WPS による各種データベース入出力方法

（WPS バージョン 3.1.1）

第1版

2015 年 1 月

株式会社ブレインパッド

目次

1. はじめに

WPS は DB2, MySQL, Netezza, Oracle, PostgreSQL, SQL Server, Teradata などの商用、またはオープンソースのデータベース（以下 DB とも表記）に対するアクセスインターフェース機能をサポートしています。インターフェース機能に含まれるのは、データの読み書き、テーブルの作成・編集・削除を含みます。サポートするインタフェース方式は、DB ごとに固有のコマンドライン接続方式（CLI ドライバー、ネイティブドライバー）に加えて、多くの DB に対してそれぞれの DB ベンダーから提供されているオープンデータベース接続方式（ODBC ドライバー）の両方です。

本資料は 64 ビット版 Windows8.1 上で動作する以下のデータベースプロダクトをローカル環境にインストールし、各プロダクトに付随しているサンプルデータベースを例として CLI ドライバー経由および ODBC ドライバ経由で WPS3.1.1 からそれぞれの DB テーブルの入出力を行う場合を例示し、WPS の DB インタフェース機能の使い方を説明したものです。

DB2 Express-C 10.5（SAMPLE データベース）
MySQL 5.6（sakila データベース（スキーマ））
ORACLE Express Edition 11.2（scott スキーマ）
PostgreSQL 9.3.5（dvdrental データベース）
SQL Server 2012 Express Edition（AdventureWorks2012 データベース）

なお、本資料では、主に SHIFT-JIS エンコード（MS932 エンコード）の日本語データを取扱っています。そのため、WPS ワークベンチの設定が、以下のとおり設定されていることを前提としています。

・起動オプションに encoding="SHIFT-JIS" [1]
・ワークスペースのテキストファイルエンコードに MS932[2]

1 「WPS サーバーエクスプローラー」ビューの「ローカルサーバー」を右クリック→「プロパティ(R)」をクリック → 「プロパティ：ローカルサーバー」画面の左領域の「起動オプション」をクリック → 右領域の「追加」をクリック → 「起動オプション」画面出現 →「選択」をクリック → 出現した「起動オプションを選択」画面のオプションリストの中から「ENCODING」を探してクリック → 「OK」をクリック → 「起動オプション」画面の「名前：」欄に ENCODING が入っていることを確認 → 「値：」欄に SHIFT-JIS を入力 → 「OK」。なお、バージョン 3.1.1 から LOCALE="JAPANESE_JAPAN"を指定できるようになり、この指定を行うと、ENCODING="SHIFT-JIS" に自動設定されるようになりました。

2 「メニューバー」の「ウィンドウ(W)」→ 「設定(P)」→ 左領域の「一般」→ 「ワークスペース」をクリック → 右領域の左下部にある「テキスト・ファイル・エンコード(T)」の「その他(O)」ラジオボタンをクリック → 入力欄に MS932 とキー入力 → 「OK」。なお、バージョン 3.1.1 では既定の設定になりました。

プログラムをバッチ実行する場合は、-encoding SHIFT-JIS （起動オプション）を指定して実行することが必要です。

［WPS がサポートしているデータベース］

WPS3.1.1 でアクセスインタフェースがサポートされている DB とそのインターフェース機能の種類とアクセスの種類は以下のとおりです。

（インタフェース機能）

データベース	読み取り	書き込み	更新	新規テーブルの作成	パススルー（暗黙）	パススルー（明示）	バルクロード	バルクアップロード
Action Matrix	○	○	○	○	○	○	×	×
DB2	○	○	○	○	○	○	○ （z/OS以外）	×
Greenplum	○	○	○	○	○	○	○	×
Informix	○	○	○	○	○	○	×	×
Kognitio	○	○	×	○	○	○	×	×
MySQL	○	○	○	○	○	○	×	×
Netezza	○	○	○	○	○	○	○	○
ODBC	○	○	○	○	○	○	○ （SQL Serverのみ）	×
OLEDB	○	○	○	○	○	○	×	×
Oracle	○	○	○	○	○	○	○	×
PostgreSQL	○	○	○	○	○	○	○	×
SAND	○	○	○	○	○	○	○	×
SQL Server	○	○	○	○	○	○	○	×
Sybase	○	○	○	○	○	○	×	×
Teradata	○	○	○	○	○	○	×	×
Vertica	○	○	○	○	○	○	×	×

OS 別サポート状況は以下のとおりです。

（OS環境）

データベース	AIX pSeries(Power)	Linux x86	Mac OS X	Solaris SPARC	Solaris x86	Windows x86	Linux System z	z/OS System z
Action Matrix	×	○	×	×	×	○	×	×
DB2	○	○	×	○	○	○	○	○
Greenplum	○	○	○	×	○	○	×	×
Informix	×	×	×	×	×	○	×	×
Kognitio	×	○	×	×	○	○	×	×
MySQL	○	○	○	○	○	○	○	×
Netezza	○	○	○	○	○	○	×	×
ODBC	○	○	○	○	○	○	×	×
OLEDB	×	×	×	×	×	○	×	×
Oracle	○	○	○	○	○	○	○	×
PostgreSQL	×	○	○	×	○	○	○	×
SAND	○	○	×	○	○	×	×	×
SQL Server	×	×	×	×	×	○	×	×
Sybase	○	○	×	○	○	○	×	×
Teradata	○	○	×	○	○	○	×	○
Vertica	○	○	×	×	○	○	×	×

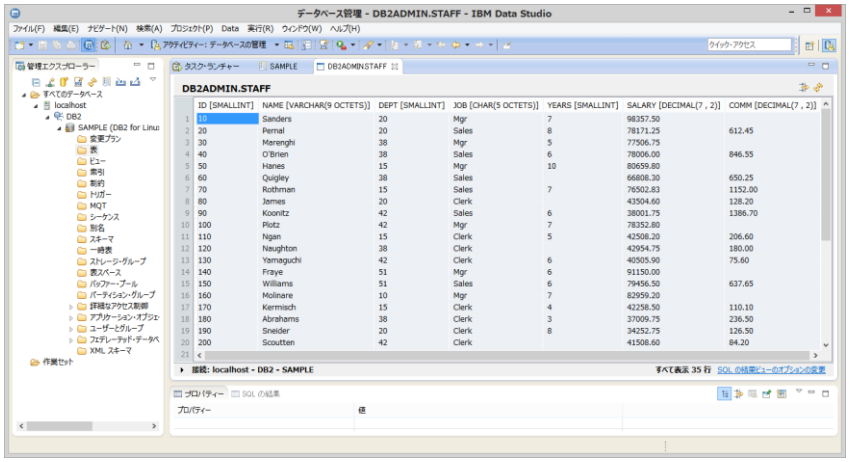
※ ODBC 接続については OS や対象 DB によってサードパーティ製品の導入が必要になる場合もあります。

