Delphi/400

FireDACを活用した Delphi/400ロジック最新化テクニック

株式会社ミガロ. RAD事業部技術支援課 雄— 佐田



略歴 生年月日:1985年12月6日 最終学歴:2009年 甲南大学 経営学部卒業 ミガロ入社年月:2009年04月 株式会社ミガロ、入社 社内経歴: 2009年04月 システム事業部配属 2019年04月 RAD事業部配属

現在の仕事内容:

Delphi/400を利用した システム開発や保守作業の経験を経て、 現在はDelphi/400のサポート業務を担当。

1.はじめに

- 2. TClientDatasetに代わる データセットTFDMemTableの活用術
 - ① TFDMemTableとは
 - TFDMemTableを使ったデータ照会方法
 - ③ TFDMemTableを使ったデータ更新方法
 - ④ TFDMemTableを使ったローカルDB活用
- 3. TSpool400に代わる

SQLメインのスプール活用術

①スプールのリスト表示

②個別スプールの取得

③取得したスプールの操作

4. まとめ

1.はじめに

Delphi/400 Version 5が2000年に日本で販売を開始して から早くも20年が経過したが、実は当時から存在するロジッ クの大半が現在の最新バージョン(Delphi/400 10.2 Tokyo)でも利用することができる。しかし当時はWindows 95や98がまだ現役だった頃で、その後のWindows OS側の 技術革新で、初期のコンポーネントやロジックの中には動作 にあたって制約が生じているものも存在する。

本稿ではSQL等のIBM i側の資産や、最新のFireDAC接続 を用いて、最新バージョンでさらに使いやすくなったコン ポーネントに関するトピックと、先述の動作に制約が発生し ているソースをリフレッシュするためのトピックの2つを紹介 する。

なお、本稿ではDelphi/400 10.2 TokyoおよびV7R1以上 のIBM i環境を使用している。

2.TClientDatasetに代わるデータセットTFDMemTableの活用術

① TFDMemTableとは

IBMiのデータベースファイルからデータを参照して表形式 FireDAC接続でのデータキャッシュにおいては後継となる TFDMemTableコンポーネントの使用を推奨している。 で表示する際に、もっとも一般的なコンポーネントの配置 順は「データセットコンポーネント(TFDQuery・ dbExpress接続では単方向での通信だったものが FireDAC接続においては双方向で通信可能なため、 TSQLQuery等) ⇒TDataSetProvider ⇒TClientDataSet FireDAC接続ではTClientDataSetを経由させずに ⇒TDataSource⇒TDBGrid となるだろう。 「TFDQuery⇒TDataSource⇒TDBGrid」といったコン ここで使用するTClientDataSetコンポーネントは取得し たデータを内部メモリでキャッシュするための汎用インメモ ポーネントの配置が可能である。一方、旧バージョンにおい リデータセットで、単独使用やdbExpress接続における てBDE接続やdbExpress接続で作成されたアプリケーショ ンをFireDAC接続に移行する場合には、元々 キャッシュ機能の実現の為に用いられる。最新の Delphi/400においてもTClientDataSetや TClientDataSetで使用されていた内部計算項目の使用 や、古いParadoxに代わるローカルデータベースとしての活 TDataSetProviderといったコンポーネントをこれまで通 用がTFDMemTableコンポーネントによって可能になって り使用することは可能であるが、古いBDE接続や dbExpress接続のアーキテクチャにあわせて作られており、 いる。 次項より、FireDAC接続のために最適化された新しいデー Version 5の頃から基本的な仕様はそのままとなっている。 また、Delphi開発元のエンバカデロ・テクノロジーズ社は今 タセット、TFDMemTableの使用方法について紹介する。 後の更新が行われる可能性は限りなく低いとしており、

TFDMemTableを使ったデータ照会方法

本項ではTClientDataSet・TDataSetProviderの代替と の各データベースコンポーネントおよび、画面操作に使用 して、TFDMemTableを使ってデータを取得する手順を するEditやButton等を配置する【図1】。TFDConnection 紹介する。 を使ったFireDAC接続の設定自体は過去のテクニカルレ Delphiを起動してプロジェクトを新規作成したら、まず ポートでも取り上げているため、本稿では割愛する。 フォーム上にFireDAC接続を行うために必要な (2018年のテクニカルレポートに掲載の拙著「FireDAC TFDConnection · TFDPhysCO400DriverLinkと、 実践プログラミングテクニック」を参照されたい。)

TFDQuery • TFDMemTable • TDataSource • TDBGrid

サンプルの画面設計



	ApplyUpdates FDMemTable OPEN
	ApplyUpdates FDMeeTable OPEN
	FDMemTable OPEN
rce1	

Delphi/400 畑中 侑

Delpl 福井

hi/400 和彦

Ъ

E

踏

Ð

存 日 日

Sma 醌 元

rtPad4i 枯二

Vale 尾崎

w c

TDBGridのDataSourceプロパティにTDataSourceを設 定するのは従来と同様だが、TDataSourceのDataSetプロ パティにはTClientDataSetの代わりにTFDMemTableを 設定する【図2】。また、オブジェクトインスペクタ上のプロパ ティではTFDMemTableとTFDQueryを接続できる箇所 がないため、代替としてロジックを使って、TFDQueryで取 得したデータをTFDMemTableへ連携する。連携する方法 はいくつか存在する【ソース1】【ソース2】。

図 2 データセット指定



ソース1

CloneCursorを使ったTFDMemTableでのデータ参	照
----------------------------------	---

FDQuery1.Close; FDMemTable1.Close; FDQuery1.SQL.Text := 'SELECT * FROM TESTTBL21 '; FDQuery1.Open; FDMemTable1.CloneCursor(FDQuery1, False, False);



Dataの代入を使ったTFDMemTableでのデータ参照

FDQuery1.Close; FDMemTable1.Close; FDQuery1.SQL.Text := 'SELECT * FROM TESTTBL21 '; FDQuery1.Open; FDMemTable1.Data := FDQuery1.Data; 【ソース1】はCloneCursorメソッドを使用する方法で、 【ソース2】はDataプロパティを代入する方法である。本項 ではこの2パターンを例示したが、それぞれの特徴や利点 を紹介する。

