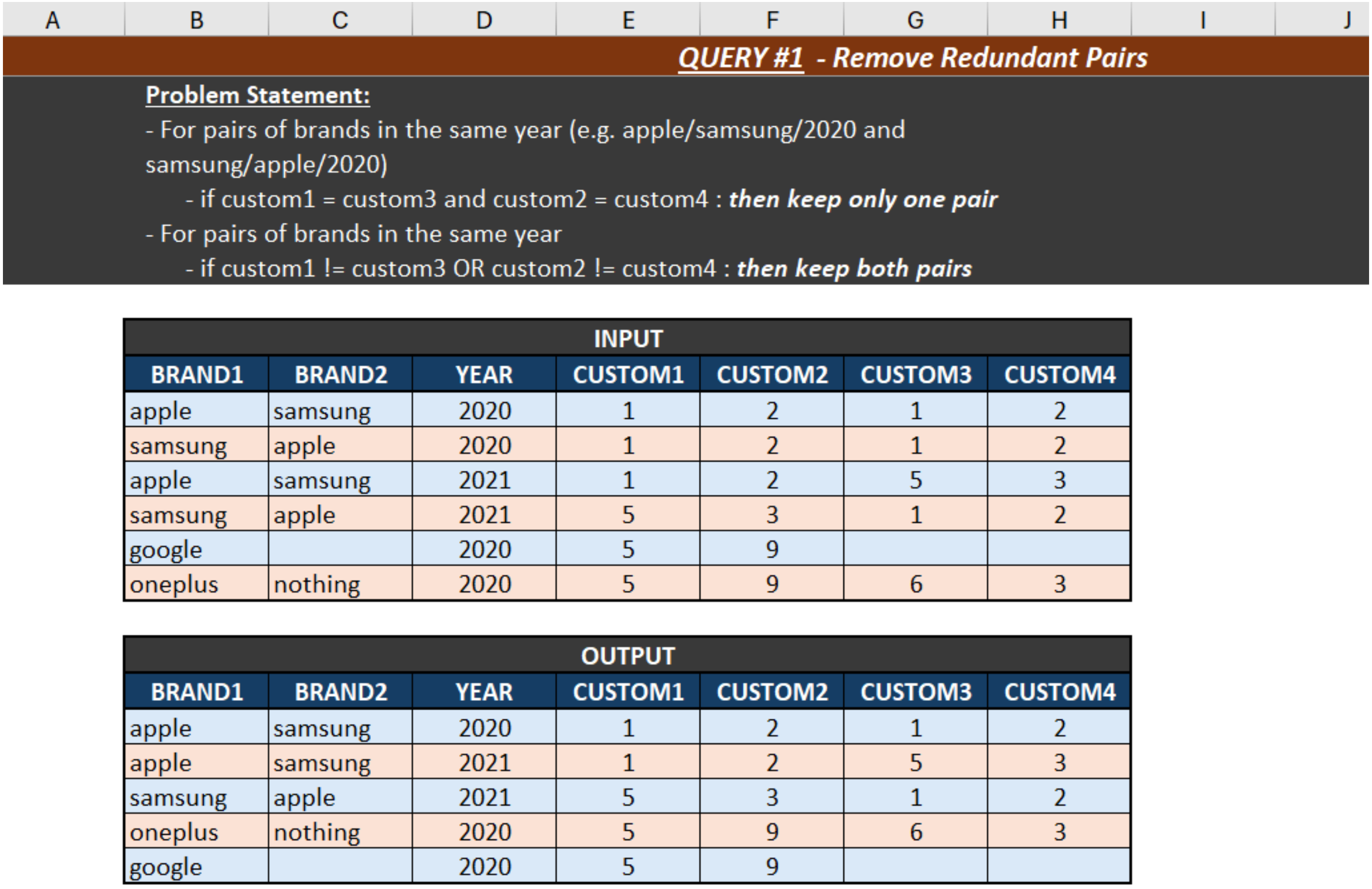


***QUERY #1* - Remove Redundant Pairs**

Problem Statement:

- For pairs of brands in the same year (e.g. apple/samsung/2020 and samsung/apple/2020)
    - if custom1 = custom3 and custom2 = custom4 : ***then keep only one pair***
- For pairs of brands in the same year
    - if custom1 != custom3 OR custom2 != custom4 : ***then keep both pairs***

INPUT

BRAND1	BRAND2	YEAR	CUSTOM1	CUSTOM2	CUSTOM3	CUSTOM4
apple	samsung	2020	1	2	1	2
samsung	apple	2020	1	2	1	2
apple	samsung	2021	1	2	5	3
samsung	apple	2021	5	3	1	2
google		2020	5	9		
oneplus	nothing	2020	5	9	6	3

OUTPUT

BRAND1	BRAND2	YEAR	CUSTOM1	CUSTOM2	CUSTOM3	CUSTOM4
apple	samsung	2020	1	2	1	2
apple	samsung	2021	1	2	5	3
samsung	apple	2021	5	3	1	2
oneplus	nothing	2020	5	9	6	3
google		2020	5	9		

Database and Table Setup:

A database named "SQL30DAYS" is created with a table named "Brands_day_1".

The table includes columns for brand names (BRAND1 and BRAND2), year (YEAR), and custom values (CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4).

Sample Data Insertion:

Sample data is added to the "Brands_day_1" table, including pairs of brands, years, and custom values.

Problem Statement Description:

The goal is to handle redundant pairs of brands in the same year based on specific conditions.

For pairs of brands in the same year:

If CUSTOM1 equals CUSTOM3 and CUSTOM2 equals CUSTOM4, keep only one pair.

If CUSTOM1 doesn't equal CUSTOM3 or CUSTOM2 doesn't equal CUSTOM4, keep both pairs.

Rows without pairs in the same year are retained.

Solution Implementation:

Common Table Expressions (CTEs) are used:

The first CTE (cte) computes a unique identifier (pair_id) for each pair of brands within the same year.

The second CTE (cte_rn) assigns a row number (rn) within each group of pair_id.

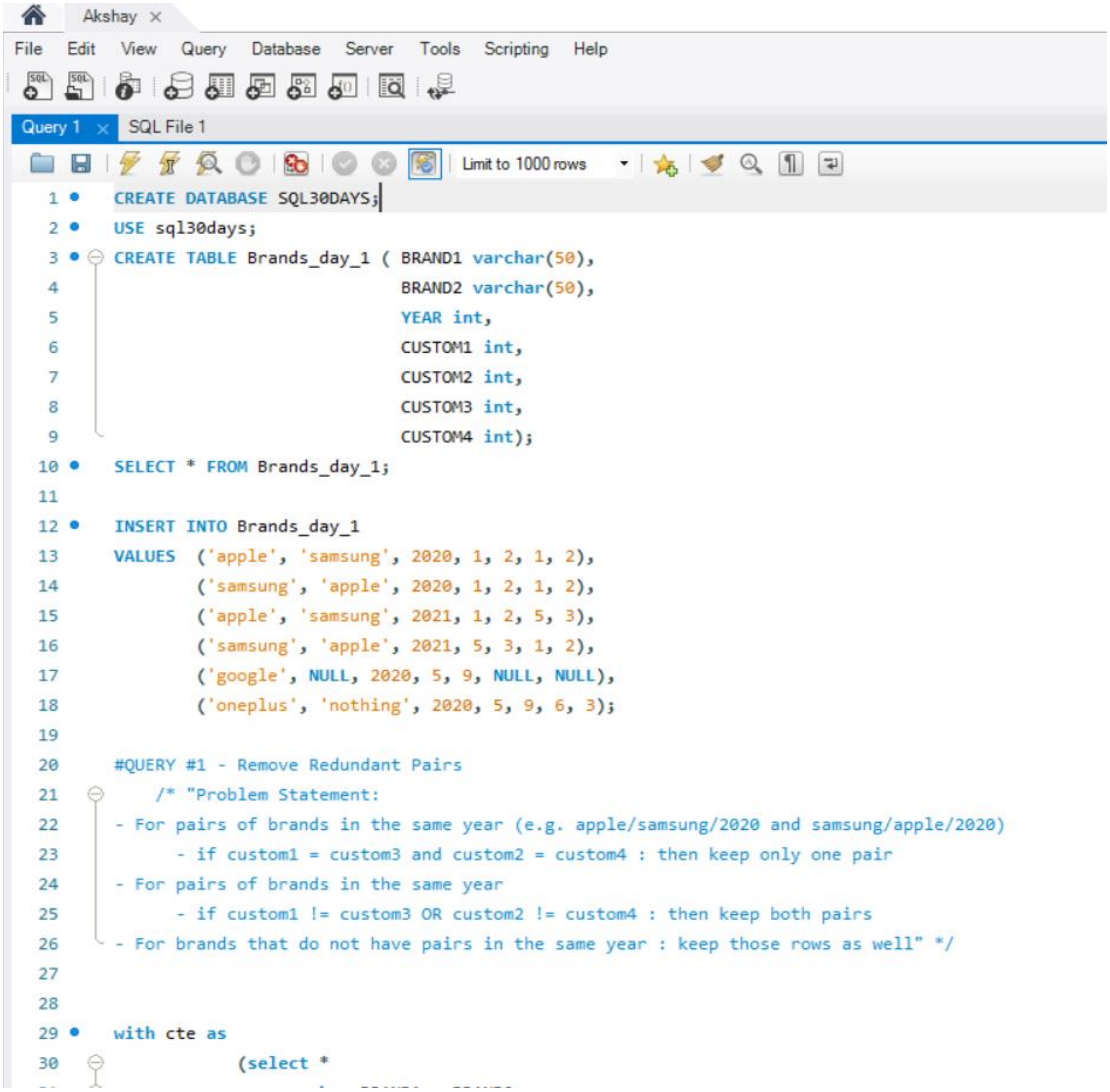
A SELECT statement retrieves data from cte_rn, filtering rows based on conditions:

It selects brand names, year, and custom values.

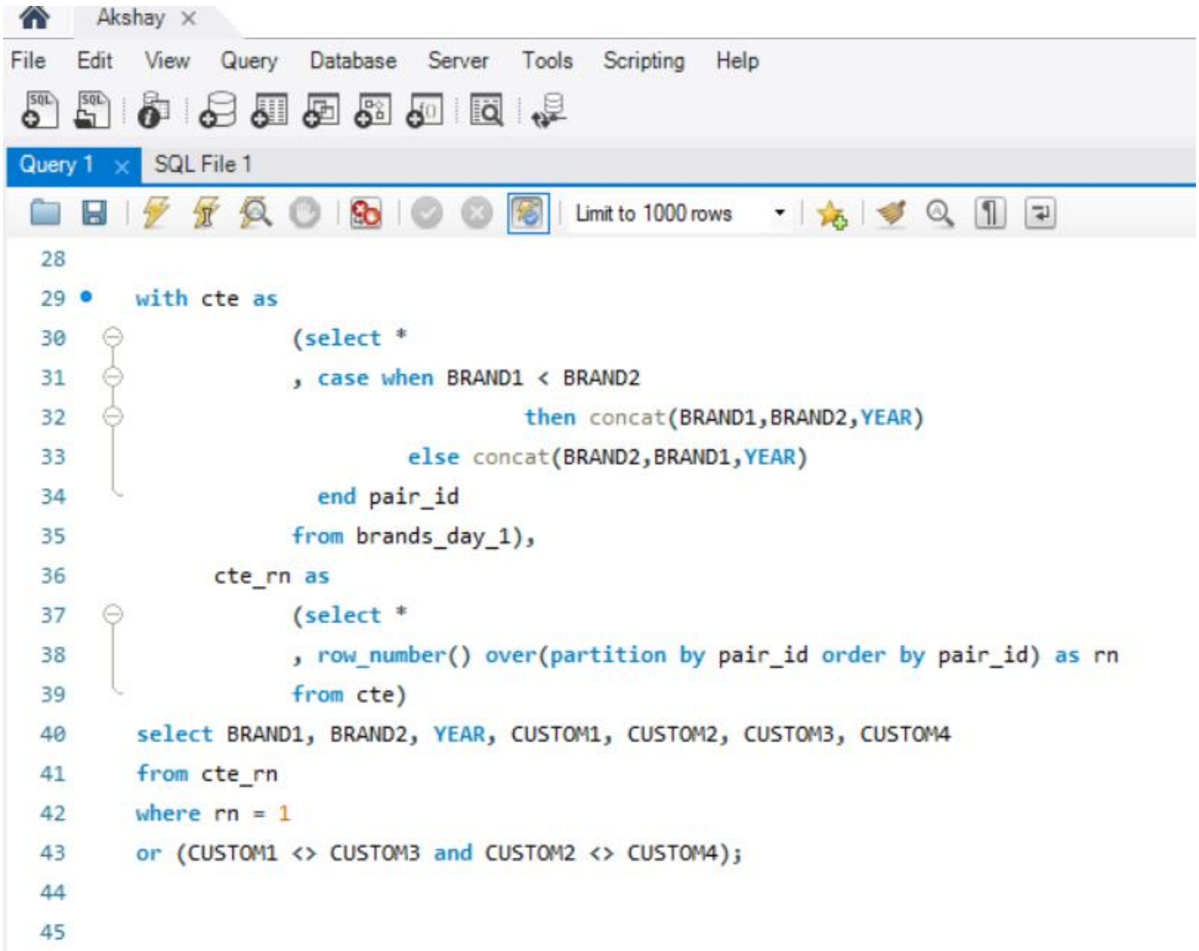
Rows with rn = 1 or where custom values differ between pairs within the same year are retained.

