

# RPGを知らなくてもここまでできる —ユーザーサイドからのアプリケーション開発

石井 裕昭 様

豊鋼材工業株式会社  
製造総括部 部長代行



豊鋼材工業株式会社  
<http://www.yutaka-steel.co.jp/>

「鋼材のことならあらゆるニーズに応える企業」をモットーに、広島から沖縄までをカバーし、鋼材およびその他金属の加工、販売を行っている。伊藤忠丸紅鉄鋼・新日本製鐵系の会社である。

## ソリューション導入の経緯と概要

豊鋼材工業株式会社の主体である溶断事業では、従来、生産に関するさまざまな情報が、紙の帳票による事後の実績入力や、あるいは担当者の経験等に基づく繁閑の判断と工程管理といったことに依存していた。したがって営業部門、経営者、さらに製造部門内からも、工場の操業状況や受注製品の進捗・計画状況はブラックボックス化して、お客様対応のレスポンスやさまざまな判断に支障をきたしていた。

このような状況を打開するため、平成17年より工場内の無線LAN強化をはじめ、各設備へのWindows CE端末、無線ハンディターミナル、およびラベルプリンタの配備と帳票へのバーコード出力等の基盤整備を開始した。【図 1-1】

Delphi/400は、この基盤整備でリアルタイムに収集される情報を定量的かつ直感的に利用する手段として導入された。本アプリケーションは、生産・出荷

計画、進捗、履歴、山積み等の情報をさまざまな角度から提供する参照系と、生産指示や計画作成等の更新系の処理を備えている。

## 開発の独創性・創意工夫

RPGを使用できるシステム室要員数が少なく、本アプリケーションで実現したい数多くの機能について、データ抽出の条件・項目等の仕様が事前に確定しにくいこともあり、開発はユーザーに近い立場の者が主担当で行った。

そのため、データ抽出はBDEのQueryで、SQL文を動的に生成して抽出する構成とした。だが、初めてのDelphi/400での開発ということもあって、当初は実用に堪えないものであった。

いかに一度に表示できる情報量が多くても、表示の速度が実用レベルになれば、また操作性等で5250での参照に対して大きな優位性がなければ、ユーザーには使われない。

そこで、次の対応を順次実行し、実用

レベルに到達させた。

- ① Query → DatasetProvider → Client Dataset の組み合わせで、抽出したデータに対して、ローカルでのフィルタリング、並べ替えのボタン、DBGridをマウス操作のみで実現するという機能を実装
- ② SQLのパフォーマンスについては、i-SeriesナビゲータのVisual Explainにより、推奨索引確認および索引自動生成での改善
- ③ 同じくVisual Explainにより、結合キーの型の妥当性確認と調整
- ④ SQL実行時のSQL文抽出機能とSQL動作検証用アプリにより、デバッグとチューニング
- ⑤ 関連データの一括抽出とマスターデータ参照による連携機能で、同時参照(マスター⇄詳細⇄最詳細)【図 2-1】
- ⑥ SQL実行待機の体感時間の軽減(ProgressBar、AVI動画再生等)
- ⑦ QueryのOnCalcFieldsイベントでの処理により、SQLの速度低下の抑制

図1-1 生産管理システム再構築イメージ

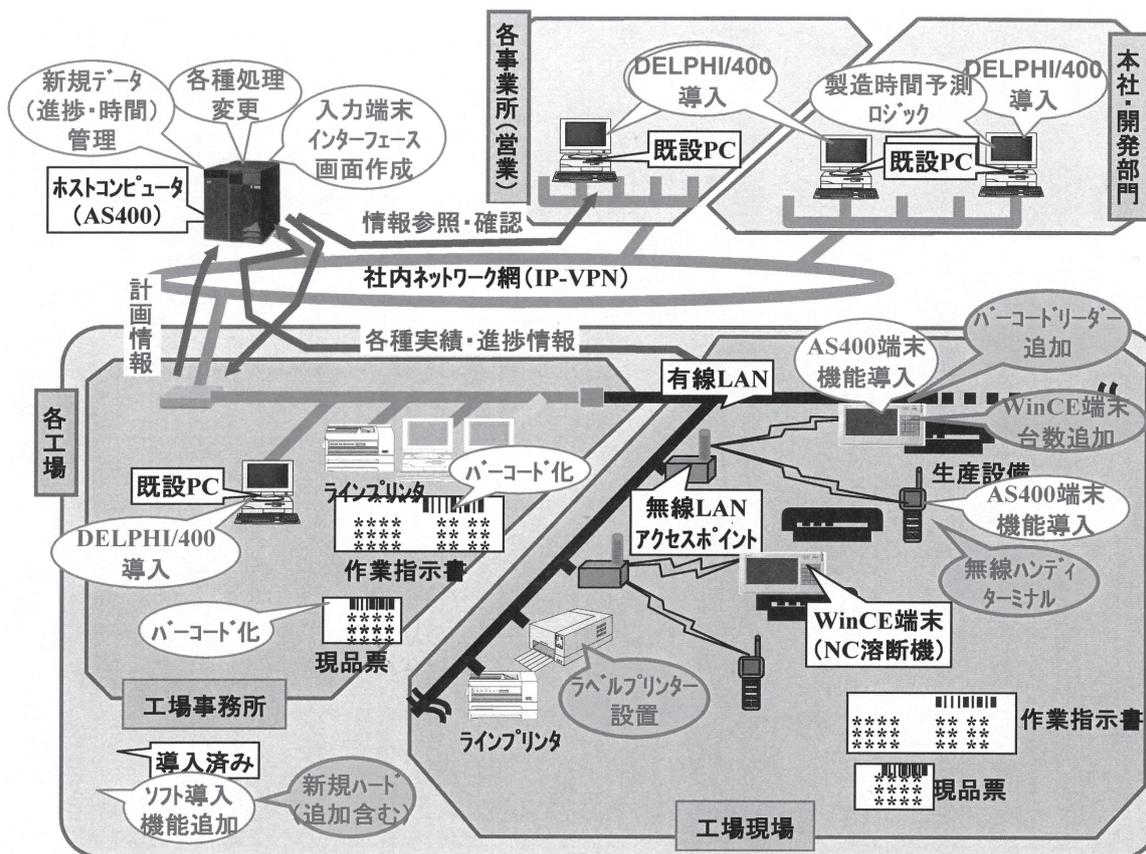


図2-1 マスタ⇔詳細⇔再詳細のデータ連携例

工場	分類	機種	納期	数	現品票数	重量	単重
330	溶断	ET	8080	64	34	1972	30.8
330	溶断	ET	8090	44	22	1360	30.9
330	溶断	ET	8090	46	1	1400	30.4
330	溶断	ET	8080	121	42	6527	53.9
330	溶断	ET	8080	51	10	3417	67
330	溶断	ET	8080				38.1
330	溶断	ET	8081				3
330	溶断	ET	8081	52	45	365	7
330	溶断	ET	8081	65	5	285	4.3

