Elektrotehnički fakultet
Univerzitet u Beogradu
Katedra za računarsku tehniku i informatiku

Baze podataka 2

XML i SQL Server

2022/2023

XML i SQL Server

- Microsoft SQL Server ima podršku za XML.
- U okviru Microsoft SQL Server-a, u poljima se može čuvati XML.
- Pored toga što standardni SELECT upiti vraćaju tabelu, može se podesiti da SELECT upiti vraćaju XML.

Korišćena baza podataka

- Primeri će biti radjeni na bazi AdvantureWorks2019.
- Link:

  https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms

Generisanje XML sadržaja

- Za generisanje XML sadržaja iz relacione baze podataka, koristi se FOR XML naredba kod SELECT upita.
- Koriste se četiri režima za kreiranje XML sadržaja:
  - RAW
  - AUTO
  - PATH
  - EXPLICIT

FOR XML RAW

- Koristan za upite kada struktura rezultata XML sadržaja nije unapred poznata.
- Rezultat FOR XML RAW se može drastično promeniti ako se promeni struktura izvorne tabele.

FOR XML AUTO

- Koristan za upite kada je potrebno da rezultujući XML sadrži imena tabela.
- Ovo je posebno korisno kada treba mapirati XML rezultat nazad u originalne kolone u okvirnu izvornih tabela.
- Kao i kod FOR XML RAW, rezultat FOR XML AUTO se može promeniti ako se promeni struktura izvorne tabele.

FOR XML PATH

- Dizajniran j aza eksplicitno definisanje strukture XML rezultata.
- Ovo je sigurnija opcija od RAW ili AUTO režima u proizvodnom kodu, jer uvek unapred znate strukturu XML rezultata, čak i ako je tabela promenila strukturu.

FOR XML EXPLICIT

- Originalna metoda za eksplicitno definisanje strukture XML rezultata.
- Operacija FOR XML EXPLICIT je složenija i manje intuitivna od FOR XML PATH.
- Trebalo bi da koristite režim PATH umesto EXPLICIT režima kada želite da definišete eksplicitnu strukturu za XML rezultat.

FOR XML opcije

	XML SCHEMA	ELEMENTS XSINIL	ELEMENTS ABSENT	BINARY BASE64	TYPE	ROOT	("ElementName")
FOR XML AUTO	✓	✓	✓	✓	✓	✓	
FOR XML RAW	✓	✓	✓	✓	✓	✓	✓
FOR XML PATH		✓	✓	✓	✓	✓	✓
FOR XML EXPLICIT				✓	✓	✓	

FOR XML opcije

- XML SCHEMA – definisanje šablona za XML fajl
- ELEMENTS XSINIL – NULL vrednost u SQL-u se u XML-a generiše kao element sa atributom xsi:nil = “true”
- ELEMENTS ABSENT – NULL vrenost u SQL-u se ne prikazuju u XML-u
- BINARY BASE64 – za binarne podatke koristi BASE64 format

FOR XML opcije

- TYPE - vraća FOR XML rezultat kao instancu tipa XML podataka. Ovo je posebno korisno kod ugnježdenih FOR XML upita, dodeliivanju rezultata XML-u promenljivoj i sačuvanju u xml koloni.
- ROOT – dodaje koreni element u rezultujući XML sa zadatim nazivom.
- (“ElementName”) – preimenovanje elemenata

FOR XML PATH – Primer 1

- Primer:

```
SELECT emp.NationalIDNumber AS "Employee/@ID",
 emp.HireDate AS "Employee/Hire-Date",
 per.LastName AS "Employee/Name/Last",
 per.FirstName AS "Employee/Name/First",
 per.MiddleName AS "Employee/Name/Middle"
FROM HumanResources.Employee emp
 INNER JOIN Person.Person per
 ON emp.BusinessEntityID = per.BusinessEntityID
WHERE emp.BusinessEntityID IN ( 2, 3 )
FOR XML PATH;
```

FOR XML PATH – Primer 1

- Rezultat:

```
<row>
  <Employee ID="245797967">
    <Hire-Date>2008-01-31</Hire-Date>
    <Name>
      <Last>Duffy</Last>
      <First>Terri</First>
      <Middle>Lee</Middle>
    </Name>
  </Employee>
</row>
<row>
  <Employee ID="509647174">
    <Hire-Date>2007-11-11</Hire-Date>
    <Name>
      <Last>Tamburello</Last>
      <First>Roberto</First>
    </Name>
  </Employee>
</row>
```

