記事
Toshihiko Minamoto · 2021年4月19日 8m read

配列プロパティ要素のための SQL インデックス

クラスの中で配列プロパティを使い、その要素 (キーと値の両方)
によってスピーディに検索を実行できると非常に便利な場合があります (EAV モデルの場合は特に重宝します)。

それでは、簡単な例を見てみましょう。

```
Class User.eav Extends %Persistent
{

Index idx1 On attributes(ELEMENTS) [ Data = entity ];

Index idx2 On (attributes(KEYS), attributes(ELEMENTS)) [ Data = entity ];

Property entity;

Property attributes As array Of %String(SQLTABLENAME = "attributes") [
SqlFieldName = attr ];

/// d ##class(User.eav).RepopulateAll()
ClassMethod RepopulateAll()
{
  d ..%DeleteExtent()

  s name=$TR("Sibe^rian pi^ne ce^dar","^",$c(769))

  s obj=..%New()
  s obj.entity="Human"
  d obj.attributes.SetAt(22,"Age")
  d obj.attributes.SetAt(186,"Height")
  d obj.attributes.SetAt("Jack","Name")
  d obj.%Save()

  s obj=..%New()
  s obj.entity="Tree"
  d obj.attributes.SetAt(186,"Age")
  d obj.attributes.SetAt(22,"Height")
  d obj.attributes.SetAt("Pines","Family")
  d obj.attributes.SetAt(name,"Name")
  d obj.%Save()

  /*

  ;or

  &sql(insert into eav(entity) select 'Human' union select 'Tree')
  &sql(insert into attributes(eav,element_key,attr)
  select 1,'Age',22 union
  select 1,'Height',186 union
  select 1,'Name','Jack' union
```

```
  select 2,'Age',186 union
  select 2,'Height',22 union
  select 2,'Family','Pines' union
  select 2,'Name',:name)
  */

  d ..Reindex()
}

/// d ##class(User.eav).Reindex()
ClassMethod Reindex()
{
  d ..%BuildIndices(,1)

  d $system.SQL.TuneTable("SQLUser.eav",1)
  d $system.SQL.TuneTable("SQLUser.attributes",1)
  d $system.OBJ.Compile($classname(),"cu/multicompile=1")
}

}
```

データを読み込んだ後に、以下を実行します。

```
USER>d ##class(User.eav).RepopulateAll()
```

すると、テーブルに以下のデータが表示されます。

ID	entity
1	Human
2	Tree

eav	ID	attr	elementkey
1	1||Age	22	Age
1	1||Name	Jack	Name
1	1||Height	186	Height
2	2||Age	186	Age
2	2||Height	22	Height
2	2||Name	Siberian pine cedar	Name
2	2||Family	Pines	Family

データを持つグローバル

```
USER>zw ^User.eavD
^User.eavD=2
^User.eavD(1)=$lb("","Human")
^User.eavD(1,"attributes","Age")=22
^User.eavD(1,"attributes","Height")=186
^User.eavD(1,"attributes","Name")="Jack"
^User.eavD(2)=$lb("","Tree")
^User.eavD(2,"attributes","Age")=186
^User.eavD(2,"attributes","Family")="Pines"
^User.eavD(2,"attributes","Height")=22
^User.eavD(2,"attributes","Name")="Sibe?rian pi?ne ce?dar"
```

インデックスを持つグローバル

```
USER>zw ^User.eavI
^User.eavI("idx1"," 186",1)=$lb("","Human")
^User.eavI("idx1"," 186",2)=$lb("","Tree")
^User.eavI("idx1"," 22",1)=$lb("","Human")
^User.eavI("idx1"," 22",2)=$lb("","Tree")
^User.eavI("idx1"," JACK",1)=$lb("","Human")
^User.eavI("idx1"," PINES",2)=$lb("","Tree")
^User.eavI("idx1"," SIBE?RIAN PI?NE CE?DAR",2)=$lb("","Tree")
^User.eavI("idx2","Age"," 186",2)=$lb("","Tree")
^User.eavI("idx2","Age"," 22",1)=$lb("","Human")
^User.eavI("idx2","Family"," PINES",2)=$lb("","Tree")
^User.eavI("idx2","Height"," 186",1)=$lb("","Human")
^User.eavI("idx2","Height"," 22",2)=$lb("","Tree")
^User.eavI("idx2","Name"," JACK",1)=$lb("","Human")
^User.eavI("idx2","Name"," SIBE?RIAN PI?NE CE?DAR",2)=$lb("","Tree")
```