なお、ここに示した例は、ローカル環境にインストールした DB に管理者ユーザーでログインできる環境で実行しています。ネットワーク環境や、セキュリティ環境の違いにより、ここで示したプログラムや設定では DB アクセスが実行できない場合もあります。その場合は、DB アクセス環境等を確認の上、必要な修正を加えて実行してください。

2. DB2 とのインタフェース

［SAMPLE サンプル DB］

DB2 Express-C 10.5 に付随する sample データベースを例として用います。



2.1. ネイティブドライバー経由の DB テーブルの入出力

まず、コマンドラインインタフェース（CLI ドライバー、ネイティブドライバー）を用いて WPS から DB テーブルの入出力を行ってみましょう。
ネイティブドライバーは DB と一緒にインストールされており、一般に ODBC ドライバーより高速に DB アクセスができる点でメリットがあると言われています。
WPS の DB2 エンジンはネイティブドライバーを経由して DB2 データベースとアクセスを行う仕組みを提供します。

［SQL 言語を使用した DB 読み取り］

DB へのアクセスは SQL 言語を用いて行うことが多く、WPS でも SQL プロシジャを用いて SQL 言語による DB アクセスが可能です。

SQL 言語を用いて WPS からネイティブドライバー経由で DB2 の DB アクセスを行う場合は、以下のように、エンジン名として DB2 を指定し、CONNECT 文のカッコ内に datasrc=DB 名 などの接続オプションや schema=スキーマ名、dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを必要に応じて指定します。

以下は sample データベースの staff テーブルから、DEPT=10 または 20 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.staff を作成しています。

```
proc sql;
  connect to DB2 (datasrc=sample);
  create table staff as
  select *
  from connection to DB2
  (
    select *
    from staff
    where DEPT in (10, 15);
  );
  disconnect from DB2;
quit;
```

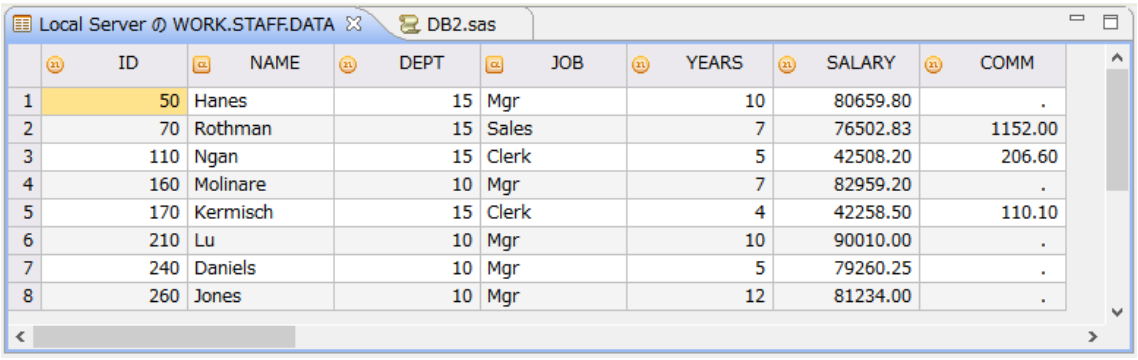
※ 3 行目の create table 文は WPS データセット WORK.staff を作成する指定です。
8 行目の from 節で指定した staff は sample データベースの staff テーブルを参照しています。（データベースのテーブル名の指定は、本来、「スキーマ名.テーブル名」の形式で指定します。しかし、テーブルの所属するスキーマ名が接続する DB2 ユーザー名と同じ場合は「スキーマ名.」の部分は省略できるため、ここでは省略しています。）

(WPS ワークベンチの実行ログ)

```
8         proc sql;
9           connect to DB2 (datasrc=sample);
NOTE: Successfully connected to database DB2 as alias DB2.
10          create table staff as
11          select *
12          from connection to DB2
13          (
14            select *
15            from staff
16            where DEPT in (10, 15);
17          );
NOTE: Data set "WORK.staff" has 8 observation(s) and 7 variable(s)
18          disconnect from DB2;
NOTE: Successfully disconnected from database DB2.
19        quit;
```

```
NOTE: Procedure sql step took :
      real time : 1.594
      cpu time  : 0.031
```

作成された WPS データセット WORK.staff データセットの内容を確認します。

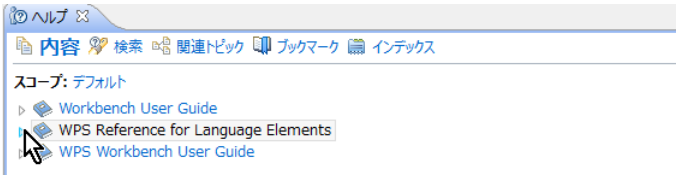


Local Server の WORK.STAFF.DATA | DB2.sas

	ID	NAME	DEPT	JOB	YEARS	SALARY	COMM
1	50	Hanes	15	Mgr	10	80659.80	.
2	70	Rothman	15	Sales	7	76502.83	1152.00
3	110	Ngan	15	Clerk	5	42508.20	206.60
4	160	Molinare	10	Mgr	7	82959.20	.
5	170	Kermisch	15	Clerk	4	42258.50	110.10
6	210	Lu	10	Mgr	10	90010.00	.
7	240	Daniels	10	Mgr	5	79260.25	.
8	260	Jones	10	Mgr	12	81234.00	.

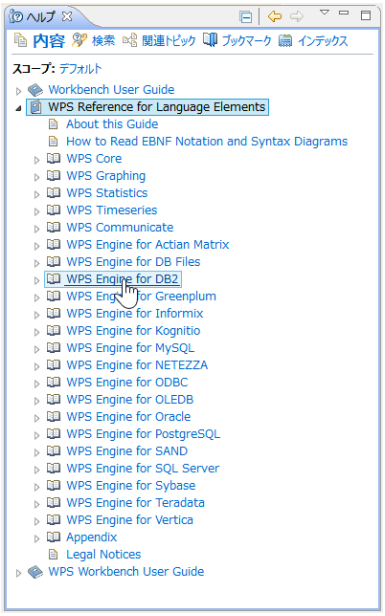
※ 注意 : DB2 ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。指定可能なオプションは、以下のように、ヘルプビューで確認できます。

まず、ヘルプビュー[3]の左上にある「内容」をクリックします。ヘルプ全体の目次が表示されますので、WPS Reference for Language Elements をクリックします。