工場	分類	機種	納期	客先	持込先	摘要	厚	規格	ロットNo	数
330	溶断	ET	8080	栄工業		08U06-1003	4.5	WH400	631695	
330	溶断	ET	8080	栄工業		08U06-1003	6	WH400	631696	
330	溶断	ET	8080	栄工業	2機轉	08-81300	8	SS400	631714	
330	溶断	ET	8080	栄工業	2機轉	08-81300	8	SS400	631713	
330	溶断	ET	8080	栄工業	2機轉	08-81300	8	SS400	631775	1
330	溶断	ET	8080	栄工業	2機轉	08-81300	8	SS400	631776	1
330	溶断	ET	8080	栄工業	2機轉	08-81320	8	SS400	631772	1

工場	分類	機種Gr	機種	次工程	納期	客先	持込先	摘要	ロットNo	部門	担当	受発注	行	品種	加工No	規格	厚	幅	長
330	溶断	LZ	ET		8080	栄工業		08U06-1003	631695	鋼材課	久保田	東E	25190	ウエルハ	P000038950	WH400	4.5	1450	2230
330	溶断	LZ	ET		8080	栄工業		08U06-1003	631695	鋼材課	久保田	東E	25190	ウエルハ	P000038952	WH400	4.5	1171	5050
330	溶断	LZ	ET		8080	栄工業		08U06-1003	631695	鋼材課	久保田	東E	25190	ウエルハ	P000038951	WH400	4.5	1171	2600
330	溶断	LZ	ET		8080	栄工業		08U06-1003	631695	鋼材課	久保田	東E	25190	ウエルハ	P000038953	WH400	4.5	1180	2230

- ⑧動的に内容を変化させる comboBox 用のデータベースの、ローカル化と定期更新
- ⑨一定の未操作時間経過で、自動的に表示情報を更新
- ⑩ Grid に表示される不要項目の非表示化、並び順変更の可能化。その条件は、次回起動時にも保持 (Ini ファイルに配列情報を書き込み)

## 実用レベルに向けた改善ポイント

アプリケーション開発の独創性・創意工夫について詳細を述べる。

①毎回、抽出条件を設定し SQL 処理を行うよりも、ある程度の範囲で抽出し、ローカルでキャッシュされたデータに対してフィルター、並べ替え、検索等を同時に実行可能にすることにした。これにより、思考の中断も少なくできる。そのために、ClientDataset の使用を基本とした。

抽出した結果の並べ替えは、任意の項目の組み合わせで可能であり、優先順が明示できるようにしている。また、フィルター機能についても、複数条件の組み合わせが可能で、条件とフィルター中であることがわかるようにした。【図 2-2】

なお、DBGrid の各列のタイトル、巾等については、開発のメンテナンス性を考慮した。Query の静的項目のみの設定で、その内容が自動的に ClientDataSet に引き継がれるように、OnAfterOpen イベントに汎用処理を記述した。【図 2-3】【図 2-4】

また DBGrid、Query、ClientDataSet、フィルター用ボタン等、関連する一連のコンポーネントを配列化した。とともに、Tag プロパティ等の活用により、呼び出し元の識別を汎用化し、コードの記述を抑えた。【図 2-5】【図 2-6】

②③ IBM の IBM i 講座「SQL パフォーマンス・チューニングの基礎」を受講し、IBM i に同梱されている i-Series ナビゲータを開発用 PC に導入した。【図 2-7】

これにより、組み込む SQL のパフォーマンスについて検証を行うことができ、適正な索引の確認と索引の生成が一連の

操作で可能となった。結合キーの型不一致のチェック等も可能であり、その際には iSeries DB2 の SQL 解説書を参考に型の変換を行った。【図 2-8-1】【図 2-8-2】

i-Series ナビゲータでは、GUI の画面で既存ライブラリに新規にテーブル、ビュー (論理ファイル)、索引等を簡単に作ることも可能であるため、情報システム室に依存せず、新たな管理項目を含むデータベースを作れるため有効である。【図 2-9】【図 2-10】

④開発時のテストにおいて、データが正しく抽出されない、またはエラーが発生する場合、実アプリケーションのコード修正を繰り返すとコンパイルの時間ロスが大きい。また、どのような SQL 文が実行されているかも、コードからは読み取りにくい。そのため、該当 SQL の配列番号指定で実行された SQL 文を、テキストで読み出せる仕組みを作った。

また、任意の SQL 文での抽出結果を確認するテスト専用のアプリケーションを、別途準備し、SQL のテストが簡単にできるようにした。【図 2-11】【図 2-12】【図 2-13】

⑥ 1 回の操作で複数の Query を OPEN する際には、その変わり目で Progress Bar の Position を変更し、SQL 処理が進んでいることをユーザーに示す、ということを実行した。

一方、単体の場合、個々の処理時間が長いと本当に処理が進んでいるか不安になり、体感時間も長くなる。そのため、動画を指定の Panel 上で再生して、ユーザーの手待ち感の軽減を試みた。【図 2-14】【図 2-15】

⑦ SQL 文には文字数の制限もあり、CASE 文等を多用するとレスポンスが低下するケースも多い。また、複雑な処理の記述が困難となるため、内容に応じて Query の OnCalcFields イベントを用いた。

⑧抽出条件の設定では、ユーザーの好みに応じてコードの Edit 入力、またはコードに対応する文字の ComboBox からの選択を行えるようにしている。

ComboBox 使用の場合、選択肢を少

なくし操作性を高めるために、例えば営業担当者の ComboBox の内容は、部門の ComboBox の選択結果に応じて動的に変更される。【図 2-16】【図 2-17】【図 2-18】

その際、IBM i の情報を参照して ComboBox を設定すると、ネットワーク、SQL の待ち時間が多少発生する。そこで、ComboBox で多用される社員マスター、客先マスター等の更新頻度が少ないデータベースは、定期的にユーザーの PC に Paradox 形式でダウンロードされる仕組みを作り、ComboBox 操作時の待ちをなくした。【図 2-19】【図 2-20】【図 2-21】

