

Derby 目視確認

Issue Fix なし

345775

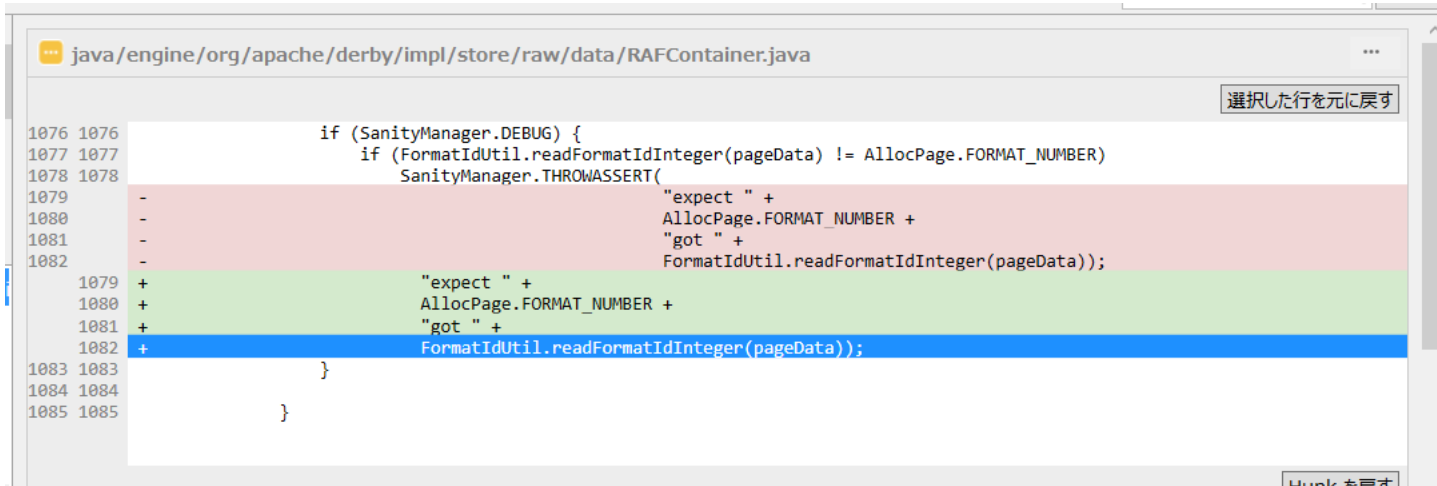
- ファイル名 :
java/engine/org/apache/derby/impl/store/raw/data/RAFContainer.java
- created : 25f99f59e6268a1db9af0469b2b1a675d656da5f
 - コミットコメント : オンラインバックアップをサポートするためのパッチ (DERBY-239)
- changed : 7739ad68f511089a94b75545c01e1e9a1d1fc225
 - コミットコメント : 以前 (DERBY-239) のオンラインバックアップパッチ (online_backup1.diff) を確認しながらOysteinが見つけた問題を修正しました。コンテナコードのバックアップでは、バックアップファイルではなくファイルコンテナでのシークが正しく行われませんでした。
- deleted : bf643fd0693dc0aeaabbaab709aede76e882f19f
 - コミットコメント : このパッチは、最初のオンラインバックアップパッチ (onlinebackup_1.diff) のレビューでOysteinが提案した改善点に対応し、さらにオンラインバックアップに関するいくつかの問題を解決しました。

345775: created

```
1. if (pageNumber == FIRST_ALLOC_PAGE_NUMBER)
2. {
3.     // write header into the alloc page array regardless of dirty
4.     // bit because the alloc page have zero'ed out the borrowed
5.     // space
6.     writeHeader(pageData);
7.
8.     if (SanityManager.DEBUG) {
9.         if (FormatIdUtil.readFormatIdInteger(pageData) != AllocPage.FORMAT_NUMBER)
10.             SanityManager.THROWASSERT(
11.                 "expect " +
12.                 AllocPage.FORMAT_NUMBER +
13.                 "got " +
14.                 FormatIdUtil.readFormatIdInteger(pageData));
15.     }
16.
17.}

1. if (pageNumber == FIRST_ALLOC_PAGE_NUMBER)
2. {
3.     // write header into the alloc page array regardless of dirty
4.     // bit because the alloc page have zero'ed out the borrowed
5.     // space
6.     writeHeader(pageData);
7.
8.     if (SanityManager.DEBUG) {
9.         if (FormatIdUtil.readFormatIdInteger(pageData) != AllocPage.FORMAT_NUMBER)
10.             SanityManager.THROWASSERT(
11.                 "expect " +
12.                 AllocPage.FORMAT_NUMBER +
13.                 "got " +
14.                 FormatIdUtil.readFormatIdInteger(pageData));
15.     }
16.
17.}
```

345775 : changed



```
java/engine/org/apache/derby/impl/store/raw/data/RAContainer.java

1076 1076         if (SanityManager.DEBUG) {
1077 1077             if (FormatIdUtil.readFormatIdInteger(pageData) != AllocPage.FORMAT_NUMBER)
1078 1078                 SanityManager.THROWASSERT(
1079 -                 "expect " +
1080 -                 AllocPage.FORMAT_NUMBER +
1081 -                 "got " +
1082 -                 FormatIdUtil.readFormatIdInteger(pageData));
1079 +                 "expect " +
1080 +                 AllocPage.FORMAT_NUMBER +
1081 +                 "got " +
1082 +                 FormatIdUtil.readFormatIdInteger(pageData));
1083 1083         }
1084 1084     }
1085 1085 }
```

選択した行を元に戻す

リンクを開く

4121139

- ファイル名 :

java/engine/org/apache/derby/impl/store/raw/RawStore.java

java/engine/org/apache/derby/impl/store/raw/log/LogToFile.java

- created : 393f243f0caa7584ce4fb5c5c9b309a5ddc0d5f6

- コミットコメント: DERBY-5232 stun READMEファイルをlogディレクトリとseg0ディレクトリに置いて、そこにファイルに触れた場合に起こる破損をユーザに警告する。Readmeファイルの作成が特権ブロック内で行われていなかったため、以前のコミットはテストの失敗を引き起こしました。このコミットはその問題を修正し、typoを修正し、finallyブロックで既に発生しているので、冗長なファイルクローズを取り除きます。

- co-changed : 1bb2c59c7f142255c0f0f2a57dc2ab69753e8f7b

- コミットコメント: DERBY-5232 (logとseg0ディレクトリにスティーンREADMEファイルを置くと、そのファイルにアクセスした場合に生じる犠牲をユーザーに警告します)
- このコミットの修正
- a) readmeファイルの内容の入力ミス。
- b) OutputStreamWrtierを使用して、readmeファイルのUTF-8エンコーディングを処理します。
- c) readmeファイルの作成に使用されるif条件を修正する