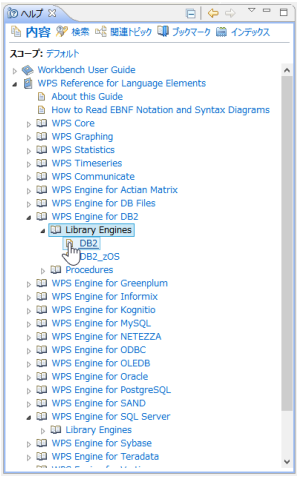


WPS レファレンスを展開し、WPS Engine for DB2 を選択します。

[3] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。

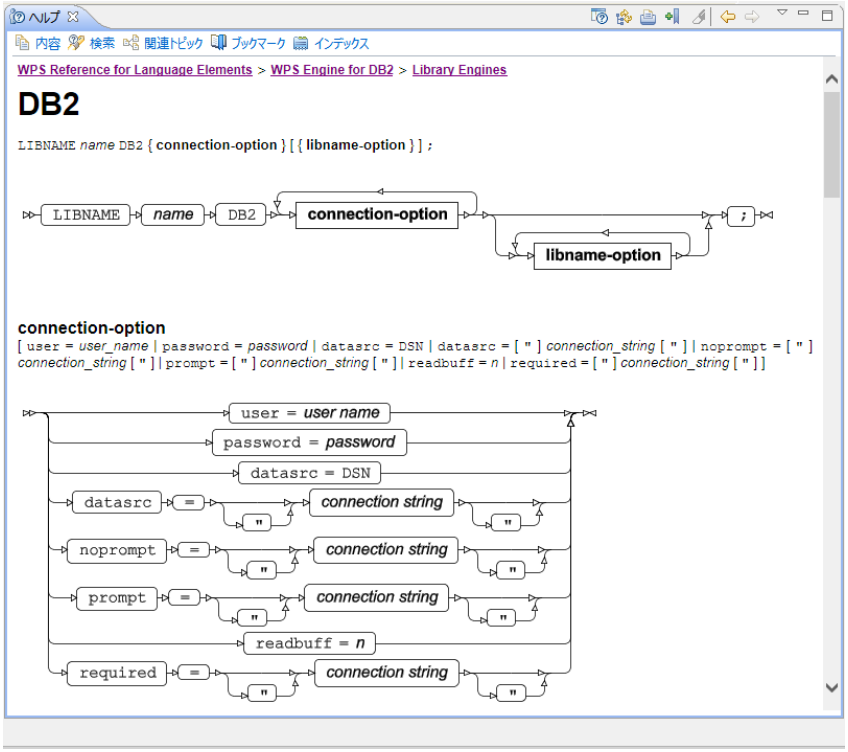


Library Engines → DB2 を選択します。（z/OS 版[4]の場合は DB2_zOS を選択）



データベース接続オプション（connection option）が表示されます。

[4] z/OS 版では procedurs にあるデータ抽出用の DB2EXT プロシジャも利用可能です。



これらの接続オプション（user= password= datasrc= noprompt= prompt= required=）は DB2 ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプション指定としても、また、次に節で説明する libname ステートメントによる DB 接続オプションとしても用いることができ、必要に応じて指定します。

［libname ステートメントによる DB 入力］

WPS のエディタから以下のプログラムを入力し、実行します。

```
libname in DB2 datasrc=sample;
```

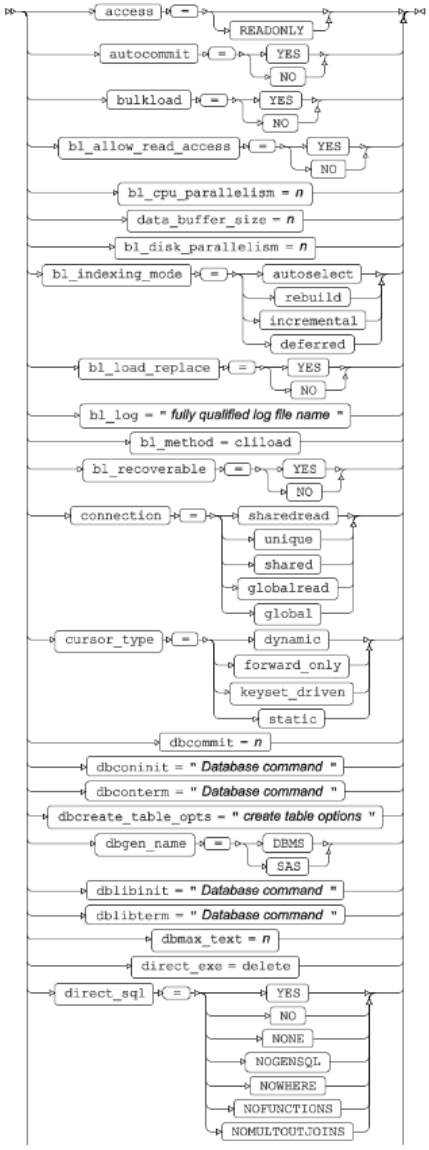
この指定は、WPS がデータベース sample に DB2 ネイティブドライバー経由で接続し、sample データベースのテーブルとビューをライブラリ名 in で参照できるようにする指定です。（入出力両方可能） ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

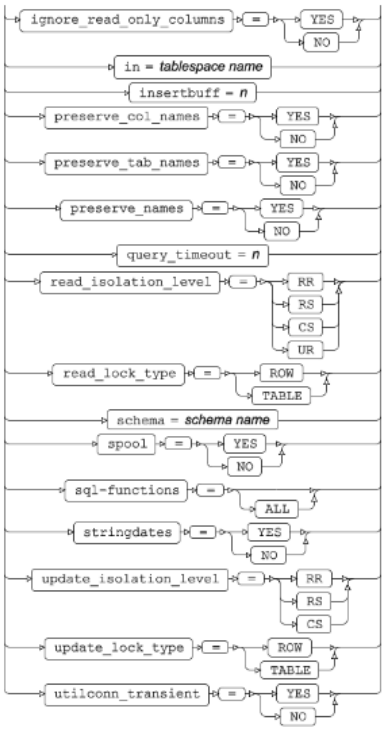
さらに、libname ステートメントには schema=スキーマ名、dbmax_text=n（文字テキストの読み取り最大長さの設定）のような libname オプションを追加指定できます。以下のような libname オプションが指定可能です。（前出の DB2 エンジンのヘルプ画面から確認できます。）なお、schema=スキーマ名を指定しない場合の既定のスキーマ名は DB 接続ユーザ

