文章
姚 鑫 · 十一月 18, 2021 阅读大约需 9 分钟

第八十章 SQL命令 UNION

第八十章 SQL命令 UNION

组合两个或多个SELECT语句。

大纲

```
select-statement {UNION [ALL] [%PARALLEL] select-statement}

select-statement {UNION [ALL]  [%PARALLEL] (query)}

(query) {UNION [ALL]  [%PARALLEL] select-statement}

(query) {UNION [ALL]  [%PARALLEL] (query)}
```

参数

- ALL - 可选——关键字字面量。
  如果指定，则返回重复的数据值。
  如果省略，重复的数据值将被抑制。
- %PARALLEL - 可选—%PARALLEL关键字。
  如果指定，则union的每一边都作为单独的进程并行运行。
- select-statement - 从数据库中检索数据的SELECT语句。
- query - 组合一个或多个SELECT语句的查询。

描述

UNION将两个或多个查询组合为一个查询，该查询将数据检索到结果中。
由UNION组合的查询可以是由单个SELECT语句组成的简单查询，也可以是复合查询。

为了在SELECT语句之间实现联合，每个分支中指定的列数必须匹配。
指定具有不同列数的select将导致SQLCODE -9错误。
可以指定一个SELECT中的NULL列与另一个SELECT中的数据列配对，以匹配列的数量。
例如:

```
SELECT Name,Salary,BirthDate
FROM Sample.Employee
UNION ALL
SELECT Name,NULL,BirthDate
FROM Sample.Person
```

SQL通过自动计算UNION查询的所有分支并返回优先级最高的数据类型来确定结果列数据类型:VARCHAR、DOUBLE、NUMERIC、BIGINT、INTEGER、SMALLINT、TINYINT。
其他数据类型，如DATE，没有分配优先级。

例如，下面的程序返回数据类型TINYINT，尽管DATE数据类型在其他上下文中具有更高的优先级。

```
SELECT MyTinyIntField FROM Table1
     UNION ALL
SELECT MyDateField FROM Table2
```

如果你想返回一个不同于所列数据类型的数据类型，你必须使用显式CAST语句，如下面的例子所示:

```
SELECT CAST(MyTinyInt AS DATE) FROM Table1
     UNION ALL
SELECT MyDateField FROM Table2
```

如果联合分支中的列在长度、精度或比例上不同，则给结果列分配最大的值。

结果列名取自联合的第一个分支中的列(或列别名)的名称。
在两个分支中对应的列没有相同名称的情况下，在所有分支中使用相同的列别名来标识结果列可能会很有用。

如果任何UNION分支中的任何列是空的，则结果列元数据报告为空的。

UNION结果中的字符串字段具有相应SELECT字段的排序规则类型，但如果字段排序规则不匹配，则分配精确排序规则。

UNION and UNION ALL

普通的UNION消除了结果中的重复行(所有值都相同)。
UNION ALL在结果中保留重复的行。

不同精度的字段不具有相同的值。
例如，值33(数据类型NUMERIC(9))和33.00(数据类型NUMERIC(9,2))并不被认为是相同的。

具有不同排序规则的字段没有相同的值。
例如，MyStringField和%SQLUPPER(MyStringField)并不被认为是相同的，即使这两个值都是大写的。

TOP和ORDER BY子句

UNION语句可以以ORDER BY子句结束，该子句对结果进行排序。
这个ORDER BY适用于整个语句;
它必须是最外层查询的一部分，而不是子查询。
它不必与TOP子句配对。
下面的例子展示了ORDER BY的使用:两个SELECT语句选择数据，数据由UNION组合，然后ORDER BY对结果进行排序:

```
SELECT Name,Home_Zip FROM Sample.Person
  WHERE Home_Zip %STARTSWITH 9
UNION
SELECT Name,Office_Zip FROM Sample.Employee
  WHERE Office_Zip %STARTSWITH 8
ORDER BY Home_Zip
```

