文章
姚 鑫 · 十一月 14, 2021 阅读大约需 8 分钟

第七十六章 SQL命令 TOP

第七十六章 SQL命令 TOP

指定返回多少行的SELECT子句。

大纲

```
SELECT  [DISTINCT clause]
  [TOP {[((]int[))] | ALL}]
  select-item{,select-item}
```

参数

- int - 限制返回到指定整数的行数。
  int参数可以是一个正整数、一个动态SQL输入参数(?)或一个解析为正整数的嵌入式SQL主机变量(:var)。在动态SQL中，int值可以选择用单括号或双括号括起来(双括号是首选语法);
  这些括号禁止在相应的缓存查询中对int值进行文字替换。
- ALL - TOP ALL仅在子查询或CREATE VIEW语句中有意义。
  它用于在这些情况下支持使用ORDER BY子句，满足在子查询或CREATE VIEW中使用的查询中ORDER BY子句必须与TOP子句配对的要求。
  TOP ALL不限制返回的行数。

描述

可选的TOP子句出现在SELECT关键字和可选的DISTINCT子句之后，以及第一个选择项之前。

TOP关键字用于动态SQL和基于指针的嵌入式SQL。
在非游标嵌入式SQL中，TOP关键字的唯一有意义的用法是TOP 0。
任何其他TOP
int(其中int是任何非零整数)都是有效的，但没有意义，因为非游标嵌入式SQL中的SELECT总是最多返回一行数据。

SELECT语句的TOP子句将返回的行数限制为int中指定的行数。
如果没有指定TOP子句，则默认显示满足SELECT条件的所有行。
如果指定了TOP子句，则显示的行数或行数要么为int，要么为满足查询谓词要求的所有行(以较小的为准)。
如果指定ALL, SELECT将返回表中满足查询谓词要求的所有行。

如果查询中没有指定ORDER BY子句，那么将哪些记录作为 “ top ” 行返回是不可预测的。
如果指定了ORDER BY子句，则顶部的行与该子句中指定的顺序一致。

DISTINCT子句(如果指定)应用于TOP之前，指定返回(最多)int个唯一值。

当所有行已交付时，TOP短路。
因此，如果选择直到获得SQLCODE 100，则设置SQLCODE 100的FETCH是即时的。

当通过视图或FROM子句子查询访问数据时，可以使用%vid视图ID而不是(或附加)TOP子句来限制返回的行数。

TOP int值

int数值可以是整数、数字字符串、动态SQL输入参数(?)或解析为整数值的输入主机变量(:var)。

int值指定要返回的行数。
允许的值是0和正数。
不能将int值指定为算术表达式、字段名、子查询列别名、标量函数或聚合函数。
小数或数字字符串被解析为其整数值。
0(0)是一个有效的整型值。
TOP 0执行查询，但不返回数据。

TOP ALL必须在查询中指定为关键字。
不能将ALL指定为?
输入参数或:var主机变量值。
查询解析器将以这种方式提供的字符串 “ ALL ” 解释为值为0的数字字符串。

注意，TOP参数元数据返回为xDBC数据类型12 (VARCHAR)，而不是4 (INTEGER)，因为可以将TOP int指定为数字字符串或整数。

TOP和缓存查询

int值可以用括号指定，也可以不使用括号指定。
这些括号影响如何缓存动态SQL查询(非游标嵌入式SQL查询不缓存)。
没有括号的整型值被转换为 a?
缓存查询中的参数变量。
这意味着重复使用不同的TOP int值调用相同的查询将调用相同的缓存查询，而不是每次都准备和优化查询。

括起来的圆括号禁止文字替换。
例如，TOP(7))。
当int被括在括号中时，缓存的查询保留特定的int值。
使用相同的TOP int值重新调用查询将使用缓存的查询;
使用不同的TOP int值调用查询将导致SQL准备、优化和缓存这个新版本的查询。

TOP ALL不是缓存为 a?
参数变量。
ALL被解析为关键字，而不是字面量。
因此，使用TOP 7和TOP ALL的相同查询将生成两个不同的缓存查询。

TOP和ORDER BY

TOP通常用于带ORDER BY子句的SELECT中。
注意，默认升序ORDER BY排序顺序认为NULL是最低值(" top ")，后面跟着空字符串(")。

当指定ORDER BY子句时，在子查询SELECT或CREATE VIEW SELECT中需要TOP。
在这些情况下，可以指定TOP int(以限制返回的行数)或TOP ALL。

TOP ALL只在子查询或CREATE VIEW语句中使用。
它用于在这些情况下支持使用ORDER BY子句，以满足在子查询或CREATE VIEW查询中ORDER BY子句必须与TOP子句配对的要求。
TOP ALL不限制返回的行数。
前所有…
ORDER BY不会改变默认的SELECT优化。
ALL关键字不能用括号括起来。

TOP 优化

默认情况下，SELECT优化以最快的时间返回所有数据。
同时添加TOP int子句和ORDER BY子句可以优化以最快的时间返回第一行。
(注意，这两个子句都需要更改优化。)
可以使用%SYS.PTools。
StatsSQL类TotalTimeToFirstRow属性返回返回第一行所需的时间。

以下是特殊情况下的优化:

- 可能希望使用TOP和ORDER BY优化策略，而不限制返回的行数;
  例如，如您正在返回以页面单元显示的数据。
  在这种情况下，能希望发出一个TOP子句，该子句的int值大于行总数。
- 可能希望限制返回的行数并指定它们的顺序，而不改变默认的SELECT优化。
  在这种情况下，指定TOP子句、ORDER
  BY子句和%NOTOPOPT关键字，以保留返回所有数据优化所需的最快时间。