⑨同一の抽出条件であれば、設定に対応して起動中に一定時間アプリケーション操作がない場合は、ClientDataSet の Refresh メソッドを実行するようにした。

これは、ApplicationEvents の OnIdle、OnMessage イベントと、Timer の OnTimer イベントで実装できる。【図 2-22】【図 2-23】

⑩各機能で抽出される項目は、ある程度汎用性を持たせるために多くなり、ユーザーによって不要だったり、または優先度が異なる。横スクロールの手間、同時参照性により、並べ替え、一部項目の非表示化等のカスタマイズのニーズがあった。【図 2-24】【図 2-25】【図 2-26】

## 教訓と今後の予定

参照系のアプリケーションについては、RPG 等の知識がなくても十分構築できる。ただし、SQL の構文、JOIN キーの型の整合性等により、パフォーマンスが大きく変化する。十分な検証が必要である。

関連する各種情報を同時参照可能とすることで、5250 アプリケーション以上の Total パフォーマンスが得られる。今後は、抽出に時間を要する機能について、パフォーマンスを改善する手段をさらに追求したい。

図2-2 フィルタ、並び替えの状況

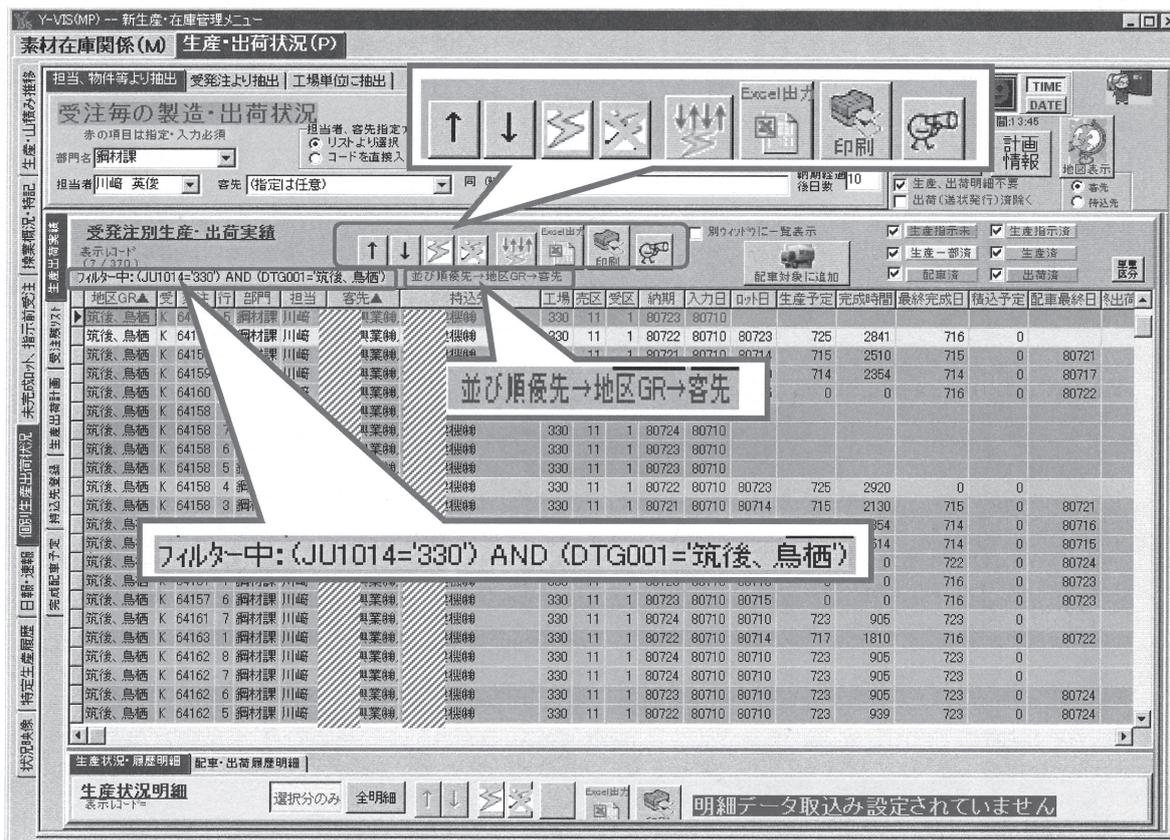
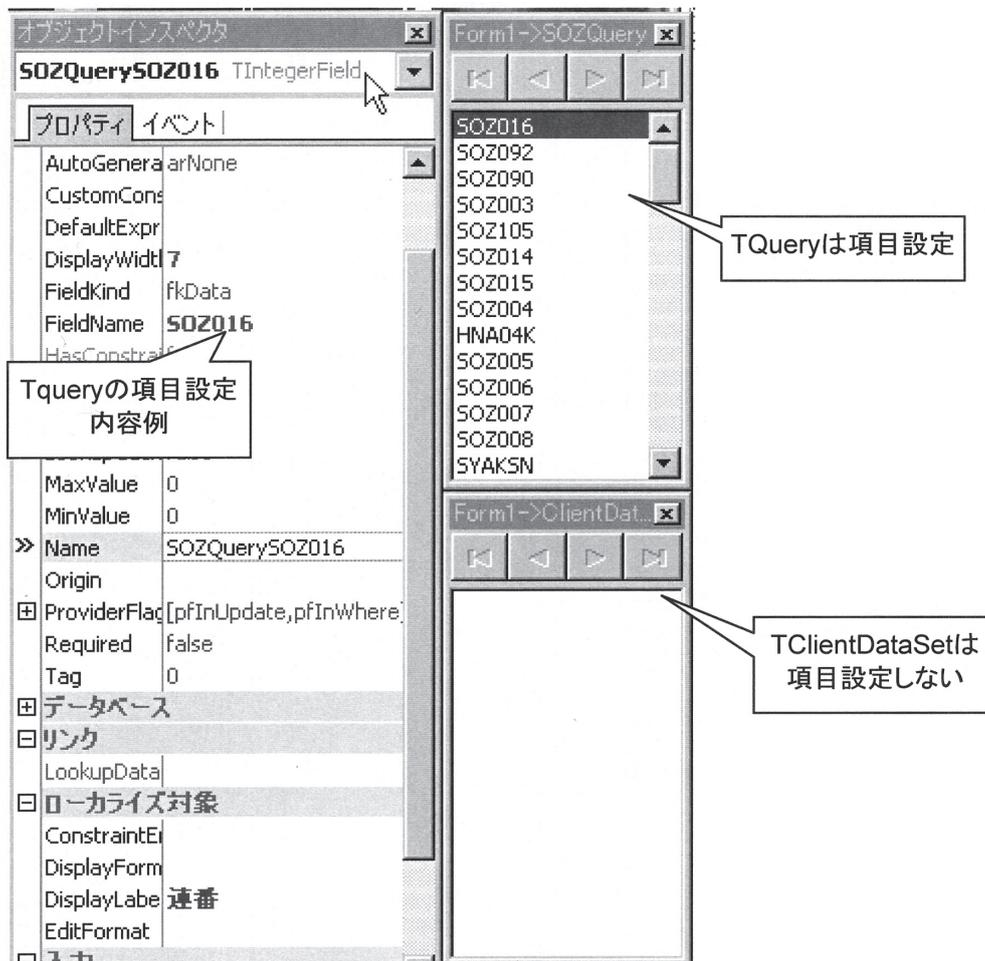


