Database - Bug #4421

blob/clob access with H2

11/22/2019 09:39 AM - Constantin Asofiei

Status:	WIP	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Constantin Asofiei	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 11/22/2019 09:45 AM - Constantin Asofiei

The Hibernate dialect has a method named useInputStreamToInsertBlob - the implementation for H2 returns true, while for PostgreSQL returns false.

For this reason, and as the H2 docs http://www.h2database.com/html/advanced.html state:

> To store a BLOB, use PreparedStatement.setBinaryStream. To store a CLOB, use PreparedStatement.setCharacterStream. To read a BLOB, use ResultSet.getBinaryStream, and to read a CLOB, use ResultSet.getCharacterStream

We can't use the setClob/getClob and setBlob/getBlob for H2. We need to use the stream versions. And PostgreSQL needs the setter/getter versions.

BlobUserType and ClobUserType need to be changed, so that the LOB access is done in a dialect-specific way; but I can't find any way to access the dialect, having access to this method signature,
(PreparedStatement ps, Object value, int index, SessionImplementor session)

Eric: I know this is related to Hibernate, which we will replace, but do you have any idea how I can know the dialect, while executing code in bindNonNull or nullSafeGet methods? If things get complicated, maybe is safe to postpone this and work on it after we remove Hibernate...

#2 - 11/22/2019 01:43 PM - Constantin Asofiei

For reference, this is the kind of error H2 throws, when using the LOB setters:

```
Caused by: org.h2.jdbc.JdbcSQLException: IO Exception: "Missing lob entry: 1" [90028-197]
        at org.h2.message.DbException.getJdbcSQLException(DbException.java:357)
        at org.h2.message.DbException.get(DbException.java:179)
        at org.h2.message.DbException.get(DbException.java:155)
        at org.h2.store.LobStorageBackend$LobInputStream.<init>(LobStorageBackend.java:656)
```

```
        at org.h2.store.LobStorageBackend.getInputStream(LobStorageBackend.java:340)
        at org.h2.value.ValueLobDb.getInputStream(ValueLobDb.java:397)
        at org.h2.value.ValueLobDb.getReader(ValueLobDb.java:377)
        at org.h2.jdbc.JdbcClob.getSubString(JdbcClob.java:192)
        at com.goldencode.p2j.util.clob.readData(clob.java:217)
        at com.goldencode.p2j.util.clob.<init>(clob.java:118)
        at com.goldencode.p2j.persist.type.ClobUserType.nullSafeGet(ClobUserType.java:172)
        at org.hibernate.type.CustomType.nullSafeGet(CustomType.java:124)
        at org.hibernate.type.AbstractType.hydrate(AbstractType.java:106)
        at org.hibernate.persister.entity.AbstractEntityPersister.hydrate(AbstractEntityPersister.java:2873)
        at org.hibernate.loader.Loader.loadFromResultSet(Loader.java:1574)
        at org.hibernate.loader.Loader.instanceNotYetLoaded(Loader.java:1506)
        at org.hibernate.loader.Loader.getRow(Loader.java:1406)
        at org.hibernate.loader.Loader.getRowFromResultSet(Loader.java:664)
        at org.hibernate.loader.Loader.doQuery(Loader.java:874)
        at org.hibernate.loader.Loader.doQueryAndInitializeNonLazyCollections(Loader.java:293)
        at org.hibernate.loader.Loader.doQueryAndInitializeNonLazyCollections(Loader.java:263)
        at org.hibernate.loader.Loader.loadEntity(Loader.java:2006)
        at org.hibernate.loader.entity.AbstractEntityLoader.load(AbstractEntityLoader.java:82)
        at org.hibernate.loader.entity.AbstractEntityLoader.load(AbstractEntityLoader.java:72)
        at org.hibernate.persister.entity.AbstractEntityPersister.load(AbstractEntityPersister.java:3887)
        at org.hibernate.event.internal.DefaultLoadEventListener.loadFromDatasource(DefaultLoadEventListener.j
ava:458)
        at org.hibernate.event.internal.DefaultLoadEventListener.doLoad(DefaultLoadEventListener.java:427)
        at org.hibernate.event.internal.DefaultLoadEventListener.load(DefaultLoadEventListener.java:204)
        at org.hibernate.event.internal.DefaultLoadEventListener.proxyOrLoad(DefaultLoadEventListener.java:260
)
        at org.hibernate.event.internal.DefaultLoadEventListener.onLoad(DefaultLoadEventListener.java:148)
        at org.hibernate.internal.SessionImpl.fireLoad(SessionImpl.java:1078)
        at org.hibernate.internal.SessionImpl.access$2000(SessionImpl.java:175)
        at org.hibernate.internal.SessionImpl$IdentifierLoadAccessImpl.load(SessionImpl.java:2424)
        at org.hibernate.internal.SessionImpl.get(SessionImpl.java:978)
        at com.goldencode.p2j.persist.dirty.DefaultDirtyShareManager.notifyInsertValidated(DefaultDirtyShareMa
nager.java:432)
        at com.goldencode.p2j.persist.dirty.DefaultDirtyShareContext.insert(DefaultDirtyShareContext.java:335)
        at com.goldencode.p2j.persist.RecordBuffer.flush(RecordBuffer.java:6051)
        at com.goldencode.p2j.persist.RecordBuffer$ValidationHelper.validateDirty(RecordBuffer.java:12039)
        at com.goldencode.p2j.persist.RecordBuffer.endBatch(RecordBuffer.java:2999)
        at com.goldencode.p2j.persist.RecordBuffer.batch(RecordBuffer.java:2902)
```

