

**Instituto Tecnológico de Culiacán**



**Carrera: Ingeniería en Sistemas Computacionales**

**Materia: Temas Selectos de Bases de Datos**

**Alumnos:**

**José Alfredo García Aguilar**

**César Alfredo Astorga Ochoa**

**Trabajo: U4 Proyecto XML**

**Fecha: 30-Mayo-2022**

**Horario de clase: 05:00 - 06:00 pm**

**Profesor: Daniel Esparza Soto**

- Migrar las tablas products, categories, suppliers (Eliminar el campo ntext e image de las tablas). Especificar paso por paso como realizaron la migración.

--Eliminar los campos ntext e image de las tablas Products, Categories y Suppliers

--Campo image y ntext en categories

```
alter table categories drop column picture
```

```
go
```

```
alter table categories drop column Description
```

```
go
```

--Campo ntext en suppliers

```
alter table suppliers drop column HomePage
```

```
go
```

--Migrar las tablas Products, Categories y Suppliers

--Products

```
select
```

```
productid "@clave", supplierid "@Proveedor", categoryid "@Categoria",
```

```
Nombre = productname,
```

```
QuantityperUnit "Unidad/CantidadPorUnidad",
```

```
unitprice "Unidad/PrecioUnitario",
```

```
UnitsInStock "Unidad/UnitInStock",
```

```
UnitsOnOrder "Unidad/UnidadesOrden",
```

```
ReorderLevel "Unidad/Reorden",
```

```
Descuento = Discontinued
```

```
from Products
```

```
for xml path('Producto'), root('Productos'), elements
```

--Categories

```
select
```

```
categoryid "@Clave",
```

```
Categoria = categoryname
```

```
from categories
```

```
for xml path('Categoria'), root('Categorias'), elements
```

--Suppliers

```
Select
```

```
SupplierId "@clave",
```

```
Compañia=Companyname,
```

```
contactname "Contacto/NombreCompañia",
```

```
ContactTitle "Contacto/TituloCompañia",
```

```
Address "Domicilio/Calle",
```

```
City "Domicilio/Ciudad",
```

```
Region "Domicilio/Region",
```

```
PostalCode "Domicilio/CodigoPostal",
```

```
Country "Domicilio/Pais",
```

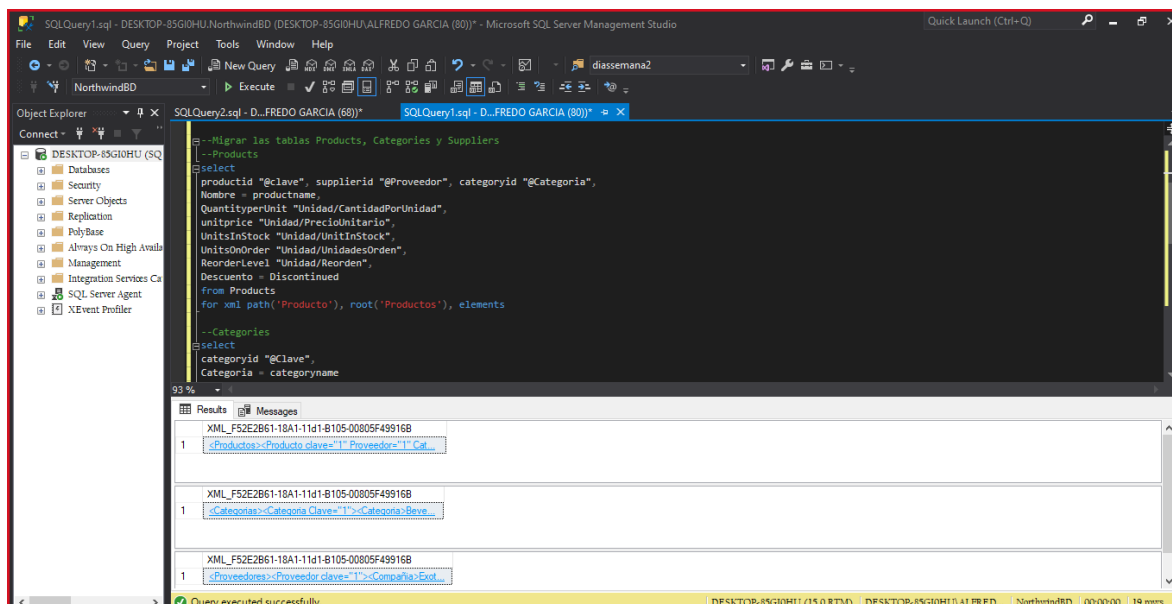
```
Phone "Contacto/Telefono",
```

```
Fax "Contacto/Fax"
```

