XEAD Driver 活用レポート

ACCESS システムのリプレイス

下山 吉洋

2013/01/16

目　　次

第１章　XEAD Driver の開発思想

参考資料・Web サイト
『仕様書で「動的制御」される業務システム』
http://homepage2.nifty.com/dbc/xeadDriver.html

１．　XEAD Driver の使い方

（1）　XEAD Driver の特徴

ACCESS で作成した業務ソフトを、XEAD Driver にリプレイスしたいと何度か挑戦してきましたが、なかなかうまくいきませんでしたが、やっと道筋がみえてきました。
いきなり、現状のデータベースを XEAD Driver に乗せかえるのは難しいです。

XEAD Driver を利用するためにはフレームワークとしての開発思想の理解が必要です。
開発思想を理解しないまま移行を行っても、従来の苦しい開発から逃れることはできません。
XFOOO 機能を使用した。JavaScript アプリの作成にしかなりません。

今までのデータベース・テーブルを XEAD Driver で動かすのではなく、設計の方法を XEAD Driver にあわせるという意識が大切です。

XEAD Driver でシステムを開発するためには
３要素分析法による完全正規化されたデータモデリングがなされたテーブル設計が必要。
データ項目間の関数従属性を、複合主キーによる制約で派生、親子、参照に整理する。

この条件を満足させることにより、テーブルおよび複数のテーブル間の編集機能（CRUD）を自動発生することが出来る。
自動発生とは、コードの自動生成ではなく、データモデルの仕様に基づき、操作シーンに合った編集パネルを都度生成します。
関連するテーブルを連携して編集する紐付けもノンプロミングで設定するだけです。

XEAD Driver は、物理テーブルを継承して、独自に仮想テーブルの仕様を保持します。物理テーブルは情報の受け皿となるだけで、個別のデータベース特有の機能は利用しません。現在、Apache Derby、PostgreSQL、MySQL に対応しています。
注：XEAD Driver では、仮想テーブルの仕様を「テーブル定義」、物理テーブルのスキーマを「テーブルモジュール」と用語が定義されています。

仮想テーブルの仕様は、拡張子 xeaf ファイルとして、XML 形式で記述されています。
テーブルの構成情報とともに固有のテーブルを一次テーブルとした関連テーブルを紐付け、関連テーブルの参照が可能になります。関連テーブルは、コネクションの異なる別なデータベースでもかまいません。
　テーブル間に複雑な制約（ビジネスロジック）が必要な場合、XML 情報の中に JavaScript 言語によるスクリプトを記述することで、対応することが出来ます。
テーブルを操作する機能により、スクリプトの指定をしたい場合は、if(instance.functionID == 'YF021')のように、何の機能が自テーブルを操作しているか認識するためのインスタンスオブジェクトが用意されています。

XEAD Driver は、業務ソフトとしてのフレームワーク機能として
ユーザーログイン管理、操作ログ記録
楽観的共有ロック
伝票連番等の自動発番
PDF 印刷、メール送信、バーコード印刷
操作メニューのドラッグアンドドロップによる生成、ログインユーザーによる利用メニューの制限などが、予め用意されています。

XEAD Driver が想定していないこと。
基本的に正規化されたテーブルに対して、操作シーンに合った機能の自動生成をおこないます。

マスタ系の単票のレコードや、見積書など伝票系の見出し明細印刷は用意されていますが、
SQL を駆使した一覧表の印刷など、非正規化された情報については、その受け皿となる機能は用意されていないので個別に作成する必要があります。

一次テーブルと関連テーブルを結合し、一覧抽出した範囲のデータであれば、表計算への連携機能があるので表計算側で整形印刷することは可能です。

テーブルがあれば良いのなら、主キーのみで、属性は全て仮想フィールドとしたテーブルを作成すれば、正規化にこだわらない処理も可能になるが、内容によってネットワークトラフィックがボトルネックになるでしょう。

第２章　Aceess からの XEAD Driver へリプレイス

Microsoft Access は、大変小回りが利き便利な業務ソフト開発環境です。
テーブルとフォーム、テーブルとレポートの連携が便利で、ODBC 経由で様々なデータベースサーバーへも接続できる。
強力なパススルークエリ、VBA を使用しての WindowsAPI を呼び出しで PC の機能をフルに利用できるなど、大変魅力のあるアプリケーションです。
ユーザーの相互協力による、Web サイト等での情報が豊富です。

しかし、次の点で苦労がありました。
商用プロダクトなので内部がブラックボックスであり、「このプログラムは不正な処理を行ったので強制終了されます。」などのエラーへの対策への原因追求が難しい。
サービスパックの適用がされるまで、対処が不可能な場合も少なくない。
ソースコードの管理が難しいのでバージョン管理に難がある。
開発環境 VBE が現状では機能不足である。

