文章
姚 鑫 · 四月 7, 2021 阅读大约需 12 分钟

第十九章 存储和使用流数据（BLOBs和CLOBs）

第十九章 存储和使用流数据（BLOBs和CLOBs）

Intersystems SQL支持将流数据存储为Intersystems Iris ® DataPlatform数据库中的 BLOBs（二进制大对象）或 CLOBs（字符大对象）的功能。

流字段和SQL

Intersystems SQL支持两种流字段：

- 字符流 Character streams，用于大量文本。
- 二进制流 Binary streams，用于图像，音频或视频。

BLOBs and CLOBs

Intersystems SQL支持将BLOBs（二进制大对象）和CLOBs（字符大对象）存储为流对象的功能。
BLOBs用于存储二进制信息，例如图像，而CLOBs用于存储字符信息。
BLOBs和CLOBs可以存储多达4千兆字节的数据（JDBC和ODBC规范所强加的限制）。

在各种方面，诸多方面的操作在通过ODBC或JDBC客户端访问时处理字符编码转换（例如Unicode到多字节）：BLOB中的数据被视为二进制数据，从未转换为二进制数据另一个编码，而CLOB中的数据被视为字符数据并根据需要转换。

如果二进制流文件（BLOB）包含单个非打印字符$CHAR(0)，则被认为是空二进制流。它相当于""空二进制流程值：它存在（不是null），但长度为0。

定义流数据字段

Intersystems SQL支持流字段的各种数据类型名称。这些Intersystems数据类型名称是与以下内容对应的同义词：

- 字符流：数据类型LONGVARCHAR，映射到%stream.globalcharacter类和ODBC / JDBC数据类型-1。
- 字符流：数据类型LONGVARBINARY，映射到%Stream.GlobalBinary类和ODBC / JDBC数据类型-4。

某些Intersystems流数据类型允许指定数据精度值。此值是no-op，对流数据的允许大小没有影响。提供它以允许用户记录预期的未来数据大小。

以下示例定义包含两个流字段的表：

```
CREATE TABLE Sample.MyTable (
    Name VARCHAR(50) NOT NULL,
    Notes LONGVARCHAR,
    Photo LONGVARBINARY)
```

分片表不能包含流数据类型字段。

流字段约束

Stream字段的定义符合以下字段数据约束：

流字段可以定义为 NOT NULL。

流字段可以占用默认值，更新值或计算码值。

流字段不能定义为唯一，主键字段或idkey。试图这样做导致SQLCode -400致命错误，其中%MSG如下:ERROR #5414: Invalid index attribute: Sample.MyTable::MYTABLEUNIQUE2::Notes, Stream property is not allowed in a unique/primary key/idkey index > ERROR #5030: An error occurred while compiling class 'Sample.MyTable'.

