Προχωρημένα Θέματα Βάσεων Δεδομένων

Μαρία Κοιλαλού | Μυρτώ Ορφανάκου
AM:03119211 | AM:

15 Ιανουαρίου 2024

Spark Session

```
from pyspark.sql import SparkSession

def create_spark_session(num_executors):

    spark = SparkSession.builder \
        .appName("ADV DATABASES") \
        .config("spark.executor.instances", str(num_executors)) \
        .config("spark.executor.cores", "1") \
        .config("spark.executor.memory", "1g") \
        .config("spark.driver.memory", "4g") \
        .config("spark.cleaner.periodicGC.interval", "1min") \
        .config("spark.memory.offHeap.enabled", "true") \
        .config("spark.memory.offHeap.size", "1g") \
        .config("spark.default.parallelism", "4") \
        .config("spark.serializer", "org.apache.spark.serializer.KryoSerializer") \
        .config("spark.dynamicAllocation.enabled", "true") \
        .config("spark.dynamicAllocation.minExecutors", "2") \
        .config("spark.dynamicAllocation.maxExecutors", "4") \
        .getOrCreate()

    return spark
```

Your File Caption

```
spark = create_spark_session(4)
print("spark session created")
```

Your File Caption

Create DataFrame

```
schema1 = "`DR_NO` STRING, \
           `Date Rptd` STRING, \
           `DATE OCC` STRING, \
           `TIME OCC` INTEGER, \
           `AREA` INTEGER, \
           `AREA NAME` STRING, \
           `Rpt Dist No` INTEGER, \
           `Part 1-2` INTEGER, \
           `Crm Cd` INTEGER, \
           `Crm Cd Desc` STRING, \
           `Mocodes` STRING, \
           `Vict Age` INTEGER, \
```

```
                `Vict Sex` STRING, \
                `Vict Descent` STRING, \
                `Premis Cd` INTEGER, \
                `Premis Desc` STRING, \
                `Weapon Used Cd` INTEGER, \
                `Weapon Desc` STRING, \
                `Status` STRING, \
                `Status Desc` STRING, \
                `Crm Cd 1` INTEGER, \
                `Crm Cd 2` INTEGER, \
                `Crm Cd 3` INTEGER, \
                `Crm Cd 4` INTEGER, \
                `LOCATION` STRING, \
                `Cross Street` STRING, \
                `LAT` DOUBLE, \
                `LON` DOUBLE"


    data1 = spark.read.csv("/user/ubuntu/ta/advanced-db/data/crime_data_2010.csv", header=True, schema=schema1)
    data2 = spark.read.csv("/user/ubuntu/ta/advanced-db/data/crime_data_2020.csv", header=True, schema=schema1)

    df = data1.union(data2).distinct()

    df = df.withColumn("Date Rptd", to_date(col("Date Rptd"), "MM/dd/yyyy hh:mm:ss a")) \
    .withColumn("DATE OCC", to_date(col("DATE OCC"), "MM/dd/yyyy hh:mm:ss a"))

    df.count()
    print(f"Total number of rows: {df.count()}")

    df.printSchema()
```

Your File Caption

```
Total number of rows: 2913595
root
 |-- DR_NO: string (nullable = true)
 |-- Date Rptd: date (nullable = true)
 |-- DATE OCC: date (nullable = true)
 |-- TIME OCC: integer (nullable = true)
 |-- AREA: integer (nullable = true)
 |-- AREA NAME: string (nullable = true)
 |-- Rpt Dist No: integer (nullable = true)
 |-- Part 1-2: integer (nullable = true)
 |-- Crm Cd: integer (nullable = true)
 |-- Crm Cd Desc: string (nullable = true)
 |-- Mocodes: string (nullable = true)
 |-- Vict Age: integer (nullable = true)
 |-- Vict Sex: string (nullable = true)
 |-- Vict Descent: string (nullable = true)
 |-- Premis Cd: integer (nullable = true)
 |-- Premis Desc: string (nullable = true)
 |-- Weapon Used Cd: integer (nullable = true)
 |-- Weapon Desc: string (nullable = true)
 |-- Status: string (nullable = true)
 |-- Status Desc: string (nullable = true)
 |-- Crm Cd 1: integer (nullable = true)
 |-- Crm Cd 2: integer (nullable = true)
 |-- Crm Cd 3: integer (nullable = true)
 |-- Crm Cd 4: integer (nullable = true)
 |-- LOCATION: string (nullable = true)
 |-- Cross Street: string (nullable = true)
 |-- LAT: double (nullable = true)
 |-- LON: double (nullable = true)
```

Query 1

Dataframe API

```python
    def query1_df(df):
    crime_date = df.withColumn("Year", year("DATE OCC")).withColumn("Month", month("DATE OCC"))

    count = crime_date.groupBy("Year", "Month").count()

    window_spec = Window.partitionBy("Year").orderBy(desc("count"))
    top_months = count.withColumn("rank", dense_rank().over(window_spec)).filter(col("rank") <= 3)

    top_months = top_months.orderBy("Year", "rank")

    return top_months
```

