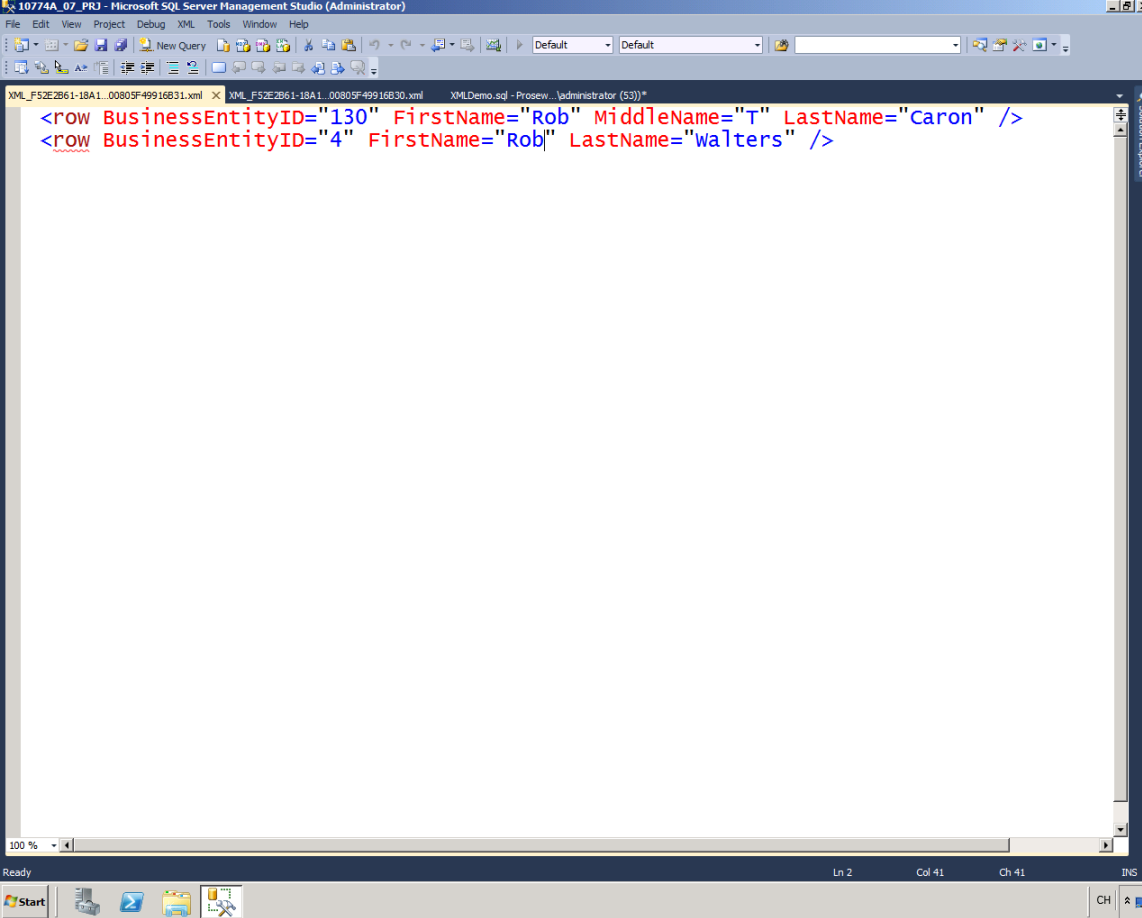


EXAMPLE 1


```

<row BusinessEntityID="130" FirstName="Rob" MiddleName="T" LastName="Caron" />
<row BusinessEntityID="4" FirstName="Rob" LastName="Walters" />

```

--RAW MODE Generates A Single Element For Each Row In The Result Set

```
SELECT e.BusinessEntityID, c.FirstName,
c.MiddleName, c.LastName
```

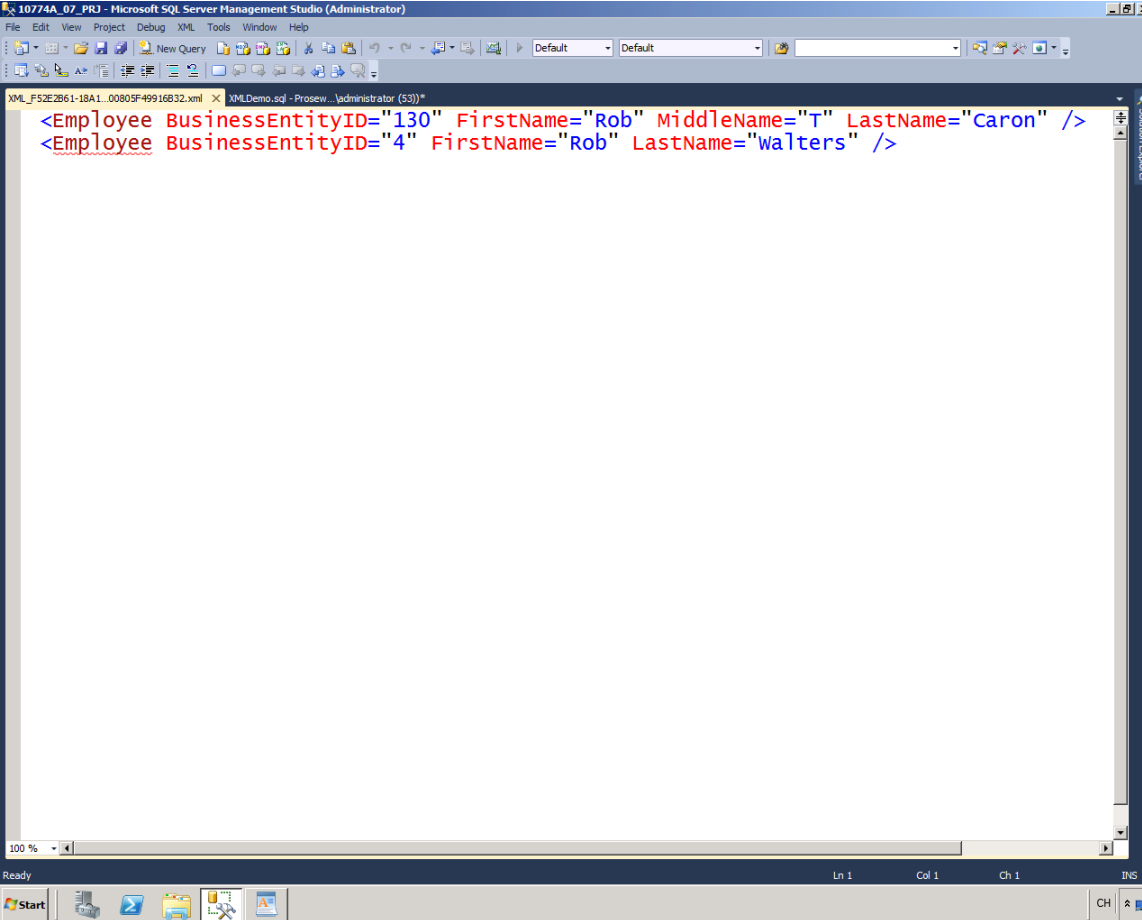
```
FROM HumanResources.Employee e INNER JOIN
Person.Person c
```

```
ON c.BusinessEntityID = e.BusinessEntityID
```

```
WHERE c.FirstName = 'Rob'
```

```
FOR XML RAW;
```

EXAMPLE 2



The screenshot shows the Microsoft SQL Server Management Studio interface. The main window displays the following XML output:

```
<Employee BusinessEntityID="130" FirstName="Rob" MiddleName="T" LastName="Caron" />
<Employee BusinessEntityID="4" FirstName="Rob" LastName="Walters" />
```

--Override Default Behaviour Providing A Name For The Element

```
SELECT e.BusinessEntityID, c.FirstName,
c.MiddleName, c.LastName
```