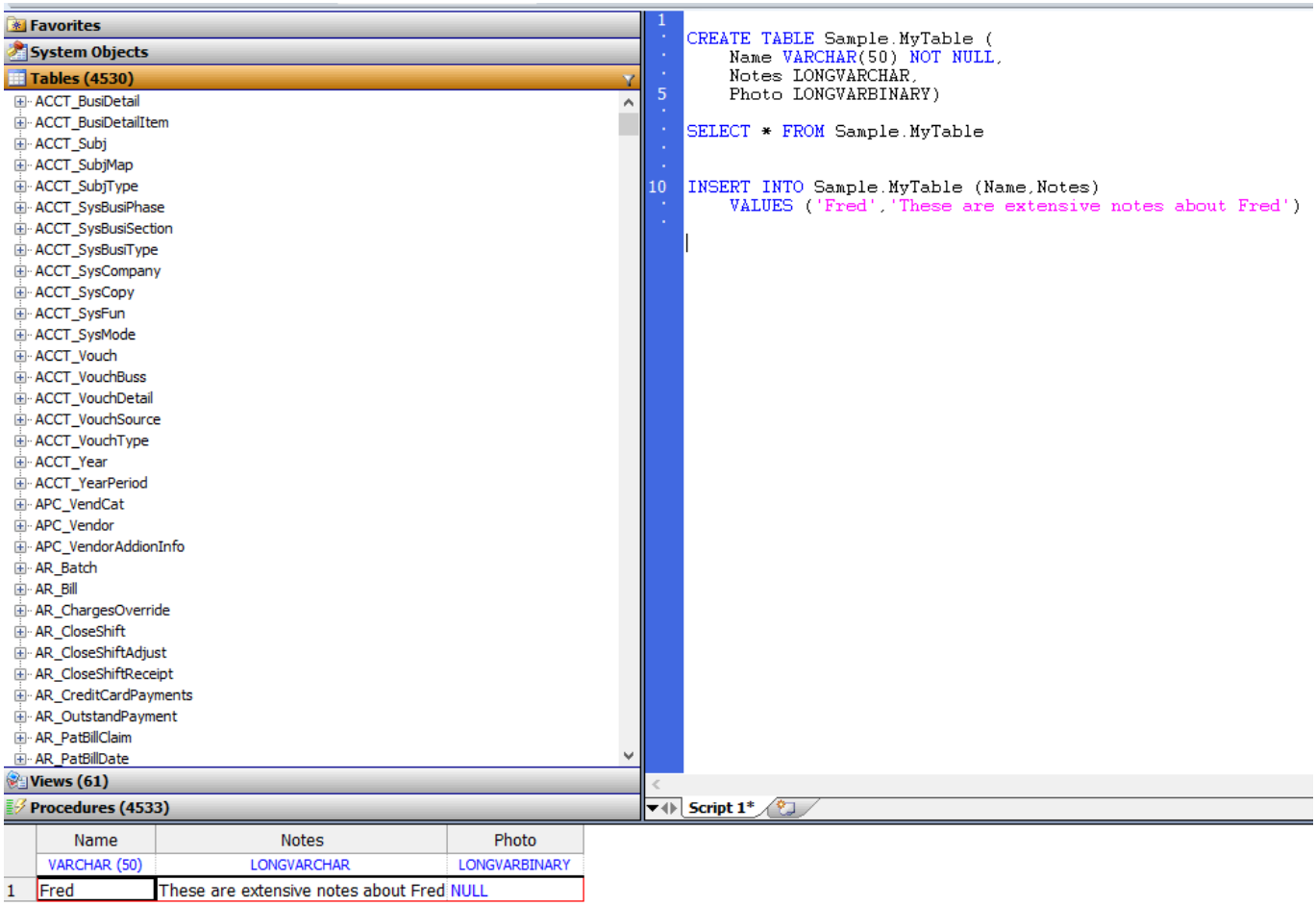
无法使用指定的COLLATE 值定义流字段。试图这样做导致SQLCode -400致命错误，其中%MSG如下：ERROR #5480: Property parameter not declared: Sample.MyTable:Photo:COLLATION > ERROR #5030: An error occurred while compiling class 'Sample.MyTable'.