Eclipse のプラグイン等で VBA が開発できれば良いのと思うのだが実現されるのは困難で、むしろ開発プラットフォームを変更するほうが、様々なリソースを活用でき、Access に固執する必要がなくなってきた。

１．　Access からの XEAD Driver へリプレイス

（１）　サンプルシステム

元ネタは http://www5a.biglobe.ne.jp/~fpeo2/mag2smpl.htm
MyACTv3r6m 会計管理システムです。
詳細な解説がメーリングリストのバックナンバーとして整理されています。

基本のデータモデルは、渡辺幸三著、業務別データベース設計のためのデータモデリング入門
第６章　会計管理が元になっています。興味のある方はこちらも参照してください。

Access のテーブルを PostgreSQL へアップサイジングしました。
XEAD Driver のお作法に合わせないと、移行が難しいので、変更点を記録します。

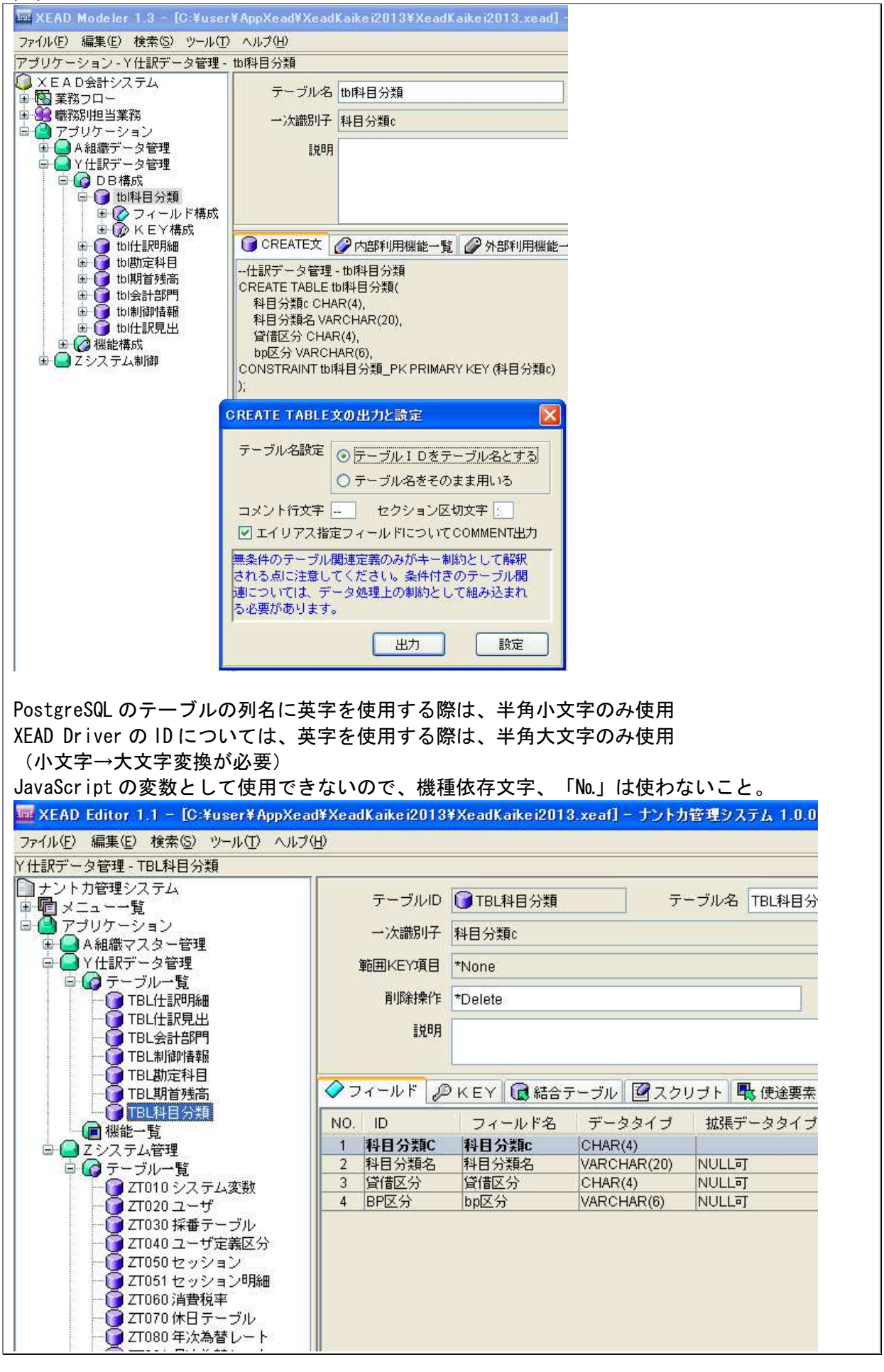
実際にシステムの開発をする場合には、細かな変更点ですがシステムの規模によっては影響の範囲が級数的に広がります。小規模なサブシステムで慣れてから全面移行することをお勧めします。