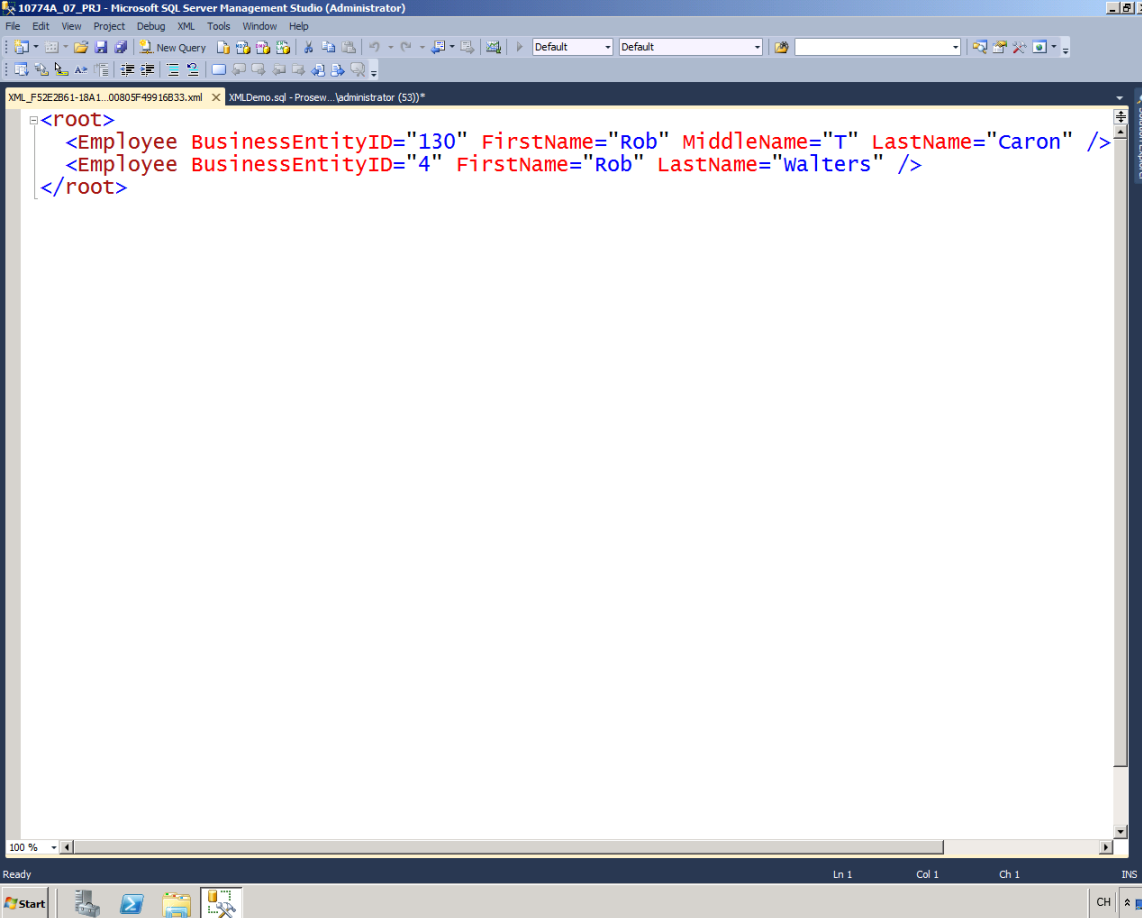
```
FROM HumanResources.Employee e INNER JOIN
Person.Person c
```

```
ON c.BusinessEntityID = e.BusinessEntityID
```

```
WHERE c.FirstName = 'Rob'
```

```
FOR XML RAW ('Employee');
```

EXAMPLE 3



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The main window displays the following XML output:

```
<root>  
  <Employee BusinessEntityID="130" FirstName="Rob" MiddleName="T" LastName="Caron" />  
  <Employee BusinessEntityID="4" FirstName="Rob" LastName="Walters" />  
</root>
```

--Also Specify A Root Element Be Created To Wrap All Other Elements Using the ROOT Keyword

```
SELECT e.BusinessEntityID, c.FirstName,  
c.MiddleName, c.LastName
```

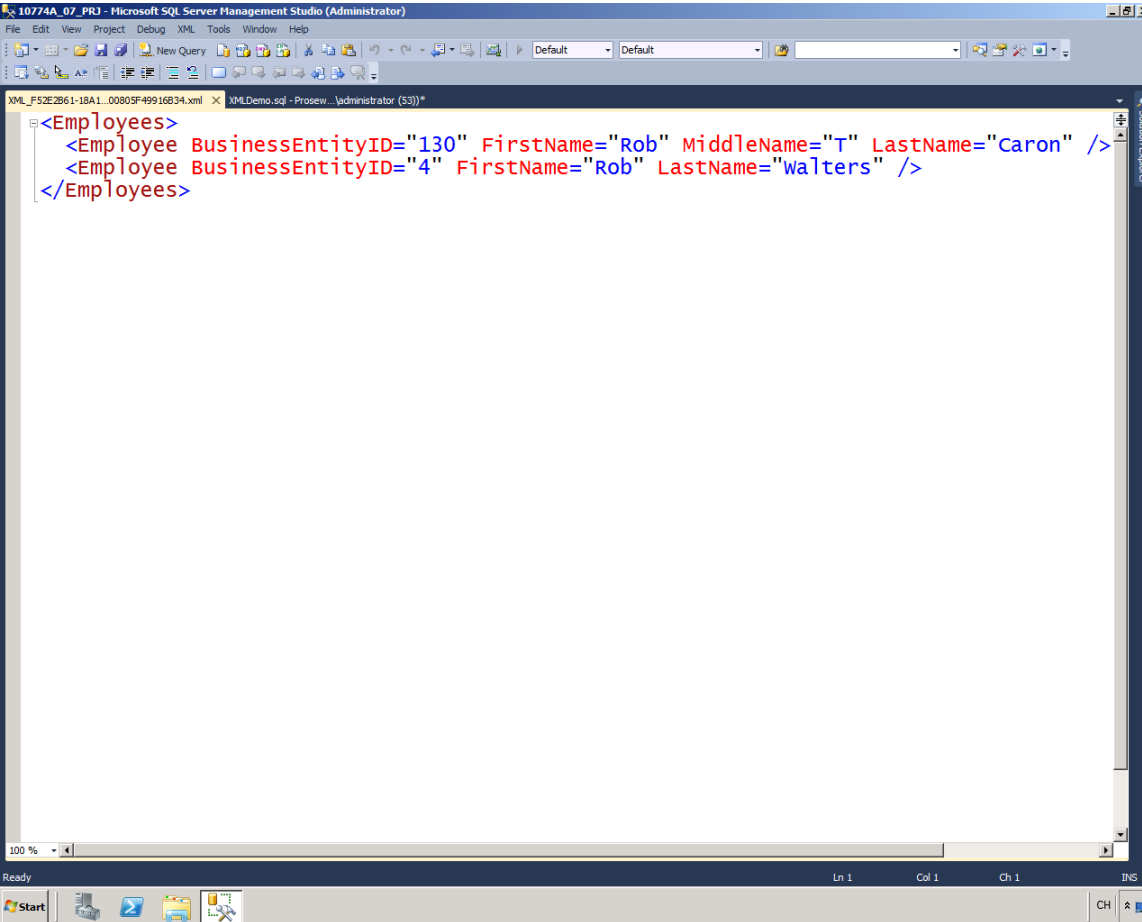
```
FROM HumanResources.Employee e INNER JOIN  
Person.Person c
```

```
ON c.BusinessEntityID = e.BusinessEntityID
```

```
WHERE c.FirstName = 'Rob'
```

```
FOR XML RAW ('Employee'), ROOT; --add a comma  
separate the elements
```

