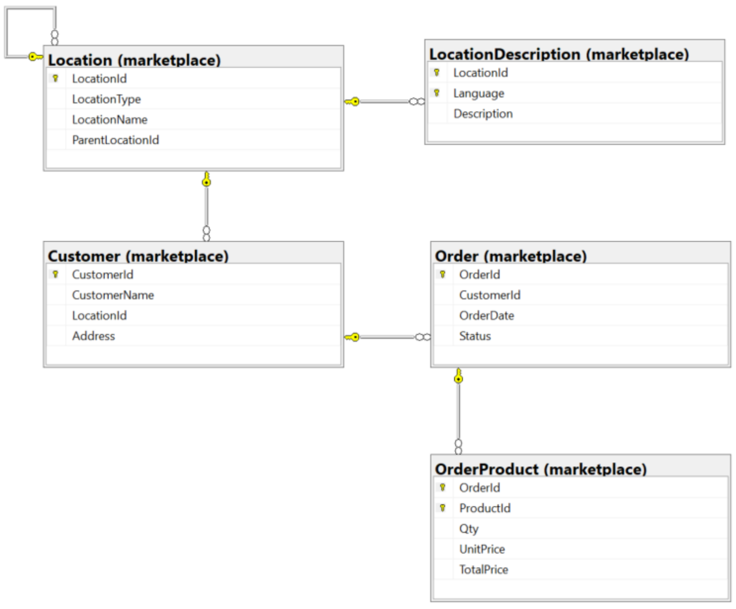
DB DEVELOPER PRACTICAL TEST

Read first: Pre-requirements

- SQL Server 2016 Express or higher (https://www.microsoft.com/en-us/sql-server/sql-server-downloads)
- SQL Server Management Studio (https://learn.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms).
- MongoDB Community Edition (v6.0)
- Mongo Shell
- Mongo Compass
- Mongo Command Line Database Tools

Let's start by implementing a database for our Marketplace.

For the data layer of our Marketplace we want to implement the following model:



1. Create the `Sales` database with basic `Order` information, using the script `1. SQL_marketplace.sql`. Verify that after the execution of the script this is the model of the database.

2. We need to perform operations of creation, update and deletion of records in the `OrderProduct` table. To achieve this, you need to complete the implementation of the stored procedure `marketplace.WriteOrderProduct`. Please note the following conditions:

   - Do not modify the parameters already defined.
   - It is possible to add, modify or delete records from the `OrderProduct` table only if the status of its parent in the `Order` table is `Open`.
   - The values for `ProductId`, `Qty`, `UnitPrice` and `TotalPrice` must be greater than zero.
   - Implement the validations that you consider necessary.
   - Use the `MERGE` statement to perform create, update and delete.

3. We also need to perform create, update and delete operations on the `Order` table. Help us to complete the implementation of the stored procedure `marketplace.WriteOrder` taking into account the following conditions:

   - Do not modify the parameters already defined.
   - Use the `MERGE` statement to perform the creation, update and deletion.
   - When a new `Order` is created, the identifier assigned to the newly created order must be returned.
   - Use `marketplace.WriteOrderProduct` if you consider it necessary.
   - Implement the validations that you consider necessary.

4. Now, it is necessary to be able to query the information of the tables Order and OrderProductComplete. To achieve this, you need to complete the implementation of the stored procedure marketplace.ReadOrder, taking into account the following conditions:

   - Do not modify the parameters already defined.
   - Implement the validations that you consider necessary.
   - This stored procedure must support at least the following 3 cases:

     A. Obtain through a single query the OrderId, CustomerId, CustomerName, OrderDate and Status of one or several Orders, given the CustomerId and/or OrderId and/or Status.

        - Name of the result: Order-Header.
        - It implements the indexes it considers appropriate.
        - Through this query we should be able to obtain:
          - All Orders
          - All the Orders for a specific customer
          - All Orders in a specific status for a specific client
          - All Orders in a specific status for all customers

     B. Obtain through a single query the OrderId, ProductId, Qty, UnitPrice and TotalPrice of one or several Orders, given the CustomerId and/or OrderId and/or Status.

        - Result Name: Order-Detail.
        - It implements the indexes it considers appropriate.
        - Through this query we should be able to obtain:
          - Details of all Orders
          - Details of all the Orders for a specific customer.
          - Details of all Orders in a specific status for a specific customer.
          - Details of all Orders in a specific status for all customers.

     C. Get in JSON format, through a single query, the historical information of Orders along with their Details, given the CustomerId and/or OrderId and/or Status.