4121139: created

java/engine/org/apache/derby/impl/store/raw/RawStore.java

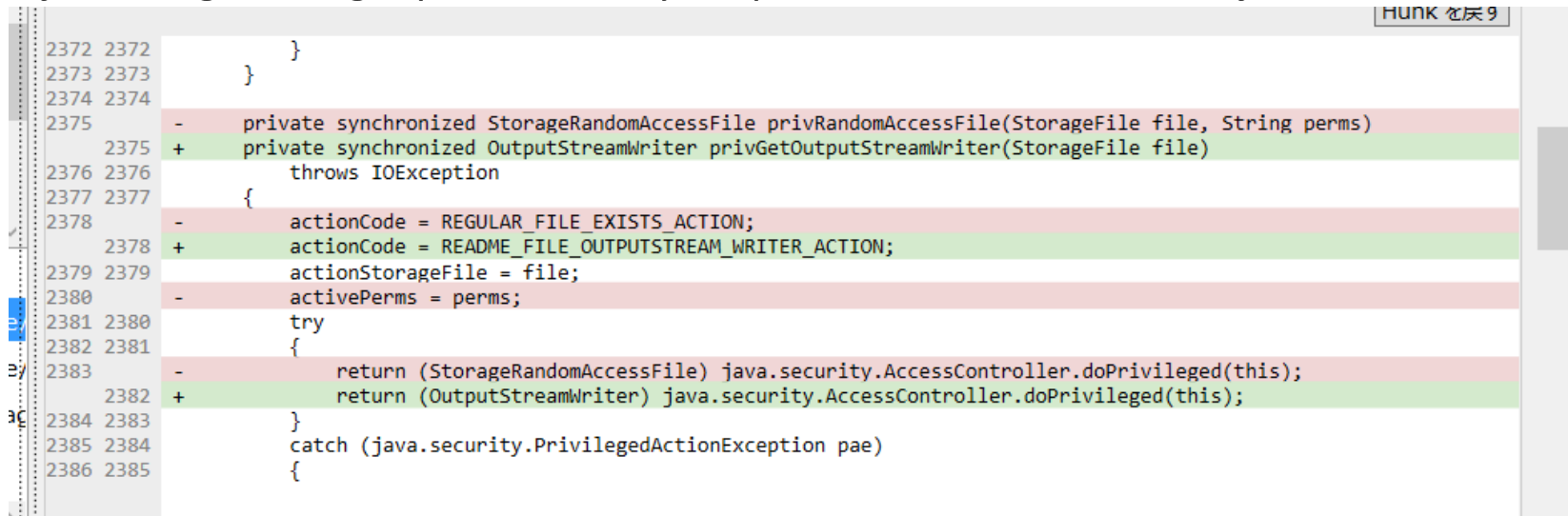
```
1. private synchronized StorageRandomAccessFile privRandomAccessFile(StorageFile file, String perms)
2.     throws IOException
3.     {
4.         actionCode = REGULAR_FILE_EXISTS_ACTION;
5.         actionStorageFile = file;
6.         activePerms = perms;
7.         try
8.         {
9.             return (StorageRandomAccessFile) java.security.AccessController.doPrivileged(this);
10.        }
11.        catch (java.security.PrivilegedActionException pae)
12.        {
13.            throw (IOException) pae.getException();
14.        }
15.    }
```

java/engine/org/apache/derby/impl/store/raw/log/LogToFile.java

```
1. private synchronized StorageRandomAccessFile privRandomAccessFile(StorageFile file, String perms)
2.     throws IOException
3.     {
4.         action = 2;
5.         activeFile = file;
6.         activePerms = perms;
7.         try
8.         {
9.             return (StorageRandomAccessFile) java.security.AccessController.doPrivileged(this);
10.        }
11.        catch (java.security.PrivilegedActionException pae)
12.        {
13.            throw (IOException) pae.getException();
14.        }
15.    }
```

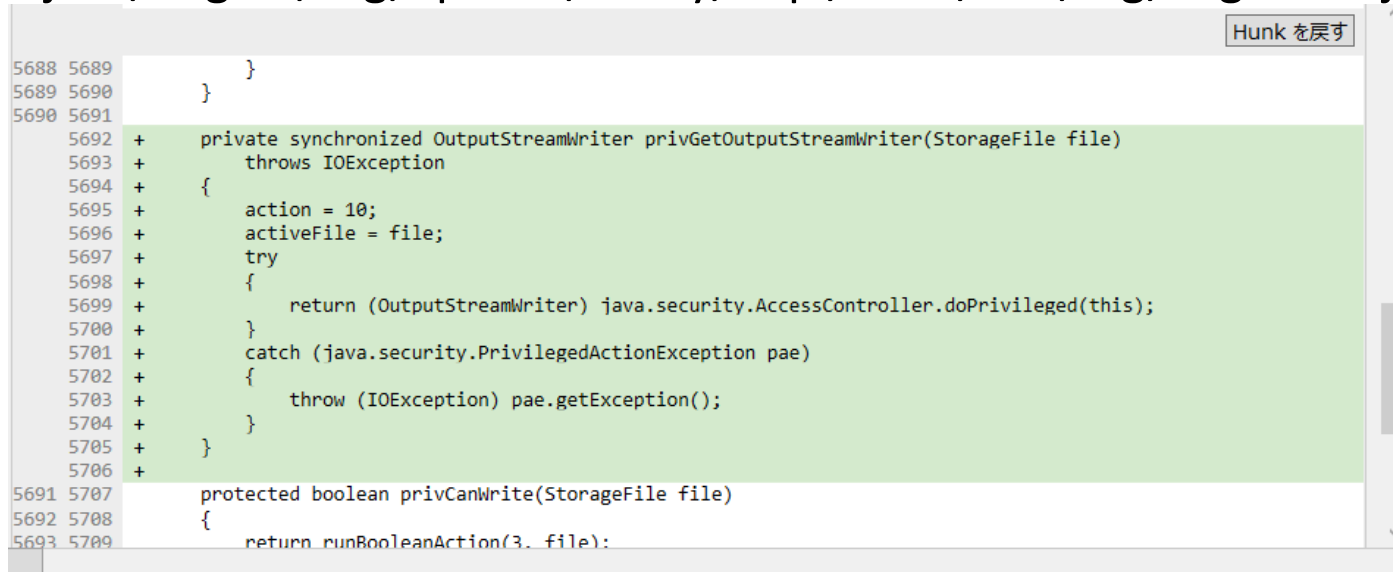
4121139:co-changed

- java/engine/org/apache/derby/impl/store/raw/RawStore.java



```
2372 2372     }
2373 2373     }
2374 2374
2375 - private synchronized StorageRandomAccessFile privRandomAccessFile(StorageFile file, String perms)
2375 + private synchronized OutputStreamWriter privGetOutputStreamWriter(StorageFile file)
2376 2376     throws IOException
2377 2377     {
2378 -     actionCode = REGULAR_FILE_EXISTS_ACTION;
2378 +     actionCode = README_FILE_OUTPUTSTREAM_WRITER_ACTION;
2379 2379     actionStorageFile = file;
2380 -     activePerms = perms;
2381 2380     try
2382 2381     {
2383 -     return (StorageRandomAccessFile) java.security.AccessController.doPrivileged(this);
2382 +     return (OutputStreamWriter) java.security.AccessController.doPrivileged(this);
2384 2383     }
2385 2384     catch (java.security.PrivilegedActionException pae)
2386 2385     {
```

- java/engine/org/apache/derby/impl/store/raw/log/LogToFile.java



```
5688 5689     }
5689 5690     }
5690 5691
5692 + private synchronized OutputStreamWriter privGetOutputStreamWriter(StorageFile file)
5693 +     throws IOException
5694 +     {
5695 +         action = 10;
5696 +         activeFile = file;
5697 +         try
5698 +         {
5699 +             return (OutputStreamWriter) java.security.AccessController.doPrivileged(this);
5700 +         }
5701 +         catch (java.security.PrivilegedActionException pae)
5702 +         {
5703 +             throw (IOException) pae.getException();
5704 +         }
5705 +     }
5706 +
5691 5707     protected boolean privCanWrite(StorageFile file)
5692 5708     {
5693 5709         return runBooleanAction(3, file);
```