将数据插入流数据字段

将数据插入流字段有三种方法：

- %Stream.Globalcharacter字段：可以直接插入字符流数据。例如，

```
INSERT INTO Sample.MyTable (Name,Notes)
    VALUES ('Fred','These are extensive notes about Fred')
```




```
Class Sample.MyTable Extends %Persistent [ ClassType = persistent, DdlAllowed, Final,
 Owner = {yx}, ProcedureBlock, SqlRowIdPrivate, SqlTableName = MyTable ]
{

Property Name As %Library.String(MAXLEN = 50) [ Required, SqlColumnNumber = 2 ];

Property Notes As %Stream.GlobalCharacter [ SqlColumnNumber = 3 ];

Property Photo As %Stream.GlobalBinary [ SqlColumnNumber = 4 ];

/// Bitmap Extent Index auto-
generated by DDL CREATE TABLE statement.  Do not edit the SqlName of this index.
Index DDLBEIndex [ Extent, SqlName = "%%DDLBEIndex", Type = bitmap ];

Storage Default
{
<Data name="MyTableDefaultData">
<Value name="1">
<Value>Name</Value>
</Value>
<Value name="2">
<Value>Notes</Value>
</Value>
<Value name="3">
<Value>Photo</Value>
</Value>
</Data>
<DataLocation>^Sample.MyTableD</DataLocation>
<DefaultData>MyTableDefaultData</DefaultData>
<IdFunction>sequence</IdFunction>
```

```
<IdLocation>^Sample.MyTableD</IdLocation>
<IndexLocation>^Sample.MyTableI</IndexLocation>
<StreamLocation>^Sample.MyTableS</StreamLocation>
<Type>%Library.CacheStorage</Type>
}

}
```

- %stream.globalcharacter和%stream.globalbinary字段：可以使用oref插入流数据。可以使用Write（）方法将字符串附加到字符流，或者写入的方法，以将具有行终结器的字符串附加到字符流。默认情况下，行终结器是$CHAR(13,10)（回车返回/线路）;可以通过设置LineTerminator 属性来更改行终结器。在以下示例中，示例的第一部分创建由两个字符串和其终端组组成的字符流，然后使用嵌入的SQL将其插入流字段。示例的第二部分返回字符流长度，并显示显示终结器的字符流数据：

```
/// d ##class(PHA.TEST.SQL).StreamField()
ClassMethod StreamField()
{
CreateAndInsertCharacterStream
    SET gcoref=##class(%Stream.GlobalCharacter).%New()
    DO gcoref.WriteLine("First Line")
    DO gcoref.WriteLine("Second Line")
    &sql(INSERT INTO Sample.MyTable (Name,Notes)
        VALUES ('Fred',:gcoref))
    IF SQLCODE<0 {
        WRITE "SQLCODE ERROR:"_SQLCODE_" "_%msg  QUIT
    } ELSE {
        WRITE "????",!
    }
DisplayTheCharacterStream
    KILL ^CacheStream
    WRITE gcoref.%Save(),!
    ZWRITE ^CacheStream
}


DHC-APP>d ##class(PHA.TEST.SQL).StreamField()
????
1
^CacheStream=1
^CacheStream(1)=1
^CacheStream(1,0)=25
^CacheStream(1,1)="First Line"_$c(13,10)_"Second Line"_$c(13,10)
```

- %stream.globalcharacter和%stream.globalbinary字段：可以通过从文件读取它来插入流数据。例如，

```
/// d ##class(PHA.TEST.SQL).StreamField1()
ClassMethod StreamField1()
{
    SET myf="E:\temp\game.jpg"
    OPEN myf:("RF"):10
    USE myf:0
    READ x(1):10
    &sql(INSERT INTO Sample.MyTable (Name,Photo) VALUES ('George',:x(1)))
    IF SQLCODE<0 {
        WRITE "SQLCODE ERROR:"_SQLCODE_" "_%msg  QUIT
    } ELSE {
        WRITE "????",!
```

```
    }
    CLOSE myf
}


DHC-APP>d ##class(PHA.TEST.SQL).StreamField1()

  WRITE "????",!
  ^
<WRITE>zStreamField1+11^PHA.TEST.SQL.1
DHC-APP 2d0>g

  WRITE "????",!
              ^
<WRITE>zStreamField1+11^PHA.TEST.SQL.1
DHC-APP 2d0>g

DHC-APP>
```

作为默认值或计算值插入的字符串数据以适合于流字段的格式存储。

查询流字段数据

选择流字段的查询选择项返回流对象的完全形成的OID（对象ID）值，如下例所示：

```
SELECT Name,Photo,Notes
FROM Sample.MyTable WHERE Photo IS NOT NULL
```