おそらく、慣れた段階で設計をし直し XEAD Driver に最適化することになると予想されます。

（２）　Access のテーブルを PostgreSQL へアップサイジング注意点

XEAD Modeler	XEAD Driver(Editor)	
tbl 科目分類(英字は半角小文字)	ID(英字は半角大文字)	フィールド名
--仕訳データ管理 - tbl 科目分類 CREATE TABLE tbl 科目分類(　科目分類 c CHAR(4), 　科目分類名 VARCHAR(20), 　貸借区分 CHAR (4), 　bp 区分 VARCHAR(6), CONSTRAINT tbl 科目分類_PK PRIMARY KEY (科目分類 c)); Sql 文の発行(英字は半角小文字)	科目分類 C 科目分類名 貸借区分 BP 区分 TableOperator は ID (英字は半角大文字) 固まる	基本的に自由、 大文字でも小文字でも良い。

（3）　Modeler のツール、create table 文作成のお手本



PostgreSQL のテーブルの列名に英字を使用する際は、半角小文字のみ使用
XEAD Driver の ID については、英字を使用する際は、半角大文字のみ使用
（小文字→大文字変換が必要）
JavaScript の変数として使用できないので、機種依存文字、「№」は使わないこと。

（４）　Access のテーブルを PostgreSQL へアップサイジング

参考資料・Web サイト

『ぷにじぃの備忘録 [PostgreSQL] left()と right()』

http://nejimakitori-chronicle.seoid.net/works/postgresql/postgresql-left%E3%81%A8-right/

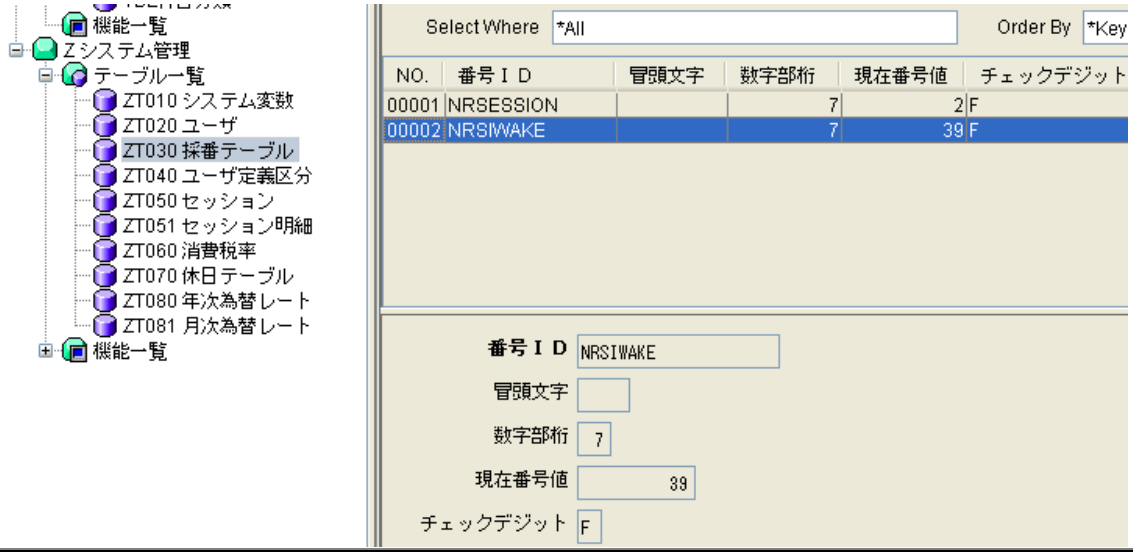
バイナリ型の主キーを文字型に変更した場合に、０埋めの文字型に変換してください。
SQL の UPDATE 文を使用する方法を例示します。
PostgreSQL には、left,right 関数がありませんので、参考サイトを参照のうえ、ユーザー関数として登録することをお勧めします。

仕訳 no の置換

```
select right('00000000' || 仕訳 no,7) from tbl 仕訳見出 ;

UPDATE tbl 仕訳見出
   SET 仕訳 no=right('00000000' || 仕訳 no,7) ;
```

（５）　自動採番値　NRSIWAKE　の追加

自動採番値　NRSIWAKE　の追加、現在番号値は最大の採番+1 を登録してください。



２．　XEAD Driver のお作法

（１）　自動採番値の型指定

自動採番値
データタイプが CHAR の場合にのみ選択可能。システム制御テーブルの「採番テーブル」に置かれた採番キーＩＤを指定することで、レコード追加時に自動的に番号値がセットされます。

