Computer and Information Science; Vol. 10, No. 4; 2017
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

Realization of Dynamic Interface and High Performance Data Retrieval

Haifeng Jiang[1] & Chang Wan[2]

[1] Humanities and Social Science College, Guangdong Polytechnic of Science and Technology, Guangdong, China

[2] School of Foreign Languages, Guangdong Polytechnic of Science and Technology, Guangdong, China

Correspondence: Haifeng Jiang, Humanities and Social Science College, Guangdong Polytechnic of Science and Technology, Zhuhai City, Guangdong Province, China. Tel: 86-138-2301-8191. E-mail: jhfone@163.com

Received: June 5, 2017 Accepted: July 01, 2017 Online Published: September 18, 2017

doi:10.5539/cis.v10n4p16 URL: http://doi.org/10.5539/cis.v10n4p16

Abstract

This paper introduces a method to realize dynamic interface, and designs a database storage model based on XML field technology to realize convenient data storage, any combination condition retrieval function and how to improve the retrieval speed in this kind of storage model. Usually a business system needs to provide information entry and retrieval functions, software designers have to design the appropriate entry items, input interface and retrieval functions for each business system and spend too much time on the repetitive works. And later engineers have to maintain the changing needs of the entry project, so we can apply the dynamic interface technology to achieve the customize needs of input items by the user, reducing the time of the repetitive works. Dynamic interface technology includes the realization of database storage and high performance data retrieval. This paper explores a storage model based on XML database to realize common and efficient storage and discuss on how to improve the retrieval speed in this kind of storage model.

Keywords: dynamic interface, XML field, high performance data retrieval

1. Technical Background

Usually a software system needs to achieve the data entry and combination of search conditions, the traditional software will usually be designed for each data input item and interface, and each entry corresponds to a database table field. Software designers often spend a lot of time on designing software entry interface and defining the corresponding database tables, and with the business changes, software engineers must also modify the software to achieve changes in demand, which greatly increases the software development and maintenance costs. To satisfy this various information entry requirements, we can use dynamic interface generation technology to achieve user-defined input information. With the development of database storage technology, especially the XML type field and the corresponding search syntax support for us to achieve dynamic interface and fast retrieval of data to provide technical support. This paper explores a storage format based on XML storage fields and the corresponding retrieval syntax to implement custom interface storage.

Traditional database storage technology can take full advantage of the database field of various features, such as keywords, indexes, restrictions and other characteristics to improve the retrieval speed.These characteristics are essential for the larger amount of data from business system. However, some of the advanced features can not be used for XML-based data type field to improve the retrieval speed, such as indexes, constraints, etc. It doesn't matter for small amount of data, but it is very important for large amount of data. So we have to adopt a certain strategy to improve the search performance.

2. Dynamic Interface Technology Design

Dynamic interface technology needs to realize dynamic interface definition and storage, dynamic generation of interface components, business data storage and combination of the conditions of the search function. The excellent dynamic interface technology includes custom flexibility and the efficiency of storage retrieval.

2.1 Dynamic Interface Definition

Likes other software development tools, first of all dynamic interface definition need to define the type of

components and the basic properties of each type, and process the interface control layout issues. For the end user, the property settings should not be too many and too complicated, and should be easy understand.According to the general input control features, the control can be divided into common attributes and special attributes:

- Common attributes:

  1. Display name
  2. Internal name (optional, default display name)
  3. Column display (single or double display, for layout)
  4. Required or not
  5. Default value
  6. Search conditions symbol (such as allow selection equal to, including, etc.)

- Special attributes

No.	field type	special attribute
1	text	
2	radio	list
3	multiple options	list
4	Numeric	numeric Type (Integer, Decimal, Currency, Percentage) calculation formula
5	Date Time	Does it contain time? calculation formula
6	text area	
7	reference field	reference entity (data table) query field filter conditions
8	multiple reference	reference entity (data table) query field
9	multi-level cascade	classification setting
10	location	automatically location?
11	pictures	Allowed to take pictures? Allow upload?
12	link type	link type (phone, URL, email, qq, etc.)
13	attachment	
14	Automatic numbering	Automatic numbering rules (eg, 1. Fixed value: enter text or letter 2. Year: {YYYY} = year; {MM} = month; {DD} = day 3. Sequence number: sequence number automatically increases.{0000} present the four-digit sequence number, the amount of digital 0 means the amount of digits. Example: Rule: Order-{YYYY}-{MM}-{DD}-{0000} Show: order -2017-01-03-0001)

