【セッションNo. 2】

BDE接続との違いから応用プログラムの作成まで

「チャレンジ！ dbExpress接続」

株式会社ミガロ.
システム事業部　システム3課
小杉　智昭

【アジェンダ】

○ dbExpress接続とは？
- なぜdbExpress接続なのか
- BDE接続との違い

○ 実践テクニックあれこれ
- テクニック紹介
- データ参照ユーティリティ

dbExpress接続とは？

dbExpress接続とは？

Delphi/400 v6より追加された接続方式です。
以下の6項目がその設計方針となります。

- システムリソースの使用を最小限に抑える
- 速度を最大化する
- クロスプラットフォームをサポートする
- 配布をより容易にする
- ドライバの開発をより容易にする
- 開発者により強力な制御手段を提供する

dbExpress接続とは？

dbExpress接続とBDE接続の比較

・dbExpress接続とBDE接続の比較

	BDE接続	dbExpress接続
クライアント／サーバ アプリケーション	◎	○
Webサーバ アプリケーション	○	◎
ユニコード対応	×	○
パフォーマンス	○	◎

dbExpress接続とは？

dbExpress接続とBDE接続の比較

・利用可能な主なデータベースとバージョンの違い

	BDE接続	dbExpress接続
OS/400	V7R1	V7R1
Oracle	8.1	11g
SQL Server	7.0	2008
DB2 UDB	5.1	8.x
MySQL	—	5.1
Paradox	7	—

dbExpress接続とは？

dbExpress接続の推奨例

① Webサーバアプリケーション
メモリ使用量が少ないため、高負荷状態に耐えやすい
単方向データセットの機能でアプリケーションが構築できる
⇒ 特にお勧め！

② CCSIDに1399を指定したシステムのアプリケーション
ユニコード文字に対応可能
⇒ ユニコードでないと表現できない文字を使用する場合は必須！

dbExpress接続とは？

dbExpress接続の推奨例

③ ピンポイントでパフォーマンスが必要となる処理
速度が最大化されている
より詳細な制御手段が用意されている
⇒ プログラム難易度がやや高いため注意

④ 複数種類のデータベースを連携
対応データベースの幅が広く、最新バージョン対応も行われている
⇒ 同じ接続方式で複数のデータベースに対応可能

dbExpress接続とは？

BDE接続との違い

・接続パラメータ

	BDE接続	dbExpress接続
接続先AS/400	DATABASE NAME	Database HostName
省略時のライブラリ	LIBRARY NAME	RoleName
ユーザ名	USER NAME	User_Name
パスワード	PASSWORD	Password
トランザクションレベル	TRANISOLATION	BlobSize
セッション分離	MULTISESSION	Multiple Transaction

dbExpress接続とは？

BDE接続との違い

＜特に注意が必要な接続パラメータ＞

[接続先AS/400]　（BDE:「DATABASE NAME」）
「Database」パラメータになり、省略ができなくなりました。
また、「HostName」パラメータに同じ値を指定する必要があります。

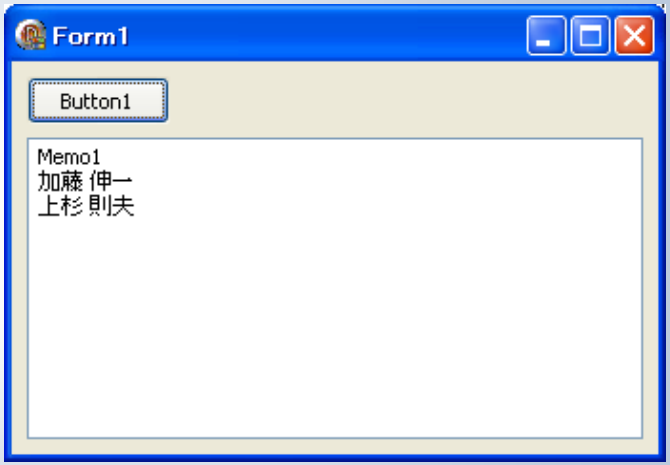
[トランザクションレベル]　（BDE:「TRAN_ISOLATION」）
v2007以降では、「BlobSize」パラメータになりました。
（ *NONE: -1、*CHG: -2、*CS: -3、*ALL: -4 ）
v2006以前では、StartTransactionメソッドのパラメータで指定します。

dbExpress接続とは？

・接続パラメータの違いの確認（補足資料：デモ①、デモ②）

AS/400　MIGAROTK
Library　D400TR
File　PNAME

ユーザ名　D400
パスワード　D400



dbExpress接続とは？

BDE接続との違い

・**単方向データセット**
速度の向上とメモリ使用量の削減のため、データセットが**単方向データセット**になりました
・・・ レコードの逆移動等ができなくなったため、**DBGridと直接連携できません**
DataSetProviderとClientDataSetを併用します

・**メンバファイルへのアクセス**
SQLベースのアクセスになりました
・・・ SQLにはメンバの概念がないため、**メンバ名を直接指定できません**
OVRDBF（コマンド）またはCREATE ALIAS（SQLステートメント）を併用します

dbExpress接続とは？

・単方向データセットの使い方（補足資料：デモ③、デモ④）

ユーザ名　D400
パスワード　D400

AS/400　MIGAROTK
Library　D400TR
File　BKMAST

dbExpress接続とは？

その他、注意事項

・バージョンアップ時の注意

v2005以前とv2006以降でdbExpress接続で使用されるミドルウェアのDLLファイルが変更になりました

・・・ このプロパティは自動的に更新されませんので、**手動で変更が必要です**

対象コンポーネント:	TSQLConnection
対象プロパティ:	LibraryName
v2005以前の値:	dbco400.dll
v2006以降の値:	dbco430.dll