Your File Caption

SQL API

```python
    def query1_sql(df):
    crime_date = df.withColumn("Year", year("DATE OCC")).withColumn("Month", month("DATE OCC"))

    # Δημιουργία προσωρινής προβολής
    crime_date.createOrReplaceTempView("crimes")

    # SQL ερώτημα για την εύρεση των τριών μηνών με τον υψηλότερο αριθμό εγκλημάτων ανά έτος
    query1 = """
    SELECT Year, Month, count, rank
    FROM (
        SELECT Year, Month, count(*) AS count,
               DENSE_RANK() OVER (PARTITION BY Year ORDER BY count(*) DESC) AS rank
        FROM crimes
        GROUP BY Year, Month
    )
    WHERE rank <= 3
    ORDER BY Year, rank
    """

    top_months = crime_date.sparkSession.sql(query1)

    return top_months
```

Your File Caption

```
+----+-----+-----+----+
|Year|Month|count|rank|
+----+-----+-----+----+
|2010|    1|19515|   1|
|2010|    3|18131|   2|
|2010|    7|17856|   3|
|2011|    1|18134|   1|
|2011|    7|17283|   2|
|2011|   10|17034|   3|
|2012|    1|17943|   1|
|2012|    8|17661|   2|
|2012|    5|17502|   3|
|2013|    8|17440|   1|
|2013|    1|16820|   2|
|2013|    7|16644|   3|
|2014|    7|12196|   1|
|2014|   10|12133|   2|
|2014|    8|12028|   3|
|2015|   10|19219|   1|
```

```
|2015|    8|19011|   2|
|2015|    7|18709|   3|
|2016|   10|19659|   1|
|2016|    8|19490|   2|
+----+-----+-----+----+
only showing top 20 rows

Q1 Dataframe time: 0.3153994083404541 seconds.
+----+-----+-----+----+
|Year|Month|count|rank|
+----+-----+-----+----+
|2010|    1|19515|   1|
|2010|    3|18131|   2|
|2010|    7|17856|   3|
|2011|    1|18134|   1|
|2011|    7|17283|   2|
|2011|   10|17034|   3|
|2012|    1|17943|   1|
|2012|    8|17661|   2|
|2012|    5|17502|   3|
|2013|    8|17440|   1|
|2013|    1|16820|   2|
|2013|    7|16644|   3|
|2014|    7|12196|   1|
|2014|   10|12133|   2|
|2014|    8|12028|   3|
|2015|   10|19219|   1|
|2015|    8|19011|   2|
|2015|    7|18709|   3|
|2016|   10|19659|   1|
|2016|    8|19490|   2|
+----+-----+-----+----+
only showing top 20 rows

Q1 SQL time: 0.6481153964996338 seconds.
```

Query 2

DataFrame\SQL API

```python

    def query2_df(df):

    def day_part(hour):
        if 500 <= hour < 1200:
            return "Πρωί"
        elif 1200 <= hour < 1700:
            return "Απόγευμα"
        elif 1700 <= hour < 2100:
            return "Βράδυ"
        else:
            return "Νύχτα"

    day_part_udf = udf(day_part, StringType())

    df_day_part = df.withColumn("DayPart", day_part_udf(col("TIME OCC")))

    df_street_crimes = df_day_part.filter(col("Premis Desc") == "STREET").groupBy("DayPart").count().orderBy(col("count").
    ↪ desc())

    return df_street_crimes
```

Your File Caption

RDD API

```
    def query2_rdd(df):

    def day_part(hour):
        if 500 <= hour < 1200:
            return "Πρωί"
        elif 1200 <= hour < 1700:
            return "Απόγευμα"
        elif 1700 <= hour < 2100:
            return "Βράδυ"
        else:
            return "Νύχτα"

    rdd = df.rdd.filter(lambda row: row['Premis Desc'] == 'STREET')

    def map_day_part(record):
        hour = int(record["TIME OCC"])
        part = day_part(hour)
        return (part, 1)

    rdd_mapped = rdd.map(map_day_part)
    rdd_reduced = rdd_mapped.reduceByKey(lambda a, b: a + b)

    rdd_street_crimes = rdd_reduced.sortBy(lambda x: x[1], ascending=False)

    return rdd_street_crimes
```

Your File Caption

```
+--------+------+
| DayPart| count|
+--------+------+
|   Νύχτα|231546|
|   Βράδυ|182141|
|Απόγευμα|143974|
|   Πρωί|120358|
+--------+------+

Q2 Dataframe time: 0.31096601486206055 seconds.
[('Νύχτα', 231546), ('Βράδυ', 182141), ('Απόγευμα', 143974), ('Πρωί', 120358)]
Q2 RDD time: 39.22880935668945 seconds.
```