EXAMPLE 4



The screenshot shows the Microsoft SQL Server Management Studio interface. The main window displays the following XML output:

```
<Employees>  
  <Employee BusinessEntityID="130" FirstName="Rob" MiddleName="" LastName="Caron" />  
  <Employee BusinessEntityID="4" FirstName="Rob" LastName="Walters" />  
</Employees>
```

--Also Provide a Specific Name For The ROOT Element

```
SELECT e.BusinessEntityID, c.FirstName,  
c.MiddleName, c.LastName
```

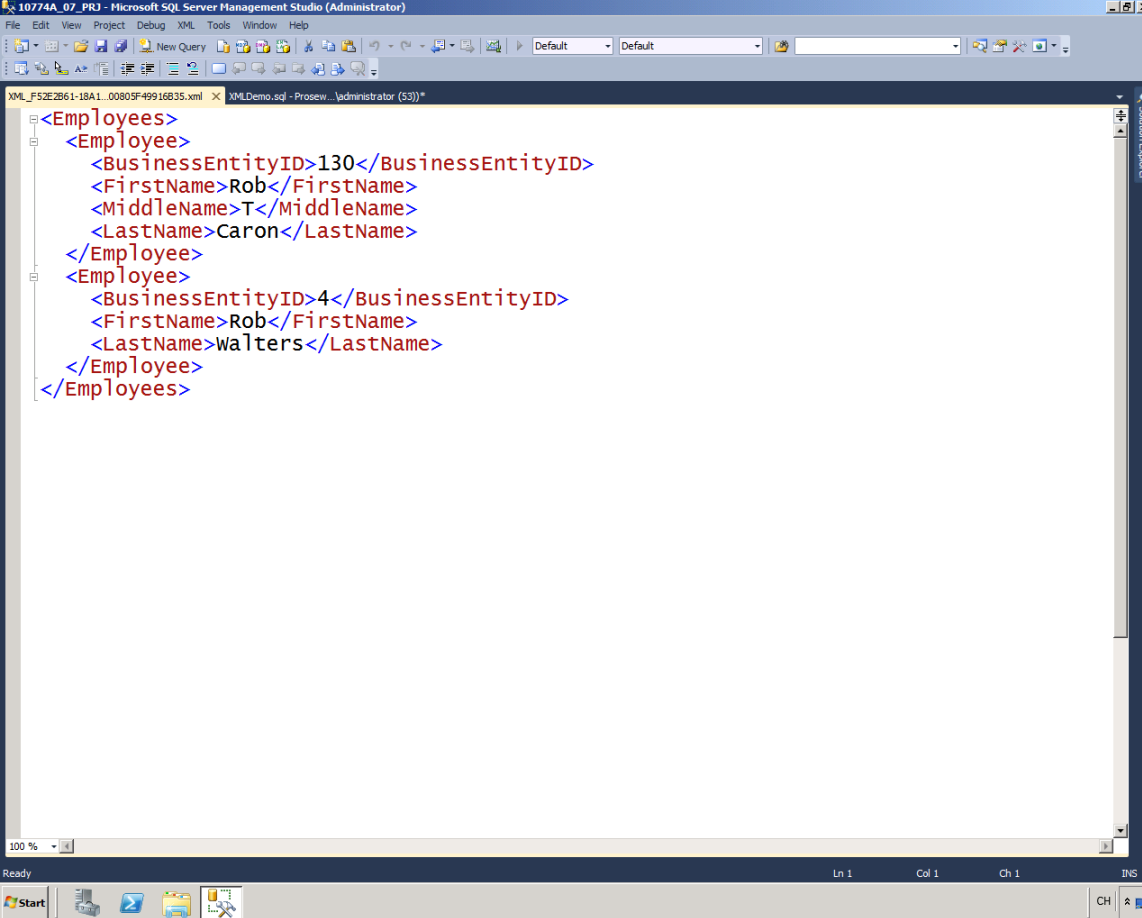
```
FROM HumanResources.Employee e INNER JOIN  
Person.Person c
```

```
ON c.BusinessEntityID = e.BusinessEntityID
```

```
WHERE c.FirstName = 'Rob'
```

```
FOR XML RAW ('Employee'), ROOT ('Employees');
```

EXAMPLE 5

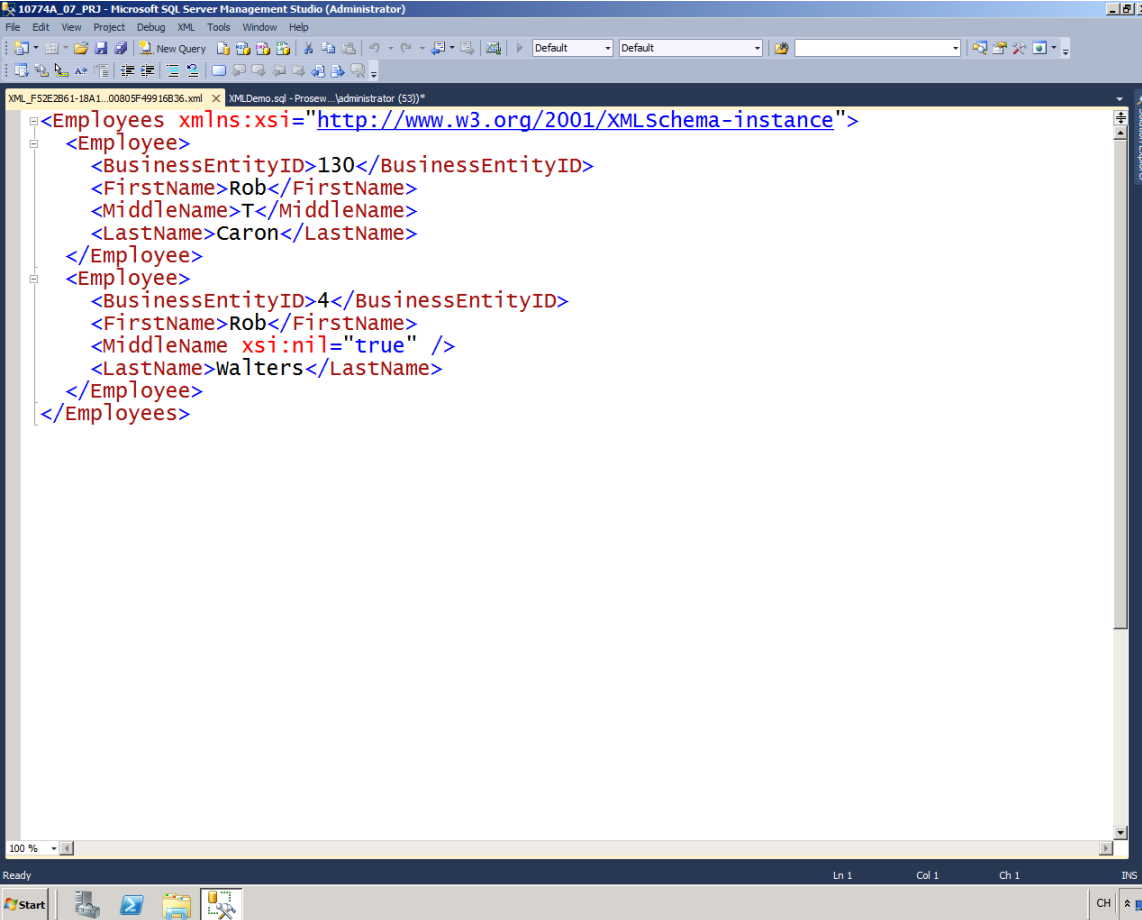


```
Microsoft SQL Server Management Studio (Administrator)
File Edit View Project Debug XML Tools Window Help
XMLDemo.sql - Prosew..._administrator (53)*
<Employees>
  <Employee>
    <BusinessEntityID>130</BusinessEntityID>
    <FirstName>Rob</FirstName>
    <MiddleName>T</MiddleName>
    <LastName>Caron</LastName>
  </Employee>
  <Employee>
    <BusinessEntityID>4</BusinessEntityID>
    <FirstName>Rob</FirstName>
    <LastName>Walters</LastName>
  </Employee>
</Employees>
```