■実践テクニックあれこれ

Tech1）接続先AS/400 NAMEを省略可能にする

<問
BD
AS
で

<対
De
Tc
を

に
続

いる
lame

```
// OnBeforeConnect：AS/400に接続する直前に発生するイベント
procedure TForm1.SQLConnection1BeforeConnect(Sender: TObject);
var
  AS400Name: String;
  AS400List: TStringList;
begin
  // Databaseパラメータから接続先AS/400名を取得します。
  AS400Name := (Sender as TSQLConnection).Params.Values['Database'];

  AS400List := TStringList.Create;
  try
    // AS/400 ListからAS/400 Nameの一覧を取得します。
    TcGetAS400(AS400List);
    // 取得した一覧の中にDatabaseパラメータから取得したAS/400名が
    // 無いか確認します。
    if AS400List.IndexOf(AS400Name) < 0 then
    begin
      // 無かった場合はDatabaseパラメータとHostNameパラメータに取得した
      // 一覧の最上位のAS/400 Nameをセットします。
      (Sender as TSQLConnection).Params.Values['Database'] := AS400List[0];
      (Sender as TSQLConnection).Params.Values['HostName'] := AS400List[0];
    end;
  finally
    AS400List.Free;
  end;
end;
```

■実践テクニックあれこれ

Tech2) 変更内容をデータベースに反映する

＜問題＞

ClientDataSetを使うと画面上でデータを更新しても実際のファイルに**変更した内容が反映されません。**

```
// OnClick：コンポーネントがクリックされたときに発生するイベント
procedure TForm1.Button1Click(Sender: TObject);
begin
  // ClientDataSetに対して行われた変更をファイルに更新する
  ClientDataSet1.ApplyUpdates(0);
end;
```

更新した内容を実際のファイルへ反映させるには、**ApplyUpdatesメソッド**を呼び出します。

尚、DataSetProvider / ClientDataSetはBDE接続でも使用可能です。

■実践テクニックあれこれ

Tech3) データの並び順を自由に変更する

＜問題＞
BDE接続でTableのIndexFieldNamesプロパティで並び順を指定する場合、予めその並び順の論理ファイルを作成しておかないと「xxx: 項目 'yyyy' のインデックスを持っていません.」とのエラーになります。

＜対策＞
dbExpress接続のSQLTableでは、SQLベースのアクセスになったため、この制限はなくなりました。
また、ClientDataSetでも同様に並び順が指定でき、この場合、キャッシュ上で並び替えが行われるため、非常に高速です。

■実践テクニックあれこれ

Tech3-2) データの並び順を自由に変更する

＜問題＞

Tech3)の方法では、昇順にしか並べ替えができません。

```
// OnCreate : コンポーネント作成時に発生するイベント
procedure TForm1.FormCreate(Sender: TObject);
begin
  // CATEGO 昇順 / BTDATE 降順 / PCode 降順 となるインデックス情報を
  // 追加します。
  ClientDataSet1.AddIndex('CDSIndex', 'CATEGO;BTDATE;PCODE', [],
                                      'BTDATE;PCODE');
  // 作成したインデックス順に並ぶよう指定します。
  ClientDataSet1.IndexName := 'CDSIndex';
end;
```

■実践テクニックあれこれ

Tech4) 抽出結果をファイルに保存する

＜問題＞
処理速度向上のため、ローカルPC上にマスタのデータをコピーしたい。

```
// OnClick : コンポーネントがクリックされたときに発生するイベント
procedure TForm1.Button1Click(Sender: TObject);
begin
  // バイナリ形式で保存する場合
  ClientDataSet1.SaveToFile('SampleData.bin');
  // XML形式で保存する場合
//ClientDataSet1.SaveToFile('SampleData.bin', dfXMLUTF8);
end;
```

また、保存の際にパラメータを指定することで、バイナリ形式以外に、
XMLとして保存することも可能です。

■実践テクニックあれこれ

Tech5) データベースとの通信状況を監視する

＜問題＞
エラー原因の追及やパフォーマンス改善のため、通信状況のログを取得したい。

```
// OnTrace：通信ログが発生する度に発生するイベント
procedure TForm1.SQLMonitor1Trace(Sender: TObject; TraceInfo: TDBXTraceInfo;
  var LogTrace: Boolean);
begin
  // Memoコンポーネントに通信ログを出力します。
  Memo1.Lines.Add(TraceInfo.Message);
end;
```

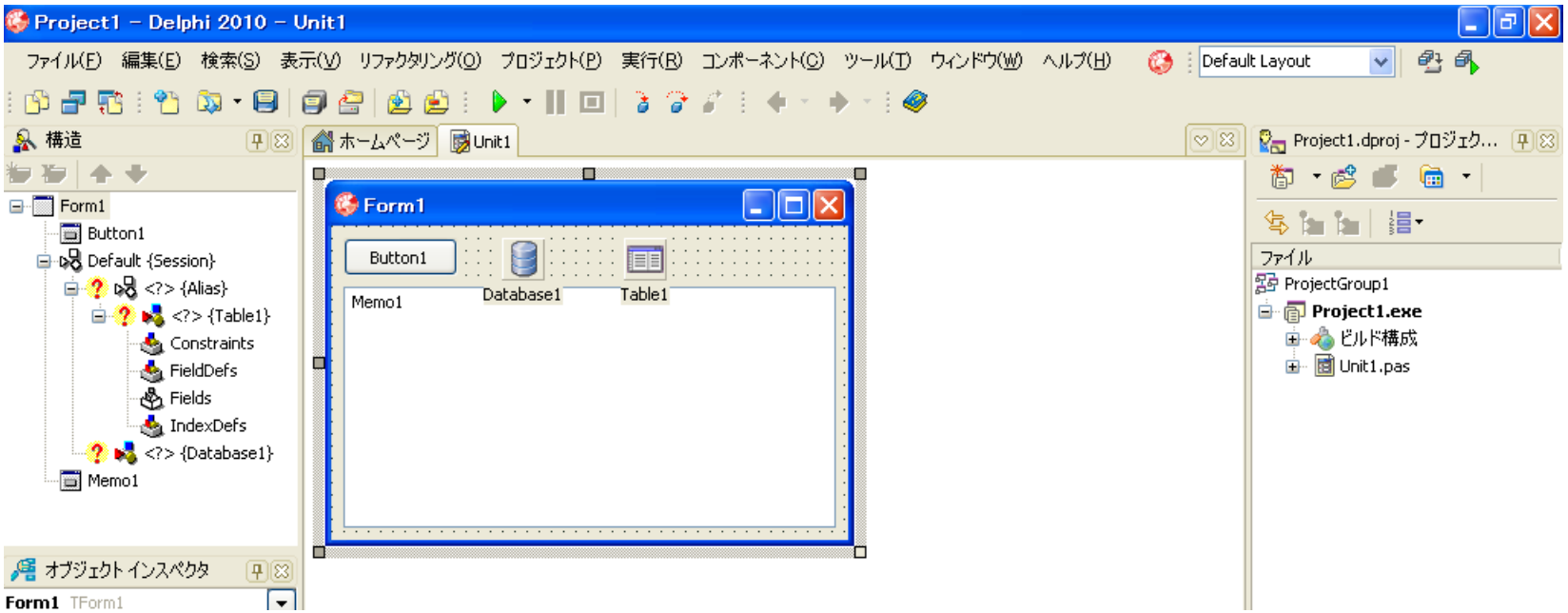
通信状況のログはPCファイルとして保存する以外に、OnTraceイベントを使ってリアルタイムに画面表示することも可能です。