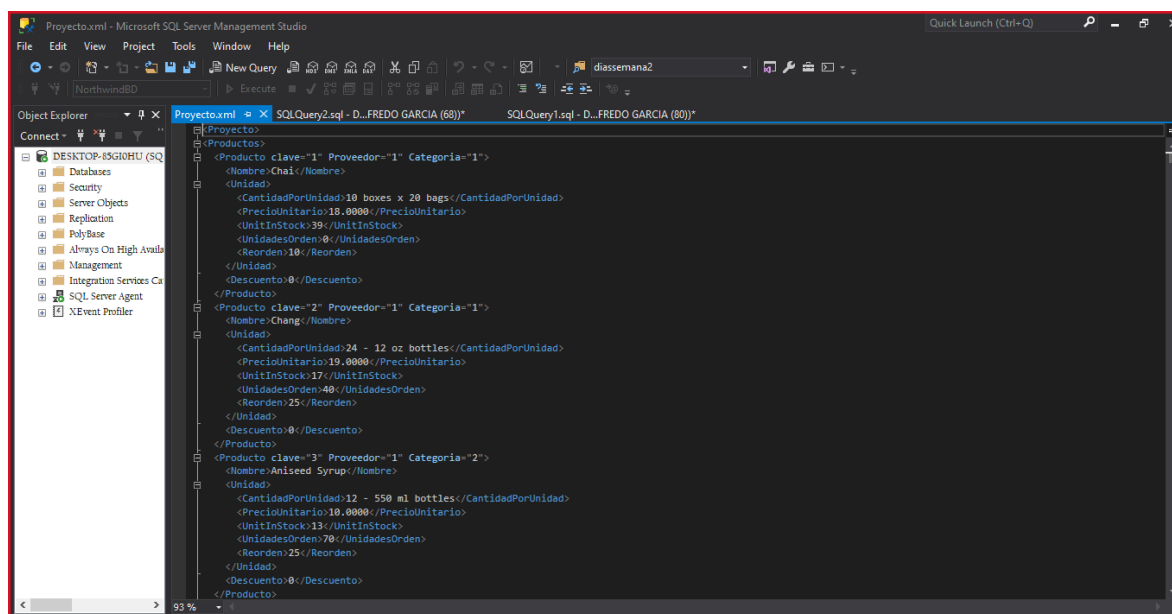
```
from Suppliers
```

```
for xml path('Proveedor'), ROOT('Proveedores'), elements
```

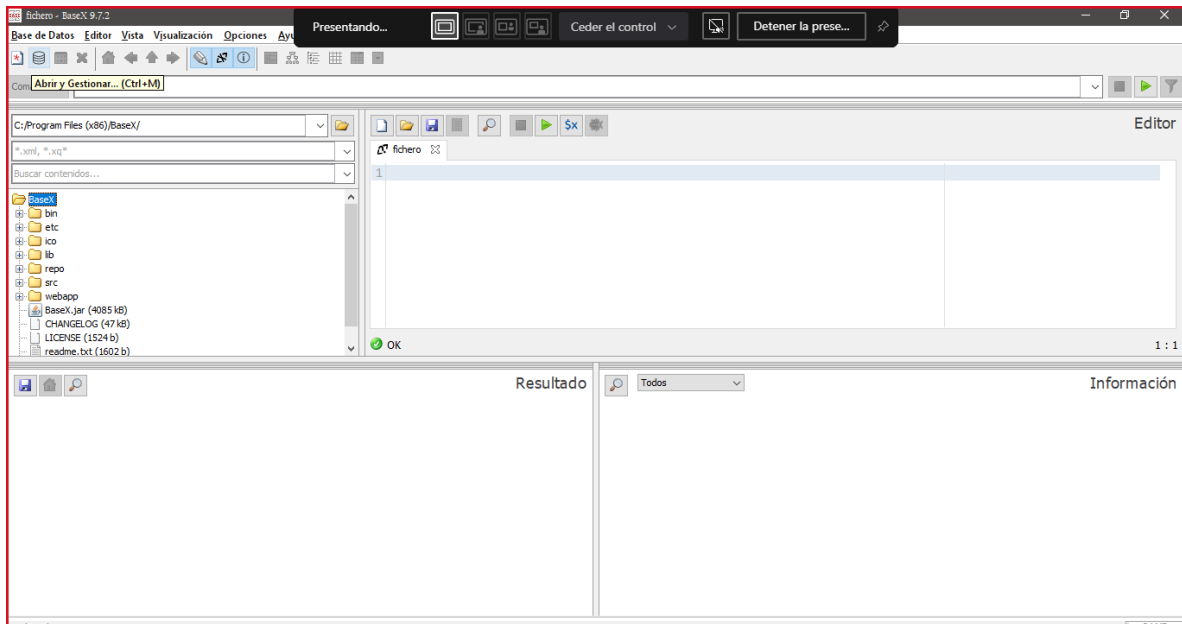
## Consultas:



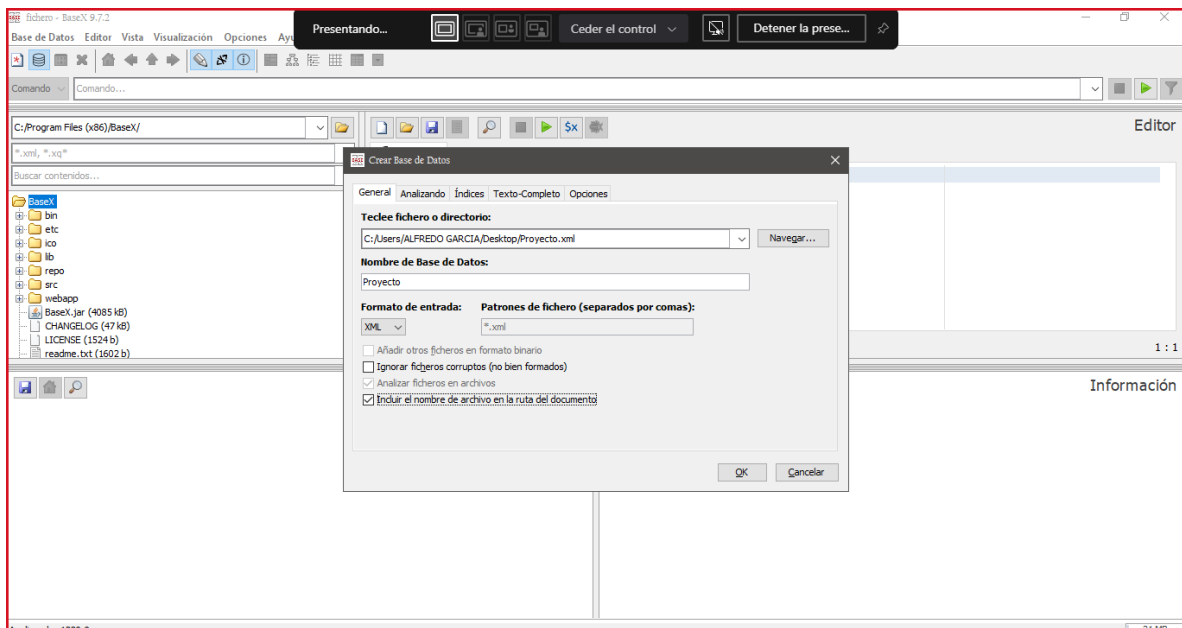
Juntamos las 3 consultas en una, le ponemos de nombre “Proyecto” y lo guardamos como archivo tipo .xml



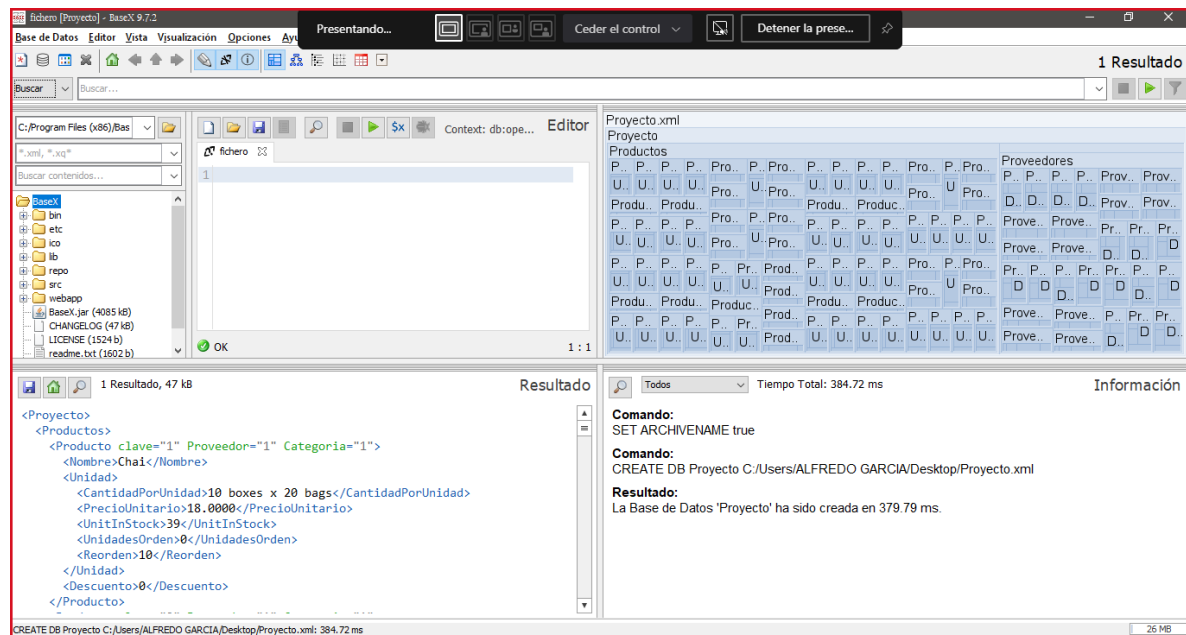
Ya estando en BaseX, damos en Abrir y Gestionar, seleccionamos nuestro archivo .xml