Query 3

```
    data3 = spark.read.csv("/user/ubuntu/ta/advanced-db/data/LA_income_2015.csv", header=True, schema=schema2)
    data4 = spark.read.csv("/user/ubuntu/ta/advanced-db/data/revgecoding.csv", header=True, schema=schema3)

    schema3 = "`LAT` DOUBLE, \
               `LON` DOUBLE, \
               `ZIPcode` INTEGER"

    schema2 = "`Zip Code` INTEGER, \
              `Community` STRING, \
              `Estimated Median Income` STRING"

    descent_mapping = {
        'A': 'Other Asian',
        'B': 'Black',
        'C': 'Chinese',
        'D': 'Cambodian',
        'F': 'Filipino',
        'G': 'Guamanian',
        'H': 'Hispanic/Latin/Mexican',
        'I': 'American Indian/Alaskan Native',
```

```
        'J': 'Japanese',
        'K': 'Korean',
        'L': 'Laotian',
        'O': 'Other',
        'P': 'Pacific Islander',
        'S': 'Samoan',
        'U': 'Hawaiian',
        'V': 'Vietnamese',
        'W': 'White',
        'X': 'Unknown',
        'Z': 'Asian Indian'
    }

    data3 = data3.withColumn("Estimated Median Income", regexp_replace(col("Estimated Median Income"), "\$", ""))
    data3 = data3.withColumn("Estimated Median Income", regexp_replace(col("Estimated Median Income"), ",", "").cast("
    ↪ float"))

    crime_year = df.withColumn("Year", year("DATE OCC"))

    crime_2015 = crime_year.filter(
       (col("Year") == 2015) &
       (col("Vict Descent").isNotNull()))

    def map_descent(code):
        return descent_mapping.get(code, "Unknown")  # Default to "Unknown" if code not found

    map_descent_udf = udf(map_descent, StringType())

    crime_2015 = crime_2015.withColumn("Vict Descent", map_descent_udf(crime_2015["Vict Descent"]))

    revgecoding = data4.dropDuplicates(['LAT', 'LON'])
```

Your File Caption

```
    def query3 (crime_2015, data3, revgecoding):

        crime_zip = crime_2015.join(revgecoding, ["LAT", "LON"], "left")

        best3_zip = data3.orderBy("Estimated Median Income", ascending=False).limit(3)
        worst3_zip = data3.orderBy("Estimated Median Income", ascending=True).limit(3)

        best3_zip_list = [row['Zip Code'] for row in best3_zip.collect()]
        worst3_zip_list = [row['Zip Code'] for row in worst3_zip.collect()]

        crimes = crime_zip.filter(
            (col("ZIPcode").isin(best3_zip_list)) |
            (col("ZIPcode").isin(worst3_zip_list))
        )

        vict_descent_count = crimes.groupBy("Vict Descent").count().orderBy("count", ascending=False)


    return vict_descent_count
```

Your File Caption

2 Executors

```
+--------------------+-----+
|       Vict Descent|count|
+--------------------+-----+
|Hispanic/Latin/Me...| 1053|
|               White|  610|
|               Black|  349|
|               Other|  272|
|             Unknown|   71|
|        Other Asian|   46|
|              Korean|    4|
```

```
|              Chinese|    1|
|American Indian/A...|    1|
+--------------------+-----+

Number of Executors: 2
Q3 time: 10.069442987442017 seconds.
```

3 Executors

```
+--------------------+-----+
|        Vict Descent|count|
+--------------------+-----+
|Hispanic/Latin/Me...| 1053|
|               White|  610|
|               Black|  349|
|               Other|  272|
|             Unknown|   71|
|         Other Asian|   46|
|              Korean|    4|
|American Indian/A...|    1|
|              Chinese|    1|
+--------------------+-----+

Number of Executors: 3
Q3 time: 5.2063148021698 seconds.
```

4 Executors

```
+--------------------+-----+
|        Vict Descent|count|
+--------------------+-----+
|Hispanic/Latin/Me...| 1053|
|               White|  610|
|               Black|  349|
|               Other|  272|
|             Unknown|   71|
|         Other Asian|   46|
|              Korean|    4|
|American Indian/A...|    1|
|              Chinese|    1|
+--------------------+-----+

Number of Executors: 3
Q3 time: 5.2063148021698 seconds.
```

Query 4

```
1
2     schema4 = "`X` DOUBLE, \
3                `Y` DOUBLE, \
4                `FID` INTEGER, \
5                `DIVISION` STRING, \
6                `LOCATION` STRING, \
7                `PREC` INTEGER"
8
9     data5 = spark.read.csv("/user/ubuntu/ta/advanced-db/data/LAPD_Police_Stations.csv", header=True, schema=schema4)
```