（２）　明細行番

明細行番は smallint(4)　→　Modeler でも統一する。

一次識別子の場合、KEY タイプの右側に「最後のフィールドを行番号として処理する」のチェックボックスが示されます。これをチェックすると、レコード追加時に明細行番号として自動的にカウントアップされます。ただし、KEY が複合キーであり、かつ最後尾フィールドが整数項目でなければなりません。そうでない場合、この指定は無視されます。

（3）　全角文字の使用（Unicode）と制限事項

XEAD Driver の仕様では、基本的には、全角文字の使用も問題ない。
　ただし、一部の文字は、JavaScript で変数名として使用できないための制限がある。

Access では、主キーに良く使われている、省略文字「№」、Unicode：0x2116 が使えません。

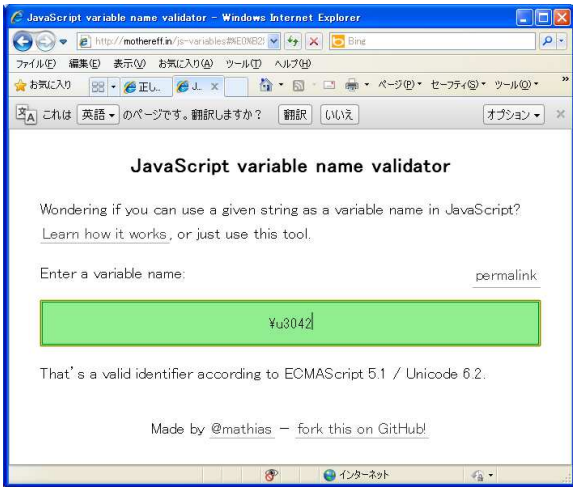
伝票№　は、伝票 no や伝票 id 等に置換すればよいでしょう。
プログラム、SQL、フィールド名を一括して変更することが可能であればの話ですが。
新規の案件にはこれらのルールを適用することで、スクリプト内での全角文字の使用が可能になります。

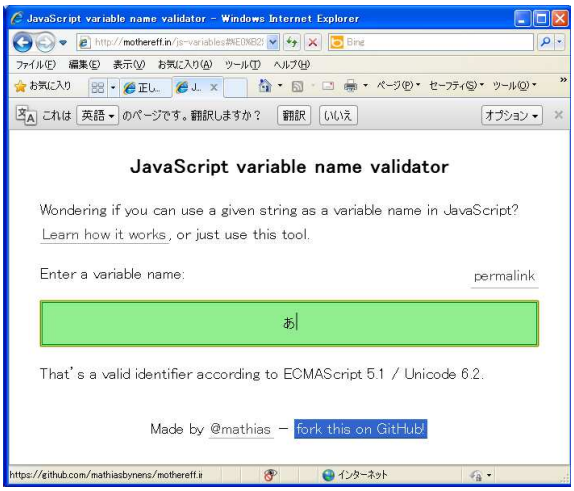
論理名と物理名を全角文字で統一して開発することが可能になるでしょう。
ACCESS での開発に慣れた者にはありがたい環境ですね。

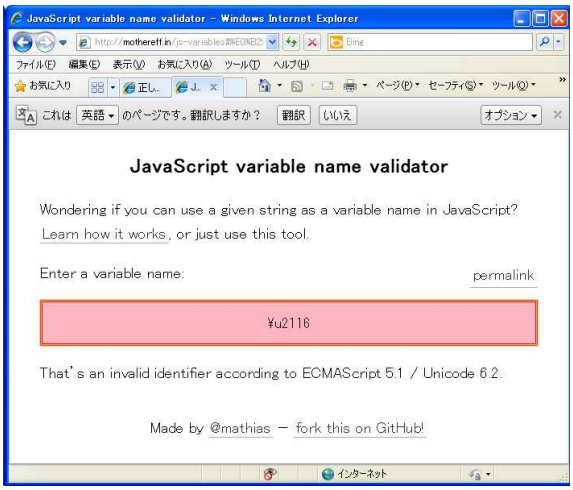
（4）　JavaScript 有効な変数名

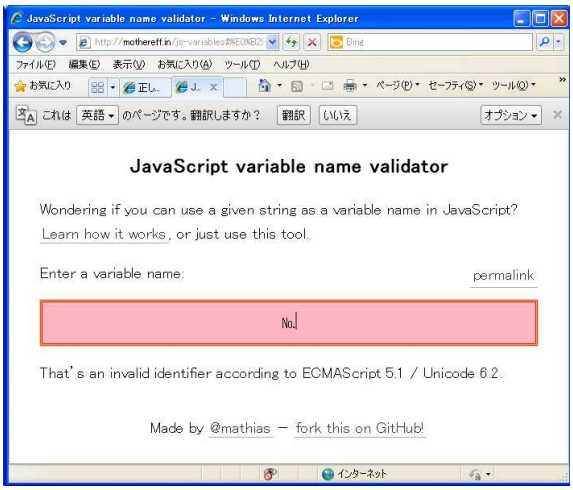
XEAD Driver のスクリプト内の変数名に一部の全角文字が使えない制約についての参考資料。