それでは、以下のクエリを実行しましょう。

entity
Human
Tree

クエリは実行されるものの、インデックスではなくフルスキャンが使われています。 SMP (System Management Portal) のテーブルを見ると、そこには生成されたはずの idx1 と idx2 がありません。

これが起るのは、SQL
エンジンには、サブテーブルの配列のフィールドだけを基に作成された配列プロパティで、キー (すなわち propArray(KEY)) を持つもののインデックスしか「見えない」ためです。 両方のインデックスに「entity」フィールドがあるにもかかわらず、「attribute」サブテーブルにはそれが見当たりません。

また、attributes(KEYS) を持たない **Index idx3 On attributes(ELEMENTS);**
も表示されません。但し、こちらのインデックス

- Index idx4 On (attributes(KEYS), attributes(ELEMENTS));
- Index idx5 On (attributes(ELEMENTS), attributes(KEYS));

は表示され、クエリに考慮されます。 しかし、これらはあらゆる種類のクエリに最適という訳ではありません。

では、SQL エンジンが配列プロパティ要素のインデックスを見えるようにするには、どのメソッドを使うのが一番簡単なのでしょうか？

Caché 2015.1 では、SetCollectionProjection/GetCollectionProjection メソッドを使ってサブテーブルに表示できるコレクションであれば、それをテーブルフィールドとして表示することができます。

ですが、この機能はデフォルトで無効になっています。

これらのメソッドは、Caché
の過去のバージョンにはありませんが、手動で有効化できるかを試すことはできます。

```
%SYS>s ^%SYS("sql","sys","collection projection")=1
```

この変更を実行した後は、必ずクラスをもう一度コンパイルしてください。

それでは、このパラメーターをオン (1) にして、その動作を見てみましょう。

SMP にインデックスが表示されるようになりました。また、「eav」テーブルには「attr」という隠れたコレクションフィールドがあります。 しかし、それでも私たちが実行したクエリはインデックス idx1/idx2
を見つけられません。

それでは、お馴染みの述語 FOR SOME %ELEMENT を使って、この状況を解決しましょう。

entity
Human
Tree

インデックス idx1 がクエリで使用されています。 これを少しだけ変更します。

entity
Human

entity
Tree

最後の 2 つの例では、インデックス idx1 の代わりにインデックス idx2 が使用されています。

UPD: SQLPROJECTION を使えば、これと同じことができます。つまり、以下を実行します。

```
Property attributes As array Of %String(SQLPROJECTION = "table/column",
SQLTABLENAME = "attributes") [ SqlFieldName = attr ];
```

この記事は、こちらの記事を翻訳したものです。 [@Evgeny Shvarov
]、翻訳作業にご協力いただきありがとうございました。

この記事は、Habrahabr露 でもお読みいただけます。

本記事を書くきっかけとなったサイト: 17383689露

WRC のフレームワークについてヒントをくださった [@Alexander.Koblov] に深く感謝します。

#ObjectScript #SQL #インデックス付け #パフォーマンス #Caché

ソースURL:
https://jp.community.intersystems.com/post/%E9%85%8D%E5%88%97%E3%83%97%E3%83%AD%E3%83%91%E3%83%86%E3%82%A3%E8%A6%81%E7%B4%A0%E3%81%AE%E3%81%9F%E3%82%81%E3%81%AE-sql-%E3%82%A4%E3%83%B3%E3%83%87%E3%83%83%E3%82%AF%E3%82%B9