■実践テクニックあれこれ

■補足資料：デモ① – BDE接続を使ったデータ表示

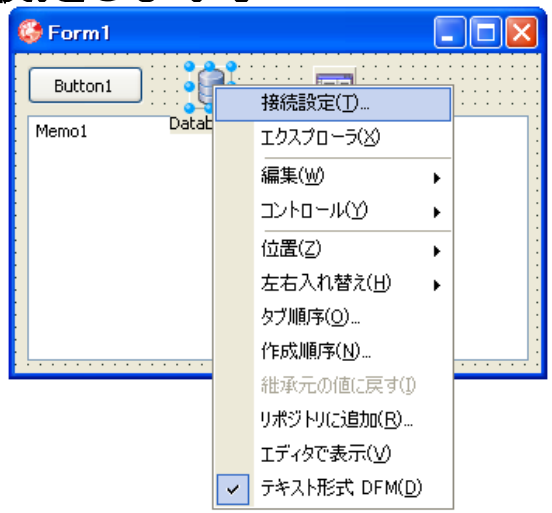
1. 新規VCLフォームアプリケーションを作成し、以下のコンポーネントを貼り付けます。

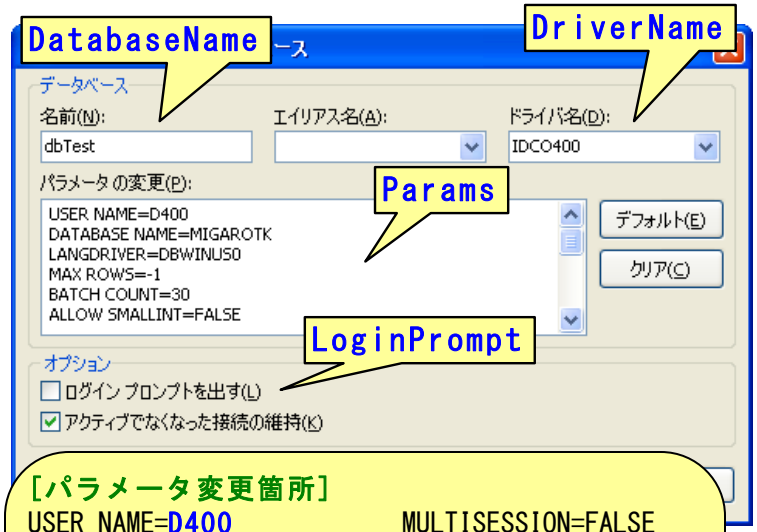
[Standard]	TButton	TMemo
[BDE]	TDatabase	TTable



■補足資料：デモ① － BDE接続を使ったデータ表示

2. Databaseコンポーネントのプロパティをコンポーネントエディタを使って設定します。





［パラメータ変更箇所］

```
USER NAME=D400                  MULTISESSION=FALSE
DATABASE NAME=MIGAROTK          TRAN_ISOLATION=*NONE
LANGDRIVER=DBWINUS0             VERIFY FILES=TRUE
MAX ROWS=-1                     LONG FIELDNAMES=FALSE
BATCH COUNT=30                  OLDFILTER=FALSE
ALLOW SMALLINT=FALSE            BOEOF=FALSE
APPC BUFFSIZE=5000              DDS PATH=DEFAULT
LIBRARY NAME=D400TR             PASSWORD=D400
LOCAL INDEXES=TRUE
```