図2-3 Tqueryへの静的項目設定例



## エンドユーザー評価

●従来からの使い慣れの点もあり、単純な照会については、レスポンスの点から5250画面が依然として使用される場合もある。だが、Delphi/400のアプリケーションでは、情報がより定量的に確認でき、かつ一度に多くの関連情報が参照できる。この部分に対する評価が高い。

●多目的なアプリケーションのため、使用者は役員～業務スタッフまでと幅広い。

■

図2-4 Tqueryへの静的項目設定内容をTClientDataSetに反映する処理例

```
n:=TClientDataSet(DataSet).Tag;
for i:=0 to mainQUERY[n].FieldCount-1 do
begin
// Tqueryの列名と同じ列がTClientDataSetに存在するか?
if Assigned(Form1.mainCDS[n].FieldByName(Form1.mainQUERY[n].Fields[i].FieldName)) then
begin
// TClientDataSetの列に対してTQueryのDisplayLabelをコピー
Form1.mainCDS[n].FieldByName(Form1.mainQUERY[n].Fields[i].FieldName).DisplayLabel :=Form1.mainQUERY[n].Fields[i].DisplayLabel;
Form1.mainCDS[n].FieldByName(Form1.mainQUERY[n].Fields[i].FieldName).DisplayWidth :=Form1.mainQUERY[n].Fields[i].DisplayWidth;
if (mainQUERY[n].Fields[i].DataType=ftFloat) Or (mainQUERY[n].Fields[i].DataType=ftInteger) then
(Form1.mainCDS[n].FieldByName(Form1.mainQUERY[n].Fields[i].FieldName)As TNumericField).DisplayFormat
:= (Form1.mainQUERY[n].Fields[i] As TNumericField).DisplayFormat;
Form1.mainCDS[n].FieldByName(Form1.mainQUERY[n].Fields[i].FieldName).Visible :=Form1.mainQUERY[n].Fields[i].Visible;
Form1.mainCDS[n].FieldByName(Form1.mainQUERY[n].Fields[i].FieldName).tag:=Form1.mainQUERY[n].Fields[i].tag;
end;
end;
```

図2-5 機能毎の共通コンポーネントを配列化

```
mainQUERY: array[1..86] of TQuery;
mainCDS: array[1..86] of TClientDataSet;
mainSource:array[1..86] of TDataSource;
mainDBGrd:array[1..86] of TDBGrid;

OrderJunLBL: array[1..86] of TLabel;
EXCELoutBTN:array[1..86] of TBitBtn;
RecNoLBL:array[1..86] of TLabel;
FiltCondLBL:array[1..86] of TLabel;
```

図2-6 汎用処理での呼び出し元特定容易化例

```
for i:=1 to HighMainN do
mainCDS[i].tag := i; //操作の呼び出し元の目印とする

for i:=1 to HighMainN do
mainSource[i].tag := i; //操作の呼び出し元の目印とする

for i:=1 to HighMainN do
mainDBGrd[i].tag := i; //操作の呼び出し元の目印とする

for i:=1 to HighMainN do
if EXCELoutBtn[i]<>Nil then
EXCELoutBtn[i].tag := i; //操作の呼び出し元の目印とする
```

配列番号を  
Tagプロパティに  
割り付け

(Formの  
OnCreate  
イベントで設定)

```
[j:=(Sender as TBitBtn).Tag;
if (mainCDS[j].Active=False) or (mainCDS[j].RecordCount=0) then
begin
showmessage('対象データを表示の上、実行しなさい!!');
exit;
end;
```

汎用化された  
エラー処理例

図2-7 iSeriesナビゲータの管理画面例 (アクティブ・ジョブでDelphi/400使用状況確認)