ではこの2パターンを例示したが、それぞれの特徴や利点 照会したデータはTClientDataSetと同じように を紹介する。 IndexDefs・IndexNameプロパティによるローカルでの いずれの例においても事前にTFDQueryをオープンして 並び替えや、後述する JSON 形式でのデータ出力など TFDQuery(またはTFDTable、以下同じ)では実現でき おく点は同じであるが、【ソース1】でCloneCursorメソッ ない機能が存在する。なお、TClientDataSetにも同様に ドを行う場合は、パラメータにデータ取得元となる TFDQueryを指定することで、そのTFDQueryにある コピーが可能な「Data」プロパティが存在しているが、 データのポインタのみを複製し、内部的なデータは同じも TFDMemTableのDataとは内部の型が異なるため互換 のを共有するイメージとなる。 性はなく、直接代入するようなロジックを記述するとコン 対して【ソース2】でDataを代入する方法ではその時点で パイルエラーになる。TClientDataSetから 取得済みのデータをそのままTFDMemTableにコピーす TFDMemTableへのデータ同期が必要な場合は るイメージとなる。そのため、TFDQuery側で CopyDataSetメソッドを使うと、異なる接続方式のデー FetchOptionsプロパティの設定によって最初の50件ま タセットからTFDMemTableにデータを転送することが でしかレコードを取得していない場合、TFDMemTable できる【ソース3】。パラメータに同じFireDAC接続の にコピーされるレコードもその件数のみとなる。たとえば TFDQueryを指定することも可能で、この場合はDataを 元のデータの件数が非常に多く時間がかかる場合に 代入する際と同じ挙動になる。 TFDQuery側でレコードの取得件数を制限する際には、

ソース3

他種のデータセットからTFDMemTableへのデータコピー

// [例] TClientDataSetでローカルデータを読み込み ClientDataSet1.LoadFromFile('C:¥~~~¥DATA.xml'); // CopyDataSetでデータコピーするとTFDMemTableでも扱える FDMemTable1.CopyDataSet(ClientDataSet1, [coStructure, coRestart, coAppend]);

044 MIGARO Technical Report

CloneCursorメソッドを使用する場合よりも処理の高速 化が期待できる。





Delphi/400 俎中 侑



③ TFDMemTableを使ったデータ更新方法

BDE接続・dbExpress接続においてはTClientDataSetの ApplyUpdatesメソッドを使うことで、接続されたQueryコン ポーネント経由でインメモリデータセットの変更内容をIBM i に更新できたが、TFDMemTableにおいてもロジックが少々 異なるものの、ApplyUpdatesメソッドで同様の更新が可能 である。

FireDAC接続においてはTFDQueryでも紐付くTDBGridを 直接編集することや、Edit~Postのメソッドを行うことで内部 的に更新SQLを発行し、接続先のDBに更新を行うことができ る。(更新自体を可能にするためのUpdateOptionsプロパ ティの設定については、前章でも紹介した2018年のテクニカ ルレポートに掲載の拙著「FireDAC 実践プログラミングテク ニック」を参照されたい。)

このとき、更新に使用するTFDQueryのCachedUpdatesプロパティがFalseならPostやDeleteの時点で即時更新が行われ、Trueなら更新情報を内部でキャッシュし、ApplyUpdates メソッドを行った時点で一括更新が行われる。

BDE接続・dbExpress接続との違いとしては更新先ファイル へのジャーナル開始やトランザクション制御が不要な点が挙 げられるが、複数件のデータを更新している途中でエラーに なった場合のリスクを考慮すれば、従来通りジャーナルやトラ ンザクション制御を使用しておいた方が安全である。

TFDMemTableを使用してApplyUpdatesメソッドを実行 する際の前提条件としては、以下の3点があげられる。 ・前項で紹介した2種類の手順のうち、【ソース1】の CloneCursorメソッドを使ってデータを取得している。 ・ApplyUpdatesメソッドを実際に行うのは紐づいている TFDQueryとなるため、ApplyUpdates実行時まで TFDQueryが開いたままになっている。(TFDMemTable側 でApplyUpdatesメソッドを実行しても更新は反映されない) ・キャッシュ更新を有効にするため、先述のTFDQueryの CachedUpdatesプロパティがTrueになっている。 これらの条件が揃った状態でTFDMemTableのデータを変

更し、【ソース4】のようなロジックでApplyUpdatesメソッド を実行すると、Open時からキャッシュされた変更がIBM iに 反映される。 前項で紹介したデータ取得手順のうちCloneCursorメ Setメソッドを使用する手順の場合は別々のデータとなる ソッドを使う手順の場合はTFDQueryと同じデータを内 【図3】ため、ApplyUpdatesメソッドを行ってもIBM iに更 部で共有しているためApplyUpdatesメソッドによって更 新は反映されない。 新が反映されるが、Dataをコピーする手順やCopyData



④TFDMemTableを使ったローカルDB活用

TFDMemTableはTClientDataSetと同じように、 LoadFromFileメソッドやSaveToFileメソッドで外部 データを読み込んだり書き出したりすることができる。例 えば処理速度向上のためローカルPC上に値が長期間変 わることのないマスタファイルのデータをコピーしておき たい場合に有効である。

ソース 5



046 MIGARO Technical Report

ソース 4

TFDMemTableのデータ更新 // 紐づくTFDQuery側でApplyUpdatesする

FDQuery1. ApplyUpdates (-1);

去の違い	
ため、	
は別々の を変更しても	

広張子で判断) イナリ形式 式		
	MIGARO Technical Report)47

Delphi/400	Delphi/400	Delphi/400	SmartPad4i	Valence
始中 街	緬井 哲凚	佐田 雄	國元祐二	黒崎 浩 司
留中。	福井 档顾 石田 踏由	府田 羅	國元祝二	調用

TClientDataSetとの違いとしては、保存形式によって出力 に使用するコンポーネントの配置が必要になること【図4】 と、出力形式の違いである。前項で紹介したDataプロパティ と同様にTFDMemTableで書き出すデータと TClientDataSetで書き出すデータの内容では大きく異な

るため互換性は無いが、TFDMemTableではXML形式に 加えて|SON形式でも出力することができる【図5-1~3】。 TClientDataSetでXMLデータを出力した場合と同様に、こ れらを他のプログラムやデータベースで流用することが可 能である。

図 4 TFDMemTable保存に必要なコンポーネント

ツール パレット	平 32
Storage	×
FireDAC Links	
TFDStanStorageBinLink	∃ TFDStanStorageBinLink :専用バイナリ形式用
TFDStanStorageXMLLink	TFDStanStorageXMLLink :XML形式用
Fon TFDStanStorageJSONLink	TFDStanStorageJSONLink:JSON形式用
BAAS Client	
TD	