Elegimos Proyecto.xml, marcamos la casilla para incluir el nombre y damos Ok



Nos mostrará así:



- Realizar consultas sobre cada tabla individual con XPath, Xquery o el lenguaje de consulta nativo.

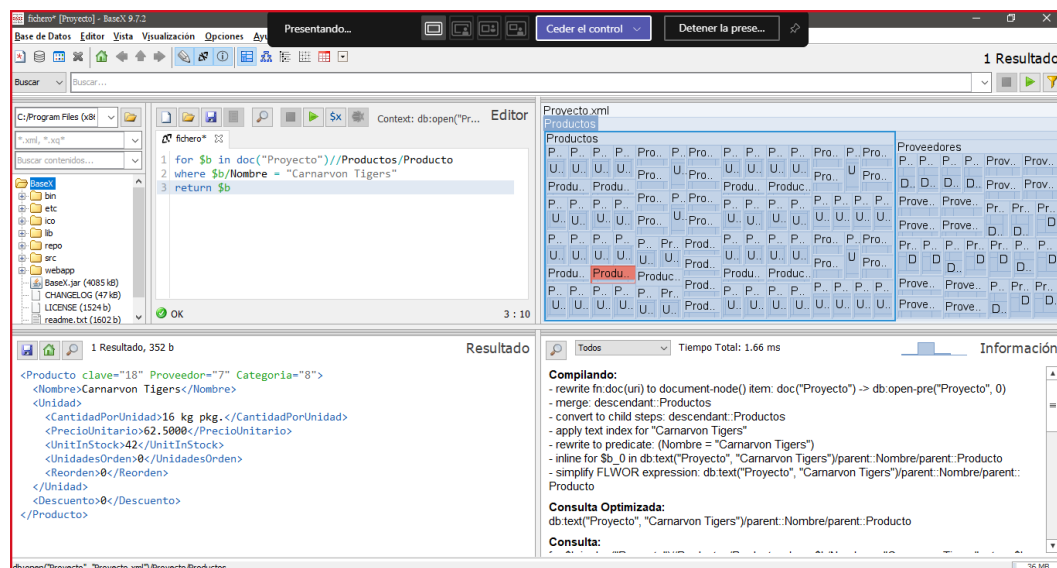
Elegimos Xquery

Products:

for \$b in doc("Proyecto")//Productos/Producto

where \$b/Nombre = "Carnarvon Tigers"

return \$b



for \$b in doc("Proyecto")//Productos/Producto

where \$b/Nombre = "Gorgonzola Telino"

return \$b/Unidad

The screenshot shows the BaseX 9.7.2 interface. The 'Editor' pane contains the following XQuery:

```
1 for $b in doc("Proyecto")//Productos/Producto
2 where $b/Nombre = "Gorgonzola Telino"
3 return $b/Unidad
```

The 'Resultado' pane shows 1 result, 215 b. The 'Información' pane displays the compilation and optimization details:

**Compilando:**

- rewrite fn:doc(uri) to document-node() item: doc("Proyecto") -> db:open-pre("Proyecto", 0)
- merge: descendant:Productos
- convert to child steps: descendant:Productos
- apply text index for "Gorgonzola Telino"
- rewrite to predicate: (Nombre = "Gorgonzola Telino")
- inline for \$b\_0 in db:text("Proyecto", "Gorgonzola Telino")/parent:Nombre/parent:Producto
- simplify FLWOR expression: db:text("Proyecto", "Gorgonzola Telino")/parent:Nombre/parent:Producto/Unidad

