**Maliheh Garoosiha**

**Part1**

As a data engineer specializing in XML, my primary focus revolves around parsing, querying, and transforming XML data using XPath and XQuery. Within our group assignment, I've taken on the responsibility of extracting and structuring data to facilitate further analysis.

To begin our data processing journey, I've utilized the BaseX application to import the 'electronicsdb.xml' file.

1-Retrieve all products in the 'Smartphones' category or supplied by SupplierID 50.

I'm working with products from our 'electronicsdb' dataset. First, I locate the CategoryID of the 'Smartphones' category. Then, I filter products based on two conditions: ensuring their CategoryID matches that of the 'Smartphones' category or their SupplierID is 50. Finally, I extract the names of these filtered products.

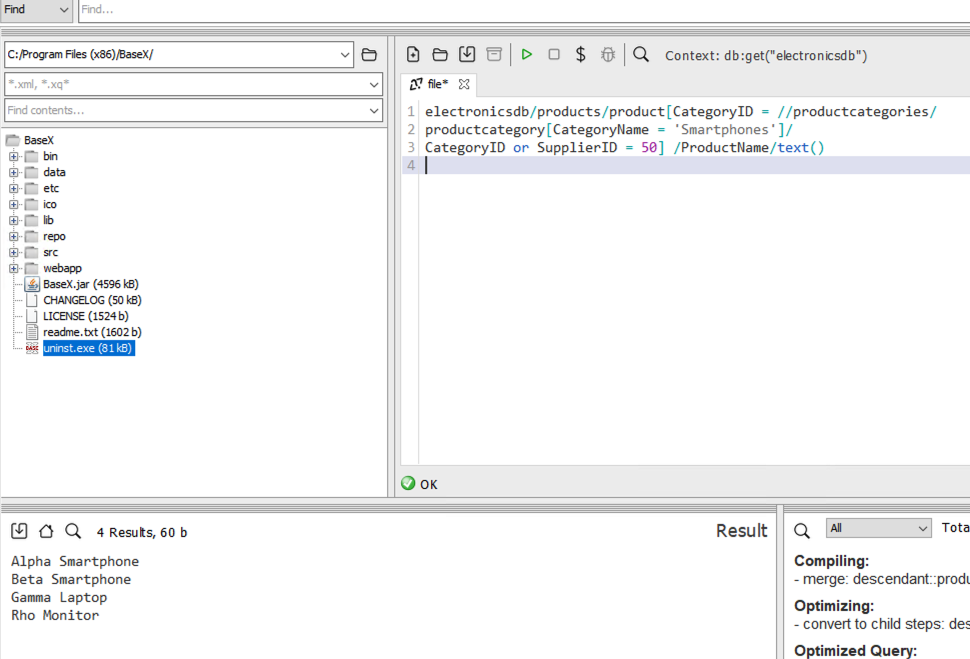
**Xpath Code:**

electronicsdb/products/product[CategoryID = //productcategories/

productcategory[CategoryName = 'Smartphones']/

CategoryID or SupplierID = 50] /ProductName/text()

**Output**:



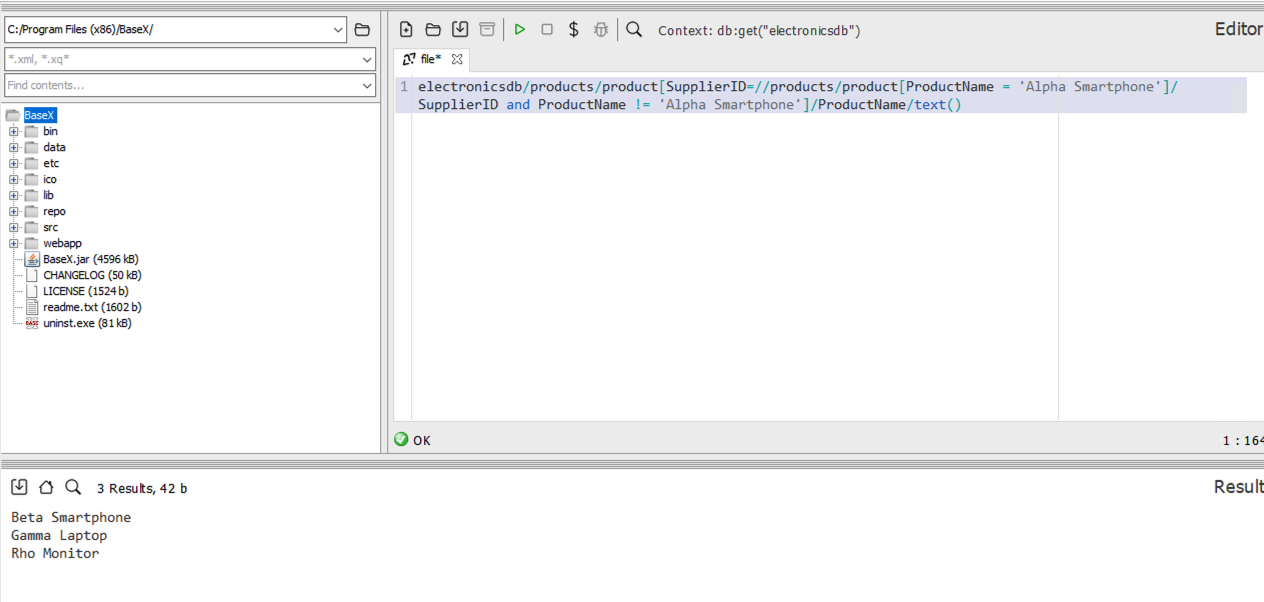
2- Find products sharing the same supplier as 'Alpha Smartphone'.