FOR XML PATH – Primer 2

- Primer:

```
SELECT emp.NationalIDNumber AS "Employee/@ID",
 emp.HireDate AS "Employee/Hire-Date",
 per.LastName AS "Employee/Name/Last",
 per.FirstName AS "Employee/Name/First",
 per.MiddleName AS "Employee/Name/Middle"
FROM HumanResources.Employee emp
 INNER JOIN Person.Person per
 ON emp.BusinessEntityID = per.BusinessEntityID
WHERE emp.BusinessEntityID IN ( 2, 3 )
FOR XML PATH,
ELEMENTS XSINIL;
```

FOR XML PATH – Primer 2

- Rezultat:

```
<row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Employee ID="245797967">
    <Hire-Date>2008-01-31</Hire-Date>
    <Name>
      <Last>Duffy</Last>
      <First>Terri</First>
      <Middle>Lee</Middle>
    </Name>
  </Employee>
</row>
<row xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Employee ID="509647174">
    <Hire-Date>2007-11-11</Hire-Date>
    <Name>
      <Last>Tamburello</Last>
      <First>Roberto</First>
      <Middle xsi:nil="true" />
    </Name>
  </Employee>
</row>
```

FOR XML RAW – Primer 1

- Primer:

```
SELECT d.DepartmentID,
  d.Name,
  d.GroupName
FROM HumanResources.Department d
WHERE d.DepartmentID IN ( 7, 8 )
FOR XML RAW ( 'MyNode' );
```

- Rezultat:

```
<MyNode DepartmentID="7" Name="Production"
  GroupName="Manufacturing" />
<MyNode DepartmentID="8" Name="Production Control"
  GroupName="Manufacturing" />
```

FOR XML RAW – Primer 2

- Primer:

```
SELECT d.DepartmentID,
  d.Name,
  d.GroupName
FROM HumanResources.Department d
WHERE d.DepartmentID IN ( 7, 8 )
FOR XML RAW ('MyNode'),
ROOT('TheRootNode'),
ELEMENTS;
```

FOR XML RAW – Primer 2

- Rezultat:

```
<TheRootNode>
  <MyNode>
    <DepartmentID>7</DepartmentID>
    <Name>Production</Name>
    <GroupName>Manufacturing</GroupName>
  </MyNode>
  <MyNode>
    <DepartmentID>8</DepartmentID>
    <Name>Production Control</Name>
    <GroupName>Manufacturing</GroupName>
  </MyNode>
</TheRootNode>
```

FOR XML AUTO – Primer 1

- Primer:

```
SELECT AddressID,
 AddressLine1,
 AddressLine2,
 City
FROM Person.Address
WHERE AddressID IN ( 532, 533 )
FOR XML AUTO;
```

- Rezultat:

```
<Person.Address AddressID="532"
 AddressLine1="#500-75 O'Connor Street" City="Ottawa" />
<Person.Address AddressID="533"
 AddressLine1="4400 March Road" City="Kanata" />
```

FOR XML AUTO – Primer 2

- Primer:

```
SELECT dep.DepartmentID,
  dep.Name,
  emp.BusinessEntityID
FROM HumanResources.Department dep
  INNER JOIN HumanResources.EmployeeDepartmentHistory emp
  ON dep.DepartmentID = emp.DepartmentID
WHERE emp.BusinessEntityID BETWEEN 20 AND 22
FOR XML AUTO,
ELEMENTS;
```

FOR XML AUTO – Primer 2

- Rezultat:

```
<dep>
  <DepartmentID>4</DepartmentID>
  <Name>Marketing</Name>
  <emp>
    <BusinessEntityID>20</BusinessEntityID>
  </emp>
  <emp>
    <BusinessEntityID>21</BusinessEntityID>
  </emp>
  <emp>
    <BusinessEntityID>22</BusinessEntityID>
  </emp>
</dep>
```

FOR XML EXPLICIT – Primer

- Primer:

```
SELECT 1 AS Tag, NULL AS Parent, p.ProductID AS [Product!1!ID],
 p.Name AS [Product!1!Name],
 p.ProductNumber AS [Product!1!Number], NULL AS [Quantity!2]
FROM Production.Product p
WHERE p.ProductID IN ( 770, 772 )
UNION ALL
SELECT 2 AS Tag, 1 AS Parent, p.ProductID,
 p.Name, p.ProductNumber, pi.Quantity
FROM Production.ProductInventory pi
 INNER JOIN Production.Product p
 ON p.ProductID = pi.ProductID
WHERE p.ProductID IN ( 770, 772 )
ORDER BY [Product!1!ID], [Product!1!Number], [Quantity!2]
FOR XML EXPLICIT;
```