Your File Caption

```
def query4(df, data5):

    def haversine(lat1, lon1, lat2, lon2):
        # Radius of the Earth in kilometers
        R = 6371.0

        lat1_rad = math.radians(lat1)
        lon1_rad = math.radians(lon1)
        lat2_rad = math.radians(lat2)
        lon2_rad = math.radians(lon2)

        dlat = lat2_rad - lat1_rad
        dlon = lon2_rad - lon1_rad

        a = math.sin(dlat / 2)**2 + math.cos(lat1_rad) * math.cos(lat2_rad) * math.sin(dlon / 2)**2
        c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))

        distance = R * c
        return distance

    def get_distance(lat1, long1, lat2, long2):

        def is_valid_coordinate(lat, lon):
            return -90 <= lat <= 90 and -180 <= lon <= 180


        if not is_valid_coordinate(lat1, long1) or not is_valid_coordinate(lat2, long2):
            # Print the invalid rows
            print(f"Invalid row: lat1={lat1}, long1={long1}, lat2={lat2}, long2={long2}")
            return -1

        try:
            return haversine(lat1, long1, lat2, long2)
        except ValueError:
            return -1

    df_4a = df.filter(
        (df["AREA NAME"] != "Null Island") &
        (df["Weapon Used Cd"].substr(1, 1) == "1")
    )

    df_4b = df.filter(
        (df["AREA NAME"] != "Null Island") &
        (df["Weapon Used Cd"].isNotNull())
    )

    joined_df_4a = df_4a.join(data5, df_4a["AREA"] == data5["PREC"])
    joined_df_4b = df_4b.join(data5, df_4b["AREA"] == data5["PREC"])

    distance_udf = udf(get_distance)

    distance_df_4a = joined_df_4a.withColumn(
        "DISTANCE",
        distance_udf(
            F.col("LAT"), F.col("LON"),
            F.col("Y"), F.col("X")
        ).cast("double")
    )

    distance_df_4b = joined_df_4b.withColumn(
        "DISTANCE",
        distance_udf(
            F.col("LAT"), F.col("LON"),
            F.col("Y"), F.col("X")
        ).cast("double")
    )

    query_4_1a = distance_df_4a.groupBy("Year").agg(
        F.count("*").alias("num_crimes"),
        F.avg("DISTANCE").alias("average_distance")
```

```
    ).orderBy("Year")

    query_4_1b = distance_df_4b.groupBy("DIVISION").agg(
        F.count("*").alias("num_crimes"),
        F.avg("DISTANCE").alias("average_distance")
    ).orderBy(F.desc("num_crimes"))

    printΑπόσταση(" από το αστυνομικό τμήμα που ανέλαβε την έρευνα για το περιστατικό:")
    print("(a)")
    query_4_1a.show()
    print("(b)")
    query_4_1b.show()

    cross_joined_df = df.crossJoin(data5.withColumnRenamed("LAT", "Y").withColumnRenamed("LON", "X"))

    cross_joined_df = cross_joined_df.withColumn(
        "DISTANCE",
        distance_udf(col("LAT"), col("LON"), col("Y"), col("X")).cast("double")
    )

    windowSpec = Window.partitionBy("DR_NO").orderBy("DISTANCE")

    nearest_station_df = cross_joined_df.withColumn(
        "row_num",
        F.row_number().over(windowSpec)
    ).filter(col("row_num") == 1).drop("row_num")

    cross_df_4a = df_4a.join(nearest_station_df.drop("Year"), "DR_NO")
    cross_df_4b = df_4b.join(nearest_station_df.drop("Year"), "DR_NO")

    query_4_2a = cross_df_4a.groupBy("Year").agg(
        F.count("*").alias("num_crimes"),
        F.avg("DISTANCE").alias("average_distance")
    ).orderBy("Year")

    query_4_2b = cross_df_4b.groupBy("DIVISION").agg(
        F.count("*").alias("num_crimes"),
        F.avg("DISTANCE").alias("average_distance")
    ).orderBy(F.desc("num_crimes"))


    printΑπόσταση(" από το πλησιέστερο αστυνομικό τμήμα:")
    print("(a)")
    query_4_2a.show()
    print("(b)")
    query_4_2b.show()
```

Your File Caption

Απόσταση από το αστυνομικό τμήμα που ανέλαβε την έρευνα για το περιστατικό:
(a)

Year	num_crimes	average_distance
2010	8213	4.315547525861609
2011	7232	2.7931783031826134
2012	6550	37.401521647671025
2013	5838	2.826412721201962
2014	4230	11.631025289489838
2015	6763	2.706097992762 39
2016	8100	2.7176445421299724
2017	7788	5.955847913803834
2018	7413	2.732823649229879
2019	7129	2.7399419721721476
2020	8491	8.614767812336167
2021	9767	30.97834129556094
2022	10025	2.608656186450 79

|2023| 8741|2.5551410574543145|
+----+----------+------------------+

(b)
+---------------+----------+------------------+
| DIVISION|num_crimes| average_distance|
+---------------+----------+------------------+
77TH STREET	94474	13.162079052889169
SOUTHEAST	72832	14.527525557922345
SOUTHWEST	72461	9.898850561769162
CENTRAL	63264	23.466578376496436
NEWTON	61160	13.979683650325562
RAMPART	55611	19.847575514305305
HOLLYWOOD	50958	27.846180453246344
OLYMPIC	48886	17.19463769893097
PACIFIC	42760	25.072161839219607
HOLLENBECK	41393	19.600667247278746
MISSION	40880	21.330305780965872
HARBOR	40637	14.160243637196546
NORTH HOLLYWOOD	39542	17.184089891996653
WILSHIRE	37712	16.080369127182834
NORTHEAST	37101	12.799215742840445
VAN NUYS	36080	19.92610084395629
WEST VALLEY	33694	15.344662228609367
TOPANGA	32340	7.019388768312766
FOOTHILL	32337	16.293601573548496
DEVONSHIRE	28673	16.711290455491336
+---------------+----------+------------------+
only showing top 20 rows

Απόσταση από το πλησιέστερο αστυνομικό τμήμα:
(a)
+----+----------+------------------+
|Year|num_crimes| average_distance|
+----+----------+------------------+
|2010| 8213| 3.965480506097993|
|2011| 7232| 2.46181888566459|
|2012| 6550| 37.04806556244542|
|2013| 5838|2.4561803379459084|
|2014| 4230|11.240705060052028|
|2015| 6763| 2.38790278176303|
|2016| 8100|2.4291509215379303|
|2017| 7788| 5.620278866952371|
|2018| 7413| 2.4090835060096955|
|2019| 7129|2.4301661049761214|
|2020| 8491| 8.305664894299344|
|2021| 9767|30.666116941658924|
|2022| 10025|2.3129679282459743|
|2023| 8741|2.2716948056968675|
+----+----------+------------------+

(b)
+---------------+----------+------------------+
| DIVISION|num_crimes| average_distance|
+---------------+----------+------------------+
77TH STREET	78830	1.6735955739672674
SOUTHWEST	78068	2.161146839862122
HOLLYWOOD	70652	1.9193991303646156
SOUTHEAST	66697	2.222232812296741
OLYMPIC	60553	1.6657781632089452
CENTRAL	59420	0.866802366302343
WILSHIRE	58032	2.4783030061160027

```
|        RAMPART|     56297| 1.361662592838379|
|       VAN NUYS|     55252|2.8073800912108116|
|         NEWTON|     45398|1.5998998286047477|
|     HOLLENBECK|     43128| 326.0868010778892|
|        PACIFIC|     40356| 3.845258334030655|
| NORTH HOLLYWOOD|     40174|2.5926392412493717|
|         HARBOR|     39433| 3.686751365873389|
|       FOOTHILL|     38327| 3.977261444227064|
|    WEST VALLEY|     34804| 2.850125414289692|
|        TOPANGA|     32648| 3.045398368787074|
|      NORTHEAST|     27283| 3.765925137599394|
|        MISSION|     27016|3.7840372209112974|
|WEST LOS ANGELES|     22376| 2.712234758681144|
+----------------+----------+------------------+
only showing top 20 rows
```

hint & explain

Query 3

```
    join_strategies = ["broadcast", "merge", "shuffle_hash", "shuffle_replicate_nl"]

    for strategy in join_strategies:
        print(f"Executing query with {strategy} join strategy")
        query3_hne_results = query3_hne.query3(crime_2015, data3, revgecoding, strategy)
        query3_hne_results.show()
```

Your File Caption

```
    def query3(crime_2015, data3, revgecoding, join_strategy):

        crime_zip = crime_2015.join(revgecoding.hint(join_strategy), ["LAT", "LON"], "left")

        best3_zip = data3.orderBy("Estimated Median Income", ascending=False).limit(3)
        worst3_zip = data3.orderBy("Estimated Median Income", ascending=True).limit(3)

        best3_zip_list = [row['Zip Code'] for row in best3_zip.collect()]
        worst3_zip_list = [row['Zip Code'] for row in worst3_zip.collect()]

        crimes = crime_zip.filter(
            (col("ZIPcode").isin(best3_zip_list)) |
            (col("ZIPcode").isin(worst3_zip_list))
        )

        vict_descent_count = crimes.groupBy("Vict Descent").count().orderBy("count", ascending=False)

        vict_descent_count.explain()

        return vict_descent_count
```

Your File Caption

```
Executing query with broadcast join strategy
== Physical Plan ==
AdaptiveSparkPlan (53)
+- == Final Plan ==
   TakeOrderedAndProject (31)
   +- * HashAggregate (30)
      +- AQEShuffleRead (29)
         +- ShuffleQueryStage (28), Statistics(sizeInBytes=1016.0 B, rowCount=28)
            +- Exchange (27)
               +- * HashAggregate (26)
                  +- * Project (25)
                     +- * BroadcastHashJoin Inner BuildRight (24)
```

```
                        :- * Project (12)
                        :  +- BatchEvalPython (11)
                        :     +- * HashAggregate (10)
                        :        +- AQEShuffleRead (9)
                        :           +- ShuffleQueryStage (8), Statistics(sizeInBytes=88.4 MiB, rowCount=1.96E+5)
                        :              +- Exchange (7)
                        :                 +- * HashAggregate (6)
                        :                    +- Union (5)
                        :                       :- * Filter (2)
                        :                       :  +- Scan csv  (1)
                        :                       +- * Filter (4)
                        :                          +- Scan csv  (3)
                        +- BroadcastQueryStage (23), Statistics(sizeInBytes=8.0 MiB, rowCount=682)
                           +- BroadcastExchange (22)
                              +- * Project (21)
                                 +- * Filter (20)
                                    +- * HashAggregate (19)
                                       +- AQEShuffleRead (18)
                                          +- ShuffleQueryStage (17), Statistics(sizeInBytes=1475.8 KiB, rowCount=3.78E+4)
                                             +- Exchange (16)
                                                +- * HashAggregate (15)
                                                   +- * Filter (14)
                                                      +- Scan csv  (13)
+- == Initial Plan ==
   TakeOrderedAndProject (52)
   +- HashAggregate (51)
      +- Exchange (50)
         +- HashAggregate (49)
            +- Project (48)
               +- BroadcastHashJoin Inner BuildRight (47)
                  :- Project (39)
                  :  +- BatchEvalPython (38)
                  :     +- HashAggregate (37)
                  :        +- Exchange (36)
                  :           +- HashAggregate (35)
                  :              +- Union (34)
                  :                 :- Filter (32)
                  :                 :  +- Scan csv  (1)
                  :                 +- Filter (33)
                  :                    +- Scan csv  (3)
                  +- BroadcastExchange (46)
                     +- Project (45)
                        +- Filter (44)
                           +- HashAggregate (43)
                              +- Exchange (42)
                                 +- HashAggregate (41)
                                    +- Filter (40)
                                       +- Scan csv  (13)

+--------------------+-----+
|        Vict Descent|count|
+--------------------+-----+
|Hispanic/Latin/Me...| 1053|
|               White|  610|
|               Black|  349|
|               Other|  272|
|             Unknown|   71|
|          Other Asian|   46|
|              Korean|    4|
|             Chinese|    1|
|American Indian/A...|    1|
+--------------------+-----+

Executing query with merge join strategy
== Physical Plan ==
AdaptiveSparkPlan (62)
+- == Final Plan ==
   TakeOrderedAndProject (37)
   +- * HashAggregate (36)
      +- AQEShuffleRead (35)
         +- ShuffleQueryStage (34), Statistics(sizeInBytes=584.0 B, rowCount=16)
```

```
            +- Exchange (33)
               +- * HashAggregate (32)
                  +- * Project (31)
                     +- * SortMergeJoin Inner (30)
                        :- * Sort (16)
                        :  +- AQEShuffleRead (15)
                        :     +- ShuffleQueryStage (14), Statistics(sizeInBytes=8.7 MiB, rowCount=1.96E+5)
                        :        +- Exchange (13)
                        :           +- * Project (12)
                        :              +- BatchEvalPython (11)
                        :                 +- * HashAggregate (10)
                        :                    +- AQEShuffleRead (9)
                        :                       +- ShuffleQueryStage (8), Statistics(sizeInBytes=88.4 MiB, rowCount=1.96E
    ↪ +5)
                        :                          +- Exchange (7)
                        :                             +- * HashAggregate (6)
                        :                                +- Union (5)
                        :                                   :- * Filter (2)
                        :                                   :  +- Scan csv  (1)
                        :                                   +- * Filter (4)
                        :                                      +- Scan csv  (3)
                        +- * Sort (29)
                           +- AQEShuffleRead (28)
                              +- ShuffleQueryStage (27), Statistics(sizeInBytes=16.0 KiB, rowCount=682)
                                 +- Exchange (26)
                                    +- * Project (25)
                                       +- * Filter (24)
                                          +- * HashAggregate (23)
                                             +- AQEShuffleRead (22)
                                                +- ShuffleQueryStage (21), Statistics(sizeInBytes=1475.8 KiB, rowCount
    ↪ =3.78E+4)
                                                   +- Exchange (20)
                                                      +- * HashAggregate (19)
                                                         +- * Filter (18)
                                                            +- Scan csv  (17)
+- == Initial Plan ==
   TakeOrderedAndProject (61)
   +- HashAggregate (60)
      +- Exchange (59)
         +- HashAggregate (58)
            +- Project (57)
               +- SortMergeJoin Inner (56)
                  :- Sort (47)
                  :  +- Exchange (46)
                  :     +- Project (45)
                  :        +- BatchEvalPython (44)
                  :           +- HashAggregate (43)
                  :              +- Exchange (42)
                  :                 +- HashAggregate (41)
                  :                    +- Union (40)
                  :                       :- Filter (38)
                  :                       :  +- Scan csv  (1)
                  :                       +- Filter (39)
                  :                          +- Scan csv  (3)
                  +- Sort (55)
                     +- Exchange (54)
                        +- Project (53)
                           +- Filter (52)
                              +- HashAggregate (51)
                                 +- Exchange (50)
                                    +- HashAggregate (49)
                                       +- Filter (48)
                                          +- Scan csv  (17)

+----------------------+-----+
|         Vict Descent|count|
+----------------------+-----+
|Hispanic/Latin/Me...| 1053|
