МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования «Казанский национальный исследовательский

технический университет им. А. Н. Туполева-КАИ»

(КНИТУ-КАИ)

Институт компьютерных технологий и защиты информации

(наименование института (факультета), филиала)

Кафедра Прикладной математики и информатики

(наименование кафедры)

(01.03.02) Прикладная математика и информатика

(шифр и наименование направления подготовки (специальности))

ОТЧЕТ

по лабораторным работам

по дисциплине «Программирование на TransactSQL»

Выполнил: студент группы 4318

Трифонов С.А.

Казань, 2024

**Лабораторная работа №1  
Краткая теория**

Возможность работы с Расширяемым Языком Разметки (XML) является общим требованием многих современных приложений. Часто разработчик приложений должен преобразовать данные между форматом XML и реляционным форматом, хранить XML и манипулировать XML-данными непосредственно в реляционной БД. Из этого модуля Вы узнаете, как преобразовать данные базы в формат XML, используя фразу FOR XML. Вы также узнаете, как преобразовать XML-документы для хранения в реляционных таблицах, используя функцию OPENXML. Наконец, Вы узнаете, как использовать тип данных xml, чтобы хранить XML-документы в базе данных в родном формате и как выполнить запросы к xml данным и их модификацию.

Фраза FOR XML является центральной для поиска данных XML в SQL Server 2005. Эта фраза является инструкцией для SQL Server, чтобы возвратить данные в формате XML, а не в виде набора строк. Разработчики приложений могут построить решения, которые выполняют извлечение бизнес XML документы, таких как заказы, счета, или каталоги непосредственно из базы данных. В этом уроке описывается, как использовать фразу FOR XML и ее различные опции, чтобы сформировать данные в формате XML.

Вы можете использовать фразу FOR XML в операторе SELECT языка Transact-SQL, чтобы получить данные в формате XML вместо строк и столбцов. Вы можете управляеть форматом XML, определяя один из четырех режимов: RAW, AUTO, EXPLICIT или PATH. Кроме того, Вы можете определить различные опции для управления выводом.

Фраза FOR XML прилагается к оператору SELECT, в соответствии со следующим синтаксисом:

FOR XML

{

{ RAW [ ( 'ElementName' ) ] | AUTO }

[

<CommonDirectives>

[ , { XMLDATA | XMLSCHEMA [ ( 'TargetNameSpaceURI' ) ] } ]

[ , ELEMENTS [ XSINIL | ABSENT ]

]

| EXPLICIT

[

<CommonDirectives>

[ , XMLDATA ]

]

| PATH [ ( 'ElementName' ) ]

[

<CommonDirectives>

[ , ELEMENTS [ XSINIL | ABSENT ] ]

]

}

<CommonDirectives> ::=

[ , BINARY BASE64 ]

[ , TYPE ]

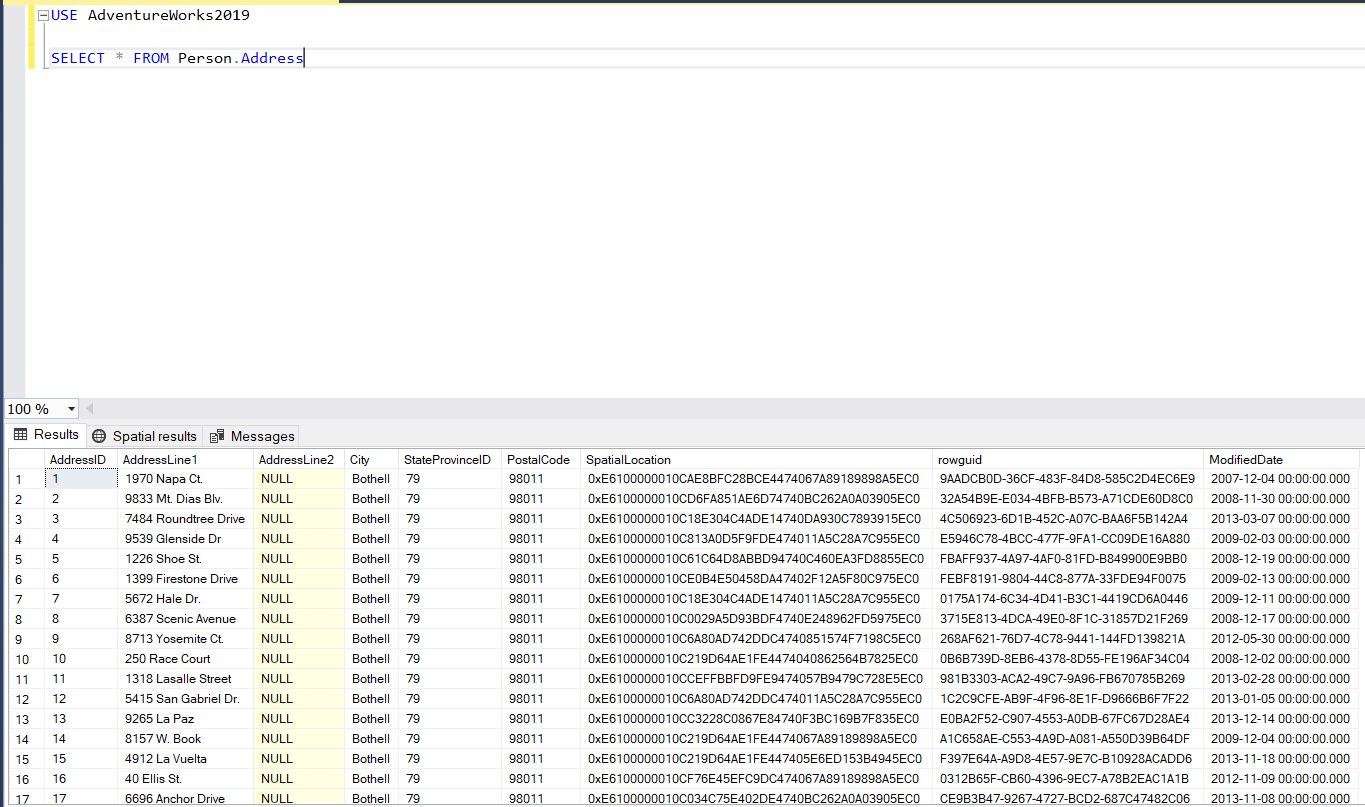
[ , ROOT [ ( 'RootName' ) ] ]

**Задание**

1. Изучить материалы урока 1 из файла "Теоретические материалы - Работа с XML".
2. Ответить на контрольные вопросы 1-6.
3. Выполнить все примеры, приведенные в уроке 1 файла "Теоретические материалы - Работа с XML".
4. Создать базу данных с 3-5 таблицами по индивидуальному заданию. Можно использовать БД, созданную для курсовой работы по дисциплине "Базы данных".
5. Для каждого режима (RAW, AUTO, PATH, EXPLICIT) создать по одному-два запроса, переводящему реляционные данные в формат XML. При этом должны быть запросы, формирующие вложенные XML-документы, содержащие данные 2-3 таблиц.

**Ход работы**

Первым делом я скачал базу данных AdventureWorks2019 с официального сайта Microsoft.



Далее приступаю к выполнению примеров, приведенных в уроке 1 файла "Теоретические материалы - Работа с XML"

**Запросы в режиме RAW**

В 1 примере показывается, как Вы можете получить XML-фрагмент, содержащий данные заказа при использовании фразы FOR XML в режиме RAW.

SELECT Cust.CustomerID CustID, SalesOrderID

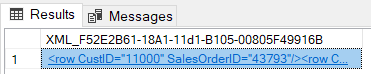
FROM Sales.Customer Cust JOIN Sales.SalesOrderHeader [Order]

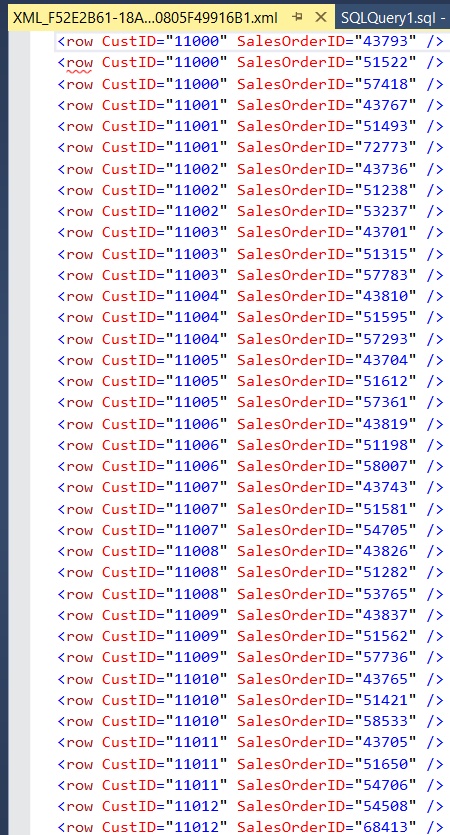
ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML RAW

Этот запрос выводит XML-фрагмент в формате, который содержит сгенерированные элементы, показанные в





В следующем примере показывается, как можно получить те же данные в виде элементов вместо атрибутов, определяя опцию ELEMENTS.

SELECT Cust.CustomerID CustID, SalesOrderID

FROM Sales.Customer Cust JOIN Sales.SalesOrderHeader [Order]

ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML RAW, ELEMENTS

Данный запрос выводит XML-фрагмент в формате:

<row>

<CustID>11000</CustID>

<SalesOrderID>43793</SalesOrderID>

</row>

<row>

<CustID>11000</CustID>

<SalesOrderID>51522</SalesOrderID>

</row>

<row>

<CustID>11000</CustID>

<SalesOrderID>57418</SalesOrderID>

</row>

<row>

<CustID>11001</CustID>

<SalesOrderID>43767</SalesOrderID>

</row>

<row>

<CustID>11001</CustID>

<SalesOrderID>51493</SalesOrderID>

</row>

…

В следующем примере показывается, как Вы можете получить те же самые данные, используя корневой элемент, определенный с помощью опции ROOT и изменяя имя элемента строки при помощи необязательного аргумента опции RAW.

