
文章

[Jingwei Wang](#) · 七月 28, 2022 阅读大约需 8 分钟

InterSystems SQL 的使用 - 第八部分 - 存储和使用流数据 (BLOBs和CLOBs)

InterSystems SQL支持在InterSystems IRIS数据平台数据库中将流数据存储为BLOB (Binary Large Objects 二进制大对象) 或CLOB (Character Large Objects字符大对象) 的能力。

InterSystems SQL支持两种流字段：

- 字符流：用于大量的文本。
- 二进制流：用于图像、音频或视频。

BLOBs和CLOBs可以存储多达4GB的数据 (JDBC和ODBC规范规定的限制)。除了在通过ODBC或JDBC客户端访问时如何处理字符编码转换 (如Unicode到多字节) 外，BLOB和CLOB的操作在各方面都是相同的：BLOB中的数据被视为二进制数据，决不转换为其他编码，而CLOB中的数据被视为字符数据，在必要时进行转换。

如果一个二进制流文件 (BLOB) 包含单一的非打印字符\$CHAR(0)，它被认为是一个空的二进制流。它相当于""空二进制流值：它存在 (不是空的)，但长度为0。

从对象的角度来看，BLOB和CLOBs被表示为流对象。更多信息请参见定义和使用类中的 "与流合作 "一章。

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的数据类型名称是对应于以下的同义词。

- 字符流：数据类型LONGVARCHAR，映射到%Stream.GlobalCharacter类和ODBC/JDBC数据类型-1。
- 二进制流：数据类型LONGVARBINARY，映射到%Stream.GlobalBinary类和ODBC/JDBC数据类型-4。

下面的例子定义了一个包含两个流字段的表。

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

流字段的限制：

- 一个流字段可以被定义为NOT NULL。
- 一个流字段可以取一个DEFAULT值，一个ON UPDATE值，或者一个COMPUTECODE值。
- 一个流字段不能被定义为UNIQUE，一个主键字段，或一个IdKey。试图这样做会导致一个SQLCODE -400的致命错误，并带有%msg，如以下。ERROR #5414:
无效的索引属性。Sample.MyTable::MYTABLEUNIQUE2::Notes,
Stream属性在唯一/主键/IdKey索引中是不允许的 > ERROR #5030: 在编译'Sample.MyTable'类时发生错误。

- 不能用指定的COLLATE值定义一个流字段。试图这样做会导致一个SQLCODE -400的致命错误，并带有%msg，如下。ERROR #5480:
属性参数没有声明。Sample.MyTable:Photo:COLLATION > ERROR #5030:
在编译'Sample.MyTable'类时发生错误。

在流数据字段中插入数据

有三种方法可以将数据插入到流字段。

- %Stream.GlobalCharacter字段：你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')
```

- %Stream.GlobalCharacter和%Stream.GlobalBinary字段：你可以使用OREF插入流数据。你可以使用Write()方法将一个字符串追加到字符流中，或者使用WriteLine()方法将一个带有行终止符的字符串追加到字符流中。默认情况下，行结束符是\$CHAR(13,10)（回车/换行）；你可以通过设置LineTerminator属性改变行结束符。在下面的例子中，第一部分创建了一个由两个字符串和它们的终止符组成的字符流，然后使用嵌入式SQL将其插入到一个流字段中。例子的第二部分返回字符流的长度并显示字符流数据，显示终止符。

```
?
ClassMethod 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 "Insert successful",!}

do ..DisplayTheCharacterStream(gcoref)
}
?
ClassMethod DisplayTheCharacterStream(gcoref As %Stream.GlobalCharacter)
{
KILL ^CacheStream
WRITE gcoref.%Save(),!
ZWRITE ^CacheStream
}
}
```

- %Stream.GlobalCharacter和%Stream.GlobalBinary字段：你可以通过从文件中读取数据来插入流数据。比如说