|                White|  610|
|                Black|  349|
|                Other|  272|
```

```
|                Unknown|   71|
|            Other Asian|   46|
|                 Korean|    4|
|American Indian/A...|    1|
|                Chinese|    1|
+-----------------------+-----+

Executing query with shuffle_hash join strategy
== Physical Plan ==
AdaptiveSparkPlan (58)
+- == Final Plan ==
   TakeOrderedAndProject (35)
   +- * HashAggregate (34)
      +- AQEShuffleRead (33)
         +- ShuffleQueryStage (32), Statistics(sizeInBytes=584.0 B, rowCount=16)
            +- Exchange (31)
               +- * HashAggregate (30)
                  +- * Project (29)
                     +- * ShuffledHashJoin Inner BuildRight (28)
                        :- AQEShuffleRead (15)
                        :  +- ShuffleQueryStage (14), Statistics(sizeInBytes=8.7 MiB, rowCount=1.96E+5)
                        :     +- Exchange (13)
                        :        +- * Project (12)
                        :           +- BatchEvalPython (11)
                        :              +- * HashAggregate (10)
                        :                 +- AQEShuffleRead (9)
                        :                    +- ShuffleQueryStage (8), Statistics(sizeInBytes=88.4 MiB, rowCount=1.96E+5)
                        :                       +- Exchange (7)
                        :                          +- * HashAggregate (6)
                        :                             +- Union (5)
                        :                                :- * Filter (2)
                        :                                :  +- Scan csv  (1)
                        :                                +- * Filter (4)
                        :                                   +- Scan csv  (3)
                        +- AQEShuffleRead (27)
                           +- ShuffleQueryStage (26), Statistics(sizeInBytes=16.0 KiB, rowCount=682)
                              +- Exchange (25)
                                 +- * Project (24)
                                    +- * Filter (23)
                                       +- * HashAggregate (22)
                                          +- AQEShuffleRead (21)
                                             +- ShuffleQueryStage (20), Statistics(sizeInBytes=1475.8 KiB, rowCount=3.78E
    ↪ +4)
                                                +- Exchange (19)
                                                   +- * HashAggregate (18)
                                                      +- * Filter (17)
                                                         +- Scan csv  (16)

+- == Initial Plan ==
   TakeOrderedAndProject (57)
   +- HashAggregate (56)
      +- Exchange (55)
         +- HashAggregate (54)
            +- Project (53)
               +- ShuffledHashJoin Inner BuildRight (52)
                  :- Exchange (44)
                  :  +- Project (43)
                  :     +- BatchEvalPython (42)
                  :        +- HashAggregate (41)
                  :           +- Exchange (40)
                  :              +- HashAggregate (39)
                  :                 +- Union (38)
                  :                    :- Filter (36)
                  :                    :  +- Scan csv  (1)
                  :                    +- Filter (37)
                  :                       +- Scan csv  (3)
                  +- Exchange (51)
                     +- Project (50)
                        +- Filter (49)
                           +- HashAggregate (48)
                              +- Exchange (47)
                                 +- HashAggregate (46)
```

```
                                        +- Filter (45)
                                           +- Scan csv  (16)

+----------------------+-----+
|         Vict Descent|count|
+----------------------+-----+
|Hispanic/Latin/Me...| 1053|
|                White|  610|
|                Black|  349|
|                Other|  272|
|              Unknown|   71|
|          Other Asian|   46|
|               Korean|    4|
|              Chinese|    1|
|American Indian/A...|    1|
+----------------------+-----+

Executing query with shuffle_replicate_nl join strategy
== Physical Plan ==
AdaptiveSparkPlan (50)
+- == Final Plan ==
   TakeOrderedAndProject (29)
   +- * HashAggregate (28)
      +- AQEShuffleRead (27)
         +- ShuffleQueryStage (26), Statistics(sizeInBytes=3.5 KiB, rowCount=100)
            +- Exchange (25)
               +- * HashAggregate (24)
                  +- * Project (23)
                     +- CartesianProduct Inner (22)
                        :- * Project (12)
                        :  +- BatchEvalPython (11)
                        :     +- * HashAggregate (10)
                        :        +- AQEShuffleRead (9)
                        :           +- ShuffleQueryStage (8), Statistics(sizeInBytes=88.4 MiB, rowCount=1.96E+5)
                        :              +- Exchange (7)
                        :                 +- * HashAggregate (6)
                        :                    +- Union (5)
                        :                       :- * Filter (2)
                        :                       :  +- Scan csv  (1)
                        :                       +- * Filter (4)
                        :                          +- Scan csv  (3)
                        +- * Project (21)
                           +- * Filter (20)
                              +- * HashAggregate (19)
                                 +- AQEShuffleRead (18)
                                    +- ShuffleQueryStage (17), Statistics(sizeInBytes=1475.8 KiB, rowCount=3.78E+4)
                                       +- Exchange (16)
                                          +- * HashAggregate (15)
                                             +- * Filter (14)
                                                +- Scan csv  (13)
+- == Initial Plan ==
   TakeOrderedAndProject (49)
   +- HashAggregate (48)
      +- Exchange (47)
         +- HashAggregate (46)
            +- Project (45)
               +- CartesianProduct Inner (44)
                  :- Project (37)
                  :  +- BatchEvalPython (36)
                  :     +- HashAggregate (35)
                  :        +- Exchange (34)
                  :           +- HashAggregate (33)
                  :              +- Union (32)
                  :                 :- Filter (30)
                  :                 :  +- Scan csv  (1)
                  :                 +- Filter (31)
                  :                    +- Scan csv  (3)
                  +- Project (43)
                     +- Filter (42)
                        +- HashAggregate (41)
                           +- Exchange (40)
```

```
                              +- HashAggregate (39)
                                 +- Filter (38)
                                    +- Scan csv  (13)

+--------------------+-----+
|        Vict Descent|count|
+--------------------+-----+
|Hispanic/Latin/Me...| 1053|
|               White|  610|
|               Black|  349|
|               Other|  272|
|             Unknown|   71|
|       Other Asian  |   46|
|              Korean|    4|
|             Chinese|    1|
|American Indian/A...|    1|
+--------------------+-----+
```

