記事
Toshihiko Minamoto · 2020年10月27日 14m read

Cachéでの照合

> 秩序（順序）はだれにとっても必要であるが、皆が同じように秩序（順序）を理解しているわけではない（ファウスト・セルチニャーニ）

免責事項: この記事では、例としてロシア語とキリル文字を使用しますが、英語以外のロケールでCachéを使用するすべての方に関連のある記事です。
この記事は主にNLS照合について言及しており、SQL照合とは異なることに注意してください。 SQL照合（SQLUPPER、SQLSTRING、照合なしを意味するEXACT、TRUCATEなど）は、値に明示的に適用される実際の関数であり、その結果はグローバルサブスクリプトに明示的に格納されることがあります。
サブスクリプトに格納されると、これら
の値は当然、有効なNLS照合（「SQLおよびNLS照合」）に従うことになります。

Cachéのデータ、メタデータ、クラス、ルーチンはすべてグローバルに格納されます。
グローバルは永続的です。 グローバルノードはサブスクリプト値によって順序付けされ、検索やディスクフェッチのパフォーマンスを向上させるために、挿入された順ではなくソート順でストレージデバイスに格納されます。

```
USER>set ^a(10)=""
USER>set ^a("??")=""
USER>set ^a("??")=""
USER>set ^a(2)=""
USER>zwrite ^a
^a(2)=""
^a(10)=""
^a("??")=""
^a("??")=""
```

ソート中、Cachéは数値と文字列を区別します。2は数値として扱われ、10より前にソートされます。
コマンドZWriteおよび関数$Orderと$Query
は、これらのサブスクリプトが格納された順でグローバルサブスクリプトを返します。初めに空の文字列（サブスクリプトとして使用できません）、その後に負の数値、ゼロ、正の数値、そして照合で定義された順で文字列を返します（照合）。

Cachéの標準的な照合は（当然ながら）Caché標準と呼ばれており、
文字列はUnicode文字コードに従ってソートされます。

現在のプロセスのローカル配列の照合はロケールによって定義されます（管理ポータル > システム管理 > 構成 > システム構成 > 国言語設定 > Locale Definitions）。
CachéのUnicodeインストールのロシア語ロケールはruswで、ruswのデフォルトの照合はCyrillic3です。
ruswロケールで指定できる照合には、Caché標準、Cyrillic1、Cyrillic3、Cyrillic4、Ukrainian1があります。

ClassMethod ##class(%Collate).SetLocalName()は現在のプロセスのローカル配列に使用する照合を設定します。

```
USER>write ##class(%Collate).GetLocalName()
Cyrillic3
USER>write ##class(%Collate).SetLocalName("Cache standard")
1
```

```
USER>write ##class(%Collate).GetLocalName()
Cache standard
USER>write ##class(%Collate).SetLocalName("Cyrillic3")
1
USER>write ##class(%Collate).GetLocalName()
Cyrillic3
```

すべての照合には、数値を文字列としてソートする照合があります。
その照合の名前には末尾に「string」が含まれています。

```
USER>write ##class(%Collate).SetLocalName("Cache standard string")
1
USER>kill test

USER>set test(10) = "", test(2) = "", test("??") = "", test("??") = ""

USER>zwrite test
test(10)=""
test(2)=""
test("??")=""
test("??")=""

USER>write ##class(%Collate).SetLocalName("Cache standard")
1
USER>kill test

USER>set test(10) = "", test(2) = "", test("??") = "", test("??") = ""

USER>zwrite test
test(2)=""
test(10)=""
test("??")=""
test("??")=""
```