図 5-1 TFDMemTableの出力前データ

SHSHCD	XX01 SHS	HBT S	SHSHNM	SHMKNM	SHSIZE	SHTANI	SHIRSU	^
101148	101148 クラ	ラブ し	LEGACY BLACKアイアン(6本)ダイナミックゴールドシャフト	キャロウェイ	R	セット		
102079	102079 クラ	ラブ 2	2014 オノフ ドライバー(赤ドライバー) MP-514Dシャフト	㈱ミガロ.		本		
102082	102082 クラ	ラブ は	2014 オノフ TYPE-S (黒)ドライバー MP-614Dシャフト	㈱ミガロ.	AA	本		
ABCDEFGHI	JABCDEF クラ	ラブ ほ	BIG BERTHA BETA ドライバー TYPE 2★	キャロウェイ		本		
ABCDEFGHI	6 ABCDEF クラ	ラブ【新】 💈	2015 ONOFF フェアウェイ;WINGS KURO(ユーティリティー)	あアアア	4444	本		
105567	105567 クラ	ラブ	IG BERTHA BETA アイアン(5本セット)	キャロウェイ		セット		
105974	105974 クラ	ラブ は	2015 BIG BERTHA アルファ815 ダブルダイヤモンド ドライバ	キャロウェイ	141	本		
106516	106516 クラ	ラブ・	JPX850 フォージドアイアン(6本セット)	ミズノ	141	セット		
106632	106632 クラ	ラブ・	JPX850 ドライバー オロチパワーマキシマイザー	ミズノ		本		
106655	106655 クラ	ラブ は	2015 R15 460 ドライバー ATTAS6☆ 6カーボンシャフト	テーラーメイド		本		
107301	107301 クラ	ラブ は	2015 XR ドライバーTOUR AD MJ-6 シャフト	キャロウェイ	252	本		
107315	107315 クラ	ラブ は	2015 XR ユーティリティー XR NS950スチール シャフト	キャロウェイ	252	本		
ABCDEFGHI	4 ABCDEF クラ	ラブ は	2015 VOLTIO KABUTO 103 パター	カタナゴルフ		本		
107926	107926 クラ	ラブ は	2015 FUTURA X5 デュアルバランス パター	スコッティキャメロン		本		
108036	108036 クラ	ラブ は	2015 AMERICAN CHANPIONSHIP パター RJB6823	ベティナルディー	363	本		
108057	108057 クラ	ラブ し	LUCKY 777 #7 SB(シングルベント) パター	オデッセイ		本		
108091	108091 クラ	ラブ は	2015 FEMINA(フェミナ) パター TX-415Pシャフト	ヤマハ		本		
108125	108125 クラ	ラブ は	2015 FEMINA(フェミナ) ユーティリティ TX-415Uシャフト	ヤマハ		本		

TFDMemTableのXML出力サンプル 図 5-2

ļ	© FDMemIami X +
	← → C û © 774/k C/Projects/TechRep2021/Tokyo_FDMemTable/FDMem2.sml
	This XML file does not appear to have any style information associated with it. The document
	▼ <fdbs version="15"> ▼(Manager UpdatesRegistry="True"></fdbs>
	* <tablelist> *<table datatype="AnsiString" enforceconstraints="Factors and the second second</th></tr><tr><th></th><th><pre>*Column.ist>
<Column Name=" name="FDQuery2" search<br="" shshod"="" size="10" sourceid="1" sourcename="SHSHCD" tabld="0">Default="True" Expr_"True" OlnUpdgte="True" OlnNhere="True" OlnKey="True" OAfterInsChanged="Tr Default="True" Expr_"True" OlnUpdgte="True" OlnNhere="True" OlnKey="True" OAfterInsChanged="True" OlnKey="True" OlnKe</table></tablelist>
	<pre>Sourcertectston= to "Sourcestze= 10 // // SourcelD="2" DataType="AnsiString" Size="14" Searcha <column datatype="AnsiString" name="StRBHT" searcha<br="" size="14" sourceld="2" sourcename="StRBHBT">Default="True" Expr="True" OlnUpdate="True" OlnWhere="True" OAfterInsChanged="True" OAfterUpdo SourcePareiism" 14" SourceSize="14" SourcePareiism" SourceID="2" DataType="AnsiString" Size="14" Searcha SourcePareiism" 14" SourceSize="14" SourceID="2" OAfterInsChanged="True" OAfterUpdo"</column></pre>
	<pre>Column Name="SKSNMM" SourceName="SKSNMM" SourceID="3" DataType="AnsiString" Size="62" Searche Default="True" Expr="True" OlnUpdate="True" OlnNhere="True" OAfterInsChanged="True" OAfterUpdC SourcePrecision="62" SourceSize="62" Column SourcePrecision" SourcePrecision="62" SourceNameSize="62" SourcePrecision" SourcePrecision="62" SourcePreci</pre>
	<column datatype="AnsiString" name="SHMKNM" searcha<br="" size="26" sourceld="4" sourcename="SHMKNM">Default="True" Expr="True" Oligodate="True" OlimNere="True" OAfterInsChanged="True" OAfterUpd SourcePrecision="26" SourceSize="26" /></column>
	<column datatype="AnsiString" name="SHSIZE" searcha<br="" size="14" sourceld="5" sourcename="SHSIZE">Default="True"ExperiTrue"OlUNDdate="True"OlNMbere="True"OAfterInsChanged="True"OAfterUpd SourcePrecision="14" SourceSize="14"/></column>
	<pre><column datatype="AnsiString" name="SHTANI" searcha<br="" size="14" sourceld="6" sourcename="SHTANI">Default="True" Expr="True" OlnUpdate="True" OlnWhere="True" OAfterInsChanged="True" OAfterUpdd SourcePrecision="14" SourceSize="14"/></column></pre>
	<column datatype="Double" name="SHIRSU" precision="5" searcha<br="" sourceid="7" sourcename="SHIRSU">Expr= True" OlnUpdate= True" OlnWhere="True" OAfterInsChanged="True" OAfterUpdChanged="True" O SearchDecision="5" Searchanged="True" OfferInsChanged="True" OAfterUpdChanged="True" OF SearchDecision="5" Searchanged="True" OfferInsChanged="True" OAfterUpdChanged="True" OfferInsChanged="True" O</column>
	<pre>dour der bc is on_ // SourceName="SHOOLR" SourceID="8" DataType="AnsiString" Size="22" Searcha <column datatype="AnsiString" name="SHOOLR" searcha<br="" size="22" sourceid="8" sourcename="SHOOLR">Default="True" Expr="True" OlnUpdate="True" OlnUhere="True" OAfterInsChanged="True" OAfterUpdo SourcePresize="22" Severalize="22" Second States" Second States" States Second States" Second States States Second States States Second States Sta</column></pre>
	<pre></pre>
	▼(RowEist) ▼(Row Row ID="0") (Original SUBHCD="101148" SHSHRT="カラブ" SHSHNM="1F64CY RL4CKアイアン(6本)ダイナミックゴ・
	SHTANI="t > F SHIRSU="1" SHCOLR="60LD"/> v.Row ID="1">