SELECT Cust.CustomerID CustID, SalesOrderID

FROM Sales.Customer Cust JOIN Sales.SalesOrderHeader [Order]

ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML RAW('Order'), ROOT('Orders')

Этот запрос предоставляет правильно построенный XML документ в формате:

<Orders>

<Order CustID="11000" SalesOrderID="43793" />

<Order CustID="11000" SalesOrderID="51522" />

<Order CustID="11000" SalesOrderID="57418" />

<Order CustID="11001" SalesOrderID="43767" />

<Order CustID="11001" SalesOrderID="51493" />

<Order CustID="11001" SalesOrderID="72773" />

<Order CustID="11002" SalesOrderID="43736" />

<Order CustID="11002" SalesOrderID="51238" />

<Order CustID="11002" SalesOrderID="53237" />

<Order CustID="11003" SalesOrderID="43701" />

<Order CustID="11003" SalesOrderID="51315" />

<Order CustID="11003" SalesOrderID="57783" />

<Order CustID="11004" SalesOrderID="43810" />

<Order CustID="11004" SalesOrderID="51595" />

<Order CustID="11004" SalesOrderID="57293" />

<Order CustID="11005" SalesOrderID="43704" />

<Order CustID="11005" SalesOrderID="51612" />

<Order CustID="11005" SalesOrderID="57361" />

<Order CustID="11006" SalesOrderID="43819" />

<Order CustID="11006" SalesOrderID="51198" />

**Запросы в режиме AUTO**

Получение вложенных данных при использовании режима AUТО

SELECT Cust.CustomerID CustID, SalesOrderID

FROM Sales.Customer Cust JOIN Sales.SalesOrderHeader [Order]

ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML AUTO

Этот запрос производит фрагмент XML в формате

<Cust CustID="11000">

<Order SalesOrderID="43793" />

<Order SalesOrderID="51522" />

<Order SalesOrderID="57418" />

</Cust>

<Cust CustID="11001">

<Order SalesOrderID="43767" />

<Order SalesOrderID="51493" />

<Order SalesOrderID="72773" />

</Cust>

<Cust CustID="11002">

<Order SalesOrderID="43736" />

<Order SalesOrderID="51238" />

<Order SalesOrderID="53237" />

</Cust>

Получение данных в виде элементов

SELECT Cust.CustomerID CustID, SalesOrderID

FROM Sales.Customer Cust JOIN Sales.SalesOrderHeader [Order]

ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML AUTO, ELEMENTS

Этот запрос производит фрагмент XML в формате

<Cust>

<CustID>11000</CustID>

<Order>

<SalesOrderID>43793</SalesOrderID>

</Order>

<Order>

<SalesOrderID>51522</SalesOrderID>

</Order>

<Order>

<SalesOrderID>57418</SalesOrderID>

</Order>

</Cust>

Так же, как и в режиме RAW, Вы можете использовать опцию ROOT, как показано в следующем примере.

SELECT Cust.CustomerID CustID, SalesOrderID

FROM Sales.Customer Cust JOIN Sales.SalesOrderHeader [Order]

ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML AUTO, ELEMENTS, ROOT('Orders')

Этот запрос представляет документ XML в формате

<Orders>

<Cust>

<CustID>11000</CustID>

<Order>

<SalesOrderID>43793</SalesOrderID>

</Order>

<Order>

<SalesOrderID>51522</SalesOrderID>

</Order>

<Order>

<SalesOrderID>57418</SalesOrderID>

</Order>

</Cust>

<Cust>

...

</Orders>

**Запросы в режиме EXPLICIT**

Создадим запрос для построения универсальной таблицы

SELECT 1 AS Tag,

NULL AS Parent,

SalesOrderID AS [Invoice!1!InvoiceNo],

OrderDate AS [Invoice!1!Date!Element]

FROM Sales.SalesOrderHeader

FOR XML EXPLICIT

Этот пример создает XML, показанный ранее в этой теме, комбинируя атрибуты и элементы для столбцов SalesOrderID для OrderDate.

<Invoice InvoiceNo="43659">

<Date>2011-05-31T00:00:00</Date>

</Invoice>

<Invoice InvoiceNo="43660">

<Date>2011-05-31T00:00:00</Date>

</Invoice>

<Invoice InvoiceNo="43661">

<Date>2011-05-31T00:00:00</Date>

</Invoice>

<Invoice InvoiceNo="43662">

<Date>2011-05-31T00:00:00</Date>

</Invoice>

**Запросы в режиме PATH**

Получение данных при использовании режима PATH

SELECT Person.BusinessEntityID "@EmpID",

FirstName "EmpName/First",

LastName "EmpName/Last"

FROM Person.Person INNER JOIN

HumanResources.Employee ON Person.Person.BusinessEntityID = HumanResources.Employee.BusinessEntityID

FOR XML PATH

Результат запроса:

<row EmpID="168">

<EmpName>

<First>Garrett</First>

<Last>Young</Last>

</EmpName>

</row>

<row EmpID="50">

<EmpName>

<First>Sidney</First>

<Last>Higa</Last>

</EmpName>

</row>

Изменение имени элемента row

SELECT Person.BusinessEntityID "@EmpID",

FirstName "EmpName/First",

LastName "EmpName/Last"

FROM Person.Person INNER JOIN

HumanResources.Employee ON Person.Person.BusinessEntityID = HumanResources.Employee.BusinessEntityID

FOR XML PATH('Employee')

Результат запроса:

<Employee EmpID="168">

<EmpName>

<First>Garrett</First>

<Last>Young</Last>

</EmpName>

</Employee>

<Employee EmpID="50">

<EmpName>

<First>Sidney</First>

<Last>Higa</Last>

</EmpName>

</Employee>

Использование режима AUTO для получения вложенных XML

SELECT Cust.CustomerID CustID, SalesOrderID

FROM Sales.Customer Cust JOIN Sales.SalesOrderHeader [Order]

ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML AUTO, ELEMENTS

Результат запроса:

<Cust>

<CustID>11000</CustID>

<Order>

<SalesOrderID>43793</SalesOrderID>

</Order>

<Order>

<SalesOrderID>51522</SalesOrderID>

</Order>

<Order>

<SalesOrderID>57418</SalesOrderID>

</Order>

</Cust>

Использование TYPE для получения типа данных xml в подзапросе.

Следующий пример показывает, как использовать директиву TYPE, чтобы вкладывать запросы FOR XML.

SELECT Name CategoryName,

(SELECT Name SubCategoryName

FROM Production.ProductSubCategory SubCategory

WHERE SubCategory.ProductCategoryID=Category.ProductCategoryID

FOR XML AUTO, TYPE, ELEMENTS)

FROM Production.ProductCategory Category

FOR XML AUTO

Результат запроса:

<Category CategoryName="Accessories">

<SubCategory>

<SubCategoryName>Bike Racks</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Bike Stands</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Bottles and Cages</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Cleaners</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Fenders</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Helmets</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Hydration Packs</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Lights</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Locks</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Panniers</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Pumps</SubCategoryName>

</SubCategory>

<SubCategory>

<SubCategoryName>Tires and Tubes</SubCategoryName>

</SubCategory>

</Category>

Вложение таблиц при использовании режима EXPLICIT

SELECT 1 AS Tag,

NULL AS Parent,

SalesOrderID AS [Invoice!1!InvoiceNo],

OrderDate AS [Invoice!1!Date!Element],

NULL AS [LineItem!2!ProductID],

NULL AS [LineItem!2]

FROM Sales.SalesOrderHeader

UNION ALL

SELECT 2 AS Tag,

1 AS Parent,

OrderDetail.SalesOrderID,

NULL,

OrderDetail.ProductID,

Product.Name

FROM Sales.SalesOrderDetail OrderDetail JOIN

Sales.SalesOrderHeader OrderHeader

ON OrderDetail.SalesOrderID= OrderHeader.SalesOrderID

JOIN Production.Product Product

ON OrderDetail.ProductID = Product.ProductID

ORDER BY [Invoice!1!InvoiceNo], [LineItem!2!ProductID]

FOR XML EXPLICIT

Результат запроса:

<Invoice InvoiceNo="43659">

<Date>2011-05-31T00:00:00</Date>

<LineItem ProductID="709">Mountain Bike Socks, M</LineItem>

<LineItem ProductID="711">Sport-100 Helmet, Blue</LineItem>

<LineItem ProductID="712">AWC Logo Cap</LineItem>

<LineItem ProductID="714">Long-Sleeve Logo Jersey, M</LineItem>

<LineItem ProductID="716">Long-Sleeve Logo Jersey, XL</LineItem>

<LineItem ProductID="771">Mountain-100 Silver, 38</LineItem>

<LineItem ProductID="772">Mountain-100 Silver, 42</LineItem>

<LineItem ProductID="773">Mountain-100 Silver, 44</LineItem>

<LineItem ProductID="774">Mountain-100 Silver, 48</LineItem>

<LineItem ProductID="776">Mountain-100 Black, 42</LineItem>

<LineItem ProductID="777">Mountain-100 Black, 44</LineItem>

<LineItem ProductID="778">Mountain-100 Black, 48</LineItem>

</Invoice>

Теперь перейдем к выполнению индивидуального задания. Я создал базу данных и таблицы к нему. Заполнил их данными:

CREATE TABLE AGENTS

(

AGENT\_CODE CHAR(6) NOT NULL PRIMARY KEY,

AGENT\_NAME VARCHAR(40),

WORKING\_AREA VARCHAR(35),

COMMISSION DECIMAL(10,2),

PHONE\_NO VARCHAR(15),

COUNTRY VARCHAR(25)

);

INSERT INTO AGENTS VALUES ('A007', 'Ramasundar', 'Bangalore', '0.15', '077-25814763', '');

INSERT INTO AGENTS VALUES ('A003', 'Alex ', 'London', '0.13', '075-12458969', '');

INSERT INTO AGENTS VALUES ('A008', 'Alford', 'New York', '0.12', '044-25874365', '');