Caché標準とCyrillic3

Caché標準は、コードに従って文字をソートします。

```
 write ##class(%Library.Collate).SetLocalName("Cache standard"),!
 write ##class(%Library.Collate).GetLocalName(),!
 set letters = "??????????????????????????????????"
 set letters = letters _ $zconvert(letters,"U")
 kill test

 //fill local array “test” with data
 for i=1:1:$Length(letters) {
     set test($Extract(letters,i)) = ""
 }

 //print test subscripts in sorted order
 set l = "", cnt = 0
 for  {
     set l = $Order(test(l))
     quit:l=""
```

```
     write l, " ", $Ascii(l),","
     set cnt = cnt + 1
     write:cnt#8=0 !
 }
```

```
USER>do ^testcol
1
Cache standard
? 1025,? 1040,? 1041,? 1042,? 1043,? 1044,? 1045,? 1046,
? 1047,? 1048,? 1049,? 1050,? 1051,? 1052,? 1053,? 1054,
? 1055,? 1056,? 1057,? 1058,? 1059,? 1060,? 1061,? 1062,
? 1063,? 1065,? 1066,? 1067,? 1068,? 1069,? 1070,? 1071,
? 1072,? 1073,? 1074,? 1075,? 1076,? 1077,? 1078,? 1079,
? 1080,? 1081,? 1082,? 1083,? 1084,? 1085,? 1086,? 1087,
? 1088,? 1089,? 1090,? 1091,? 1092,? 1093,? 1094,? 1095,
? 1097,? 1098,? 1099,? 1100,? 1101,? 1102,? 1103,? 1105,
```

キリル文字は、ロシア語のアルファベット順に出力されますが、「ë」と「Ë」は例外です。
これらのUnicode文字コードは順不同であり、
「Ë」は「E」と「Д」の間、「ë」は「e」と「д」の間で照合されます。 そのため、ロシア語ロケールには独自の照合として、ロシア語のアルファベットと同じ順に並ぶ文字を使用するCyrillic3が必要となります。

```
USER>do ^testcol
1
Cyrillic3
? 1040,? 1041,? 1042,? 1043,? 1044,? 1045,? 1025,? 1046,
? 1047,? 1048,? 1049,? 1050,? 1051,? 1052,? 1053,? 1054,
? 1055,? 1056,? 1057,? 1058,? 1059,? 1060,? 1061,? 1062,
? 1063,? 1065,? 1066,? 1067,? 1068,? 1069,? 1070,? 1071,
? 1072,? 1073,? 1074,? 1075,? 1076,? 1077,? 1105,? 1078,
? 1079,? 1080,? 1081,? 1082,? 1083,? 1084,? 1085,? 1086,
? 1087,? 1088,? 1089,? 1090,? 1091,? 1092,? 1093,? 1094,
? 1095,? 1097,? 1098,? 1099,? 1100,? 1101,? 1102,? 1103,
```

Cachéのobjectscriptには、「後にソート」という特殊なバイナリ演算子「]]」があります。 最初のオペランドを持つサブスクリプトが2番目のオペランドを持つサブスクリプトの後にソートされる場合は1を返し、そうでない場合は0を返す演算子です。

```
USER>write ##class(%Library.Collate).SetLocalName("Cache standard"),!
1
USER>write "?" ]] "?"
1
USER>write ##class(%Library.Collate).SetLocalName("Cyrillic3"),!
1
USER>write "?" ]] "?"
0
```

グローバルと照合

同一のデータベースに含まれるグローバルには、さまざまな照合が使用される場合があります。
各データベースには構成オプションがあり、新しいグローバルにはデフォルト照合が使用されます。
インストール直後、USERを除くすべてのデータベースはCaché標準のデフォルト照合を使用します。

USERデータベースのデフォルト照合はインストールロケールによって決まるため、
ruswの場合はCyrillic3となります。

データベースに対しデフォルト以外の照合を使用してグローバルを作成するには、##class(%GlobalEdit).Createメソッドを使用します。

```
USER>kill ^a
USER>write ##class(%GlobalEdit).Create(,"a",##class(%Collate).DisplayToLogical("Cache
 standard"))
```

管理ポータル（システムエクスプローラ >
Globals）のグローバルのリストに、各グローバルの照合列があります。

既存のグローバルの照合を変更することはできないため、
新しい照合でグローバルを作成して、Mergeコマンドを使ってデータをコピーする必要があります。
グローバルの一括変換を行うには##class(SYS.Database).Copy()を使用します。