3352090

- ファイル名 :
java/engine/org/apache/derby/impl/store/raw/xact/XactFactory.java
- created : 95e7b46943f64284752472e55efb29f9715c3986
 - コミットコメント : Initial code import
- co-changed : eac036999fab0e9f9509022d833a1c99cad1b415
 - コミットコメント : Set eol-style to native for all text files.
- delete : ceaf7dfd1aeec51b52713792391ec6a3b77f4a1e
 - コミットコメント : トランザクションの「タイプ」をカウントするのではなく、トランザクションがコミットされたときに、呼び出しコードがデフォルトのコミット同期動作を設定できるように、このインタフェースが変更されました。ID列に使用されるネストされたユーザー・トランザクションは、デフォルトでは同期化されず、残りのネストされたユーザー・トランザクションはデフォルトで同期化されます。他のタイプのトランザクションの動作は影響を受けてはなりません。ユーザー・トランザクションはデフォルトではまだ同期されており、内部とnttはデフォルトでは同期していません。

3352090: created

```
1.      public RawTransaction startTransaction(  
2.      RawStoreFactory rsf,  
3.      ContextManager cm,  
4.      String transName)  
5.      throws StandardException  
6.      {  
7.          return(startCommonTransaction(  
8.              rsf, cm, false, null, USER_CONTEXT_ID, transName, true));  
9.      }
```

```
1.      public RawTransaction startNestedUpdateUserTransaction(  
2.      RawStoreFactory rsf,  
3.      ContextManager cm,  
4.      String      transName)  
5.      throws StandardException  
6.      {  
7.          return(startCommonTransaction(  
8.              rsf, cm, false, null,  
9.              NESTED_UPDATE_USER_CONTEXT_ID, transName, true));  
10.     }
```

3352090:co-changed

- 全て変更

4121243

- ファイル名 :
java/engine/org/apache/derby/impl/store/access/RAMTransaction.java
- created : 95e7b46943f64284752472e55efb29f9715c3986
 - コミットコメント : Initial code import
- co-changed : eac036999fab0e9f9509022d833a1c99cad1b415
 - コミットコメント : Set eol-style to native for all text files.
- co-changed : b71ac67f1e03fd18e2d73276320808b0d54a2689
 - コミットコメント : DERBY-2149: RAMTransactionのVectorsとHashtablesをArrayListsとHashMapsに置き換える
- co-changed : 7e7a589002905adf5d945daef7b8fd7bd8dbc5bd
 - コミットコメント : DERBY-6213: impl / storeパッケージを生成します。
- co-changed : ce40a31daf0c8c7c2d8a7059e5c0a6d01d3ed440
 - コミットコメント : DERBY-6856: DerbyがJDK 9ビルド116でコンパイルされている場合、廃止予定の警告を修正します。derby-6856-04-ab-autoboxingDeprecationWarnings-part2.diffをコミットします。
 - DERBY-6856 : Make it possible to build Derby using JDK 9

4121243:created

openSort メソッド

```
1. if (sorts == null || id >= sorts.size()
2.     || (sort = ((Sort) sorts.elementAt((int) id))) == null)
3. {
4.     throw StandardException.newException(
5.         SQLState.AM_NO_SUCH_SORT, new Long(id));
6. }
```

openSortScan メソッド

```
1. if (sorts == null || id >= sorts.size()
2.     || (sort = ((Sort) sorts.elementAt((int) id))) == null)
3. {
4.     throw StandardException.newException(
5.         SQLState.AM_NO_SUCH_SORT, new Long(id));
6. }
```

openSortRowSource メソッド

```
1. if (sorts == null || id >= sorts.size()
2.     || (sort = ((Sort) sorts.elementAt((int) id))) == null)
3. {
4.     throw StandardException.newException(
5.         SQLState.AM_NO_SUCH_SORT, new Long(id));
6. }
```

4121243

- co-changed : b71ac67f1e03fd18e2d73276320808b0d54a2689

openSort メソッド

```
1855 1852 // Find the sort in the sorts list, throw an error
1856 1853 // if it doesn't exist.
1857 1854 if (sorts == null || id >= sorts.size()
1858 - || (sort = ((Sort) sorts.elementAt((int) id))) == null)
1855 + || (sort = ((Sort) sorts.get((int) id))) == null)
1859 1856 {
1860 1857     throw StandardException.newException(
1861 1858         SQLState.AM_NO_SUCH_SORT, new Long(id));
```

openSortScan メソッド

```
1925 1922 // Find the sort in the sorts list, throw an error
1926 1923 // if it doesn't exist.
1927 1924 if (sorts == null || id >= sorts.size()
1928 - || (sort = ((Sort) sorts.elementAt((int) id))) == null)
1925 + || (sort = ((Sort) sorts.get((int) id))) == null)
1929 1926 {
1930 1927     throw StandardException.newException(
1931 1928         SQLState.AM_NO_SUCH_SORT, new Long(id));
```

openSortRowSource メソッド

```
1952 1949 // Find the sort in the sorts list, throw an error
1953 1950 // if it doesn't exist.
1954 1951 if (sorts == null || id >= sorts.size()
1955 - || (sort = ((Sort) sorts.elementAt((int) id))) == null)
1952 + || (sort = ((Sort) sorts.get((int) id))) == null)
1956 1953 {
1957 1954     throw StandardException.newException(
1958 1955         SQLState.AM_NO_SUCH_SORT, new Long(id));
```

4121139

- co-changed : 7e7a589002905adf5d945daef7b8fd7bd8dbc5bd

openSort メソッド

```
1822 1819 // Find the sort in the sorts list, throw an error
1823 1820 // if it doesn't exist.
1824 1821 if (sorts == null || id >= sorts.size()
1825 - || (sort = ((Sort) sorts.get((int) id))) == null)
1822 + || (sort = (sorts.get((int) id))) == null)
1826 1823 {
1827 1824     throw StandardException.newException(
1828 1825         SQLState.AM_NO_SUCH_SORT, new Long(id));
```

openSortRowSource メソッド

```
1918 1915 // Find the sort in the sorts list, throw an error
1919 1916 // if it doesn't exist.
1920 1917 if (sorts == null || id >= sorts.size()
1921 - || (sort = ((Sort) sorts.get((int) id))) == null)
1918 + || (sort = (sorts.get((int) id))) == null)
1922 1919 {
1923 1920     throw StandardException.newException(
1924 1921         SQLState.AM_NO_SUCH_SORT, new Long(id));
```

4121139

- co-changed : ce40a31daf0c8c7c2d8a7059e5c0a6d01d3ed440

openSort メソッド

```
1822 1822         || (sort = (sorts.get((int) id))) == null)
1823 1823     {
1824 1824         throw StandardException.newException(
1825 1825 -         SQLState.AM_NO_SUCH_SORT, new Long(id));
1825 1825 +         SQLState.AM_NO_SUCH_SORT, id);
1826 1826     }
```

openSortRowSource メソッド

```
1918 1918         || (sort = (sorts.get((int) id))) == null)
1919 1919     {
1920 1920         throw StandardException.newException(
1921 1921 -         SQLState.AM_NO_SUCH_SORT, new Long(id));
1921 1921 +         SQLState.AM_NO_SUCH_SORT, id);
1922 1922     }
```

119210

- ファイル名 :
java/engine/org/apache/derby/impl/store/access/heap/Heap.java
- created : bbc927cf0f76893a5c0e351bacd81e1847c8b1ac
 - コミットコメント: DERBY-132 (an inplace compressを追加する)に対処するためのチェックインの第1段階です。
- delete : 965be700e2859b4ad642d59ded025cdb02e69b58
 - DERBY-132, inplace compress. Add btree truncate support. Enhance OnlineCompressTest.java test.