FOR XML EXPLICIT – Primer

- Rezultat:

```
<Product ID="770" Name="Road-650 Black, 52" Number="BK-R50B-52">
  <Quantity>104</Quantity>
  <Quantity>123</Quantity>
</Product>
<Product ID="772" Name="Mountain-100 Silver, 42" Number="BK-M82S-42">
  <Quantity>65</Quantity>
  <Quantity>88</Quantity>
</Product>
```

XML tip

- XML tip se može dobiti implicitnim kastovanjem iz varchar ili nvarchar tipa ili eksplicitnim kastovanjem korišćenjem funkcija CAST i CONVERT.
- Primer:

```
DECLARE @imp_x xml, @exp_x xml, @source nvarchar(200);
SET @source = N'<?xml version = "1.0"?>
     <message>
     <to>SQL Server Team</to>
     <from>Michael Coles</from>
     <subject>Thanks</subject>
     <content>Thanks for the new version of SQL Server</content>
  </message>';
SET @imp_x = @source; /* Implicit conversion to xml */
SET @exp_x = CAST(@source AS xml); /* Explicit conversion to xml */
```

XML tip

- U okviru SQL Servera postoje dva tipa XML podataka:
  - DOCUMENT – dobro formatiran XML podatak
  - CONTENT – XML fragment koji ne mora da ima jedan XML koreni element.
- XML podaci mogu da zadovoljavaju konkretnu strukturu ili da budu nedefinisane stukture.
- Primer 1:

```
CREATE TABLE Production.Illustration (
    IllustrationID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
    Diagram xml,
    ModifiedDate datetime NOT NULL DEFAULT (GETDATE())
);
```

XML tip

- Primer 2:

```
CREATE TABLE Production.ProductModel(
 ProductModelID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
 [Name] name NOT NULL,
 CatalogDescription xml (CONTENT
    Production.ProductDescriptionSchemaCollection),
 Instructions xml (CONTENT
    Production.ManuInstructionsSchemaCollection),
 rowguid uniqueidentifier ROWGUIDCOL NOT NULL DEFAULT (NEWID()),
 ModifiedDate datetime NOT NULL DEFAULT (GETDATE())
);

CREATE XML SCHEMA COLLECTION
Production.ManuInstructionsSchemaCollection
AS N'<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    ...
 </xsd:schema>';
```

Metode nad XML tipom

- query() – Omogućava izvršavanje XQuery upita nad XML podatkom i vraća XML podatak nedefinisanog tipa.
- value() – Omogućava izvršavanje XQuery upita nad XML podatkom i vraća SQL skalarnu vrednost.
- exist() – Omogućava izvršavanje XQuery upita nad XML podatkom i vraća SQL vrednost 1 ako XQuery vraća rezultat, 0 ako XQuery ne vraća rezultat ili NULL ako je XML podatak NULL.
- modify() – Omogućava izvršavanje XML Data Manipulation upita (XML DML) nad XML podatkom. Metoda modify() se koristi samo u okviru SET naredbe.
- nodes() – Omogućava konvertovanje XML podatka u relacioni oblik.

Query metoda

- Primer:

```
DECLARE @x xml;
SET @x = N'<?xml version = "1.0"?>
   <bookstores company = "Borders Group">
      <store name = "Borders">
         <address>
           <street>2 PENN PLAZA</street>
           ...
         </address>
      </store>
      ...
   </bookstores>';
SELECT @x.query(N'//street');
```

Value metoda

- Primer:

```
DECLARE @x xml;
SET @x = N'<?xml version = "1.0"?>
 <book>
    <title>Harry Potter and the Deathly Hallows</title>
    <author>Rowling, J.K.</author>
    <isbn>0545010225</isbn>
    <release-date>2007-07-21Z</release-date>
    <price>34.99</price>
 </book>';
SELECT @x.value(N'(/book/price)[1]', 'decimal(5, 2)') AS Price,
 @x.value(N'(/book/release-date)[1]', 'date') AS Release_Date;
```