OID是一个 %List 格式化数据地址，如以下内容：$lb("1","%Stream.GlobalCharacter","^EW3K.Cn9X.S")。

- OID的第一个元素是一个连续的正整数(从1开始)，它被分配给每个插入到表中的流数据值。
  例如，如果第1行插入流字段Photo和Notes的值，则将它们赋值为1和2。
  如果第2行插入了一个Notes值，则将该值赋给3。
  如果用Photo和Notes的值插入第3行，则将它们赋值为4和5。
  分配顺序是表定义中列出字段的顺序，而不是INSERT命令中指定字段的顺序。
  默认情况下，使用单个整数序列，它对应于流位置全局计数器。
  然而，一个表可能有多个流计数器，如下所述。
- 更新操作不会改变初始整数值。
  DELETE操作可以在整型序列中创建空白，但不会改变这些整型值。
  使用DELETE删除所有记录不会重置此整数计数器。
  如果所有表流字段都使用默认的StreamLocation值，则使用TRUNCATE TABLE删除所有记录将重置此整数计数器。
  不能使用TRUNCATE表为嵌入式对象(%SerialObject)类重置流整数计数器。
- OID的第二个元素是流数据类型，可以是%Stream.GlobalCharacter 或%Stream.GlobalBinary。
- OID的第三个元素是一个全局变量。
  默认情况下，它的名称是从与表对应的包名和持久类名生成的。
  一个“ S ”(用于流)被追加。
  - 如果表是使用SQL CREATE

TABLE命令创建的，这些包和持久化类名称将被散列为每个4个字符(例如，^EW3K.Cn9X.S)。
这个全局变量包含流数据插入计数器最近分配的值。
如果没有插入流字段数据，或者使用TRUNCATE
TABLE删除所有表数据，那么这个全局变量是未定义的。

- 如果表是作为一个持久化类创建的，那么这些包和持久化类名不会被散列(例如^Sample.MyTableS)。默认情况下，这是StreamLocation存储关键字<StreamLocation>^Sample.MyTableS</StreamLocation> 价值。

默认流位置是全局位置，如^Sample.MyTableS。此全局变量用于计算插入到没有自定义位置的所有流属性(字段)的次数。例如，如果Sample.MyTable中的所有流属性都使用默认流位置，则在Sample.MyTable的流属性中插入了10个流数据值时，^Sample.MyTableS全局变量包含值10。此全局变量包含最近分配的流数据插入计数器的值。如果没有插入流字段数据，或者使用截断表删除了所有表数据，则此全局变量未定义。

定义流字段属性时，可以定义自定义位置，如下所示：Property Note2 As %Stream.GlobalCharacter (LOCATION="^MyCustomGlobalS")；。在这种情况下，^MyCustomGlobalS全局用作指定此位置的流属性(或多个属性)的流数据插入计数器；未指定位置的流属性使用默认流位置全局(^Sample.MyTableS)作为流数据插入计数器。每个全局计数与该位置相关联的流属性的插入。如果没有插入流场数据，则位置GLOBAL是未定义的。如果一个或多个流属性定义了位置，则截断表不重置流计数器。

这些流位置全局变量的下标包含每个流字段的数据。例如，^EW3K.Cn9X.S(3)表示第三个插入的流数据项。^EW3K.Cn9X.S(3，0)是数据的长度。^EW3K.Cn9X.S(3，1)是实际的流数据值。

注意：流字段的OID与RowID或Reference字段返回的OID不同。%OID函数返回RowID或引用字段的OID；%OID不能与流字段一起使用。试图将流字段用作%OID的参数会导致SQLCODE-37错误。

在查询的WHERE子句或HAVING子句中使用流字段受到严格限制。不能将相等条件或其他关系运算符(=, !=, <, >)或包含运算符(])或跟随运算符([)与流字段一起使用。尝试将这些运算符与流字段一起使用会导致SQLCODE-313错误。