119210

- created : bbc927cf0f76893a5c0e351bacd81e1847c8b1ac

purgeConglomerate メソッド

```
1. if (open_for_ddl_lock.init(
2.     (ContainerHandle) null,
3.     this,
4.     this.format_ids,
5.     xact_manager,
6.     rawtran,
7.     false,
8.     TransactionController.OPENMODE_FORUPDATE,
9.     TransactionController.MODE_RECORD,
10.    null,
11.    null) == null)
12.{
13.    throw StandardException.newException(
14.        SQLState.HEAP_CONTAINER_NOT_FOUND,
15.        new Long(id.getContainerId()));
16.}
```

compressConglomerate メソッド

```
1. if (open_conglom.init(
2.     (ContainerHandle) null,
3.     this,
4.     this.format_ids,
5.     xact_manager,
6.     rawtran,
7.     false,
8.     TransactionController.OPENMODE_FORUPDATE,
9.     TransactionController.MODE_RECORD,
10.    null,
11.    null) == null)
12.{
13.    throw StandardException.newException(
14.        SQLState.HEAP_CONTAINER_NOT_FOUND,
15.        new Long(id.getContainerId()));
16.}
```

119210

- delete : 965be700e2859b4ad642d59ded025cdb02e69b58

purgeConglomerate メソッド

```
757 757         this,
758 758         this.format_ids,
759 759         nested_xact,
760 760         - rawtran,
761 761         + nested_xact.getRawStoreXact(),
762 762         true,
763 763         TransactionController.OPENMODE_FORUPDATE,
764 764         TransactionController.MODE_RECORD,
765 765         - null,
766 766         + nested_xact.getRawStoreXact().newLockingPolicy(
767 767             + LockingPolicy.MODE_RECORD,
768 768             + TransactionController.ISOLATION_REPEATABLE_READ, true),
769 769         null) == null)
770 770     {
771 771         throw StandardException.newException(
```

compressConglomerate メソッド

```
843 853         false,
844 854         TransactionController.OPENMODE_FORUPDATE,
845 855         TransactionController.MODE_RECORD,
846 856         - null,
847 857         + rawtran.newLockingPolicy(
848 858             + LockingPolicy.MODE_RECORD,
849 859             + TransactionController.ISOLATION_REPEATABLE_READ, true),
860 860         null) == null)
861 861     {
862 862         throw StandardException.newException(
```

4121153

- ファイル名 :
java/engine/org/apache/derby/impl/store/access/btree/BTreeController.java
- created:0efe521c2aecaa41de78954d7a4769e5eeef680f
 - コミットコメント: Anurag Shekharによって提出. このコミットはderby-3330_followup_1_modified.diffのコミットです。
 - DERBY-3330: provide support for unique constraint over keys that include one or more nullable columns.

4121153

- created:0efe521c2aecaa41de78954d7a4769e5eeef680f

doIns メソッド

```
1. if (getConglomerate().isUniqueWithDuplicateNulls())
2. {
3.     int ret = compareLeftAndRightSiblings(rowToInsert,
4.         insert_slot, targetleaf);
5.     if (ret == MATCH_FOUND)
6.     {
7.         ret_val = ConglomerateController.ROWISDUPLICATE;
8.         break;
9.     }
10.    if (ret == RESCAN_REQUIRED)
11.        continue;
12. }
```

doIns メソッド

```
1. if (getConglomerate().isUniqueWithDuplicateNulls())
2. {
3.     int ret = compareLeftAndRightSiblings(rowToInsert,
4.         insert_slot, targetleaf);
5.     if (ret == MATCH_FOUND)
6.     {
7.         ret_val = ConglomerateController.ROWISDUPLICATE;
8.         break;
9.     }
10.    if (ret == RESCAN_REQUIRED)
11.        continue;
12. }
```

870373

- ファイル名 :
java/engine/org/apache/derby/impl/store/raw/data/RAFContainer.java
- created : 9008adea3f807cacd7fa89214f2a1309b7a2d75f
 - コミットコメント: DERBY-733, committed on behalf of Knut Anders Hatlen. 例外処理のためのMikeの懸案事項に対応する添付パッチ(DERBY-733-more-exception-handling.diff)。ロック時に問題が発生した場合、Derbyは古い動作に戻ります。
 - DERBY-733: Starvation in RAFContainer.readPage()
- delete : f0cd4bf3d362ab1e868da2a32afae15808ebc23b
 - コミットコメント: DERBY-801(部分)データファイルへの並列アクセスを許可する。DERBY-733用のパッチをもう必要としなくなり、コードがなくてもきれいになります。
 - DERBY-801: Allow parallel access to data files.

870373

- created : 9008adea3f807cacd7fa89214f2a1309b7a2d75f

readPage メソッド

```
1. if (hasJava5FairLocks) {
2.     try {
3.         lock.invoke(fairLock, null);
4.     } catch (Exception e) {
5.         // Something bad happened while trying to lock the
6.         // region. Since the locking is not required for
7.         // anything other than ensuring fairness, it is ok to
8.         // fall back to pre-1.5 behaviour.
9.         hasJava5FairLocks = false;
10.        if (SanityManager.DEBUG) {
11.            SanityManager.THROWASSERT(
12.                "failed invoking ReentrantLock.lock()", e);
13.        }
14.    }
15.}
```

readPage メソッド

```
1. if (hasJava5FairLocks) {
2.     try {
3.         unlock.invoke(fairLock, null);
4.     } catch (Exception e) {
5.         // An error occurred while unlocking the
6.         // region. The region might still be locked, so
7.         // we'd better stop using this kind of
8.         // locking. There will be no loss of
9.         // functionality, only a possible loss of
10.        // fairness.
11.        hasJava5FairLocks = false;
12.        if (SanityManager.DEBUG) {
13.            SanityManager.THROWASSERT(
14.                "failed invoking ReentrantLock.unlock()", e);
15.        }
16.    }
17.}
```

870373

- delete : f0cd4bf3d362ab1e868da2a32afae15808ebc23b

```
289 - // Use Java 1.5 fair locks if they are available.
290 - if (hasJava5FairLocks) {
291 -     try {
292 -         lock.invoke(fairLock, null);
293 -     } catch (Exception e) {
294 -         // Something bad happened while trying to lock the
295 -         // region. Since the locking is not required for
296 -         // anything other than ensuring fairness, it is ok to
297 -         // fall back to pre-1.5 behaviour.
298 -         hasJava5FairLocks = false;
299 -         if (SanityManager.DEBUG) {
300 -             SanityManager.THROWASSERT(
301 -                 "failed invoking ReentrantLock.lock()", e);
302 -         }
303 -     }
304 - }
```

```
316 - // Unlock this section.
317 - if (hasJava5FairLocks) {
318 -     try {
319 -         unlock.invoke(fairLock, null);
320 -     } catch (Exception e) {
321 -         // An error occurred while unlocking the
322 -         // region. The region might still be locked, so
323 -         // we'd better stop using this kind of
324 -         // locking. There will be no loss of
325 -         // functionality, only a possible loss of
326 -         // fairness.
327 -         hasJava5FairLocks = false;
328 -         if (SanityManager.DEBUG) {
329 -             SanityManager.THROWASSERT(
330 -                 "failed invoking ReentrantLock.unlock()", e);
331 -         }
332 -     }
333 - }
```