**Consulta Optimizada:**

db:text("Proyecto", "Gorgonzola Telino")/parent:Nombre/parent:Producto/Unidad

**Consulta:**

db:open("Proyecto", "Proyecto.xml")/Proyecto

## Categories:

for \$b in doc("Proyecto")//Categorias/Categoria

where \$b/Categoria = "Seafood"

return \$b

The screenshot shows the BaseX 9.7.2 interface. The 'Editor' pane contains the following XQuery:

```
1 for $b in doc("Proyecto")//Categorias/Categoria
2 where $b/Categoria = "Seafood"
3 return $b
```

The 'Resultado' pane shows 1 result, 69 b. The 'Información' pane displays the compilation and optimization details:

**Compilando:**

- rewrite fn:doc(uri) to document-node() item: doc("Proyecto") -> db:open-pre("Proyecto", 0)
- merge: descendant:Categories
- convert to child steps: descendant:Categories
- rewrite to predicate: (Categoria = "Seafood")
- inline for \$b\_0 in db:open-pre("Proyecto", 0)/\*:Proyecto/\*:Categories/Categoria[(Categoria = "Seafood")]
- simplify FLWOR expression: db:open-pre("Proyecto", 0)/\*:Proyecto/\*:Categories/Categoria[(Categoria = "Seafood")]

**Consulta Optimizada:**

db:open-pre("Proyecto", 0)/\*:Proyecto/\*:Categories/Categoria[(Categoria = "Seafood")]

**Consulta:**

db:open-pre("Proyecto", 0)/\*:Proyecto/\*:Categories/Categoria[(Categoria = "Seafood")]

for \$b in doc("Proyecto")//Categorias/Categoria

where \$b/Categoria = "Grains/Cereals"

return \$b

The screenshot shows the BaseX 9.7.2 interface. The editor window contains the following XQuery:

```
1 for $b in doc("Proyecto")//Categorias/Categoria
2 where $b/Categoria = "Grains/Cereals"
3 return $b
```

The results pane shows 1 result, 76 b. The XML output is:

```
<Categoria clave="5">
  <Categoria>Grains/Cereals</Categoria>
</Categoria>
```

The right pane shows the XQuery compilation process and the optimized query:

```
Consulta Optimizada:
db:open-pre("Proyecto", 0)/Proyecto/Categorias/Categoria[
  (Categoria = "Grains/Cereals")
]
```

Suppliers:

for \$b in doc("Proyecto")//Proveedores/Proveedor

where \$b/Compañia = "Exotic Liquids"

return \$b

The screenshot shows the BaseX 9.7.2 interface. The editor window contains the following XQuery:

```
1 for $b in doc("Proyecto")//Proveedores/Proveedor
2 where $b/Compañia = "Exotic Liquids"
3 return $b
```

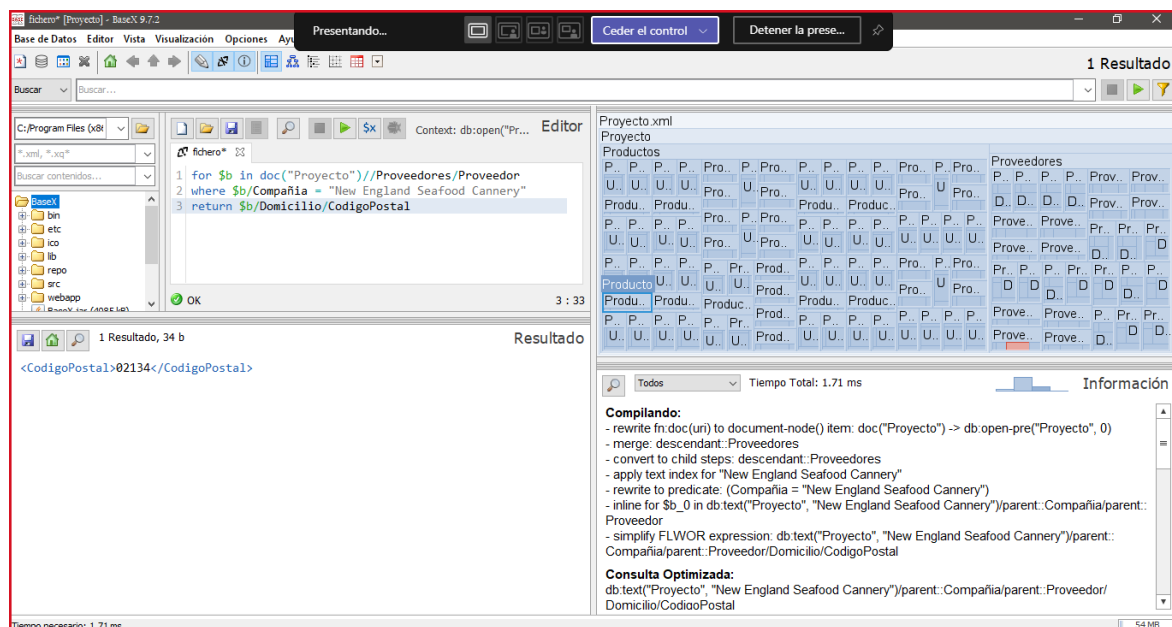
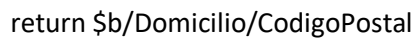
The results pane shows 1 result, 449 b. The XML output is:

```
<Proveedor clave="1">
  <Compañia>Exotic Liquids</Compañia>
  <Contacto>
    <NombreCompañia>Charlotte Cooper</NombreCompañia>
    <TituloCompañia>Purchasing Manager</TituloCompañia>
  </Contacto>
  <Domicilio>
    <Calle>49 Gilbert St.</Calle>
    <Ciudad>London</Ciudad>
    <CodigoPostal>EC1 4SD</CodigoPostal>
    <País>UK</País>
  </Domicilio>
  <Contacto>
    <Telefono>(171) 555-2222</Telefono>
  </Contacto>
</Proveedor>
```

The right pane shows the XQuery compilation process and the optimized query:

```
Consulta Optimizada:
db:text("Proyecto", "Exotic Liquids")/parent::Compañia/parent::Proveedor
```

return \$b/Domicilio





- Realizar una consulta combinando las 3 tablas con XPath, Xquery o el lenguaje de consulta nativo.

Elegimos Xquery

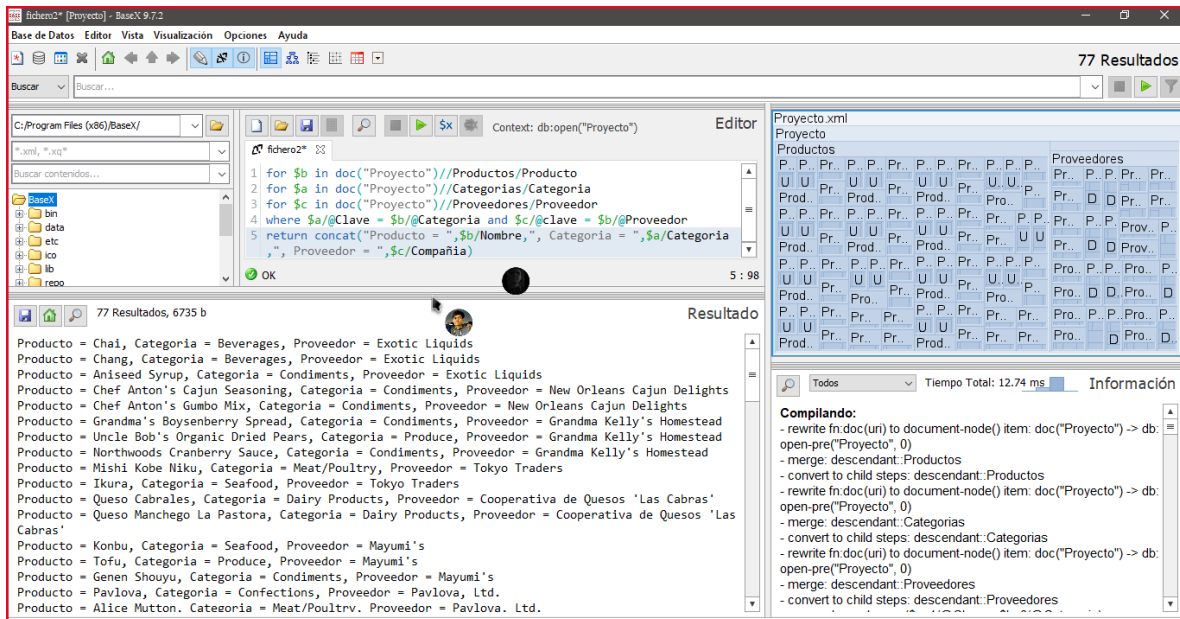
for \$b in doc("Proyecto")//Productos/Producto

for \$a in doc("Proyecto")//Categorias/Categoria

for \$c in doc("Proyecto")//Proveedores/Proveedor

where \$a/@Clave = \$b/@Categoria and \$c/@clave = \$b/@Proveedor

return concat("Producto = ", \$b/Nombre, ", Categoria = ", \$a/Categoria, ", Proveedor = ", \$c/Compañia)



- Verificar y especificar si se puede Insertar un registro en cada tabla.

