1
2	Soybean	364.98	2
3	Soybean	364.79	3
4	Soybean	364.51	4
5	Soybean	364.44	5
6	Soybean	364.39	6
7	Soybean	364.38	7
8	Soybean	364.32	8
9	Soybean	364.17	9
10	Rice	364.11	10
11	Rice	363.99	11
12	Soybean	363.99	12
13	Wheat	363.87	13

8 Locations where the current market price of a crop is more than 20% above the average price of that crop across all locations.

Welcome Page | XE_Connection

Worksheet | Query Builder

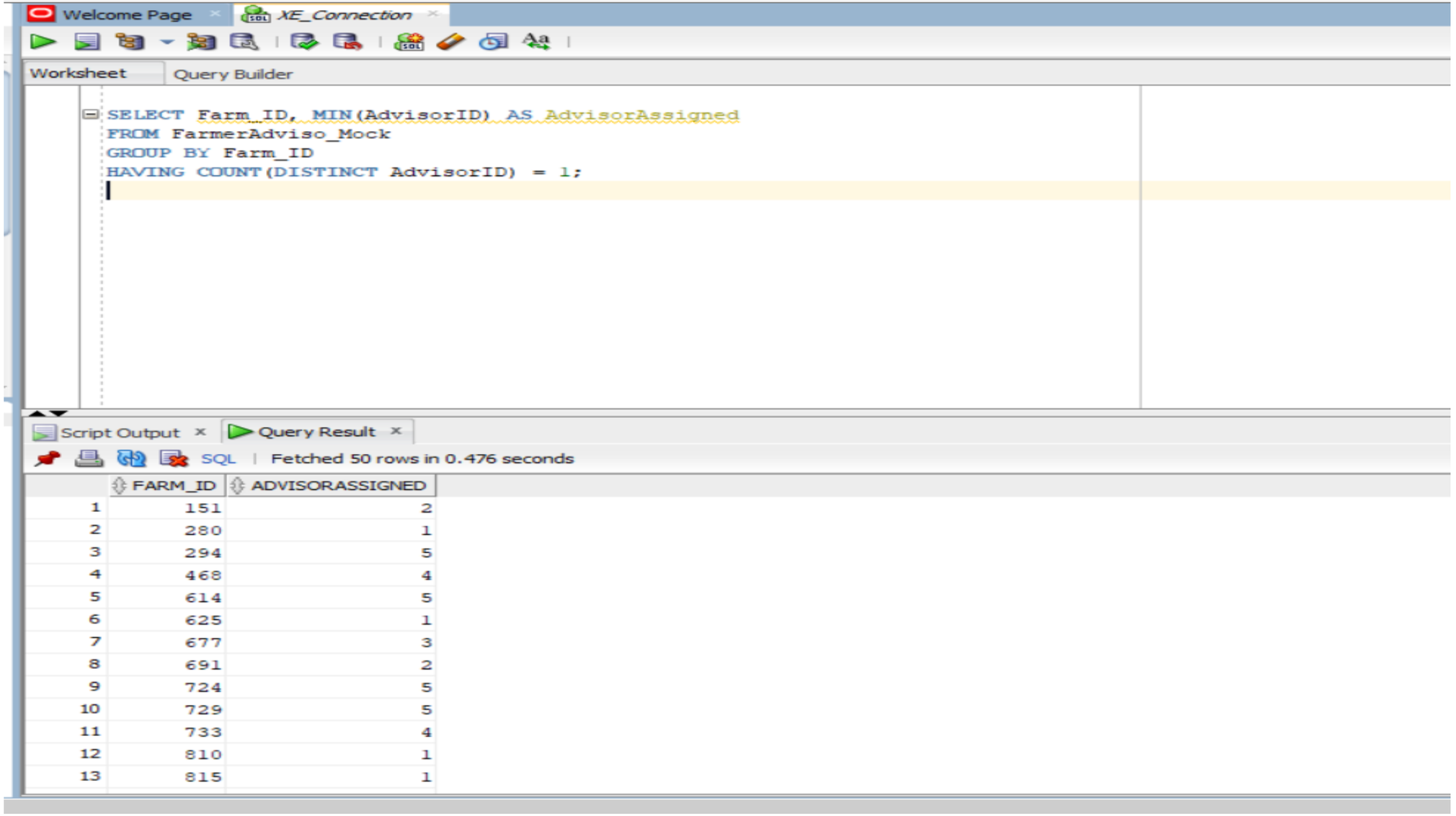
```
WITH AvgPrice AS (
    SELECT Product, AVG(Market_Price_per_ton) AS Avg_Price
    FROM MarketResearcher
    GROUP BY Product
)
SELECT m.*
FROM MarketResearcher m
JOIN AvgPrice a ON m.Product = a.Product
WHERE m.Market_Price_per_ton > a.Avg_Price * 1.2;
```