Issue Fix あり

3099225

- ファイル名 :
java/engine/org/apache/derby/impl/store/raw/data/RAFContainer4.java
- created : 1cfe34ee06688596e4b30087fb855948b537e40a
 - Issue(Bug) Fix : DERBY- 5185 シャットダウン時に
store/rollForwardRecovery.sqlが
RAFContainer4.recoverContainerAfterInterrupt()にスタックしました
 - コミットコメント: FILE_IO_INTERRUPTEDを投げたときの状態保守のバグ (threadDoingRestore / restoreChannelInProgress内) の状態を修正するパッチderby-5185-2b 最初のはすぐに問題を解決します。
- changed : b7f22c34f55ee6f42e38607e985ea481aa446775
 - Issue(Bug) Fix : DERBY-5325 InterruptResilienceTestの
ClosedChannelExceptionでチェックポイントが失敗する
 - コミットコメント: RAFContainer#cleanはすでに
InterruptDetectedExceptionを処理しており、この変更で
ClosedByInterruptException、AsynchronousCloseException、または
ClosedChannelExceptionが表示されなくなります。これはDERBY-5325
(このバグ)を解決するはずです。

3099225

- deleted : b0a89283484329288e432a3872b24af99b4a10bc
 - Issue(Bug) Fix : DERBY-5325 InterruptResilienceTestのClosedChannelExceptionでチェックポイントが失敗する
 - コミットコメント : Patch derby-5325-refactor-b, 冗長コードをリファクタし、いくつかのコメントとJavadocをクリーンアップします。

3099225 : created

readPage メソッド

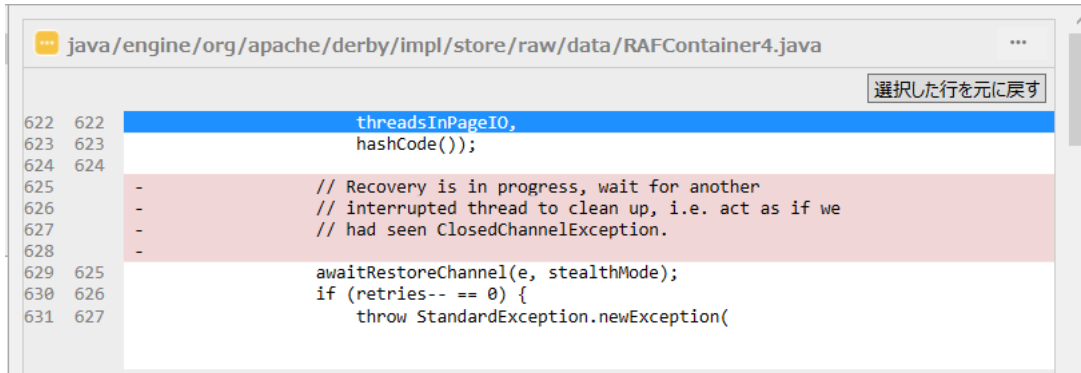
```
1. catch (ClosedChannelException e) {
2.         // We are not the thread that first saw the channel interrupt,
3.         // so no recovery attempt.
4.
5.         // if we also have seen an interrupt, we might as well take
6.         // notice now.
7.         InterruptStatus.noteAndClearInterrupt(
8.             "readPage in ClosedChannelException",
9.             threadsInPageIO,
10.            hashCode());
11.
12.        // Recovery is in progress, wait for another interrupted thread
13.        // to clean up.
14.        awaitRestoreChannel(e, stealthMode);
15.
16.        if (retries-- == 0) {
17.            throw StandardException.newException(
18.                SQLState.FILE_IO_INTERRUPTED);
19.        }
20.    }
```

writePage メソッド

```
1. catch (ClosedChannelException e) {
2.         // We are not the thread that first saw the channel interrupt,
3.         // so no recovery attempt.
4.
5.         InterruptStatus.noteAndClearInterrupt(
6.             "writePage in ClosedChannelException",
7.             threadsInPageIO,
8.            hashCode());
9.
10.        // Recovery is in progress, wait for another
11.        // interrupted thread to clean up, i.e. act as if we
12.        // had seen ClosedChannelException.
13.
14.        awaitRestoreChannel(e, stealthMode);
15.        if (retries-- == 0) {
16.            throw StandardException.newException(
17.                SQLState.FILE_IO_INTERRUPTED);
18.        }
19.    }
```

3099225:changed

writePage メソッド



```
java/engine/org/apache/derby/impl/store/raw/data/RAFContainer4.java
...
622 622 threadsInPageIO,
623 623 hashCode());
624 624
625 - // Recovery is in progress, wait for another
626 - // interrupted thread to clean up, i.e. act as if we
627 - // had seen ClosedChannelException.
628 -
629 625 awaitRestoreChannel(e, stealthMode);
630 626 if (retries-- == 0) {
631 627 throw StandardException.newException(
```

3099225

- deleted : b0a89283484329288e432a3872b24af99b4a10bc

readPage メソッド

```
401 374         } catch (ClosedChannelException e) {
402 -             // We are not the thread that first saw the channel interrupt,
403 -             // so no recovery attempt.
404 -
405 -             // if we also have seen an interrupt, we might as well take
406 -             // notice now.
407 -             InterruptStatus.noteAndClearInterrupt(
408 -                 "readPage in ClosedChannelException",
409 -                 threadsInPageIO,
410 -                 hashCode());
411 -
412 -             // Recovery is in progress, wait for another interrupted thread
413 -             // to clean up.
414 -             awaitRestoreChannel(e, stealthMode);
415 -
416 -             if (retries-- == 0) {
417 -                 throw StandardException.newException(
418 -                     SQLState.FILE_IO_INTERRUPTED);
419 -             }
420 375 +             handleClosedChannel(e, stealthMode, retries--);
420 376         }
```

writePage メソッド

```
616 546         } catch (ClosedChannelException e) {
617 -             // We are not the thread that first saw the channel interrupt,
618 -             // so no recovery attempt.
619 -
620 -             InterruptStatus.noteAndClearInterrupt(
621 -                 "writePage in ClosedChannelException",
622 -                 threadsInPageIO,
623 -                 hashCode());
624 -
625 -             awaitRestoreChannel(e, stealthMode);
626 -             if (retries-- == 0) {
627 -                 throw StandardException.newException(
628 -                     SQLState.FILE_IO_INTERRUPTED);
629 -             }
630 547 +             handleClosedChannel(e, stealthMode, retries--);
630 548         }
```

4121132

- ファイル名 :

java/engine/org/apache/derby/impl/store/raw/data/FileContainer.java

java/engine/org/apache/derby/impl/store/raw/data/RAFContainer.java

- created : bb8f25af4a4b5d601b194615c631f58ee4acb8cc

- コミットコメント : Patch derby-4741-b-04-nio : NIO RAFContainer4にコンテナの回復を追加します。古いテストDerby151Testは廃止され、新しいテストであるInterruptResilienceTestが追加されました。現在のところ、2スレッドのシナリオでは、基本的な復旧のケースのみをテストしています。テストを追加する必要があります。デバッグフラグ "RAF4" は、recovery eventsを監視するためにsaneビルドで使用できます。
- DERBY- 4741 (Bug) Fix: Make Derby work reliably in the presence of thread interrupts