Final Query Description:

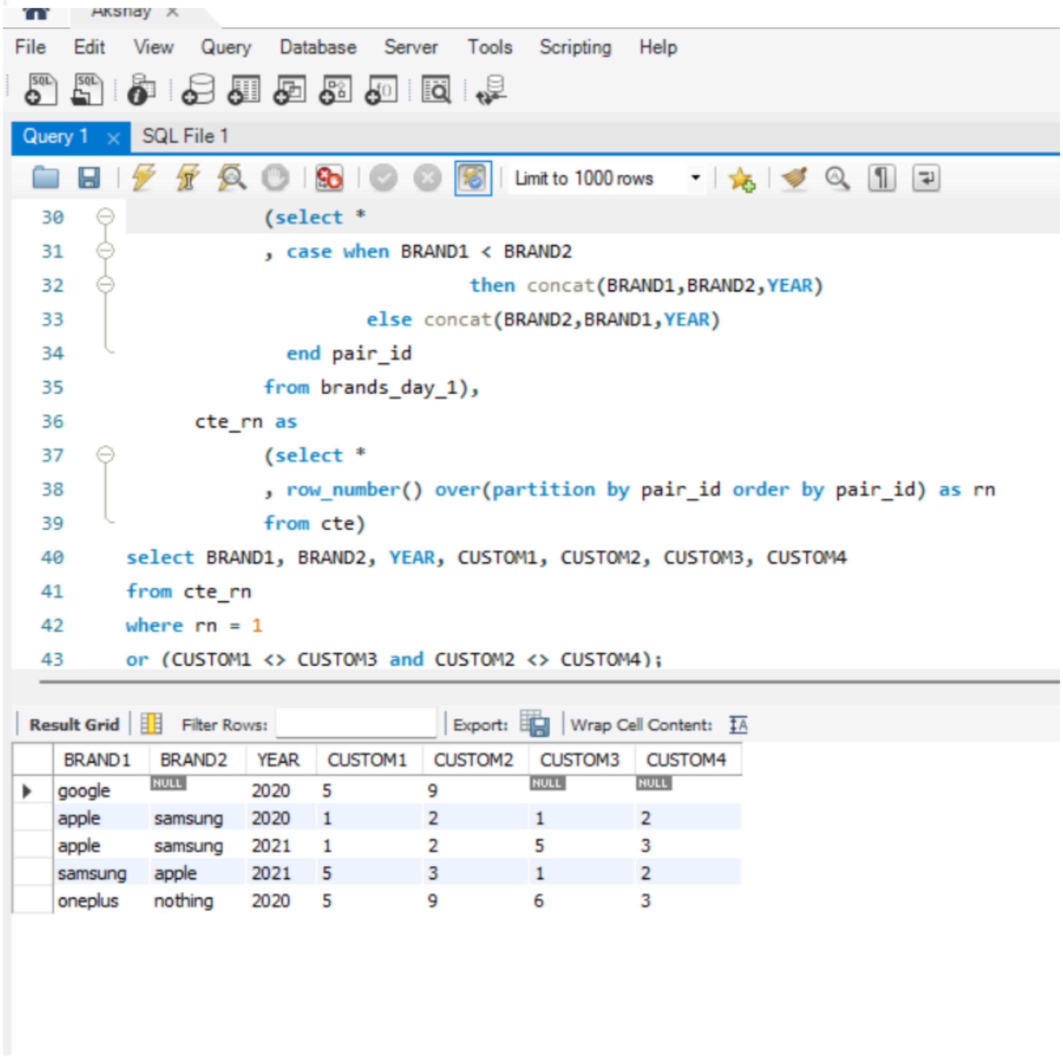
The query ensures that redundant pairs of brands are handled properly based on the conditions specified. It retains only one pair if custom values match, both pairs if they don't, and also retains rows without pairs in the same year.



```
CREATE DATABASE SQL30DAYS;
USE sql30days;
CREATE TABLE Brands_day_1 ( BRAND1 varchar(50),
                            BRAND2 varchar(50),
                            YEAR int,
                            CUSTOM1 int,
                            CUSTOM2 int,
                            CUSTOM3 int,
                            CUSTOM4 int);
SELECT * FROM Brands_day_1;

INSERT INTO Brands_day_1
VALUES  ('apple', 'samsung', 2020, 1, 2, 1, 2),
        ('samsung', 'apple', 2020, 1, 2, 1, 2),
        ('apple', 'samsung', 2021, 1, 2, 5, 3),
        ('samsung', 'apple', 2021, 5, 3, 1, 2),
        ('google', NULL, 2020, 5, 9, NULL, NULL),
        ('oneplus', 'nothing', 2020, 5, 9, 6, 3);

#QUERY #1 - Remove Redundant Pairs
    /* "Problem Statement:
- For pairs of brands in the same year (e.g. apple/samsung/2020 and samsung/apple/2020)
      - if custom1 = custom3 and custom2 = custom4 : then keep only one pair
- For pairs of brands in the same year
      - if custom1 != custom3 OR custom2 != custom4 : then keep both pairs