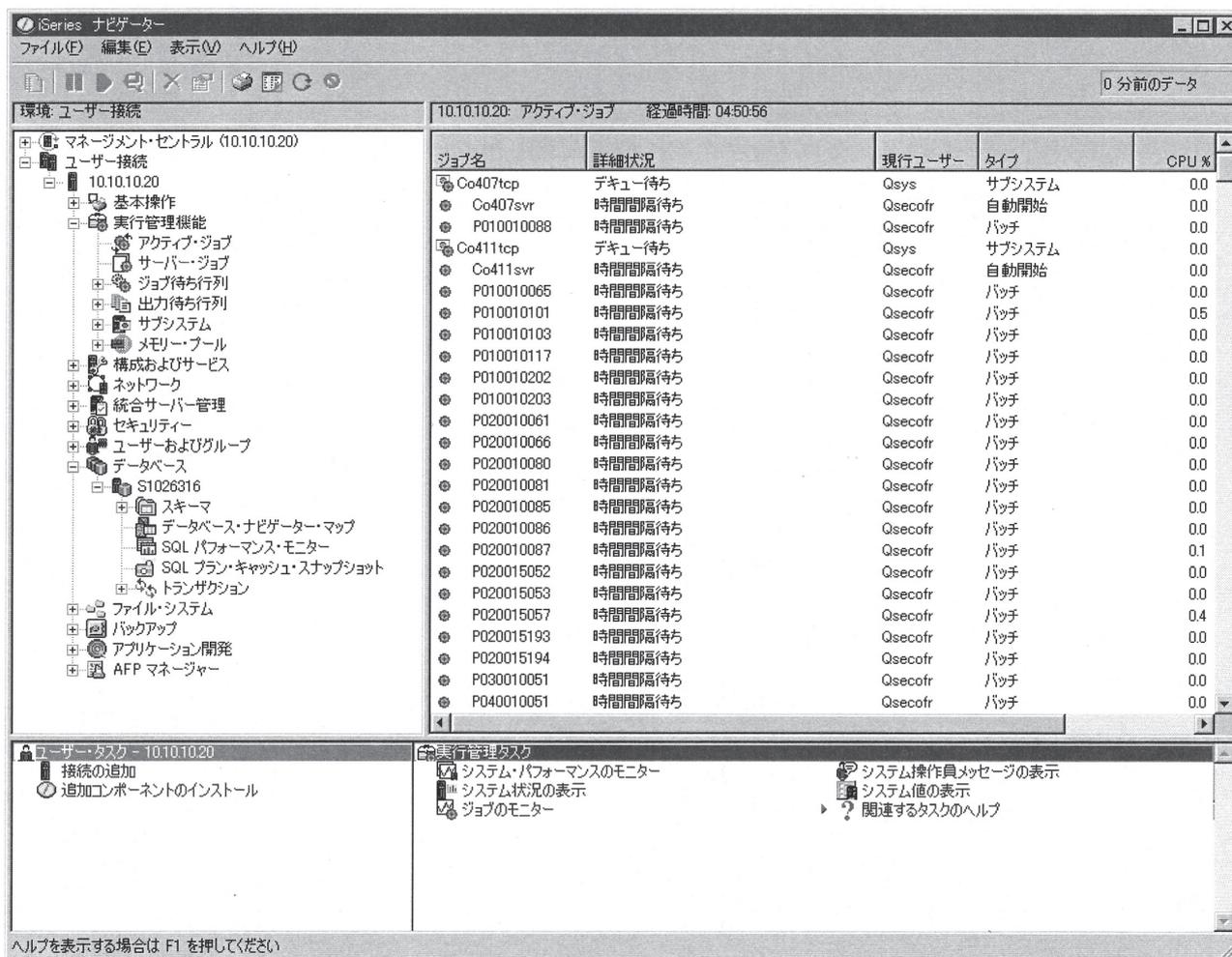


図2-8-1 iSeriesナビゲータでのSQLパフォーマンス検証例

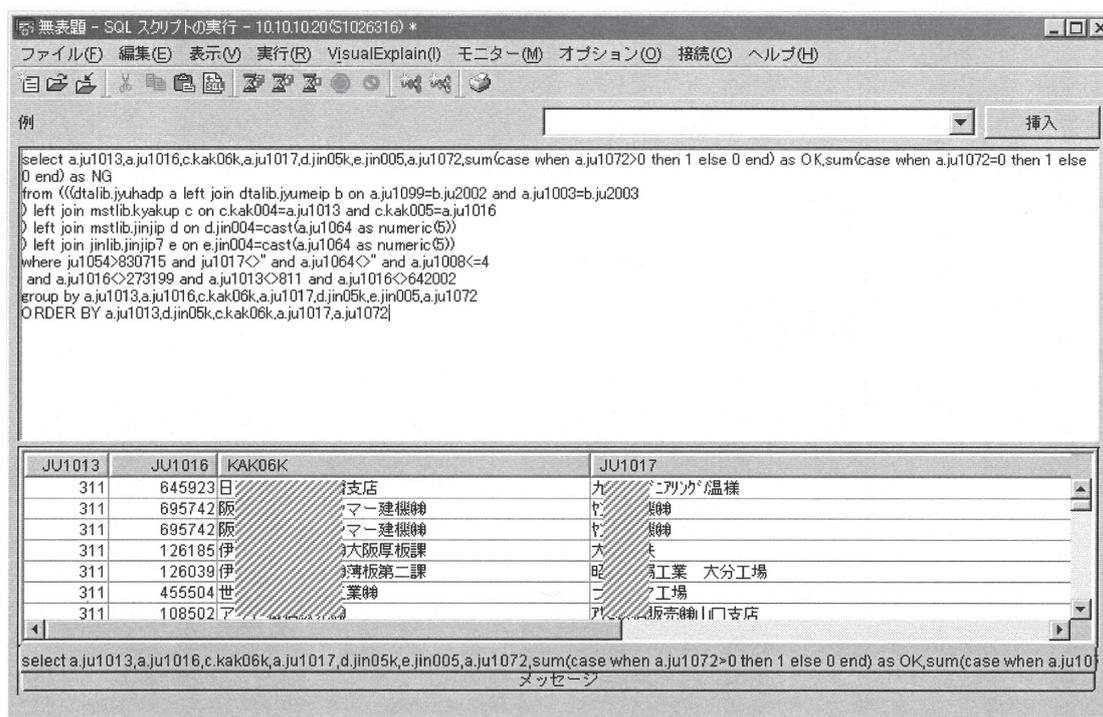


図2-8-2 iSeriesナビゲータ(Visual Explain)でのアクセプラン、索引処理確認例

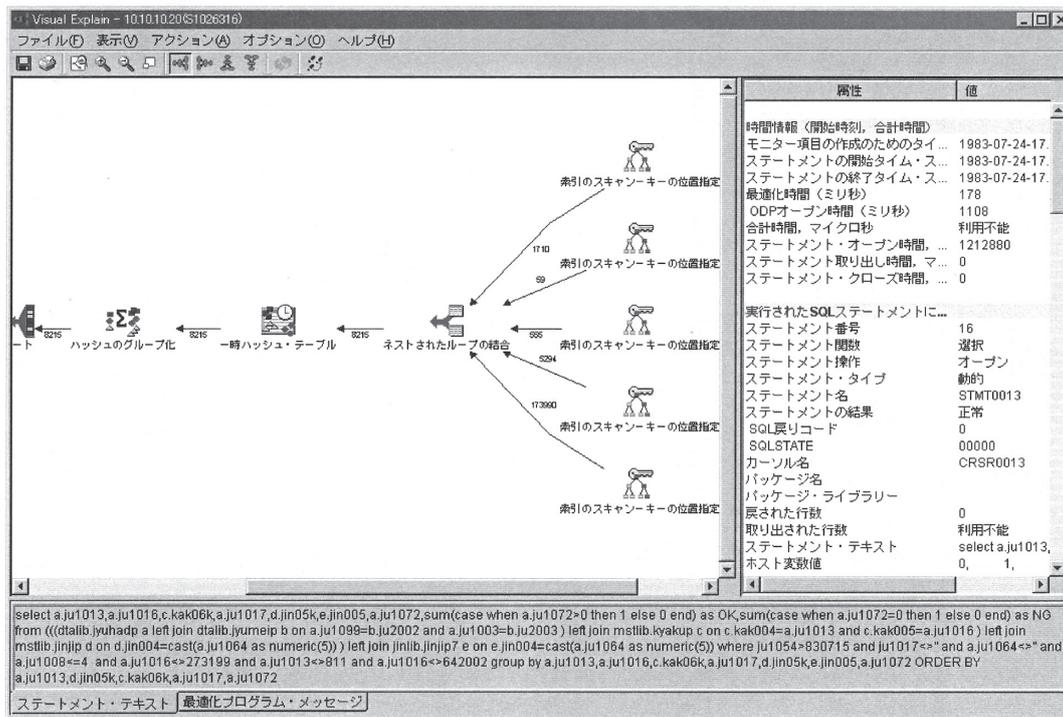


図2-9 iSeriesナビゲータによるGUIでのテーブル作成例

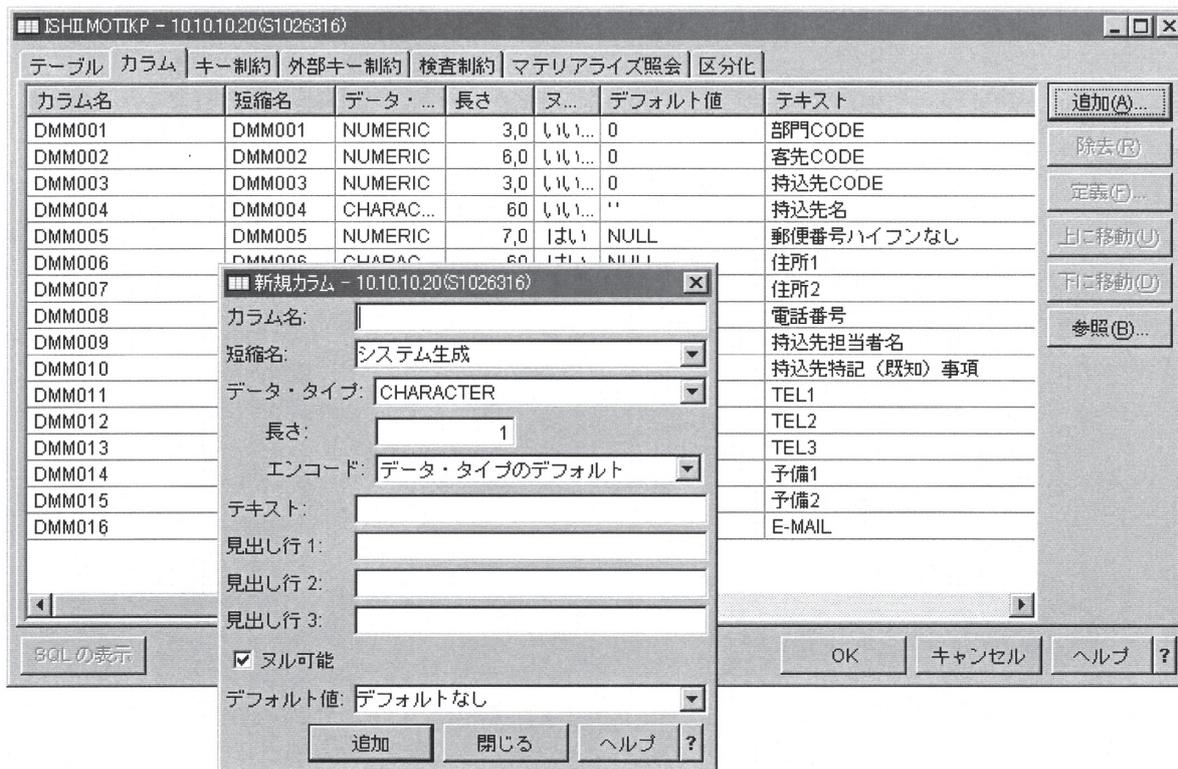