4121132

- changed: 5d9742250f70ec93331188b37ed163babadef2d
 - コミットコメント: Patch derby-4741-sleeps-waits-3: Object #waitの中の割り込みと "org.apache.derby.impl.store.*"の中のThread #sleepを見るときすべての動作を変更します。Thewの新しいコードでは、割り込みが発生した後に再試行することに注意しています。古いコードは時々08000(CONN_INTERRUPT)を投げたか、割り込みだけを無視しました。
 - DERBY-4741 (Bug) Fix: Make Derby work reliably in the presence of thread interrupts
- co-changed: abf8151ced3b29509be8abb9360ef8fccb98c0f1
 - コミットコメント: パッチd5498bは、DirFile4 #getExclusiveFileLockの割り込みによるNIOチャネルクロージャの処理に穴を開ける
 - DERBY-5498 (Bug) Fix: ClosedByInterruptException in AuthenticationTest

4121132:created

- created:bb8f25af4a4b5d601b194615c631f58ee4acb8cc

java/engine/org/apache/derby/impl/store/raw/data/RAFContainer.java

```
clean(boolean forRemove) メソッド
1.  catch (InterruptDetectedException e) {
2.      if (--maxTries > 0) {
3.          success = false;
4.
5.          // Wait a bit so recovery can take place before
6.          // we re-grab monitor on "this" (which recovery
7.          // needs) and retry writeRAFHeader.
8.          try {
9.              Thread.sleep(500); // 0.5s
10.         } catch (InterruptedException ee) {
11.             // This thread received an interrupt as
12.             // well, make a note.
13.             InterruptStatus.setInterrupted();
14.         }
15.
16.         continue; // retry write of RAFHeader
17.     } else {
18.         // We have tried for a minute, not sure what's
19.         // going on, so to be on safe side we can't
20.         // continue
21.         throw StandardException.newException(
22.             SQLState.FILE_IO_INTERRUPTED, e);
23.     }
24. }
```

java/engine/org/apache/derby/impl/store/raw/data/FileContainer.java

```
pageValid(BaseContainerHandle handle, long pagenum) メソッド
1.  catch (InterruptDetectedException e) {
2.      // Retry. We needed to back all the way up here in the case
3.      // of the (file) container having been closed due to an
4.      // interrupt since the recovery needs the monitor to
5.      // allocCache
6.      if (--maxTries > 0) {
7.          done = false;
8.
9.          // Wait a bit so recovery can take place before
10.         // we re-grab monitor on "this" (which recovery
11.         // needs) and retry writeRAFHeader.
12.         try {
13.             Thread.sleep(INTERRUPT_RETRY_SLEEP);
14.         } catch (InterruptedException ee) {
15.             // This thread received an interrupt as
16.             // well, make a note.
17.             InterruptStatus.setInterrupted();
18.         }
19.
20.         continue;
21.     } else {
22.         throw StandardException.newException(
23.             SQLState.FILE_IO_INTERRUPTED, e);
24.     }
25. }
```

4121132: changed

- changed: 5d9742250f70ec93331188b37ed163babadefdf2d

java/engine/org/apache/derby/impl/store/raw/data/RAFContainer.java
clean(boolean forRemove) メソッド

```
570 570 // we re-grab monitor on "this" (which recovery
571 571 // needs) and retry writeRAFHeader.
572 572 try {
573 - Thread.sleep(500); // 0.5s
+ Thread.sleep(INTERRUPT_RETRY_SLEEP);
574 574 } catch (InterruptedException ee) {
575 575 // This thread received an interrupt as
576 576 // well, make a note.
```

4121132: co-changed

- co-changed: abf8151ced3b29509be8abb9360ef8fccb98c0f1

java/engine/org/apache/derby/impl/store/raw/data/RAFContainer.java
clean(boolean forRemove) メソッド

```
512 512 // we re-grab monitor on "this" (which recovery
513 513 // needs) and retry writeRAFHeader.
514 514 try {
515 - Thread.sleep(INTERRUPT_RETRY_SLEEP);
+ Thread.sleep(
516 + InterruptStatus.INTERRUPT_RETRY_SLEEP);
516 517 } catch (InterruptedException ee) {
517 518 // This thread received an interrupt as
518 519 // well, make a note.
```

java/engine/org/apache/derby/impl/store/raw/data/FileContainer.java
pageValid(BaseContainerHandle handle, long pagenum) メソッド

```
2095 2094 // we re-grab monitor on "this" (which recovery
2096 2095 // needs) and retry writeRAFHeader.
2097 2096 try {
2098 - Thread.sleep(INTERRUPT_RETRY_SLEEP);
+ Thread.sleep(InterruptStatus.INTERRUPT_RETRY_SLEEP);
2099 2098 } catch (InterruptedException ee) {
2100 2099 // This thread received an interrupt as
2101 2100 // well, make a note.
```

2387289

- ファイル名 :
java/engine/org/apache/derby/impl/store/access/btree/BTreeMaxScan.java
java/engine/org/apache/derby/impl/store/access/btree/BTreeScan.java
- created : 95e7b46943f64284752472e55efb29f9715c3986
 - コミットコメント : Initial code import
- co-changed : eac036999fab0e9f9509022d833a1c99cad1b415
 - コミットコメント : Set eol-style to native for all text files.

- delete : be4b84e8cb3a4846f2ea165cb1da271609dd3206
 - Issue(Bug) fix : DERBY-4193 ASSERT FAILED Scan position already saved with multi-threaded insert/update/delete
 - 添付されたScanPosSaved.javaプログラムは、以下のエラーを生成します。プログラムには3つのスレッドがあります。1つはID列を持つテーブルに挿入し、もう1つは最大IDで行を更新し、もう1つは最大IDで行を削除します。
 - コミットコメント: スキャンの終点で再配置する前に現在の位置を忘れてしまいます。その後、位置を再度保存する必要がある場合、ASSERTは混乱しません(位置が既に保存されていることが判明すると失敗します)。
BTreeScan.positionAtStartForBackwardScan()にもこの問題がありましたが、呼び出し元がないため、このメソッドは修正されずに削除されました。

2387289:created

java/engine/org/apache/derby/impl/store/access/btree/BTreeScan.java

```
1. if (init_startKeyValue == null)
2.     {
3.         // No start given, position at last slot + 1 of rightmost leaf
4.         pos.current_leaf = (LeafControlRow) root.searchRight(this);
5.
6.         pos.current_slot = pos.current_leaf.page.recordCount();
7.         exact          = false;
8.     }
```

java/engine/org/apache/derby/impl/store/access/btree/BTreeMaxScan.java

```
1. if (init_startKeyValue == null)
2.     {
3.         // No start given, position at last slot + 1 of rightmost leaf
4.         pos.current_leaf = (LeafControlRow) root.searchRight(this);
5.
6.         pos.current_slot = pos.current_leaf.page.recordCount();
7.         exact          = false;
8.     }
```

2387289:co-changed

- 全て変更

2387289:delete

- delete : be4b84e8cb3a4846f2ea165cb1da271609dd3206

java/engine/org/apache/derby/impl/store/access/btree/BTreeScan.java

```
519 -         if (init_startKeyValue == null)
520 -         {
521 -             // No start given, position at last slot + 1 of rightmost leaf
522 -             pos.current_leaf = (LeafControlRow) root.searchRight(this);
523 -
524 -             pos.current_slot = pos.current_leaf.page.recordCount();
525 -             exact = false;
526 -         }
```