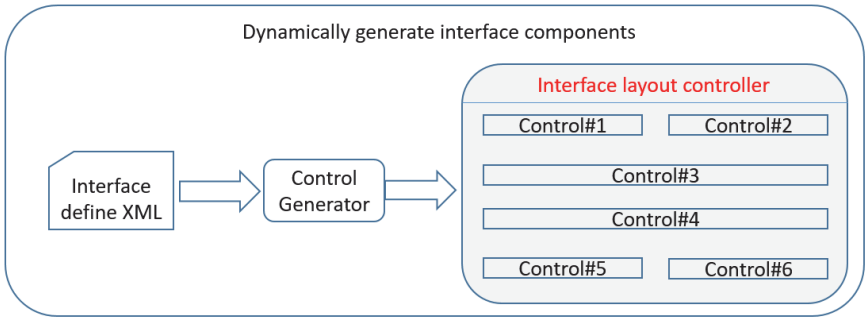
Xml can be used to express interface field storage:

```
<config ver=’1.0’>
<field type=’text’ name=’ field name #1’ {more attibutes}/>
<field type=’radio’ name=’ dictionary#1’ {more attibutes}>
<data>option#1</data>
<data>option#2</data>
</field>
```

```
</config>
```

2.2 Dynamically Generate Interface Components

The main function of dynamical generation for interface components, is based on the interface definition description xml.It dynamically generates the controls and adjust the display location and size according to the layout definition of each control by the layout controller. The functions are show as the below figure:



- Interface definition xml: defined by the dynamic interface generated by the xml information
- Control generator: According to the interface definition xml file definition information, it dynamically generates standard or user-defined controls, and submits the generated controls to the interface layout controller.
- Interface layout controller: According to the order and spatial layout properties of the controls, this controller adjust the position and size of the controls.

2.3 Business Data Storage

For dynamic interfaces, since all data is dynamically configured, the traditional way that each input data corresponds to a database table field is no longer applicable, so storage for dynamic interfaces requires formatted xml. A common storage module can automatically traverse all the controls in the interface and store the necessary information according to the type of control. At present, many databases already support the xml type of field, you can store business data in a single xml field by using the xml format definition, in order to achieve the need to adjust the purpose of the database table field, the actual use of the process is very convenient and flexible. A typical xml storage format needs to take into account the characteristics of storing data and store enough information to retrieve or reproduce data. The storage structure is designed as follows:

```
    <?xml version="1.0"?>
    <data>
      <field name#1>
      <![CDATA[content input by user]]>
    </field name#1>
    <dictionary name#1>
        <![CDATA[option value]]>
    </ dictionary name#1>
    < dictionary name#1_text>
        <![CDATA[option text]]>
    </ dictionary name#1_text>
</data>
```

Information about the text type needs to be protected using the CDATA syntax to avoid being parsed as an xml tag.

2.4 Combination of the Search Function

Data retrieval based on relational databases is a function often used by business systems, and the speed of

retrieval is critical to a system with large amounts of data. The general database provides a variety of techniques to improve the speed of retrieving data, such as keywords, constraints, indexes, etc., make full use of these features of the database is essential to improve the speed of data retrieval.

For applications that use the xml field type to store dynamic interface data, two major issues need to be addressed: data item fetching and any combination of conditional queries.

2.4.1 Data Item Extraction

Data items can take full advantage of their own database system to provide the xml operating syntax to the Microsoft database MSSQL 2005, for example, from the xml type to extract a project data syntax format:

```
SELECT {xml field name} .value ('{{xml node path}) [1]', '{converted data type}') as {alias} FROM {data table name}
```

For example, you can use the following syntax to extract "word dictionary 1_text" from above data storage structure xml:

```
SELECT xmlfield.value ('(data / dictionary dictionary 1_text) [1]', '{Nvarchar (50)}') as dictionary 1 FROM tablename
```

You can sort or filter the data which is extracted from the xml field data.