INSERT INTO AGENTS VALUES ('A011', 'Ravi Kumar', 'Bangalore', '0.15', '077-45625874', '');

INSERT INTO AGENTS VALUES ('A010', 'Santakumar', 'Chennai', '0.14', '007-22388644', '');

INSERT INTO AGENTS VALUES ('A012', 'Lucida', 'San Jose', '0.12', '044-52981425', '');

INSERT INTO AGENTS VALUES ('A005', 'Anderson', 'Brisban', '0.13', '045-21447739', '');

INSERT INTO AGENTS VALUES ('A001', 'Subbarao', 'Bangalore', '0.14', '077-12346674', '');

INSERT INTO AGENTS VALUES ('A002', 'Mukesh', 'Mumbai', '0.11', '029-12358964', '');

INSERT INTO AGENTS VALUES ('A006', 'McDen', 'London', '0.15', '078-22255588', '');

INSERT INTO AGENTS VALUES ('A004', 'Ivan', 'Torento', '0.15', '008-22544166', '');

INSERT INTO AGENTS VALUES ('A009', 'Benjamin', 'Hampshair', '0.11', '008-22536178', '');

CREATE TABLE "CUSTOMER"

( "CUST\_CODE" VARCHAR(6) NOT NULL PRIMARY KEY,

"CUST\_NAME" VARCHAR(40) NOT NULL,

"CUST\_CITY" CHAR(35),

"WORKING\_AREA" VARCHAR(35) NOT NULL,

"CUST\_COUNTRY" VARCHAR(20) NOT NULL,

"GRADE" DECIMAL,

"OPENING\_AMT" DECIMAL(12,2) NOT NULL,

"RECEIVE\_AMT" DECIMAL(12,2) NOT NULL,

"PAYMENT\_AMT" DECIMAL(12,2) NOT NULL,

"OUTSTANDING\_AMT" DECIMAL(12,2) NOT NULL,

"PHONE\_NO" VARCHAR(17) NOT NULL,

"AGENT\_CODE" CHAR(6) NOT NULL REFERENCES AGENTS

);

INSERT INTO CUSTOMER VALUES ('C00013', 'Holmes', 'London', 'London', 'UK', '2', '6000.00', '5000.00', '7000.00', '4000.00', 'BBBBBBB', 'A003');

INSERT INTO CUSTOMER VALUES ('C00001', 'Micheal', 'New York', 'New York', 'USA', '2', '3000.00', '5000.00', '2000.00', '6000.00', 'CCCCCCC', 'A008');

INSERT INTO CUSTOMER VALUES ('C00020', 'Albert', 'New York', 'New York', 'USA', '3', '5000.00', '7000.00', '6000.00', '6000.00', 'BBBBSBB', 'A008');

INSERT INTO CUSTOMER VALUES ('C00025', 'Ravindran', 'Bangalore', 'Bangalore', 'India', '2', '5000.00', '7000.00', '4000.00', '8000.00', 'AVAVAVA', 'A011');

INSERT INTO CUSTOMER VALUES ('C00024', 'Cook', 'London', 'London', 'UK', '2', '4000.00', '9000.00', '7000.00', '6000.00', 'FSDDSDF', 'A006');

INSERT INTO CUSTOMER VALUES ('C00015', 'Stuart', 'London', 'London', 'UK', '1', '6000.00', '8000.00', '3000.00', '11000.00', 'GFSGERS', 'A003');

INSERT INTO CUSTOMER VALUES ('C00002', 'Bolt', 'New York', 'New York', 'USA', '3', '5000.00', '7000.00', '9000.00', '3000.00', 'DDNRDRH', 'A008');

INSERT INTO CUSTOMER VALUES ('C00018', 'Fleming', 'Brisban', 'Brisban', 'Australia', '2', '7000.00', '7000.00', '9000.00', '5000.00', 'NHBGVFC', 'A005');

INSERT INTO CUSTOMER VALUES ('C00021', 'Jacks', 'Brisban', 'Brisban', 'Australia', '1', '7000.00', '7000.00', '7000.00', '7000.00', 'WERTGDF', 'A005');

INSERT INTO CUSTOMER VALUES ('C00019', 'Yearannaidu', 'Chennai', 'Chennai', 'India', '1', '8000.00', '7000.00', '7000.00', '8000.00', 'ZZZZBFV', 'A010');

INSERT INTO CUSTOMER VALUES ('C00005', 'Sasikant', 'Mumbai', 'Mumbai', 'India', '1', '7000.00', '11000.00', '7000.00', '11000.00', '147-25896312', 'A002');

INSERT INTO CUSTOMER VALUES ('C00007', 'Ramanathan', 'Chennai', 'Chennai', 'India', '1', '7000.00', '11000.00', '9000.00', '9000.00', 'GHRDWSD', 'A010');

INSERT INTO CUSTOMER VALUES ('C00022', 'Avinash', 'Mumbai', 'Mumbai', 'India', '2', '7000.00', '11000.00', '9000.00', '9000.00', '113-12345678','A002');

INSERT INTO CUSTOMER VALUES ('C00004', 'Winston', 'Brisban', 'Brisban', 'Australia', '1', '5000.00', '8000.00', '7000.00', '6000.00', 'AAAAAAA', 'A005');

INSERT INTO CUSTOMER VALUES ('C00023', 'Karl', 'London', 'London', 'UK', '0', '4000.00', '6000.00', '7000.00', '3000.00', 'AAAABAA', 'A006');

INSERT INTO CUSTOMER VALUES ('C00006', 'Shilton', 'Torento', 'Torento', 'Canada', '1', '10000.00', '7000.00', '6000.00', '11000.00', 'DDDDDDD', 'A004');

INSERT INTO CUSTOMER VALUES ('C00010', 'Charles', 'Hampshair', 'Hampshair', 'UK', '3', '6000.00', '4000.00', '5000.00', '5000.00', 'MMMMMMM', 'A009');

INSERT INTO CUSTOMER VALUES ('C00017', 'Srinivas', 'Bangalore', 'Bangalore', 'India', '2', '8000.00', '4000.00', '3000.00', '9000.00', 'AAAAAAB', 'A007');

INSERT INTO CUSTOMER VALUES ('C00012', 'Steven', 'San Jose', 'San Jose', 'USA', '1', '5000.00', '7000.00', '9000.00', '3000.00', 'KRFYGJK', 'A012');

INSERT INTO CUSTOMER VALUES ('C00008', 'Karolina', 'Torento', 'Torento', 'Canada', '1', '7000.00', '7000.00', '9000.00', '5000.00', 'HJKORED', 'A004');

INSERT INTO CUSTOMER VALUES ('C00003', 'Martin', 'Torento', 'Torento', 'Canada', '2', '8000.00', '7000.00', '7000.00', '8000.00', 'MJYURFD', 'A004');

INSERT INTO CUSTOMER VALUES ('C00009', 'Ramesh', 'Mumbai', 'Mumbai', 'India', '3', '8000.00', '7000.00', '3000.00', '12000.00', 'Phone No', 'A002');

INSERT INTO CUSTOMER VALUES ('C00014', 'Rangarappa', 'Bangalore', 'Bangalore', 'India', '2', '8000.00', '11000.00', '7000.00', '12000.00', 'AAAATGF', 'A001');

INSERT INTO CUSTOMER VALUES ('C00016', 'Venkatpati', 'Bangalore', 'Bangalore', 'India', '2', '8000.00', '11000.00', '7000.00', '12000.00', 'JRTVFDD', 'A007');

INSERT INTO CUSTOMER VALUES ('C00011', 'Sundariya', 'Chennai', 'Chennai', 'India', '3', '7000.00', '11000.00', '7000.00', '11000.00', 'PPHGRTS', 'A010');

CREATE TABLE "ORDERS"

(

"ORD\_NUM" DECIMAL(6,0) NOT NULL PRIMARY KEY,

"ORD\_AMOUNT" DECIMAL(12,2) NOT NULL,

"ADVANCE\_AMOUNT" DECIMAL(12,2) NOT NULL,

"ORD\_DATE" DATE NOT NULL,

"CUST\_CODE" VARCHAR(6) NOT NULL REFERENCES CUSTOMER,

"AGENT\_CODE" CHAR(6) NOT NULL REFERENCES AGENTS,

"ORD\_DESCRIPTION" VARCHAR(60) NOT NULL

);

С помощью команды «for xml» перевел данные, полученные используя команду select, в xml формат. Были использованы разные опции для конвертации: RAW, AUTO, PATH, EXPLICIT. А приставка elements, выводила данные в более читабельном виде.

1)Опция Raw.