図2-10 iSeriesナビゲータによるGUIでの索引作成例

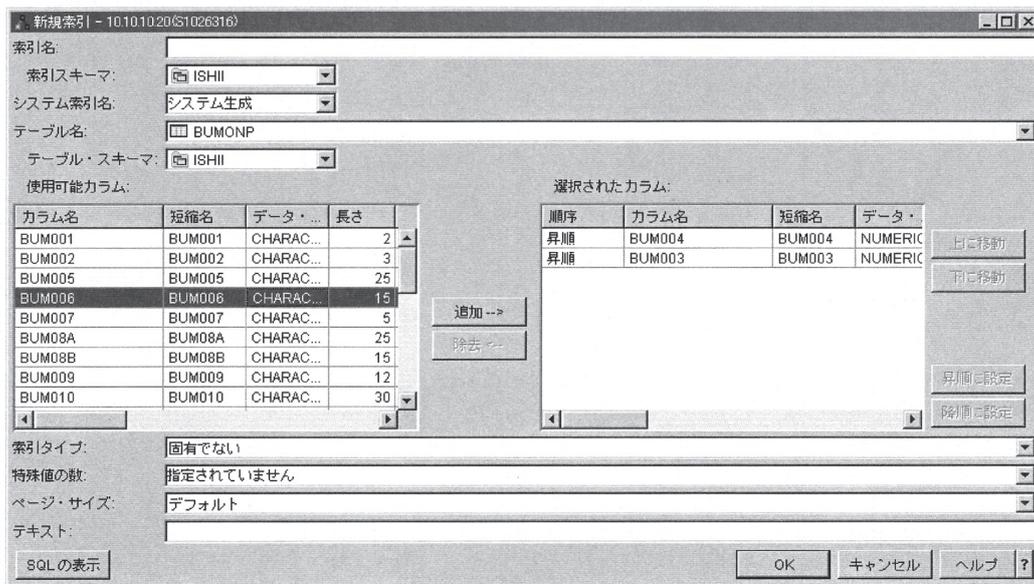


図2-11 Queryの配列番号でSQL文抽出

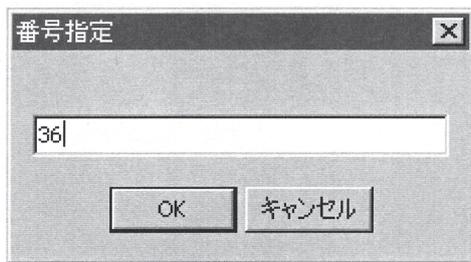


図2-12 抽出されたSQL文

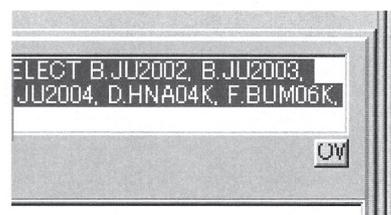


図2-13 単体テスト(SQL検証)用アプリケーション使用例



図2-14 SQL実行中の状況例-1

Y-VIS(MP) - 新生産・在庫管理メニュー

素材在庫関係(M) 生産・出荷状況(P)

