Archived:Using Symbian SQL API for attaching and detaching databases

Archived: This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Overview

This code snippet shows how to attach a database to another database. Attaching a database means making it appear to be part of the primary database. This gives an opportunity to temporarily view multiple databases as if they were a single database.

The `RSqlDatabase::Attach()` method requires two parameters. The first one is the name of the database to be attached and the second one is a logical name to be assigned to the database. If both databases have tables with the same name, the logical database name allows to explicitly scope the right one.

The `RSqlDatabase::Detach()` method requires the logical name of the database as a parameter, because it is possible to attach multiple databases to one primary database. `RSqlDatabase::Detach()` needs the information of which database to detach.

This snippet can be self-signed.

MMP file

The following libraries are required:

```
LIBRARY euser.lib
LIBRARY sqldb.lib
```

The following capabilities are needed to test the secure database in this example:

```
CAPABILITY ReadUserData
CAPABILITY WriteUserData
```

Preconditions

Databases nonsecure.db and [UID3]secure.db need to be created before executing this code snippet. See Archived:Using SQL API for creating non-secure and secure databases on Symbian.

Source file

```
#include <e32base.h>
#include <SqlDb.h>

void AttachDatabasesL()
  {
  RSqlDatabase database;

  _LIT(KNonSecureDbName, "\\nonsecure.db");
  _LIT(KSecureDbName, "[E80000AF]secure.db");
  _LIT(KLogicalName,"ATTACHED");

  TInt error = database.Open(KNonSecureDbName);

  if (error == KErrNone)
    {
    CleanupClosePushL(database);
    User::LeaveIfError(database.Attach(KSecureDbName,KLogicalName));

    _LIT(KSqlAttachSelect,"SELECT BOOKS.TITLE FROM ATTACHED.BOOKS WHERE BOOKS.TITLE =
 (SELECT TITLE FROM MOVIES)");

    RSqlStatement sqlAttachSelectStatement;
    TInt ret = sqlAttachSelectStatement.Prepare(database, KSqlAttachSelect);
    TInt columnIndex = sqlAttachSelectStatement.ColumnIndex(_L("TITLE"));

    if (ret == KErrNone)
      {
      CleanupClosePushL(sqlAttachSelectStatement);
      TInt err = KErrNone;

      while((err = sqlAttachSelectStatement.Next()) == KSqlAtRow)
        {
        //process next record
        TPtrC title = sqlAttachSelectStatement.ColumnTextL(columnIndex);
        //do something with title...
```

 (C) Copyright Nokia 2013. All rights reserved.

```
        }
      if(err == KSqlAtEnd)
        {
        //no more records
        }
      else
        {
        //error has occured
        }
      CleanupStack::PopAndDestroy(); //sqlAttachSelectStatement
      }
    else
      {
      //prepare sql statement failed
      }

    database.Detach(KLogicalName);
    CleanupStack::PopAndDestroy(); //database
    }
  else
    {
    //open database failed
    }


  }
```

Postconditions

The database [UID3]secure.db has been attached to nonsecure.db (the primary database) and all 'TITLE' column values from [UID3]secure.db/BOOKS table that exist in the nonsecure.db/MOVIES 'TITLE' column have been fetched. Finally, the other database is detached and the primary database is closed.

See also

- Archived:Using SQL API for creating non-secure and secure databases on Symbian
- Archived:Using Symbian SQL API with SQL statements which do not return data
- Archived:Using Symbian SQL API with SQL statements which return data
- Archived:Using Symbian SQL API with scalar queries
- Archived:Using Symbian SQL API with data streams

 (C) Copyright Nokia 2013. All rights reserved.

 (C) Copyright Nokia 2013. All rights reserved.