Script Output | Query Result

SQL | Fetched 50 rows in 0.185 seconds

	MARKET_ID	PRODUCT	MARKET_PRICE_PER_TON	DEMAND_INDEX	SUPPLY_INDEX	COMPETITOR_PRICE_PER_TON	ECONOMIC_INDICATOR	WEATHER_IMPACT_SCORE	SEASONAL
1	2	Rice	420.52796955955307	188.45240008580845	150.789482977595	492.097797644394	0.5263068815761089	70.97806264396277	High
2	3	Wheat	457.2603975211701	171.17938395811063	78.98932630036724	323.00334230465387	1.29239334879865	80.8535915904238	Low
3	6	Corn	389.6482051471298	123.08872887995155	190.90647508558038	181.9354392180609	0.8761502603945046	81.36985531316705	Low
4	9	Soybean	447.4117730632625	188.31754490852956	171.82949278671953	210.50295551774195	0.7078028072550187	29.150676979735856	Low
5	14	Soybean	481.94841517479534	164.5570086363353	117.89743276774203	315.894413578798	0.996883786192946	81.29448800288094	Medium
6	16	Corn	400.6447487209915	192.60030048359053	68.85381701077962	268.40720091667856	0.8406059954568866	98.0521766129821	Medium
7	20	Soybean	468.08534637097773	64.18598493416194	108.30769511092905	375.33084499206666	0.9957607808823131	67.39407678002567	Low
8	21	Soybean	404.3796066798186	170.12732957924175	198.9322484582829	367.5274051914306	1.195187014934815	72.08959196466841	High
9	22	Wheat	444.2102233468652	132.45943830884892	85.20097918199315	355.1809845681516	1.2960913631131243	8.891188062752187	Low
10	24	Wheat	431.8034792818116	101.57379387931167	62.578446491784575	393.4910002637361	0.8898999419714463	5.924634912520432	High
11	27	Soybean	443.2644718458053	176.67465539540592	163.34341019096973	489.31539884979804	0.6639799275587567	23.220849964990066	High
12	33	Rice	426.99328668626333	66.85687441609332	73.33422299655902	166.84376039274733	1.4116215192869785	36.302470441482 78	High
13	35	Corn	419.87995300664267	187.06816235053117	164.57811954468576	382.46982659607687	1.4104332604287757	26.963045337249	

9 Farmers who have been assigned the same advisor for all their crops.



	FARM_ID	ADVISORASSIGNED
1	151	2
2	280	1
3	294	5
4	468	4
5	614	5
6	625	1
7	677	3
8	691	2
9	724	5
10	729	5
11	733	4
12	810	1
13	815	1

10. CTE showing average profit per crop type by farmer and use it to list farmers making below-average profits on any crop.

Welcome Page | XE_Connection

Worksheet | Query Builder

```
WITH ProfitPerCrop AS (
    SELECT
        Farm_ID,
        Crop_Type,
        Crop_Yield_ton,
        Fertilizer_Usage_kg,
        Pesticide_Usage_kg,
        (Crop_Yield_ton - (Fertilizer_Usage_kg + Pesticide_Usage_kg)) AS Profit
    FROM FarmerAdvisor
),
AvgProfitByFarmerCrop AS (
    SELECT
        Farm_ID,
        Crop_Type,
        AVG(Profit) AS Avg_Profit
    FROM ProfitPerCrop
    GROUP BY Farm_ID, Crop_Type
)
SELECT
    p.Farm_ID,
    p.Crop_Type,
    p.Profit,
    a.Avg_Profit
FROM ProfitPerCrop p
JOIN AvgProfitByFarmerCrop a
  ON p.Farm_ID = a.Farm_ID AND p.Crop_Type = a.Crop_Type
WHERE p.Profit < a.Avg_Profit;
```

Script Output | Query Result

SQL | All Rows Fetched: 0 in 0.335 seconds

FARM_ID	CROP_TYPE	PROFIT	AVG_PRO...

11 Use a window function to find the price change of each crop over the last 3 entries

Welcome Page | XE_Connection

Worksheet | Query Builder

```
SELECT
    curr.Crop_Type,
    curr.PriceDate,
    curr.Market_Price_per_ton AS Current_Price,
    NVL(curr.Market_Price_per_ton - prev1.Market_Price_per_ton, 0) AS Change_1,
    NVL(curr.Market_Price_per_ton - prev2.Market_Price_per_ton, 0) AS Change_2,
    NVL(curr.Market_Price_per_ton - prev3.Market_Price_per_ton, 0) AS Change_3
FROM
    MarketResearcher_Mock curr
LEFT JOIN
    MarketResearcher_Mock prev1
    ON curr.Crop_Type = prev1.Crop_Type AND prev1.PriceDate = curr.PriceDate - 1
LEFT JOIN
    MarketResearcher_Mock prev2
    ON curr.Crop_Type = prev2.Crop_Type AND prev2.PriceDate = curr.PriceDate - 2
LEFT JOIN
    MarketResearcher_Mock prev3
    ON curr.Crop_Type = prev3.Crop_Type AND prev3.PriceDate = curr.PriceDate - 3
ORDER BY
    curr.Crop_Type, curr.PriceDate;
```