外注倉庫は除外  在庫無しは除外  紐付きは除外  ロット明細も取り込み  契約情報も取り込み  ロット明細も取り込み

工場 ALL 品種コード 2250 規格 ALL 幅 ALL 長さ ALL 板厚 ALL 寸法 ALL 自支区分 ALL メーカー ALL

在庫一覧表示 クロス集計(グラフ) | 任意集計 |

表示コード (1/678) 合計重量(kg) 4335982

ProgressBar表示

素材No	元コードNo	工場	分譲元	自支	規格	品種	板厚	幅	長さ	尺寸	WI	L1	数量	重量	橋梁表面	引当	引当	引当	引当
1706137		330	430	1	SM490A	2250	9	2438	12192	8 x 40	0	0	2	4326	0	0	0	0	0
1734505		330	0	1	SM490A	2250	10	2100	6096	x 20	0	0	5	5175	0	0	0	0	0
1676206		330	0	1	SM490A	2250	16	2438	12192	8 x 40	0	0	1	3845	0	0	0	0	0
1686344		330	0	1	SM490A	2250	22	2438	12192	8 x 40	0	0	1	3845	0	0	0	0	0
1678450		330	0	1	SM490A	2250	25	2438	12192	8 x 40	0	0	1	3845	0	0	0	0	0
1678449		330	0	1	SM490A	2250	28	2438	12192	8 x 40	0	0	1	6728	0	0	0	0	0
1678448		330	0	1	SM490A	2250	32	2438	12192	8 x 40	0	0	1	7690	0	0	0	0	0
1771163		330	430	1	SM490B	2250	9	2310	6230		0	0	1	1048	0	0	0	0	0
1771164		330	430	1	SM490B	2250	9	2310	6430		0	0	1	1080	0	0	0	0	0
1771165		330	430	1	SM490B	2250	9	2320	9130		0	0	1	1541	0	0	0	0	0
1770868		330	430	1	SM490B	2250	12	1810	6360		0	0	1	1117	0	0	0	0	0

AVIファイル(動画)再生

詳細(ファーンは二コから) 受発注詳細 | ロット内容 |

板番の明細 選択のみ | 全明細 |

抽出済み(在庫なし) 12 即日更新の在庫在庫 6ヶ月以上の中期在庫

図2-15 SQL実行中の状況例-2

Y-VIS(MP) - 新生産・在庫管理メニュー

素材在庫関係(M) 生産・出荷状況(P)

生産・山積み推移

90%

今月の生産推移 長期生産推移 山積み表照会

納期範囲: 14 日前 ~ 14 日先まで

長期推移 長期山修正・登録 | 生産目標入力 | 稼働(FT)率・休止 | 歩留管理 | 納期達成率

Gauge表示

重量(日別)(トン) 数量(日別)(枚) 単重(日別) 重量(累計)(トン) 数量(累計)(枚) 単重(累計)(kg/s)

表示単位区分 工場無視(合算) 機械単位 品種単位 工場単位

AVIファイル(動画)再生

図2-16 ComboBoxのItem内容動的変更例-1

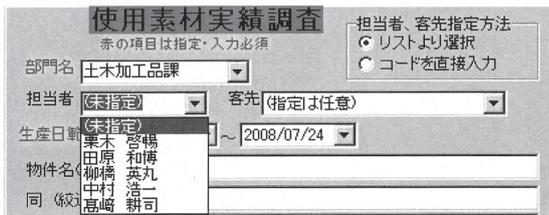


図2-17 ComboBoxのItem内容動的変更例-2

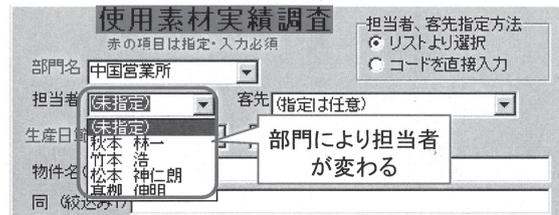


図2-18 ComboBoxのItem内容動的変更例-3

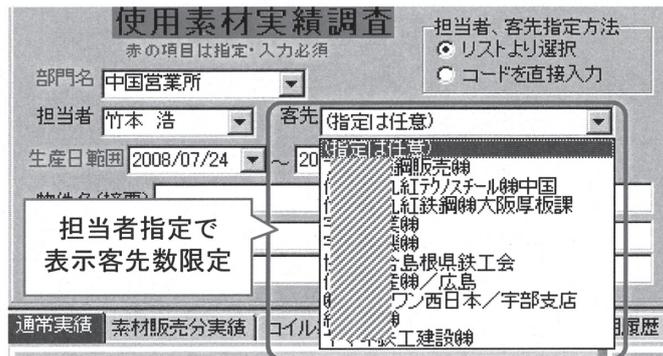


図2-19 各種マスターのローカルPC保存例



図2-20 各種マスターの定期更新例(起動時に実行)



図2-21 各種マスターの更新処理

```

for i:=1 to high(masterTB) do
begin
  BatchMoveMaster.Destination:=masterTB[i];
  BatchMoveMaster.Source:=masterOri[i];
  BatchMoveMaster.Execute;
end;

```

配列化したマスターをBatchMoveで更新 (batCopyモード)

図2-22 一定時間 アプリケーション操作無しでの表示更新処理

```

procedure TForm1.ApplicationEventsRefIdle(Sender: TObject;
var Done: Boolean);
begin
  RefreshTimer.Enabled:=true;
  Done:=true;
end;

procedure TForm1.ApplicationEventsRefMessage(var Msg: tagMSG;
var Handled: Boolean);
begin
  case Msg.message of
    WM_MOUSEMOVE, WM_KEYDOWN, WM_NCLBUTTONDOWN, WM_NCRBUTTONDOWN
  begin
    RefreshTimer.Enabled:=False;
  end;
end;
end;

```

アプリケーション操作が無ければTimerを起動

アプリケーション操作があればTimer停止  
→ Intervalのカウントは一旦リセットされる

↓

```

procedure TForm1.RefreshTimerTimer(Sender: TObject);
var
  t: integer;
begin
  // 定時更新処理

  for t:=1 to high(AutoRefreshBTN) do
  begin
    if AutoRefreshYN[t].checked=true then
      AutoRefreshBTN[t].onClick(sender);
  end;
end;