一名です。

libname-option

[access = [READONLY] | autocommit = [YES | NO] | bulkload = [YES | NO] | bl_allow_read_access = [YES | NO] | bl_cpu_parallelism = *n* | data_buffer_size = *n* | bl_disk_parallelism = *n* | bl_indexing_mode = [autoselect | rebuild | incremental | deferred] | bl_load_replace = [YES | NO] | bl_log = " *fully_qualified_log_file_name* " | bl_method = cliload | bl_recoverable = [YES | NO] | connection = [sharedread | unique | shared | globalread | global] | cursor_type = [dynamic | forward_only | keyset_driven | static] | dbcommit = *n* | dbconinit = " *Database_command* " | dbconterm = " *Database_command* " | dbcreate_table_opts = " *create_table_options* " | dbgen_name = [DBMS | SAS] | dblibinit = " *Database_command* " | dblibterm = " *Database_command* " | dbmax_text = *n* | direct_exe = delete | direct_sql = [YES | NO | NONE | NOGENSQL | NOWHERE | NOFUNCTIONS | NOMULTOUTJOINS] | ignore_read_only_columns = [YES | NO] | in = *tablespace_name* | insertbuff = *n* | preserve_col_names = [YES | NO] | preserve_tab_names = [YES | NO] | preserve_names = [YES | NO] | query_timeout = *n* | read_isolation_level = [RR | RS | CS | UR] | read_lock_type = [ROW | TABLE] | schema = *schema_name* | spool = [YES | NO] | sql-functions = [ALL] | stringdates = [YES | NO] | update_isolation_level = [RR | RS | CS] | update_lock_type = [ROW | TABLE] | utilconn_transient = [YES | NO]]





Unsupported Options

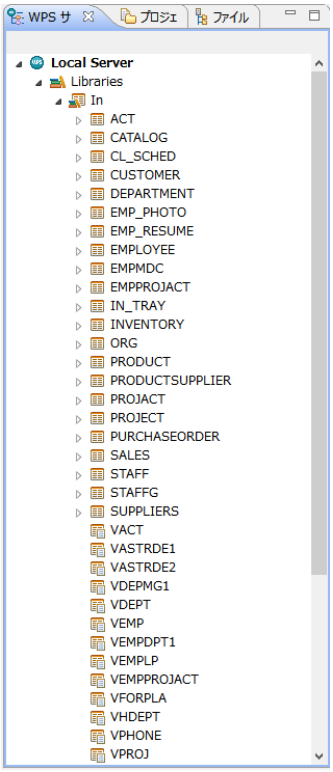
The following options are not yet supported:

- AUTHDOMAIN
- BL_ALLOW_WRITE_ACCESS
- BL_COPY_LOCATION
- BL_EXCEPTION
- BL_OPTIONS
- BL_REMOTE_FILE
- CONNECTION_GROUP
- DBINDEX
- DBNULLKEYS
- DBPROMPT
- DBSASLABEL
- DBSLICEPARM
- DEFER
- REREAD_EXPOSURE
- SQL_FUNCTIONS_COPY

さて、実行すると、ログに 「ライブラリ名 in が割り当てられた」というメッセージが表示されます。このメッセージは DB2 エンジン経由で DB と接続できたことを表します。

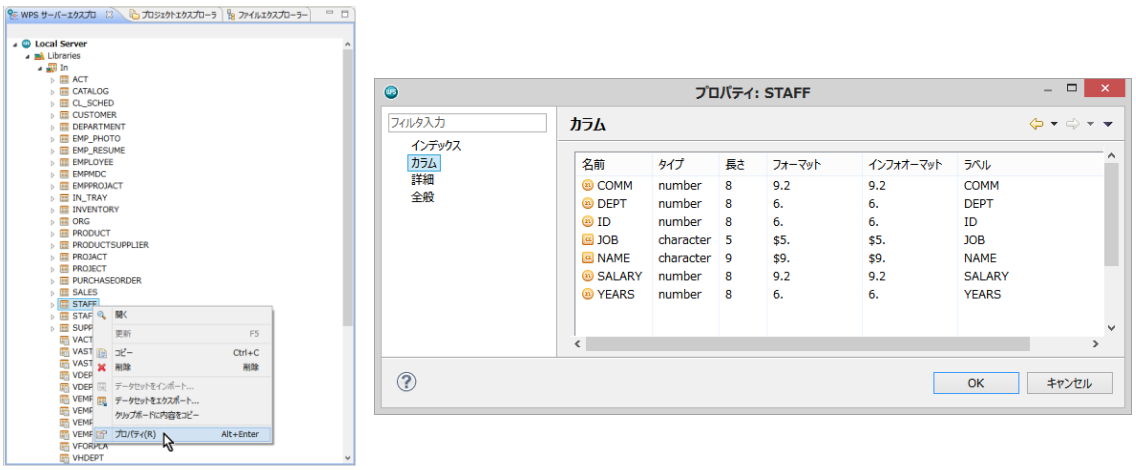
```
29        libname in DB2 datasrc=sample;
NOTE: Library in assigned as follows:
      Engine:        DB2
      Physical Name: sample
```

実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に sample データベースのテーブル名およびビュー名が含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、付いていないのはビュー名です。

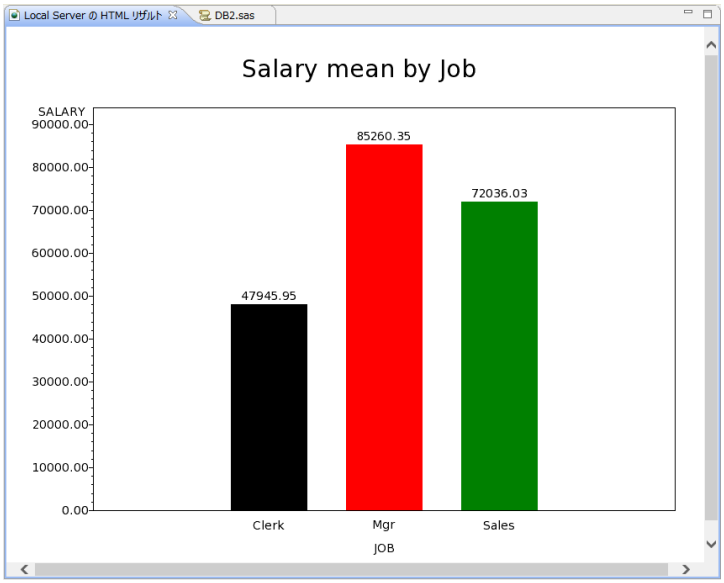
これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。



※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、DB2 エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、staff テーブルを入力として職種（JOB）別の平均給料（SALARY）の棒グラフを描いています。

```
goptions cback=white;
title "Salary mean by Job";
proc gchart data=in.staff;
  vbar job/sumvar=salary type=mean patternid=midpoint width=10 space=5 mean;
run;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、staff テーブルから職種（JOB）が"Mgr"か"Sales"のいずれかに該当するレコードを選択した変数 Name, Job, Salary を含む WPS データセット WORK.Mgr_or_Sales を DATA ステップで作成しています。