在ORDER BY中使用与SELECT列表列不对应的列号会导致SQLCODE -5错误。
在ORDER BY中使用与SELECT列表列不对应的列名会导致SQLCODE -6错误。

union 的SELECT语句(或两者)也可以包含ORDER BY子句，但它必须与TOP子句配对。
这个ORDER BY用于确定TOP子句选择了哪些行。
下面的示例展示了ORDER BY的使用:两个SELECT语句都使用ORDER
BY对它们的行进行排序，这决定了哪些行被选为顶部行。
选定的数据由UNION组合，然后最终的ORDER by对结果进行排序:

```
SELECT TOP 5 Name,Home_Zip FROM Sample.Person
  WHERE Home_Zip %STARTSWITH 9
  ORDER BY Name
UNION
SELECT TOP 5 Name,Office_Zip FROM Sample.Employee
  WHERE Office_Zip %STARTSWITH 8
  ORDER BY Office_Zip
ORDER BY Home_Zip
```

TOP可以应用于union中的第一个SELECT，也可以应用于union的结果，这取决于ORDER BY子句的位置:

- TOP...ORDER BY应用于UNION结果：如果UNION位于FROM子句的子查询中，则TOP和ORDER BY将应用于UNION的结果。例如：

```
SELECT TOP 10 Name,Home_Zip
  FROM (SELECT Name,Home_Zip FROM Sample.Person
          WHERE Name %STARTSWITH 'A'
        UNION
        SELECT Name,Home_Zip FROM Sample.Person
          WHERE Home_Zip %STARTSWITH 8)
ORDER BY Home_Zip
```

- TOP适用于第一个SELECT；ORDER BY适用于UNION结果。例如：

```
SELECT TOP 10 Name,Home_Zip
  FROM Sample.Person
  WHERE Name %STARTSWITH 'A'
UNION
SELECT Name,Home_Zip FROM Sample.Person
  WHERE Home_Zip %STARTSWITH 8
ORDER BY Home_Zip
```

括起圆括号

UNION支持对其中一条SELECT语句或两条SELECT语句或整个UNION语句使用可选的圆括号。可以指定一对或多对括号。以下是括号的所有有效用法：

```
(SELECT ...) UNION SELECT ...
(SELECT ...) UNION (SELECT ...)
((SELECT ...)) UNION ((SELECT ...))
(SELECT ... UNION SELECT ...)
((SELECT ...) UNION (SELECT ...))
```

每次使用圆括号都会生成一个单独的缓存查询。

UNION/OR 优化

默认情况下，SQL自动优化会在认为合适的情况下将UNION子查询转换为OR条件。此UNION/OR转换允许EXISTS和其他低级谓词迁移到顶级条件，以便它们可用于 IRIS查询优化器索引。此默认转换在大多数情况下都是可取的。但是，在某些情况下，这种UNION/OR转换会带来很大的开销负担。%NOUNIONOROPT查询优化选项为与FROM子句关联的WHERE子句中的所有条件禁用此自动UNION/OR转换。因此，在复杂查询中，可以对一个子查询禁用自动UNION/OR优化，而在其他子查询中允许它。

如果将包含子查询的条件应用于UNION，则该条件将在每个UNION操作数内应用，而不是在末尾应用。这允许在每个UNION操作数中应用子查询优化。有关子查询优化选项的说明，请参阅FROM子句。在下面的示例中，WHERE子句条件应用于联合中的每个子查询，而不是联合的结果：

```
SELECT Name,Age FROM
  (SELECT Name,Age FROM Sample.Person
   UNION SELECT Name,Age FROM Sample.Employee)
WHERE Age IN (SELECT TOP 5 Age FROM Sample.Employee WHERE Age>55 ORDER BY Age)
```

联合所有聚合优化

UNION ALL的SQL自动优化将顶级聚合推入UNION的分支中。
无论是否使用%PARALLEL关键字，都可以显著提高性能，例如:

```
SELECT COUNT(*) FROM (SELECT item1 FROM table1 UNION ALL SELECT item2 FROM table2)
```

优化:

```
SELECT SUM(y) FROM (SELECT COUNT(*) AS y FROM table1 UNION ALL SELECT COUNT(*) AS y F
ROM table2)
```

此优化适用于所有顶级聚合函数(不仅仅是COUNT)，包括具有多个顶级聚合函数的查询。
要应用此优化，外部查询必须是一个 “ onerow ” 查询，没有WHERE或GROUP BY子句，它不能引用%VID，并且UNION ALL必须是其FROM子句中的唯一流。
聚合不能嵌套，任何使用的聚合函数都不能使用%FOREACH() grouping或DISTINCT。