TFDMemTableのJSON出力サンプル 🗵 5-3

× + ∂ FDMem2json × + - → C O 0 77-11/ν | C/Projects/Tect

("FDBS": ("Version": 15. "Manager": ("UpdatesRegistry": true. "TableList":	
[{"class":"Table", "Name":"FDOuerv2", "SourceName":"CO400TEMP/SHOHMF", "SourceID":1, "TablD":0, "]	Enfor
ckNotNull":false."ColumnList":	
[{"class":"Column", "Name": "SHSHCD", "SourceName": "SHSHCD", "SourceID":1, "DataType": "AnsiString	". "Si
Only":true "Default":true "Expr":true "OlpUpdate":true "OlpWhere":true "OlpKey":true "OAfter	InsCh
Name": "SHSHCD" "SourcePrecision": 10 "SourceSize": 10]	
["class":"Column" "Name":"SHSHRT" "SourceName":"SHSHRT" "SourceID":? "DataType":"AnsiString"	"Siz
nly" true "Default" true "Evor" true "Olnlindate" true "OlnWhere" true "OAfter InsChanged" true	e "04
"SourcePrecision": 14 "SourceSize": 14]	o, on
"class"""(Augustante and a second and a second a	"Ciz
lu strug "Default" trug "Expr" trug "Olphadata" trug "Olphaca" trug "Trug "Olphaca" trug "Trug "	· "0A
"Source Denaite - Crue, LADI - Crue, Ornopulate - Crue, Orninere - Crue, Okrterinsonangeu - Cru	C, UA
Sourcesterorston . 02, Sourcestze . 02, ["algorithms". "Clukelin" "Course 10". 4 "DeteTupe". "And Ctring"	"Cia
Class - Column, Maile - Simiryam, Sour Cervaile - Simiryam, Sour Cerb - 4, Data 1996 - Alistot Ing	- "OA
"Course Desaire" 26 "Course Size" 26	e, ua
Sourcerrectston .20, Sourcestze .20),	"0:-
Class Column, Name ShSize, SourceName ShSize, SourceID 5, DataType Ansistring	. 312
niy true, betault true, Expr. true, Ulnupdate true, Ulnumere true, OAtterinschanged tru	e, ua
SourcePrecision :14, SourceSize :14,	"
[class : Golumn , Name : SHIANI , SourceName : SHIANI , SourceID :6, Datalype : AnsiString	, SIZ
nly":true, "Default":true, "Expr":true, "OInUpdate":true, "OInWhere":true, "OAfterInsChanged":tru	e, OA
"SourcePrecision": 14, "SourceSize": 14},	
["class" Column", Name: "SHIRSU", "SourceName"; SHIRSU", "SourceID"; /, "Datalype" Double", Pri	BCISI
It":true, "Expr":true, "OInUpdate":true, "OInWhere":true, "OAfterInsChanged":true, "OAfterUpdChan	ged :
:"BCD", "SourcePrecision":5},	
{"class":"Column", "Name":"SHCOLR", "SourceName": "SHCOLR", "SourceID":8, "DataType":"AnsiString",	, "Siz
nly":true, "Default":true, "Expr":true, "OlnUpdate":true, "OlnWhere":true, "OAfterInsChanged":tru	e, "OA
"SourcePrecision":22, "SourceSize":22]], "ConstraintList":[], "ViewList":[], "RowList":[{"RowID"	:0, "0
ブ", "SHSHNM":"LEGACY BLACKアイアン(6本)ダイナミックゴールドシャフト", "SHMKNM":"キャロウェイ	". "S
ト", "SHIRSU":1, "SHCOLR": "GOLD"}}, {"RowID":1, "Original": {"SHSHCD": "102079", "SHSHBT": "クラブ",	"SHSH
MP-514Dシャフト", "SHMKNM":"(株)ミガロ, ", "SHSIZE":"", "SHTANI":"本", "SHIRSU":1, "SHCOLR":"RED"}},	{"Row
{ "SHSHCD": "102082", "SHSHBT": "クラブ", "SHSHNM": "2014 オノフ TYPE-S (黒) ドライバー MP-614Dシ	ヤフト
口, ", "SHSIZE": "AA", "SHTANI": "本", "SHIRSU": 1, "SHCOLR": "BLACK"}}, {"RowID": 3, "Original": {"SHSHC	D":"A
BERTHA BETA ドライバー TYPE 2★". "SHMKNM": "キャロウェイ". "SHS1ZE": "". "SHTANI": "本". "SHIRSU":	1, "SH
["SHSHCD": "ABCDEFGH16", "SHSHBT": "クラブ【新】", "SHSHNM": "2015 ONOFF フェアウェイ: WINGS KURO	(ユー
", "SHTANI": ", "SHIRSU":1, "SHCOLR": ""], ["Row1D":5, "Original": ["SHSHCD": "105567", "SHSHBT":	"クラ
ト) ", "SHMKNM": "キャロウェイ", "SHSIZE": "", "SHTANI": "セット", "SHIRSU": 1, "SHCOLR": ""}}, {"RowID"	":6."

これ以外にTFDMemTableで実現可能な機能について は、docwiki(エンバカデロ社の公式オンラインヘルプ)に も多数掲載されているので、そちらも参照されたい。

048 MIGARO Technical Report







油中 侑	Delphi/400	
福井和彦 石山 智也	Delphi/400	
佐田 雄一	Delphi/400	
正顧	Sm	
苦	artPad4i	

従来のTClientDataSetでもプログラムを作成することは 可能だが、今後の新規開発においてはTFDMemTableコ ンポーネントの使用も検討いただきたい。

3. TSpool400に代わるSQLメインのスプール活用術

Delphi/400では、IBMiのスプールファイルを参照するため のTListSpool400・TSpool400コンポーネントが存在する。 しかしこれらはBDEベースの古い技術で作成されており、最 新のDelphi/400 10.2 Tokyoでも動作にはBDEが必要と なっている。(Windows10ではBDEが互換での動作となるた め、アプリケーションを実行するには管理者権限が必要で、か つ64bitアプリケーションには対応していない)

本章ではこのTListSpool400・TSpool400コンポーネント に代わり、FireDAC接続のSQLでスプール情報を参照する方 法について紹介する。Delphi/400 10.2 Tokyoにおいては FireDAC接続も64bitアプリケーションには対応していない が、本章で記載の各SQLはdbExpress接続でも実行が可能 である。64bitアプリケーションで本章のスプール活用を実施 される場合は、dbExpress接続を利用して頂き、本文中の 「TFDQuery」を「TSQLQuery」に読み替えていただきたい。

①スプールのリスト表示

ソース6

従来はTListSpool400コンポーネントを使って指定した OUTQ内にあるスプールの一覧を表示し、その中にある個別 のスプールの内容をTSpool400コンポーネントで表示して いたが、この一連の操作をFireDAC接続のSQLで代替して行

う手順を紹介する。

まず、TListSpool400コンポーネントでは「LibraryName」 「OutQName」プロパティでOUTQのライブラリと名前を指 定しているが、どのライブラリに目的のOUTQが存在するか わからない場合があっただろう。IBM i(V7R1以上)ではシス テムライブラリ「QSYS2」内にOUTQの一覧情報を保持して おり、『SELECT OUTQ, OUTQLIB, FILES, TEXT FROM QSYS2/OUTPUT_QUEUE_INFO』というSQLを実行する ことでその一覧を表示することができる。

(「OUTPUT_QUEUE_INFO」はビューの名前となってお り、代わりに実ファイル名の「OUTQ_DTL」も指定可能であ る。またこのファイルにはDelphi/400が対応していない型の フィールドが存在するため、上記のフィールドのみを指定する のが望ましい。)

取得した4つのフィールドはそれぞれ
 ・OUTQ=対象OUTQの名前
 ・OUTQLIB=対象OUTQが格納されたライブラリ
 ・FILES=現在そのOUTQに格納されているスプールの数
 ・TEXT=テキスト
 を示しているので、目的のOUTQの名前とライブラリを確認
 する。【ソース6】【図6】

る。こちらも先ほどと同様にIBM i(V7R1以上)でシステム ライブラリ「QSYS2」内にスプールファイルの一覧情報を 保持しており、 『SELECT SPOOLED_FILE_NAME, FILE_NUMBER,

次に指定したOUTQから個別スプールの一覧を取得す

JOB_NAME,CREATE_TIMESTAMP FROM

ソース 7

OUTQ内のスプール一覧をSQLで取得する例(W
FDQuery2.Close: FDQuery2.SQL.Text := 'SELECT SPOOLED_FILE_NAME, FILE_NUMBER, JOI 'FROM QSYS2/OUTPUT_QUEUE_ENTRIES '+ 'WHERE OUTQ =: OUTQ AND OUTQLIB =: OUTQLIB 'AND CREATE_TIMESTAMP >= :CRTDATE ': FDQuery2.ParamByName('OUTQ').AsString := 'QPRINT'; // OUTQ: FDQuery2.ParamByName('OUTQLIB').AsString := 'QGPL': // ライ FDQuery2.ParamByName('CRTDATE').AsDateTime := StrToDateTime('202 FDQuery2.FetchOptions.Mode := fmAll; // 一度に全件取得 FDQuery2.Open;

図 7 スプールの一覧表示

	SPOOLED FILE NAME	FILE NUMBER	JOB NAME	CREATE_TIMESTAMP	^
•	QPDSPJOB	2	671593/QTMHHTTP/VALENCE6D	2021/08/21 3:40:28	
	QPDSPJOB	2	671632/QTMHHTTP/IWAIAS	2021/08/21 3:40:26	
	QPDSPJOB	2	671588/QTMHHTTP/VALENCE6T	2021/08/21 3:40:26	
	QPDSPJ0B	2	671628/QTMHHTTP/GITSERVER	2021/08/21 3:40:25	
	QPDSPJ0B	2	671592/QTMHHTTP/VALENCE52	2021/08/21 3:40:28	
	QPDSPJ0B	2	671591/QTMHHTTP/VALENCE52S	2021/08/21 3:40:27	
	QPDSPJ0B	2	671626/QTMHHTTP/VALENCE6S	2021/08/21 3:40:22	
	QPDSPJ0B	2	671589/QTMHHTTP/VALENCE6S	2021/08/21 3:40:27	
	QPDSPJ0B	2	671627/QTMHHTTP/VALENCE6T	2021/08/21 3:40:25	
	QPDSPJ0B	2	671603/QTMHHTTP/IWAIAS	2021/08/21 3:40:25	
					~
			見た目はE TListSpoo	BDE接続の 1400と類似している	

IBM i 内のOUTQ一覧をSQLで取得する例(V7R1以上)

FDQuery1.Close; FDQuery1.SQL.Text := 'SELECT OUTQ, OUTQLIB, FILES, TEXT FROM QSYS2/OUTPUT_QUEUE_INFO'; FDQuery1.FetchOptions.Mode := fmAll; // 一度に全件取得 FDQuery1.Open;

図 6 OUTQの一覧表示

	OUTQの一覧I	取得		≥ × × ▲ = 	ම¶ ජාරා≊ර A - [27 x 132]	
	OUTQ	OUTQLIB	FILES	TEXT	7744日 編集日 表示() 通信() 75/12(4) 90ド 9(8)	
	0P10	QGPL	0			データの表示
	QDKT	QGPL	0	省略時のディスケット出力待ち行列	行の位置指定	
	QPFROUTQ	QGPL	0		OUTPUT_QUEUE_NAME OUTQLIB	4+5+6+7+8+9 PILES TEXT
Þ	QPRINT	QGPL	101128	省略時の印刷装置出力待ち行列	MATQ QGPL OP10 QGPL	0
	QPRINTS	QGPL	30	特殊用紙用の印刷装置出力待ち行列	QDKT QGPL QPFROUTO QGPL	 省略時のディスケット出力待ち行列 0
	QPRINT2	QGPL	0	2部用紙用の印刷装置出力待ち行列	QPRINT QGPL 100 OPRINTS OCPL	5,620 省略時の印刷装置出力待ち行列 30 熱楽用新用の印刷装置出力満ち行列
	RANQ	QGPL	6		QPRINT2 QGPL RMNO OCEL	0 2部用紙用の印刷装置出力待ち行列
	WMSQ	QGPL	0		WISQ QGPL	0
	YAGI001	QGPL	0		QSPRCLOUTQ QRCL	 システム作成出力待ち行列。
	QSPRCLOUTQ	QRCL	0	システム作成出力待ち行列。	QSRVNON QSERVICE QS9SRVAGT QSRVAGT	0
					FINEPRINT QUSRSYS KIJICHI QUSRSYS KOMPDF QUSRSYS KOMPRI26 QUSRSYS KOMPRI56 QUSRSYS	0 印刷装置 XIJICHI の省略時の出力待ち行列 0 印刷装置 XMPSF の省略時の出力待ち行列 0 印刷装置 XMPSF の省略時の出力待ち行列 1 印刷装置 XMPSF の省略時の出力待ち行列 1 印刷装置 XMPSF の省略時の出力待ち行列
				黒画面で同じSQLを実行しても、 同様の結果が得られる	200 0008013 内=終了 F12=取り消し F12=左 「 a Mail 英数 半角 gF 1900-ゼッションが正常に開いたれました	1 中時最低 1:200 01年8月470日23月53日 F20= 右 F21= 分割 F22= 幅 80 83/632

050 MIGARO Technical Report

QSYS2/OUTPUT_QUEUE_ENTRIES WHERE OUTQ = (OUTQ名) AND OUTQLIB = (ライブラリ名) AND CREATE_TIMESTAMP = (作成日時)』 というSQLを実行することで、その一覧を表示することが

というSQLを実行することで、その一覧を表示することができる。【ソース7】【図7】



Delphi/400	Delphi/400	Delphi/400	SmartPad4i	Valence	
焰中 有	福井和彦 石山智也	依田 雄 —	國元 祐二	尾崎 浩司	

(「OUTPUT_QUEUE_ENTRIES」はビューの名前となっ ており、代わりに実ファイル名の「OUTQ_INFO」も指定可 能である。またすべてのスプールの情報を取得しようとする と処理時間および処理結果の件数が膨大になるため、取得 フィールドを絞るとともにWHERE句を使用することで処理

時間の短縮を図っている。このほかにも、WHERE句に 「JOB_NAME LIKE '%~~~%'」を付加してスプールを発 行したユーザー名で絞り込んだり、SQL文の末尾に 「FETCH FIRST ~~ ROWS ONLY」を付加して取得件数 を絞り込んだりするのも有効である。)

分割する必要があるため、【ソース8】のロジックでうまくい

かない場合はご利用のIBM i環境の設定値をご確認いただ

また、出力先となる一時ファイルを事前に作成しておくのが

ポイントで、このファイルには対象スプールの桁数以上の長

さを持つ〇タイプフィールド1つのみを定義しておく。作成方

法はDDSの作成以外にも、【図8】のようなコマンド発行や、 CREATE TABLEのSQL発行でも問題ない。この一時ファイ

ルは本稿では便宜上通常のライブラリに作成しているが、

QTEMPに作成しても問題はない。

きたい。

②個別スプールの取得

前項で取得したスプールの情報をパラメータとしてIBM iに 対してCPYSPLFコマンドを発行することで、個別スプール の内容を物理ファイルに出力することができる。BDE接続を 使用していた従来のTSpool400コンポーネントでも、内部 的には同様のコマンドを発行してQTEMPに出力される一 時ファイルを通して処理が行われていた。

CPYSPLFコマンドにはいくつかのパラメータが必要だが、 それぞれ前項で取得したスプールファイル名、ジョブ名、タイ ムスタンプ等の各フィールド値を使用する。具体的には 【ソース8】のように記述する。タイムスタンプは日付と時刻に



r sFIL: String; // 出力先ファイル名 sCMD: String; // コマンド文の保管 	
// 出力先ファイルの前回出力値をクリア(ファイルは既に存在する前提) sFIL := 'YSADALIB/SPL2102'; 	【図8】で作成した一時ファイル
// タイムスタンプの作成(書式例『'2020/02/03' '17:34:10'』)	コマンドにはこの書式で
<pre>sDAT := Trim(FDQuery2.FieldByName('CREATE_TIMESTAMP').AsString); sDAT := QuotedStr(Copy(sDAT, 1, 10)) + ' + QuotedStr(Copy(sDAT, 12, 8));</pre>	
sDAT := Trim (FDQuery2. FieldByName ('CREATE_TIMESTAMP'). AsString); sDAT := QuotedStr(Copy(sDAT, 1, 10)) + ' + QuotedStr(Copy(sDAT, 12, 8)); // コマンド文の作成 sCMD := 'CPYSPLF FILE(' + Trim (FDQuery2. FieldByName ('SPOOLED_FILE_NAME')) TOFILE(' + sFIL + ') SPLNBR(' + Trim (FDQuery2. FieldByName ('FILE_NUMBER'). AsString ') CRIDATE(' + sDAT + ') TOMBR (#FIRST)' + // 最初のメンバーに出力 ' CTLCHAR (*FCFC)'; // 制御文字 (改行・改ページ設定時に使用) // TAS400コンポーネントで作成したコマンドを発行し、一時ファイルに書き出し AS400. RemoteCnd (sCMD):	AsString) + // スプールファイル名 // コピー先 ing) + // スプールファイル番号) + // ジョブ番号/ユーザー/ジョブ名 // 作成日+作成時刻
sDAT := Trim (FDQuery2. FieldByName ('CREATE_TIMESTAMP'). AsString); sDAT := QuotedStr(Copy(sDAT, 1, 10)) + ' ' + QuotedStr(Copy(sDAT, 12, 8)); // コマンド文の作成 SCMD := 'CPYSPLF FILE(' + Trim(FDQuery2. FieldByName('SPOOLED_FILE_NAME') ') TOFILE(' + sFIL + ') SPLNBR(' + Trim(FDQuery2. FieldByName('JOB_NAME'). AsString ') CRTDATE(' + sDAT + ') JOB(' + rim(FDQuery2. FieldByName('JOB_NAME'). AsString ') CRTDATE(' + sDAT + ') TOMBR(*FIRST)' + // 最初のメンバーに出力 '' CTLCHAR(*FCFC)'; // 制御文字(改行・改ページ設定時に使用) // TAS400コンポーネントで作成したコマンドを発行し、一時ファイルに書き出し AS400. RemoteCmd(sCMD): d;	AsString) + // スプールファイル名 // コピー先 ing) + // スプールファイル番号) + // ジョブ番号/ユーザー/ジョブ名 // 作成日+作成時刻



③ 取得したスプールの操作

前項のCPYSPLFコマンドで取得した個別スプールを参照	
するには、通常のファイルを参照する際と同様に	
「FDQueryで一時ファイルをSELECTする。すると、 【図9】	
のように各レコードの1桁目に内部識別用の文字(ANSI	
月紙制御コード)を保持していることを確認できる。ⅠBM i	
のスプールファイルをWRKSPLFコマンドで参照した場合	
や従来のTSpool400コンポーネントで参照した場合は、	
この制御コードによって空行や改ページの挿入を内部的	
こ判断している。	

2) RCDLEN(133) IGCDTA(*YES)
桁数はスプールの幅+1 以上を設定する
) 79ション(A) 94ンドウ(W) ヘルブ(H) IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
ー ド 設 計 書 日付 21/08/27 時刻 15:56:39 様式名 SPL2102 レコード長 133
キー順 開始 終了 テキスト記述/欄見出し

- 主な制御コードはそれぞれ以下の意味を持っている。
- ・「」(スペース):何もしない。
- ・「0」:上に1行あけて出力する。
- ・「-」:上に2行あけて出力する。
- ・「+」:改行せずに前の行の内容を上書きする。(ただしス ペース文字の場合は上書きしない)
- 「1|:改ページを行う。
- これらを実装するためには【ソース9】のように記述する。