```
data Mgr_or_Sales;
  set in.staff(keep=Name Job Salary);
  where Job in ('Mgr', 'Sales');
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ
③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステー

トメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名やスキーマ名）を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで sample テーブルから職種が "Mgr" か "Sales" に該当するレコードを選択した WPS データセット WORK.Mgr_or_Sales を DB2 の Mgr_or_Sales テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=Mgr_or_Sales dbms=DB2;
  datasrc=sample;
  table=Mgr_or_Sales;
  load;
run;quit;
```

※ 注意： DB にアクセスするため DB2 エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は sample データベースの sales テーブルを読んで、SALES_PERSON="LEE"に該当するレコードのみ抽出した全カラムを含むテーブル LEE を作成しています。

```
libname sales DB2 datasrc=sample;
proc sql;
  create table sales.LEE as
  select *
  from sales.sales
  where sales_person = "LEE";
quit;
```

※ この例では、Sales ライブラリは DB2 の sample データベースを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の Sales ライブラリは WPS データライブラリではなく、sample DB を表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname sales DB2 datasrc=sample;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=sales;
  delete LEE;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data sales.LEE;
  set sales.sales;
  where sales_person = "LEE";
run;
```

2.2. ODBC ドライバー経由の DB テーブルの入出力

ODBC ドライバーは、Windows 環境や Linux 環境から各種 DB とのインタフェースを共通方式で行うためのソフトウェアです。ほとんどの DB の ODBC ドライバーが DB ベンダやサードパーティから提供されており、ODBC ドライバーの設定を行えば、DB ごとの細かいアクセス条件等の設定を気にせずに、共通の構文で DB アクセスが実現できるという利点があります。

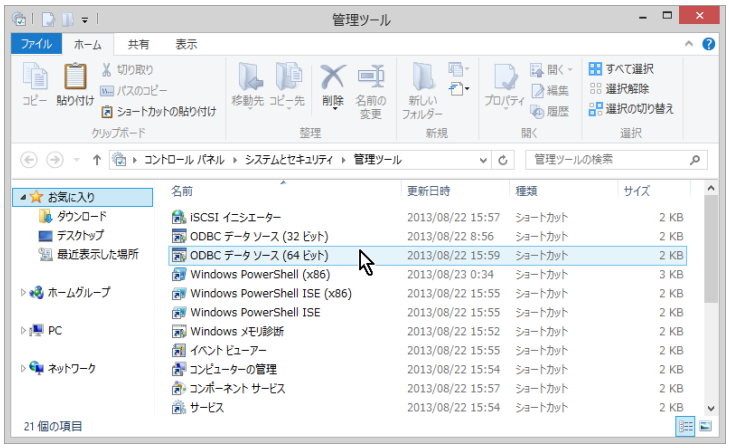
［ODBC ドライバーのインストール］

DB2 の ODBC ドライバーは DB2 インストール時に自動でインストールされます。もしも、次の［PC 側の ODBC ドライバーの設定］で DB2 の ODBC ドライバーが見当たらない場合は、インストール環境を確認してください。

［PC 側の ODBC ドライバーの設定］

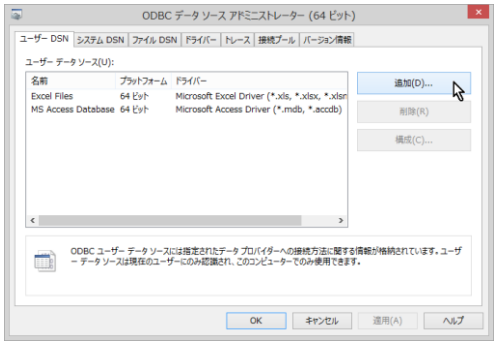
WPS が導入されている PC 側では、以下のような手順で、DB2 の sample データベースを ODBC データソースとして登録しておきます。

コントロールパネル → システムとセキュリティ → 管理ツールで ODBC データソースを選択します。



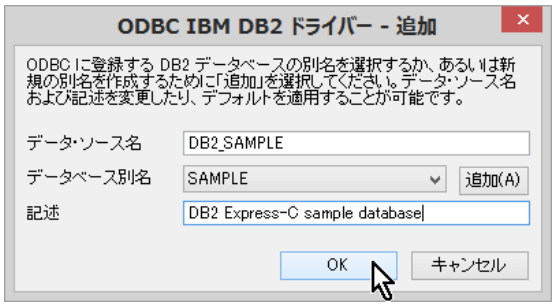
（左図）「追加」をクリックします。

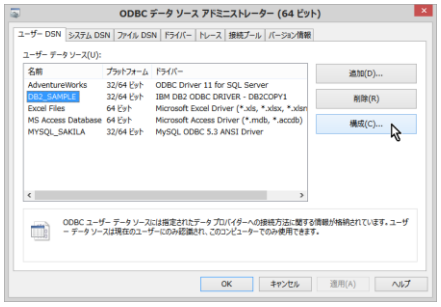
（右図）リストにある DB2 用 ODBC ドライバーを選択し、「完了」を押します。



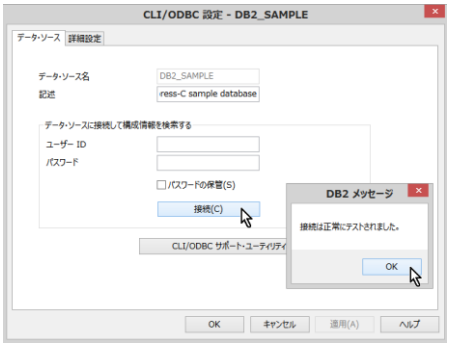
（左図）新規データ ソースの名前と説明にはユーザーが自由に入力できます。ここでは、名前に「DB2_SAMPLE」、説明には「DB2 Express-C sample database」と入力し「OK」を押します。

（右図）ODBC データ ソース リストに DB2 用 ODBC ドライバー「DB2_SAMPLE」が追加されたことを確認し、これを選択してから「構成」を押します。





「接続」をクリックし接続テストを行います。



正常に接続できたなら「OK」を押し ODBC ドライバーの設定を終了します。

[SQL 言語を使用した DB 読み取り]

SQL 言語を用いて WPS から ODBC ドライバー経由で DB2 の DB アクセスを行う場合は、以下のように、エンジン名として ODBC を指定し、CONNECT 文のカッコ内に datasrc=ODBC データソース名 などの接続オプションや dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを必要に応じて指定します。

下記は、ODBC ドライバーを用いて、sample データベースの staff テーブルから、DEPT=10 または 20 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.staff を作成する例となっています。

```
proc sql;
  connect to ODBC (datasrc=DB2_SAMPLE);
  create table staff as
  select *
  from connection to ODBC
  (
    select *
    from staff
    where DEPT in (10, 15);
  );
  disconnect from ODBC;
quit;
```