Cyrillic4、Cyrillic3、およびウムラウト

文字列サブスクリプトを内部形式に変換するには、Cyrillic3照合には、Caché標準照合にかかる時間よりもはるかに長い時間がかかるため、Cyrillic3照合を使用したグローバル（またはローカル）配列の挿入とルックアップの処理が遅くなります。 Caché 2014.1には新しい照合であるCyrillic4が含まれており、Cyrillic3と同じ正しい文字順になりますが、パフォーマンスが改善されます。

```
for collation="Cache standard","Cyrillic3","Cyrillic4" {
     write ##class(%Library.Collate).SetLocalName(collation),!
     write ##class(%Library.Collate).GetLocalName(),!
     do test(100000)
 }
 quit
test(C)
 set letters = "????????????????????????????????"
 set letters = letters _ $zconvert(letters,"U")

 kill test
 write "test insert: "
 //fill local array “test” with data
 set z1=$zh
 for c=1:1:C {
     for i=1:1:$Length(letters) {
         set test($Extract(letters,i)_"???? ??????? ??????? ?????" _ $Extract(letters
,i)) = ""
     }
 }
 write $zh-z1,!

 //looping through test subscripts
 write "test $Order: "
 set z1=$zh
 for c=1:1:C {
     set l = ""
     for  {
         set l = $Order(test(l))
         quit:l=""
```

```
    }
 }
 write $zh-z1,!
```

```
USER>do ^testcol
1
Cache standard
test insert: 1.520673
test $Order: 2.062228
1
Cyrillic3
test insert: 3.541697
test $Order: 5.938042
1
Cyrillic4
test insert: 1.925205
test $Order: 2.834399
```

Cyrillic4は、まだruswロケールのデフォルトの照合ではありませんが、ruswに基づいて独自のロケールを定義し、Cyrillic4をローカル配列のデフォルトの照合として指定することができます。
または、データベース設定でグローバルの新しいデフォルト照合としてCyrillic4を設定することもできます。

Cyrillic3は、2つの文字列を各文字コードに基づいてソートするのに比べてさらに一般的なアルゴリズムに基づいているため、Caché標準とCyrillic4よりも低速になります。

ドイツ語の場合、ソート時には文字はssとして照合されます。 Cachéは次の通り、そのルールを尊重します。

```
USER>write ##class(%Collate).GetLocalName()
German3
USER>set test("Straßer")=1
USER>set test("Strasser")=1
USER>set test("Straster")=1
USER>zwrite test
test("Strasser")=1
test("Straßer")=1
test("Straster")=1
```

サブスクリプトの文字列のソート順に注目してください。 特に、最初の文字列の頭の4文字は「Stras」で、次に「Straß」、そしてもう一度「Stras」となっているところです。 照合が個別の文字のコードに基づいてソートされているだけであれば、この方法で文字列をソートすることは不可能です。

もう1つの例として、フィンランド語の場合、「v」と「w」は同一の文字として照合される必要があります。 ロシア語の照合ルールはより単純であり、各文字にある特定のコードを与え、これらのコードでソートするだけで十分です。 この方法で、照合Cyrillic4 のパフォーマンスはCyrillic3と比べて改善されています。

照合とSQL

配列の照合とSQL照合を混同しないようにしてください。 SQL照合は、比較の前に文字列に適用される変換、またはインデックスグローバルでサブスクリプトとして使用する変換です。
CachéのデフォルトのSQL照合はSQLUPPERで、
すべての文字を大文字に変換し、スペース文字を削除して、文字列の先頭にスペースを1つ追加します。
ほかのSQL照合（EXACT、SQLSTRING、
TRUNCATE）については、こちらのドキュメントに説明されています。

同一のデータベース内の異なるグローバルにさまざまな照合が使用されており、ローカル配列にも別の照合が使用されている場合、たやすく混乱が生じます。 SQLは一時データにCACHETEMPデータベースを使用しますが、 CACHETEMPのグローバルのデフォルト照合は、Cachéのインストールロケールの照合とは異なる可能性があります。