Script Output | Query Result

SQL | Fetched 100 rows in 0.846 seconds

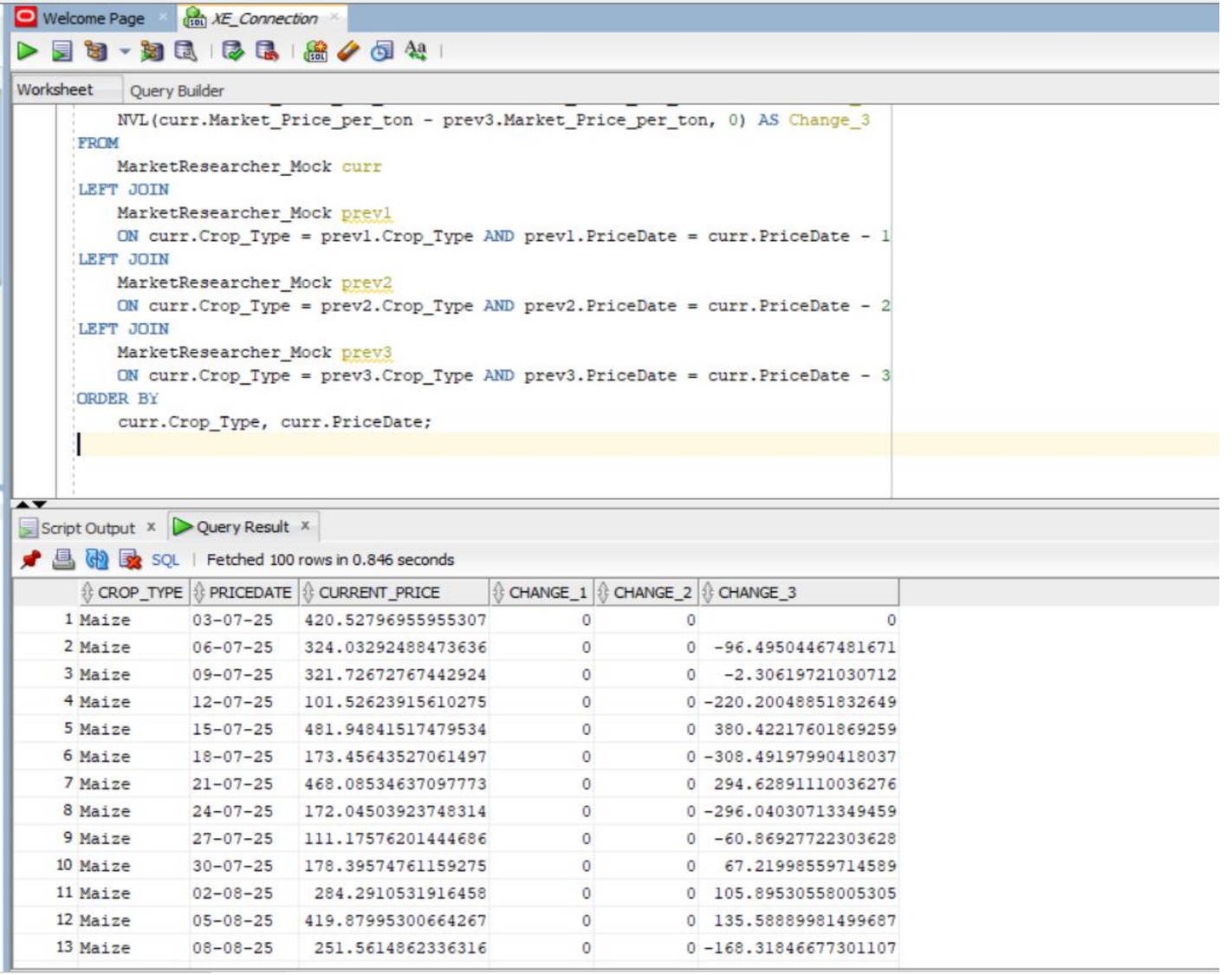
	CROP_TYPE	PRICEDATE	CURRENT_PRICE	CHANGE_1	CHANGE_2	CHANGE_3
1	Maize	03-07-25	420.52796955955307	0	0	0
2	Maize	06-07-25	324.03292488473636	0	0	-96.49504467481671
3	Maize	09-07-25	321.72672767442924	0	0	-2.30619721030712
4	Maize	12-07-25	101.52623915610275	0	0	-220.20048851832649
5	Maize	15-07-25	481.94841517479534	0	0	380.42217601869259
6	Maize	18-07-25	173.45643527061497	0	0	-308.49197990418037
7	Maize	21-07-25	468.08534637097773	0	0	294.62891110036276
8	Maize	24-07-25	172.04503923748314	0	0	-296.04030713349459
9	Maize	27-07-25	111.17576201444686	0	0	-60.86927722303628
10	Maize	30-07-25	178.39574761159275	0	0	67.21998559714589
11	Maize	02-08-25	284.2910531916458	0	0	105.89530558005305
12	Maize	05-08-25	419.87995300664267	0	0	135.58889981499687
13	Maize	08-08-25	251.5614862336316	0	0	-168.31846677301107

Welcome Page | XE_Connection

Worksheet | Query Builder

Script Output | Query Result

SQL | Fetched 100 rows in 0.846 seconds



12 Create a new column that classifies crop growth Count the number of crops in each category.

Welcome Page | XE_Connection

Worksheet | Query Builder

```
SELECT
    CASE
        WHEN ((DEMAND_INDEX - SUPPLY_INDEX) / SUPPLY_INDEX) * 100 < 20 THEN 'Low'
        WHEN ((DEMAND_INDEX - SUPPLY_INDEX) / SUPPLY_INDEX) * 100 BETWEEN 20 AND 50 THEN 'Medium'
        WHEN ((DEMAND_INDEX - SUPPLY_INDEX) / SUPPLY_INDEX) * 100 > 50 THEN 'High'
    END AS GrowthCategory,
    COUNT(*) AS CropCount
FROM
    MarketResearcher
GROUP BY
    CASE
        WHEN ((DEMAND_INDEX - SUPPLY_INDEX) / SUPPLY_INDEX) * 100 < 20 THEN 'Low'
        WHEN ((DEMAND_INDEX - SUPPLY_INDEX) / SUPPLY_INDEX) * 100 BETWEEN 20 AND 50 THEN 'Medium'
        WHEN ((DEMAND_INDEX - SUPPLY_INDEX) / SUPPLY_INDEX) * 100 > 50 THEN 'High'
    END;
```