```
ClassMethod InsertDataFromImage()
{
Set myf = "C:\Temps\IMG.png"
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 "INSERT Failed:"_SQLCODE_" "%msg QUIT}
?
CLOSE myf
}
```

作为DEFAULT值或计算值插入的字符串数据将以适合于流字段的格式存储。

查询流字段数据

二进制流字段返回字符串<binary>。

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

DISTINCT, GROUP BY, 和 ORDER BY

每个流数据字段的OID值都是唯一的，即使数据本身包含重复的内容。这些SELECT子句对流的OID值进行操作，而不是数据值。因此，当应用于查询中的流字段时。

- DISTINCT子句对重复的流数据值没有影响。DISTINCT子句将流字段为NULL的所有记录减少到一个NULL记录。DISTINCT对流字段的OID进行操作，而不是它的实际数据。
- 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NULL的所有记录数减少到一个NULL记录。GROUP BY StreamField的操作对象是一个流字段的OID，而不是它的实际数据。
- ORDER BY子句根据流数据值的OID值，而不是数据值来排序。ORDER BY子句在列出有流字段数据值的记录之前，先列出流字段为NULL的记录。

predicate **条件和流**

- IS [NOT] NULL 可以应用于流字段的数据值，如下面的例子中所示。

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

- BETWEEN, EXISTS, IN, %INLIST, LIKE, %MATCHES, 和 %PATTERN谓词可以应用于流对象的OID值，如下面的例子所示。

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

试图在一个流字段上使用任何其他的predicate条件会导致SQLCODE -313错误。

聚合函数和流

COUNT聚合函数接收一个流字段，并对包含该字段非空值的记录进行计数，如下面的例子所示：

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

然而，COUNT(DISTINCT)不支持流字段。对流字段不支持其他聚合函数。试图用任何其他聚合函数来使用流字段会导致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不能将任何函数应用到流字段。试图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)，如以下例子所示:

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

- CHARACTERLENGTH, CHARLENGTH和DATALENGTH函数取一个流字段，并返回实际的数据长度，如下面的例子所示。

```
SELECT Name,DATALENGTH(Notes) AS NotesNumChars
FROM Sample.MyTable WHERE Notes IS NOT NULL
```

- SUBSTRING函数接收一个流字段，并返回流字段实际数据值的指定子串，如下面的例子所示。

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

当从管理门户的SQL执行界面发出时，SUBSTRING函数最多返回流字段数据的100个字符的子串。如果指定的流数据子串长于100个字符，会在第100个字符后面用省略号 (...) 表示。

- CONVERT函数可以用来将流数据类型转换为VARCHAR，如下面的例子所示。

```
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，它将返回值截断为VARCHAR精度。如果VARCHAR精度大于实际流数据的长度，返回值就有实际流数据的长度。不进行填充。

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

- %INTERNAL函数可以在流字段上使用，但不执行任何操作。

流字段并发锁定

InterSystems IRIS通过在流数据上加锁来保护流数据值不受另一个进程的并发操作。

InterSystems IRIS在执行写操作之前会拿出一个独占锁。写操作完成后，独占锁会立即释放。

InterSystems IRIS在第一次读操作发生时取出一个共享锁。只有在实际读取流时才会获得共享锁，并且在整个流从磁盘读入内部临时输入缓冲区后立即释放。

在InterSystems IRIS方法中使用流字段

你不能在InterSystems IRIS方法中直接使用嵌入式SQL或动态SQL来使用BLOB或CLOB值；而是要使用SQL来找到BLOB或CLOB的流标识符，然后创建%AbstractStream对象的实例来访问数据。

从ODBC使用流字段

ODBC规范没有为BLOB和CLOB字段提供任何识别或特殊处理。

InterSystems SQL在ODBC中表示CLOB字段为LONGVARCHAR (-1)类型。BLOB字段被表示为LONGVARBINARY类型 (-4)。

从JDBC使用流字段

在一个Java程序中，你可以使用标准的JDBC BLOB和CLOB接口从BLOB或CLOB中检索或设置数据。比如说

```
Statement st = conn.createStatement();
ResultSet rs = st.executeQuery("SELECT MyCLOB,MyBLOB FROM MyTable");
rs.next(); // ??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:当完成对BLOB或CLOB的处理时,你必须明确地调用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)。

[#SQL](#) [#InterSystems](#) IRIS for Health

源

URL:<https://cn.community.intersystems.com/post/intersystems-sql-%E7%9A%A8%E4%BD%BF%E7%94%A8-%E7%AC%AC%E5%85%AB%E9%83%A8%E5%88%86-%E5%AD%98%E5%82%A8%E5%92%8C%E4%BD%BF%E7%94%A8%E6%B5%81%E6%95%B0%E6%8D%AE%EFF%BC%88plobs%E5%92%8Cclobs%EFF%BC%89>