主なルールが1つあります。期待される順序で行を返すSQLクエリのORDER BYについては、データと関連するテーブルのインデックスがソートされるグローバルの照合は、CACHETEMPデータベースのデフォルトの照合と ローカル配列の照合と同一である必要があります。
詳細については、ドキュメントの「SQLとNLS照合」の段落を参照してください。

では、テストクラスを作成しましょう。

```
Class Collation.test Extends %Persistent
{

Property Name As %String;

Property Surname As %String;

Index SurInd On Surname;

ClassMethod populate()
{
    do ..%KillExtent()

    set t = ..%New()
    set t.Name = "?????", t.Surname = "?????"
    write t.%Save()

    set t = ..%New()
    set t.Name = "????", t.Surname = "??????"
    write t.%Save()

    set t = ..%New()
    set t.Name = "??????", t.Surname = "???????????"
    write t.%Save()
}

}
```

クラスにデータを入力します（後で、ドイツ語を使った前の例の単語を使用してください）。

```
USER>do ##class(Collation.test).populate()
```

クエリを実行します。

```
select name from collation.test order by name
```

Row count: 3 Performance: **0.002** seconds **60** global references Cached Query: %sqlcq.USER.cls1 Last update: 2016-05-1

Name
Пётр
Павел
Прохор

3 row(s) affected

予想外の結果となりました。 主な疑問は、名前がなぜアルファベット順に並べられないのか（Павел、Пётр、Прохор）というところにあります。 クエリプランを見てみましょう。

このプランのキーワードは、「一時ファイルを作成する」です。
SQLエンジンは、このクエリの実行に一時的な構造を使用することに決定しました。「ファイル」と呼ばれてはいますが、実際にはプロセスプライベートのグローバルであり、場合によってはローカル配列です。
このグローバルのサブスクリプトは、この特定のケースでは個人名で順序付けする値です。 プロセスプライベートのグローバルは、CACHETEMPデータベースに格納され、CACHETEMPの新しいグローバルのデフォルト照合はCaché標準です。

また、「ë」がなぜ最後ではなく最初に返されているのかという別の合理的な疑問もあります（Caché標準では「ë」はロシア語のすべての文字と「Ë」の前ではなく、後にソートされます）。 一時グローバルのサブスクリプトはNameフィールドの実際の値では
なく、Nameを大文字化した値（SQLUPPERは文字列のデフォルトのSQL照合
）であるため、「Ë」がほかの文字の前に返されているのです。

%Exact
関数を使用してデフォルトの照合を変更すると、依然として誤りではありますが、少なくとも「ë」がほかの文字の後にソートされるという、期待された結果が得られます。

ここでは、CACHETEMPのデフォルトの照合を変更せずに、Surname列のクエリを確認してみましょう。
この列のインデックスは、^Collation.testIグローバルに格納されます。
そのグローバルの照合はCyrillic3であるため、行の正しい順序を確認できるはずです。

またしても、誤りです。「Ë」は 「A」と「И」の間である必要があります。 クエリプランを見てみましょう。

SQLUPPERがSurInd インデックスの値に適用されているため、Surnameフィールドの元の値を出力できる十分なインデックスデータがありません。 SQLエンジンは、Name列と同様に、テーブル自体の値を使用して一時ファイルの値を並べ替えるように決定しています。

クエリに、Surnameが大文字でも良いことを記述することができます。 行はインデックスグローバル ^Collation.testI から直接取得されるため、順序は正しくなります。

クエリプランは期待通りです。

では、ずっと前に行っているはずの、CACHETEMPデータベースのデフォルト照合をCyrillic3（またはCyrillic4）に変更する作業を行いましょう。

一時ファイルを使用するクエリは、正しい順で行を出力します。

要約

- ローカルアルファベットのニュアンスを気にしない場合は、Caché標準の照合を使用します。
- 一部の照合（Cyrillic4）は、ほかの照合（Cyrillic3）よりもパフォーマンスが優れています。
- CACHETEMPの照合がメインデータベースとローカル配列の照合と同じであることを確認します。

#ObjectScript #SQL #グローバル #Caché

ソースURL:
https://jp.community.intersystems.com/post/cach%C3%A9%E3%81%A7%E3%81%AE%E7%85%A7%E5%90%88