参考：正しい JavaScript の変数名の話をしよう
http://tech.a-listers.jp/2012/02/24/valid-javascript-variable-names/
変数チェッカー
http://mothereff.in/js-variables#%E0%B2%A0%5f%E0%B2%A0
全角の「あ」と省略文字の「№」、グリーンの背景は、有効な変数、ピンクの背景は無効。









※将来的には JavaScript 変数の仕様が改定され、制約が変更されれば省略文字が利用できるようになるかもしれません。

(５)　設計手順の概要

３要素分析法によって、テーブルの設計がなされているならば、
XEAD Driver 上のテーブル設計が終われば、画面パネルは、自動的に生成され、少しの調整で使い易いシステムが構築できます。

パネル上に表示されるテキストエリアなどの設定は、*Auto 指定がされており適切なサイズで表示されます。個別に幅や行数を設定することは勿論可能です。

もし、なんらか特殊な条件が必要な場合、必要最低限のプログラムはテーブル側にスクリプトとして登録します。あらかじめ用意されている機能パネル側へは一切コードを書き加えることはありません。

(６)　スクラッチなプログラミングの概要

もし、開発者が独自にプログラミングしたい場合は機能 Script ランチャー(XF000)を利用し、JavaScript を記述して自由にシステムを構築することが出来ます。

サンプルとして、スクリプト関数に、JDBCSwingQuery を組み込みました。
MyACTv3r6 では、Access のクエリ機能を組み合わせて各帳票のレコードソースが生成されていますが、PostgreSQL では、SQL 文の中で With 句を使用することが出来るので、補助的なクエリは With 句で置き換え、各帳票の SQL 文に埋め込みました。
SQL のレコードのリザルトセットを Jtable で表示します。

(７)　開発効率

　３要素分析法によるデータモデリングを習得し、XEAD Driver の操作に慣れれば、おそらく XEAD Driver を利用した場合の開発効率は、Access で開発する場合の１０倍から２０倍それ以上に相当するでしょう。それにあわせ、ストレスも少なくなり、開発コストも下がります。
　ノンプログラミングで業務ソフトを開発することが可能になるので、プログラミングができない人でも表計算程度の関数記述、設定でシステムを構築が可能になります。
　今後の開発は、データモデリングの習得が一番の鍵となるでしょう。

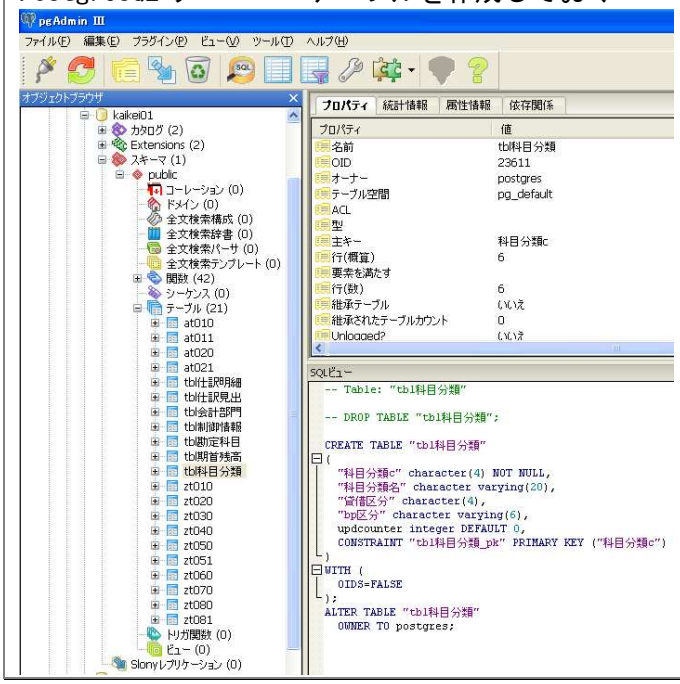
(８)　XEAD Driver を利用するメリット

OSS である。
３要素分析法に基づいたシステム構築が実現できる。
XEAD Driver は Rhino を利用して、JavaScript を動作させることが出来る。
JavaScript は、Swing のような class など、Java との相互利用がし易い。
XEAD Driver は java で記述されているので、ソースを閲覧変更するために Eclipse が利用できる。