■補足資料：デモ① – BDE接続を使ったデータ表示

3. Tableコンポーネントのプロパティをオブジェクトインスペクタを使って設定します。

③ ①②の設定後、Trueにします

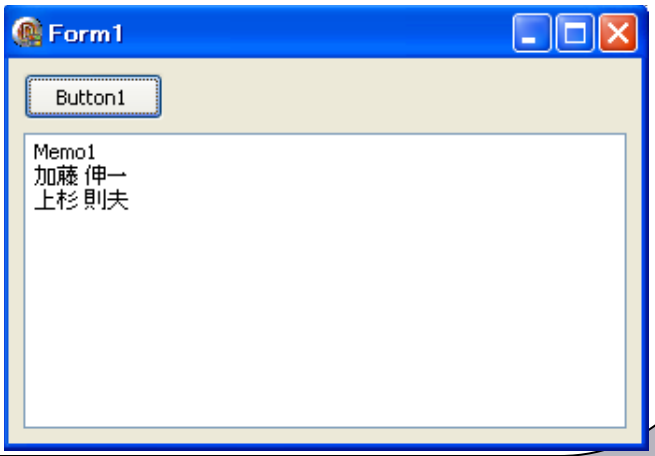
① 2. のDatabaseNameと同じものを指定します

② AS/400のファイル名を指定します

■補足資料:デモ① – BDE接続を使ったデータ表示

4. ButtonコンポーネントのOnClickイベントを作成して、以下のコードを記述、プログラムを実行します。

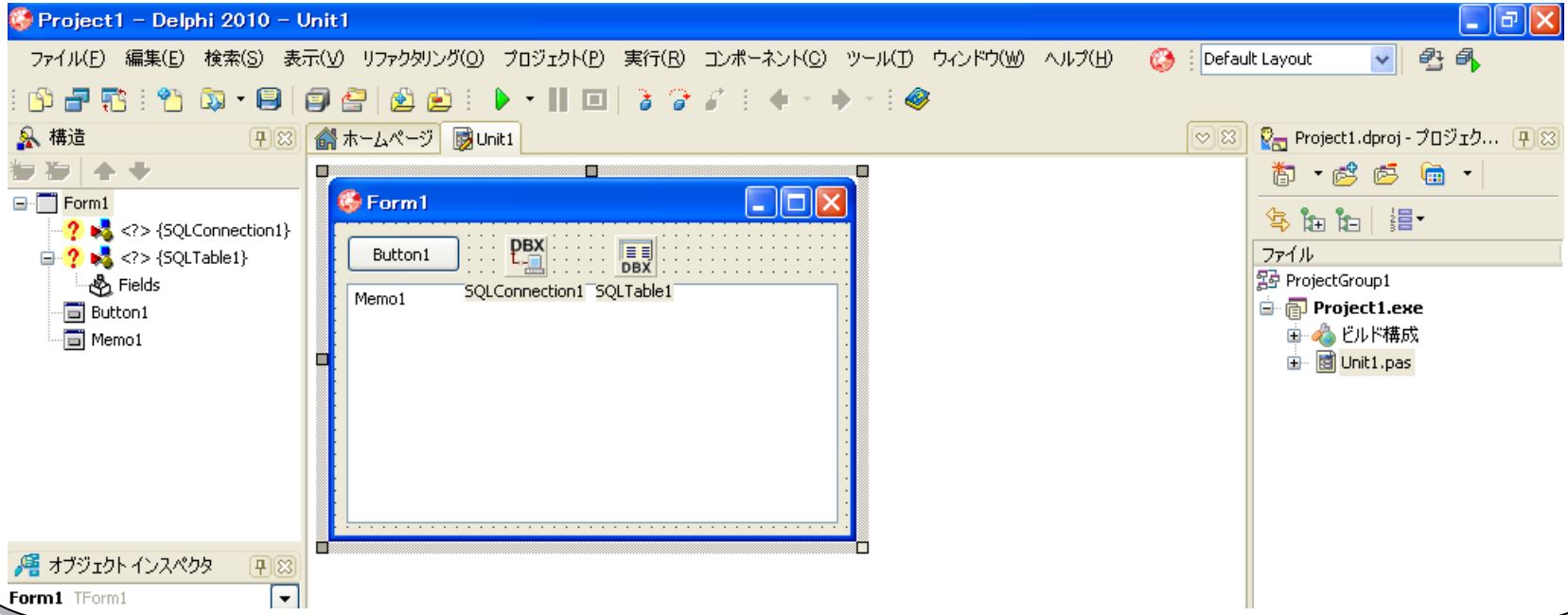
```
procedure TForm1.Button1Click(Sender: TObject);
var
  i: Integer;
begin
  Memo1.Lines.Add(Table1.FieldByName('PNAME').AsString);
  Table1.Next;
end;
```



■補足資料：デモ② － dbExpress接続を使ったデータ表示

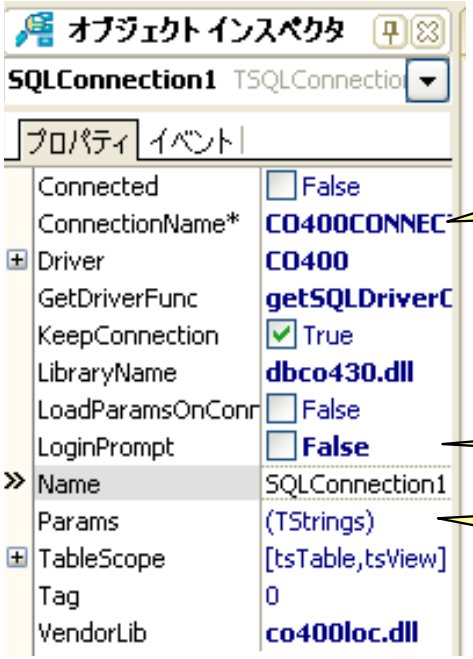
1. 新規VCLフォームアプリケーションを作成し、以下のコンポーネントを貼り付けます。

[Standard]	TButton	TMemo
[dbExpress]	TSQLConnection	TSQLTable



■補足資料：デモ② － dbExpress接続を使ったデータ表示

2. SQLConnectionコンポーネントのプロパティをコンポーネントエディタを使って設定します。



① CO400CONNECTIONを選択します

以下のプロパティが自動的に変更されます
Driver / GetDriveFunc / LibraryName / VenderLib

② Falseにします

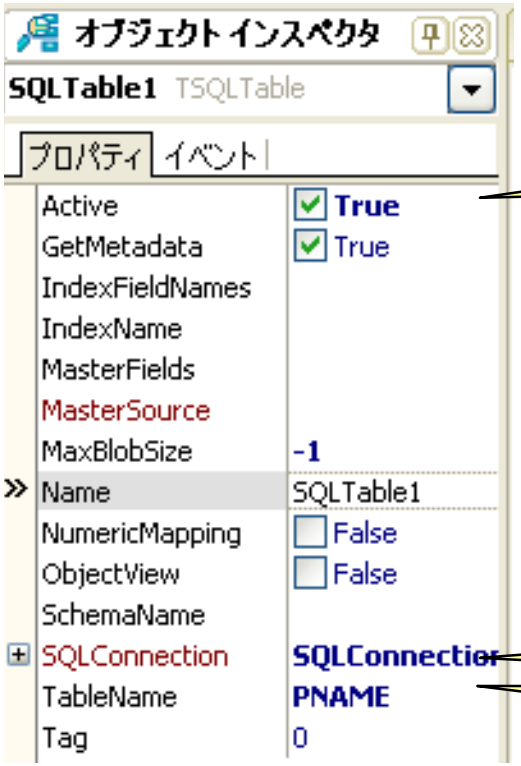
③ プロパティエディタでパラメータを編集します

[パラメータ変更箇所]