        - Name of the Result: Order-History-Json.
        - It implements the indexes it considers appropriate.
        - Through this query we should be able to obtain:
          - All Orders and their Details
          - All Orders and their Details for a specific customer
          - All Orders and their Details in a specific state for a specific client
          - All Orders and their Details in a specific status for all customers.

     The result in JSON format must have the structure shown in Annex 1. Please note that:

        - _id corresponds to the Order #:
        - The value for CustomerLocationHierarchy must correspond according to the CustomerLocation hierarchy of the Order and depending on the Language indicated in the @Language parameter.
        - ProductsCount corresponds to the quantity of products included in the Order.
        - TotalOrder corresponds to the sum of the selling price of all the products in the Order.

💪 If you have experience using **tSQLt** for unit test creation, you can use it to validate the creation, update and deletion in the above points. Otherwise, please share with us evidence of the execution of each stored procedure along with the result obtained.

Now, our analytics area is interested in taking advantage of the data from our Marketplace but in order to use it in analytics tools they require us to deliver the information through MongoDB.

1. Environment preparation

   - Create a database named `Marketplace`.
   - Create the product collection and load the information provided in the `product.json` file.
   - Create the `orderHistory` collection and load the information provided in the `orderHistory.json` file.

2. Perform a query through which to obtain the `ID` and `Name` of the 3 best selling products during January 2023, taking into account that:

   - For each `product` show the quantity of products sold, and the sum of sales value.
   - The result should be presented in order x total sales price from highest to lowest.
   - Create the indexes you consider appropriate.

   The expected result is the following, according to the information loaded in step 1:

```
[
    {
        "TotalSoldItems": 9,
        "TotalPrice": 90,
        "ProductId": 66,
        "ProductName": "Divi Engine String Bag (Big Logo)"
    },
    {
        "TotalSoldItems": 4,
        "TotalPrice": 80,
        "ProductId": 114,
        "ProductName": "WordPress Tee - Small"
    },
    {
        "TotalSoldItems": 7,
        "TotalPrice": 35,
        "ProductId": 117,
        "ProductName": "Mens Divi Hoodie"
    }
]
```

Share with us evidence of the process, instruction, method, command used, etc., and where applicable, the result obtained.

ANNEX 1. EXAMPLE OF RESULT FOR ORDERS AND THEIR DETAILS HISTORICAL INFORMATION

```
[
    {
        "_id": 1,
        "OrderDate": "2023-01-15T00:00:00",
        "CustomerId": 1,
        "CustomerName": "Tom Cruise",
        "CustomerLocationCity": "New York",
        "CustomerLocationHierarchy": "America | United States | New York",
        "CustomerAddress": "1100 Main Street 248 PH98, Manhattan, NYC",
        "Products": [
            {
                "ProductId": 66,
                "Qty": 8,
                "UnitPrice": 10.0,
                "TotalPrice": 80.0
            },
            {
                "ProductId": 100,
                "Qty": 3,
                "UnitPrice": 10.0,
                "TotalPrice": 30.0
            },
            {
                "ProductId": 117,
                "Qty": 7,
                "UnitPrice": 5.0,
                "TotalPrice": 35.0
            }
        ],
        "ProductsCount": 3,
        "TotalOrder": 145.0
    },
    {
        "_id": 2,
        "OrderDate": "2023-01-20T00:00:00",
        "CustomerId": 3,
        "CustomerName": "Shakira Mebarak",
        "CustomerLocationCity": "Barcelona",
        "CustomerLocationHierarchy": "Europe | Spain | Barcelona",
        "CustomerAddress": "Ciudad Diagonal, Esplugas de Llobregat",
        "Products": [
            {
                "ProductId": 132,
                "Qty": 1,
                "UnitPrice": 14.0,
                "TotalPrice": 14.0
            }
        ],
        "ProductsCount": 1,
        "TotalOrder": 14.0
    },
    {
        "_id": 3,
        "OrderDate": "2023-01-30T00:00:00",
        "CustomerId": 1,
        "CustomerName": "Tom Cruise",
        "CustomerLocationCity": "New York",
        "CustomerLocationHierarchy": "America | United States | New York",
        "CustomerAddress": "1100 Main Street 248 PH98, Manhattan, NYC",
        "Products": [
            {
                "ProductId": 66,
                "Qty": 1,
                "UnitPrice": 10.0,
                "TotalPrice": 10.0
            },
            {
                "ProductId": 114,
                "Qty": 4,
                "UnitPrice": 20.0,
                "TotalPrice": 80.0
            }
        ],
        "ProductsCount": 2,
        "TotalOrder": 90.0
    },
    ...
]
```