--Specify That The Column values Be Added AS Child Elements To The Row Elements Using The ELEMENTS Option

```
SELECT e.BusinessEntityID, c.FirstName,
c.MiddleName, c.LastName
FROM HumanResources.Employee e INNER JOIN
Person.Person c
ON c.BusinessEntityID = e.BusinessEntityID
WHERE c.FirstName = 'Rob'
FOR XML RAW ('Employee'), ROOT ('Employees'),
ELEMENTS;
```

EXAMPLE 6

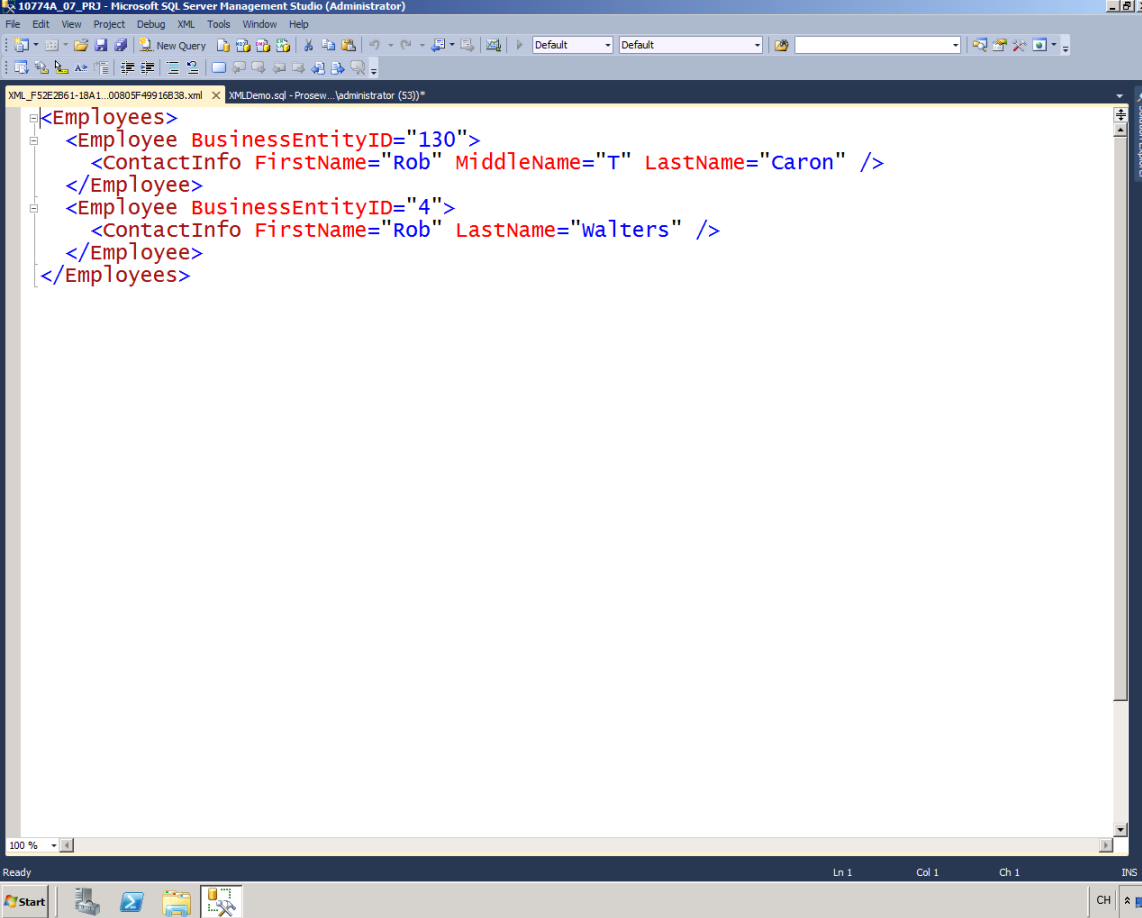


```
<Employees xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Employee>
    <BusinessEntityID>130</BusinessEntityID>
    <FirstName>Rob</FirstName>
    <MiddleName>T</MiddleName>
    <LastName>Caron</LastName>
  </Employee>
  <Employee>
    <BusinessEntityID>4</BusinessEntityID>
    <FirstName>Rob</FirstName>
    <MiddleName xsi:nil="true" />
    <LastName>Walters</LastName>
  </Employee>
</Employees>
```

--No Elements Are Created For Columns with Null Value Can Override Using XSINIL Keyword To Elements Option

```
SELECT e.BusinessEntityID, c.FirstName,
c.MiddleName, c.LastName
FROM HumanResources.Employee e INNER JOIN
Person.Person c
ON c.BusinessEntityID = e.BusinessEntityID
WHERE c.FirstName = 'Rob'
FOR XML RAW ('Employee'), ROOT ('Employees'),
ELEMENTS XSINIL;
```

EXAMPLE 7

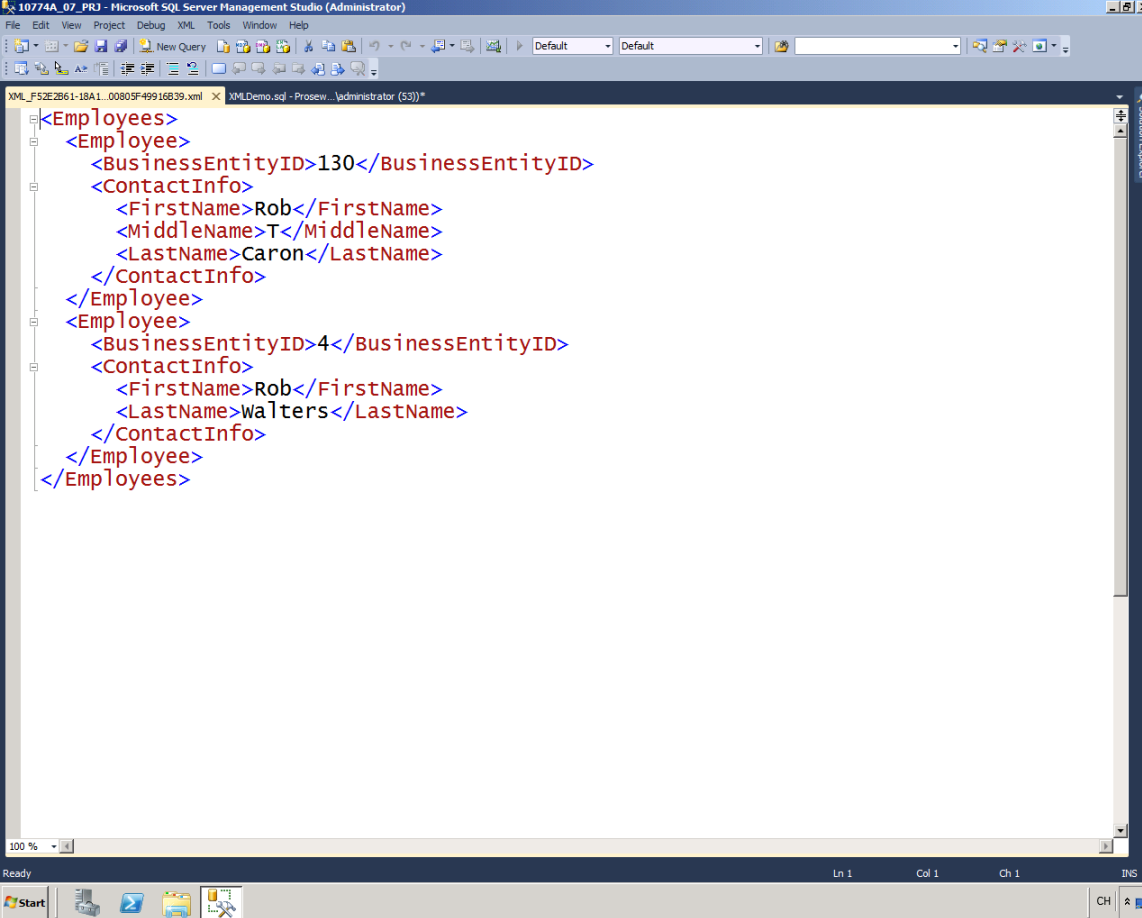


The screenshot shows the Microsoft SQL Server Management Studio interface. The main window displays the following XML output:

```
<Employees>  
  <Employee BusinessEntityID="130">  
    <ContactInfo FirstName="Rob" MiddleName="T" LastName="Caron" />  
  </Employee>  
  <Employee BusinessEntityID="4">  
    <ContactInfo FirstName="Rob" LastName="Walters" />  
  </Employee>  
</Employees>
```