BlobSize=-1
ErrorResourcefile=
DriverName=CO400
Database=MIGAROTK
LocaleCode=0000
IsolationLevel=ReadCommited
Decimal Separator=.
Password=D400
User_Name=D400
HostName=MIGAROTK
RoleName=D400TR
Multiple Transaction=False

■補足資料：デモ② – dbExpress接続を使ったデータ表示

3. SQLTableコンポーネントのプロパティをオブジェクトインスペクタを使って設定します。



③ ①②の設定後、Trueにします

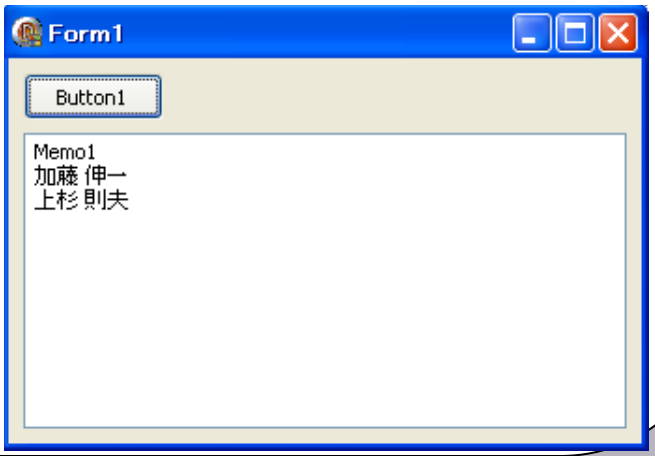
① 貼り付けたSQLConnectionコンポーネントを指定します

② AS/400のファイル名を指定します

■補足資料:デモ② – dbExpress接続を使ったデータ表示

4. ButtonコンポーネントのOnClickイベントを作成して、以下のコードを記述、プログラムを実行します。

```
procedure TForm1.Button1Click(Sender: TObject);
var
  i: Integer;
begin
  Memo1.Lines.Add(SQLTable1.FieldByName('PNAME').AsString);
  SQLTable1.Next;
end;
```



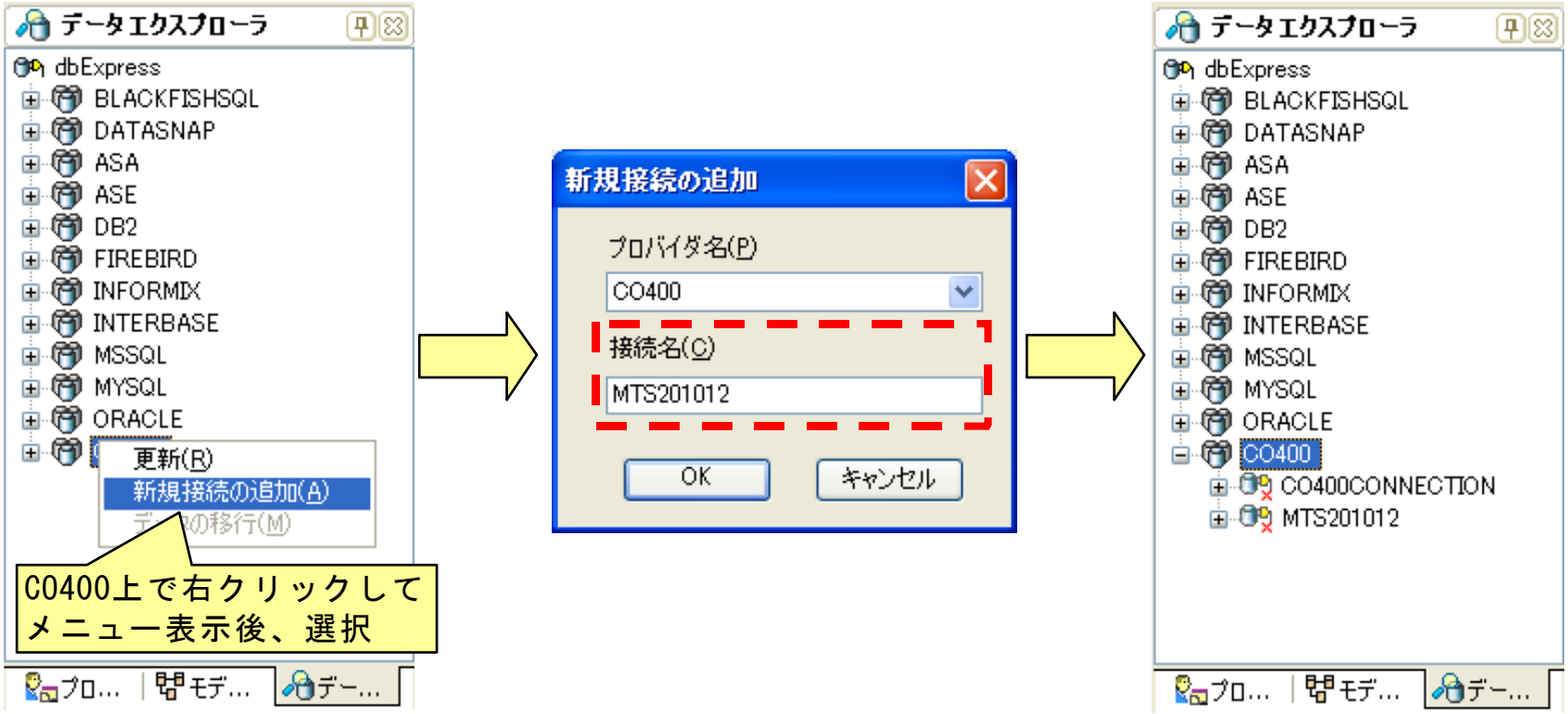
■補足資料：デモ③ －【参考】新しい接続を追加する

よく使用するパラメータの組合せを予め追加しておくことで効率化を図ることができます。（BDE接続におけるエリアスに相当します）
コンポーネントのプロパティを個別に設定する場合は、この処理は必要ありません。

