CHAPTER 3
DESIGN AND DEVELOPMENT OF A NOVEL FUZZY XQUERY PROCESSING

With the popularity of XML technology and Native XML database, efficiently retrieving XML data in XML database become an important research topic. As business and enterprises generate and exchange XML data more often, there is an increasing need for efficient processing of queries on XML data. Most of the existing XML (eXtensible Markup Language) query languages are based on SQL (Structured Query Language). Unlike queries on traditional relational databases whose results are always flat relations, the results for XML queries are complex. Querying XML data involves two key steps: query formulation and efficient processing of the formulated query. The current state of the art in querying XML data is represented by XPath and XQuery, both of which rely on Boolean conditions. Boolean selection is too restrictive when users do not use or even know the data structure precisely. A Novel approach for XML querying called Fuzzy XQuery based on Fuzzy set theory is described in this chapter. Fuzzy XQuery processing relies on fuzzy conditions for the definition of flexible constraints on stored data.

3.1 Novel Fuzzy XQuery Processing

XQuery combines the features from several earlier XML query languages, in particular XPath. Through XPath, Document Fragments can be extracted from an XML document. Nested loops iterate over these fragments to further extract the document fragments and construct sequences of output Document Fragments. Variable assignment supports complex computations based on content and structure of the input. But XQuery with Boolean expressions is having the following limitations.

- XQuery is forced to make arbitrary determinations about what it can do.
- XQuery does not fit the exact criteria people have in their minds.
- XQuery commands are executed on the basis of only crisp or classical logic.

In order to overcome the limitations of XQuery, Novel Fuzzy XQuery processing is proposed in the thesis. Fuzzy XQuery is based on Fuzzy set theory. The goals of Fuzzy XQuery are to store imprecise data and to process user’s imprecise queries. Fuzzy XQuery provides correct information to the users and overcomes the drawbacks of the normal XQuery Operations (Calms, M.D., et al., 2007).

Fuzzy XQueries are working on the basis of Fuzzy sets. In any business concern, a salesman who works hard and collect very high sales amount, he is considered as a very good salesman. Good salesman, average salesman and bad salesman can be represented using the Fuzzy sets by using the sales amount criteria. A reasonable expression of these concepts by triangular membership functions $S_1$, $S_2$, and $S_3$ is shown in Fig.3.1. $S_1$, $S_2$, and $S_3$ represent good salesman, average salesman and bad salesman. These Fuzzy membership functions are defined on the sales amount interval [1000,5000] as follows:

$$
S_1(x) = \begin{cases} 
1 & \text{when } x<1000 \\
(3000-x)/2000 & \text{when } 1000 \leq x \leq 3000 \\
0 & \text{when } x>300 \\
0 & \text{when } x<4000 \\
(\text{4000} - x)/1000 & \text{when } 4000 \leq x \leq 5000 \\
1 & \text{when } x>5000 \\
0 & \text{when } x<2500 \\
(\text{2500} - x)/2000 & \text{when } 2500 \leq x \leq 4500 \\
1 & \text{when } x>4500
\end{cases}
$$
Fig 3.1 Membership functions representing the concepts of a bad, average and good salesman.

Fuzzy XQuery provides a representation scheme for dealing with vague or uncertain concepts. This thesis presents novel Fuzzy XQuery processing for XML databases, where the weights of attributes of the user’s XQueries can be represented by Fuzzy sets. The proposed Fuzzy XQuery processing allows the users to use linguistic terms in the queries represented by Fuzzy Sets. The attributes are assigned with Fuzzy Weights (W), where $W \in [0, 1]$. Fuzzy Scaled Weight (FSW), where $FSW \in [0, 1]$ is calculated by using the Fuzzy Weights assigned for attributes in the Fuzzy XQuery. FSW is calculated by applying the Fuzzy Sets arithmetic operations. The proposed Fuzzy XQuery processing can deal with the users’ Fuzzy XQueries in a flexible and intelligent manner.

Unlike Boolean logic Fuzzy XQueries deal with data that is vague, ambiguous, incomplete and imprecise. Instead of applying crisp boundaries to delineate the search space, the space can be represented linguistically using the concept of Fuzzy logic (Shyi-Ming Chen and Yu-Chuan Chen 2003).
In a relational fuzzy database system, the users can use linguistic terms to describe the weights of query items. For example, consider the following Fuzzy SQL statements of the user’s query shown as follows:

**SELECT rollno, name from students where height='Very Tall' and Weight='Heavy'**

In the above Fuzzy SQL, Very Tall and Heavy are linguistic terms represented by Triangular Fuzzy sets.