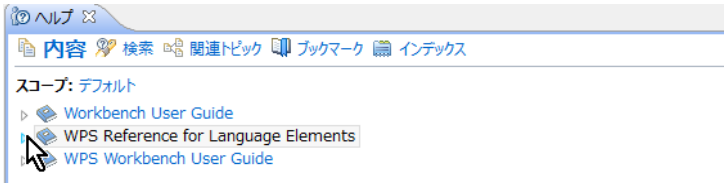
(WPS ワークベンチの実行ログ)

```
314       proc sql;
315         connect to ODBC (datasrc=DB2_SAMPLE);
NOTE: Successfully connected to database ODBC as alias ODBC.
316         create table staff as
317         select *
318         from connection to ODBC
319         (
320           select *
321           from staff
322           where DEPT in (10, 15);
```

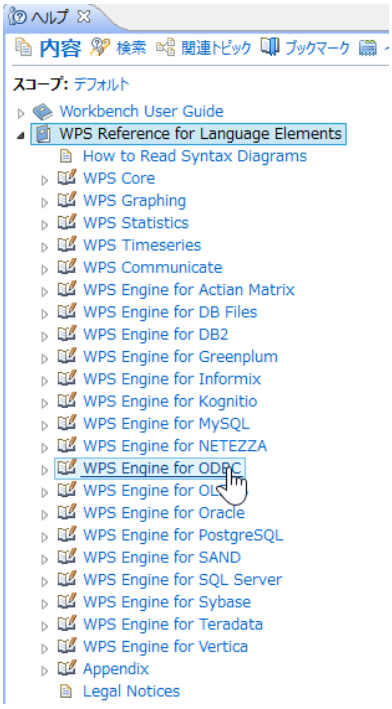
```
323         );
NOTE: Data set "WORK.staff" has 8 observation(s) and 7 variable(s)
324         disconnect from ODBC;
NOTE: Successfully disconnected from database ODBC.
325       quit;
NOTE: Procedure sql step took :
      real time : 0.359
      cpu time  : 0.015
```

※ ODBC ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の () 内のオプションに、その他の指定が必要となる場合があります。以下のようにヘルプビューで指定可能なオプションが確認できます。

ヘルプビュー[5]の左上にある「内容」をクリックします。
ヘルプ全体の目次が表示されます。WPS Reference for Language Elements をクリックします。

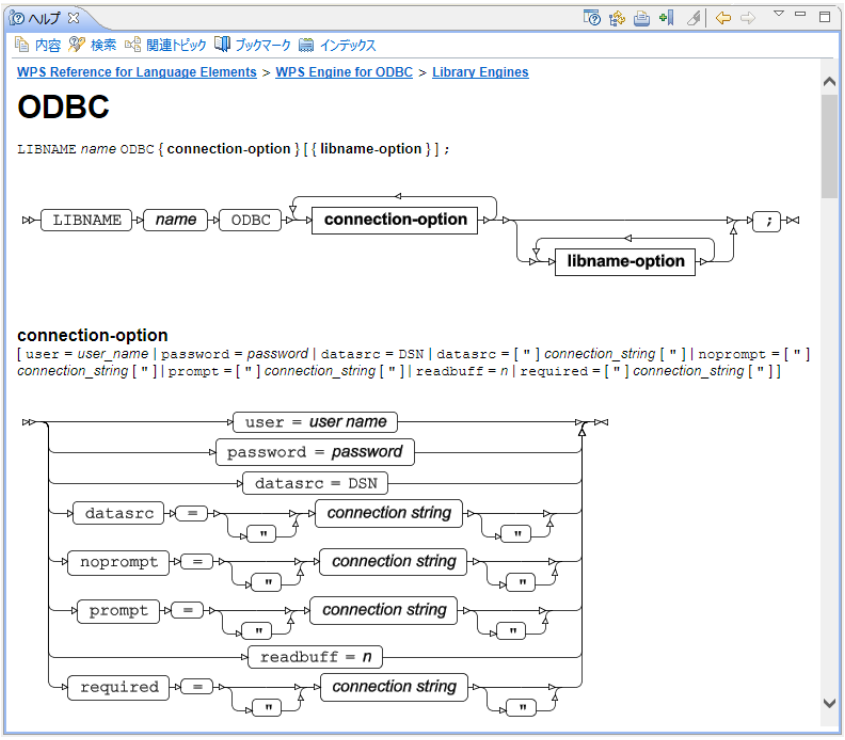


WPS レファレンスを展開し、WPS Engine for ODBC を選択します。



[5] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。

その後、展開を何度か行うと接続オプション（connection option） が表示されます。



これらのオプション（user= password= datasrc= noprompt= prompt= readbuff= required=）は ODBC ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプション指定としても、また、次に説明する libname ステートメントによるデータベースアクセスの接続オプションとしても用いることができ、必要に応じて指定します。

［libname ステートメントによる DB 入力］

WPS のエディタから以下のプログラムを実行します。

```
libname in ODBC datasrc=DB2_SAMPLE;
```

この指定は、WPS がデータソース DB2_SAMPLE に ODBC ドライバー経由で接続し、ライブラリ名 in で参照できるようにする指定です。（入出力両方可能）
ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

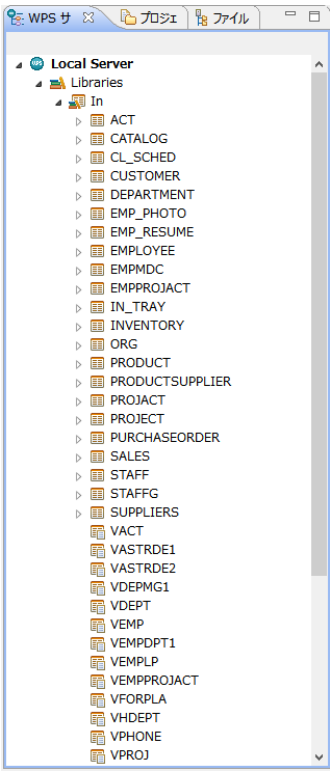
さらに、libname ステートメントには dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを追加指定できます。指定可能な libname オプションは

DB2 エンジンの libname オプションとほぼ同じですが、詳細は前出の ODBC エンジンのヘルプ画面から確認してください。(Connection-option の下に libname-option が記載されています。)