1. データエクスプローラを表示します。

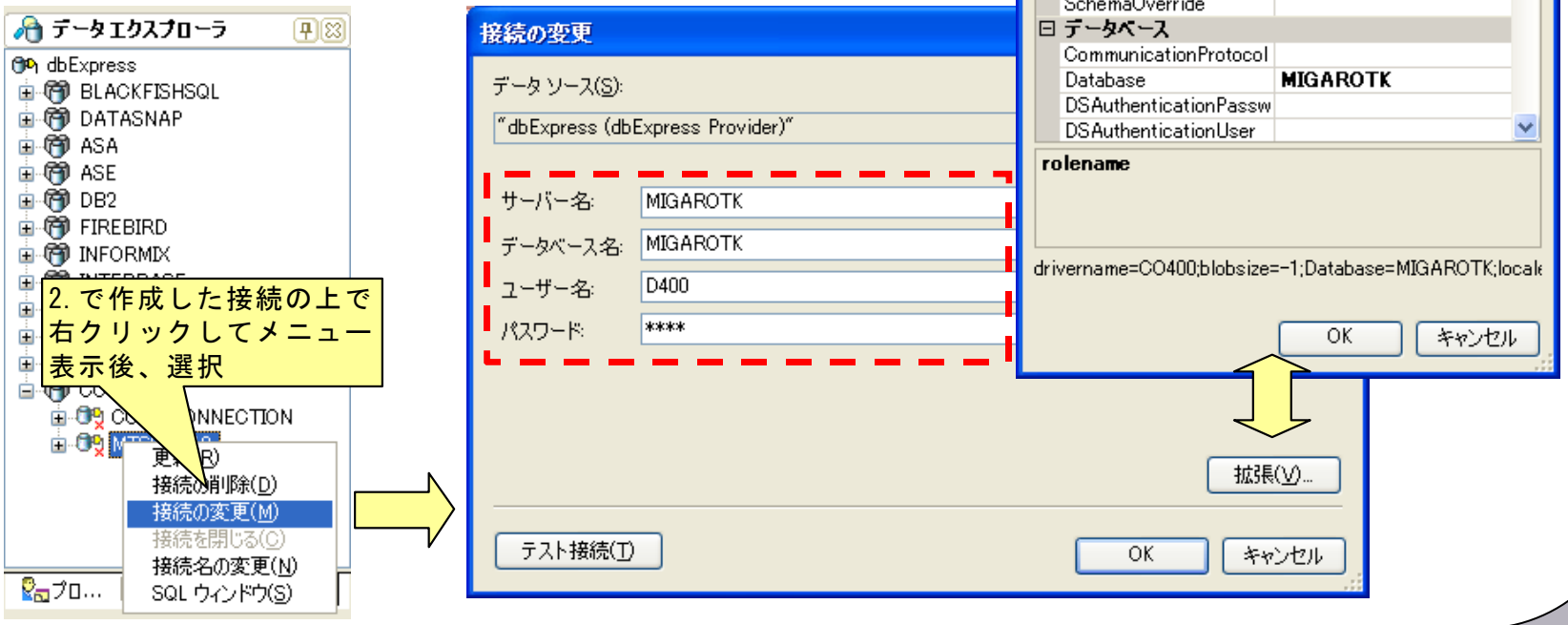
■補足資料：デモ③ －【参考】新しい接続を追加する

2. データエクスプローラでCO400プロバイダの接続を追加します。



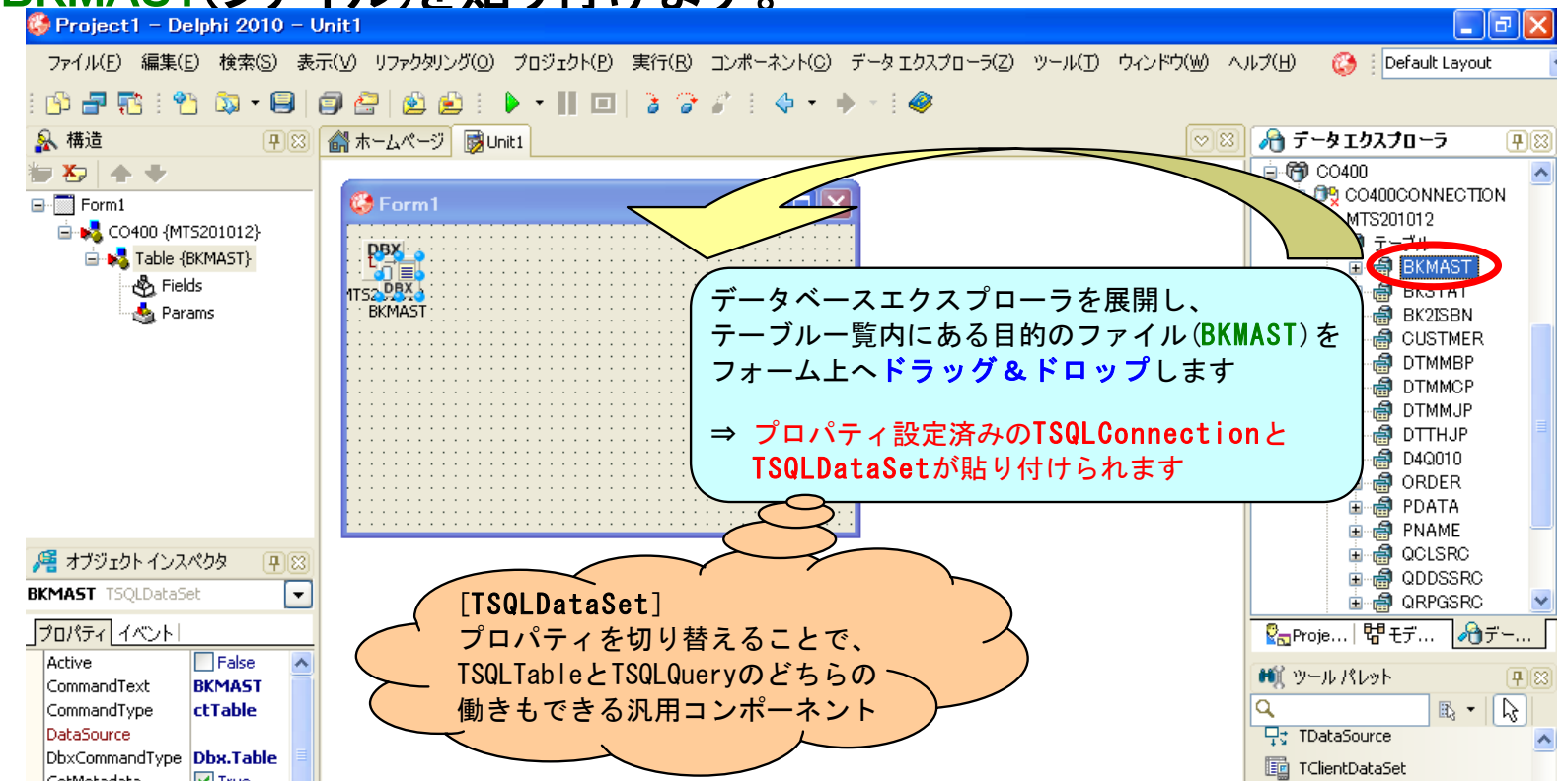
■補足資料：デモ③ －【参考】新しい接続を追加する

3. 新しく作成した接続のパラメータを変更します。



■補足資料：デモ④ – 単方向データセット

1. 新規VCLフォームアプリケーションを作成し、データエクスプローラから BKMAST(ファイル)を貼り付けます。

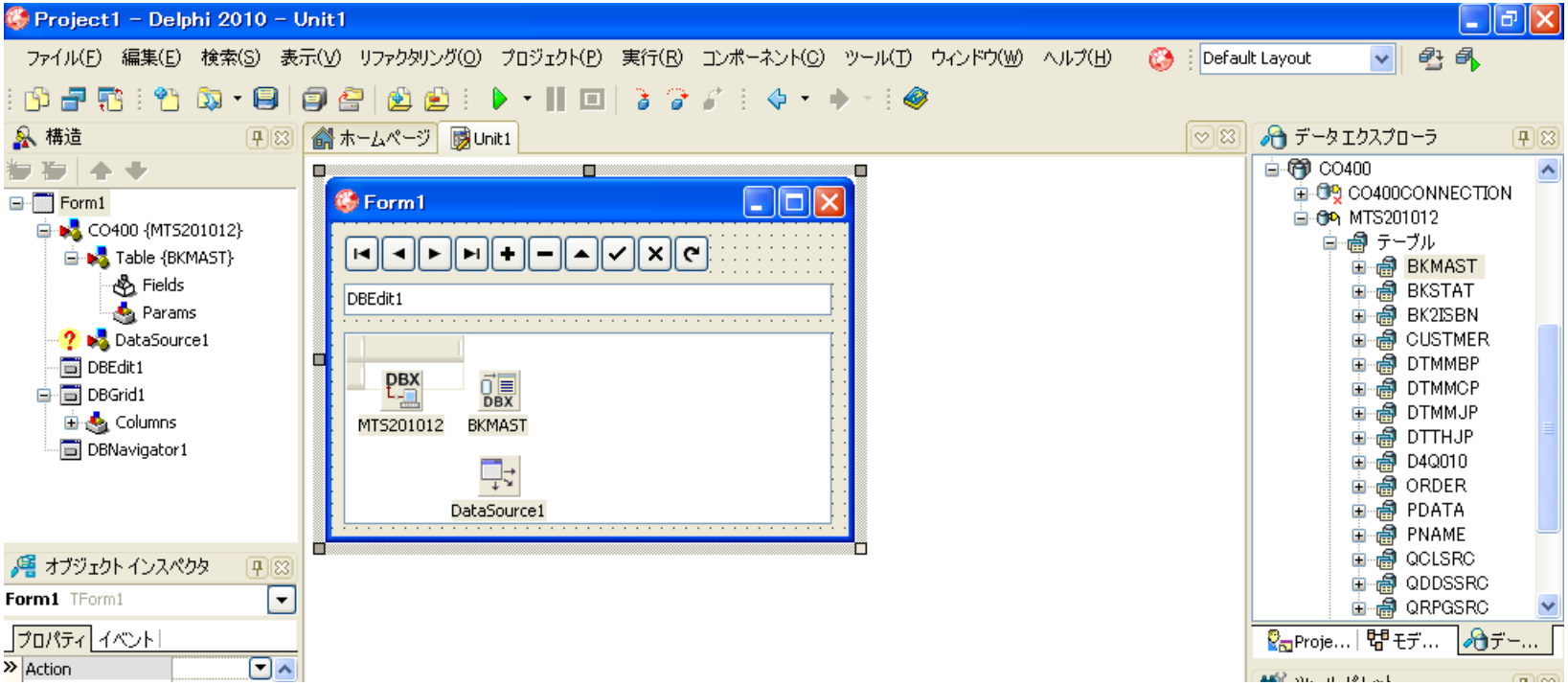


■補足資料：デモ④ – 単方向データセット

2. 以下のコンポーネントを貼り付け、位置や大きさを調整します。

[Data Access] TDataSource

[Data Controls] TDBNavigator TDBEdit TDBGrid



■補足資料：デモ④ – 単方向データセット

3. 1.で貼り付けられたBKMAST(SQLDataSetコンポーネント)のActiveプロパティをTrueにし、DataSourceコンポーネントのDataSetプロパティにBKMASTを指定します。

[SQLDataSetコンポーネント]

[DataSourceコンポーネント]

■補足資料：デモ④ – 単方向データセット

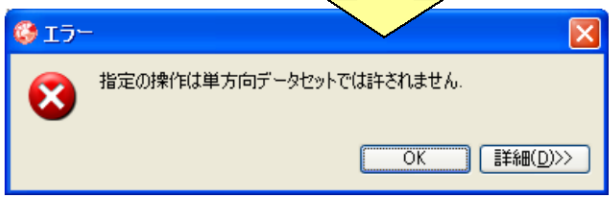
4. DBNavigator、DBEdit、DBGridの各コンポーネントのDataSourceプロパティに貼り付けたDataSourceコンポーネントを指定します。

※ DBEditコンポーネントのみ、DataFieldプロパティも同時に設定します。

[DBNavigatorコンポーネント]

[DBEditコンポーネント]

[DBGridコンポーネント]



単方向データセットのため、DBGridを直接結びつけることができません。

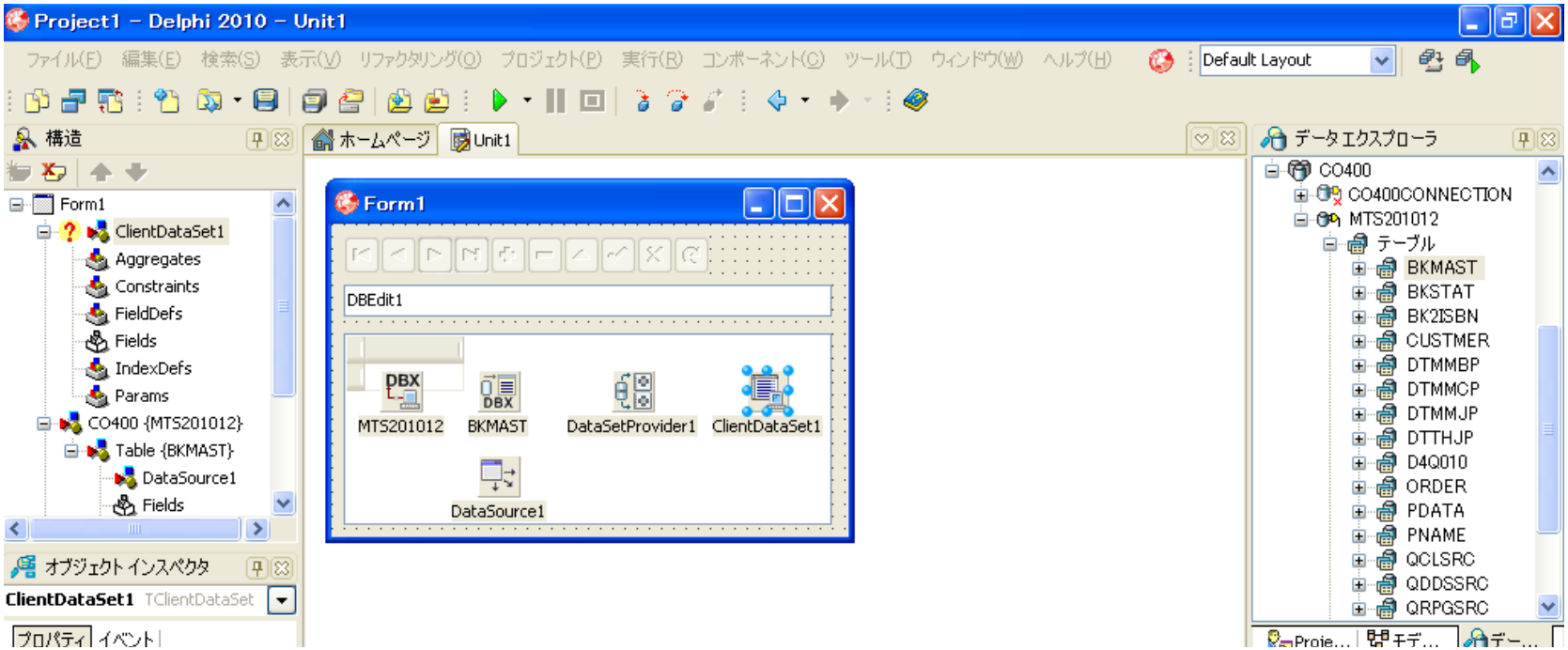
■補足資料：デモ④ – 単方向データセット

5. 3. で変更したBKMAST(SQLDataSetコンポーネント)のActiveプロパティをFalseに戻します。

[SQLDataSetコンポーネント]

■補足資料：デモ④ – 単方向データセット

6. 以下のコンポーネントを更に追加します。
　　[Data Controls]　　TDataSetProvider　　TClientDataSet



■補足資料：デモ④ – 単方向データセット

7. DataSetProviderコンポーネントのDataSetプロパティにBKMAST(SQLDataSetコンポーネント)を指定後、ClientDataSetコンポーネントのProviderNameプロパティにDataSetProviderコンポーネントを指定し、Activeプロパティ をTrueにします。

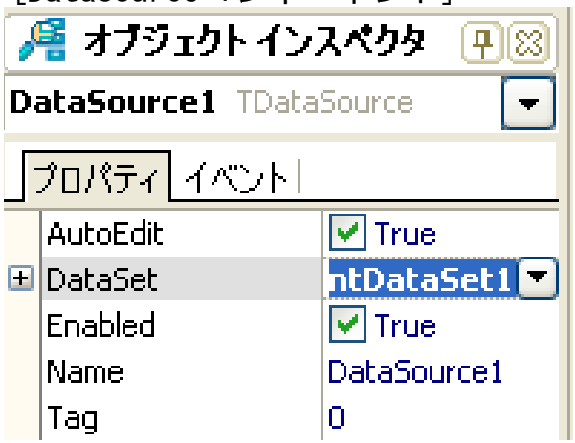
[DataSetProviderコンポーネント]

[ClientDataSetコンポーネント]

■補足資料：デモ④ – 単方向データセット

8. 3.で設定したDataSourceコンポーネントのDataSetプロパティを ClientDataSetコンポーネントに変更します。
その後、4.で設定できなかったDBGridコンポーネントのDataSourceプロパティを再度設定します。⇒ 今度は正しく設定できます。

[DataSourceコンポーネント]



[DBGridコンポーネント]