Un insert por tabla ☺

Categories:

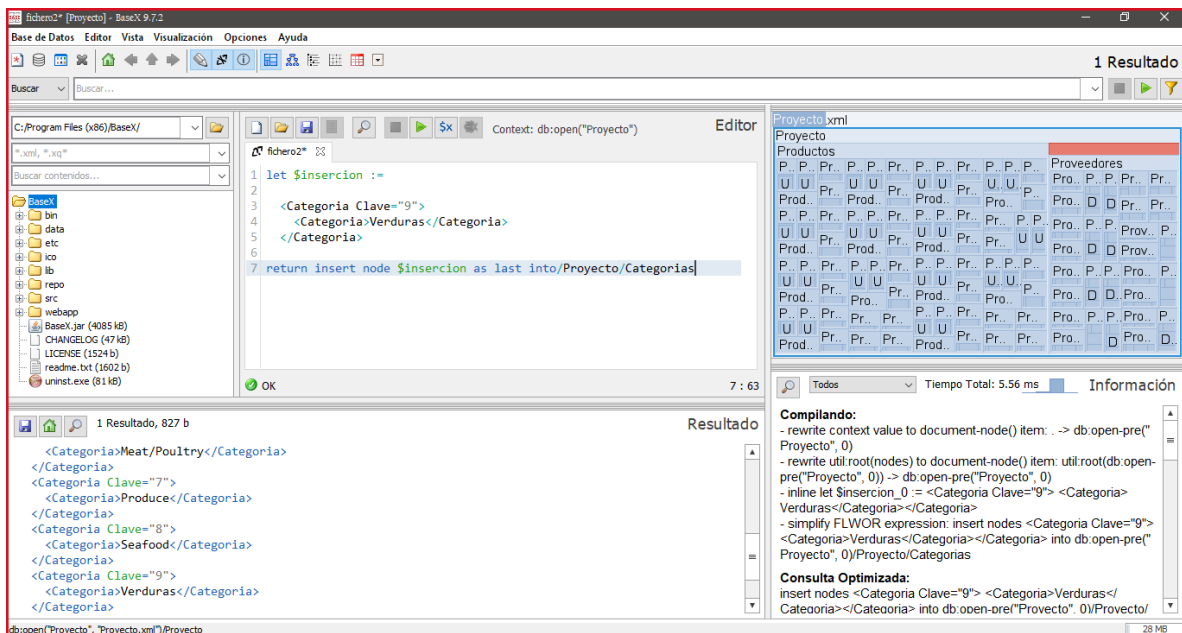
let \$insercion :=

<Categoria Clave="9">

<Categoria>Verduras</Categoria>

</Categoria>

return insert node \$insercion as last into/Proyecto/Categorias



## Suppliers:

let \$insercion :=

<Proveedor clave="30">

<Compañía>Coppel</Compañía>

<Contacto>

<NombreCompañía>Enrique Coppel</NombreCompañía>

<TituloCompañía>Dueño</TituloCompañía>

</Contacto>

<Domicilio>

<Calle>Obregón</Calle>

<Ciudad>Culiacán</Ciudad>

<CodigoPostal>80200</CodigoPostal>

<Pais>México</Pais>

</Domicilio>

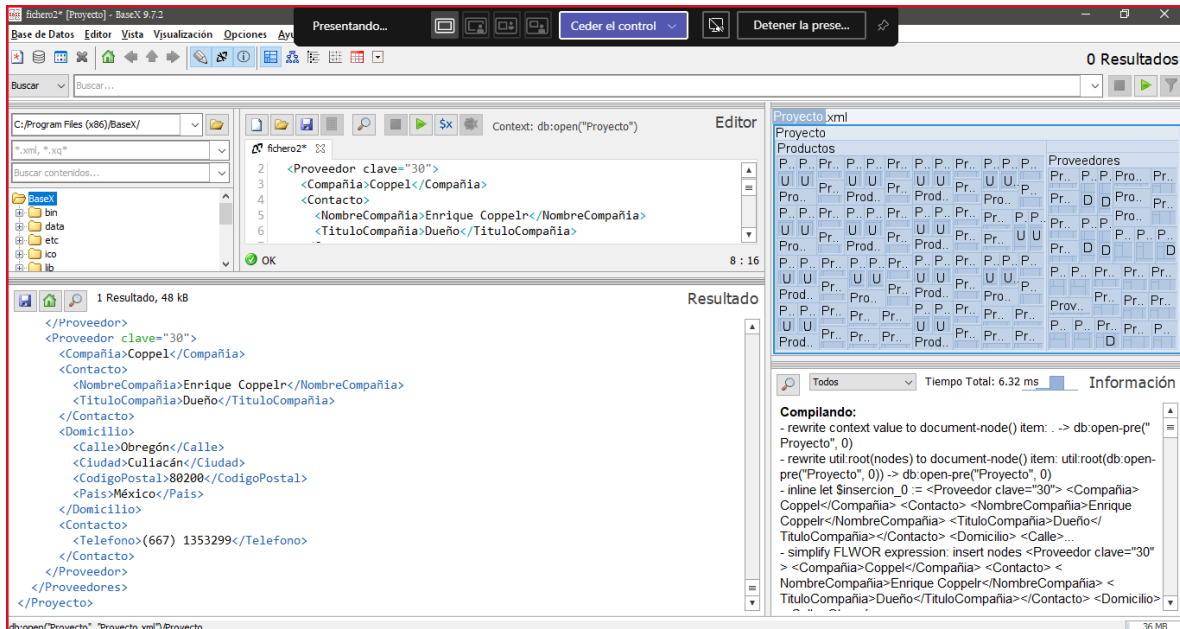
<Contacto>

<Telefono>(667) 1353299</Telefono>

</Contacto>

</Proveedor>

return insert node \$insercion as last into/Proyecto/Proveedores



## Products:

let \$insercion :=

<Producto clave="80" Proveedor="30" Categoria="9">

<Nombre>Margarina</Nombre>

<Unidad>

<CantidadPorUnidad>15 boxes x 12 piezas</CantidadPorUnidad>

<PrecioUnitario>24.0000</PrecioUnitario>

<UnitInStock>80</UnitInStock>

<UnidadesOrden>2</UnidadesOrden>

<Reorden>14</Reorden>

</Unidad>

<Descuento>5</Descuento>

</Producto>

return insert node \$insercion as last into/Proyecto/Productos

The screenshot shows the BaseX 9.7.3 interface with the following components:

- Editor:** Displays the XSLT script. The script includes a context declaration `Context: db:open("Proyecto")` and an XSLT transformation that inserts a new product node into the `/Proyecto/Productos` path. The script is as follows:

```
<?xml version="1.0"?>
<Proyecto>
  <Productos>
    <Producto>
      <Nombre>Margarina</Nombre>
      <Unidad>