--AUTO MODE Generates The XML Based On How select Is Defined

```
SELECT Employee.BusinessEntityID,  
ContactInfo.FirstName,  
        ContactInfo.MiddleName, ContactInfo.LastName  
FROM HumanResources.Employee AS Employee  
INNER JOIN Person.Person AS ContactInfo  
ON ContactInfo.BusinessEntityID =  
Employee.BusinessEntityID  
WHERE ContactInfo.FirstName = 'Rob'  
FOR XML AUTO, ROOT ('Employees');
```

EXAMPLE 8


```

<Employees>
  <Employee>
    <BusinessEntityID>130</BusinessEntityID>
    <ContactInfo>
      <FirstName>Rob</FirstName>
      <MiddleName>T</MiddleName>
      <LastName>Caron</LastName>
    </ContactInfo>
  </Employee>
  <Employee>
    <BusinessEntityID>4</BusinessEntityID>
    <ContactInfo>
      <FirstName>Rob</FirstName>
      <LastName>Walters</LastName>
    </ContactInfo>
  </Employee>
</Employees>

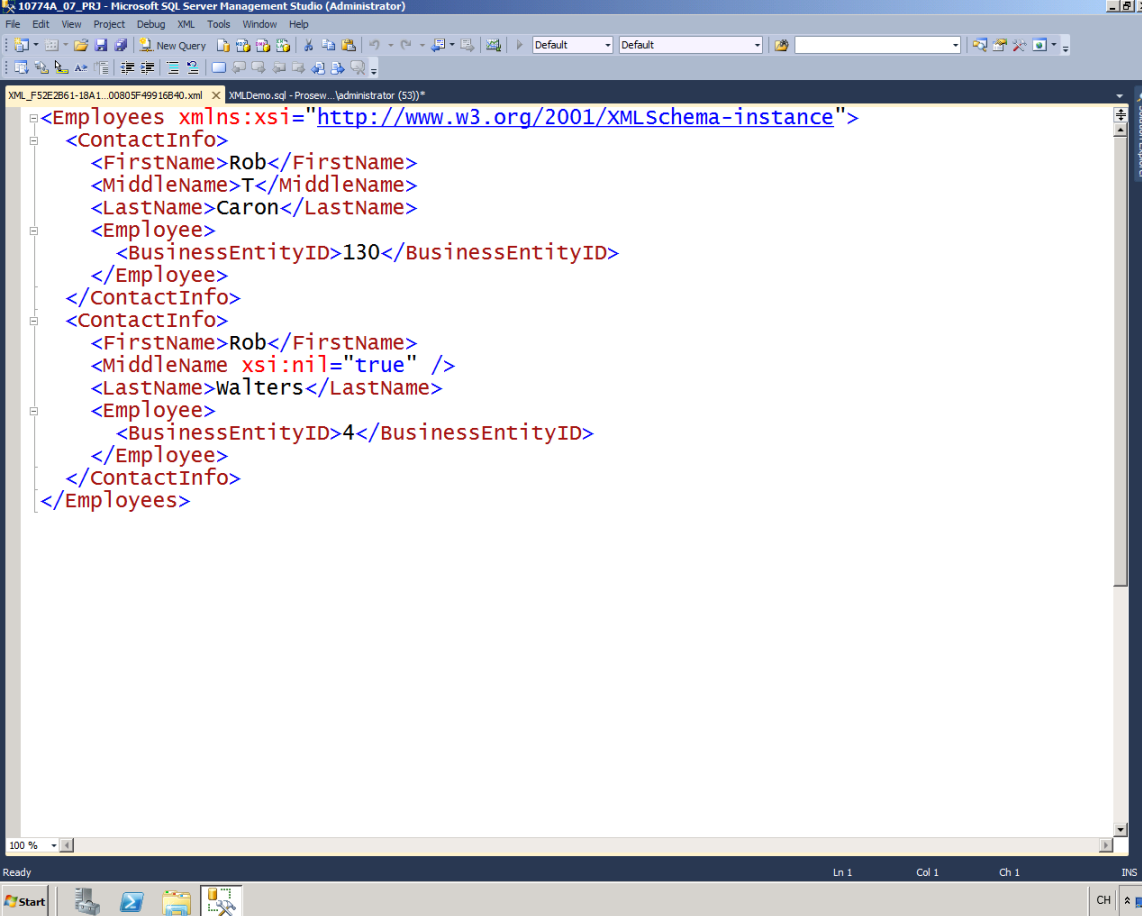
```

-Using ELEMENTS Options Show As Child elements Rather Than Attributes

```

SELECT Employee.BusinessEntityID,
ContactInfo.FirstName,
      ContactInfo.MiddleName, ContactInfo.LastName
FROM HumanResources.Employee AS Employee
INNER JOIN Person.Person AS ContactInfo
ON ContactInfo.BusinessEntityID =
Employee.BusinessEntityID
WHERE ContactInfo.FirstName = 'Rob'
FOR XML AUTO, ROOT ('Employees'), ELEMENTS;