SELECT

AGENT\_CODE AS "AgentCode",

AGENT\_NAME AS "AgentName",

WORKING\_AREA AS "WorkingArea",

COMMISSION AS "Commission",

PHONE\_NO AS "PhoneNo",

COUNTRY AS "Country"

FROM AGENTS

FOR XML RAW ('Agent'), ROOT ('Agents')

<Agents>

<Agent AgentCode="A001 " AgentName="Subbarao" WorkingArea="Bangalore" Commission="0.14" PhoneNo="077-12346674" Country="" />

<Agent AgentCode="A002 " AgentName="Mukesh" WorkingArea="Mumbai" Commission="0.11" PhoneNo="029-12358964" Country="" />

<Agent AgentCode="A003 " AgentName="Alex " WorkingArea="London" Commission="0.13" PhoneNo="075-12458969" Country="" />

<Agent AgentCode="A004 " AgentName="Ivan" WorkingArea="Torento" Commission="0.15" PhoneNo="008-22544166" Country="" />

<Agent AgentCode="A005 " AgentName="Anderson" WorkingArea="Brisban" Commission="0.13" PhoneNo="045-21447739" Country="" />

<Agent AgentCode="A006 " AgentName="McDen" WorkingArea="London" Commission="0.15" PhoneNo="078-22255588" Country="" />

<Agent AgentCode="A007 " AgentName="Ramasundar" WorkingArea="Bangalore" Commission="0.15" PhoneNo="077-25814763" Country="" />

<Agent AgentCode="A008 " AgentName="Alford" WorkingArea="New York" Commission="0.12" PhoneNo="044-25874365" Country="" />

<Agent AgentCode="A009 " AgentName="Benjamin" WorkingArea="Hampshair" Commission="0.11" PhoneNo="008-22536178" Country="" />

<Agent AgentCode="A010 " AgentName="Santakumar" WorkingArea="Chennai" Commission="0.14" PhoneNo="007-22388644" Country="" />

<Agent AgentCode="A011 " AgentName="Ravi Kumar" WorkingArea="Bangalore" Commission="0.15" PhoneNo="077-45625874" Country="" />

<Agent AgentCode="A012 " AgentName="Lucida" WorkingArea="San Jose" Commission="0.12" PhoneNo="044-52981425" Country="" />

</Agents>

2) Опция Raw. С применением elements.  
 SELECT

AGENT\_CODE AS "AgentCode",

AGENT\_NAME AS "AgentName",

WORKING\_AREA AS "WorkingArea",

COMMISSION AS "Commission",

PHONE\_NO AS "PhoneNo",

COUNTRY AS "Country"

FROM AGENTS

FOR XML RAW, ELEMENTS

<row>

<AgentCode>A001 </AgentCode>

<AgentName>Subbarao</AgentName>

<WorkingArea>Bangalore</WorkingArea>

<Commission>0.14</Commission>

<PhoneNo>077-12346674</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A002 </AgentCode>

<AgentName>Mukesh</AgentName>

<WorkingArea>Mumbai</WorkingArea>

<Commission>0.11</Commission>

<PhoneNo>029-12358964</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A003 </AgentCode>

<AgentName>Alex </AgentName>

<WorkingArea>London</WorkingArea>

<Commission>0.13</Commission>

<PhoneNo>075-12458969</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A004 </AgentCode>

<AgentName>Ivan</AgentName>

<WorkingArea>Torento</WorkingArea>

<Commission>0.15</Commission>

<PhoneNo>008-22544166</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A005 </AgentCode>

<AgentName>Anderson</AgentName>

<WorkingArea>Brisban</WorkingArea>

<Commission>0.13</Commission>

<PhoneNo>045-21447739</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A006 </AgentCode>

<AgentName>McDen</AgentName>

<WorkingArea>London</WorkingArea>

<Commission>0.15</Commission>

<PhoneNo>078-22255588</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A007 </AgentCode>

<AgentName>Ramasundar</AgentName>

<WorkingArea>Bangalore</WorkingArea>

<Commission>0.15</Commission>

<PhoneNo>077-25814763</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A008 </AgentCode>

<AgentName>Alford</AgentName>

<WorkingArea>New York</WorkingArea>

<Commission>0.12</Commission>

<PhoneNo>044-25874365</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A009 </AgentCode>

<AgentName>Benjamin</AgentName>

<WorkingArea>Hampshair</WorkingArea>

<Commission>0.11</Commission>

<PhoneNo>008-22536178</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A010 </AgentCode>

<AgentName>Santakumar</AgentName>

<WorkingArea>Chennai</WorkingArea>

<Commission>0.14</Commission>

<PhoneNo>007-22388644</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A011 </AgentCode>

<AgentName>Ravi Kumar</AgentName>

<WorkingArea>Bangalore</WorkingArea>

<Commission>0.15</Commission>

<PhoneNo>077-45625874</PhoneNo>

<Country></Country>

</row>

<row>

<AgentCode>A012 </AgentCode>

<AgentName>Lucida</AgentName>

<WorkingArea>San Jose</WorkingArea>

<Commission>0.12</Commission>

<PhoneNo>044-52981425</PhoneNo>

<Country></Country>

</row>

3)Опция auto, elements.   
SELECT

AGENT\_CODE,

AGENT\_NAME,

WORKING\_AREA,

COMMISSION,

PHONE\_NO,

COUNTRY

FROM AGENTS

FOR XML AUTO,ELEMENTS

<AGENTS>

<AGENT\_CODE>A001 </AGENT\_CODE>

<AGENT\_NAME>Subbarao</AGENT\_NAME>

<WORKING\_AREA>Bangalore</WORKING\_AREA>

<COMMISSION>0.14</COMMISSION>

<PHONE\_NO>077-12346674</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A002 </AGENT\_CODE>

<AGENT\_NAME>Mukesh</AGENT\_NAME>

<WORKING\_AREA>Mumbai</WORKING\_AREA>

<COMMISSION>0.11</COMMISSION>

<PHONE\_NO>029-12358964</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A003 </AGENT\_CODE>

<AGENT\_NAME>Alex </AGENT\_NAME>

<WORKING\_AREA>London</WORKING\_AREA>

<COMMISSION>0.13</COMMISSION>

<PHONE\_NO>075-12458969</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A004 </AGENT\_CODE>

<AGENT\_NAME>Ivan</AGENT\_NAME>

<WORKING\_AREA>Torento</WORKING\_AREA>

<COMMISSION>0.15</COMMISSION>

<PHONE\_NO>008-22544166</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A005 </AGENT\_CODE>

<AGENT\_NAME>Anderson</AGENT\_NAME>

<WORKING\_AREA>Brisban</WORKING\_AREA>

<COMMISSION>0.13</COMMISSION>

<PHONE\_NO>045-21447739</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A006 </AGENT\_CODE>

<AGENT\_NAME>McDen</AGENT\_NAME>

<WORKING\_AREA>London</WORKING\_AREA>

<COMMISSION>0.15</COMMISSION>

<PHONE\_NO>078-22255588</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A007 </AGENT\_CODE>

<AGENT\_NAME>Ramasundar</AGENT\_NAME>

<WORKING\_AREA>Bangalore</WORKING\_AREA>

<COMMISSION>0.15</COMMISSION>

<PHONE\_NO>077-25814763</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A008 </AGENT\_CODE>

<AGENT\_NAME>Alford</AGENT\_NAME>

<WORKING\_AREA>New York</WORKING\_AREA>

<COMMISSION>0.12</COMMISSION>

<PHONE\_NO>044-25874365</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A009 </AGENT\_CODE>

<AGENT\_NAME>Benjamin</AGENT\_NAME>

<WORKING\_AREA>Hampshair</WORKING\_AREA>

<COMMISSION>0.11</COMMISSION>

<PHONE\_NO>008-22536178</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A010 </AGENT\_CODE>

<AGENT\_NAME>Santakumar</AGENT\_NAME>

<WORKING\_AREA>Chennai</WORKING\_AREA>

<COMMISSION>0.14</COMMISSION>

<PHONE\_NO>007-22388644</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A011 </AGENT\_CODE>

<AGENT\_NAME>Ravi Kumar</AGENT\_NAME>

<WORKING\_AREA>Bangalore</WORKING\_AREA>

<COMMISSION>0.15</COMMISSION>

<PHONE\_NO>077-45625874</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

<AGENTS>

<AGENT\_CODE>A012 </AGENT\_CODE>

<AGENT\_NAME>Lucida</AGENT\_NAME>

<WORKING\_AREA>San Jose</WORKING\_AREA>

<COMMISSION>0.12</COMMISSION>

<PHONE\_NO>044-52981425</PHONE\_NO>

<COUNTRY></COUNTRY>

</AGENTS>

4) Опция auto. Получена вложенная xml

SELECT

Cust.CustomerID AS "@CustID",

(

SELECT

SalesOrderID

FROM Sales.SalesOrderHeader [Order]

WHERE Cust.CustomerID = [Order].CustomerID

FOR XML AUTO, TYPE

) AS "SalesOrders"

FROM Sales.Customer Cust

JOIN Sales.SalesOrderHeader [Order] ON Cust.CustomerID = [Order].CustomerID

ORDER BY Cust.CustomerID

FOR XML AUTO, ROOT('Customers');

<Customers>

<Cust \_x0040\_CustID="11000">

<SalesOrders>

<Order SalesOrderID="43793" />

<Order SalesOrderID="51522" />

<Order SalesOrderID="57418" />

</SalesOrders>

</Cust>

<Cust \_x0040\_CustID="11000">

<SalesOrders>

<Order SalesOrderID="43793" />

<Order SalesOrderID="51522" />

<Order SalesOrderID="57418" />

</SalesOrders>

</Cust>

<Cust \_x0040\_CustID="11000">

<SalesOrders>

<Order SalesOrderID="43793" />

<Order SalesOrderID="51522" />

<Order SalesOrderID="57418" />

</SalesOrders>

</Cust>

<Cust \_x0040\_CustID="11001">

<SalesOrders>

<Order SalesOrderID="43767" />

<Order SalesOrderID="51493" />

<Order SalesOrderID="72773" />

</SalesOrders>

</Cust>

<Cust \_x0040\_CustID="11001">

<SalesOrders>

<Order SalesOrderID="43767" />

<Order SalesOrderID="51493" />

<Order SalesOrderID="72773" />

</SalesOrders>

</Cust>

5)Опция path.  
SELECT

AGENT\_CODE AS "@AgentCode",

AGENT\_NAME AS "Agent/AgentName",

WORKING\_AREA AS "Agent/WorkingArea",

COMMISSION AS "Agent/Commission",

PHONE\_NO AS "Agent/PhoneNo",

COUNTRY AS "Agent/Country"

FROM AGENTS

FOR XML PATH('Agent')

<Agent AgentCode="A001 ">

<Agent>

<AgentName>Subbarao</AgentName>

<WorkingArea>Bangalore</WorkingArea>

<Commission>0.14</Commission>

<PhoneNo>077-12346674</PhoneNo>

<Country></Country>

</Agent>

</Agent>

<Agent AgentCode="A002 ">

<Agent>

<AgentName>Mukesh</AgentName>

<WorkingArea>Mumbai</WorkingArea>

<Commission>0.11</Commission>

<PhoneNo>029-12358964</PhoneNo>

<Country></Country>

</Agent>

</Agent>

<Agent AgentCode="A003 ">

<Agent>

<AgentName>Alex </AgentName>

<WorkingArea>London</WorkingArea>

<Commission>0.13</Commission>

<PhoneNo>075-12458969</PhoneNo>

<Country></Country>

</Agent>

</Agent>

<Agent AgentCode="A004 ">

<Agent>

<AgentName>Ivan</AgentName>

<WorkingArea>Torento</WorkingArea>

<Commission>0.15</Commission>

<PhoneNo>008-22544166</PhoneNo>

<Country></Country>

</Agent>

</Agent>

<Agent AgentCode="A005 ">

<Agent>

<AgentName>Anderson</AgentName>

<WorkingArea>Brisban</WorkingArea>

<Commission>0.13</Commission>

<PhoneNo>045-21447739</PhoneNo>

<Country></Country>

</Agent>

</Agent>

<Agent AgentCode="A006 ">

<Agent>

<AgentName>McDen</AgentName>

<WorkingArea>London</WorkingArea>

<Commission>0.15</Commission>

<PhoneNo>078-22255588</PhoneNo>

<Country></Country>

</Agent>

</Agent>

<Agent AgentCode="A007 ">

<Agent>

<AgentName>Ramasundar</AgentName>

</Agent>  
6)Опция explicit

SELECT

1 AS Tag,

NULL AS Parent,

AGENT\_CODE AS [Agent!1!AgentCode],

AGENT\_NAME AS [Agent!1!AgentName!Element]

FROM AGENTS

FOR XML EXPLICIT  
<Agent AgentCode="A001 ">

<AgentName>Subbarao</AgentName>

</Agent>

<Agent AgentCode="A002 ">

<AgentName>Mukesh</AgentName>

</Agent>

<Agent AgentCode="A003 ">

<AgentName>Alex </AgentName>

</Agent>

<Agent AgentCode="A004 ">

<AgentName>Ivan</AgentName>

</Agent>

<Agent AgentCode="A005 ">

<AgentName>Anderson</AgentName>

</Agent>

<Agent AgentCode="A006 ">

<AgentName>McDen</AgentName>

</Agent>

<Agent AgentCode="A007 ">

<AgentName>Ramasundar</AgentName>

</Agent>

<Agent AgentCode="A008 ">

<AgentName>Alford</AgentName>

</Agent>

<Agent AgentCode="A009 ">

<AgentName>Benjamin</AgentName>

</Agent>

<Agent AgentCode="A010 ">

<AgentName>Santakumar</AgentName>

</Agent>

<Agent AgentCode="A011 ">

<AgentName>Ravi Kumar</AgentName>

</Agent>

<Agent AgentCode="A012 ">

<AgentName>Lucida</AgentName>

</Agent>

**Выводы**

В ходе выполнения лабораторной работы были получены практические навыки по конвертации данных из реляционной таблицы в xml формат, который активно используется для передачи данных, например, по сети. Для этого использована операция “for xml”, с различными опциями. Разные опции выводили данные по-разному или позволяли настроить их отображение вручную. В качестве базы данных была использована БД «Интернет магазин». Материалы по этой теме с примерами содержались в файле «Теоретические материалы - Работа с XML»

**Лабораторная работа №2**

**Задание**

1. Изучить материалы урока 2 из файла "Теоретические материалы - Работа с XML".
2. Ответить на контрольные вопросы 7-11.
3. Выполнить все примеры, приведенные в уроке 2 файла "Теоретические материалы - Работа с XML".
4. Добавить столбец XML в одну из таблиц собственной БД. Заполнить его данными.
5. Преобразовать данные добавленного столбца в реляционный формат.

**Ход работы**

Сперва добавляю столбец XML в таблицу AGENTS собственной БД:

USE LAB1

ALTER TABLE AGENTS ADD LMX xml

Затем заполняю столбец XML данными:

UPDATE AGENTS

SET LMX = (

SELECT

1 AS Tag,

NULL AS Parent,

AGENT\_NAME AS [Agent!1!Name],

WORKING\_AREA AS [Agent!1!WorkingArea],

COMMISSION AS [Agent!1!Commission],

PHONE\_NO AS [Agent!1!PhoneNo],

COUNTRY AS [Agent!1!Country]

FROM AGENTS

FOR XML EXPLICIT, Root('Agents')

)

Получаю XML документ:

DECLARE @XML AS XML, @hDoc AS INT

SELECT @XML = LMX FROM AGENTS

Генерирую внутреннее представление дерева:

EXEC sp\_xml\_preparedocument @hDoc OUTPUT, @XML, ''

Формирую и обрабатываю строки:  
SELECT \* FROM OPENXML(@hDoc, '/Agents/Agent', 1)

WITH

(

Name varchar(40),

WorkingArea varchar(35),

Commission decimal(10,2),

PhoneNo varchar(15),

Country varchar(25)

)

В конце удаляю внутренне дерево:

EXEC sp\_xml\_removedocument @hDoc

**Листинг**

USE LAB1

--Добавление столбца XML в таблицу AGENTS:

ALTER TABLE AGENTS ADD LMX xml

SELECT \* FROM AGENTS

--Заполнение столбца XML данными

UPDATE AGENTS

SET LMX = (

SELECT

1 AS Tag,

NULL AS Parent,

AGENT\_NAME AS [Agent!1!Name],

WORKING\_AREA AS [Agent!1!WorkingArea],

COMMISSION AS [Agent!1!Commission],

PHONE\_NO AS [Agent!1!PhoneNo],

COUNTRY AS [Agent!1!Country]

FROM AGENTS

FOR XML EXPLICIT, Root('Agents')

)

--Получение XML-документа

DECLARE @XML AS XML, @hDoc AS INT

SELECT @XML = LMX FROM AGENTS

--Генерация внутреннего представления дерева

EXEC sp\_xml\_preparedocument @hDoc OUTPUT, @XML, ''

--Формирование и обработка строк

SELECT \* FROM OPENXML(@hDoc, '/Agents/Agent', 1)

WITH

(

Name varchar(40),

WorkingArea varchar(35),

Commission decimal(10,2),

PhoneNo varchar(15),

Country varchar(25)

)

**Лабораторная работа №3**

**«Использование типа данных xml»**

Краткая теория

* XQuery используется, чтобы формировать запросы к данным XML. Синтаксис XQuery включает и расширяет выражения XPath 2.0 и позволяет выполнить сложные запросы к данным XML. Тип данных xml в SQL Server поддерживает методы, с помощью которых данные в xml могут быть получены или обновлены посредством выражения XQuery.

Поддержка XQuery в SQL Server 2005 основана на рабочем проекте W3C языковой спецификации XQuery 1.0 (доступного на Вебсайте W3C), и поэтому могут быть незначительные несоответствия с выпущенной спецификацией.

* Метод query используется, чтобы извлечь XML-данные из типа данных xml. XML-документ, полученный методом query, определяется по выражению XQuery, который является параметром. Следующий пример показывает, как использовать метод query.

SELECT xmlCol.query('declare default element namespace

"http://schemas.adventure-works.com/InvoiceList"; <InvoiceNumbers>

{

for $i in /InvoiceList/Invoice

return <InvoiceNo>

{number($i/@InvoiceNo)}

</InvoiceNo> } </InvoiceNumbers>')

* Метод value используется, чтобы возвратить одно значение из документа XML. Чтобы использовать метод value, необходимо определить выражение XQuery, которое идентифицирует единственный узел в запрашиваемом XML-документе и тип данных Transact-SQL возвращаемого значения. Следующий пример показывает, как использовать метод value.

SELECT xmlCol.value('declare default element

"http://schemas.adventure-works.com/InvoiceList"; /InvoiceList/Invoice/@InvoiceNo)[1]', 'int')

* Метод exist используется для определения того, существует ли указанный узел в XML документе. Метод exist возвращает 1, если существует один или более экземпляров указанного узла в документе, и возвращает 0, если узел не существует. Следующий пример показывает, как использовать метод exist.

SELECT xmlCol.exist('declare default element namespace

"http://schemas.adventure-works.com/InvoiceList";

/InvoiceList/Invoice[@InvoiceNo=1000]')

* Метод modify используется, чтобы обновить данные XML в типе данных xml. Mетод modify использует три расширения для языковой спецификации XQuery: insert, replace, и delete. Эти расширения упоминаются как XML DML.
* Тип данных xml поддерживает метод nodes, который нужно использовать, чтобы сгенерировать реляционное представление данных XML. Метод nodes возвращает набор строк, в котором каждый узел, идентифицированный выражением XQuery, возвращается как контекстный узел, из которого могут извлечь данные последующие запросы.

Задание

1. Изучить материалы урока 3 из файла "Теоретические материалы - Работа с XML".
2. Ответить на контрольные вопросы 12-15.
3. Выполнить примеры, приведенные в уроке 3 файла “Теоретически материалы – Работа с XML”. Альтернативно для тех, кто работает с последними версиями MS SQL Server, можно выполнить примеры для каждого метода типа данных XML, приведенные на сайте <https://docs.microsoft.com/ru-ru/sql/t-sql/xml/xml-data-type-methods?view=sql-server-ver15>
4. Для столбца XML из таблицы собственной БД сформулировать и выполнить запросы с использованием методов query, value и exist.

Для этого столбца сформулировать и выполнить запросы с использованием методов modify и nodes.