#3 - 11/22/2019 02:44 PM - Ovidiu Maxiniuc

Constantin,
As a temporary workaround until a better solution is found, try to use session.connection().getMetaData().getDatabaseProductName(), session.connection().getMetaData().getURL() or session.connection().getMetaData().getDriverName(). If the first is not giving you the name of the driver (like H2), you will need to parse a string like: jdbc:h2:mem:fwd or H2 JDBC Driver.

#4 - 11/22/2019 03:39 PM - Constantin Asofiei

Ovidiu, your solution is working, but I need one more criteria, to distinguish between session-local tables and physical tables (aka _temp vs dirty). This is because _temp uses 'temporary tables' which are local to the H2 session, and in this case (for some reason) H2 doesn't want the streams, wants the old setters/getters...

#5 - 11/22/2019 03:53 PM - Constantin Asofiei

Even after I added the :_temp exclusion (so that only dirty and meta will use the stream lob APIs), it still fails. I need to build some tests and check with a H2 physical database.

#6 - 11/22/2019 05:03 PM - Eric Faulhaber

Constantin Asofiei wrote:

> BlobUserType and ClobUserType need to be changed, so that the LOB access is done in a dialect-specific way; but I can't find any way to access the dialect, having access to this method signature,
> (PreparedStatement ps, Object value, int index, SessionImplementor session)

You can invoke session.getInterceptor(), which will return an instance of ChangeBroker$SessionInterceptor. You will have to cast it. You can modify that class to expose the dialect from within the Persistence instance it holds. This should be much faster than querying and parsing database metadata.

> Eric: I know this is related to Hibernate, which we will replace, but do you have any idea how I can know the dialect, while executing code in bindNonNull or nullSafeGet methods? If things get complicated, maybe is safe to postpone this and work on it after we remove Hibernate...

If this is something urgent to fix, you can use the above mechanism. We are still weeks away from having a working FWD without Hibernate.

#7 - 11/24/2019 08:42 AM - Constantin Asofiei

Eric Faulhaber wrote:

> You can invoke session.getInterceptor(), which will return an instance of ChangeBroker$SessionInterceptor. You will have to cast it. You can modify that class to expose the dialect from within the Persistence instance it holds. This should be much faster than querying and parsing database metadata.

Using the ChangeBroker$SessionInterceptor works for runtime, but not for import. Looks like H2 and SQLServer are true, while PostgreSQL is false.

#8 - 11/24/2019 12:32 PM - Constantin Asofiei

Eric, my conclusion is this: LOB in page-store is broken, only in MVStore is working - if I remove the mv_store=false;mvcc=false (so it remains the default, which is MVStore, right?) from H2Helper.setCommonInMemoryProperties, then the abend is seen, but only if the LOB size exceeds the H2's BLOCK_LENGTH (20000 bytes).

Also, it seems that both the stream and getter/setter approaches are working in H2 MVStore, when creating a LOB.

So, beside the sortNullsHigh, we need to fix the LOB in H2's page-store, too.

#9 - 11/25/2019 04:35 AM - Constantin Asofiei

Constantin Asofiei wrote:

> ... then the abend is seen, but only if the LOB size exceeds the H2's BLOCK_LENGTH (20000 bytes).

Here I meant that the abend is seen in page-store only if the lob size exceeds 20000 bytes. In MV store, the abend is not seen.

#10 - 12/02/2019 09:21 AM - Constantin Asofiei

H2 1.4.200 with page-store still has the LOB bugs.