Intelligent Data Management with SQL Server

Session: 8

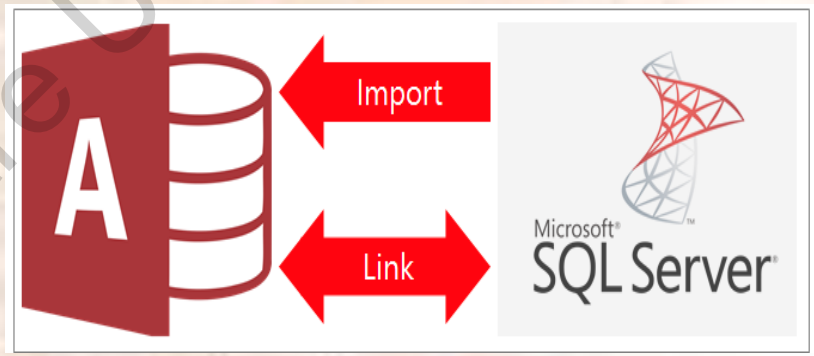
Accessing Data

Objectives

- Describe SELECT statement, its syntax, and use
- Explain various clauses used with SELECT
- State the use of ORDER BY clause
- Describe working with typed and untyped XML
- Explain the procedure to create, use, and view XML schemas

Introduction

- The SELECT statement is a core command used to access data in SQL Server 2019.
- XML allows developers to develop their own set of tags and makes it possible for other programs to understand these tags.
- XML is the preferred means for developers to store, format, and manage data on the Web.



SELECT Statement

Data in a table can be viewed using SELECT statement

This statement:

- Displays required information in a table
- Retrieves rows and columns from one or more tables
- Defines columns to be used for a query
- Consists of a series of expressions separated by commas
- Retrieves rows from database and enables selection of one or many rows or columns

Syntax:

```
SELECT <column_name1>...<column_nameN> FROM <table_name>
```

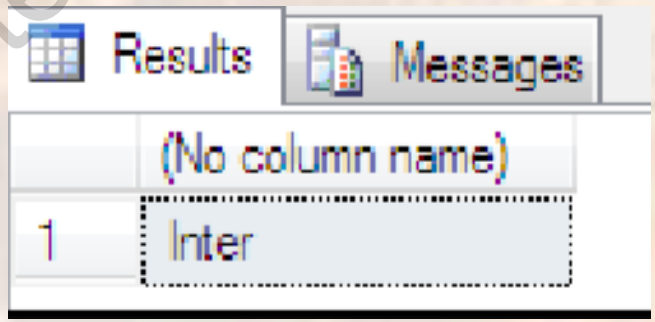
SELECT Without FROM

- Many SQL versions use FROM in their query, but in all the versions from SQL Server 2005, including SQL Server 2019, one can use SELECT statements without using the FROM clause.

- Following code will display only first five characters from extreme left of the word 'International'.

```
SELECT LEFT('International',5)
```

The output is as follows:



First Five Characters from the Extreme Left of the Word

Displaying All Columns

- The asterisk (*) is used in the SELECT statement to retrieve all the columns from the table.
- It is used as a shorthand to list all the column names in the table named in the FROM clause.

Following is the syntax for selecting all columns

```
SELECT * FROM <table_name>
```

```
USE AdventureWorks2019
SELECT * FROM HumanResources.Employee
GO
```