```


EXAMPLE 9


```

<Employees xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ContactInfo>
    <FirstName>Rob</FirstName>
    <MiddleName>T</MiddleName>
    <LastName>Caron</LastName>
    <Employee>
      <BusinessEntityID>130</BusinessEntityID>
    </Employee>
  </ContactInfo>
  <ContactInfo>
    <FirstName>Rob</FirstName>
    <MiddleName xsi:nil="true" />
    <LastName>Walters</LastName>
    <Employee>
      <BusinessEntityID>4</BusinessEntityID>
    </Employee>
  </ContactInfo>
</Employees>

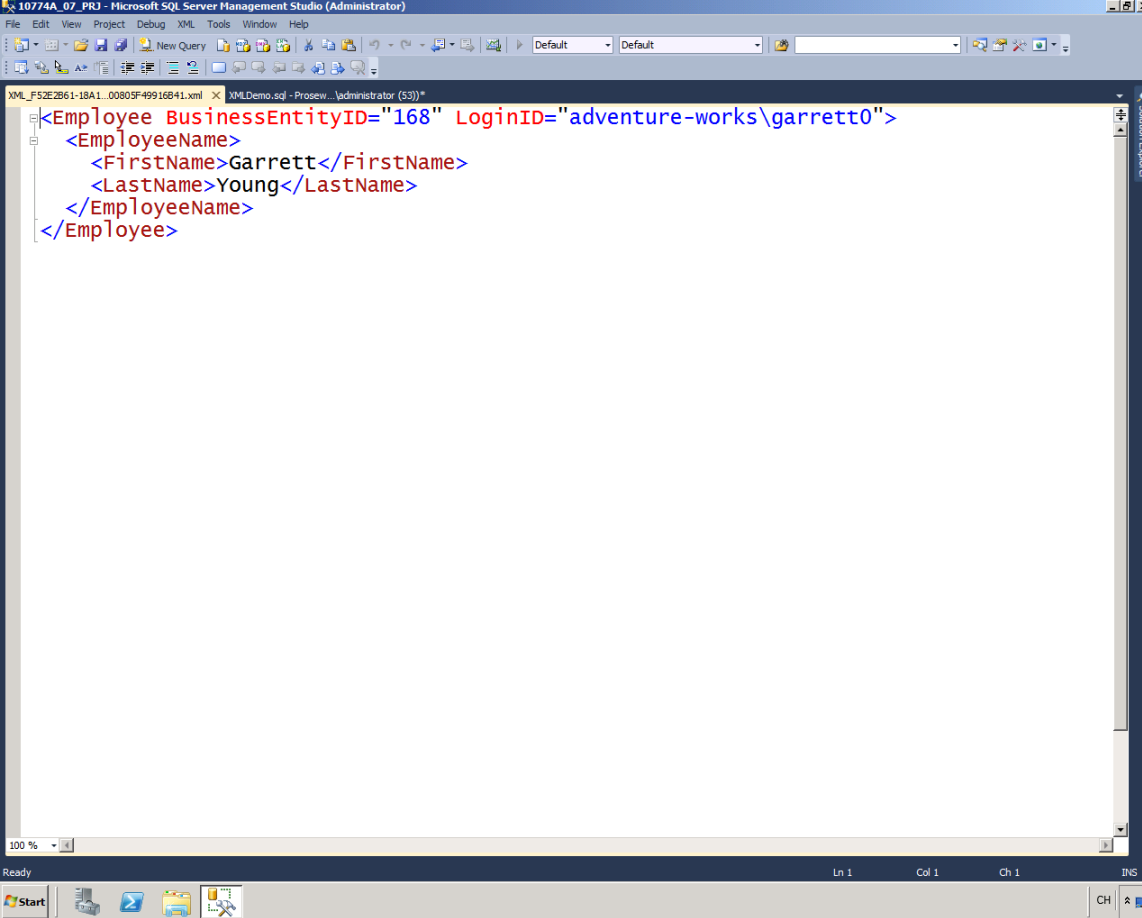
```

-Using XSINIL Option With AUTO Mode

```

SELECT ContactInfo.FirstName,
ContactInfo.MiddleName,
      ContactInfo.LastName,
Employee.BusinessEntityID
FROM HumanResources.Employee AS Employee
      INNER JOIN Person.Person AS ContactInfo
      ON ContactInfo.BusinessEntityID =
Employee.BusinessEntityID
WHERE ContactInfo.FirstName = 'Rob'
FOR XML AUTO, ROOT ('Employees'), ELEMENTS XSINIL;

```

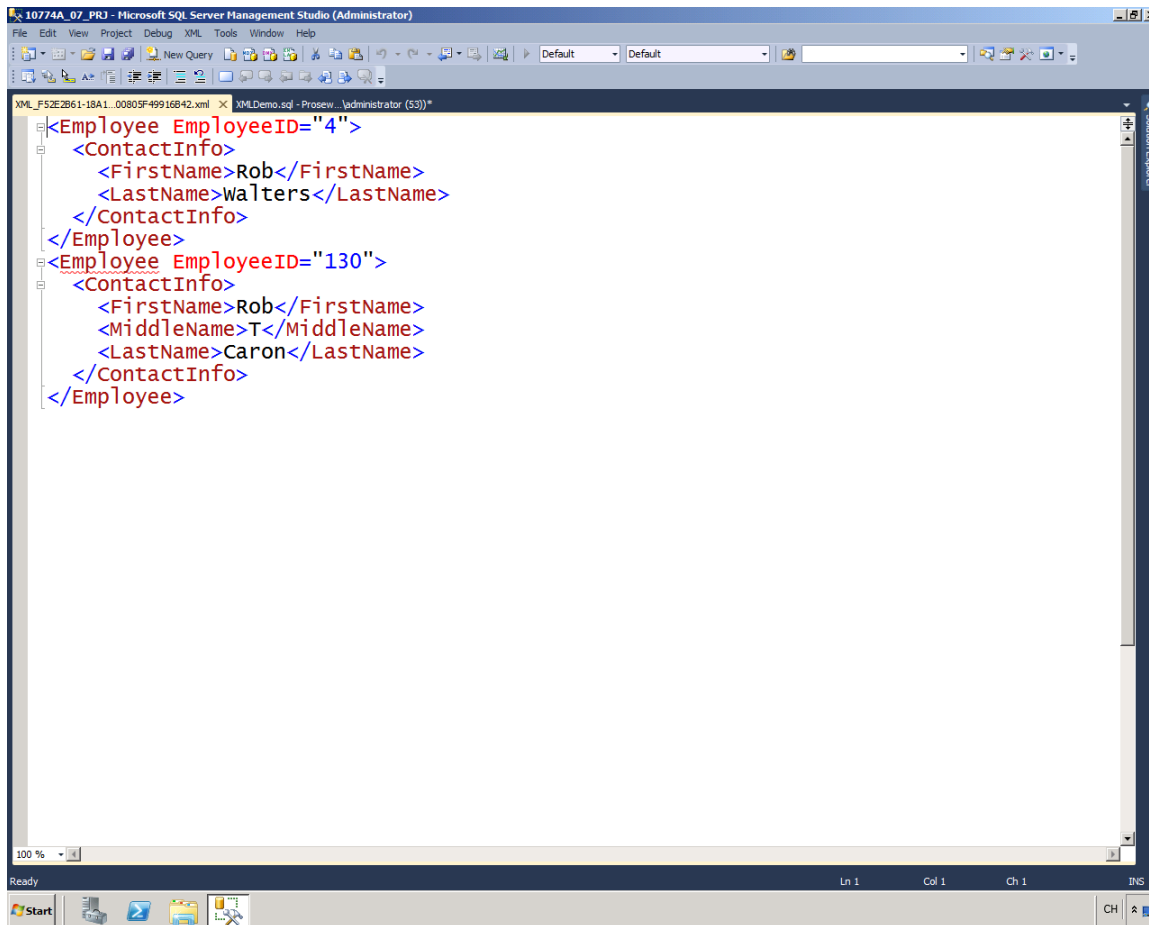
EXAMPLE 10


The screenshot shows the Microsoft SQL Server Management Studio interface. The main window displays the following XML result:

```
<Employee BusinessEntityID="168" LoginID="adventure-works\garrett0">
  <EmployeeName>
    <FirstName>Garrett</FirstName>
    <LastName>Young</LastName>
  </EmployeeName>
</Employee>
```

--Return Some Data As Attributes And Some As Child Elements. Use SubQuery Method.--TYPE Option Preserve The Data As XML Returned By SubQuery To Outer Select

```
SELECT BusinessEntityID, LoginID,
       (SELECT FirstName, LastName
        FROM Person.Person AS EmployeeName
        WHERE EmployeeName.BusinessEntityID =
Employee.BusinessEntityID
        FOR XML AUTO, TYPE, ELEMENTS)
FROM HumanResources.Employee AS Employee
WHERE BusinessEntityID = 168
FOR XML AUTO;
```