Delphi/400 俎中 侑 Delphi/400 福井和彦 石山 깸 Ð Delphi/400 佐田 雄 --SmartPad4i 國元祐二 Valence 尾崎 浩司



ソース9

<pre>int file = 'YSADALIB/SPL2102': // ー時ファイル名 ANS: String: // ANSI用紙制御コードの保管用 TXT: String: // ANSI用紙制御コードの保管用 TXT: String: // 指合先 (上側) の行の文字列 AT: AnsiString: // 指合先 (上側) の行の文字列 AT: AnsiString: // 指合先 (上側) の行の文字列 in // 内容をメモに表示 DOLery3.Open: Memol.Lines.Clear: // DOLery3.Fields[0].AString: // 整形しないでそのまま出力する場合 if not(chKSEIKEL.Ghecked) then begin Memol.Lines.Add(sTXT): end // 準形する場合 else begin // 1桁目=ANSI用紙制御コード SMKS := Copy(STXT.2, Length(STXT)): // 0 = Lt:1行あけて出力する if (SANS = '0') then begin // - = Lt:2行あけて出力する else if (SANS = '-') then begin // 1桁目=「-」の場合の処理 // 1桁目=「-」の場合の処理 // 1桁目=「-」の場合の処理 // 1桁目=「-」の場合の処理 // 1桁目=「-」の場合の処理 // 1 の //</pre>		
AMS: String: // AMSI用紙制御コードを除いた文字列 : Integer: // For文用 BEF: AnsiString: // 結合元 (下側) の行の文字列 AFT: AnsiString: // 結合元 (下側) の行の文字列 AFT: AnsiString: // 結合元 (下側) の行の文字列 AFT: AnsiString: // 結合元 (下側) の行の文字列 in // 内容をメモに表示 DOuery3. Close: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. SoL Text := 'SELECT * FROM ' + oFILE: DOuery3. Flow SoL Text := 'SELECT * FROM ' + oFILE: DOUEry3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEry3. Close: DOUEry3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEry3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEry3. Close: DOUEry3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. SoL Text := 'SELECT * FROM ' + oFILE: DOUEY3. Fold * SELECT * FROM ' + oFILE: DOUEY3. Fold *		
: Integer: // For文用 BEF: AnsiString: // 結合先(上側)の行の文字列 AFT: AnsiString: // 結合先(上側)の行の文字列 AFT: AnsiString: // 結合元(下側)の行の文字列 AFT: AnsiString: // 結合元(下側)の行の文字列 in // 内容をメモに表示 DDuery3.Close: DDuery3.Close: DDuery3.Fet:='SELECT * FROM ' + oFILE: DDuery3.Fet:='SELECT * FROM ' + oFILE: DDuery3.Fet:='SELECT * FROM ' + oFILE: DDuery3.Gen: Memol.Lines.Clear: // ENDUERY3.Fet:='SELECT * FROM ' + oFILE: DDuery3.Open: Memol.Lines.Clear: // ENDUERY3.Fields[0].ASString: // 整形する場合 else begin // TFGDuery3.Fields[0].ASString: 1, 1): // WENT Memol.Lines.Add(sTXT): end // United Lines.Add(sTXT): end // 0 = L(c1行あけて出力する if (sANS = '0') then begin Memol.Lines.Add(sTXT): end // - = L(c2行あけて出力する else if (sANS = '-') then begin		
<pre>// 内容をメモに表示 DDuery3.Glose: DDuery3.Glose: DDuery3.Grat:='SELECT * FROM ' + cFILE: DDuery3.FacthOptions.Mode := fmAl1: // -度に全件取得 TY FDDuery3.Open: Memo1.Lines.Clear: vhile not FDOuery3.Eof do begin sIXT := FDOuery3.Fields[0].AsString: // 整形しないでそのまま出力する場合 if not(chkSEIKE1.Checked) then begin Memo1.Lines.Add(sIXT): end // 推目=ANSI用紙制御コード sANS := Copy(FDOuery3.Fields[0].AsString, 1, 1): // ZfHEL以降=制御コードを除いた文字列 sIXT := Copy(sIXT, 2, Length(sIXT)): // 0 = Ltcl行あけて出力する if (sANS = '0') then begin Memo1.Lines.Add(sIXT): end // - =Ltc2行あけて出力する else if (sANS = '-') then begin</pre>		
ry FDOUery3. Open: Memol.Lines. Clear: vhile not FDQuery3. Eof do begin STXT := FDQuery3. Fields[0]. AsString: // 整形しないでそのまま出力する場合 if not (chKSEIKEI. Checked) then begin Memol.Lines. Add (sTXT): end // 整形する場合 else begin // 打相目=ANSI用紙制御コード sANS := Copy (FDQuery3. Fields[0]. AsString, 1, 1): // 2析目以降=制御コードを除いた文字列 sTXT := Copy (STXT, 2, Length (sTXT)): // 0 = L:L1行あけて出力する if (sANS = '0') then begin Memol.Lines. Add (sTXT): end // = =L:2行あけて出力する else if (sANS = '-') then begin		
while not FDQuery3.Eof do begin STXT := FDQuery3.Fields[0].AsString: // 整形しないでそのまま出力する場合 if not(chkSEIKE1.Checked) then begin Memo1.Lines.Add(sTXT): end // 整形する場合 else begin // 1桁目=ANSI用紙制御コード sANS := Copy(FDQuery3.Fields[0].AsString, 1, 1): // 2桁目と2桁目以降を分割する sTXT := Copy(sTXT, 2, Length(sTXT)): // 0 = Ltc1行あけて出力する if (sANS = '0') then begin Memo1.Lines.Add(sTXT): end // - = Ltc2行あけて出力する else if (sANS = '-') then begin 1桁目=「-」の場合の処理		
<pre>// 整形しないでそのまま出力する場合 if not(chkSEIKEI.Checked) then begin // 整形する場合 else begin // 1桁目=ANSI用紙制御コード sANS := Copy(FDOuery3.Fields[0].AsString, 1, 1): // 2桁目以降=制御コードを除いた文字列 I析目と2桁目以降を分割する STX := Copy(sTX, 2, Length(sTXT)): // 0 = Ltcl行あけて出力する if (sANS = '0') then begin Memol.Lines.Add(sTXT): end // - = Ltc2行あけて出力する else if (sANS = '-') then begin </pre>		
<pre>// 整形する場合 else begin // 1桁目=ANSI用紙制御コード sANS := Copy(FD0uery3.Fields[0].AsString, 1, 1): // 2桁目以降=制御コードを除いた文字列 sIXT := Copy(sIXT, 2, Length(sIXT)): // 0 = L(:1行あけて出力する if (sANS = '0') then begin Memol.Lines.Add('): Memol.Lines.Add(sIXT): end // - = L(:2行あけて出力する else if (sANS = '-') then begin </pre>		
sANS := Copy(FDQuery3.Fields[0].Astring, 1, 1): // 2桁目以降=制御コードを除いた文字列 sTXT := Copy(sTXT, 2, Length(sTXT)): // 0 = 上に1行あけて出力する if (sANS = '0') then begin Memol.Lines.Add('): Memol.Lines.Add(sTXT): end // - =上に2行あけて出力する else if (sANS = '-') then begin 1桁目=「-」の場合の処理		
// 0 =上に1行あけて出力する if (sANS = '0') then begin Memol.Lines.Add('): Memol.Lines.Add(sTXT): end // - =上に2行あけて出力する else if (sANS = '-') then begin		
begin Memol.Lines.Add('): end // - =上に2行あけて出力する else if (sANS = '-') then begin		
// - =上に2行あけて出力する else if (sANS = '-') then begin		
if (sANS = '-') then begin		
Memo1.Lines.Add(''); Memo1.Lines.Add('); Memo1.Lines.Add(sTXT); end		
// + =改行せずに前の行の内容を上書きする 1桁目=「+」の場合の処理		
if (sANS = '+') then begin // 結合先(上側)の行の文字列 sBEF := AnsiString(Memol.Lines[Memol.Lines.Count - 1]); sBEF := AndSiSUs(BEF): // シフト文字代替スペースの付加[ソース10] // 結合元(下側)の行の文字列 sAFT := AnsiString(sTXT): sAFT := AndSiSUs(AFT): // シフト文字代替スペースの付加[ソース10] // 結合先(上側)が短い場合は結合元(下側)と長さを合わせる if (Length(sBEF) < Length(sAFT)) then begin		
sBEF := sBEF + StringOfChar(' ', Length(sAFT) - Length(sBEF)); end; // バイト単位で文字を上書きする for i := 1 to (length(sAFT)) do		
begin if (sAFT[[] <> '') then begin		
<pre>sBEF[i] := sAFT[i]; end: end:</pre>	_	
// 結合後の文字列をメモの最下行に再セット sBEF := RmvS1S0(sBEF); // シフト文字代替スペースの除去【ソース11】 Memol.Lines[Memol.Lines.Count - 1] := sEEF; end		
// 1 =改ページを行う else		
if (sANS = '1') then 1桁目=「1」の場合の処理		
// 改ページ時の処理は要件によって異なるため割愛 Memo1. Lines. Add (★★ここで改ページ★★'): Memo1. Lines. Add (sTXT); end		
// スペース=何もしない1桁目=それ以外の場合の処理		
else begin Memol Lines Add(sTYT):		
meanwr, Llifes, Auw (SiAl), end; end;		
FDQuery3.Next: // 一時ファイルの次の行へ		
ena.		
FDQuery3. Close; nd;		