実行すると、ログに「ライブラリ名 in が割り当てられた」というメッセージが表示されます。このメッセージは ODBC 経由で DB と接続できたことを表します。

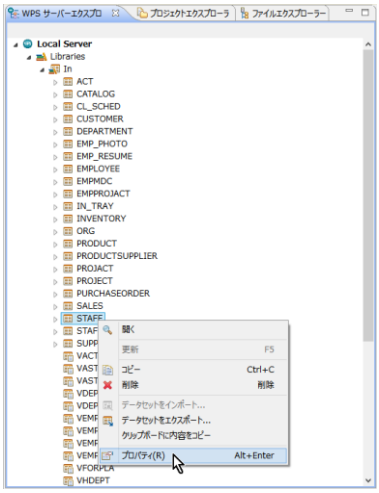
```
345       libname in ODBC datasrc=DB2_sample;
NOTE: Library in assigned as follows:
      Engine:        ODBC
      Physical Name: DB2_sample
```

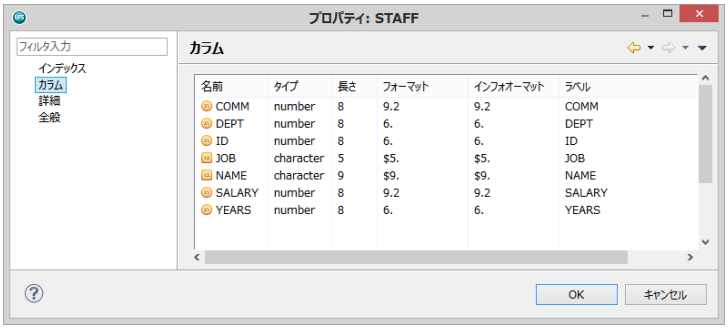
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に sample データベースのテーブル名およびビュー名が含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。)、付いていないのはビュー名です。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。

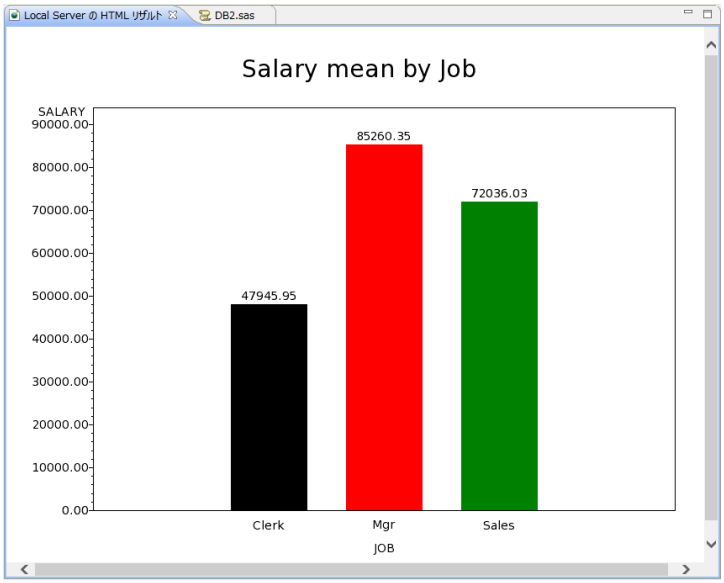




※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、ODBC エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、staff テーブルを入力として職種（JOB）別の平均給料（SALARY）の棒グラフを描いています。

```
goptions cback=white;
title "Salary mean by Job";
proc gchart data=in.staff;
  vbar job/sumvar=salary type=mean patternid=midpoint width=10 space=5 mean;
run;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、staff テーブルから職種（JOB）が"Mgr"か"Sales"のいずれかに該当するレコードを選択した変数 Name, Job, Salary を含む WPS データセット WORK.Mgr_or_Sales を DATA ステップで作成しています。

```
data Mgr_or_Sales;
  set in.staff(keep=Name Job Salary);
  where Job in ('Mgr', 'Sales');
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ
③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステー

トメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名やスキーマ名）を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで sample テーブルから職種が "Mgr" か "Sales" に該当するレコードを選択した WPS データセット WORK.Mgr_or_Sales を DB2 の Mgr_or_Sales テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=Mgr_or_Sales dbms=ODBC;
  datasrc=DB2_SAMPLE;
  table=Mgr_or_Sales;
  load;
run;quit;
```

※ 注意：DB にアクセスするための権限や ODBC エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は sample データベースの sales テーブルを読んで、SALES_PERSON="LEE"に該当するレコードのみ抽出した全カラムを含むテーブル LEE を作成しています。

```
libname sales ODBC datasrc=DB2_SAMPLE;
proc sql;
  create table sales.LEE as
  select *
  from sales.sales
  where sales_person = "LEE";
quit;
```

※ この例では、Sales ライブラリは ODBC の DB2_SAMPLE データソースを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の Sales ライブラリは WPS データライブラリではなく、sample DB を表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

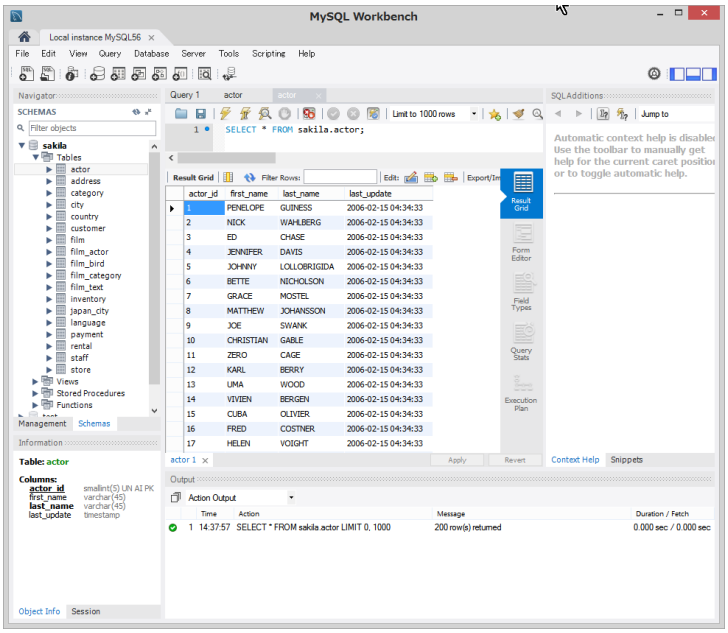
以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname sales ODBC datasrc=DB2_SAMPLE;
/* 同じ名前のテーブルを一旦削除 */
proc sql;
  drop table sales.LEE;
quit;
/* DATAステップでDBテーブルを読み書き */
data sales.LEE;
  set sales.sales;
  where sales_person = "LEE";
run;
```