Ход работы

1. Выполнение запроса с использованием query:

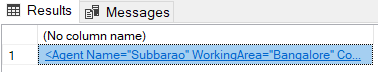
use lab1

DECLARE @XML AS XML

SELECT @XML = LMX FROM AGENTS

select @XML.query('/Agents/Agent')

РЕЗУЛЬТАТ



<Agent Name="Subbarao" WorkingArea="Bangalore" Commission="0.14" PhoneNo="077-12346674" Country="" />

<Agent Name="Mukesh" WorkingArea="Mumbai" Commission="0.11" PhoneNo="029-12358964" Country="" />

<Agent Name="Alex " WorkingArea="London" Commission="0.13" PhoneNo="075-12458969" Country="" />

<Agent Name="Ivan" WorkingArea="Torento" Commission="0.15" PhoneNo="008-22544166" Country="" />

<Agent Name="Anderson" WorkingArea="Brisban" Commission="0.13" PhoneNo="045-21447739" Country="" />

<Agent Name="McDen" WorkingArea="London" Commission="0.15" PhoneNo="078-22255588" Country="" />

<Agent Name="Ramasundar" WorkingArea="Bangalore" Commission="0.15" PhoneNo="077-25814763" Country="" />

<Agent Name="Alford" WorkingArea="New York" Commission="0.12" PhoneNo="044-25874365" Country="" />

<Agent Name="Benjamin" WorkingArea="Hampshair" Commission="0.11" PhoneNo="008-22536178" Country="" />

<Agent Name="Santakumar" WorkingArea="Chennai" Commission="0.14" PhoneNo="007-22388644" Country="" />

<Agent Name="Ravi Kumar" WorkingArea="Bangalore" Commission="0.15" PhoneNo="077-45625874" Country="" />

<Agent Name="Lucida" WorkingArea="San Jose" Commission="0.12" PhoneNo="044-52981425" Country="" />

1. Выполнение запроса с использованием value:

DECLARE @XML AS XML

SELECT @XML = LMX FROM AGENTS

DECLARE @Name AS VARCHAR(50)

SET @Name = @XML.value('(/Agents/Agent/@Name)[1]', 'varchar(50)')

SELECT @Name

РЕЗУЛЬТАТ



1. Выполнение запроса с использованием exist:

DECLARE @XML AS XML

SELECT @XML = LMX FROM AGENTS

DECLARE @Name AS VARCHAR(50)

SET @Name = @XML.exist('/Agents/Agent[@Name="Ivan"]')

SELECT @Name

РЕЗУЛЬТАТ



1. Выполнение запроса с использованием modify:

declare @myDoc XML

set @myDoc = '<Agents>

<Agent Name="Subbarao" WorkingArea="Bangalore" Commission="0.14" PhoneNo="077-12346674" Country="" />

<Agent Name="Mukesh" WorkingArea="Mumbai" Commission="0.11" PhoneNo="029-12358964" Country="" />

</Agents>'

set @myDoc.modify('insert <Agent Name="Petr" WorkingArea="Kazan" Commission="0.1" PhoneNo="555" Country="Russia"/> as first into (/Agents)[1]')

select @myDoc

РЕЗУЛЬТАТ



<Agents>

<Agent Name="Petr" WorkingArea="Kazan" Commission="0.1" PhoneNo="555" Country="Russia" />

<Agent Name="Subbarao" WorkingArea="Bangalore" Commission="0.14" PhoneNo="077-12346674" Country="" />

<Agent Name="Mukesh" WorkingArea="Mumbai" Commission="0.11" PhoneNo="029-12358964" Country="" />

</Agents>

1. Выполнение запроса с использованием nodes:

declare @XML AS XML

SELECT @XML = LMX FROM AGENTS

SELECT

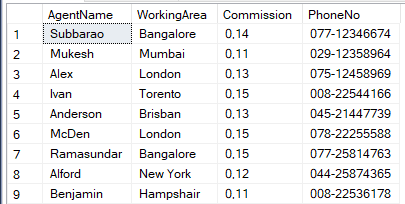
Agent.value('@Name', 'VARCHAR(50)') AS AgentName,

Agent.value('@WorkingArea', 'VARCHAR(50)') AS WorkingArea,

Agent.value('@Commission', 'FLOAT') AS Commission,

Agent.value('@PhoneNo', 'VARCHAR(50)') AS PhoneNo

РЕЗУЛЬТАТ



Листинг

use lab1

DECLARE @XML AS XML

SELECT @XML = LMX FROM AGENTS

------------- query

select @XML.query('/Agents/Agent')

------------- value

DECLARE @XML AS XML

SELECT @XML = LMX FROM AGENTS

DECLARE @Name AS VARCHAR(50)

SET @Name = @XML.value('(/Agents/Agent/@Name)[1]', 'varchar(50)')

SELECT @Name

---------------- exist

DECLARE @XML AS XML

SELECT @XML = LMX FROM AGENTS

DECLARE @Name AS VARCHAR(50)

SET @Name = @XML.exist('/Agents/Agent[@Name="Ivan"]')

SELECT @Name

------------------- modify

declare @myDoc XML

set @myDoc = '<Agents>

<Agent Name="Subbarao" WorkingArea="Bangalore" Commission="0.14" PhoneNo="077-12346674" Country="" />

<Agent Name="Mukesh" WorkingArea="Mumbai" Commission="0.11" PhoneNo="029-12358964" Country="" />

</Agents>'

set @myDoc.modify('insert <Agent Name="Petr" WorkingArea="Kazan" Commission="0.1" PhoneNo="555" Country="Russia"/> as first into (/Agents)[1]')

select @myDoc

-------------- nodes

declare @XML AS XML

SELECT @XML = LMX FROM AGENTS

SELECT

Agent.value('@Name', 'VARCHAR(50)') AS AgentName,

Agent.value('@WorkingArea', 'VARCHAR(50)') AS WorkingArea,

Agent.value('@Commission', 'FLOAT') AS Commission,

Agent.value('@PhoneNo', 'VARCHAR(50)') AS PhoneNo

FROM @XML.nodes('/Agents/Agent') Agents(Agent)

Заключение

Мною была выполнена лабораторная работа №3 для дисциплины “Программирование на Transact SQL”, в ходе реализации которой были рассмотрены основы работы с XQuery для формирования запросов к данным XML. Были усвоены и закреплены навыки работы с основными методами: query, value, exist, modify, nodes. В ходе выполнения лабораторных работ уровень знаний для взаимодействия с XML был повышен и будет эффективно использован для решения будущих задач.

**Лабораторная работа №4**

**«Создание и настройка индексов»**

Краткая теория

Индекс - коллекция страниц, связанных с таблицей (или представлением), предназначенная для повышения скорости поиска строк таблицы или для обеспечения уникальности. Например, без индекса необходимо было бы пролистать весь учебник страница за страницей, чтобы найти информацию о нужной теме. Microsoft® SQL Server ™ 2005 использует индексы, чтобы указать местоположение строки на странице данных вместо того, чтобы просматривать все страницы данных таблицы.

Индекс содержит ключи, построенные из одного или более столбцов в таблице. Эти ключи хранятся таким образом, что SQL Server 2005 ищет строки по этому ключу быстро и эффективно.

SQL Server получает доступ к данным одним из двух способов:

• сканирование таблицы (полный просмотр таблицы). Когда SQL Server выполняет просмотр таблицы, он:

1. Начинает просмотр с начала таблицы.
2. Сканирует страницу за страницей, чтобы просмотреть все строки таблицы.
3. Извлекает строки, которые отвечают критериям запроса.

• индексное сканирование. Когда SQL Server использует индекс, он:

1. Проходит структуру дерева индекса, чтобы найти строки, соответствующие запросу.
2. Извлекает строки, которые отвечают критериям запроса.

SQL Server сначала определяет, существует ли индекс. Затем оптимизатор запросов – компонент, ответственный за построение оптимального плана выполнения – планирует запрос, т.е. определяет, что будет эффективнее, просмотреть всю таблицу или использовать индекс для того, чтобы найти нужные данные.

Задание

1. Изучить материалы всех уроков файла "Теоретические материалы - Создание и настройка индексов".
2. Ответить на контрольные вопросы.
3. Выполнить примеры, приведенные в файле "Теоретические материалы - Создание и настройка индексов".
4. Определить, какие индексы уже созданы в собственной БД. При этом следует использовать разные способы получения информации об индексах.
5. Сформулировать 5-6 сложных запросов SELECT к собственной БД, содержащих соединения таблиц, сортировку, группировку данных.
6. Просмотреть предполагаемый план выполнения каждого запроса п.5. Как посмотреть план запроса, см. в полезных ссылках выше.
7. Каждый запрос п.5. выполнить и просмотреть фактический план выполнения. Отличаются ли ли фактические планы выполнения от предполагаемых?
8. Попытаться улучшить производительность запросов п.5. с помощью создания индексов.

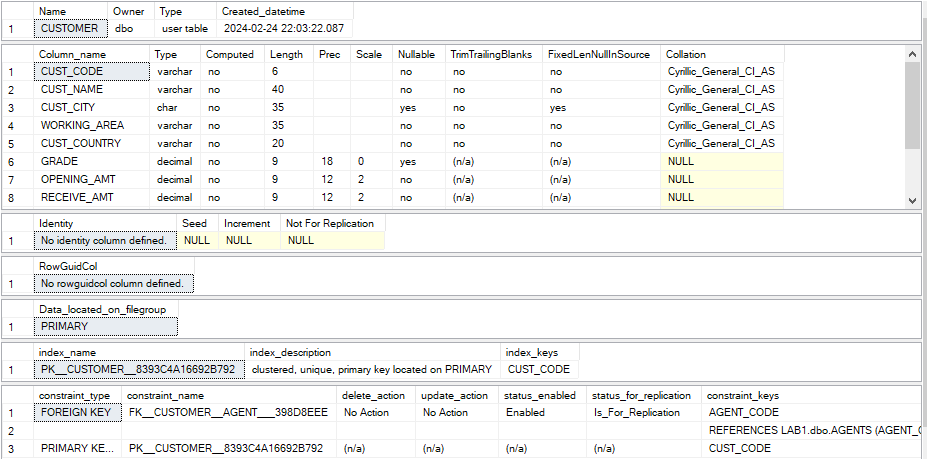
Ход работы

Сначала определю какие индексы уже созданы в собственной БД

EXEC sp\_helpindex [AGENTS]



EXEC sp\_help customer



EXEC sp\_helpindex ORDERS



Теперь сформулирую 5-6 сложных запросов SELECT к собственной БД.