EXAMPLE 11

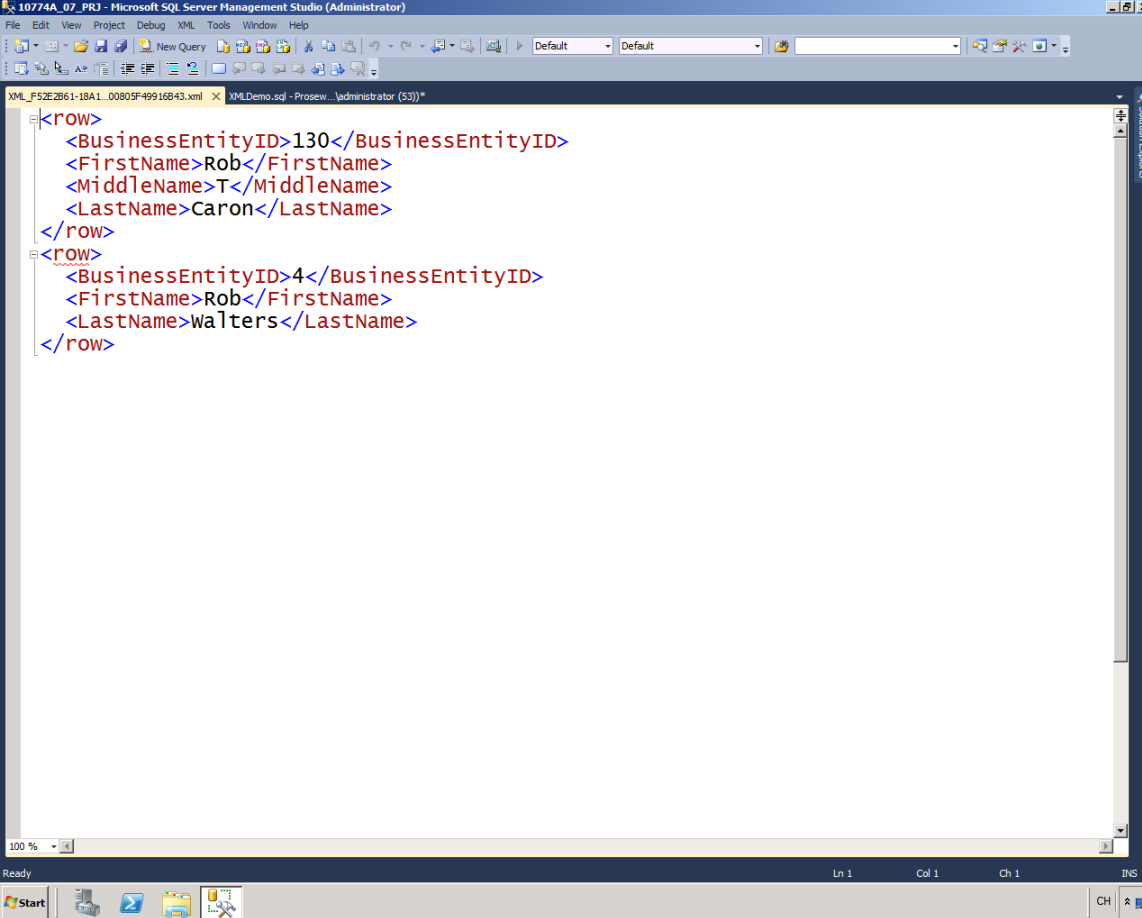
The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The main window displays XML data for two employees. The first employee has EmployeeID "4" and ContactInfo with FirstName "Rob" and LastName "Walters". The second employee has EmployeeID "130" and ContactInfo with FirstName "Rob", MiddleName "T", and LastName "Caron". The XML is color-coded and has a tree view on the left. The status bar at the bottom shows "Ready", "Ln 1", "Col 1", "Ch 1", and "INS".

```
<Employee EmployeeID="4">
  <ContactInfo>
    <FirstName>Rob</FirstName>
    <LastName>Walters</LastName>
  </ContactInfo>
</Employee>
<Employee EmployeeID="130">
  <ContactInfo>
    <FirstName>Rob</FirstName>
    <MiddleName>T</MiddleName>
    <LastName>Caron</LastName>
  </ContactInfo>
</Employee>
```

--EXPLICIT Mode Build Your Select In A Way As To Define The XML Hierarchy And Structure

```
SELECT 1 AS Tag, --numerical value each level in the hierarchy
        NULL AS Parent, --numerical value identifies the parent NULL
        top of hierarchy
        e.BusinessEntityID AS [Employee!1!EmployeeID],
        NULL AS [ContactInfo!2!FirstName!ELEMENT],
        NULL AS [ContactInfo!2!MiddleName!ELEMENT], --column Alias TAG
        number the value should be assigned to
        NULL AS [ContactInfo!2!LastName!ELEMENT]--optional directive
        how to construct the XML
FROM HumanResources.Employee e INNER JOIN Person.Person c
        ON c.BusinessEntityID = e.BusinessEntityID
WHERE c.FirstName = 'Rob'
UNION ALL
SELECT 2 AS Tag,
        1 AS Parent,
        e.BusinessEntityID,
        c.FirstName,
        c.MiddleName,
        c.LastName
FROM HumanResources.Employee e INNER JOIN Person.Person c
        ON e.BusinessEntityID = c.BusinessEntityID
WHERE c.FirstName = 'Rob'
ORDER BY [Employee!1!EmployeeID],
[ContactInfo!2!FirstName!ELEMENT]
FOR XML EXPLICIT;
```

EXAMPLE 12



The screenshot shows the XML output of a query in SQL Server Management Studio. The XML is as follows:

```
<row>
  <BusinessEntityID>130</BusinessEntityID>
  <FirstName>Rob</FirstName>
  <MiddleName>T</MiddleName>
  <LastName>Caron</LastName>
</row>
<row>
  <BusinessEntityID>4</BusinessEntityID>
  <FirstName>Rob</FirstName>
  <LastName>Walters</LastName>
</row>
```

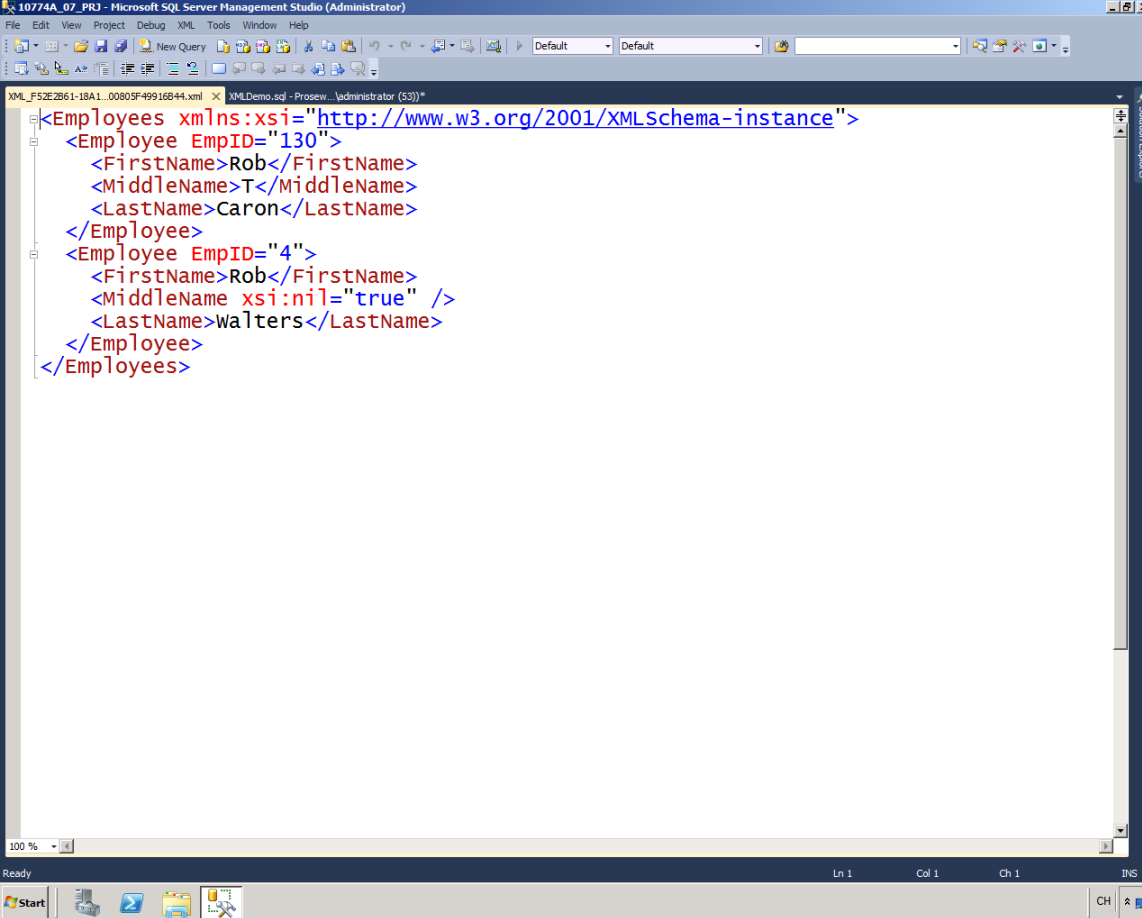
---PATH Mode Column Names Treated As XPath Expressions That Determine How Data Values Mapped To XML Result Set