Fuzzy logic provides a flexible and fluid method of defining semantic concepts within the XML database and provides the basis for a much richer and much more powerful method of looking through a XML database. In a Fuzzy XQuery, the selected records are ranked according to their compatibility with the semantics – the intent - of the query. This provides a measure of how well a record fits in with the complete set of XML records retrieved.

The Fuzzy membership value for the Fuzzy XQuery is calculated using the following formula (Equation 3.1). If the attribute value of x is known, the lower range value a, and higher range value b,

Fuzzy membership value for x can be calculated using the following formula:

\[
f_{\text{close to } a}(x) = \frac{1}{1 + \left( \frac{x - a}{b - a} \right)^2}
\]

Assume that the value of the attribute AGE of a tuple in a XML database is “25” and the query condition of the user’s query is "AGE = young", then the degree of matching of the tuple with respect to the user's query "AGE = young" is equal to

\[
f_{\text{young}}(25) = \left( 1 + \left( \frac{25 - 20}{15} \right) \right)^{-1} = 0.9 \quad (3.2)
\]

The users can use linguistic terms to describe the weights of query items in XML Databases. These linguistic terms are represented by Fuzzy Triangular sets. The linguistic terms based Fuzzy weights and the corresponding Triangular Fuzzy sets are shown in the following Table 3.1.
In Fuzzy XQuery, if the weight $\gamma$ of an attribute $A$ is a crisp value represented as "WEIGHT A = $\gamma$", where $\gamma$ is a real value between 0 and 1, then the crisp value $\gamma$ can be extended into the Triangular Fuzzy set representation ($\gamma$, $\gamma$, $\gamma$). If the weight of an attribute is 0.6, then the value 0.6 can be extended into the Triangular Fuzzy Set representation (0.6, 0.6, 0.6).

Let $W_1$, $W_2$ be two Triangular Fuzzy sets representing the Fuzzy weights of the attributes $A_1$ and $A_2$, respectively, where

$W_1 = (a_1, b_1, c_1)$

$W_2 = (a_2, b_2, c_2)$

and $\overline{W_1}$ and $\overline{W_2}$ are “Fuzzy Scaled Weights (FSW)” of $W_1$ and $W_2$. Then by using the Fuzzy set arithmetic operations, Fuzzy Scaled Weights ($\overline{W_1}$ and $\overline{W_2}$) are calculated using the equations (3.3) and (3.4).

$\overline{W_1} = W_1 \odot (W_1 \oplus W_2)$

$\overline{W_1} = (a_1, b_1, c_1) \odot (a_1 + a_2, b_1 + b_2, c_1 + c_2)$

$\overline{W_1} = \left( \frac{a_1}{c_1 + c_2}, \frac{b_1}{b_1 + b_2}, \frac{c_1}{a_1 + a_2} \right)$  \hspace{1cm} (3.3)$

$\overline{W_2} = W_2 \odot (W_1 \oplus W_2)$

$\overline{W_2} = (a_2, b_2, c_2) \odot (a_1 + a_2, b_1 + b_2, c_1 + c_2)$

<table>
<thead>
<tr>
<th>Linguistic Variable</th>
<th>Triangular Fuzzy Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absolutely Low (AL)</td>
<td>(0.0,0.0,0.0)</td>
</tr>
<tr>
<td>Extremely Low (EH)</td>
<td>(0.0,0.1,0.2)</td>
</tr>
<tr>
<td>Very Low (VL)</td>
<td>(0.1,0.2,0.3)</td>
</tr>
<tr>
<td>Low (L)</td>
<td>(0.2,0.3,0.4)</td>
</tr>
<tr>
<td>Medium (M)</td>
<td>(0.4,0.5,0.6)</td>
</tr>
<tr>
<td>High (H)</td>
<td>(0.6,0.7,0.8)</td>
</tr>
<tr>
<td>Very High (VH)</td>
<td>(0.7,0.8,0.9)</td>
</tr>
<tr>
<td>Extremely High (EH)</td>
<td>(0.8,0.9,1.0)</td>
</tr>
<tr>
<td>Absolutely High (AH)</td>
<td>(1.0,1.0,1.0)</td>
</tr>
</tbody>
</table>

