Improve the execution plan of a query with a materialized view

Note:

You are not required to complete the processes, tasks, activities, or steps presented in this example. The various samples provided are for illustrative purposes only and it's likely that if you try this out you will encounter issues in your system.

1. Run the query with the **`EXPLAIN`** directive (note the **`WITH_RECOMMENDATIONS`** option as well):

```
EXPLAIN WITH_RECOMMENDATIONS
SELECT
    T.TransactionItemsCountBucket
    ,count(*) as CustomersCount
FROM
    (
        SELECT
            CustomerId,
            (
                COUNT(*) -
                (
                    SELECT
                        MIN(TransactionItemsCount)
                    FROM
                    (
                        SELECT
                            COUNT(*) as TransactionItemsCount
                        FROM
                            [wwi_perf].[Sale_Hash]
                        GROUP BY
                            CustomerId
                    ) X
                )
            ) / 100 as TransactionItemsCountBucket
        FROM
            [wwi_perf].[Sale_Hash]
        GROUP BY
            CustomerId
    ) T
GROUP BY
    T.TransactionItemsCountBucket
ORDER BY
    T.TransactionItemsCountBucket
```

2. Analyze the resulting execution plan. Take a close look to the **<materialized_view_candidates>** section which suggests possible materialized views you can create to improve the performance of the query.

```
1
2
```

```
<?xml version="1.0" encoding="utf-8"?>
<dsql_query number_nodes="5" number_distributions="60" number_distributions_per_n
ode="12">
<sql>SELECT
    T.TransactionItemsCountBucket
    ,count(*) as CustomersCount
FROM
```

```
    (
        SELECT
            CustomerId,
            (
                COUNT(*) -
                (
                    SELECT
                        MIN(TransactionItemsCount)
                    FROM
                    (
                        SELECT
                            COUNT(*) as TransactionItemsCount
                        FROM
                            [wwi_perf].[Sale_Hash]
                        GROUP BY
                            CustomerId
                    ) X
                )
            ) / 100 as TransactionItemsCountBucket
        FROM
            [wwi_perf].[Sale_Hash]
        GROUP BY
            CustomerId
    ) T
GROUP BY
    T.TransactionItemsCountBucket
ORDER BY
    T.TransactionItemsCountBucket</sql>
<materialized_view_candidates>
    <materialized_view_candidates with_constants="False">CREATE MATERIALIZED VIEW
View1 WITH (DISTRIBUTION = HASH([Expr0])) AS
SELECT [SQLPool01].[wwi_perf].[Sale_Hash].[CustomerId] AS [Expr0],
    COUNT(*) AS [Expr1]
FROM [wwi_perf].[Sale_Hash]
GROUP BY [SQLPool01].[wwi_perf].[Sale_Hash].[CustomerId]</
materialized_view_candidates>
```

3. Create the suggested materialized view:

```
1
2
3
4
5
6
```

```
CREATE MATERIALIZED VIEW
    mvTransactionItemsCounts
WITH
(
    DISTRIBUTION = HASH([CustomerId])
)
AS
SELECT
    CustomerId
    ,COUNT(*) AS ItemsCount
FROM
    [wwi_perf].[Sale_Hash]
GROUP BY
    CustomerId
```

4. Check the execution plan again:

```
```

```
EXPLAIN WITH_RECOMMENDATIONS
SELECT
    T.TransactionItemsCountBucket
    ,count(*) as CustomersCount
FROM
    (
        SELECT
            CustomerId,
            (
                COUNT(*) -
                (
                    SELECT
                        MIN(TransactionItemsCount)
                    FROM
                    (
                        SELECT
                            COUNT(*) as TransactionItemsCount
                        FROM
                            [wwi_perf].[Sale_Hash]
                        GROUP BY
                            CustomerId
                    ) X
                )
            ) / 100 as TransactionItemsCountBucket
        FROM
            [wwi_perf].[Sale_Hash]
        GROUP BY
            CustomerId
    ) T
GROUP BY
    T.TransactionItemsCountBucket
ORDER BY
```

```
T.TransactionItemsCountBucket
```

The resulting execution plan indicates now the use of the **`mvTransactionItemsCounts`** (the **`BROADCAST_MOVE`** distributed SQL operation) materialized view which provides improvements to the query execution time:

```
<?xml version="1.0" encoding="utf-8"?>
<dsql_query number_nodes="5" number_distributions="60" number_distributions_per_n
ode="12">
<sql>SELECT
    T.TransactionItemsCountBucket
    ,count(*) as CustomersCount
FROM
    (
        SELECT
            CustomerId,
            (
                COUNT(*) -
                (
                    SELECT
                        MIN(TransactionItemsCount)
                    FROM
                    (
                        SELECT
                            COUNT(*) as TransactionItemsCount
                        FROM
                            [wwi_perf].[Sale_Hash]
                        GROUP BY
                            CustomerId
                    ) X
                )
            ) / 100 as TransactionItemsCountBucket
        FROM
            [wwi_perf].[Sale_Hash]
        GROUP BY
            CustomerId
    ) T
GROUP BY
    T.TransactionItemsCountBucket
ORDER BY
    T.TransactionItemsCountBucket</sql>
<materialized_view_candidates>
    <materialized_view_candidates with_constants="False">CREATE MATERIALIZED VIEW
View1 WITH (DISTRIBUTION = HASH([Expr0])) AS
SELECT [SQLPool01].[wwi_perf].[Sale_Hash].[CustomerId] AS [Expr0],
    COUNT(*) AS [Expr1]
FROM [wwi_perf].[Sale_Hash]
GROUP BY [SQLPool01].[wwi_perf].[Sale_Hash].[CustomerId]</
materialized_view_candidates>
```