Your File Caption

Query 4

```python
    def query4(df, data5, join_strategy):

        def haversine(lat1, lon1, lat2, lon2):
            # Radius of the Earth in kilometers
            R = 6371.0

            lat1_rad = math.radians(lat1)
            lon1_rad = math.radians(lon1)
            lat2_rad = math.radians(lat2)
            lon2_rad = math.radians(lon2)

            dlat = lat2_rad - lat1_rad
            dlon = lon2_rad - lon1_rad

            a = math.sin(dlat / 2)**2 + math.cos(lat1_rad) * math.cos(lat2_rad) * math.sin(dlon / 2)**2
            c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))

            distance = R * c
            return distance

        def get_distance(lat1, long1, lat2, long2):

            def is_valid_coordinate(lat, lon):
                return -90 <= lat <= 90 and -180 <= lon <= 180


            if not is_valid_coordinate(lat1, long1) or not is_valid_coordinate(lat2, long2):
                # Print the invalid rows
                print(f"Invalid row: lat1={lat1}, long1={long1}, lat2={lat2}, long2={long2}")
                return -1

            try:
                return haversine(lat1, long1, lat2, long2)
            except ValueError:
                return -1

        df_4a = df.filter(
            (df["AREA NAME"] != "Null Island") &
            (df["Weapon Used Cd"].substr(1, 1) == "1")
        )

        df_4b = df.filter(
            (df["AREA NAME"] != "Null Island") &
            (df["Weapon Used Cd"].isNotNull())
        )

        joined_df_4a = df_4a.join(data5.hint(join_strategy), df_4a["AREA"] == data5["PREC"])
```

```
joined_df_4b = df_4b.join(data5.hint(join_strategy), df_4b["AREA"] == data5["PREC"])

joined_df_4a.explain()
joined_df_4b.explain()

distance_udf = udf(get_distance)

distance_df_4a = joined_df_4a.withColumn(
    "DISTANCE",
    distance_udf(
        F.col("LAT"), F.col("LON"),
        F.col("Y"), F.col("X")
    ).cast("double")
)

distance_df_4b = joined_df_4b.withColumn(
    "DISTANCE",
    distance_udf(
        F.col("LAT"), F.col("LON"),
        F.col("Y"), F.col("X")
    ).cast("double")
)

query_4_1a = distance_df_4a.groupBy("Year").agg(
    F.count("*").alias("num_crimes"),
    F.avg("DISTANCE").alias("average_distance")
).orderBy("Year")

query_4_1b = distance_df_4b.groupBy("DIVISION").agg(
    F.count("*").alias("num_crimes"),
    F.avg("DISTANCE").alias("average_distance")
).orderBy(F.desc("num_crimes"))

query_4_1a.explain()
query_4_1b.explain()

print("Απόσταση από το αστυνομικό τμήμα που ανέλαβε την έρευνα για το περιστατικό:")
print("(a)")
query_4_1a.show()
print("(b)")
query_4_1b.show()

cross_joined_df = df.crossJoin(data5.withColumnRenamed("LAT", "Y").withColumnRenamed("LON", "X"))


cross_joined_df = cross_joined_df.withColumn(
    "DISTANCE",
    distance_udf(col("LAT"), col("LON"), col("Y"), col("X")).cast("double")
)

windowSpec = Window.partitionBy("DR_NO").orderBy("DISTANCE")

nearest_station_df = cross_joined_df.withColumn(
    "row_num",
    F.row_number().over(windowSpec)
).filter(col("row_num") == 1).drop("row_num")

cross_df_4a = df_4a.join(nearest_station_df.drop("Year"), "DR_NO")
cross_df_4b = df_4b.join(nearest_station_df.drop("Year"), "DR_NO")

query_4_2a = cross_df_4a.groupBy("Year").agg(
    F.count("*").alias("num_crimes"),
    F.avg("DISTANCE").alias("average_distance")
).orderBy("Year")

query_4_2b = cross_df_4b.groupBy("DIVISION").agg(
    F.count("*").alias("num_crimes"),
    F.avg("DISTANCE").alias("average_distance")
).orderBy(F.desc("num_crimes"))

print("Απόσταση από το πλησιέστερο αστυνομικό τμήμα:")
```

```
119         print("(a)")
120         query_4_2a.show()
121         print("(b)")
122         query_4_2b.show()
```

Your File Caption

Your File Caption

Your File Caption

Your File Caption

Your File Caption