**Xpath Code:**

electronicsdb/products/product[SupplierID=//products/product[ProductName = 'Alpha Smartphone']/SupplierID and ProductName != 'Alpha Smartphone']/ProductName/text()

I'm focusing on products within our 'electronicsdb' dataset. First, I locate the SupplierID of the 'Alpha Smartphone' product. Then, I filter products based on two conditions: ensuring their SupplierID matches that of the 'Alpha Smartphone' and excluding the 'Alpha Smartphone' itself. Finally, I extract the names of these filtered products.

**Output:**



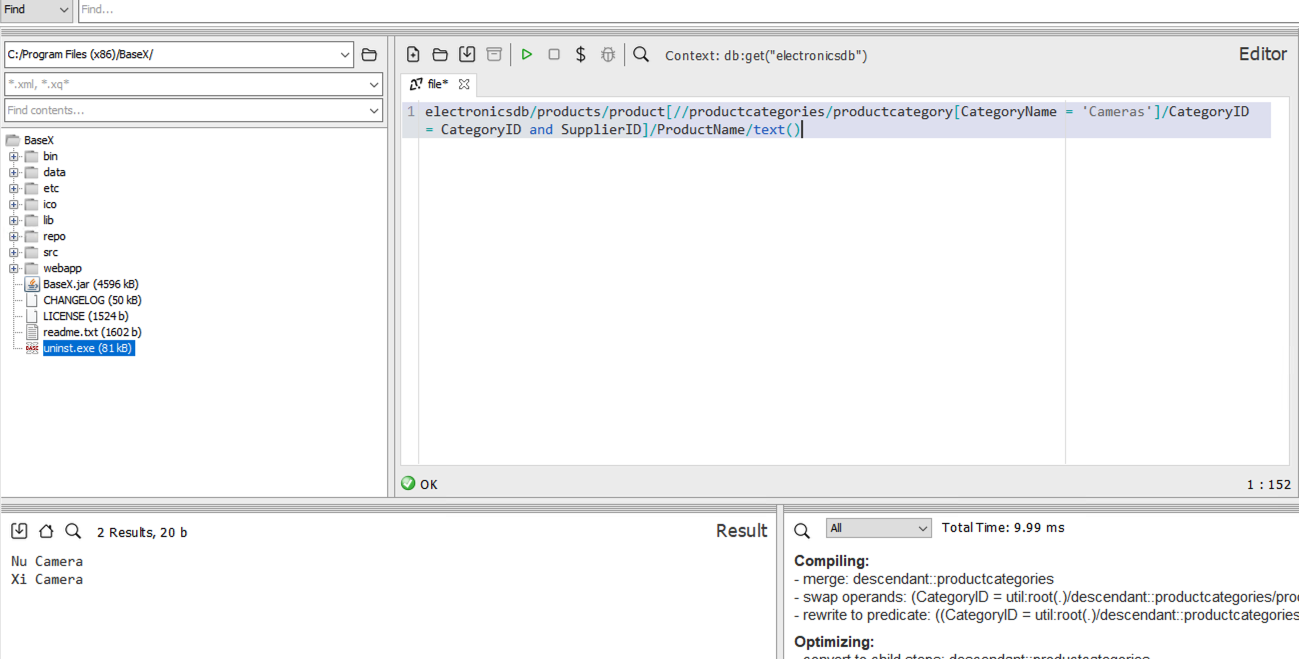
3-Identify 'Cameras' category products supplied by at least one supplier.

**XQuery (FLWOR) Code:**

electronicsdb/products/product[//productcategories/productcategory[CategoryName = 'Cameras']/CategoryID = CategoryID and SupplierID]/ProductName/text()

I'm targeting products from our 'electronicsdb' dataset. First, I navigate through the product categories to find the 'Cameras' category and grab its ID. Then, I filter the products based on two conditions: ensuring their CategoryID matches that of the 'Cameras' category and that they have a valid SupplierID. Finally, I extract the names of these filtered products.

**output:**



4-List all products alongside their category and supplier names.

So, in this code, I go through each product in our "electronicsdb" dataset. Then, for each product, I find its category by matching the product's CategoryID with the CategoryID in our product categories. After that, I grab the category name.Then, I move on to fetch the supplier name by matching the product's SupplierID with the SupplierID in our suppliers dataset.Lastly, I put together a string for each product, including its name, category, and supplier information. This string is what I get as a result after running the query.