Table 3.1 Linguistic Variable and its corresponding Triangular Fuzzy Set
\[ \bar{W}_2 = \left( \frac{a_2}{c_1 + c_2} : \frac{b_2}{b_1 + b_2} : \frac{c_2}{a_1 + a_2} \right) \]  

(3.4)

\( \emptyset \) and \( \oplus \) are Fuzzy sets division and Fuzzy sets addition operators respectively. Assume that \( W_1 \) is assigned with the linguistic variable based weight “high” and \( W_2 \) is assigned with the linguistic variable based weight “very high”. According to Table 3.1, the Triangular Fuzzy set for “high” is \((0.6,0.7,0.8)\) and the Triangular Fuzzy set for “very high” is \((0.7,0.8,0.9)\). Then the fuzzy weights \( \bar{W}_1 \) and \( \bar{W}_2 \) are calculated according to the equations (3.3) and (3.4).

\( W_1 = (a_1, b_1, c_1) = (0.6, 0.7, 0.8) \)

\( W_2 = (a_2, b_2, c_2) = (0.7, 0.8, 0.9) \)

\( \bar{W} = (0.6, 0.7, 0.8) \emptyset (1.3, 1.5, 1.8) \)

\( \bar{W}_1 = \left( \begin{array}{ccc} 0.6 & 0.7 & 0.8 \\ 1.7 & 1.5 & 1.3 \end{array} \right) \)  

(3.5)

\( \bar{W}_2 = (0.7, 0.8, 0.9) \emptyset (1.3, 1.5, 1.8) \)

\( \bar{W}_2 = \left( \begin{array}{ccc} 0.7 & 0.8 & 0.9 \\ 1.7 & 1.5 & 1.3 \end{array} \right) \)

\( \bar{W}_2 = (0.4, 0.5, 0.6) \)  

(3.6)

Fuzzy XQueries provide a representation scheme for dealing with vague or uncertain concepts. Unlike Boolean logic, Fuzzy XQueries deal with data that is vague, ambiguous, incomplete and imprecise. Instead of applying crisp boundaries to delineate the search space, the space can be represented linguistically using the concept of Fuzzy Logic. Fuzzy Logic provides a flexible and fluid method of defining semantic concepts within the XML database. In a Fuzzy XQuery, the selected records are ranked according to their compatibility with the semantics – the intent - of the query. Fuzzy XQuery retrieves the relevant records from the
complete set of XML records in a XML Database. Fig.3.1 shows the working of Fuzzy XQuery in a XML Database.

**Comparison of XQuery Processing with Fuzzy XQuery Processing**

The problems occurring in XQuery Commands are explained using an example. Assume that a manager of a Sales division wants to find the best performed salesman during the past 10 festival season days. He fixes the criteria for giving bonus to the salesman who achieves sales for Rs.4000 within 10 days. Then the manager applies the following XQuery command for giving bonus.

```
<output>{
    for $emp in doc("sale.xml")/salesman/emp
    let $eid := $emp/empid/text()
    let $en := $emp/ename/text()
    let $sa := $emp/sales/text()
    let $d := $emp/days/text()
    return if ($sa >=4000 and $d <=10) then
        <emp>
            <empid> {$eid} </empid>
            <ename> {$en} </ename>
            <sales> {$sa} </sales>
            <days> {$d} </days>
        </emp>
    else ()
}</output>
```

Then the output for the above command is:

```
<output>
<emp>
<empid>1003</empid>
<ename>Joshua</ename>
<sales>5000</sales>
<days>10</days>
```
<emp id="1004">
  <ename>Kevin</ename>
  <sales>4998</sales>
  <days>3</days>
</emp>

<emp id="1005">
  <ename>Davidson</ename>
  <sales>4100</sales>
  <days>9</days>
</emp>

<emp id="1006">
  <ename>Martin</ename>
  <sales>4095</sales>
  <days>2</days>
</emp>

<emp id="1007">
  <ename>Jason</ename>
  <sales>4000</sales>
  <days>10</days>
</emp>

<emp id="1009">
  <ename>George</ename>
  <sales>4100</sales>
  <days>11</days>
</emp>

<emp id="1008">
  <ename>Henry</ename>
  <sales>3999</sales>
</emp>
If the above output is analyzed, the XQuery doesn’t display the George and Henry salesman records. (George and Henry records have been strike out). Since the XQuery processing implements only Boolean conditions, it skips the George and Henry records. Although George and Henry are very close to the Manager’s criteria for getting bonus, XQuery doesn’t display their records. Moreover, if Joshua and Kevin records are compared, Kevin sales performance is better than Josha. But Kevin record order is behind Joshua. Similarly, if Davidson and Martin records are compared, Martin sales performance is better than Davidson. It is understood from the analysis that the XQuery with Boolean expression doesn’t exactly match the bonus criteria of the manager.

In order to overcome the limitations of XQuery processing with Boolean operations, this thesis proposes Fuzzy XQuery operations on the XML Database. The same problem discussed earlier is taken as an example here. Now the manager of Sales Division applies the Fuzzy XQuery to find out the best performed salesman.

<output> {
for $salesman in doc(salesman.xml)/salesman /record
let $eid := $record/empid/text()
let $en := $record/ename/text()
let $sales:= $record/sales/text()
let $days:= $record/days/text()
return if $sales=%very high and $days=%very low and $fuzzy_sales weight=W_1 and $fuzzy_days_weight=W_2 then
<record> <empid> {$eid} </empid>
<ename> {$en} </ename>
<sales>{$sa}</sales>
}
\[ \text{else ( ) } \] \text{ } \text{/output} \\

\( W_1 \) and \( W_2 \) are the Fuzzy weights of the attributes Sales and Days. Fuzzy parser using Java program is developed to convert the Linguistic terms “very high” and “very low” into Fuzzy values. When the Fuzzy parser scans the token “%”, it identifies the specific linguistic term. \( W_1 \) is assigned with “Very High” and \( W_2 \) is assigned with “Very Low” Fuzzy weight. The values of the linguistic terms are assigned from Table 3.1. But \( \overline{W}_1 \) and \( \overline{W}_2 \) are the Fuzzy Scaled Weights of \( W_1 \) and \( W_2 \) respectively, where

\[
\overline{W}_1 = W_1 \emptyset (W_1 \ominus W_2) = (a_1, b_1, c_1).
\]

(3.7)

\[
\overline{W}_2 = W_2 \emptyset (W_1 \ominus W_2) = (a_2, b_2, c_2).
\]

(3.8)

\[
F = \overline{W}_1 \otimes (x_1, y_1, z_1) \oplus \overline{W}_2 \otimes (x_2, y_2, z_2)
\]

(3.9)

Where \( (x_1, y_1, z_1) \) and \( (x_2, y_2, z_2) \) are Fuzzy membership values of linguistic terms.

Assume that the Fuzzy membership value of the Fuzzy XQuery condition "Sales = very high" is 0.9 and assume that the Fuzzy membership value of the query condition "Days = Very Low" is 0.2, then the Fuzzy Value can be extended to the Triangular Fuzzy Set \((0.9, 0.9, 0.9)\) and extend the value 0.2 to the Triangular Fuzzy set \((0.2, 0.2, 0.2)\). After performing the Fuzzy set arithmetic operations, the degree of matching of the XML record with respect to the user’s weighted Fuzzy XQuery represented by a Fuzzy number \( F \) is calculated. Fuzzy Scaled Weights (FSW) \( \overline{W}_1 \) and \( \overline{W}_2 \) are assigned from Equations (3.5) and (3.6).

\[
F = \overline{W}_1 \otimes (0.9,0.9,0.9) \oplus \overline{W}_2 \otimes (0.2,0.2,0.2)
\]

\[
= (0.3,0.4,0.6) \otimes (0.9,0.9,0.9) \oplus
\]

\[
(0.4,0.5,0.6) \otimes (0.2,0.2,0.2)
\]

\[
= (0.27,0.36,0.54) \oplus (0.08,0.1,0.12)
\]

\[
= (0.35,0.46,0.66)
\]
Then, a defuzzification method is used to defuzzify the triangular fuzzy number $F$ into a crisp value. The value is regarded as the matching degree of the XML record with respect to the user’s weighted fuzzy query. Defuzzification of a triangular fuzzy set into a crisp value is done using the formula 3.10. Assume that $F$ is a triangular fuzzy number, $G = (a, b, c)$, and $\text{Def}(F)$ denotes the defuzzified value of the triangular fuzzy number $G$, then

$$\text{Def}(F) = \frac{(a + 2b + c)}{4} \quad \text{(3.10)}$$

Output for the above Fuzzy XQuery is the following:

```xml
<output>
<emp>
  <empid>1004</empid>
  <ename>Kevin</ename>
  <sales>4998</sales>
  <days>3</days>
</emp>
<emp>
  <empid>1003</empid>
  <ename>Joshua</ename>
  <sales>5000</sales>
  <days>10</days>
</emp>
<emp>
  <empid>1006</empid>
  <ename>Martin</ename>
  <sales>4095</sales>
  <days>2</days>
</emp>
<emp>
  <empid>1008</empid>
  <ename>Henry</ename>
  <sales>3999</sales>
</emp>
```
If the above output is compared with the Normal XQuery output, the manager of the Sales division would be satisfied with the output. The normal XQuery output skips George and Henry records for getting bonus. Since salesman George and Henry records didn’t satisfy the boolean criteria of the sales amount should be above or equal to Rs.4000 and within 10 days, the normal XQuery output skips George and Henry records for getting bonus. But the Fuzzy XQuery considers both the salesman George and Henry for getting bonus. Moreover, the arranged order for the performance of the seven salesmen is confusing in Normal XQuery.
But Fuzzy XQuery arranges all the salesmen according to the generated Fuzzy values. Normal XQuery gave only second position for Kevin. But Kevin is given first position in Fuzzy XQuery. Because Kevin collected Rs.4998 within 3 days, whereas Joshua collected only Rs.5000 in 10 days. Martin is given fourth position in Normal XQuery output, whereas Fuzzy XQuery gives third position. The above analysis shows that Fuzzy XQuery overcomes the limitations of Normal XQuery operations.

3.2 Algorithm for Fuzzy XQuery Processing

Input: Fuzzy XQuery
Output: Displays the Relevant XML Records

Begin

Step 1. Execute Fuzzy XQuery Command

Step 2. Invoke Fuzzy Parser to identify Linguistic Term in the Fuzzy XQuery

Step 3. Execute For Loop 1 to n times (For i = 1 to n Do)

   Step 3.1 If $K_i$ is a linguistic term Then
   
   Compute the Fuzzy matching degree $F(D_i(A))$ using the Fuzzy Membership Function;
   
   End If

   Step 3.2 Extend the crisp value $F(D_i(A))$ into the Triangular Fuzzy set

   $(F(D_i(A)), F(D_i(A)), F(D_i(A)))$;

   Step 3.3 If $W_1$ (Weight1) is a linguistic term Then
   
   Find the corresponding Triangular Fuzzy set of the linguistic term based on Table 3.1.
   
   End If
Step 3.4 If $K_2$ is a linguistic term Then

Compute the Fuzzy matching degree $F(D_i(B))$ using the Fuzzy Membership Function;

End If

Step 3.5 If $W_2$ (Weight2) is a linguistic term Then

Find the corresponding triangular Fuzzy set of the linguistic term based on Table 3.1;

End If

Step 3.6 Calculate Fuzzy Scaled Weights $\overline{W_1}$ and $\overline{W_2}$ of $W_1$ and $W_2$ respectively, where

$$\overline{W_1} = W_1 \otimes (W_1 \oplus W_2^2),$$

$$\overline{W_2} = W_2 \otimes (W_1 \oplus W_2^2)$$

Step 3.7 Find the Fuzzy matching degree $F(R_i)$ of the XML Record $R_i$ where

$$F(R_i) = \left( F(D_i(A)), F(D_i(A)), F(D_i(A)) \right)$$

$$\otimes \overline{W_1} \oplus \left( F(D_i(B)), F(D_i(B)), F(D_i(B)) \right) \otimes \overline{W_2}$$

$\overline{W_1}$ and $\overline{W_2}$ are the Fuzzy Scaled Weights (FSW) of $W_1$ and $W_2$ respectively, and “$\otimes$” and “$\oplus$” are the multiplication operator and the addition operator of the Fuzzy sets respectively.

Step 3.8. Calculate the defuzzified value $Def \left( F(R_i) \right)$ of $F(R_i)$ based on Equation (3.11), where $Def \left( F(R_i) \right) \in [0, 1]$

Step 3.9 If any XML record satisfies the defuzzified fuzzy value Then

Display the XML record.

Else
XML Record Ri does not satisfy the user’s query;

End If

End For;

End;

The above algorithm takes Fuzzy XQuery from the user. Fuzzy parser is invoked to identify the Linguistic terms in the Fuzzy XQuery. If there are ‘n’ number of records in the XML Database, the algorithm will be executed for ‘n’ number of times. If a linguistic term \( K_1 \) is found, the Fuzzy Membership value is calculated using the formula

\[
f(x) = \left( 1 + \frac{(x-a)^2}{(b-a)^2} \right)^{-1}
\]

for all “n” records.

Table 3.1 gives the Fuzzy set values for the corresponding linguistic term. The triangular Fuzzy set value for Fuzzy weight \( W_1 \) is obtained from the Table 3.1. If a linguistic term \( K_2 \) is found, the Fuzzy Membership value is calculated same like the linguistic term \( K_1 \). The triangular Fuzzy set value for Fuzzy weight \( W_2 \) is obtained from the defined Table 3.1.

Fuzzy Scaled Weights (FSW) \( W_1 \) and \( W_2 \) are calculated using the Fuzzy arithmetic set operations. The formula for calculating \( W_1 \) and \( W_2 \) is

\[
W_1 = W_1 \ominus (W_1 \oplus W_2), \quad W_2 = W_2 \ominus (W_1 \oplus W_2).
\]

Fuzzy matching degree \( F(R_i) \) of the XML Record \( R_i \) is calculated for all the XML records using the formula

\[
F = W_1 \times (x_1, y_1, z_1) \oplus W_2 \times (x_2, y_2, z_2).
\]

The defuzzified value of \( F(R_i) \) is calculated using the formula

\[
Def(F(R_i)) = \frac{(a+2b+c)}{4}.
\]

The Fuzzy XQuery displays the XML
records which match the defuzzified value. The algorithm will display the result of the Fuzzy XQuery according to the Fuzzy membership values of the XML records a descending order sequence.

The working model of Fuzzy XQuery is illustrated in the following diagram (Fig 3.2).

![Diagram](image-url)

**Fig. 3.2 Working of Fuzzy XQuery**

### 3.3 Results and Discussion

Testing is carried out to compare the performance of XQuery and Fuzzy XQuery and to compare the output of a XQuery and Fuzzy XQuery.

**Performance Analysis of XQuery and Fuzzy XQuery**

Ten Fuzzy XQueries and ten Normal XQueries are taken for testing. The two types of queries are tested by using the computer with i3 processor and 2 GB RAM.

The execution time is calculated using Java program. Fig 3.3 shows the graphical representation of both the queries. It is understood from the results that the average overhead involved in Fuzzy XQueries translation is only 10-20% when compared to Normal XQueries.
Fig 3.3 Performance Analysis of XQuery and Fuzzy XQuery

Output Analysis of XQuery and Fuzzy XQuery

Testing is also carried out to analyse the output of Fuzzy XQuery and Normal XQuery. In order to carry out the testing, 100 records are stored in Salesman XML file. The different form of 10 Normal XQueries and Fuzzy XQueries are considered for conducting testing. It is found out from the testing that 10% to 25% more number of records is returned by Fuzzy XQuery than Normal XQuery. The results from the below graph (Fig 3.4) show that Fuzzy XQueries are producing better output than XQuery. But the performance wise, Fuzzy XQueries are 10% to 20% slower than Normal XQueries.

Fig 3.4 Output Analysis of Fuzzy XQuery and XQuery
3.4 Summary

Since normal XQuery use Boolean expressions for checking conditions, it fails to express user preferences and provide only discriminated output. Despite the expressive power of XQuery language, it has the problem of producing relevant output. This chapter deals with the problem of giving more flexibility to XQuery by means of Fuzzy logic use. Instead of Boolean conditions, users can give linguistic term based Conditions in Fuzzy XQuery. It is understood from the testing that Fuzzy XQueries are producing 10% to 25% better output than XQuery.

Fuzzy XQueries are 10% to 20% slower than Normal XQueries in the performance. Fuzzy XQueries are useful for decision making in E-Commerce business transactions. Fuzzy XQuery mimics human decision making in retrieving records from XML database. Fuzzy XQuery has got several advantages. But due to the complexity of syntax, users find it difficult to write XQuery and Fuzzy XQuery. In the next chapter, the user friendly GUI based tool for automatic generation of XQuery and Fuzzy XQuery is presented.