Script Output | Query Result

SQL | All Rows Fetched: 3 in 0.854 seconds

	GROWTHCATEGORY	CROPCOUNT
1	Medium	1269
2	Low	6410
3	High	2321

13 Join the tables and display all farmers whose crop is sold in the same district at the highest price.

Welcome Page | XE_Connection

Worksheet | Query Builder

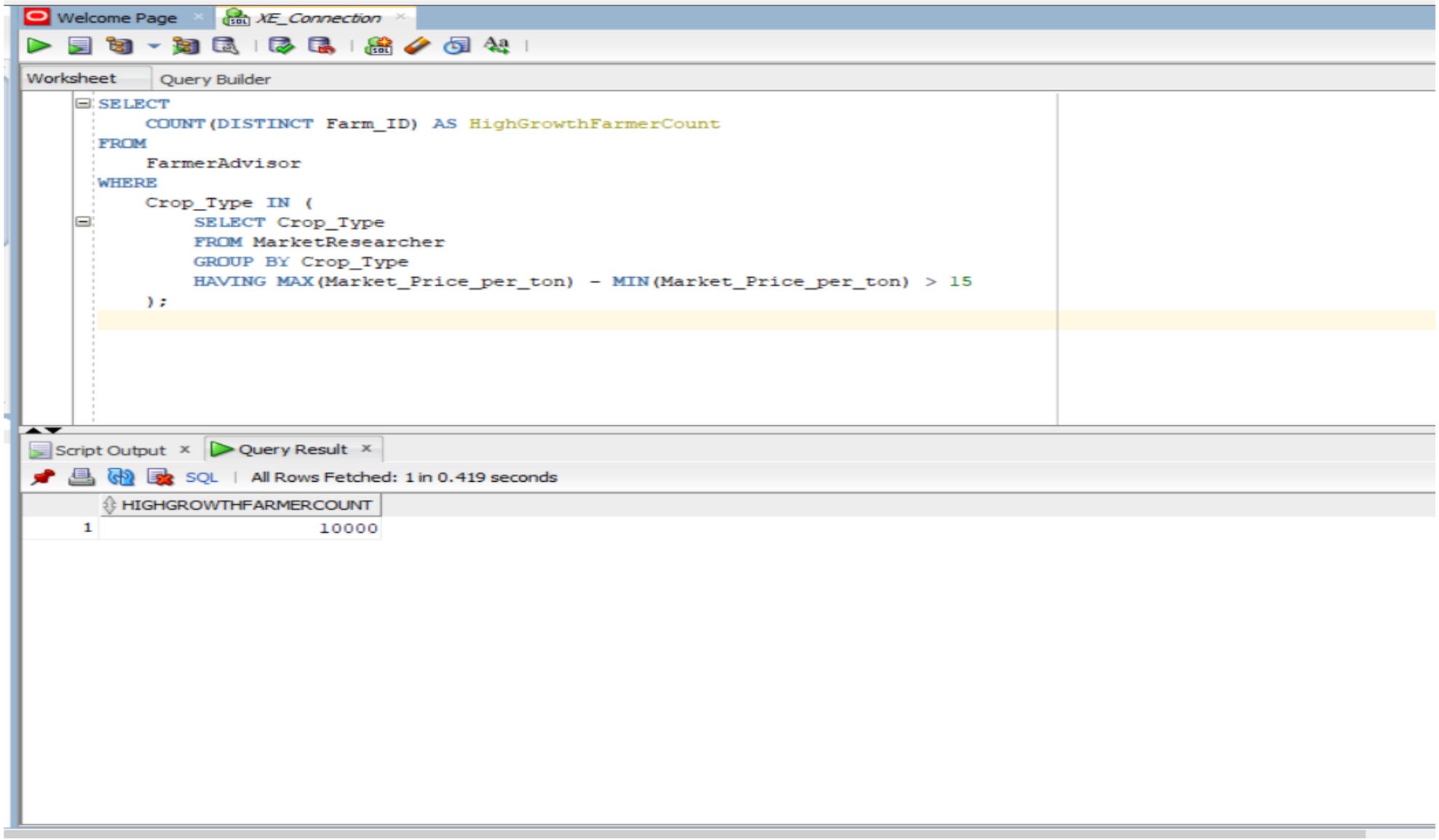
```
WITH MaxPricePerCrop AS (
    SELECT
        PRODUCT,
        MAX(MARKET_PRICE_PER_TON) AS MaxPrice
    FROM
        MarketResearcher
    GROUP BY
        PRODUCT
)

SELECT
    f.FARM_ID,
    f.CROP_TYPE,
    f.CROP_YIELD_TON,
    m.MARKET_PRICE_PER_TON
FROM
    FarmerAdvisor f
JOIN
    MarketResearcher m
    ON f.CROP_TYPE = m.PRODUCT
JOIN
    MaxPricePerCrop mp
    ON mp.PRODUCT = m.PRODUCT
   AND mp.MaxPrice = m.MARKET_PRICE_PER_TON;
```