XEAD Driver を構築するための XEAD Editor はデータベースの接続、操作が容易で強力な開発ツールとしても利用できる。

（9）　テーブルのアップサイジングの手順

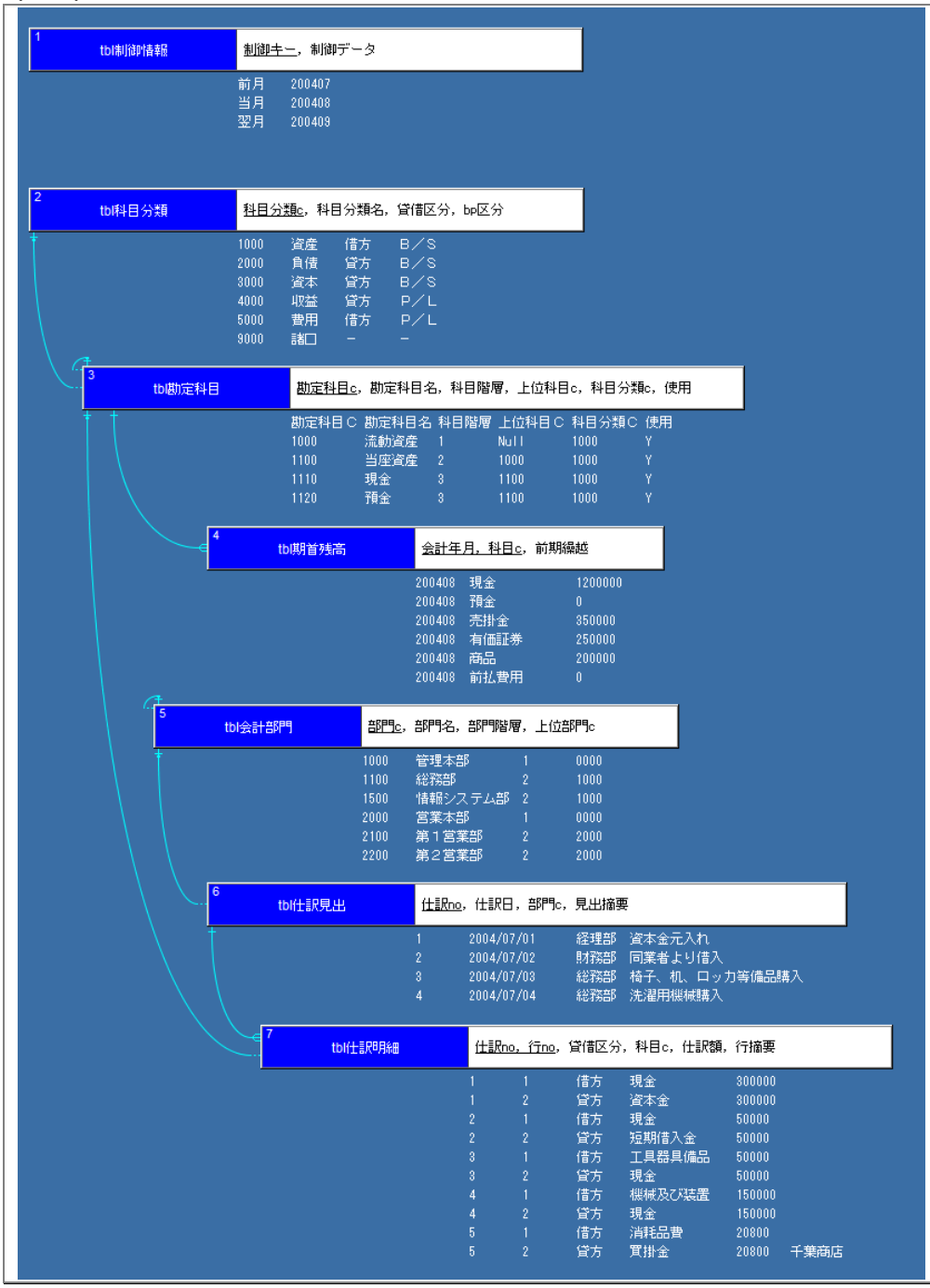
PostgreSQL サーバーへテーブルを作成しておく



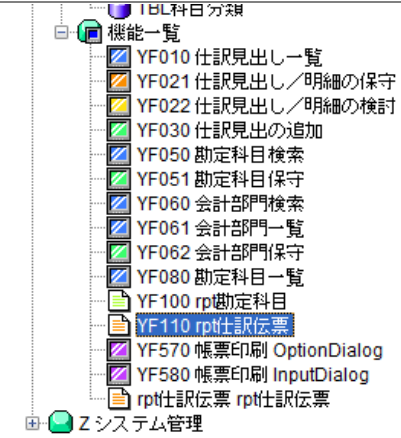
Access　MDB　テーブルを VBA で、PostgreSQL サーバーへ送る

```
Option Compare Database
Option Explicit

Private Const cModuleName = "basAccess"

Public Sub Postgres2Access04()
    On Error GoTo Err

    Dim strSQL As String
    Dim strServer As String
    Dim Conn As ADODB.Connection
    Set Conn = CurrentProject.Connection

    Dim stime As Variant
    stime = Timer
    strServer = "[ODBC;DRIVER={PostgreSQL Unicode};DATABASE=kaikei01;SERVER=localhost;UID=postgres;PWD='';]"

    ' ＳＱＬの作成
    strSQL = "delete from " & strServer & ".tbl 科目分類 ;"

    Conn.Execute strSQL

    ' ＳＱＬの作成
    strSQL = "insert INTO " & strServer & ".tbl 科目分類 select * from tbl 科目分類 ;"

    Conn.Execute strSQL
    Conn.Close
    Set Conn = Nothing
    Debug.Print Format(Timer - stime, "#0.000sec")

Ext:
    On Error Resume Next
    Exit Sub

Err:
    MsgBox "プロシージャ : " & cModuleName & ".Postgres2Access04" & vbCrLf & _
           "エラー番号　: " & Err.Number & vbCrLf & vbCrLf & _
            Err.Description, vbOKOnly + vbCritical
    On Error Resume Next
    Resume Ext
End Sub
```

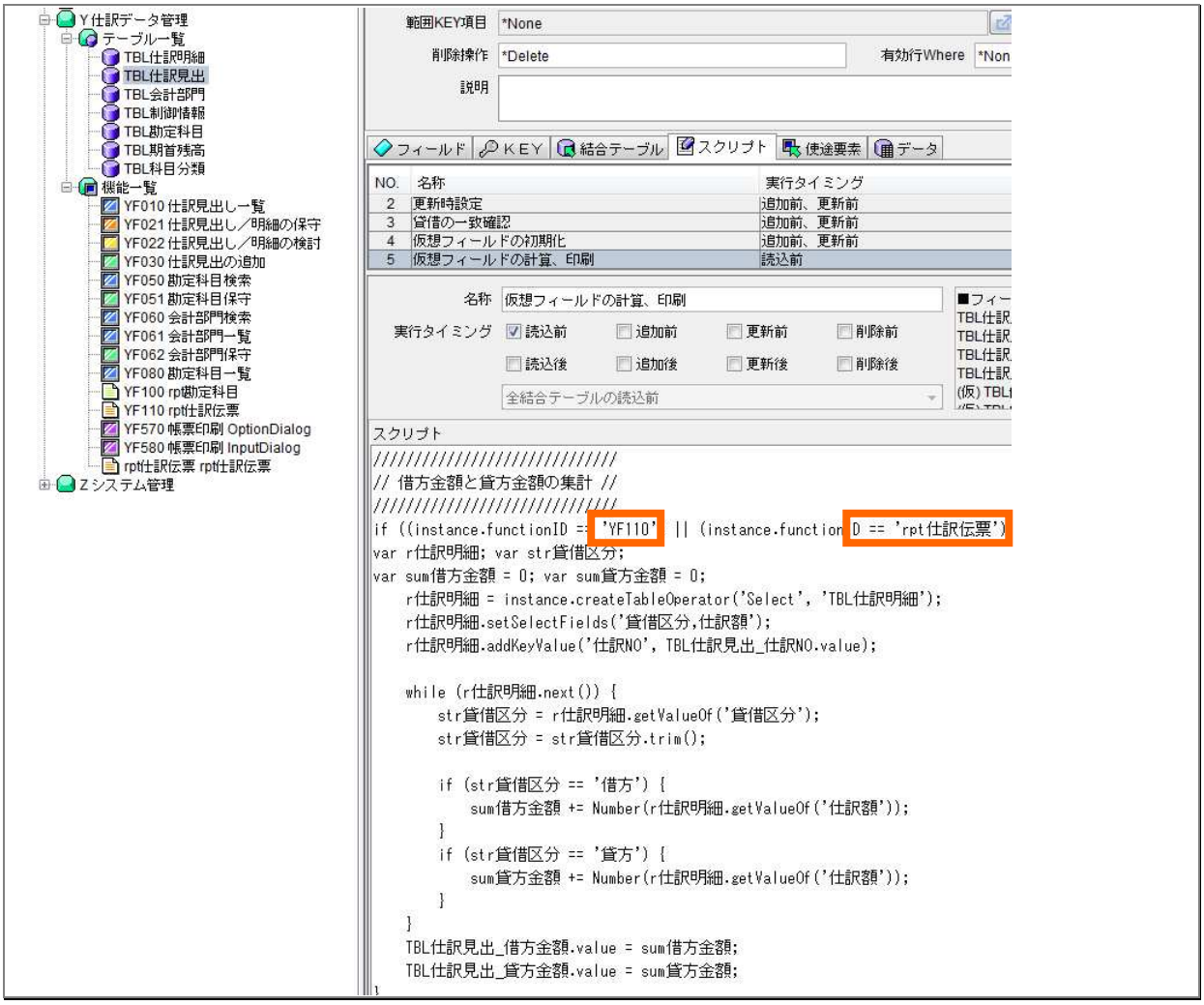
（１０）サンプルシステムのデータモデル



（１１）帳票印刷



YF100、YF110　rpt 仕訳伝票のように ID に全角を使用することが可能です。
仕訳見出の仮想フィールド借方金額、貸方金額のスクリプト


Y仕訳データ管理
テーブル一覧
TBL仕訳明細
TBL仕訳見出
TBL会計部門
TBL制御情報
TBL勘定科目
TBL期首残高
TBL科目分類
機能一覧
YF010 仕訳見出し一覧
YF021 仕訳見出し／明細の保守
YF022 仕訳見出し／明細の検討
YF030 仕訳見出の追加
YF050 勘定科目検索
YF051 勘定科目保守
YF060 会計部門検索
YF061 会計部門一覧
YF062 会計部門保守
YF080 勘定科目一覧
YF100 rpt勘定科目
YF110 rpt仕訳伝票
YF570 帳票印刷 OptionDialog
YF580 帳票印刷 InputDialog
rpt仕訳伝票 rpt仕訳伝票
Zシステム管理
範囲KEY項目 *None
削除操作 *Delete
有効行Where *Non
説明
フィールド
KEY
結合テーブル
スクリプト
使途要素
データ
NO. 名称 実行タイミング
2 更新時設定 追加前、更新前
3 貸借の一致確認 追加前、更新前
4 仮想フィールドの初期化 追加前、更新前
5 仮想フィールドの計算、印刷 読込前
名称 仮想フィールドの計算、印刷
実行タイミング 読込前 追加前 更新前 削除前
読込後 追加後 更新後 削除後
全結合テーブルの読込前
スクリプト
////////////////////////////////
// 借方金額と貸方金額の集計 //
////////////////////////////////
if ((instance.functionID == 'YF110' || (instance.function D == 'rpt仕訳伝票')
var r仕訳明細; var str貸借区分;
var sum借方金額 = 0; var sum貸方金額 = 0;
r仕訳明細 = instance.createTableOperator('Select', 'TBL仕訳明細');
r仕訳明細.setSelectFields('貸借区分,仕訳額');
r仕訳明細.addKeyValue('仕訳NO', TBL仕訳見出_仕訳NO.value);
while (r仕訳明細.next()) {
str貸借区分 = r仕訳明細.getValueOf('貸借区分');
str貸借区分 = str貸借区分.trim();
if (str貸借区分 == '借方') {
sum借方金額 += Number(r仕訳明細.getValueOf('仕訳額'));
}
if (str貸借区分 == '貸方') {
sum貸方金額 += Number(r仕訳明細.getValueOf('仕訳額'));
}
}
TBL仕訳見出_借方金額.value = sum借方金額;
TBL仕訳見出_貸方金額.value = sum貸方金額;

要望