        <CantidadPorUnidad>15 boxes x 12 piezas</CantidadPorUnidad>
        <PrecioUnitario>24.0000</PrecioUnitario>
        <UnitInStock>80</UnitInStock>
        <UnidadesOrden>2</UnidadesOrden>
        <Reorden>14</Reorden>
      </Unidad>
      <Descuento>5</Descuento>
    </Producto>
  </Productos>
</Proyecto>
```

The script is followed by the instruction: `return insert node $insercion as last into/Proyecto/Productos`.
- Result:** Shows the transformed XML document. The result is a single XML document with the following structure:

```
<?xml version="1.0"?>
<Proyecto>
  <Productos>
    <Producto>
      <Nombre>Margarina</Nombre>
      <Unidad>
        <CantidadPorUnidad>15 boxes x 12 piezas</CantidadPorUnidad>
        <PrecioUnitario>24.0000</PrecioUnitario>
        <UnitInStock>80</UnitInStock>
        <UnidadesOrden>2</UnidadesOrden>
        <Reorden>14</Reorden>
      </Unidad>
      <Descuento>5</Descuento>
    </Producto>
  </Productos>
</Proyecto>
```
- Compilando:** Shows the compilation log. The log includes the following messages:
  - rewrite context value to document-node() item: -> db:open-pre("Proyecto", 0)
  - rewrite util.root(nodes) to document-node() item: util.root(db:open-pre("Proyecto", 0)) -> db:open-pre("Proyecto", 0)
  - inline let \$insercion\_0 := <Producto clave="80" Proveedor="30" Categoria="9"> <Nombre>Margarina</Nombre> <Unidad> <CantidadPorUnidad>15 boxes x 12 piezas</CantidadPorUnidad> <PrecioUnitario>24.0000</PrecioUnitario> <UnitInStock>80</UnitInStock> <UnidadesOrden>2</UnidadesOrden> <Reorden>14</Reorden> </Unidad> <Descuento>5</Descuento> </Producto>