Script Output | Query Result

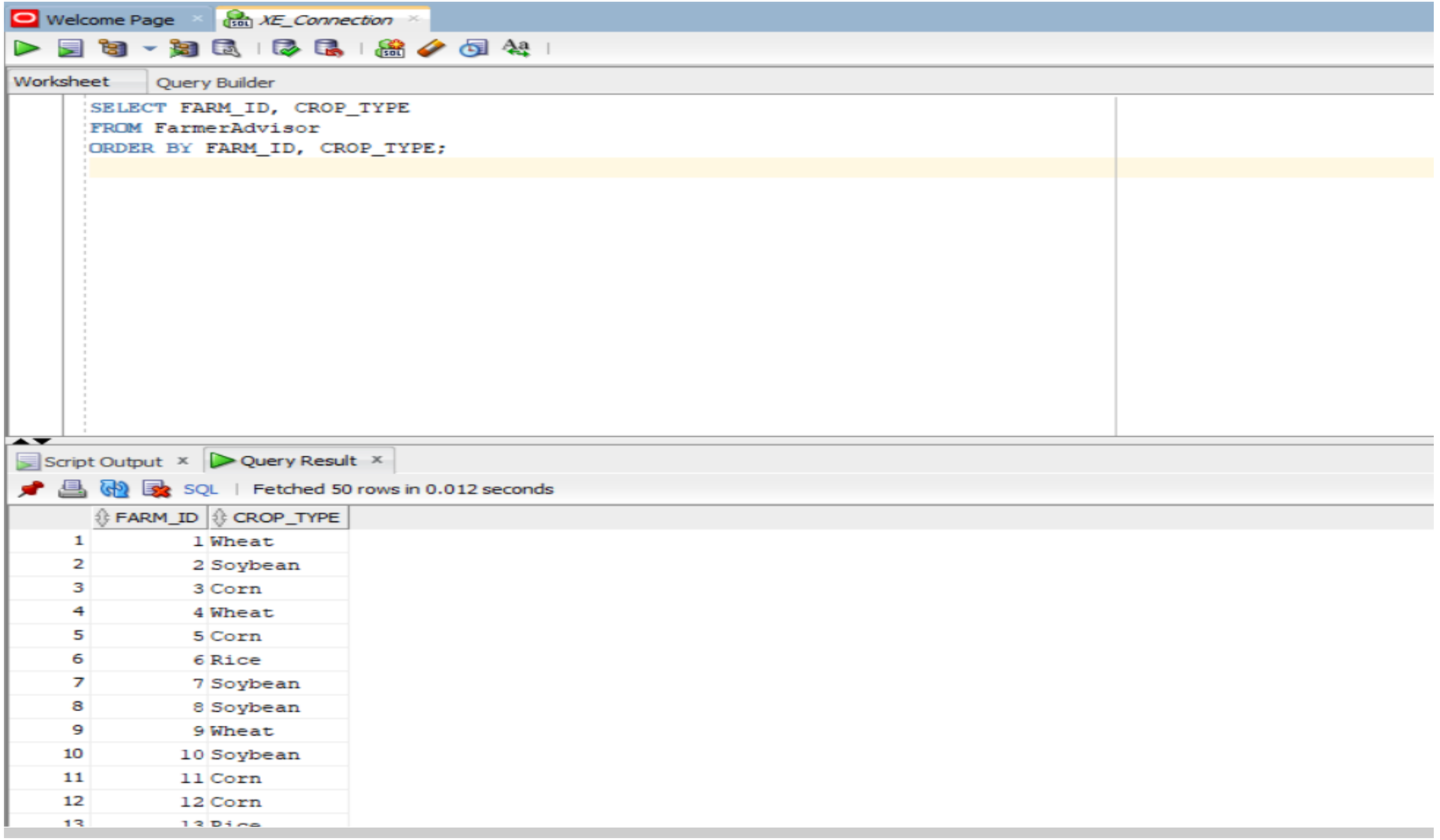
SQL | Fetched 50 rows in 0.239 seconds

	FARM_ID	CROP_TYPE	CROP_YIELD_TON	MARKET_PRICE_PER_TON
1	1	Wheat	1.5769199239944838	499.99905612561844
2	2	Soybean	3.824685657452959	499.8579128747474
3	3	Corn	1.1331976262416603	499.8942115591666
4	4	Wheat	8.870540370316064	499.99905612561844
5	5	Corn	8.779317423442908	499.8942115591666
6	6	Rice	2.0437081720414625	499.9078128975721
7	7	Soybean	6.177211062248057	499.8579128747474
8	8	Soybean	5.218131637064065	499.8579128747474
9	9	Wheat	7.801282070734073	499.99905612561844
10	10	Soybean	6.221388077188259	499.8579128747474
11	11	Corn	2.27026393857822	499.8942115591666
12	12	Corn	2.847876691130483	499.8942115591666
13	13	Rice	9.944867740520532	499.9078128975721