并行处理

关键字%PARALLEL支持多处理器系统上的并行和分布式处理。它使IRIS对UNION查询执行并行处理，将每个查询分配给同一台机器上的单独进程。在某些情况下，该过程会将查询发送到另一台机器进行处理。这些进程通过管道进行通信， IRIS创建一个或多个临时文件来保存子查询结果。主进程组合结果行并返回最终结果。比较带和不带%Parallel关键字的Show Plan。要确定当前系统上的处理器数量，请使用%SYSTEM.Util.NumberOfCPU()方法。

通常，生成每一行所花费的精力越多，%Parallel就会变得越有利。

指定%PARALLEL关键字将禁用自动并或优化。

下面的例子展示了%PARALLEL关键字的用法:

```
SELECT Name FROM Sample.Employee WHERE Name %STARTSWITH 'A'
UNION %PARALLEL
SELECT Name FROM Sample.Person WHERE Name %STARTSWITH 'A'
ORDER BY Name
```

```
SELECT Name FROM Sample.Employee WHERE Name %STARTSWITH 'A'
UNION ALL %PARALLEL
SELECT Name FROM Sample.Person WHERE Name %STARTSWITH 'A'
ORDER BY Name
```

%PARALLEL用于SELECT查询及其子查询。
INSERT命令子查询不能使用%PARALLEL。

添加%PARALLEL关键字可能不适用于所有UNION查询，并可能导致错误。
以下SQL构造通常不支持UNION %PARALLEL执行:外部连接、相关字段、包含子查询的IN谓词条件或集合谓词。
for SOME谓词支持UNION %PARALLEL，但for SOME %ELEMENT集合谓词不支持UNION %PARALLEL。
要确定UNION查询是否能够成功使用%PARALLEL，请分别测试UNION的每个分支。
通过添加FROM %PARALLEL关键字分别测试每个分支查询。
如果其中一个FROM %PARALLEL查询生成的查询计划没有显示并行化，那么UNION查询将不支持%PARALLEL。

UNION ALL和聚合函数

SQL自动优化将UNION ALL聚合函数推入UNION分支子查询。
SQL计算每个子查询的聚合值，然后组合结果返回原始聚合值。
例如:

```
SELECT COUNT(Name) FROM (SELECT Name FROM Sample.Person
                    UNION ALL SELECT Name FROM Sample.Employee)
```

优化:

```
SELECT SUM(y) FROM (SELECT COUNT(Name) AS y FROM Sample.Person
                    UNION ALL SELECT COUNT(Name) AS y FROM Sample.Employee)
```

这可以带来实质性的性能改进。
无论是否使用%PARALLEL关键字，都将应用此优化。
该优化应用于多个聚合函数。

这种优化变换只在以下情况下发生:
- 外部查询FROM子句必须只包含一个UNION ALL语句。
- 外部查询不能包含WHERE子句或GROUP BY子句。
- 外部查询不能包含%VID(视图ID)字段。
- 聚合函数不能包含DISTINCT或%FOREACH关键字。
- 聚合函数不能嵌套。

示例

下面的示例创建一个结果，其中包含两个表中每个Name的一行;

如果在两个表中都找到Name，则创建两行。
当Name是雇员时，它列出办公地点，并将单词 “ office ” 连接为州，以及雇员的头衔。
当Name是一个人时，它列出主位置，将单词 “ home ” 连接为状态，并将<null>表示标题。
ORDER BY子句对结果进行操作;
合并的行按名称排序:

```
SELECT Name,Office_State||' office' AS State,Title
FROM Sample.Employee
UNION
SELECT Name,Home_State||' home',NULL
FROM Sample.Person
ORDER BY Name
```

下面两个示例展示了ALL关键字的效果。
在第一个示例中，UNION只返回惟一的值。
在第二个示例中，UNION ALL返回所有值，包括重复值:

```
SELECT Name
FROM Sample.Employee
WHERE Name %STARTSWITH 'A'
UNION
SELECT Name
FROM Sample.Person
WHERE Name %STARTSWITH 'A'
ORDER BY Name
```

```
SELECT Name
FROM Sample.Employee
WHERE Name %STARTSWITH 'A'
UNION ALL
SELECT Name
FROM Sample.Person
WHERE Name %STARTSWITH 'A'
ORDER BY Name
```

#SQL #Caché

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