1. Запрос для получения списка всех клиентов, которые разместили заказы на сумму более 5000, и агентов, которые обслуживали этих клиентов. Результат должен быть отсортирован по фамилии клиента в порядке возрастания:

SELECT C.CUST\_NAME, A.AGENT\_NAME, SUM(O.ORD\_AMOUNT) AS TOTAL\_ORD\_AMOUNT

FROM CUSTOMER C

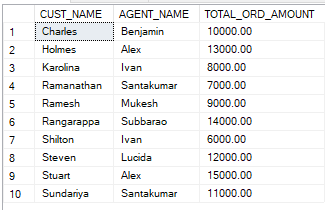
JOIN AGENTS A ON C.AGENT\_CODE = A.AGENT\_CODE

JOIN ORDERS O ON C.CUST\_CODE = O.CUST\_CODE

GROUP BY C.CUST\_NAME, A.AGENT\_NAME

HAVING SUM(O.ORD\_AMOUNT) > 5000

ORDER BY C.CUST\_NAME ASC;



1. Запрос для получения списка всех агентов, которые обслуживают клиентов из одной страны, и суммы всех заказов, размещенных этими клиентами. Результат должен быть отсортирован по стране в порядке возрастания:

SELECT A.AGENT\_NAME, C.CUST\_COUNTRY, SUM(O.ORD\_AMOUNT) AS TOTAL\_ORD\_AMOUNT

FROM AGENTS A

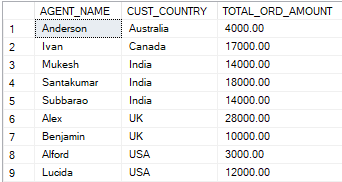
JOIN CUSTOMER C ON A.AGENT\_CODE = C.AGENT\_CODE

JOIN ORDERS O ON C.CUST\_CODE = O.CUST\_CODE

GROUP BY A.AGENT\_NAME, C.CUST\_COUNTRY

HAVING COUNT(DISTINCT C.CUST\_COUNTRY) = 1

ORDER BY C.CUST\_COUNTRY ASC;



1. Запрос для получения списка всех агентов, которые обслуживают клиентов, разместивших заказы на сумму более 10000, и имеющих задолженность более 5000. Результат должен быть отсортирован по фамилии агента в порядке возрастания:

SELECT A.AGENT\_NAME, C.CUST\_NAME, SUM(O.ORD\_AMOUNT) AS TOTAL\_ORD\_AMOUNT, C.OUTSTANDING\_AMT

FROM AGENTS A

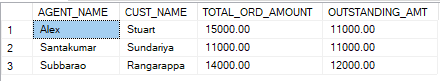
JOIN CUSTOMER C ON A.AGENT\_CODE = C.AGENT\_CODE

JOIN ORDERS O ON C.CUST\_CODE = O.CUST\_CODE

GROUP BY A.AGENT\_NAME, C.CUST\_NAME, C.OUTSTANDING\_AMT

HAVING SUM(O.ORD\_AMOUNT) > 10000 AND C.OUTSTANDING\_AMT > 5000

ORDER BY A.AGENT\_NAME ASC;



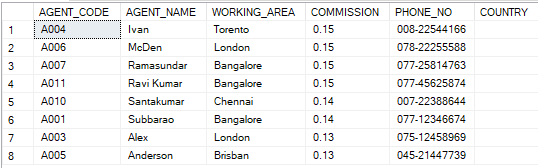
1. Запрос для получения списка всех агентов, которые имеют комиссию более 12%, отсортированных по комиссии в порядке убывания:

SELECT AGENT\_CODE, AGENT\_NAME, WORKING\_AREA, COMMISSION, PHONE\_NO, COUNTRY

FROM AGENTS

WHERE COMMISSION > 0.12

ORDER BY COMMISSION DESC;



1. Запрос для получения суммы всех заказов, размещенных клиентами, которые работают с агентами, работающими в Лондоне:

SELECT SUM(ORD\_AMOUNT) AS TOTAL\_ORD\_AMOUNT

FROM ORDERS

JOIN CUSTOMER ON ORDERS.CUST\_CODE = CUSTOMER.CUST\_CODE

JOIN AGENTS ON CUSTOMER.AGENT\_CODE = AGENTS.AGENT\_CODE

WHERE AGENTS.WORKING\_AREA = 'London';



1. Запрос для получения списка всех заказов, размещенных клиентами, которые работают с агентом с кодом 'A003', отсортированных по номеру заказа в порядке возрастания:

SELECT ORD\_NUM, ORD\_AMOUNT, ADVANCE\_AMOUNT, ORD\_DATE, ORD\_DESCRIPTION

FROM ORDERS

JOIN CUSTOMER ON ORDERS.CUST\_CODE = CUSTOMER.CUST\_CODE

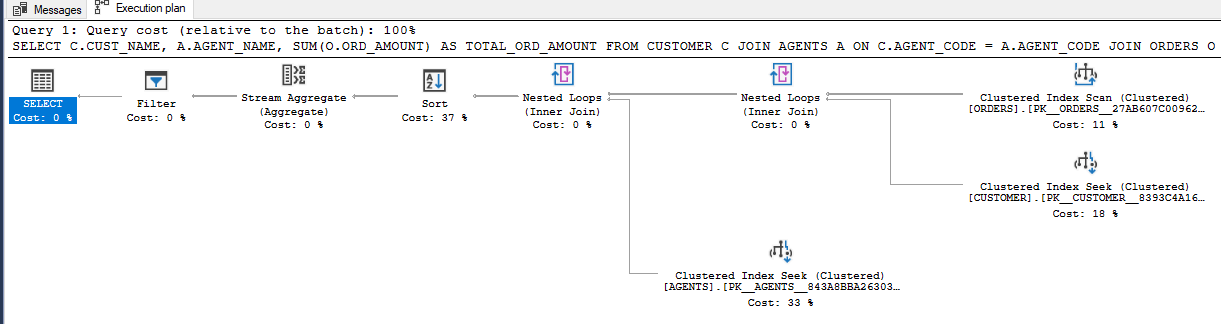
WHERE CUSTOMER.AGENT\_CODE = 'A003'

ORDER BY ORD\_NUM ASC;

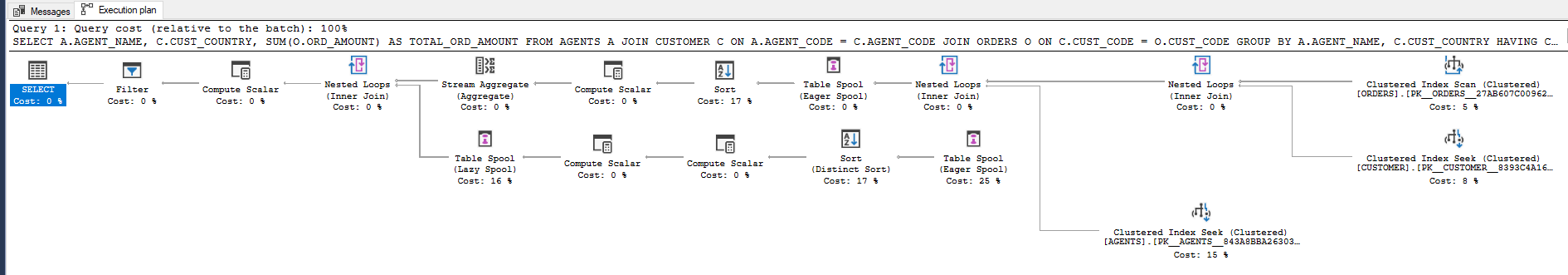


Рассмотрим предполагаемый план выполнения каждого запроса:

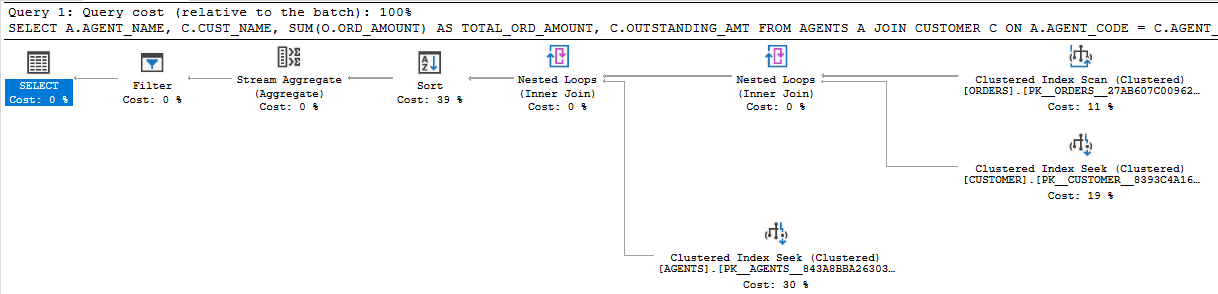
Запрос 1:



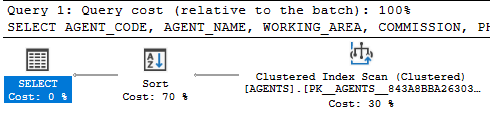
Запрос 2:

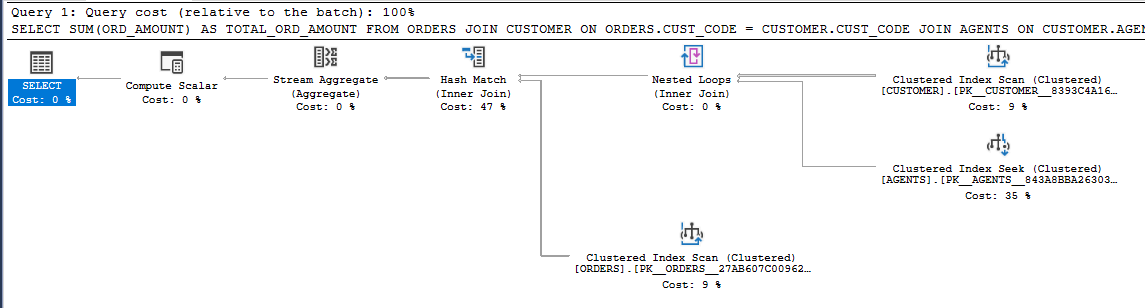


Запрос 3:

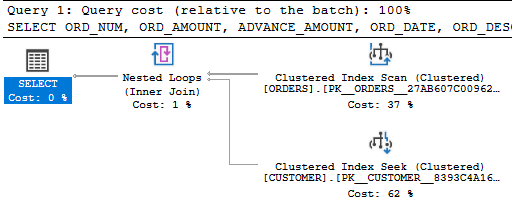


Запрос 4:

  
Запрос 5:

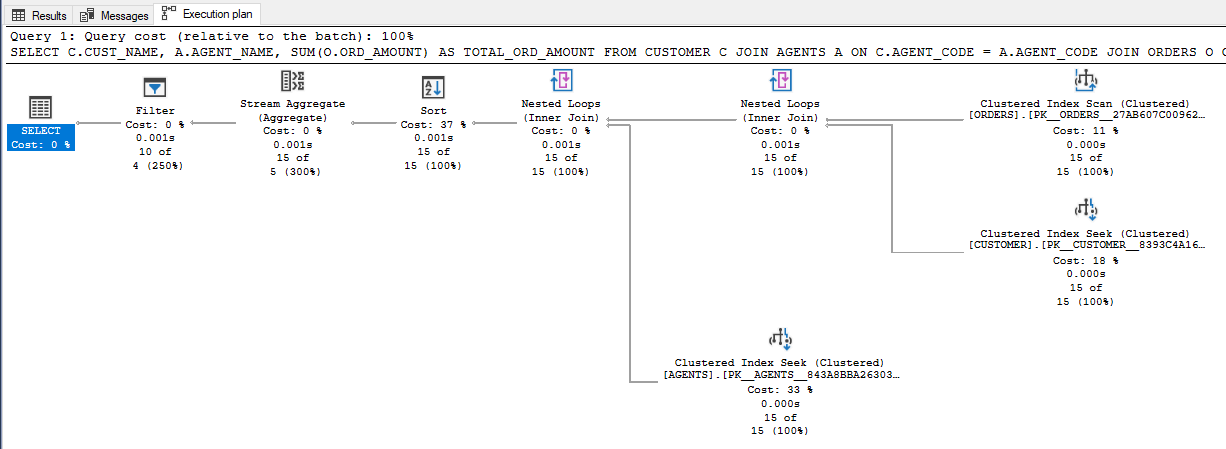


Запрос 6:

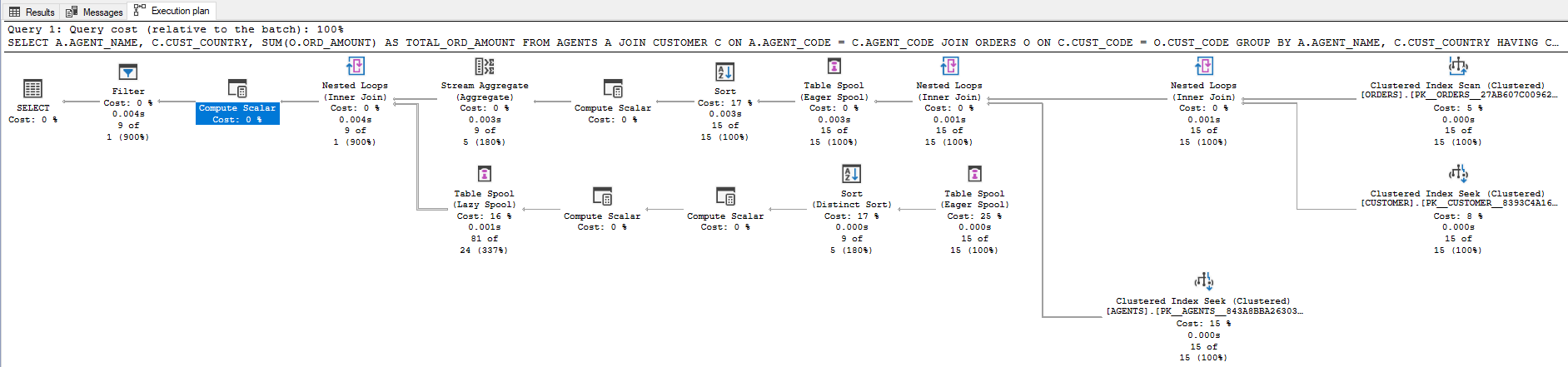


Фактический план выполнения каждого запроса:

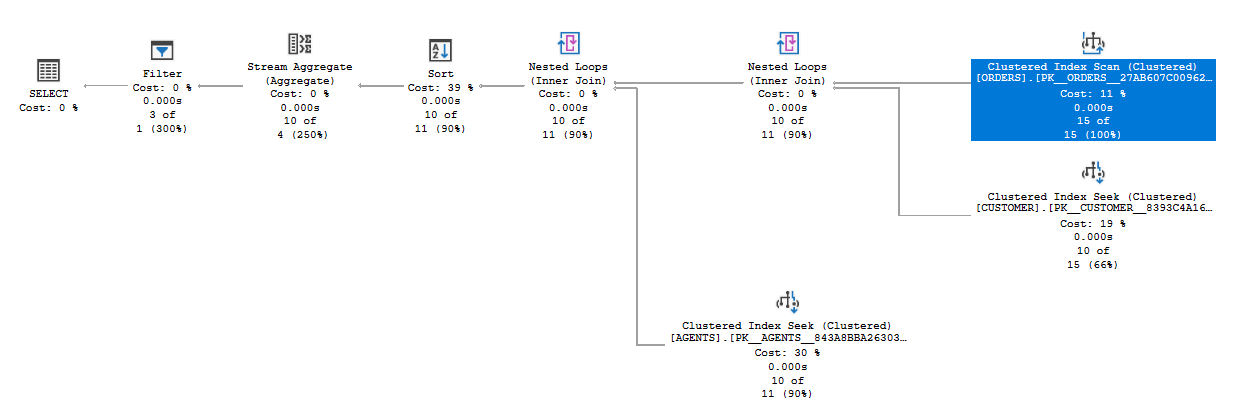
Запрос 1:



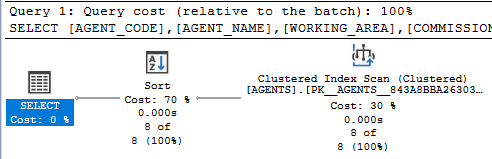
Запрос 2:



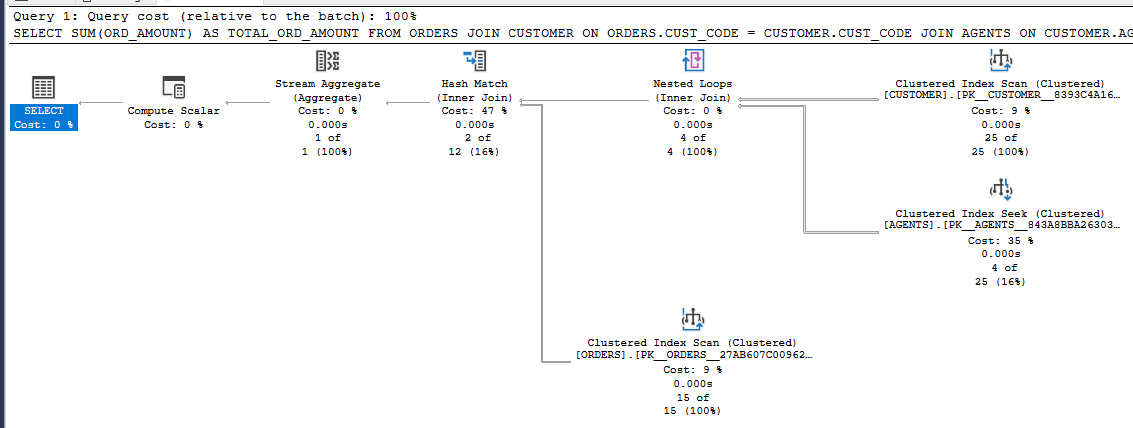
Запрос 3:



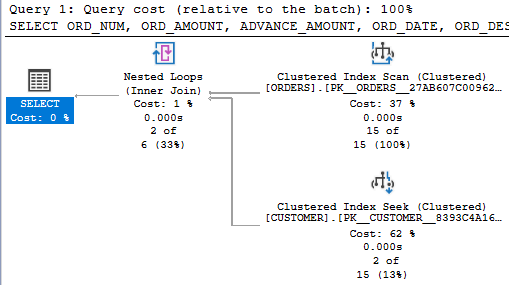
Запрос 4:



Запрос 5:

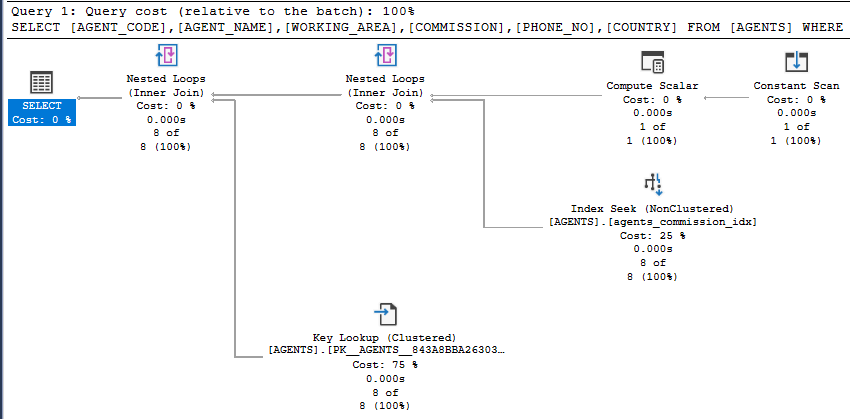


Запрос 6:



Попытаюсь улучшить производительность запроса 4, создав индекс индекс для столбца AGENTS.COMMISSION:

create index agents\_commission\_idx ON agents (commission)



Получилось немного понизить запроса.

Вывод

В ходе выполнения лабораторной работы №4 я познакомился с тремя фундаментальными вариантами индексации и определил преимущества и недостатки каждого типа. Я смог определить какие индексы уже созданы в моей БД (некластеризованные), а также научился определять фактический и предполагаемый план запроса