Exist metoda

- Primer:

```
DECLARE @x xml;
SET @x = N'<?xml version = "1.0"?>
 <dessert-menu>
    <item type = "pie">
       <name>Cherry Pie</name>
    </item>
    ...
 </dessert-menu>';
SELECT CASE @x.exist(N'/dessert-menu/item[@type eq "pie"]')
 WHEN 1 THEN N'Pie is on the menu'
 WHEN 0 THEN N'Pie is not on the menu'
 ELSE 'The XML instance is NULL'
END;
```

Nodes metoda

- Primer:

```
DECLARE @x xml;
SET @x = N'<?xml version = "1.0"?>
  <bill-of-materials>
      <finished-good name = "kiddie picnic table">
          <material name = "pine lumber">
              <item qty = "2">
                  <dimensions uom = "mm">50 x 50 x 1100</dimensions>
              </item>
              ...
          </material>
          ...
      </finished-good>
  </bill-of-materials>';
SELECT my_table.my_col.value(N'../@name', N'nvarchar(100)') AS Material,
  my_table.my_col.value(N'(./dimensions)[1]', N'nvarchar(50)') AS Dimensions,
  my_table.my_col.value(N'(./dimensions/@uom)[1]', N'nvarchar(10)') AS UOM,
  my_table.my_col.value(N'./@qty', N'int') AS Quantity
FROM @x.nodes(N'//item') AS my_table ( my_col );
```

Modify metoda

- Modify podržava delete, insert i replace value of.
- Primer:

```
DECLARE @x xml;
SET @x = N'<?xml version = "1.0"?>
  <inventory store-number = "9834">
     <product ean = "051500241776">
        <name>Jif Creamy Peanut Butter</name>
        <size>28 oz</size>
     </product>
     ...
  </inventory>';
SET @x.modify (N'delete
     (/inventory/product[@ean = "0086600000138"]/name)');
SELECT @x;
```

XQuery Axis

- Podržane opcije u Microsoft SQL Serveru:
  - attribute
  - child
  - descendant
  - descendant-or-self
  - parent
  - self
- Primer:

```
SELECT @x.query('/Geocode-Results/child::Result');
```

XQuery Predicates

- Omogućavaju da se filtrira rezultat.
- Navode se unutar [].
- Mogu da sadrže numeričke vrednosti ili uslove.
- Primeri:

```
SELECT @x.query('/Geocode-Results[1]/Result[2]');


SELECT @x.query('/Geocode-Results/Result
 [fn:not(@Name eq "Apple Inc.") and @Name eq "Microsoft Corp."]');
```

XQuery Quantified Expressions

- Podržane opcije every i some.
- Primer:

```
DECLARE @x xml;
SET @x = '';

SELECT @x.query ('some $x in (1, 2, 3)
 satisfies $x * $x = 9');

SELECT @x.query ('every $x in (1, 2, 3)
 satisfies $x * $x = 9');
```

FLWOR Expressions – Primer 1

- Primer (Varijanta 1):

```
SELECT Resume.query(
 'declare default element namespace
    "http://schemas.microsoft.com/sqlserver/2004/07/adventure-
    works/Resume";
 for $i in (/Resume/Employment)
 return ($i/Emp.JobTitle)')
FROM HumanResources.JobCandidate
WHERE JobCandidateId = 1;
```

FLWOR Expressions – Primer 1

- Primer (Varijanta 2):

```
SELECT Resume.query(
 'declare default element namespace
    "http://schemas.microsoft.com/sqlserver/2004/07/adventure-
    works/Resume";
  Resume/Employment/Emp.JobTitle')
FROM HumanResources.JobCandidate
WHERE JobCandidateId = 1;
```

FLWOR Expressions – Primer 1

- Rezultat:

```
<Emp.JobTitle
 xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
 works/Resume">Lead Machinist</Emp.JobTitle>
<Emp.JobTitle
 xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
 works/Resume">Machinist</Emp.JobTitle>
<Emp.JobTitle
 xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
 works/Resume">Assistant Machinist</Emp.JobTitle>
```

FLWOR Expressions – Primer 2

- Primer:

```
SELECT Resume.query(
 'declare default element namespace
    "http://schemas.microsoft.com/sqlserver/2004/07/adventure-
    works/Resume";
  for $i in (/Resume)
  let $j := $i/Employment
  return ($j/Emp.JobTitle)')
FROM HumanResources.JobCandidate
WHERE JobCandidateId = 1;
```

FLWOR Expressions – Primer 3

- Primer:

```
SELECT Resume.query(
 'declare default element namespace
    "http://schemas.microsoft.com/sqlserver/2004/07/adventure-
    works/Resume";
 for $i in (/Resume/Employment)
 where ($i/Emp.JobTitle ne "Machinist")
 return ($i/Emp.JobTitle)')
FROM HumanResources.JobCandidate
WHERE JobCandidateId = 1;
```

FLWOR Expressions – Primer 4

- Primer:

```
SELECT Resume.query(
 'declare default element namespace
    "http://schemas.microsoft.com/sqlserver/2004/07/adventure-
    works/Resume";
 for $i in (/Resume/Employment)
 let $j := ($i/Emp.JobTitle)
 order by fn:string($j) descending
 return ($j)')
FROM HumanResources.JobCandidate
WHERE JobCandidateId = 1;
```

Konstruisanje XML-a – Primer 1

- Primer:

```
DECLARE @x XML;
SET @x = N'';
SELECT @x.query ('
  <Measurement>
     One dozen is equal to { 5 + 7 }
  </Measurement>');
```

- Rezultat:

```
<Measurement> One dozen is equal to 12</Measurement>
```

Konstruisanje XML-a – Primer 2

- Primer:

```
SELECT Resume.query ('declare namespace
ns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
works/Resume";
<Education.History> {
 for $i in (/ns:Resume/ns:Education)
 return (
    <Level>
       <Degree>{ fn:data($i/ns:Edu.Level) }</Degree>
       <Date>{ fn:data($i/ns:Edu.EndDate) }</Date>
    </Level>
 )
} </Education.History>')
FROM HumanResources.JobCandidate
WHERE JobCandidateId = 2;
```

Konstruisanje XML-a – Primer 2

- Rezultat:

```
<Education.History>
  <Level>
    <Degree>Bachelor</Degree>
    <Date>1997-06-03Z</Date>
  </Level>
  <Level>
    <Degree>High School</Degree>
    <Date>1993-06-12Z</Date>
  </Level>
</Education.History>
```

Konstruisanje XML-a – Primer 3

- Primer:

```
SELECT Resume.query ('declare namespace
 ns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-
 works/Resume";
 for $i in (/ns:Resume/ns:Education)
 return (
     element Education.History {
         element Level {
             attribute School { fn:data($i/ns:Edu.School) },
             element Degree { fn:data($i/ns:Edu.Level) },
             element Date { fn:data($i/ns:Edu.EndDate) }
         }
     }
 )')
FROM HumanResources.JobCandidate
WHERE JobCandidateId = 2;
```

Konstruisanje XML-a – Primer 3

- Rezultat:

```
<Education.History>
  <Level School="Everglades State College">
    <Degree>Bachelor</Degree>
    <Date>1997-06-03Z</Date>
  </Level>
</Education.History>
<Education.History>
  <Level School="Evergreen High School">
    <Degree>High School</Degree>
    <Date>1993-06-12Z</Date>
  </Level>
</Education.History>
```

XML i SQL – Primer 1

- Primer:

```
SELECT ProductModelId, CatalogDescription.value ('declare
 namespace ns =
    "http://schemas.microsoft.com/sqlserver/2004/07/adventure-
 works/ProductModelDescription";
 (/ns:ProductDescription/ns:Specifications/Material/text())[1]',
 'VARCHAR(100)') AS Material
FROM Production.ProductModel
WHERE ProductModelId = 23;
```

XML i SQL – Primer 2

- Primer:

```
DECLARE @material VARCHAR(50);
SET @material = 'Alloy';

SELECT pm.ProductModelId, pm.Name, pm.CatalogDescription
FROM Production.ProductModel pm
WHERE pm.CatalogDescription.exist(
  'declare namespace ns =
     "http://schemas.microsoft.com/sqlserver/2004/07/adventure-
     works/ProductModelDescription";
   /ns:ProductDescription/ns:Specifications/Material
     [fn:contains( . , sql:variable("@material") ) ]') = 1;
```

XML i SQL – Primer 3

- Primer:

```
DECLARE @x xml;
SET @x = N'<?xml version = "1.0"?>
  <music>
      <album name = "Fullmetal Alchemist Complete Best">
          <genre>Rap</genre>
          <song artist="L&apos;arc-en-Ciel">Ready Steady Go</song>
          <song artist="Nana Kitade">Indelible Sin</song>
      </album>
</music>';
SET @x.modify ('replace value of
(/music/album[ @name = "Fullmetal Alchemist Complete Best"]
  /genre/text())[1]
with "Anime &amp; Manga"');
SELECT @x;
```