

In a BI scenario, arrays of multisets can be useful for representing and analyzing data with multiple values for a single attribute. Multiset arrays can be particularly relevant in scenarios where you have multivalued or hierarchical data. Here we deal with a common use case, namely product category management, where a product may belong to multiple categories.

Start up Code

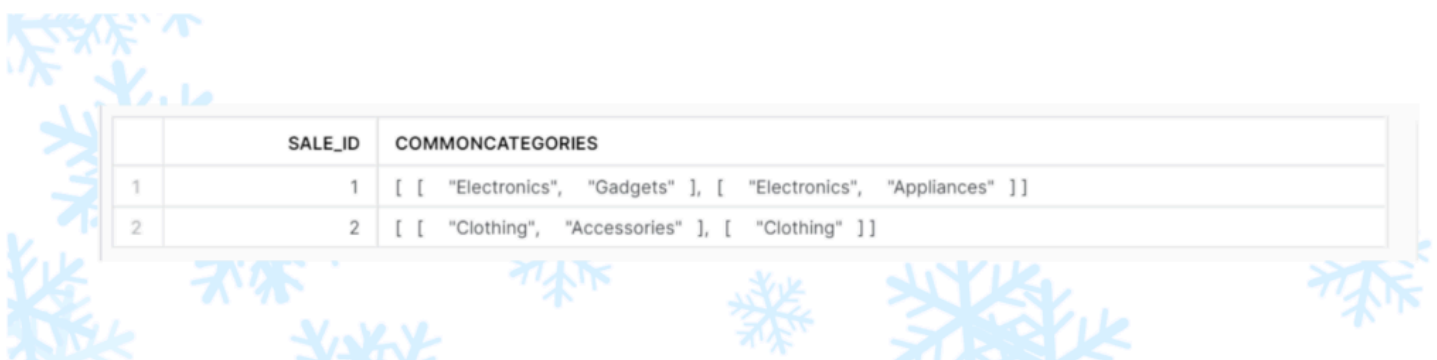
```
-- Create the Sales table
CREATE OR REPLACE TABLE Sales (
    Sale_ID INT PRIMARY KEY,
    Product_IDs VARIANT --INT
);

-- Inserting sample sales data
INSERT INTO Sales (Sale_ID, Product_IDs) SELECT 1,
PARSE_JSON('[1, 3]');-- Products A and C in the same sale
INSERT INTO Sales (Sale_ID, Product_IDs) SELECT 2,
PARSE_JSON('[2, 4]');-- Products B and D in the same sale

-- Create the Products table
CREATE OR REPLACE TABLE Products (
    Product_ID INT PRIMARY KEY,
    Product_Name VARCHAR,
    Product_Categories VARIANT --VARCHAR
);

-- Inserting sample data into Products
INSERT INTO Products (Product_ID, Product_Name,
Product_Categories) SELECT 1, 'Product A',
ARRAY_CONSTRUCT('Electronics', 'Gadgets');
INSERT INTO Products (Product_ID, Product_Name,
Product_Categories) SELECT 2, 'Product B',
ARRAY_CONSTRUCT('Clothing', 'Accessories');
INSERT INTO Products (Product_ID, Product_Name,
Product_Categories) SELECT 3, 'Product C',
ARRAY_CONSTRUCT('Electronics', 'Appliances');
INSERT INTO Products (Product_ID, Product_Name,
Product_Categories) SELECT 4, 'Product D',
ARRAY_CONSTRUCT('Clothing');
```

Using the tables that we have just created, how can we find common categories among products sold together in a single transaction? Like in the example shown below.



	SALE_ID	COMMONCATEGORIES
1	1	[["Electronics", "Gadgets"], ["Electronics", "Appliances"]]
2	2	[["Clothing", "Accessories"], ["Clothing"]]