- For brands that do not have pairs in the same year : keep those rows as well" */


with cte as
            (select *
```



```

with cte as
            (select *
            , case when BRAND1 < BRAND2
                         then concat(BRAND1,BRAND2,YEAR)
                    else concat(BRAND2,BRAND1,YEAR)
              end pair_id
            from brands_day_1),
        cte_rn as
            (select *
            , row_number() over(partition by pair_id order by pair_id) as rn
            from cte)
select BRAND1, BRAND2, YEAR, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4
from cte_rn
where rn = 1
or (CUSTOM1 <> CUSTOM3 and CUSTOM2 <> CUSTOM4);


```



```
            (select *
            , case when BRAND1 < BRAND2
                         then concat(BRAND1,BRAND2,YEAR)
                    else concat(BRAND2,BRAND1,YEAR)
              end pair_id
            from brands_day_1),
        cte_rn as
            (select *
            , row_number() over(partition by pair_id order by pair_id) as rn
            from cte)
select BRAND1, BRAND2, YEAR, CUSTOM1, CUSTOM2, CUSTOM3, CUSTOM4
from cte_rn
where rn = 1
or (CUSTOM1 <> CUSTOM3 and CUSTOM2 <> CUSTOM4);
```

BRAND1	BRAND2	YEAR	CUSTOM1	CUSTOM2	CUSTOM3	CUSTOM4
google	NULL	2020	5	9	NULL	NULL
apple	samsung	2020	1	2	1	2
apple	samsung	2021	1	2	5	3
samsung	apple	2021	5	3	1	2
oneplus	nothing	2020	5	9	6	3