Result Set Display

- 从程序执行的动态SQL以$lb("6","%Stream.GlobalCharacter","^EW3K.Cn9X.S").格式返回OID。
- SQL Shell作为动态SQL执行，并以$lb("6","%Stream.GlobalCharacter","^EW3K.Cn9X.S")格式返回OID。
- 嵌入式SQL返回相同的OID，但以编码%LIST的形式返回。可以使用$LISTTOSTRING函数将OID显示为元素以逗号分隔的字符串：6，%Stream.GlobalBinary，^EW3K.Cn9X.S。

从管理门户SQL执行界面运行查询时，不返回OID。取而代之的是：
- 字符流字段返回字符流数据的前100个字符。如果字符流数据超过100个字符，则用省略号(...)表示。在第100个字符之后。这等效于SUBSTRING(cstream field，1,100)。
- 二进制流字段返回字符串<binary>。

在表数据的管理门户SQL界面打开表显示中显示相同的值。

要从管理门户SQL执行界面显示OID值，请将空字符串连接到流值，如下所示：SELECT Name, ''¦¦Photo, ''¦¦Notes FROM Sample.MyTable。

DISTINCT, GROUP BY, and ORDER BY

每个流数据字段的OID值是唯一的，即使数据本身包含重复。
这些SELECT子句操作的是流的OID值，而不是数据值。
因此，当应用到查询中的流字段时:
- 不同的子句对重复的流数据值没有影响。
DISTINCT子句将流字段为NULL的记录数减少为一个NULL记录。
- GROUP BY子句对重复的流数据值没有影响。
GROUP BY子句将流字段为空的记录数量减少为一个空记录。
- ORDER BY子句根据数据流的OID值来排序数据，而不是数据值。
ORDER BY子句列出流字段为空的记录，然后列出带有流字段数据值的记录。

谓词条件和流

IS [NOT] NULL谓词可以应用于流字段的数据值，示例如下:

```
SELECT Name,Notes
FROM Sample.MyTable WHERE Notes IS NOT NULL
```

BETWEEN, EXISTS, IN, %INLIST, LIKE, %MATCHES, and %PATTERN
谓词可以应用于流对象的OID值，示例如下:

```
SELECT Name,Notes
FROM Sample.MyTable WHERE Notes %MATCHES '*1[0-9]*GlobalChar*'
```

尝试在流字段上使用任何其他谓词条件会导致SQLCODE -313错误。

聚合函数和流

COUNT聚合函数接受一个流字段，并对该字段中包含非空值的行进行计数，示例如下:

```
SELECT COUNT(Photo) AS PicRows,COUNT(Notes) AS NoteRows
FROM Sample.MyTable
```

但是，流字段不支持COUNT(DISTINCT)。

对于流字段不支持其他聚合函数。
尝试将流字段与任何其他聚合函数一起使用会导致SQLCODE -37错误。

标量函数和流

除了%OBJECT、CHARACTERLENGTH(或CHARLENGTH或DATALENGTH)、SUBSTRING、CONVERT、XMLCONCAT、XMLELEMENT、XMLFOREST和%INTERNAL函数外，InterSystems SQL不能对流字段应用任何函数。
尝试使用流字段作为任何其他SQL函数的参数会导致SQLCODE -37错误。

尝试使用流字段作为任何其他SQL函数的参数会导致SQLCODE -37错误。

- %OBJECT函数打开一个流对象(接受一个OID)并返回oref(对象引用)，示例如下:

```
SELECT Name,Notes,%OBJECT(Notes) AS NotesOref
FROM Sample.MyTable WHERE Notes IS NOT NULL
```

- CHARACTERLENGTH、CHARLENGTH和DATALENGTH函数接受流字段并返回实际的数据长度，如下面的示例所示:

```
SELECT Name,DATALENGTH(Notes) AS NotesNumChars,DATALENGTH(Photo) AS PhotoNumChars
FROM Sample.MyTable
```