4121146

- ファイル名 :
java/engine/org/apache/derby/impl/store/raw/log/LogToFile.java
- created : 95e7b46943f64284752472e55efb29f9715c3986
 - コミットコメント : Initial code import
- co-changed : eac036999fab0e9f9509022d833a1c99cad1b415
 - コミットコメント : Set eol-style to native for all text files.
- changed : 8fbfcc4995e1e2ffb1605d330d4bb829d96a98c6
 - コミットコメント : 可能な最大ログファイル数を222-1から233-1に増やすためのパッチが添付されています。
- co-changed : a0dbbd7285ba1ad5e560e5fc18c336c4fa5890dc
 - コミットコメント : DERBY-6648: ContextServiceのthe public static entry pointsをユーザコードが呼び出さないようにします。 derby-6648-01-ae-regressionTests.diffをコミットします。
 - DERBY-6648 (Bug) Fix: Application code should not be able to call ContextService.getContextOrNull()

4121146:created

- created : 95e7b46943f64284752472e55efb29f9715c3986

enableLogArchiveMode()メソッド

```
1. if (af != null)
2. {
3.     TransactionController tc = null;
4.     tc = af.getTransaction(ContextService.getFactory().getCurrentContextManager());
5.     tc.setProperty(Property.LOG_ARCHIVE_MODE , "true", true);
6. }
```

disableLogArchiveMode()メソッド

```
1. if (af != null)
2. {
3.     TransactionController tc = null;
4.     tc = af.getTransaction(ContextService.getFactory().getCurrentContextManager());
5.     tc.setProperty(Property.LOG_ARCHIVE_MODE , "false", true);
6. }
```

4121146 : co-changed

- co-changed : eac036999fab0e9f9509022d833a1c99cad1b415
- 全て変更

4121146: changed

- changed : 8fbfcc4995e1e2ffb1605d330d4bb829d96a98c6

enableLogArchiveMode()メソッド

```
4308 4333         if (af != null)
4309 4334         {
4310 4335             TransactionController tc = null;
4311         - tc = af.getTransaction(ContextService.getFactory().getCurrentContextMana
4336         + tc = af.getTransaction(
4337         +     ContextService.getFactory().getCurrentContextManager());
4312 4338             tc.setProperty(Property.LOG_ARCHIVE_MODE , "true", true);
4313 4339         }
4314 4340     }
```

4121146: co-changed

- co-changed : a0dbbd7285ba1ad5e560e5fc18c336c4fa5890dc

enableLogArchiveMode()メソッド

```
4956 4957         {
4957 4958             TransactionController tc = null;
4958 4959             tc = af.getTransaction(
4959         -             ContextService.getFactory().getCurrentContextManager());
4960         +             getContextService().getCurrentContextManager());
4960 4961             tc.setProperty(Property.LOG_ARCHIVE_MODE , "true", true);
4961 4962         }
4962 4963     }
```

disableLogArchiveMode()メソッド

```
4970 4971         if (af != null)
4971 4972         {
4972 4973             TransactionController tc = null;
4973         -             tc = af.getTransaction(ContextService.getFactory().getCurrentContextManager());
4974         +             tc = af.getTransaction(getContextService().getCurrentContextManager());
4974 4975             tc.setProperty(Property.LOG_ARCHIVE_MODE , "false", true);
4975 4976         }
4976 4977         logArchived = false;
```

4121110

- ファイル名 :

java/engine/org/apache/derby/impl/store/raw/log/LogToFile.java

- created : beac4f3997415eb8989163736b6a6cfd8b54b265

- コミットコメント : DERBY-4072 ログファイルが書き込み可能でない場合、エラーメッセージを改善します。
- DERBY-4072 (Bug) Fix : shutdown with incorrect permission on log files shows java.lang.NullPointerException at org.apache.derby.impl.store.raw.log.LogToFile.flush(LogToFile.java:3964). Should give better message.

4121110: created

- created: beac4f3997415eb8989163736b6a6cfd8b54b265

recover(DataFactory df, TransactionFactory tf) メソッド

```
1.  if (theLog == null || !privCanWrite(logFile))
2.  {
3.      if (theLog != null)
4.          theLog.close();
5.
6.      theLog = null;
7.      Monitor.logTextMessage(MessageId.LOG_CHANGED_DB_TO_READ_ONLY);
8.      if (accessException != null)
9.          Monitor.logThrowable(accessException);
10.     ReadOnlyDB = true;
11. }
```

recover(DataFactory df, TransactionFactory tf) メソッド

```
1.  if (theLog == null || !privCanWrite(logFile))
2.  {
3.      if (theLog != null)
4.          theLog.close();
5.      theLog = null;
6.      Monitor.logTextMessage(MessageId.LOG_CHANGED_DB_TO_READ_ONLY);
7.      if (accessException != null)
8.          Monitor.logThrowable(accessException);
9.      ReadOnlyDB = true;
10.
11. }
```

4113256

- ファイル名 :
java/engine/org/apache/derby/impl/store/access/btree/BTreeScan.java
- created:395bf0402c9324781d3d62db458a67146585810d
 - コミットコメント:AM_RECORD_NOT_FOUND例外の引数を修正しました。ページ番号の引数がありませんでした。
 - bug Issue Fix : DERBY-4690 Too few arguments when generating AM_RECORD_NOT_FOUND error
- delete:ce40a31daf0c8c7c2d8a7059e5c0a6d01d3ed440
 - コミットコメント:DERBY-6856:DerbyがJDK 9ビルド116でコンパイルされている場合、廃止予定警告を修正しました。derby-6856-04-ab-autoboxingDeprecationWarnings-part2.diffをコミットします。
 - DERBY-6856: Make it possible to build Derby using JDK 9

4113256

- created:395bf0402c9324781d3d62db458a67146585810d

delete() メソッド

```
1. if (!reposition(scan_position, false))
2.     {
3.         throw StandardException.newException(
4.             SQLState.AM_RECORD_NOT_FOUND,
5.             new Long(err_containerid),
6.             new Long(scan_position.current_rh.getPageNumber()),
7.             new Long(scan_position.current_rh.getId()));
8.     }
```

delete() メソッド

```
1. if (!reposition(scan_position, false))
2.     {
3.         throw StandardException.newException(
4.             SQLState.AM_RECORD_NOT_FOUND,
5.             new Long(err_containerid),
6.             new Long(scan_position.current_rh.getPageNumber()),
7.             new Long(scan_position.current_rh.getId()));
8.     }
```

fetch(DataValueDescriptor[] row, boolean qualify) メソッド

```
1. if (!reposition(scan_position, false))
2.     {
3.         // TODO - write unit test to get here, language always calls
4.         // isCurrentPositionDeleted() right before calling this, so
5.         // hard to write .sql test to exercise this.
6.
7.         throw StandardException.newException(
8.             SQLState.AM_RECORD_NOT_FOUND,
9.             new Long(err_containerid),
10.            new Long(scan_position.current_rh.getPageNumber()),
11.            new Long(scan_position.current_rh.getId()));
12.     }
```