14 Count how many of their farmers grow crops that fall under the “High” growth classification

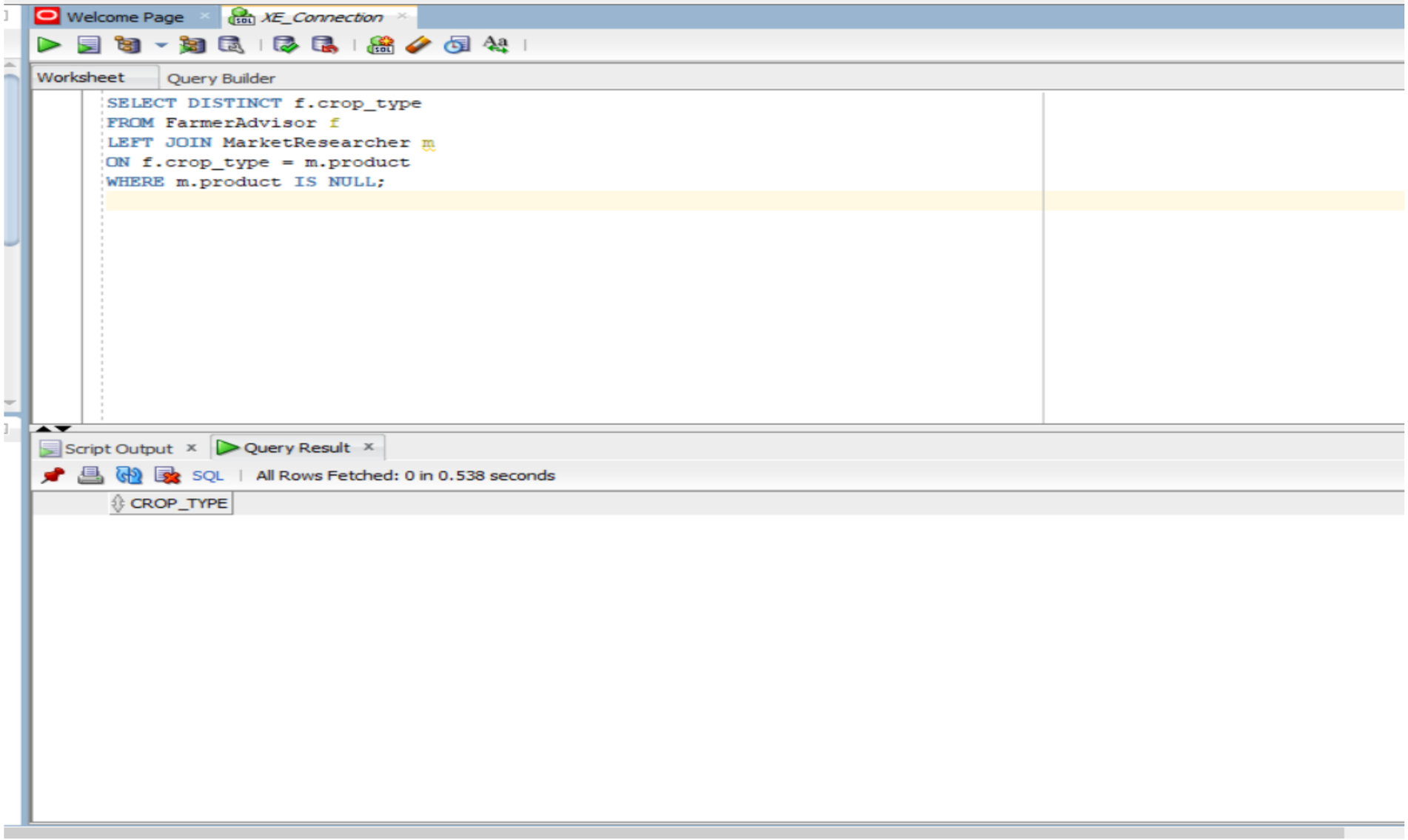


15 Identify if any farmer has duplicate crop entries.

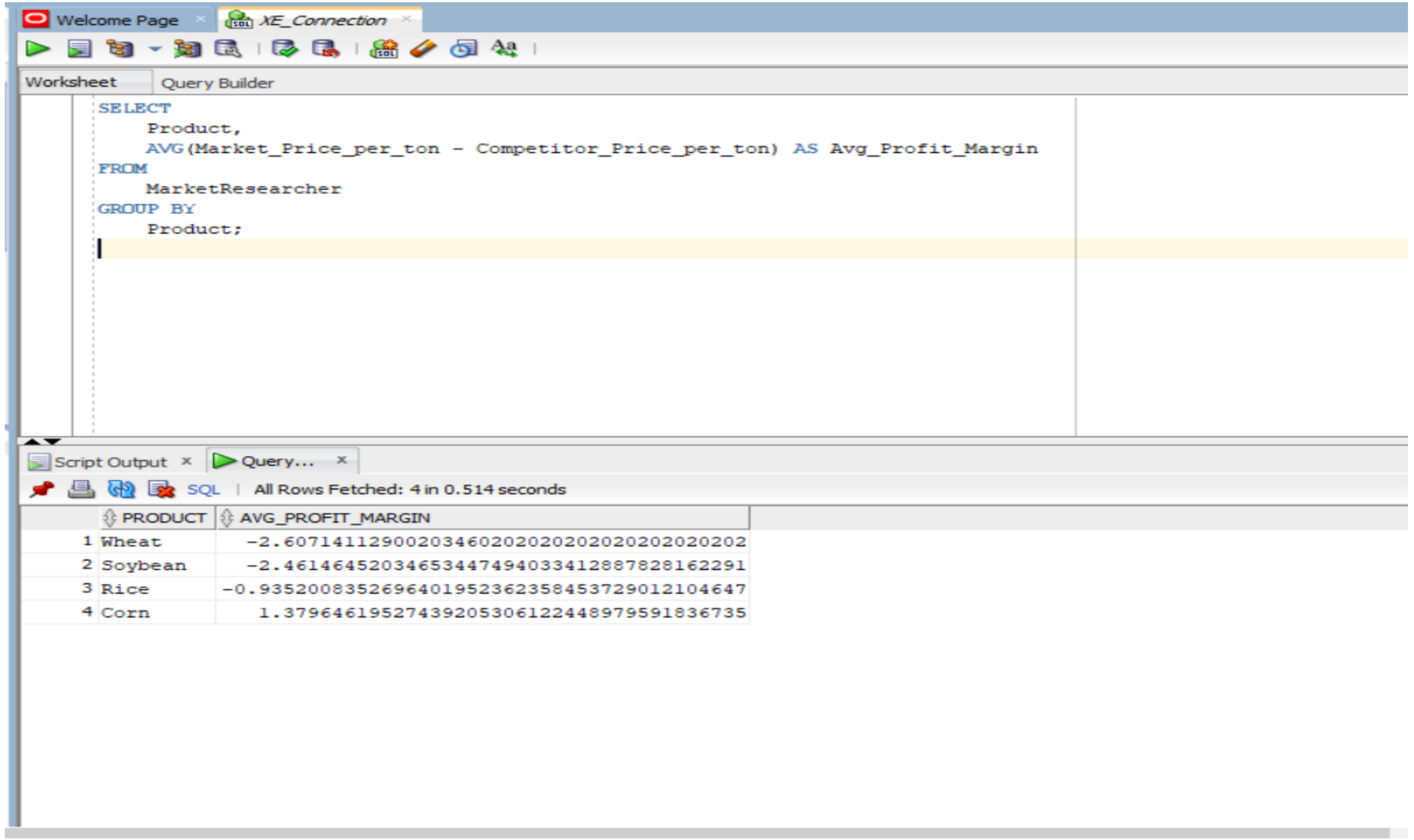


	FARM_ID	CROP_TYPE
1	1	Wheat
2	2	Soybean
3	3	Corn
4	4	Wheat
5	5	Corn
6	6	Rice
7	7	Soybean
8	8	Soybean
9	9	Wheat
10	10	Soybean
11	11	Corn
12	12	Corn
13	13	Rice