TOP与聚合和函数

聚合函数或标量函数只能返回单个值。
如果查询选择项列表中只包含聚合和函数，则TOP子句的应用如下:

- 如果选择项列表包含聚合函数，例如COUNT(*)或AVG(Age)，且不包含任何字段引用，则返回的行数不超过一行，无论TOP int值或ORDER BY子句是否存在。
  这些子句被验证，但被忽略。
  以下例子显示了这一点:

```
SELECT TOP 5 AVG(Age),CURRENT_TIMESTAMP(3) FROM Sample.Person
  /* returns 1 row */


SELECT TOP 1 AVG(Age),CURRENT_TIMESTAMP(3) FROM Sample.Person ORDER BY Age
  /* returns 1 row */
```

- 如果选择项列表包含一个或多个标量函数、表达式、文字(如%TABLENAME)、子查询或宿主变量，并且不包含任何字段引用或聚合，则应用TOP子句。
  下面的例子显示了这一点:

```
SELECT TOP 5 ROUND(678.987,2),CURRENT_TIMESTAMP(3) FROM Sample.Person
  /* returns 5 identical rows */
```

返回的实际行数取决于表中的行数，即使在没有引用表字段时也是如此。
例如:

```
SELECT TOP 300 CURRENT_TIMESTAMP(3) FROM Sample.Person
  /* returns either the number of rows in Sample.Person
     or 300 rows, whichever is smaller */
```

当查询受到谓词条件的限制时，即使在选择项列表中没有引用表字段，返回的行数也会受到该条件的限制。
例如:

```
SELECT TOP 300 CURRENT_TIMESTAMP(3) FROM Sample.Person WHERE Home_State = 'MA'
  /* returns either the number of rows in Sample.Person
     where Home_State = 'MA'
```

```
     or 300 rows, whichever is smaller */
```

- 如果SELECT语句不包含FROM子句，则不管TOP值如何，最多返回一行。
  例如:

```
SELECT TOP 5 ROUND(678.987,2),CURRENT_TIMESTAMP(3)
  /* returns 1 row */
```

- DISTINCT子句进一步限制了TOP子句。
  如果不同的值比TOP值少，则只返回具有不同值的行。
  当仅引用标量函数时，只返回一行。
  例如:

```
SELECT DISTINCT TOP 15 CURRENT_TIMESTAMP(3) FROM Sample.Person
  /* returns 1 row */
```

- TOP 0总是不返回任何行，不管选择项列表的内容是什么，也不管SELECT语句是包含FROM子句还是DISTINCT子句。

在非游标嵌入式SQL中，TOP 0查询不返回任何行，并设置SQLCODE=100;带有TOP 1(或任何其他TOP int值)的非游标嵌入式SQL查询返回一行并设置SQLCODE=0。
在基于指针的嵌入式SQL中，获取循环的完成总是设置SQLCODE=100，而不管TOP int值如何。

示例

下面的查询返回从Sample检索到的前20行。
人按他们在数据库中的存储顺序排列。
这个记录顺序通常是不可预测的。

```
SELECT TOP 20 Home_State,Name FROM Sample.Person
```

下面的查询返回从Sample检索到的前20个不同的HomeState值。
人在升序排列顺序。

```
SELECT DISTINCT TOP 20 Home_State FROM Sample.Person ORDER BY Home_State
```

下面的查询返回前40个不同的FavoriteColor值。
“ top ”行反映了Sample中所有行的ORDER BY子句排序。
按降序(DESC)排序的人。
使用降序排序序列而不是默认的升序排序序列，因为众所周知FavoriteColors字段有null，它将出现在升序排序序列的顶部。

```
SELECT DISTINCT TOP 40 FavoriteColors FROM Sample.Person
      ORDER BY FavoriteColors DESC
```

还要注意，在前面的示例中，由于FavoriteColors是一个列表字段，排序序列包括元素长度字节。
因此，六个字母的元素(YELLOW, PURPLE, ORANGE)被放在一起整理，在五个字母的元素(WHITE, GREEN等)之前列出。

动态SQL可以指定int值作为输入参数(用 “ ? ” 表示)。
在下面的例子中，TOP ?
输入参数被%Execute方法设置为10:

```
ClassMethod Top()
{
    s myquery = "SELECT TOP ? Name,Age FROM Sample.Person"
    s tStatement = ##class(%SQL.Statement).%New()
    s qStatus = tStatement.%Prepare(myquery)
    s rset = tStatement.%Execute(10)
    d rset.%Display()
}
```

```
DHC-APP>d ##class(PHA.TEST.SQLCommand).Top()
Name    Age
yaoxin  31
xiaoli
??    7
??    7
...
10 Rows(s) Affected
```

以下基于游标的嵌入式SQL示例执行相同的操作:

```
ClassMethod Top1()
{
    SET topnum = 10
    &sql(
        DECLARE pCursorT CURSOR FOR
        SELECT TOP :topnum Name,Age INTO :name,:years FROM Sample.Person
    )
    &sql(OPEN pCursorT)
    q:(SQLCODE'=0)
    for {
        &sql(FETCH pCursorT)
        q:SQLCODE
        w "Name=",name," Age=",years,!
    }
    &sql(CLOSE pCursorT)
}
```

```
DHC-APP>d ##class(PHA.TEST.SQLCommand).Top1()
Name=yaoxin Age=31
Name=xiaoli Age=
Name=?? Age=7
Name=?? Age=7
Name=?? Age=43
Name=?? Age=
```

#SQL #Caché

源URL:
https://cn.community.intersystems.com/post/%E7%AC%AC%E4%B8%83%E5%8D%81%E5%85%AD%E7%AB%A0-sql%E5%91%BD%E4%BB%A4-top