**XQuery (FLWOR) Code:**

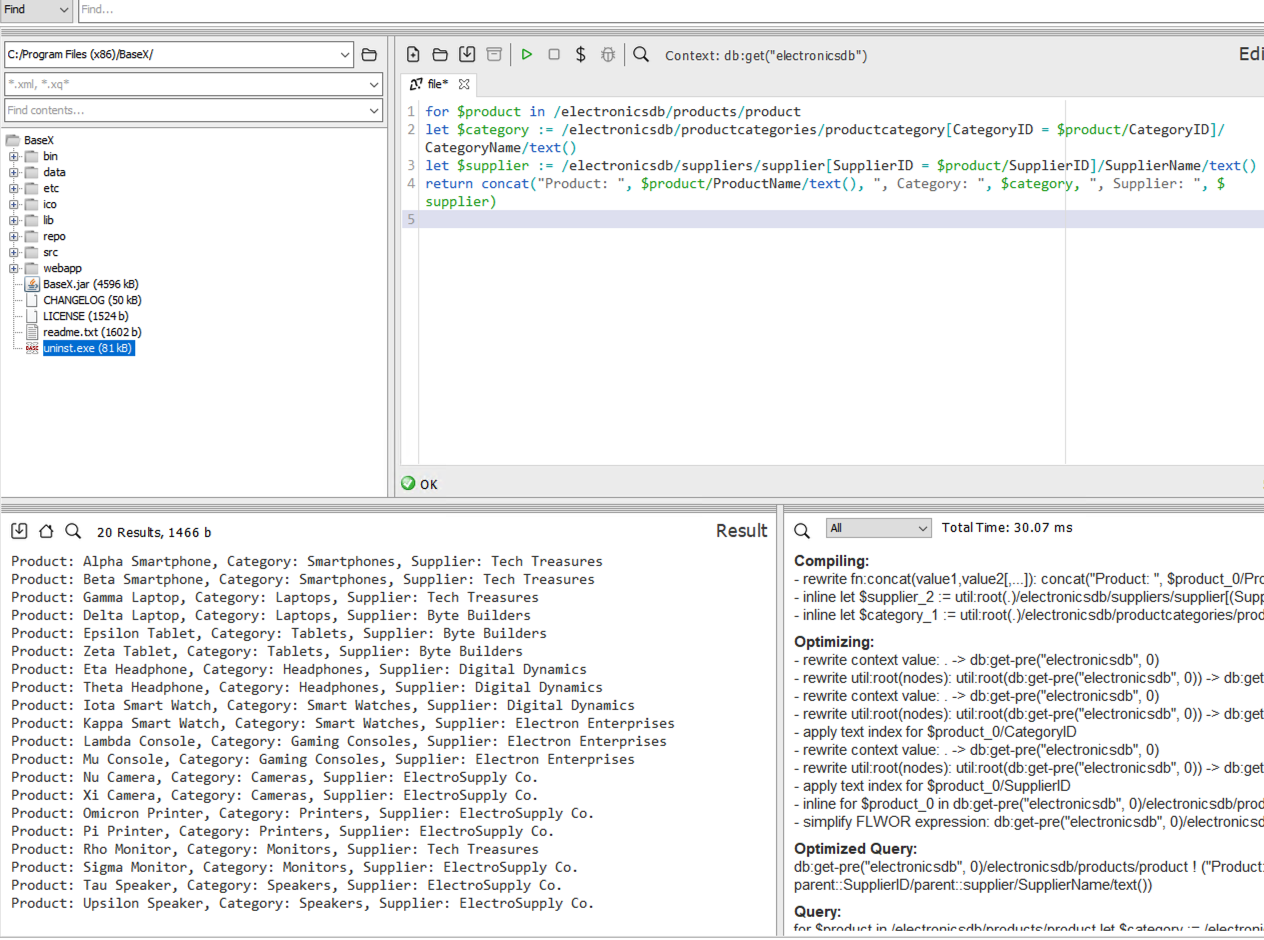
for $product in /electronicsdb/products/product

let $category := /electronicsdb/productcategories/productcategory[CategoryID = $product/CategoryID]/CategoryName/text()

let $supplier := /electronicsdb/suppliers/supplier[SupplierID = $product/SupplierID]/SupplierName/text()

return concat("Product: ", $product/ProductName/text(), ", Category: ", $category, ", Supplier: ", $supplier)

**Output:**



5-Compute the total stock quantity for each product.

In this code, I start by looping through each product in our "electronicsdb" dataset. For each product, I calculate the total stock quantity by summing up the quantities in stock for all instances of that product across all store inventory items.Then, I construct a string for each product, including its name and the total stock quantity. This string is what I get as the result after running the query.

**XQuery (FLWOR) Code:**

for $product in /electronicsdb/products/product

let $total\_stock := sum(/electronicsdb/storeinventory/storeinventoryitem[ProductID = $product/ProductID]/QuantityInStock)

return concat("Product: ", $product/ProductName/text(), ", Total Stock Quantity: ", $total\_stock)

**Output**:

