福井 和彦

株式会社ミガロ.

システム事業部 プロジェクト推進室

[Delphi/400] 新データベースエンジンFireDACを使ってみよう!

略歴
1972 年 3 月 20 日生まれ
1994 年 大阪電気通信大学工学部卒業
2001 年 4 月 株式会社ミガロ. 入社
2001 年 4 月 システム事業部配属

現在の仕事内容
主に Delphi/400 を使用したシステムの受託開発を担当しており、要件確認から納品・フォローに至るまで、システム開発全般に携わっている。また、Delphi/400 の導入支援やセミナーの講師も行っている。

1.はじめに

業務アプリケーションを開発していくうえで、データベースの利用は不可欠である。そして、Delphi/400 はデータベースを使ったアプリケーション開発を得意としている。なぜなら、Delphi/400 はデータベースエンジンという機能を備えており、これによりどのデータベースに対しても、共通のプログラミングで簡単に開発できるからである。

このデータベースエンジンと各種データベースドライバを組み合わせることで、IBM i や SQL Server などさまざまなデータベースへ接続できる。【図 1】

これまでの開発で使用されてきたデータベースエンジンには、BDE や dbExpress がある。本稿で主題としている「FireDAC」は、BDE や dbExpress に続く新しいデータベースエンジンである。

FireDAC 自体は Delphi の XE3 から実装されており、これまでも Oracle や SQL Server などでは使用可能であった。そして Delphi 10 Seattle からは、Delphi/400 のドライバが対応し、IBM i でも活用可能になっている。

そこで本稿では、FireDAC で IBM i を利用する基本的な方法、そしてすでに BDE や dbExpress を使用して IBM i へ接続しているプログラムを FireDAC へ移行するポイントについて説明する。

2.FireDACを使った IBM i アプリケーション開発

2-1. FireDAC とは

本題に入る前に、FireDAC はこれまでのデータベースエンジンと比べて、どのような違いがあるかを確認する。FireDAC の特徴としては、以下の点が挙げられる。

(1) Windows 32bit ／ 64bit に対応
(2) FireMonkey に対応
(3) BDE のような初期インストールは不要（配布が簡単）
(4) 双方向データセット形式
(5) 高いパフォーマンスのデータアクセス（BDE と同等以上）

FireDAC と BDE、dbExpress との機能比較については【表 1】に示す。

これらの特徴や比較から、FireDAC は dbExpress と同様の環境対応機能があり、BDE と同等の双方向データセット形式を備えていることがわかる。

つまり、FireDAC は両データベースエンジンのよい部分を組み合わせた新しいデータベースエンジンといえる。次に、この FireDAC を使用して IBM i へ接続し、ファイルを参照する基本的な使い方を説明していく。

2-2. FireDAC の使い方

FireDAC で IBM i へ接続するために使用する基本コンポーネントには、以下が用意されている。

(1) TFDConnection

データベースへの接続を制御するコ

図1

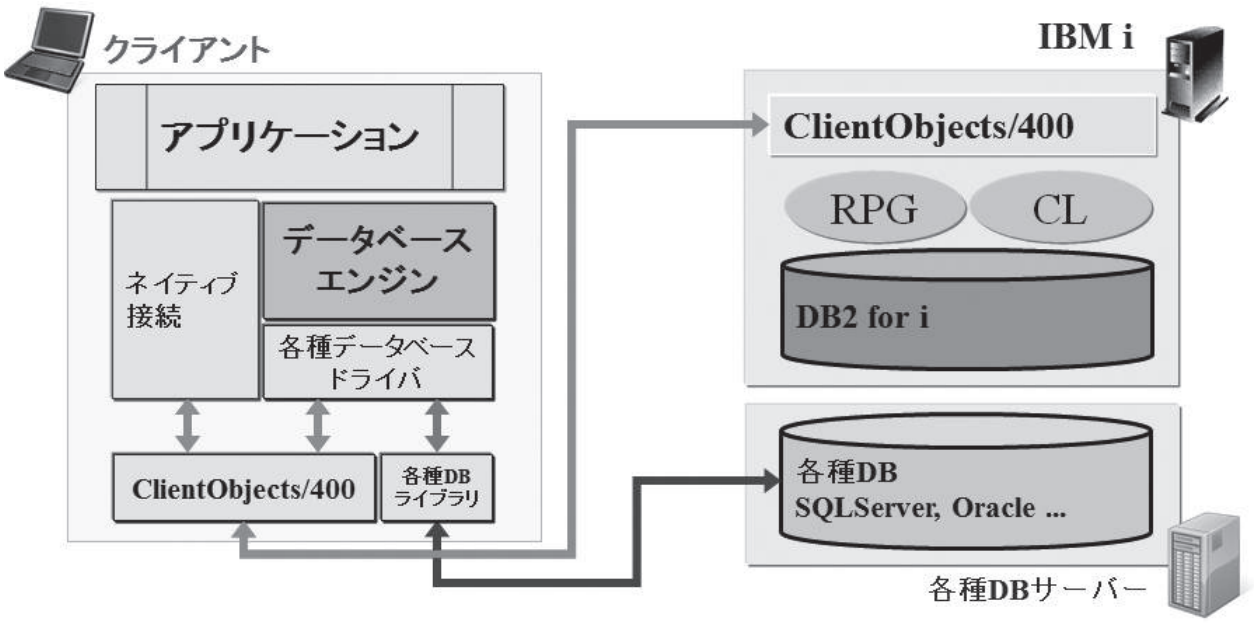


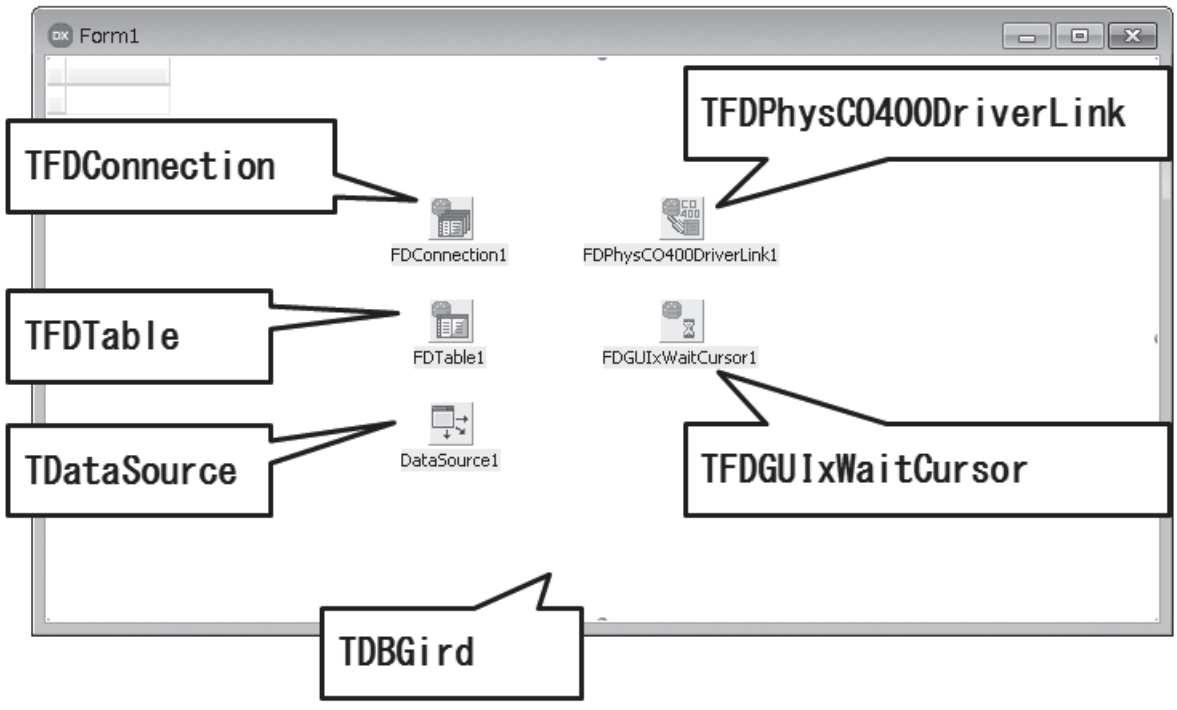
表1

FireDACと他のデータベースエンジンとの比較表

	FireDAC	dbExpress	BDE
Windows32bit	○	○	△
Windows64bit	○	○	×
FireMonkey	○	○	×
初期インストール	不要	不要	必要
データセット形式	双方向	単方向	双方向
速度	○	△	○

※dbExpressの速度は単方向でClientDataSetの使用率が高いため△としている

図2



ンポーネント
(2) TFDPhysCO400DriverLink
TFDConnection に Delphi/400 の IBM i 用ドライバ情報を提供するコンポーネント
(3) TFDTable
単一のファイルを指定して、データを取得・操作するコンポーネント
(4) TFDQuery
SQL を実行して、データを取得・操作するコンポーネント
(5) TFDGUIxWaitCursor
待機カーソルなどを制御するコンポーネント

それでは FireDAC を使用して IBM i へ接続し、ファイルを参照する手順を順番に確認していく。ファイルへの接続は、TFDTable コンポーネントを使用した基本的な構成とする。

①コンポーネントの配置

【図 2】に従って、新規フォームに各コンポーネントを配置する。配置できたら、FireDAC の各コンポーネントのプロパティを順番に設定していく。

最初に IBM i への接続設定を、TFDConnection コンポーネントで行う。フォームに貼り付けた TFDConnection コンポーネントをダブルクリックすると、FireDAC 接続エディタが起動する。【図 3】

起動した FireDAC 接続エディタの上部にある、接続定義名のプルダウンより“CO400DEF”を指定すると、パラメータが表示される。パラメータの“Database”“User_Name”“Password”“ODBCAdvanced”を、【図 4】に従って設定する。

TFDPhysCO400DriverLink コンポーネントと TFDGUIxWaitCursor コンポーネントは、フォームに貼り付けるだけでとくに設定を行う必要はない。

続いて、ファイルの参照設定を TFDTable コンポーネントで行う。TFDTable コンポーネントの Connection プロパティは、TFDConnection コンポーネントが自動で初期セットされているので、TableName プロパティに参照するファイル名を設定する。【図 5】

このプログラムでは、【図 4】の TFDConnection コンポーネントの ODBCAdvanced パラメータにライブラリ名を指定しているため、TableName プロパティではリストが自動表示されて選択できる。

あとは TDataSource コンポーネントの DataSet プロパティと、TDBGrid の DataSource プロパティを設定すれば、各コンポーネントの設定は完了となる。【図 6】

ここまでで、プログラム上の設定は完了である。

実際に FireDAC を使用してデータへアクセスするには、TFDTable コンポーネントの Active プロパティを True にする。これによって、TDBGrid 上にデータを表示できる。【図 7】

FireDAC は新しいデータベースエンジンではあるが、Delphi/400 ではこれまでのプログラムと互換性を維持できる形でコンポーネントが用意されている。ここまでの実装手順を確認すると、BDE や dbExpress の開発とほとんど違いはなく、また TClientDataSet を必要としない分、よりシンプルに開発できることがわかる（もちろん TClientDataSet を使用することも可能である）。

3. 既存プログラムの FireDACへの移行

3-1. FireDAC へのプログラム変更

ここまで FireDAC の新規プログラムを作成する方法を説明したが、次に BDE や dbExpress で作成されているプログラムを FireDAC へ移行する手順について説明する。

本稿では、次のような RPG を使った標準的な仕組みの照会画面を題材に、FireDAC へ変更するポイントを確認していく。

FireDAC へ変更する照会画面プログラムの処理

(1) データを RPG で抽出する
(2) 抽出データを QTEMP のワークファイルに作成する
(3) ワークファイルを BDE または dbExpress で画面表示する

3-2. BDE からの移行ポイント

ここでは、BDE から FireDAC への変更方法について説明する。変更する BDE の照会画面の構成は、【図 8】のとおりである。

照会画面の構成では、上段に抽出条件を指定する項目と検索ボタンを配置し、中段に明細表を配置している。動作としては、検索ボタンを押下することで抽出データを明細表に表示する。

また使用しているコンポーネントと設定しているプロパティについては、【表 2】と【図 9】のとおり、検索実行時のソースは【図 10】のとおりである。データ抽出には RPG を用いているので、TAS400 コンポーネントと TCall400 コンポーネントを使用しているが、データ抽出の処理ロジックは主題から外れるため、本稿では割愛する。

以下に、プログラムで BDE を使用している箇所を FireDAC に変更する手順を説明する。

①データベース接続処理を BDE から FireDAC へ変更

BDE を使用して作成した照会画面に、2-2 に記載した FireDAC の基本コンポーネントである「TFDConnection」「TFDTable」「TFDPhysCO400DriverLink」「TFDGUIxWaitCursor」の 4 つを、【図 11】のように配置し、TFDConnection コンポーネントのプロパティを設定する。

ただし BDE の照会画面では、【図 9】にあるように TDatabase コンポーネントのプロパティ設定をソースで行っている。そのため、TFDConnection コンポーネントには接続定義名に“CO400DEF”だけを設定し、残りの設定はソースで実装する。

ここから、ソースの変更箇所の詳細を説明する。

【図 12】のように、FormCreate イベントに記述している TDatabase コンポーネントの設定を、TFDConnection コンポーネントの設定に変更する。

ライブラリリストを使う場合、BDE では TDatabase コンポーネントの“LIBRARY NAME”に、“*LIBL”をセットするが、FireDAC の場合は TFDConnection コンポーネントの“ODBCAdvanced”に“LibraryOption=（ブランク、シングルコーテーションなし）”をセットする。これでデータベース接続処理の FireDAC への変更は、完

図3

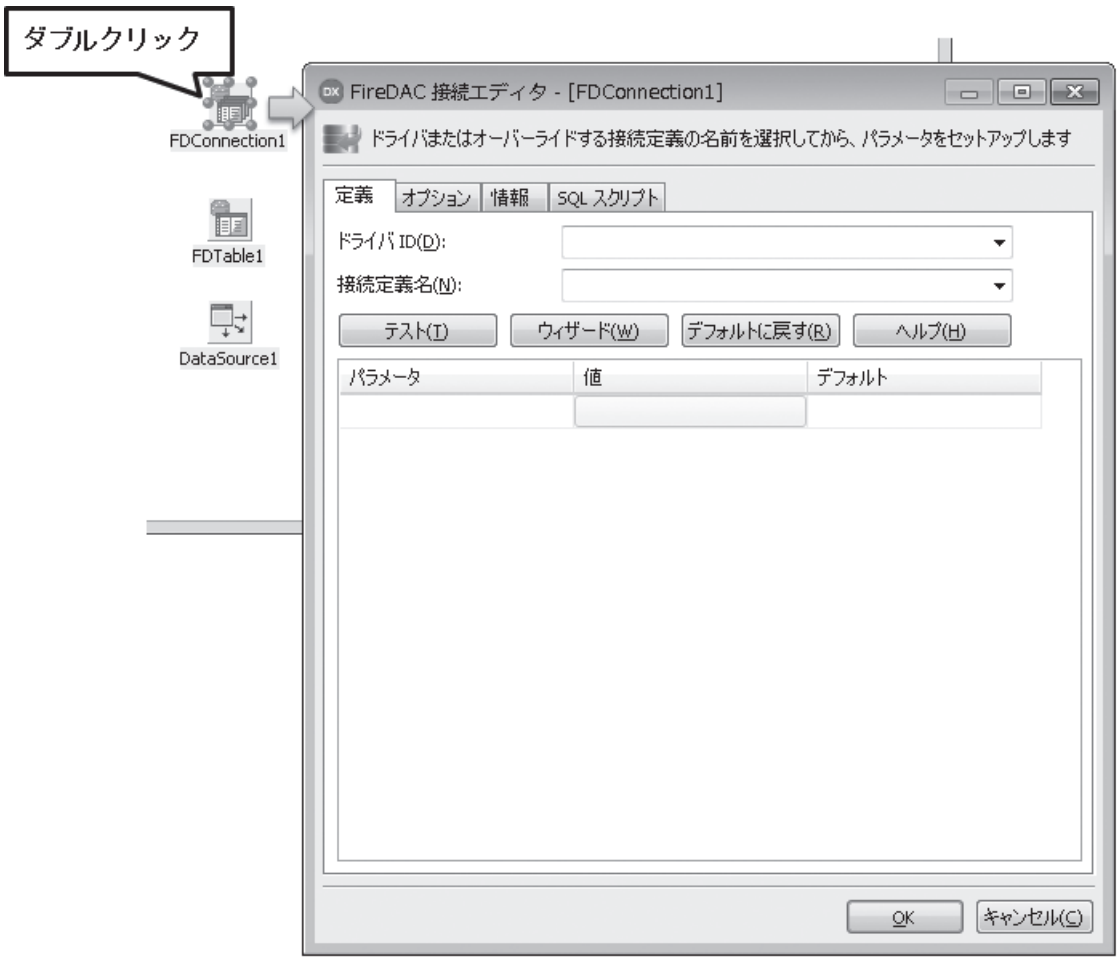
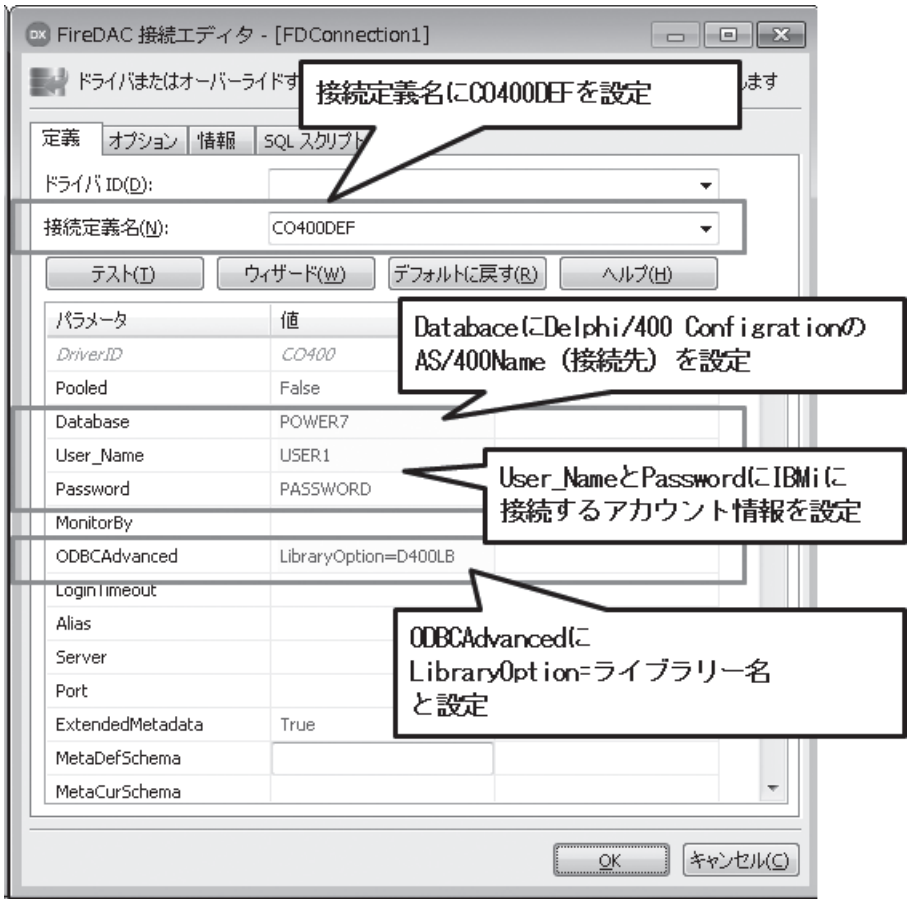

ダブルクリック
FDConnection1
FDTable1
DataSource1
FireDAC 接続エディタ - [FDConnection1]
ドライバまたはオーバーライドする接続定義の名前を選択してから、パラメータをセットアップします
定義
オプション
情報
SQL スクリプト
ドライバ ID(D):
接続定義名(N):
テスト(T)
ウィザード(W)
デフォルトに戻す(R)
ヘルプ(H)
パラメータ
値
デフォルト
OK
キャンセル(C)

図4


FireDAC 接続エディタ - [FDConnection1]
接続定義名にCO400DEFを設定
定義
オプション
情報
SQL スクリプト
ドライバ ID(D):
接続定義名(N):
CO400DEF
テスト(T)
ウィザード(W)
デフォルトに戻す(R)
ヘルプ(H)
パラメータ
値
DriverID
CO400
Pooled
False
Database
POWER7
User_Name
USER1
Password
PASSWORD
MonitorBy
ODBCAdvanced
LibraryOption=D400LB
LoginTimeout
Alias
Server
Port
ExtendedMetadata
True
MetaDefSchema
MetaCurSchema
DatabaseにDelphi/400 ConfigrationのAS/400Name（接続先）を設定
User_NameとPasswordにIBMiに接続するアカウント情報を設定
ODBCAdvancedにLibraryOption=ライブラリー名と設定
OK
キャンセル(C)

了である。

②データ表示処理をBDEからFireDACへ変更

次に、TFDTableコンポーネントのConnectionプロパティに、TFDConnectionコンポーネントが指定されていることを確認し、TableNameプロパティに参照するファイル名を設定する。

このプログラムでは、TFDConnectionコンポーネントにライブラリ名を指定していないため、リストから選択するのではなく、ファイル名を直接設定する。あとはTDataSourceコンポーネントのDataSetプロパティを、TTableからTFDTableに変更する。【図13】

最後に、【図14】のようにソース上でTTableコンポーネントを使用している部分をTFDTableコンポーネントに変更すれば、変更は完了となる。

変更したプログラムを実行すると、FireDACの仕組みでIBM iのデータを照会可能なことが確認できる。【図15】

ただし明細表のタイトル行やデータ書式を設定している場合、それが反映されていないはずである。

そのため、TFDTableコンポーネントで各フィールドの詳細設定を行う必要がある。項目ごとにDisplayLabelプロパティやDisplayFormatプロパティ等を設定していくのは面倒な作業だが、移行の場合はすでにBDEのTTableコンポーネントのフィールドに設定されているはずなので、それをコピーすればよい。

【図16】のように、TTableコンポーネントとTFDTableコンポーネントのフィールドエディタを開き、TTableコンポーネントの全フィールドを選択したあと、コピー＆ペーストでTFDTableコンポーネントのフィールドエディタへ貼り付ける。これだけでフィールド設定情報のコピーは完了である。TFDTableコンポーネントの各フィールドのプロパティを確認すると、TTableの設定がすべてコピーできているとわかる。

再度プログラムをコンパイルして実行すると、変更しているプログラムによっては、【図17】のようなエラーの出るケースがある。これは、FireDACのデータ型（数値）のマッピングルールが、これまでのデータベースエンジンと異なる場合に発生するエラーである。ここではTFDTableコンポーネントのフィールドはInteger型であるが、FireDACは数値型をBCD型として認識するため、このようなエラーが発生する。

このエラーに対応する方法として、FireDACにはデータ型のマッピングルールを変更する機能が用意されている。データ型のマッピングルールの変更は、TFDConnectionコンポーネントで設定できる。

まずは、TFDConnectionコンポーネントをダブルクリックして、FireDAC接続エディタを開く。次にオプションタブを選択し、オプションタブにある「継承したルールを無視」チェックボックスをチェックすると、データマッピングルールの明細が入力可能になる。

明細は【図18】のように設定し、OKボタンでFireDAC接続エディタを完了する。これで、FireDACで数値項目をBCD型で認識した場合、Integer型に変換可能となる。再度プログラムをコンパイルして実行すると、データ型のエラーが解消されて、正しく実行できる。【図19】

ここまで確認できたら、FireDACで従来どおりの動作が実現できたことになる。使わなくなったBDEのTDatabaseコンポーネントとTTableコンポーネントを削除して、FireDACへの移行が完了である。

3-3. dbExpressからの移行ポイント

ここでは、dbExpressからFireDACへの変更方法について説明する。変更するdbExpressの照会画面の構成は、【図20】のとおりである。

この照会画面の構成や機能は、前述したBDEの照会画面と同じである。また、使用しているコンポーネントと設定しているプロパティについては【表3】と【図21】のとおり、検索実行時のソースについては【図22】のとおりである。

BDEとの大きな違いはデータを表示する際に、TDataSetProviderコンポーネントとTClientDataSetコンポーネントを使用している点である。これはdbExpressを使用して、直接画面にデータを表示する際に必要な構成となっている。

以下に、dbExpressを使用している箇所をFireDACに変更する手順を説明する。

①データベース接続処理をdbExpressからFireDACへ変更

前述したBDEのケースと同様に、dbExpressを使用して作成した照会画面に、FireDACの基本コンポーネントである「TFDConnection」「TFDPhysCO400DriverLink」「TFDTable」「TFDGUIxWaitCursor」の4つを、【図23】のように配置し、TFDConnectionコンポーネントのプロパティを設定する。

ただし、dbExpressの照会画面もBDEのケースと同様に、TSQLConnectionコンポーネントのプロパティ設定をソースで行っている。従ってソースの変更は、【図24】のように、TFDConnectionを設定する。これで、データベース接続処理のFireDACへの変更が完了である。

②データ表示処理をdbExpressからFireDACへ変更

dbExpressからFireDACへの変更では、TClientDataSetコンポーネントをそのまま利用することもできる。その場合、フィールド設定の移行も必要ないので、BDEより簡単にFireDACへ移行できる。

まずはBDEのケースと同様に、TFDTableコンポーネントのTableNameプロパティに参照するファイル名を直接設定する。次に、TDataSetProviderコンポーネントのDataSetプロパティをTSQLTableからTFDTableに変更する。【図25】

データ表示処理のソースはTClientDataSetコンポーネントを残しているため、変更は不要である。したがって、ここまでの作業でFireDACへの変更は完了となる。ただしプログラムを実行すると、BDEのケースで説明したデータ型のマッピングエラーが発生する可能性がある。発生する場合は、同じ対応が必要となる。

プログラムをコンパイルして実行すると、FireDACで従来どおりの動作が確認できる。最後に、使わなくなったdbExpressのTSQLConnectionコンポーネントとTSQLTableコンポーネントを削除して、FireDACへの移行は

図5

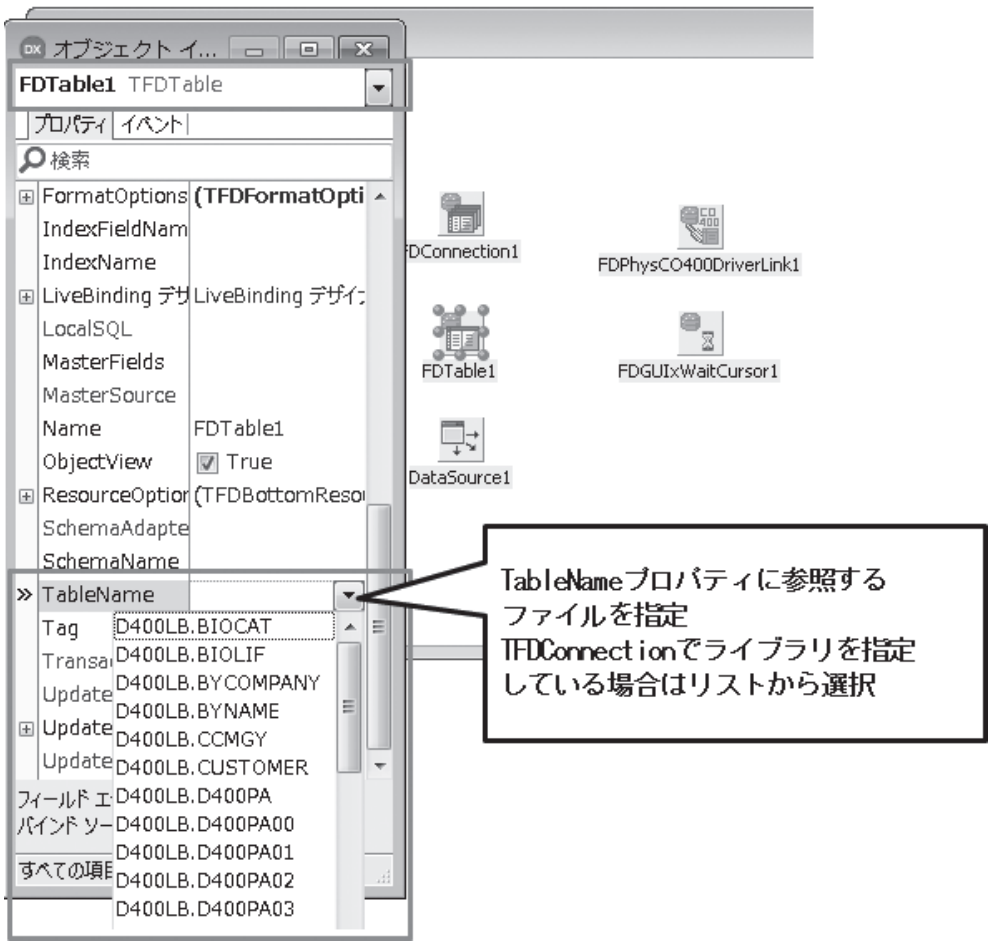

オブジェクト イ...
FDTable1 TFDTable
プロパティ
イベント
検索
FormatOptions
IndexFieldNam
IndexName
LiveBinding デザ
LocalSQL
MasterFields
MasterSource
Name
FDTable1
ObjectView
True
ResourceOptior
SchemaAdapte
SchemaName
TableName
Tag
D400LB.BIOCAT
D400LB.BIOLIF
D400LB.BYCOMPANY
D400LB.BYNAME
D400LB.CCMGY
D400LB.CUSTOMER
D400LB.D400PA
D400LB.D400PA00
D400LB.D400PA01
D400LB.D400PA02
D400LB.D400PA03
FDConnection1
FDPhysCO400DriverLink1
FDTable1
FDGUIxWaitCursor1
DataSource1
TableNameプロパティに参照する
ファイルを指定
TFDConnectionでライブラリを指定
している場合はリストから選択

図6


オブジェクト ...
DataSource1 TDataSource
プロパティ
イベント
検索
AutoEdit
True
DataSet
FDTable1
Enabled
True
LiveBinding デ
LiveBinding デザイナ
Name
DataSource1
Tag
0
FDConnection1
FDTable1
DataSource1
オブジェクト ...
DBGrid1 TDBGrid
プロパティ
イベント
検索
Constraints
(TSizeConstraints)
Ctl3D
True
Cursor
crDefault
CustomHint
DataSource
DataSource1
DefaultDrawir
True
DragCursor
crDrag
DragKind
dkDrag
DragMode
dmManual
FDConnection1
FDTable1
DataSource1

完了である。

4. まとめ

本稿ではFireDACの特徴や基本的な使用方法を確認し、BDEやdbExpressで作成されている既存プログラムからの移行ポイントを説明した。

BDEやdbExpressでプログラムを開発された経験があれば、FireDACがこれまでとほとんど同じ構成で使えることを確認いただけたと思う。既存のプログラムをFireDACへ移行する場合はいくつかのポイントがあるが、Delphi/400は非常に互換性が高いので、定型的な作業で簡単に変更できる。

本稿のノウハウを参考に、既存プログラムやこれからの新規開発でFireDACを活用していただければ幸いである。

M

図7

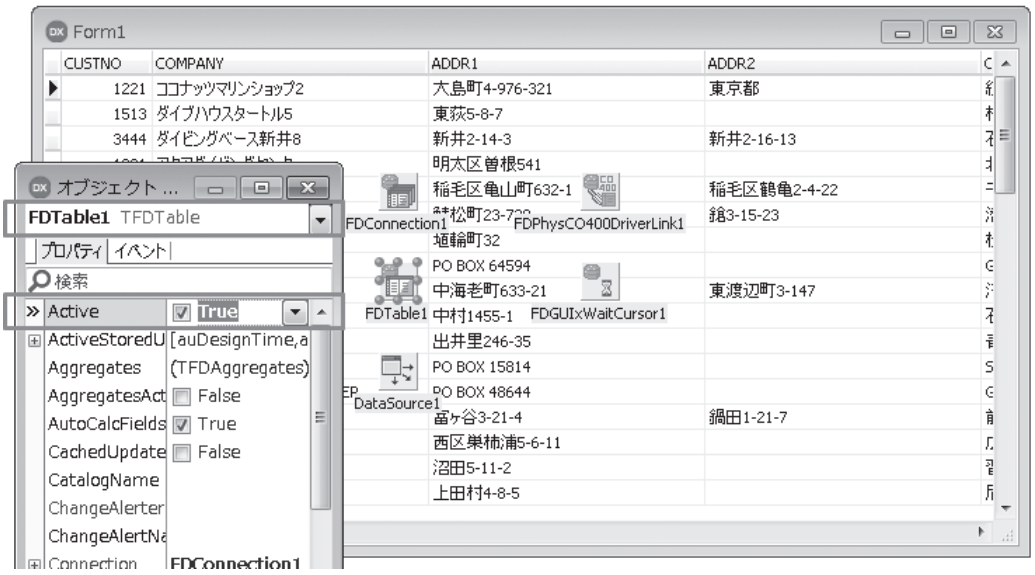


図8

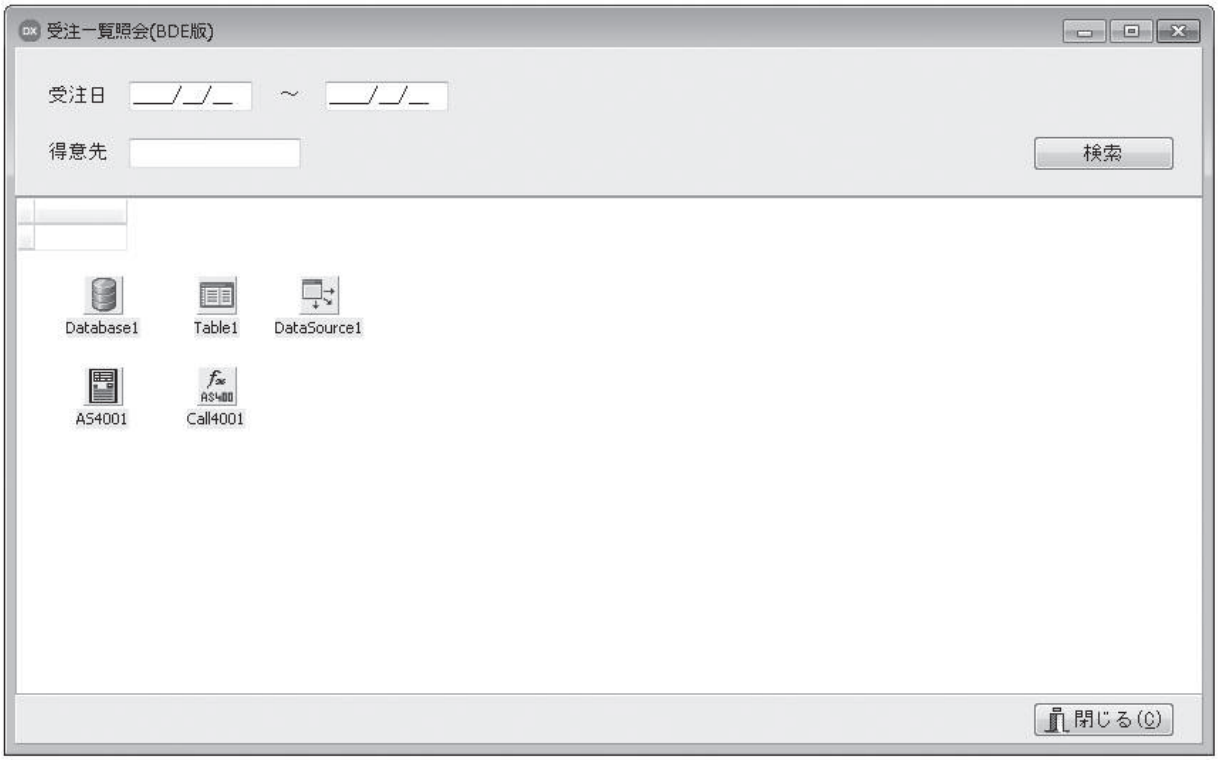
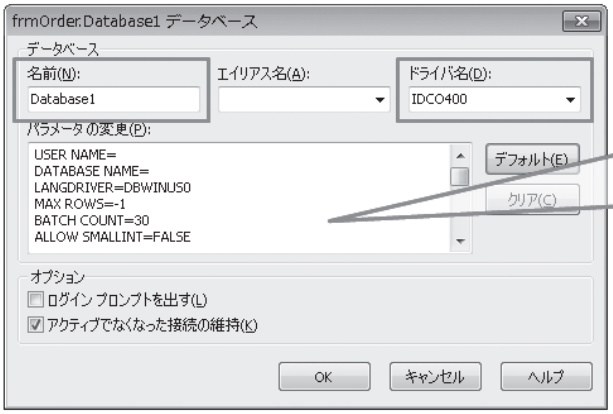


表2

BDE版照会画面　使用コンポーネント

使用コンポーネント	設定プロパティ	設定値
TDatabase	Params	【図9】参照
TTable	DatabaseName	Database1
	TableName	参照するファイル名
TDataSource	DataSet	Table1
TDBGrid	DataSource	DataSource1

図9



パラメータの設定はソースに記述（FormCreateイベント）

```
with Database1 do
begin
  Params.Values['USER NAME']     := 'USER';
  Params.Values['PASSWORD']      := 'PASSWORD';
  Params.Values['DATABASE NAME'] := 'POWER7';
  Params.Values['LIBRARY NAME']  := '*LIBL';
end;
```

図10

```
if Table1.Active then Table1.Close;
```

データ抽出前にTTableを閉じる

```
with Call4001 do
begin
  Value[0] := MaskEdit1.Text; // 受注日From
  Value[1] := MaskEdit2.Text; // 受注日To
  Value[2] := Edit1.Text;     // 得意先CD
  Value[3] := '';             // エラーCD
  Value[4] := '';             // エラーメッセージ

  Execute;

  // エラー処理
  if Value[3] <> '' then
  begin
    MaskEdit1.SetFocus;
    raise Exception.Create(Value[4]);
  end;
end;
```

データ抽出処理
TCall400コンポーネントを使用して
CLをCallし、QTEMPのワークファイル
へデータを出力

```
Table1.Open;
```

データ抽出後にTTableを開き
画面にデータを表示

図11

図12

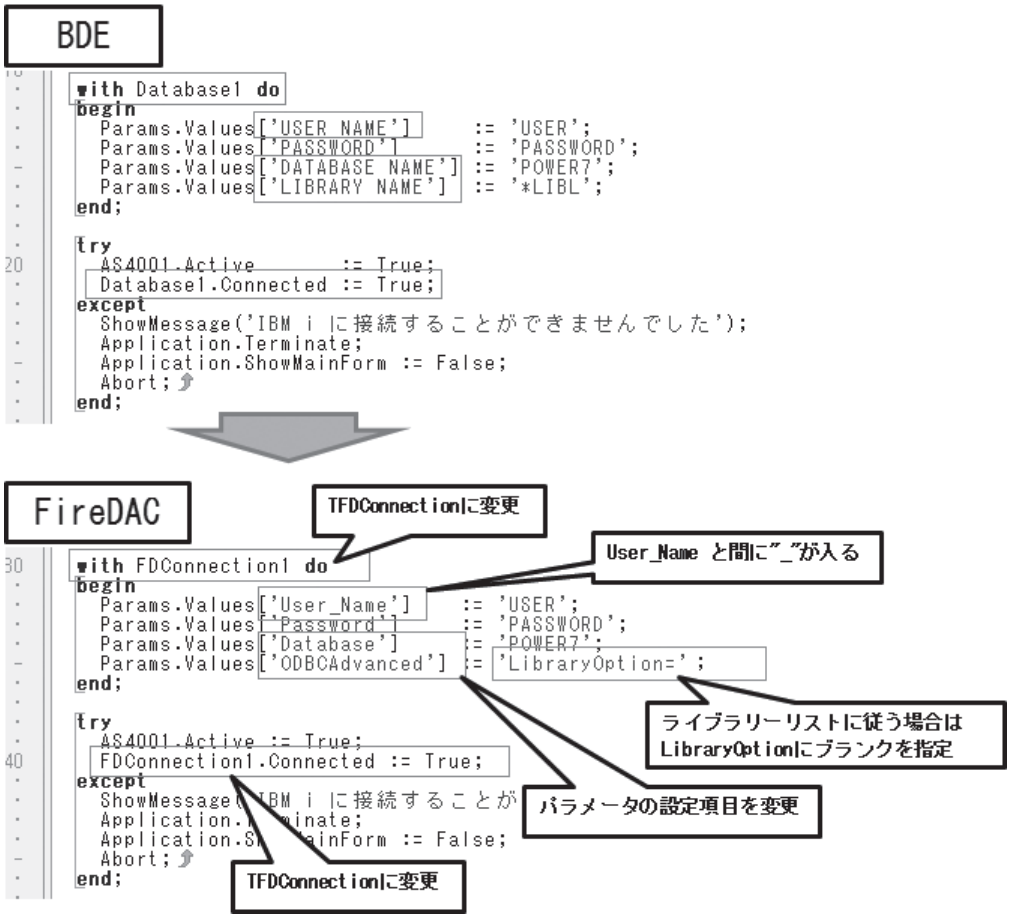

BDE
with Database1 do
begin
Params.Values['USER NAME'] := 'USER';
Params.Values['PASSWORD'] := 'PASSWORD';
Params.Values['DATABASE NAME'] := 'POWER7';
Params.Values['LIBRARY NAME'] := '*LIBL';
end;
try
AS400.Active := True;
Database1.Connected := True;
except
ShowMessage('IBM i に接続することができませんでした');
Application.Terminate;
Application.ShowMainForm := False;
Abort;
end;
FireDAC
TFDConnectionに変更
User_Name と間に"_"が入る
with FDConnection1 do
begin
Params.Values['User_Name'] := 'USER';
Params.Values['Password'] := 'PASSWORD';
Params.Values['Database'] := 'POWER7';
Params.Values['ODBCAdvanced'] := 'LibraryOption=';
end;
try
AS400.Active := True;
FDConnection1.Connected := True;
except
ShowMessage('IBM i に接続することが
Application.Terminate;
Application.ShowMainForm := False;
Abort;
end;
ライブラリーリストに従う場合は
LibraryOptionにブランクを指定
パラメータの設定項目を変更
TFDConnectionに変更

図13

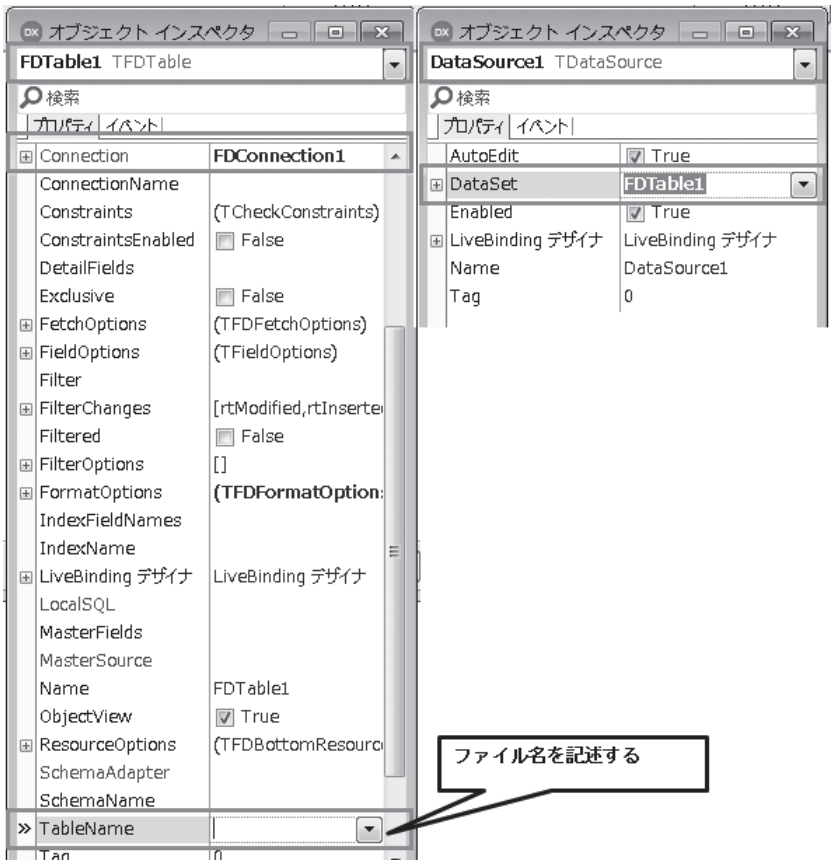

オブジェクト インスペクタ
FDTable1 TFDTable
検索
プロパティ イベント
Connection FDConnection1
ConnectionName
Constraints (TCheckConstraints)
ConstraintsEnabled False
DetailFields
Exclusive False
FetchOptions (TFDFetchOptions)
FieldOptions (TFieldOptions)
Filter
FilterChanges [rtModified,rtInserte
Filtered False
FilterOptions []
FormatOptions (TFDFormatOption
IndexFieldNames
IndexName
LiveBinding デザイナ LiveBinding デザイナ
LocalSQL
MasterFields
MasterSource
Name FDTable1
ObjectView True
ResourceOptions (TFDBottomResourc
SchemaAdapter
SchemaName
TableName
オブジェクト インスペクタ
DataSource1 TDataSource
検索
プロパティ イベント
AutoEdit True
DataSet FDTable1
Enabled True
LiveBinding デザイナ LiveBinding デザイナ
Name DataSource1
Tag 0
ファイル名を記述する

図14

BDE

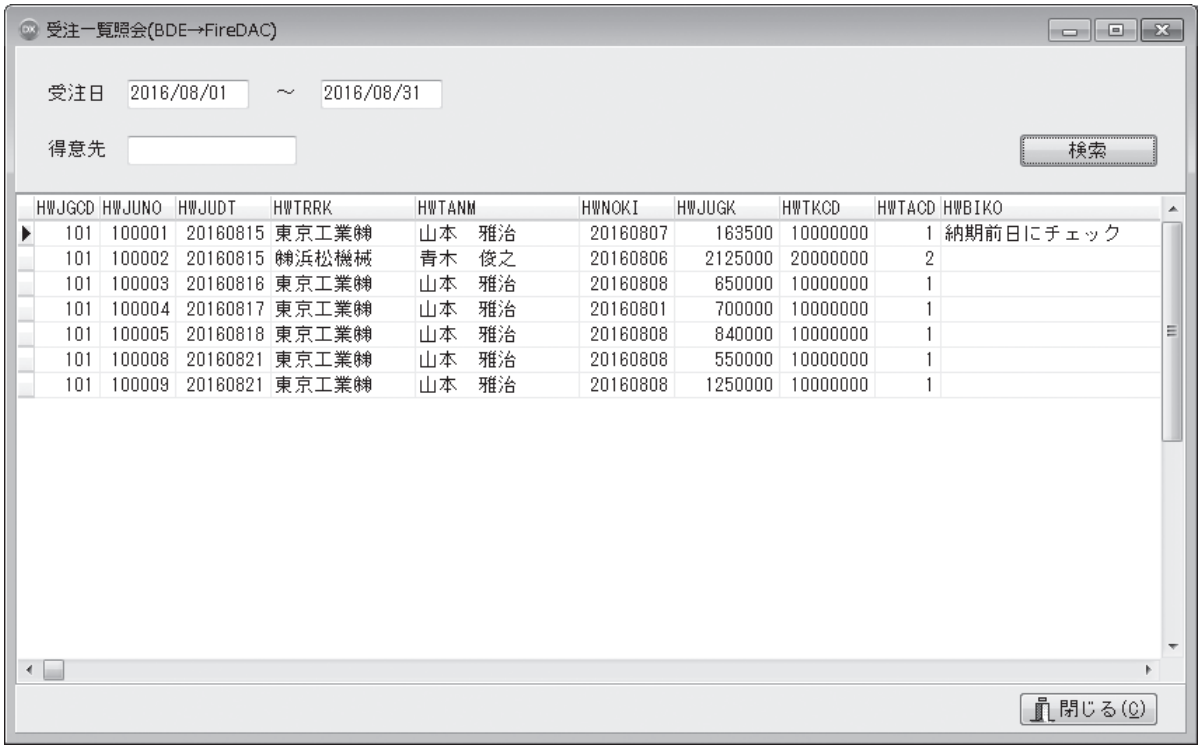
```
if Table1.Active then Table1.Close;

with Call4001 do
begin
  Value[0] := MaskEdit1.Text; // 受注日From
  Value[1] := MaskEdit2.Text; // 受注日To
  Value[2] := Edit1.Text;     // 得意先CD
  Value[3] := '';             // エラーCD
  Value[4] := '';             // エラーメッセージ

  Execute;

  // エラー処理
  if Value[3] <> '' then
  begin
    MaskEdit1.SetFocus;
    raise Exception.Create(Value[4]);
  end;
end;

Table1.Open;
```

TTableを使用している部分をTFDTableに変更

FireDAC

```
if FDTable1.Active then FDTable1.Close;

with Call4001 do
begin
  Value[0] := MaskEdit1.Text; // 受注日From
  Value[1] := MaskEdit2.Text; // 受注日To
  Value[2] := Edit1.Text;     // 得意先CD
  Value[3] := '';             // エラーCD
  Value[4] := '';             // エラーメッセージ

  Execute;

  // エラー処理
  if Value[3] <> '' then
  begin
    MaskEdit1.SetFocus;
    raise Exception.Create(Value[4]);
  end;
end;

FDTable1.Open;
```

図15



HWJGCD	HWJUNO	HWJUDT	HWTRRK	HWTANM	HWNOKI	HWJUGK	HWTKCD	HWTACD	HWBIKO
101	100001	20160815	東京工業㈱	山本　雅治	20160807	163500	10000000	1	納期前日にチェック
101	100002	20160815	㈱浜松機械	青木　俊之	20160806	2125000	20000000	2	
101	100003	20160816	東京工業㈱	山本　雅治	20160808	650000	10000000	1	
101	100004	20160817	東京工業㈱	山本　雅治	20160801	700000	10000000	1	
101	100005	20160818	東京工業㈱	山本　雅治	20160808	840000	10000000	1	
101	100008	20160821	東京工業㈱	山本　雅治	20160808	550000	10000000	1	
101	100009	20160821	東京工業㈱	山本　雅治	20160808	1250000	10000000	1	

図16

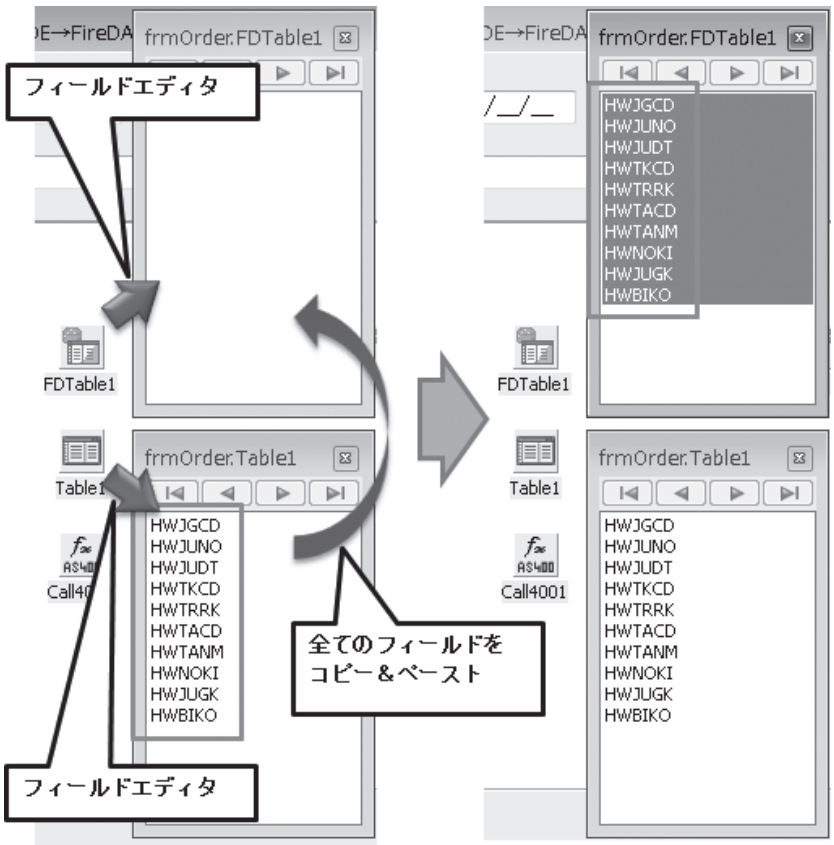

フィールドエディタ
frmOrder.FDTable1
frmOrder.Table1
FDTable1
Table1
Call4001
HWJGCD
HWJUNO
HWJUDT
HWTKCD
HWTRRK
HWTACD
HWTANM
HWNOKI
HWJUGK
HWBIKO
全てのフィールドを
コピー＆ペースト
フィールドエディタ

図17

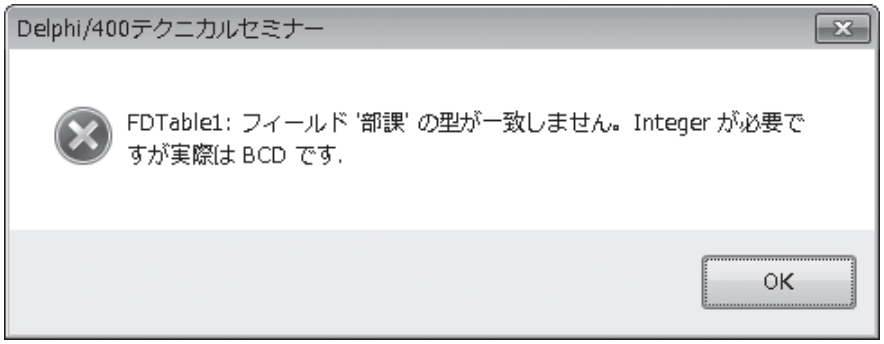

Delphi/400テクニカルセミナー
FDTable1: フィールド '部課' の型が一致しません。Integer が必要で
すが実際は BCD です.
OK

図18


FireDAC 接続エディタ - [FDConnection1]
定義
オプション
情報
SQL スクリプト
形式オプション
データ マッピング ルール
継承したルールを無視(G)
SourceDataType
TargetDataType
PrecMin
dtBCD
dtInt32

図19

部課	受注No.	受注日	得意先CD	得意先名	担当CD	営業担当者名	納期	受注金額	備考
101	100001	2016/08/15	10000000	東京工業㈱	1	山本　雅治	2016/08/07	163,500	納期前日にチェック
101	100002	2016/08/15	20000000	㈱浜松機械	2	青木　俊之	2016/08/06	2,125,000	
101	100003	2016/08/16	10000000	東京工業㈱	1	山本　雅治	2016/08/08	650,000	
101	100004	2016/08/17	10000000	東京工業㈱	1	山本　雅治	2016/08/01	700,000	
101	100005	2016/08/18	10000000	東京工業㈱	1	山本　雅治	2016/08/08	840,000	
101	100008	2016/08/21	10000000	東京工業㈱	1	山本　雅治	2016/08/08	550,000	
101	100009	2016/08/21	10000000	東京工業㈱	1	山本　雅治	2016/08/08	1,250,000	

図20

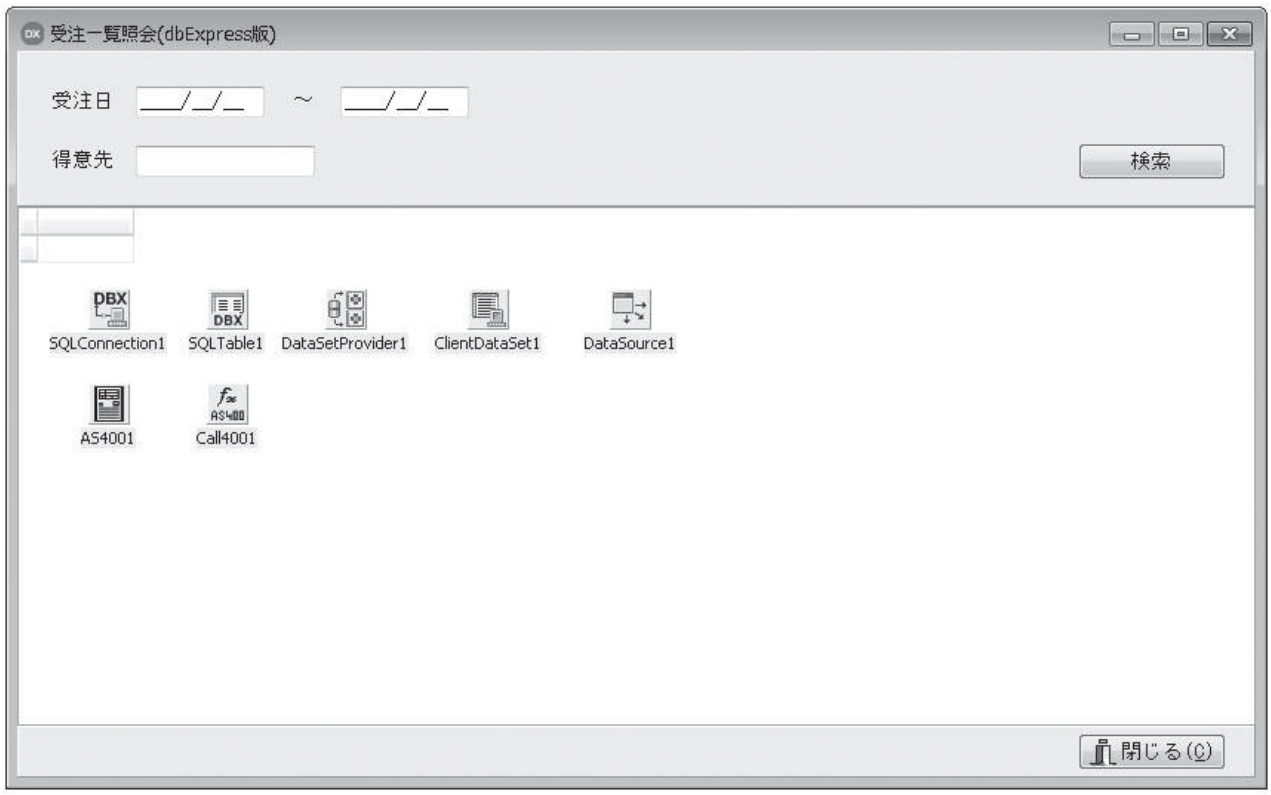


表3

dbExpress版照会画面　使用コンポーネント

使用コンポーネント	設定プロパティ	設定値
TSQLConnection	Params	【図21】参照
TSQLTable	SQLConnection	SQLConnection1
	TableName	参照するファイル名
TDataSetProvider	DataSet	SQLTable1
TClientDataSet	ProviderName	DataSetProvider1
TDataSource	DataSet	ClientDataSet1
TDBGrid	DataSource	DataSource1

図21

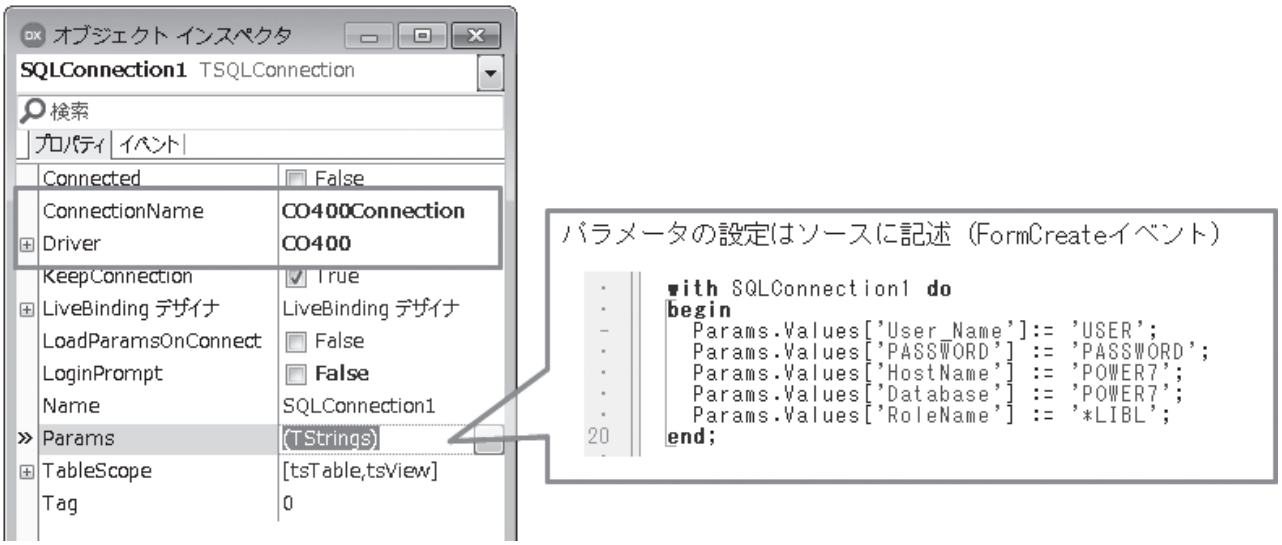


図22

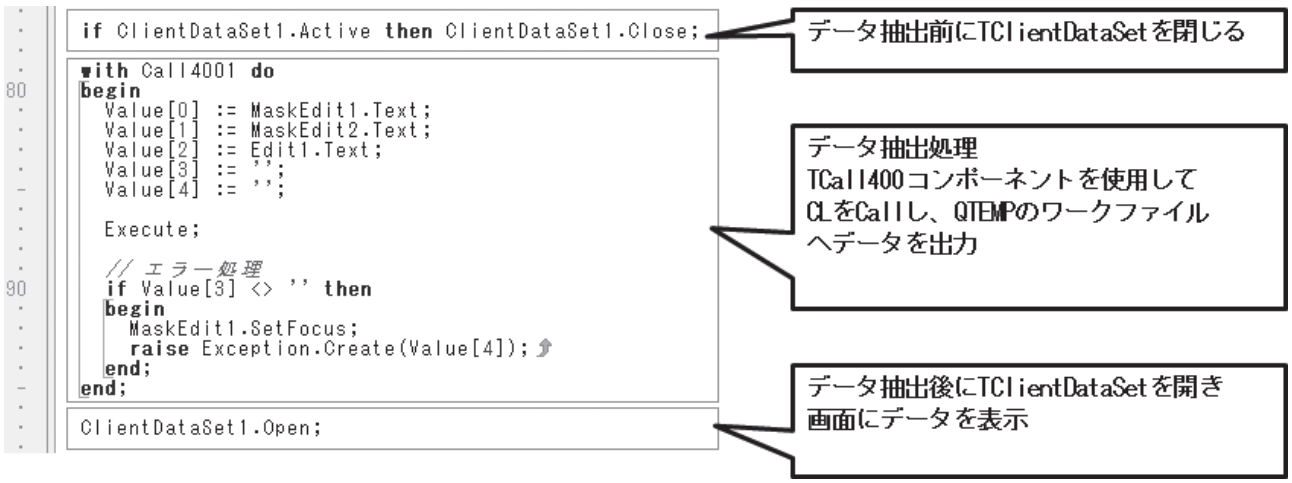


図23

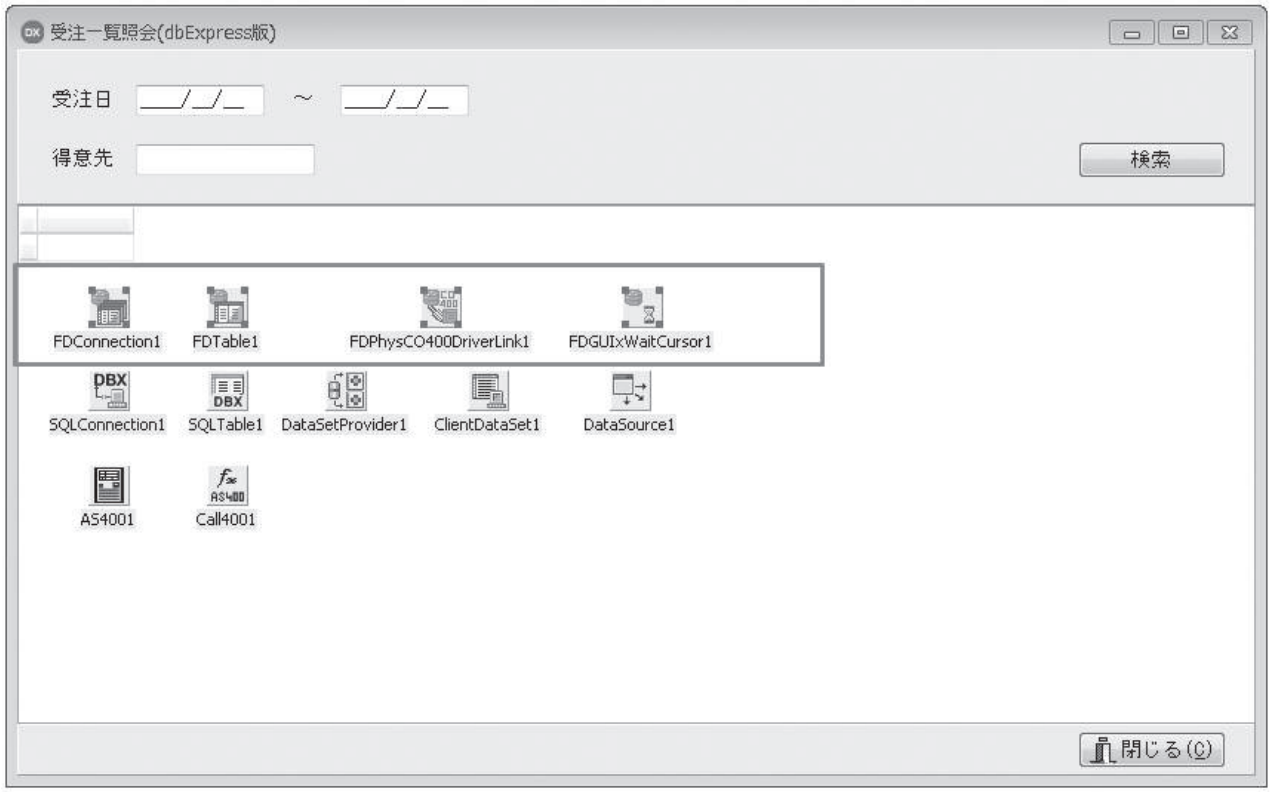


図24

dbExpress

```
with SQLConnection1 do
begin
  Params.Values['User_Name']:= 'USER';
  Params.Values['PASSWORD'] := 'PASSWORD';
  Params.Values['HostName'] := 'POWER7';
  Params.Values['Database'] := 'POWER7';
  Params.Values['RoleName'] := '*LIBL';
end;

try
  AS4001.Active := True;
  SQLConnection1.Connected := True;
except
  ShowMessage('IBM i に接続することができませんでした');
  Application.Terminate;
  Application.ShowMainForm := False;
  Abort;
end;
```

FireDAC

TFDConnectionに変更

パラメータの"HostName"は
FireDACでは不要

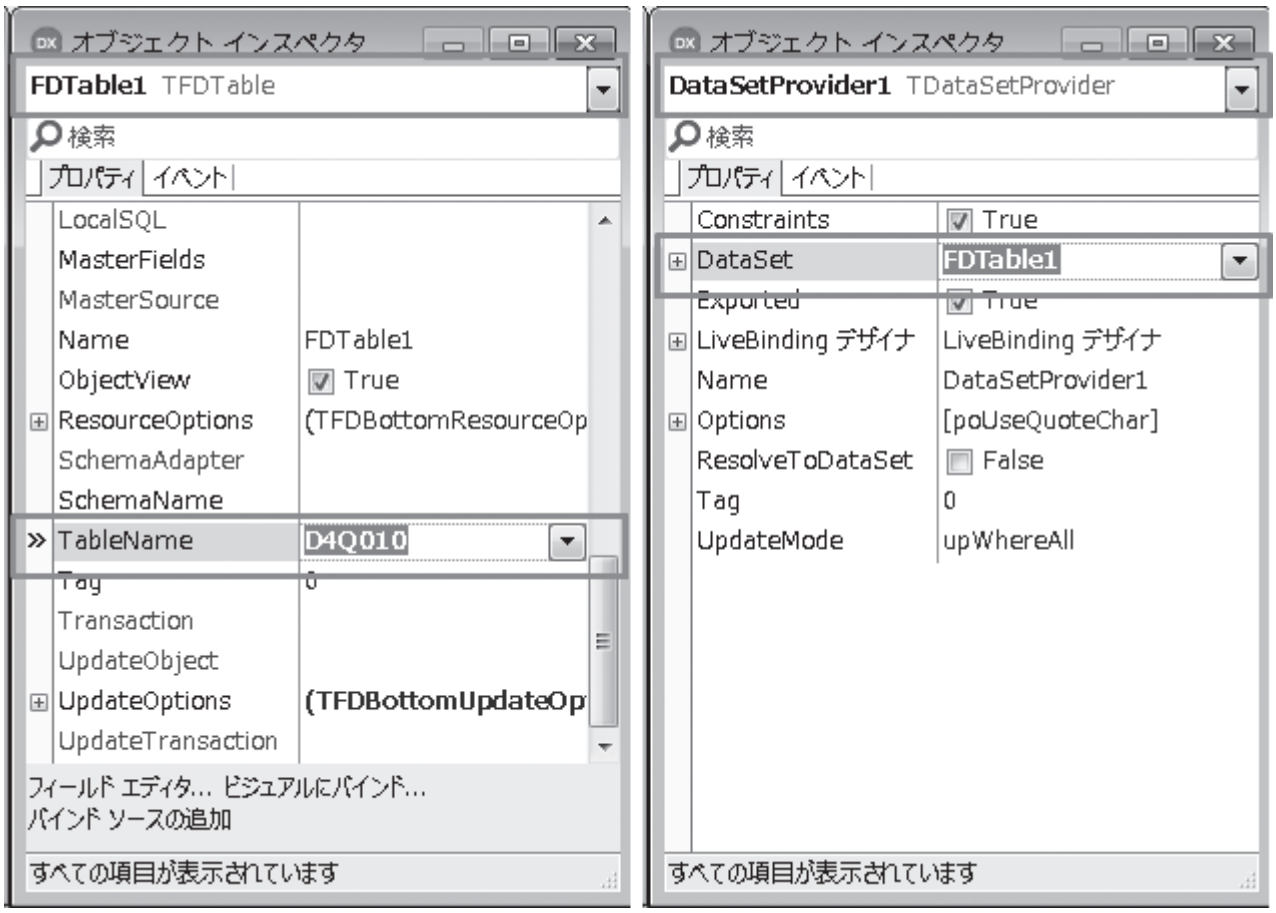
```
with FDConnection1 do
begin
  Params.Values['User_Name']     := 'USER';
  Params.Values['Password']      := 'PASSWORD';
  Params.Values['Database']      := 'POWER7';
  Params.Values['ODBCAdvanced'] := 'LibraryOption=''''';
end;

try
  AS4001.Active := True;
  FDConnection1.Connected := True;
except
  ShowMessage('IBM i に接続することが
  Application.Terminate;
  Application.ShowMainForm := False;
  Abort;
end;
```

ライブラリーリストに従う場合は
LibraryOptionにブランクを指定

パラメータの設定項目を変更

TFDConnectionに変更

図25


オブジェクト インスペクタ
FDTable1 TFDTable
検索
プロパティ
イベント
LocalSQL
MasterFields
MasterSource
Name
FDTable1
ObjectView
True
ResourceOptions
(TFDBottomResourceOp
SchemaAdapter
SchemaName
TableName
D4Q010
Tag
0
Transaction
UpdateObject
UpdateOptions
(TFDBottomUpdateOp
UpdateTransaction
フィールド エディタ...
ビジュアルにバインド...
バインド ソースの追加
すべての項目が表示されています
オブジェクト インスペクタ
DataSetProvider1 TDataSetProvider
検索
プロパティ
イベント
Constraints
True
DataSet
FDTable1
LiveBinding デザイナ
LiveBinding デザイナ
Name
DataSetProvider1
Options
[poUseQuoteChar]
ResolveToDataSet
False
Tag
0
UpdateMode
upWhereAll
すべての項目が表示されています