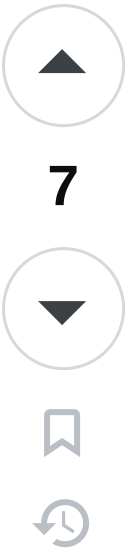
Need Pattern for dynamic search of multiple sql tables

Asked 16 years, 4 months ago Modified 11 years, 10 months ago

Viewed 4k times



I'm looking for a pattern for performing a dynamic search on multiple tables.

I have no control over the legacy (and poorly designed) database table structure.

Consider a scenario similar to a resume search where a user may want to perform a search against any of the data in the resume and get back a list of resumes that match their search criteria. Any field can be searched at anytime and in combination with one or more other fields.

The actual sql query gets created dynamically depending on which fields are searched. Most solutions I've found involve complicated if blocks, but I can't help but think there must be a more elegant solution since this must be a solved problem by now.

Yeah, so I've started down the path of dynamically building the sql in code. Seems godawful. If I really try to support the requested ability to query any combination of any field in any table this is going to be one MASSIVE set of if statements. *shiver*

I believe I read that COALESCE only works if your data does not contain NULLs. Is that correct? If so, no go, since I have NULL values all over the place.

sql database-design design-patterns dynamic-sql table-structure

Share

Improve this question

Follow

edited Feb 19, 2013 at 0:10

henser

3,317 ● 2 ● 38 ● 48

asked Aug 13, 2008 at 12:06

Geoff

9,340 ● 7 ● 39 ● 48

3 Answers

Sorted by: Highest score (default)

5

As far as I understand (and I'm also someone who has written against a horrible legacy database), there is no such thing as dynamic WHERE clauses. It has NOT been solved.

Personally, I prefer to generate my dynamic searches in code. Makes testing convenient. Note, when you create your sql queries in code, don't concatenate in user input. Use your @variables!

The only alternative is to use the COALESCE operator. Let's say you have the following table:

```
Users
-----------
Name nvarchar(20)
Nickname nvarchar(10)
```

and you want to search optionally for name or nickname. The following query will do this:

```
SELECT Name, Nickname
FROM Users
WHERE
    Name = COALESCE(@name, Name) AND
    Nickname =  COALESCE(@nick, Nickname)
```

If you don't want to search for something, just pass in a null. For example, passing in "brian" for @name and null for @nick results in the following query being evaluated:

```
SELECT Name, Nickname
FROM Users
WHERE
    Name = 'brian' AND
    Nickname =  Nickname
```

The coalesce operator turns the null into an identity evaluation, which is always true and doesn't affect the where clause.

Share Improve this answer Follow

answered Aug 13, 2008 at 14:29

user1228

1

Search and normalization can be at odds with each other. So probably first thing would be to get some kind of "view" that shows all the fields that can be searched as a single row with a single key getting you the resume. then you can throw something like Lucene in front of that to give you a full text index of those rows, the way that works is, you ask it for "x" in this view and it returns to you the key. Its a great solution and come recommended by joel himself on the podcast within the first 2 months IIRC.

Share Improve this answer

Follow

answered Aug 13, 2008 at 14:56

DevelopingChris

40.7k ● 30 ● 89 ● 119

1

What you need is something like SphinxSearch (for MySQL) or Apache Lucene.

As you said in your example lets imagine a Resume that will composed of several fields:

- List item
- Name,
- Adreess,
- Education (this could be a table on its own) or
- Work experience (this could grow to its own table where each row represents a previous job)

So searching for a word in all those fields with WHERE rapidly becomes a very long query with several JOINS.

Instead you could change your framework of reference and think of the Whole resume as what it is a Single Document and you just want to search said document.

This is where tools like Sphinx Search do. They create a FULL TEXT index of your 'document' and then you can query sphinx and it will give you back where in the Database that record was found.

Really good search results.

Don't worry about this tools not being part of your RDBMS it will save you a lot of headaches to use the appropriate model "Documents" vs the incorrect one "TABLES" for this application.

Share Improve this answer Follow

answered Jul 4, 2009 at 0:59

community wiki
elviejo79