Results | Messages

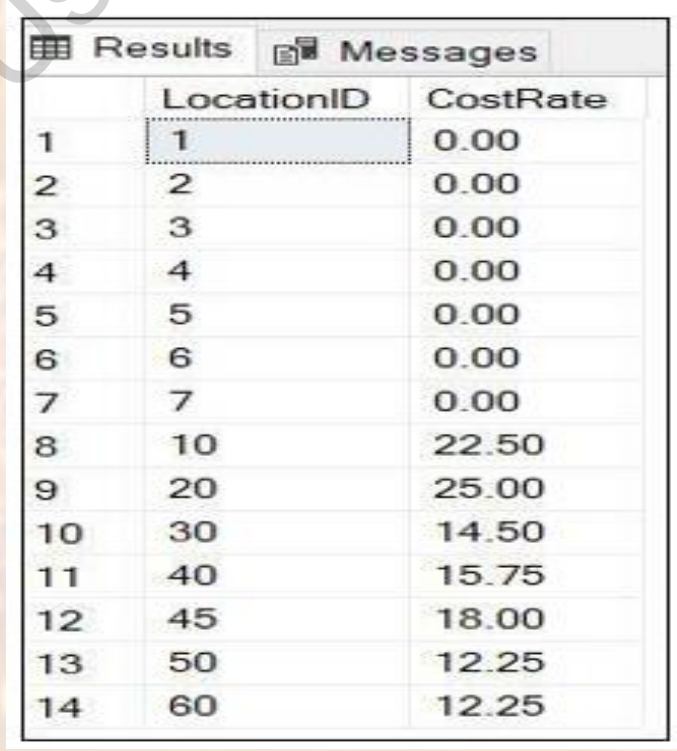
	BusinessEntityID	NationalIDNumber	LoginID	OrganizationNode	OrganizationLevel	JobTitle
1	1	295847284	adventure-works\ken0	NULL	NULL	Chief Executive Officer
2	2	245797967	adventure-works\terri0	0x58	1	Vice President of Engineering
3	3	509647174	adventure-works\roberto0	0x5AC0	2	Engineering Manager
4	4	112457891	adventure-works\rob0	0x5AD6	3	Senior Tool Designer
5	5	695256908	adventure-works\gail0	0x5ADA	3	Design Engineer
6	6	998320692	adventure-works\josef0	0x5ADE	3	Design Engineer
7	7	134969118	adventure-works\dylan0	0x5AE1	3	Research and Development Manager
8	8	811994146	adventure-works\diane1	0x5AE158	4	Research and Development Engineer
9	9	658797903	adventure-works\gigi0	0x5AE168	4	Research and Development Engineer
10	10	879342154	adventure-works\michael6	0x5AE178	4	Research and Development Manager

Displaying All Columns

Displaying Selected Columns

- The SELECT statement displays or returns certain relevant columns that are chosen by the user or mentioned in the statement.
- To display specific columns, knowledge of the relevant column names in the table is required.

```
USE AdventureWorks2019
SELECT LocationID, CostRate FROM
Production.Location
GO
```



Results | Messages

	LocationID	CostRate
1	1	0.00
2	2	0.00
3	3	0.00
4	4	0.00
5	5	0.00
6	6	0.00
7	7	0.00
8	10	22.50
9	20	25.00
10	30	14.50
11	40	15.75
12	45	18.00
13	50	12.25
14	60	12.25

LocationID and CostRate Columns

Different Expressions with SELECT

- SELECT statement allows users to specify different expressions in order to view the resultset in an ordered manner.
- These expressions assign different names to columns in the resultset, compute values, and eliminate duplicate values

Using Constants in Result Sets

- Used when character columns are joined
- Help in proper formatting or readability
- Not specified as a separate column in the resultset
- More efficient for an application to build the constant values into the results

```
USE AdventureWorks2019
SELECT Name +':'+ CountryRegionCode +'->'+ Group FROM Sales.SalesTerritory
GO
```

Results | Messages

	(No column name)
1	Northwest:US->North America
2	Northeast:US->North America
3	Central:US->North America
4	Southwest:US->North America
5	Southeast:US->North America
6	Canada:CA->North America
7	France:FR->Europe
8	Germany:DE->Europe
9	Australia:AU->Pacific
10	United Kingdom:GB->Europe

Country Name, Country Region Code, and Corresponding Group

Renaming ResultSet Column Names 1-2

- Columns displayed in resultsets of queries have corresponding headings specified in the table.

These headings can be:
- Changed
- Renamed
- Can be assigned a new name by using AS clause

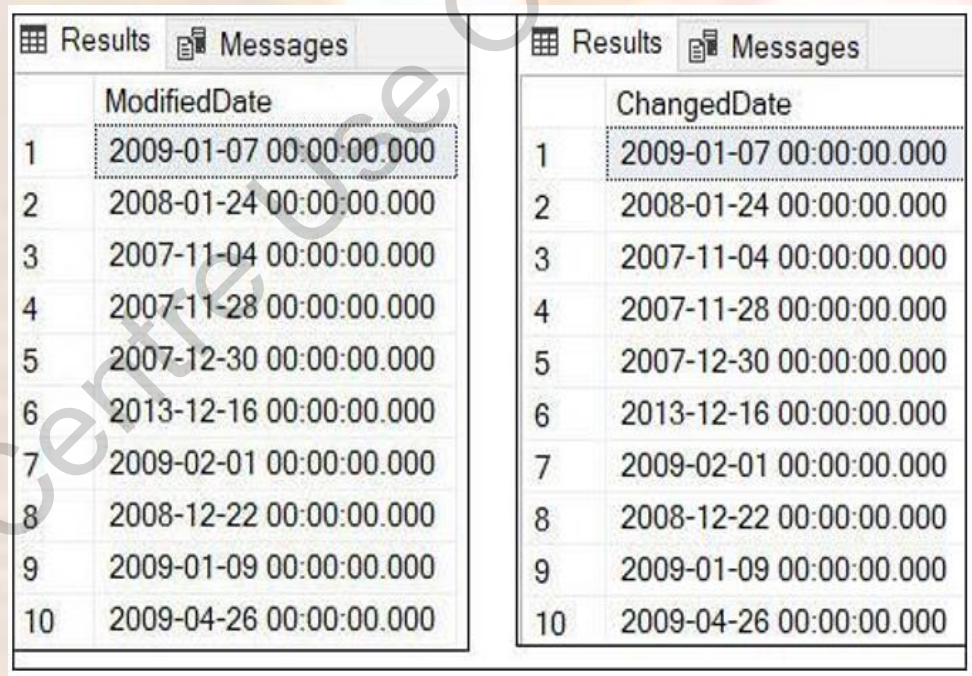
- By customizing the headings, they become more understandable and meaningful.

Renaming ResultSet Column Names 2-2

Results | Messages

	NameRegionGroup
1	Northwest:US->North America
2	Northeast:US->North America
3	Central:US->North America
4	Southwest:US->North America
5	Southeast:US->North America
6	Canada:CA->North America
7	France:FR->Europe
8	Germany:DE->Europe
9	Australia:AU->Pacific
10	United Kingdom:GB->Europe

Column Heading Modified to NameRegionGroup



Results | Messages

	ModifiedDate
1	2009-01-07 00:00:00.000
2	2008-01-24 00:00:00.000
3	2007-11-04 00:00:00.000
4	2007-11-28 00:00:00.000
5	2007-12-30 00:00:00.000
6	2013-12-16 00:00:00.000
7	2009-02-01 00:00:00.000
8	2008-12-22 00:00:00.000
9	2009-01-09 00:00:00.000
10	2009-04-26 00:00:00.000

Results | Messages

	ChangedDate
1	2009-01-07 00:00:00.000
2	2008-01-24 00:00:00.000
3	2007-11-04 00:00:00.000
4	2007-11-28 00:00:00.000
5	2007-12-30 00:00:00.000
6	2013-12-16 00:00:00.000
7	2009-02-01 00:00:00.000
8	2008-12-22 00:00:00.000
9	2009-01-09 00:00:00.000
10	2009-04-26 00:00:00.000

Column Heading Modified to ChangedDate

Computing Values in ResultSet

- A SELECT statement can contain mathematical expressions by applying operators to one or more columns.
- It allows a resultset to contain values that do not exist in the base table, but are calculated from the values stored in the base table.

Results | Messages

	ProductID	StandardCost	Discount
1	707	12.0278	1.804170
2	707	13.8782	2.081730
3	707	13.0863	1.962945
4	708	12.0278	1.804170
5	708	13.8782	2.081730
6	708	13.0863	1.962945

Calculated Discount Amount

Using DISTINCT

- The keyword DISTINCT prevents the retrieval of duplicate records. It eliminates rows that are repeating from the resultset of a SELECT statement.

For example,

- If the StandardCost column is selected without using the DISTINCT keyword, it will display all the standard costs present in the table.
- On using the DISTINCT keyword in the query, SQL Server will display every record of StandardCost only once.

Using TOP and PERCENT

- The TOP keyword will display only first few set of rows as a resultset

- The TOP expression can also be used with other statements such as INSERT, UPDATE, and DELETE.

Syntax:

```
SELECT [ALL|DISTINCT] [TOP expression [PERCENT] [WITH TIES]]
```

where,

expression: is the number or the percentage of rows to be returned as the result.

PERCENT: returns the number of rows limited by percentage.

WITH TIES: is the additional number of rows that is to be displayed.

SELECT with INTO

- The INTO clause creates a new table and inserts rows and columns listed in the SELECT statement into it.

INTO clause also inserts existing rows into the new table.

Results | Messages

	ProductModelID	Name
1	122	All-Purpose Bike Stand
2	119	Bike Wash
3	115	Cable Lock
4	98	Chain
5	1	Classic Vest
6	2	Cycling Cap
7	121	Fender Set - Mountain
8	102	Front Brakes
9	103	Front Derailleur
10	3	Full-Finger Gloves
11	4	Half-Finger Gloves

New Table

SELECT with WHERE 1-3

- The WHERE clause with SELECT statement is used to conditionally select or limit the records retrieved by the query.
- A WHERE clause specifies a Boolean expression to test the rows returned by the query.
- The row is returned if the expression is true and is discarded if it is false.

Operator	Description
=	Equal to
< >	Not equal to
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
!	Not

Operator	Description
BETWEEN	Between a range
LIKE	Search for an ordered pattern
IN	Within a range

Operators

SELECT with WHERE 2-3

Results | Messages

	ProductID	StartDate	EndDate	StandardCost	ModifiedDate
1	707	2012-05-30 00:00:00.000	2013-05-29 00:00:00.000	13.8782	2013-05-29 00:00:00.000
2	708	2012-05-30 00:00:00.000	2013-05-29 00:00:00.000	13.8782	2013-05-29 00:00:00.000
3	711	2012-05-30 00:00:00.000	2013-05-29 00:00:00.000	13.8782	2013-05-29 00:00:00.000
4	712	2012-05-30 00:00:00.000	2013-05-29 00:00:00.000	5.2297	2013-05-29 00:00:00.000
5	713	2012-05-30 00:00:00.000	2013-05-29 00:00:00.000	29.0807	2013-05-29 00:00:00.000

SELECT with WHERE clause

Results | Messages

	DepartmentID	Name	GroupName	ModifiedDate
1	1	Engineering	Research and Development	2008-04-30 00:00:00.000
2	2	Tool Design	Research and Development	2008-04-30 00:00:00.000
3	3	Sales	Sales and Marketing	2008-04-30 00:00:00.000
4	4	Marketing	Sales and Marketing	2008-04-30 00:00:00.000
5	5	Purchasing	Inventory Management	2008-04-30 00:00:00.000
6	6	Research and Development	Research and Development	2008-04-30 00:00:00.000
7	7	Production	Manufacturing	2008-04-30 00:00:00.000
8	8	Production Control	Manufacturing	2008-04-30 00:00:00.000
9	9	Human Resources	Executive General and Administration	2008-04-30 00:00:00.000

Output of Where Clause with < Operator

SELECT with WHERE 3-3

Wildcard	Description	Example
_	It will display a single character	`SELECT * FROM Person.Contact WHERE Suffix LIKE 'Jr_'`
%	It will display a string of any length	`SELECT * FROM Person.Contact WHERE LastName LIKE 'B%'`
[ ]	It will display a single character within the range enclosed in the brackets	`SELECT * FROM Sales.CurrencyRate WHERE ToCurrencyCode LIKE 'C[AN][DY]'`
[^]	It will display any single character not within the range enclosed in the brackets	`SELECT * FROM Sales.CurrencyRate WHERE ToCurrencyCode LIKE 'A[^R][^S]'`

Wildcard Characters

GROUP BY Clause

- The GROUP BY clause partitions the resultset into one or more subsets.
- Each subset has values and expressions in common.
- If an aggregate function is used in the GROUP BY clause, the resultset produces single value per aggregate.

Results | Messages

	WorkOrderID	(No column name)
1	13	17.6000
2	14	17.6000
3	15	4.0000
4	16	4.0000
5	17	4.0000
6	18	4.0000
7	19	4.0000

Output of GROUP BY Clause

ORDER BY Clause

- It specifies the order in which the columns should be sorted in a resultset.
- It sorts query results by one or more columns.

- A sort can be in either ascending (ASC) or descending (DESC) order. By default, records are sorted in an ASC order.

Results | Messages

	TerritoryID	Name	CountryRegionCode	Group	SalesYTD	SalesLastYear
1	8	Germany	DE	Europe	3805202.3478	1307949.7917
2	10	United Kingdom	GB	Europe	5012905.3656	1635823.3967
3	9	Australia	AU	Pacific	5977814.9154	2278548.9776
4	7	France	FR	Europe	4772398.3078	2396539.7601
5	3	Central	US	North America	3072175.118	3205014.0767
6	1	Northwest	US	North America	7887186.7882	3298694.4938
7	2	Northeast	US	North America	2402176.8476	3607148.9371
8	5	Southeast	US	North America	2538667.2515	3925071.4318
9	4	Southwest	US	North America	10510853.8739	5366575.7098
10	6	Canada	CA	North America	6771829.1376	5693988.86

Output of ORDER BY Clause

Working with XML 1-2

- Extensible Markup Language (XML) allows developers to develop their own set of tags and makes it possible for other programs to understand these tags.

- XML is the preferred means for developers to store, format, and manage data on the Web

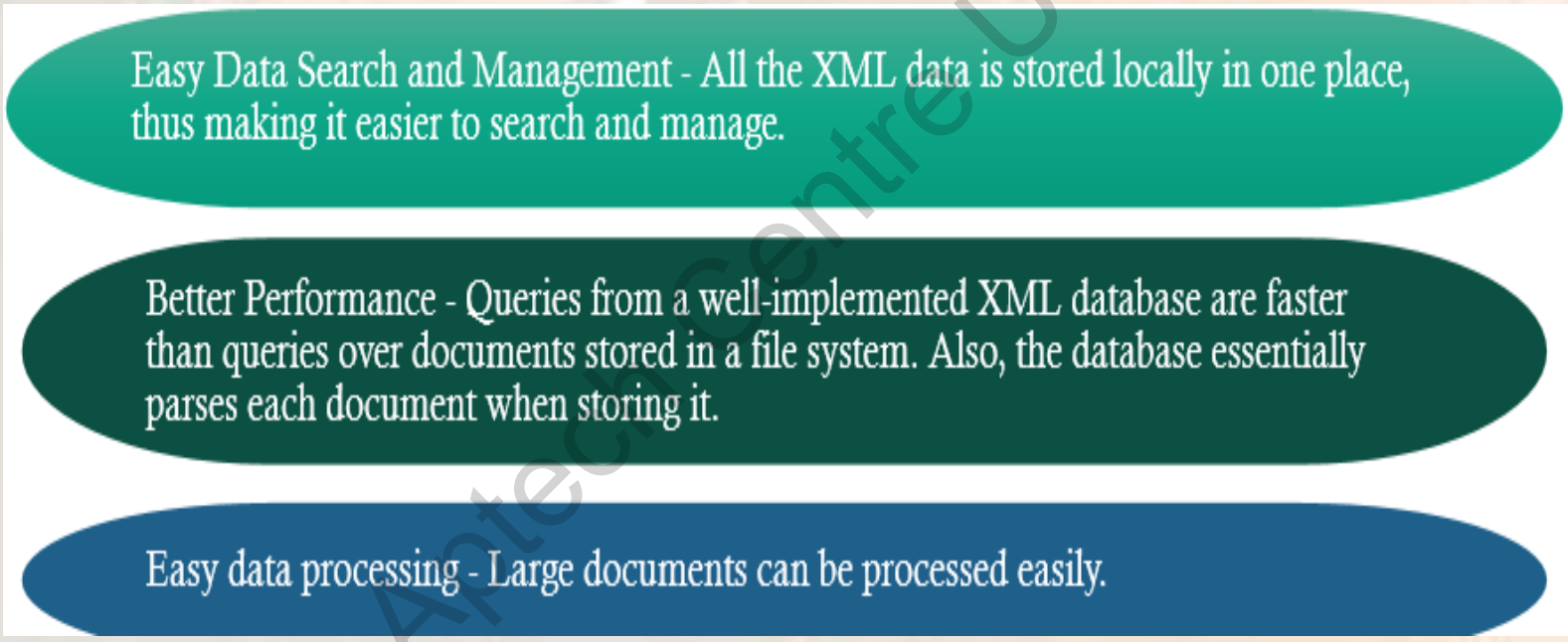
Applications of today have a mix of technologies such as:

- ASP
- Microsoft .NET technologies
- XML
- SQL Server 2019 working in tandem

In such a scenario, it is better to store XML data within SQL Server 2019.

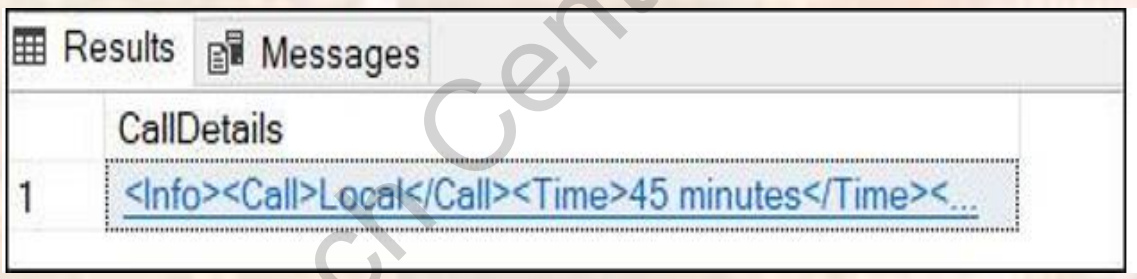
Working with XML 2-2

Native XML databases in SQL Server 2019 have a number of advantages. Some of them are listed as follows:



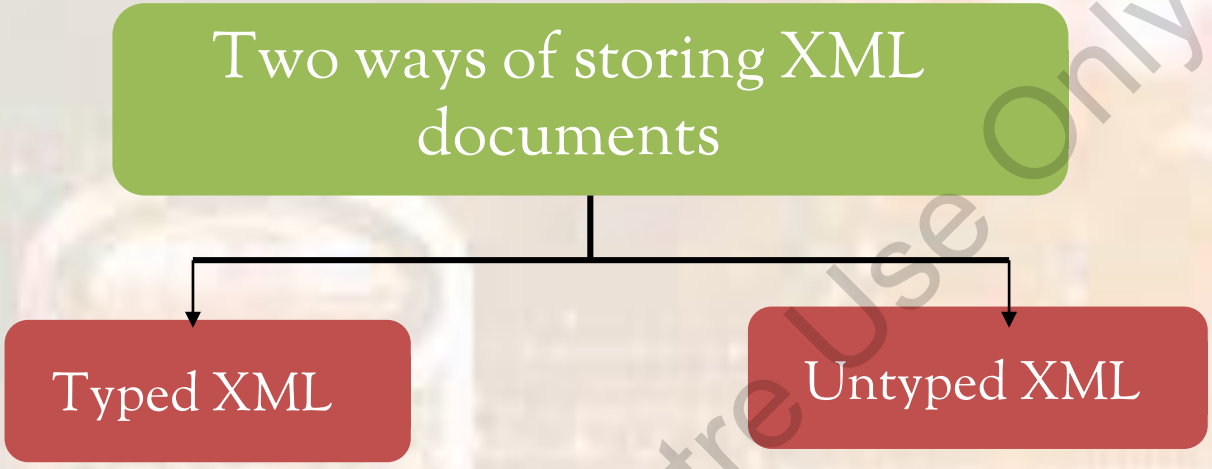
XML Data Type

- The xml data type is used to store XML documents and fragments in an SQL Server database.
- An XML fragment is an XML instance with the top-level element missing from its structure.



XML Data in Columns

Typed and Untyped XML 1-2



Typed XML instance:

- An XML instance which has a schema associated
- It describes the structure and limits the contents of XML documents

Typed and Untyped XML 2-2

Untyped XML instance:

- Data can be created and stored in either table columns or variables depending upon the need and scope of data

Displaying XML Column with SELECT

Expanded XML Data in Column

Summary

- The SELECT statement retrieves rows and columns from tables.
- SELECT statement allows users to specify different expressions in order to view the resultset in an ordered manner.
- A SELECT statement can contain mathematical expressions by applying operators to one or more columns.
- The keyword DISTINCT prevents the retrieval of duplicate records.
- XML allows developers to develop their own set of tags and makes it possible for other programs to understand these tags.
- A typed XML instance is an XML instance which has a schema associated with it.
- XML data can be queried and retrieved using XQuery language.