4113256

- delete:ce40a31daf0c8c7c2d8a7059e5c0a6d01d3ed440

delete() メソッド

```
1120 1120      {
1121 1121          throw StandardException.newException(
1122 1122              SQLState.AM RECORD NOT FOUND,
1123 1123              - new Long(err_containerid),
1124 1124              - new Long(scan_position.current_rh.getPageNumber()),
1125 1125              - new Long(scan_position.current_rh.getId()));
1126 1126          + err_containerid,
1127 1127          + scan_position.current_rh.getPageNumber(),
1128 1128          + scan_position.current_rh.getId());
1129 1129      }
```

delete() メソッド

```
1162 1162      {
1163 1163          throw StandardException.newException(
1164 1164              SQLState.AM RECORD NOT FOUND,
1165 1165              - new Long(err_containerid),
1166 1166              - new Long(scan_position.current_rh.getPageNumber()),
1167 1167              - new Long(scan_position.current_rh.getId()));
1168 1168          + err_containerid,
1169 1169          + scan_position.current_rh.getPageNumber(),
1170 1170          + scan_position.current_rh.getId());
1171 1171      }
```

fetch(DataValueDescriptor[] row, boolean qualify) メソッド

```
1356 1356      throw StandardException.newException(
1357 1357          SQLState.AM RECORD NOT FOUND,
1358 1358          - new Long(err_containerid),
1359 1359          - new Long(scan_position.current_rh.getPageNumber()),
1360 1360          - new Long(scan_position.current_rh.getId()));
1361 1361          + err_containerid,
1362 1362          + scan_position.current_rh.getPageNumber(),
1363 1363          + scan_position.current_rh.getId());
1364 1364      }
```

まとめ

修正漏れの例

DERBY- 4741 (Bug) - reporter Dag H. Wanvik
: Make Derby work reliably in the presence of thread interrupts

2010年11月24日 11:00:32

bb8f25af4a4b5d601b194615c631f58ee4acb8cc

created : Dag H. Wanvik

```
2075 } catch (InterruptedException e) {  
2076     // Retry. We needed to back all the way up here in the case  
2077     // of the (file) container having been closed due to an  
2078     // interrupt since the recovery needs the monitor to  
2079     // allocCache  
2080     if (--maxTries > 0) {  
2081         done = false;  
2082  
2083         // Wait a bit so recovery can take place before  
2084         // we re-grab monitor on "this" (which recovery  
2085         // needs) and retry writeRAFHeader.  
2086         try {  
2087             Thread.sleep(INTERRUPT_RETRY_SLEEP);  
2088         } catch (InterruptedException ee) {  
2089             // This thread received an interrupt as  
2090             // well, make a note.  
2091             InterruptStatus.setInterrupted();  
2092         }  
2093         continue;  
2094     } else {  
2095         throw StandardException.newException(  
2096             SQLState.FILE_IO_INTERRUPTED, e);  
2097     }  
2098 }  
2099 }
```

```
564 } catch (InterruptedException e) {  
565     if (--maxTries > 0) {  
566         success = false;  
567  
568         // Wait a bit so recovery can take place before  
569         // we re-grab monitor on "this" (which recovery  
570         // needs) and retry writeRAFHeader.  
571         try {  
572             Thread.sleep(500); // 0.5s  
573         } catch (InterruptedException ee) {  
574             // This thread received an interrupt as  
575             // well, make a note.  
576             InterruptStatus.setInterrupted();  
577         }  
578         continue; // retry write of RAFHeader  
579     } else {  
580         // We have tried for a minute, not sure what's  
581         // going on, so to be on safe side we can't  
582         // continue  
583         throw StandardException.newException(  
584             SQLState.FILE_IO_INTERRUPTED, e);  
585     }  
586 } catch (IOException ioe) {  
587 }
```

2011年1月21日 6:11:19

5d9742250f70ec93331188b37ed163babadefdf2d

changed : Dag H. Wanvik

```
572 572 try {  
573 - Thread.sleep(500); // 0.5s  
573 + Thread.sleep(INTERRUPT_RETRY_SLEEP);  
574 574 } catch (InterruptedException ee) {
```



別のコードの修正によっておこる呼び出し方法の変更

DERBY-5498 (Bug) - reporter Knut Anders Hatlen
: ClosedByInterruptedException in AuthenticationTest

2011年11月12日 2:52:16

co-changed: Dag H. Wanvik

```
1676 1674      try {  
1677 -         Thread.sleep(INTERRUPT_RETRY_SLEEP);  
1675 +         Thread.sleep(  
1676 +             InterruptStatus.INTERRUPT_RETRY_SLEEP);  
1678 1677     } catch (InterruptedException ee) {
```

```
514 514      try {  
515 -         Thread.sleep(INTERRUPT_RETRY_SLEEP);  
515 +         Thread.sleep(  
516 +             InterruptStatus.INTERRUPT_RETRY_SLEEP);  
516 517     } catch (InterruptedException ee) {
```

Initial Commitから変更がなく、バグ修正のためクローンのコード片が削除される

2004年8月26日

1cfe34ee06688596e4b30087fb855948b537e40a

created : Dag H. Wanvik

java/engine/org/apache/derby/impl/store/access/btree/BTreeScan.java

```
538         if (init_startKeyValue == null) ↓
539         { ↓
540             // No start given, position at last slot + 1 of rightmost leaf ↓
541             pos.current_leaf = (LeafControlRow) root.searchRight(this); ↓
542         ↓
543             pos.current_slot = pos.current_leaf.page.recordCount(); ↓
544             exact          = false; ↓
545         } ↓
```

java/engine/org/apache/derby/impl/store/access/btree/BTreeMaxScan.java

```
303         if (init_startKeyValue == null) ↓
304         { ↓
305             // No start given, position at last slot + 1 of rightmost leaf ↓
306             pos.current_leaf = (LeafControlRow) root.searchRight(this); ↓
307         ↓
308             pos.current_slot = pos.current_leaf.page.recordCount(); ↓
309             exact          = false; ↓
310         } ↓
```

2009年5月6日

b0a89283484329288e432a3872b24af99b4a10bc

delete : Knut Anders Hatlen

```
519 -         if (init_startKeyValue == null)
520 -         {
521 -             // No start given, position at last slot + 1 of rightmost leaf
522 -             pos.current_leaf = (LeafControlRow) root.searchRight(this);
523 -
524 -             pos.current_slot = pos.current_leaf.page.recordCount();
525 -             exact          = false;
526 -         }
```

DERBY-4193 reporter Kathey Marsden

: ASSERT FAILED Scan position already saved with multi-threaded insert/update/delete

コミットコメント: スキャンの終点で再配置する前に現在の位置を忘れてしまいます。その後、位置を再度保存する必要がある場合、ASSERTは混乱しません(位置が既に保存されていることが判明すると失敗します)。

BTreeScan.positionAtStartForBackwardScan()にもこの問題がありましたが、呼び出し元がないため、このメソッドは修正されずに削除されました。

別のコードの修正によっておこる呼び出し方法の変更

co-changed : a0dbbd7285ba1ad5e560e5fc18c336c4fa5890dc

committer : Richard N. Hillegas

コミットコメント: DERBY-6648: ContextServiceのthe public static entry pointsをユーザコードが呼び出さないようにします。 derby-6648-01-ae-regressionTests.diffをコミットします。

ISSUE - DERBY-6648 (reporter) Rick Hillegas

: Application code should not be able to call ContextService.getContextOrNull()

enableLogArchiveMode()メソッド

```
4956 4957         {
4957 4958             TransactionController tc = null;
4958 4959             tc = af.getTransaction(
4959 -             ContextService.getFactory().getCurrentContextManager());
4960 +             getContextService().getCurrentContextManager());
4960 4961             tc.setProperty(Property.LOG_ARCHIVE_MODE , "true", true);
4961 4962         }
4962 4963     }
```

disableLogArchiveMode()メソッド

```
4970 4971         if (af != null)
4971 4972         {
4972 4973             TransactionController tc = null;
4973 -             tc = af.getTransaction(ContextService.getFactory().getCurrentContextManager())
4974 +             tc = af.getTransaction(getContextService().getCurrentContextManager());
4974 4975             tc.setProperty(Property.LOG_ARCHIVE_MODE , "false", true);
4975 4976         }
4976 4977         logArchived = false;
```