「+」の制御時のみDelphi側で実装する手順が少々複雑で、 IBM i側のシフト文字を考慮して文字列の結合を行う必要 がある。具体的には、結合元と結合先の文字列の中の全角 文字と半角文字の境界に疑似的にシフト文字の代替となる スペースを付加【ソース10】し、結合完了後には逆に全角文 字と半角文字の境界にセットされた疑似的なスペースを除 去する【ソース11】。また、【ソース9】のロジックで結合を行っ ている箇所はバイト単位で結合する必要があるため、 AnsiStringにキャストしている。この結果、2バイト(全角)文 字の1バイト目または2バイト目だけが結合対象となった場 合は文字化けを起こすことがあるが、IBM i側でWRKSPLF コマンド等を使ってスプールの内容を参照しても同様の結 果になる。

ソース 10

シフト文字代替スペースの付加を行う関数の例
<pre>{++++++++++++++++++++++++++++++++++++</pre>
function AddSISO(AStr: AnsiString): String; var
1: Integer: S: AnsiString: begin
// (初期値) 1バイト目が全角の場合 if (System.AnsiStrings.ByteType(AStr. 1) = mbLeadByte) then S := ' '
// (初期値) 1パイト目が半角の場合 else S := '';
// 文字列を確認し、全角と半角の切替ボイントにダミーの半角スペースをセット for i := 1 to Length(AStr) do begin S := S + AStr[i]:
<pre>if (System.AnsiStrings.ByteType(AStr, i) = mbSingleByte) and (System.AnsiStrings.ByteType(AStr, i + 1) = mbLeadByte) then begin S := S + ' ';</pre>
end:
if (System. AnsiStrings. ByteType(AStr, i) = mbTrai/Byte) and (System. AnsiStrings. ByteType(AStr, i + 1) = mbSingleByte) then begin
s := S + ' ' ; end:
Result := String(S); // 結果を返却 end:

ソース 11

	シフト文字代替スペースの除去を行う関数の
	[************************************
	var i: Integer; bS0: Realess;
	begin Result := '':
	bSO := False; if AStr ◇ '' then begin
	// 文字単位で後ろからカウント for i := Length (AStr) downto 2 do
	begin // 1つ後の文字でフラグセットされた半角スペース(SO)のとき、セットしない if (bSO) then bSO := False
	else // 半角スペース(SI)かつ1つ前の文字が全角のとき、セットしない if (AStr[i] = ' ') and (Length(AnsiString(AStr[i - 1])) = 2) then begin end // 何もしない
	erse begin // 文字をセット Result := AStr[i] + Result:
	// 全角かつつ方向文字が半角スペース(SO)のとき、フラグセット if (Length(AnsiString(AStr[i])) = 2) and (AStr[i - 1] = ' ') then bSO := True; end: end:
	// 全角始まりでない場合は1文字目(SIでない)を最後に足す if (not bSD) then
	Result := AStr[1] + Result; end; end:
I	enu,

なお、制御コードを全く使用せずにただスプールの内容を 順次参照するだけでよい場合は、【ソース8】のCPYSPLFコ マンド発行時に『CTLCHAR(*FCFC)』パラメータを除外 すれば、ANSI用紙制御コードが無い状態の文字列を取得 できる。状況に合わせて使い分けていただきたい。

4.まとめ

Delphi/400では過去バージョンでコーディングされたプ ログラムや古くから存在するIBM iの資産を活用できるこ とが大きな利点ではあるが、Windowsや周囲の環境の技 術革新に伴ってDelphi/400も進歩を続けており、特にア プリケーションを新規開発する場合は最新のFireDAC接 続を活用した方が得策であると言える。 本稿の記述を参考に、最新バージョンのDelphiならびに Delphi/400に関心をもっていただければ幸いである。