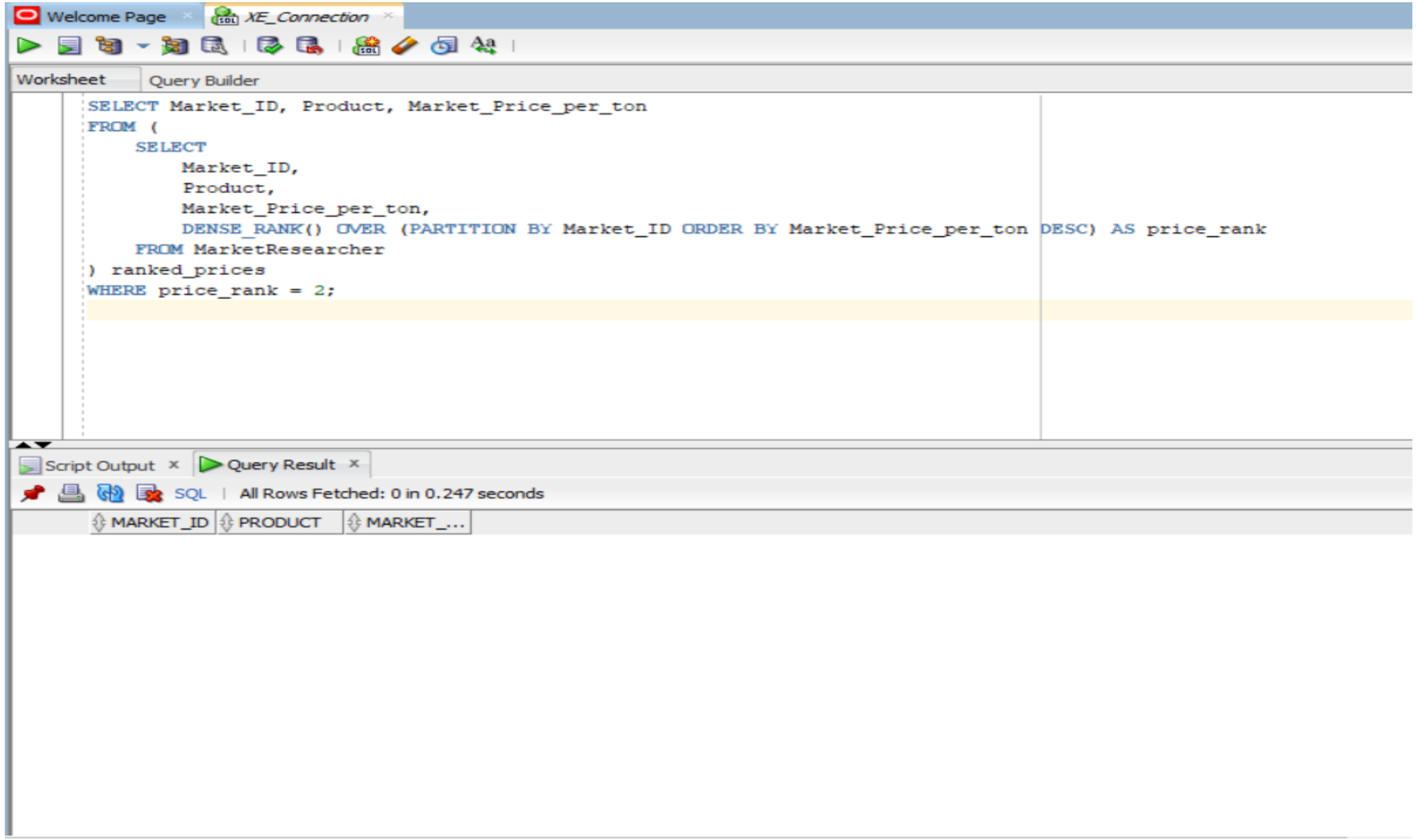
16 All crops grown by farmers that are not listed in the MarketResearcher table.



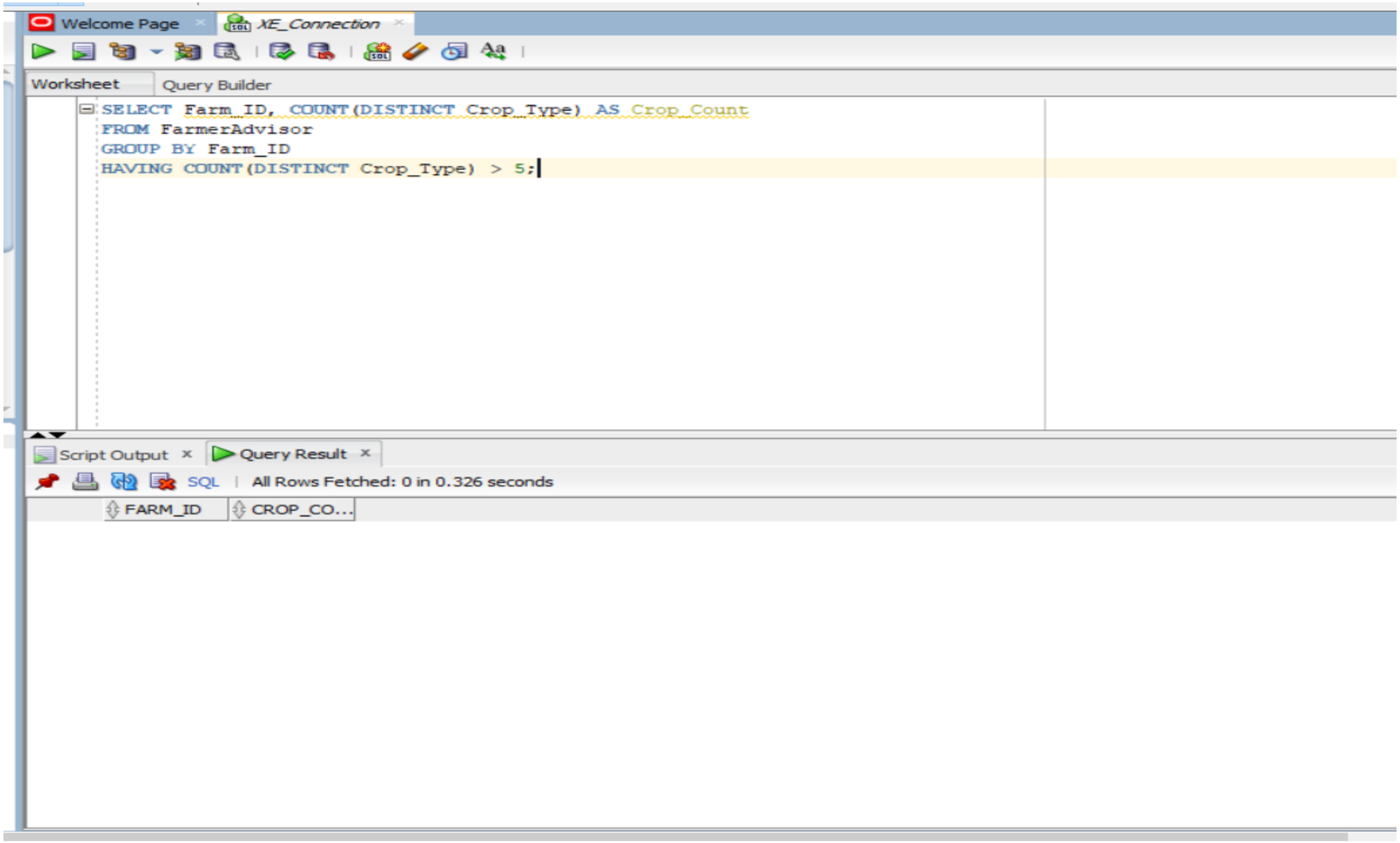
17 Determine which location has the highest average profit margin.