SUBSTRING函数接受一个流字段，并返回流字段的实际数据值的指定子字符串，如下面的示例所示:

```
SELECT Name,SUBSTRING(Notes,1,10) AS Notes1st10Chars
FROM Sample.MyTable WHERE Notes IS NOT NULL
```

当从管理门户SQL Execute接口发出时，子字符串函数返回流字段数据最多100个字符的子字符串。
如果流数据的指定子字符串大于100个字符，则在第100个字符后用省略号(…)表示。

- CONVERT函数可用于将流数据类型转换为VARCHAR，示例如下:

```
SELECT Name,CONVERT(VARCHAR(100),Notes) AS NotesTextAsStr
FROM Sample.MyTable WHERE Notes IS NOT NULL
```

CONVERT(datatype,expression)语法支持流数据转换。
如果VARCHAR精度小于实际流数据的长度，则将返回值截断为VARCHAR精度。
如果VARCHAR精度大于实际流数据的长度，则返回值为实际流数据的长度。
不执行填充。

{fn CONVERT(expression,datatype)}语法不支持流数据转换;
它发出一个SQLCODE -37错误。

- %INTERNAL函数可以用于流字段，但不执行任何操作。

流字段并发锁

InterSystems IRIS通过取出流数据上的锁来保护流数据值不被另一个进程并发操作。

InterSystems IRIS在执行写操作之前取出一个排他锁。
排他锁在写操作完成后立即释放。

当第一个读操作发生时，InterSystems IRIS取出共享锁。
只有当流实际被读取时才会获取共享锁，并且在整个流从磁盘读取到内部临时输入缓冲区后立即释放共享锁。

在Intersystems中使用流字段IRIS方法

不能在Intersystems Iris方法中直接使用嵌入式SQL或动态SQL使用BLOB或CLOB值;相反，使用SQL来查找Blob或Clob的流标识符，然后创建%AbstractStream对象的实例以访问数据。

使用来自ODBC的流字段

ODBC规范不提供对BLOB和CLOB字段的任何识别或特殊处理。
InterSystems SQL将ODBC中的CLOB字段表示为具有LONGVARCHAR(-1)类型。
BLOB字段表示为类型为LONGVARBINARY(-4)。
对于流数据类型的ODBC/JDBC数据类型映射，请参考InterSystems SQL
reference中的数据类型引用页中的数据类型整数代码。

ODBC驱动程序/服务器使用一种特殊协议来访问BLOB和CLOB字段。
通常，必须在ODBC应用程序中编写特殊的代码来使用CLOB和BLOB字段;
标准的报告工具通常不支持它们。

使用来自JDBC的流字段

在Java程序中，可以使用标准的JDBC BLOB和CLOB接口从BLOB或CLOB检索或设置数据。
例如:

```
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery("SELECT MyCLOB,MyBLOB FROM MyTable");
rs.next();      // fetch the Blob/Clob

java.sql.Clob clob = rs.getClob(1);
java.sql.Blob blob = rs.getBlob(2);

// Length
System.out.println("Clob length = " + clob.length());
System.out.println("Blob length = " + blob.length());

// ...
```

注意:当使用BLOB或CLOB结束时，必须显式调用free()方法来关闭Java中的对象，并向服务器发送消息以释放流资源(对象和锁)。
仅仅让Java对象超出范围并不会发送清理服务器资源的消息。

#SQL #Caché #InterSystems IRIS #InterSystems IRIS for Health

源URL:
https://cn.community.intersystems.com/post/%E7%AC%AC%E5%8D%81%E4%B9%9D%E7%AB%A0-%E5%AD%98%E5%82%A8%E5%92%8C%E4%BD%BF%E7%94%A8%E6%B5%81%E6%95%B0%E6%8D%AE%EF%BC%88blobs%E5%92%8Cclobs%EF%BC%89