スクリプトで XEAD Driver のエラーメッセージを参照したい。
Res_ja を参照できるとありがたい。

開発時の作業効率化のため、起動時メニューのアクティブページをトグルで指定できるとありがたい。
起動時に、毎回デバッグしたいページをクリックするか、テスト機能をトップページに貼り付けている。最終的には然るべきページに整理することになるので、開発の進捗に合わせてメニューの開始ページを切り替えたい。
ユーザが業務で利用する場合も、ログイン後自動的に良く使うメニューが開くと幸せかも。

テキスト選択の右クリックのショートカットメニュー、コピー、ペーストが欲しい。

XF310 見出し明細保守
デバッグに showMessageDialog を動作させると、明細の表示がおかしくなる。

第３章　　スクリプトの追加

（1）　rpt 仕訳伝票　印刷のために仮想フィールドを計算するスクリプト

実行タイミング、レコード読み込み前

```
////////////////////////////
// 借方金額と貸方金額の集計 //
////////////////////////////
if ((instance.functionID == 'YF110') || (instance.functionID == 'rpt 仕訳伝票')) {
var r 仕訳明細; var str 貸借区分;
var sum 借方金額 = 0; var sum 貸方金額 = 0;
          r 仕訳明細 = instance.createTableOperator('Select', 'TBL 仕訳明細');
          r 仕訳明細.setSelectFields('貸借区分,仕訳額');
          r 仕訳明細.addKeyValue('仕訳 NO',          TBL 仕訳見出_仕訳 NO.value);

          while (r 仕訳明細.next()) {
                    str 貸借区分 = r 仕訳明細.getValueOf('貸借区分');
                    str 貸借区分 = str 貸借区分.trim();

                    if (str 貸借区分 == '借方') {
                              sum 借方金額 += Number(r 仕訳明細.getValueOf('仕訳額'));
                    }
                    if (str 貸借区分 == '貸方') {
                              sum 貸方金額 += Number(r 仕訳明細.getValueOf('仕訳額'));
                    }
          }
          TBL 仕訳見出_借方金額.value = sum 借方金額;
          TBL 仕訳見出_貸方金額.value = sum 貸方金額;
}
```