```

TimerのInterval設定時間に到達すればOnTimerイベントで更新処理起動

図2-23 表示自動更新の設定例

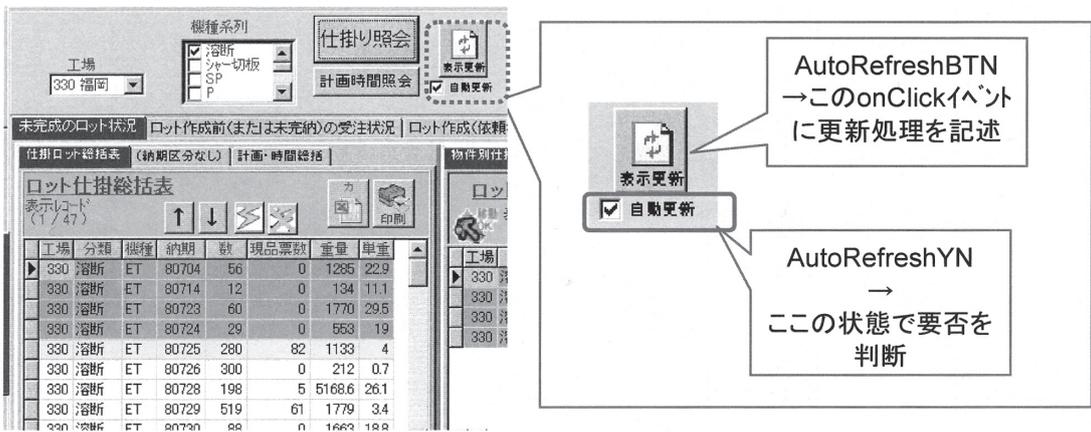


図2-24 項目の並び順カスタマイズの実現方法

```

procedure TForm1.ANYDBGridColumnMoved(Sender: TObject; FromIndex,
ToIndex: Integer); //GRIDの列並べ替えあれば並び順定義のStringListを書き直し
var
n,c:integer;
begin
n:=(Sender As TDBGrid).Tag;
mainQUERYorder[n].Clear; //Public変数の配列

for c:=0 to mainDBGrd[n].Columns.Count-1 do
mainQUERYorder[n].Add(mainDBGrd[n].Columns[c].FieldName); //並び順おりの配列生成
end;

↓

for n:=1 to HighMainN do //列の並び順,非表示項目をPublic変数より書き込み
begin
Ini.WriteString('Form',IntToStr(n)+'ColumnORDER',mainQUERYorder[n].CommaText);
Ini.WriteString('Form',IntToStr(n)+'ColumnNoVisible',mainQNoVisible[n].CommaText);
end;

↓

for i:=1 to HighMainN do //各系列項目並び順のINIファイル定義を使用して表示順を入れ替え
begin
mainQUERYorder[i]:=TStringList.Create; //順番用の配列生成
mainQUERYorder[i].CommaText:=Ini.ReadString('Form',IntToStr(i)+'ColumnORDER',''); //該当INIファイルより取込み
if mainQUERYorder[i].CommaText<>' ' then //順番定義あれば以下処理
begin
for j:=0 to mainQUERYorder[i].Count-1 do
begin
for k:=0 to mainQUERY[i].FieldCount-1 do
begin
begin
if mainQUERY[i].Fields[k].FieldName=mainQUERYorder[i].Strings[j] then
begin
mainQUERY[i].Fields[k].Index:=j; //INIファイルの順番にIndexを振りなおし
break;
end;

```

並べ替え時にOnColumnMoved イベントで並び順リスト(PublicのStringList)を変更

FormのCloseイベントにIniファイルに並び順等書き込み

FormのCreateイベントにIniファイルより並び順を設定

図2-25 項目の並び順のIniファイル記述例

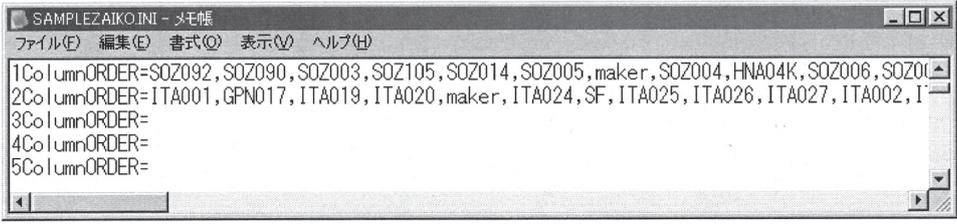


図2-26 項目の非表示化、再表示化操作例

素材No	元モデルNo	工場	分譲元	自丈	規格	メーカー	品種	品名	板厚	幅	長さ	寸	WI	LI	数量	重量	橋梁表面積	引当鉄引	引当重	引付部門	積付客先
0000001	330	430	1		新日鐵	2100	生産定尺	1.6	914	1829	3 x 6	0	0	42	882	0	0	0	0	0	0
1744532	330	0	1		新日鐵	2100	生産定尺	1.6	914	1829	3 x 6	0	0	4	84	0	0	0	0	0	0
0000002	330	430	1		新日鐵	2100	生産定尺	1.6	1219	2438	4 x 8	0	0	26	970	2	75	0	0	0	0
1351330	330	430	1		新日鐵	2100	生産定尺	1.6	1524	3048	5 x 10	0	0	154	8978	0	0	0	0	0	0
0000003	330	430	1		新日鐵	2100	生産定尺	2.3	914	1829	3 x 6	0	0	76	2296	0	0	0	0	0	0
0000005	330	430	1		新日鐵	2100	生産定尺	2.3	1524	3048	5 x 10	0	0	40	3356	0	0	0	0	0	0
1752379	330	430	1		新日鐵	2100	生産定尺	2.3	1524	3048	5 x 10	0	0	49	4112	0	0	0	0	0	0
0000006	330	430	1		新日鐵	2100	生産定尺	3.2	914	1829	3 x 6	0	0	169	7090	10	420	0	0	0	0
0000007	330	430	1		新日鐵	2100	生産定尺	3.2	1219	2438	4 x 8	0	0	32	2391	12	897	0	0	0	0
0000008	330	430	1		新日鐵	2100	生産定尺	3.2	1524	3048	5 x 10	0	0	58	6786	0	0	0	0	0	0
0000010	330	430	1		新日鐵	2100	生産定尺	4.5	914	1829	3 x 6	0	0	64	3782	0	0	0	0	0	0

再表示対象を選択

再表示する項目の番号を半角数字(1~2)で入力下さい

選択番号: 項目名

1: 製鉄所

2: 連番

0

OK      キャンセル