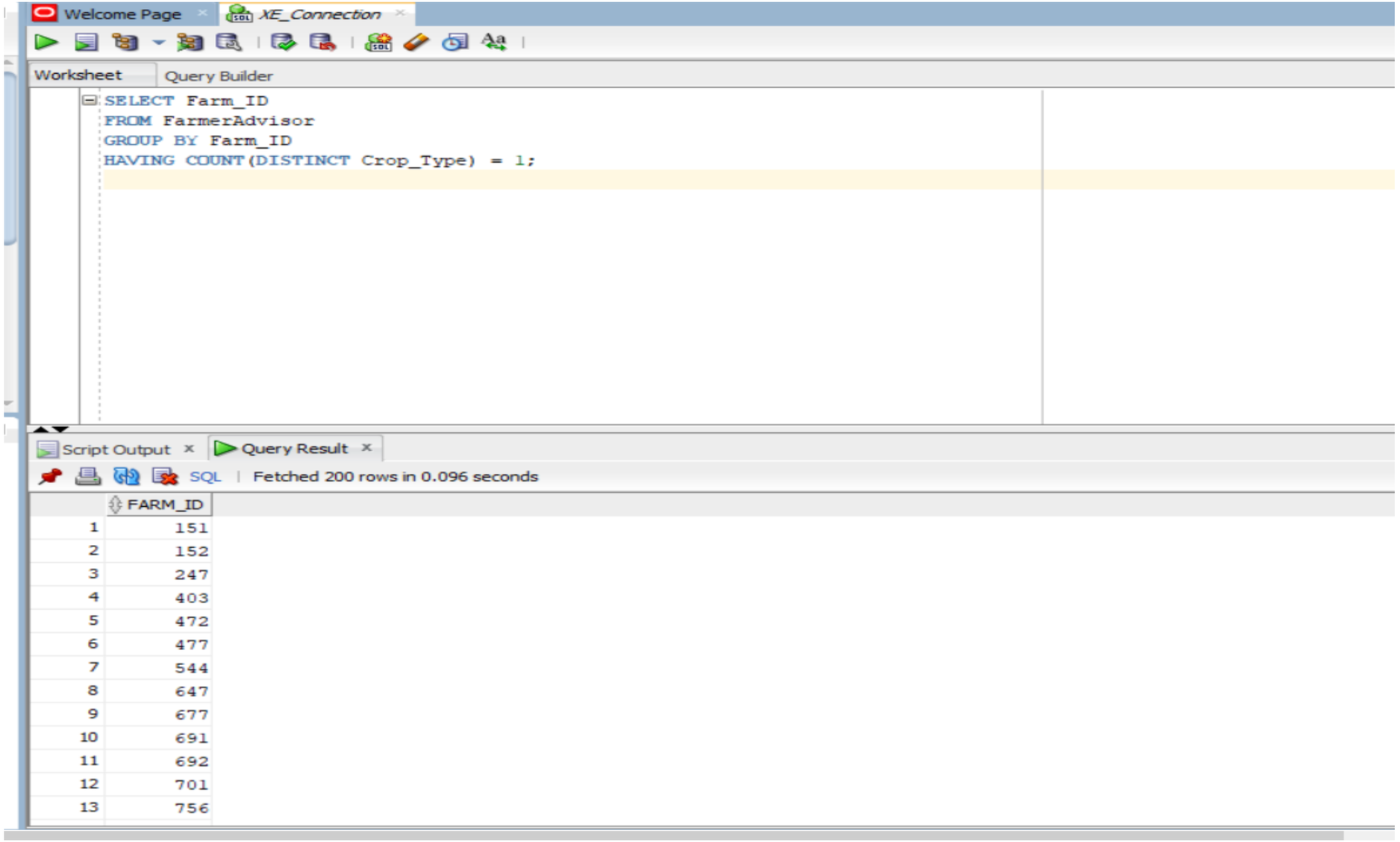
18 Use window functions to find the second-highest market price crop per district.



19 All advisors associated with more than 5 distinct crop types.



20 Farmers who consistently grow the same crop type for all seasons



Conclusion

Through this SQL-based exploration of the agricultural dataset involving the *FarmerAdvisor* and *MarketResearcher* tables, I gained practical experience in writing queries to extract, join, filter, group, and summarize data efficiently. By solving multiple scenario-based questions, I was able to:

- Identify crop growth and profitability patterns
- Analyze farmer-advisor relationships
- Track market price trends across crops
- Derive insights from regional and crop-specific data

Key Takeaways:

- SQL enables powerful data analysis and insight generation directly from relational databases.
- Writing diverse queries improved my understanding of database structure, keys, and data relationships.
- The insights obtained can support better decisions in agriculture, such as advisor assignments, market pricing strategy, and crop planning.
- This exercise also forms a strong base for integrating SQL findings with data visualization tools like Power BI or Tableau for more intuitive presentations.

Thank You