

Re: Java vs SQL Server float datatype limits

Source:

<http://www.tech-archive.net/Archive/SQL-Server/microsoft.public.sqlserver.jdbcdriver/2007-09/msg00000.html>

- *From:* DGA <XXXdalbertsonYYY@xxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sat, 01 Sep 2007 22:23:24 -0400
-

Joe Weinstein said (on or about) 06/28/2007 12:59:

isyourfriend@xxxxxxxx wrote:

On Jun 28, 11:52 am, Joe Weinstein <joeNOS...@xxxxxxx> wrote:

isyourfri...@xxxxxxxx wrote:

SQL Server float limits are : - 1.79E+308 to
-2.23E-308, 0 and
2.23E-308 to 1.79E+308
However Java double has a much wider
range : 4.94065645841247E-324
-4.94065645841247E-324

There is no "apparent" problem on using
java limit values but if the
table is checked for data purity it will of
course produce errors. BCP
also complains when using native format.

Is there a setting to force the driver to limit
the values for SQL
Server to those supported on SQL 92 ?

Many thanks,

Re: Java vs SQL Server float datatype limits

–Noel (DBA)

No, and the driver doesn't know you're trying to stuff a Java double into a float column. Java does have a float too, which corresponds correctly to the DBMS column. It should be up to

the application author to match data to target column.

However,

I ran this code, and it seems to work:

```
System.out.println("Driver version is " +
c.getMetaData().getDriverVersion() );
System.out.println("DBMS version is " +
c.getMetaData().getDatabaseProductVersion() );
```

```
Statement s = c.createStatement();
try{s.executeUpdate("drop procedure
joe_proc");}catch(Exception e) { }
s.executeUpdate("create table #foo(joe float)");
PreparedStatement p = c.prepareStatement("insert #foo
values(?)");
double dd = Double.MAX_VALUE / 2;
double ddd = Double.MIN_VALUE * 2;
```

```
p.setDouble(1, dd);
p.execute();
p.setDouble(1, ddd);
p.execute();
```

```
ResultSet r = s.executeQuery("select * from #foo");
while (r.next()) System.out.println("sent " + dd + " return " +
r.getDouble(1) );
```

I get:

Driver version is 1.1.1320.0

DBMS version is 8.00.818

sent 8.988465674311579E307 return

8.988465674311579E307

sent 1.0E-323 return 1.0E-323

Joe Weinstein at BEA Systems

Joe, as you can see 1.0E-323 is *out of range* for the float definition on sql server which by the way float = float(53) = double precision

Re: Java vs SQL Server float datatype limits

It is read and written correctly from the query engine but BCP chokes and DBCC CHECKTABLE('yourtablename') WITH DATA_PURITY complains!!!

So I would have imagined that an SQL Server driver either handle the case or it doesn't but that some things work and some others don't is not nice ;)

Thanks for your reply.

-Noel

Hi, sure. I'm not clear on how the JDBC driver is involved. It seems to work with that code, so if you can show me an alteration of that code that shows a problem, that would be good. Regarding BCP or DBCC, those are DBMS issues. It seems the driver is faithfully correctly transferring those values, and the DBMS is returning them without corruption in this case. What case do you want the driver to handle?
Joe

The JDBC specification says something about the driver being expected to coerce the Java data type into the correct data type for the server the driver is representing. This works reasonably well in the case of the `java.sql.ResultSet` methods `getDouble(colNum/colName)`, `getDate(col)`, `getString(col)` etc. Also if you have a `ResultSet` which is updateable and you use `setDouble()` etc. If the Java native value of a double is greater than the native type that the database can handle, the driver would be expected to bitch, whine, piss, moan, throw exceptions, or otherwise let you know that as a programmer you have made an invalid assumption about the database.

I would expect similar behaviour from `PreparedStatement`, `CallableStatement`, etc. based on the declared type of the parameter for the SQL-side function/procedure.