--Each Column Added As A Child Element

```
SELECT e.BusinessEntityID, c.FirstName,
       c.MiddleName, c.LastName
FROM HumanResources.Employee AS e
     INNER JOIN Person.Person AS c
     ON c.BusinessEntityID = e.BusinessEntityID
WHERE c.FirstName = 'Rob'
```

FOR XML PATH;

EXAMPLE 13

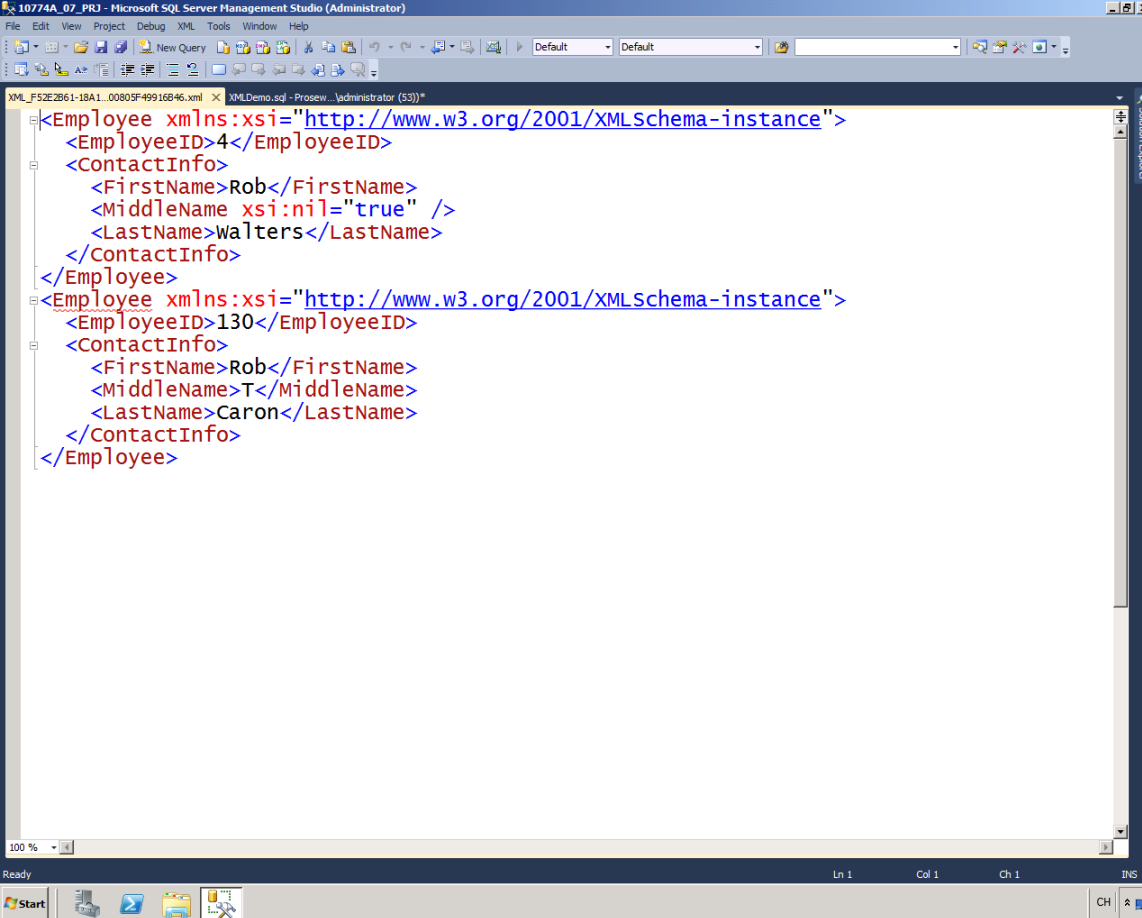


```
<Employees xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Employee EmpID="130">
    <FirstName>Rob</FirstName>
    <MiddleName>T</MiddleName>
    <LastName>Caron</LastName>
  </Employee>
  <Employee EmpID="4">
    <FirstName>Rob</FirstName>
    <MiddleName xsi:nil="true" />
    <LastName>Walters</LastName>
  </Employee>
</Employees>
```

--Return Null values

```
SELECT e.BusinessEntityID AS "@EmpID",
       c.FirstName, c.MiddleName, c.LastName
FROM HumanResources.Employee AS e
     INNER JOIN Person.Person AS c
     ON c.BusinessEntityID = e.BusinessEntityID
WHERE c.FirstName = 'Rob'
FOR XML PATH ('Employee'), ROOT ('Employees'),
ELEMENTS XSINIL;
```

EXAMPLE 14



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The main window displays XML data in a grid view. The XML is structured as follows:

```
<Employee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  
  <EmployeeID>4</EmployeeID>  
  <ContactInfo>  
    <FirstName>Rob</FirstName>  
    <MiddleName xsi:nil="true" />  
    <LastName>Walters</LastName>  
  </ContactInfo>  
</Employee>  
<Employee xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">  
  <EmployeeID>130</EmployeeID>  
  <ContactInfo>  
    <FirstName>Rob</FirstName>  
    <MiddleName>T</MiddleName>  
    <LastName>Caron</LastName>  
  </ContactInfo>  
</Employee>
```

The grid view shows two rows of data. The first row corresponds to EmployeeID 4, with FirstName 'Rob', MiddleName nil, and LastName 'Walters'. The second row corresponds to EmployeeID 130, with FirstName 'Rob', MiddleName 'T', and LastName 'Caron'. The status bar at the bottom indicates 'Ready' and 'Ln 1 Col 1 Ch 1 INS'.

--Ensure Columns with Null values Display XSINIL

```
SELECT 1 AS Tag,
        NULL AS Parent,
        e.BusinessEntityID AS [Employee!1!EmployeeID!ELEMENT],
        NULL AS [ContactInfo!2!FirstName!ELEMENT],
        NULL AS [ContactInfo!2!MiddleName!ELEMENTXSINIL], --Null
value
        NULL AS [ContactInfo!2!LastName!ELEMENT]
FROM HumanResources.Employee e INNER JOIN Person.Person c
     ON c.BusinessEntityID = e.BusinessEntityID
WHERE c.FirstName = 'Rob'
UNION ALL
SELECT 2 AS Tag,
        1 AS Parent,
        e.BusinessEntityID,
        c.FirstName,
        c.MiddleName,
        c.LastName
FROM HumanResources.Employee e INNER JOIN Person.Person c
     ON e.BusinessEntityID = c.BusinessEntityID
WHERE c.FirstName = 'Rob'
ORDER BY [Employee!1!EmployeeID!ELEMENT],
         [ContactInfo!2!FirstName!ELEMENT]
FOR XML EXPLICIT;
```