2.4.2 Any Combination of Conditions

The data items defined by the custom interface are completely unknown and therefore can not be fixed as needed for traditional software designs. So, we can provide a common, more flexible way of customizing query criteria to provide powerful data filtering and analysis capabilities. In the generic attribute defined by the interface control, there is a search conforming to the configuration, which is used to limit the way in which different data items can be used, including: equal to, not equal, greater than or equal to, less than or equal to, greater than, less than,contains,not contains and so no.At the same time,we have to consider the combination of "and" and "or" for the priority issues.The combination of priority can be achieved by allowing users to use parentheses, for example, assuming that users need to query the company for the head office, and the accounting system for the small business accounting system or small business accounting standards of the enterprise, you can meet the conditions of the input module to meet the requirements of the search filter conditions:

Relationship	Brackets	Item	Comparison	Value	Brackets
⦿And ○Or		course	equal	mathematics	
⦿And ○Or	（	Score	Large than	90	
○And ⦿Or		Evaluation	contains	good	）
⦿And ○Or					
⦿And ○Or					

3. Search Performance Optimization

Using the xml field to store data, you can take advantage of the XML syntax provided by the database and the XQuery function to get the XML node data. This provides a convenient way to read node information from a small number of result set. However, for fuzzy search based on XML node content, the search performance will be significantly reduced. We created a table with 1 million data records and created the xml master index, the xml auxiliary index -path index, the value index, and the fuzzy retrieval performance of 1 million records using different syntaxes is show as below:

Grammar and example	Execution (seconds)	Times
SELECT * FROM tablename WHERE xml_field_name.value('(/data/field_name1)[1]','nvarchar(50)') LIKE '%searchtext%'	11.5	
SELECT * FROM tablename WHERE xml_field_name.exist('/ data/field_name1/text()[contains(.,"searchtext ")]') = 1	8.3	

From the search results, fuzzy search is so much time-consuming, for the larger amount of data system, this search method performance is bad. For a particular application system, if some of the node content stored in the xml field is used for retrieval, then additional fields can be created for this node, and then the fields are indexed to improve the retrieval speed. However, for the dynamic interface technology discussed in this paper, because all the nodes are dynamically generated, the retrieval conditions are defined according to the user's need for free combination. The system can not create reasonable fields for indexing.

To improve the fuzzy lookup performance of the xml field, we need to introduce full-text indexing. The full-text engine needs to use this unique index to map each row in the table to a unique compressible key. Full-text indexes can be used for char, varchar, nchar, nvarchar, text, ntext, image, xml, varbinary, and varbinary (max) columns. To apply fulltext technology, you need to do the following:

- Create an index field and create a unique index
- Create a full-text index catalog
- Create a full-text index for the xml field

Example:

```
create table myTable
(
id int identity(1,1) primary key
,item_number int not null
,item_desc xml null
);
--fulltext unique index creating
CREATE UNIQUE INDEX uix_item_number ON myTable(item_number);
--fulltext catelog creating
CREATE FULLTEXT CATALOG ftxt_ItemInfo AS DEFAULT;
--create fulltext index
CREATE FULLTEXT INDEX ON myTable(item_desc)
KEY INDEX uix_item_number
ON ftxt_ItemInfo;
```

After applying the full-text search technique, we use the ‘contains’ function for fuzzy lookup:

Grammar and example	Execution Times (seconds)
SELECT * FROM myTable where contains(item_desc,’searchtext’)	1.230

As the results show, full-text technology can provide a high-performance search support for the custom combination of queries.

4. Results

This article describes the custom interface technology from the custom control to the database storage, retrieval and other aspects of the design ideas and practical examples.It’s very valuable for a small business system, because it can greatly shorten the development cycle, allowing users to easily generate the data needed for the business system by custom configure. Through the support of the database XML field type, we can realize the data storage and retrieval.And through the introduction and use of full-text search technology, we can greatly increase fuzzy query speed for large data of the business system. The data storage discussed in this paper is based on the string data for the application, for a large number of binary data business systems, storage methods and performance optimization also need further study.

References

Chen, C. L., & Wu, Y. Q. (2015). *SQL Server Performance Tuning In Action*:China Machine Press. https://doi.org/10.1007/978-1-4842-0061-2

Hillary Cotter, MichaelColes. (2009). *Pro Full-Text Search in SQL Server 2008*: APress. https://doi.org/10.1007/978-1-4302-1595-0_1

James, R. G., Paul, N. W., & Andrew, J. O. (2009). *SQL: The Complete Reference, 3rd Edition*, McGraw-Hill Education. https://doi.org/10.1097/00003643-200206000-00014

Jenifer, T. (2011). *Designing Interfaces*: O'Reilly. https://doi.org/10.1287/inte.1110.0544

Yang, H., Hou, R. G., & Tain, Q. H. (2010). Research on the Model of Supporting Interface Automatic Generation. China. *Computer Engineering, 36*(3), 79-82. httpS://doi.org/10.6125/14-0629-801