3. MySQL とのインタフェース

［sakila サンプル DB］

MySQL Community Server 5.6 に付随する sakila データベースを例として用います。



3.1. ネイティブドライバー経由の DB テーブルの入出力

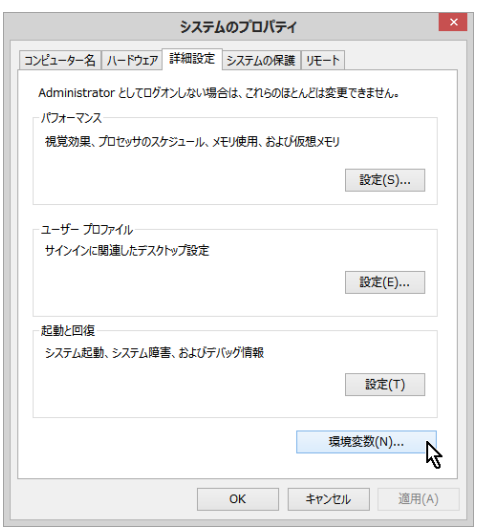
まず、コマンドラインインタフェース（CLI ドライバー、ネイティブドライバー）を用いて

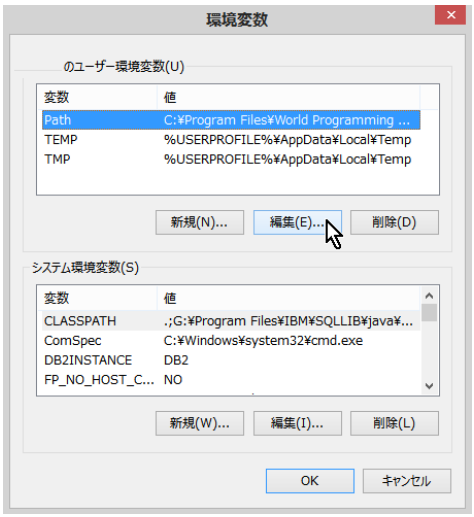
WPS から DB テーブルの入出力を行ってみましょう。

［ネイティブドライバのインストールと Path の定義］

WPS は MySQL との接続を行うネイティブドライバーとして、Connector/C(libmysqlclient) ライブラリを用います。このファイル（libmysqlclient.dll）はインストールすると、MySQL インストールディレクトリの下の lib ディレクトリ内に保管されます。これをアクセスできるように、以下のように、環境変数 Path に定義しておく必要があります。

（左図）コントロールパネル → システム → システムの詳細設定を開きます。
「環境変数」をクリック → 「Path」をクリックします。
（右図）ユーザー環境変数 Path を選択し、「編集」をクリックします。なお、他のユーザーでも有効な Path を設定したい場合はシステム環境変数の Path を選択し編集してください。





変数値の最後に、以下のように、MySQL データベースの lib ディレクトリを追記し、「OK」を押します。（実際に DB をインストールしたディレクトリ名を入力してください）

```
;C:¥Program Files¥MySQL¥MySQL Server 5.6¥lib¥
```

WPS ワークベンチを起動している状態の場合は、Path の設定後、WPS ワークベンチを再起動[6]してください。

ネイティブドライバーは、一般に ODBC ドライバーより高速に DB アクセスができる点でメリットがあると言われています。
WPS の MYSQL エンジンはネイティブドライバーを経由して MySQL データベースとアクセスを行う仕組みを提供します。

［SQL 言語を使用した DB 読み取り］

DB へのアクセスは SQL 言語を用いて行うことが多く、WPS でも SQL プロシジャを用いて SQL 言語による DB アクセスが可能です。

SQL 言語を用いて WPS からネイティブドライバー経由で MySQL の DB アクセスを行う場合は、以下のように、エンジン名として MYSQL を指定し、CONNECT 文のカッコ内に server=サーバー名、database=DB 名、user=ユーザー名、password=パスワード などの接続オプションや、後述する dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを指定します。

以下は sakila データベースの city テーブルから、country_id=50 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.japan_city を作成しています。

```
proc sql;
  connect to MYSQL (server=localhost database=sakila user=XXX
password=password);
  create table japan_city as
  select *
  from connection to MYSQL
  (
    select *
    from city
    where country_id=50
  );
  disconnect from MYSQL;
quit;
```

※ 2 行目の user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。
3 行目の create table 文は WPS データセット WORK.japan_city を作成する指定です。

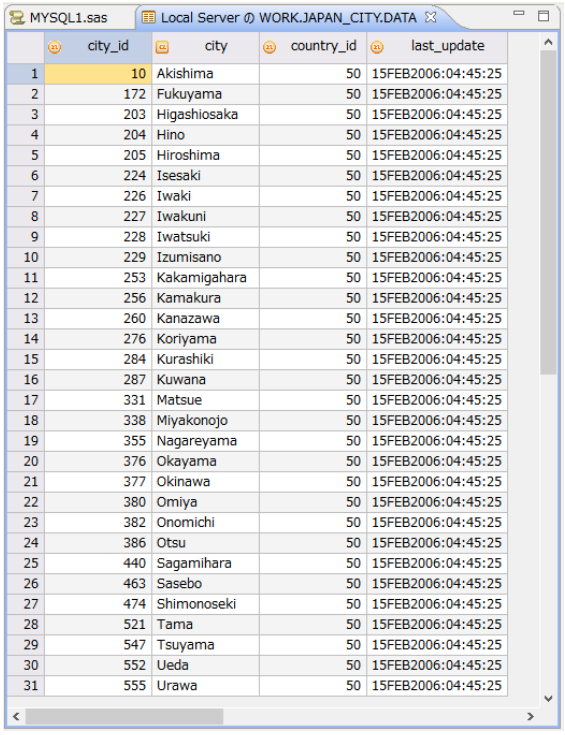
[6] メニューバーの「ファイル(F)」 → 「再起動」をクリックします。

8 行目の from 節で指定した city は sakila データベースの city テーブルを参照しています。

(WPS ワークベンチの実行ログ)

```
8         proc sql;
9           connect to MYSQL (server=localhost database=sakila user=XXX password=XXXX);
NOTE: Successfully connected to database MYSQL as alias MYSQL.
10          create table japan_city as
11          select *
12          from connection to MYSQL
13          (
14            select *
15            from city
16            where country_id=50
17          );
NOTE: Data set "WORK.japan_city" has 31 observation(s) and 4 variable(s)
18          disconnect from MYSQL;
NOTE: Successfully disconnected from database MYSQL.
19        quit;
NOTE: Procedure sql step took :
      real time : 0.015
      cpu time  : 0.000
```

作成された WPS データセット WORK.staff データセットの内容を確認します。

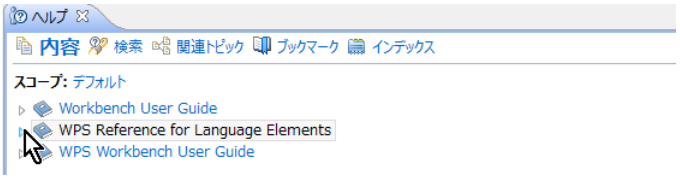


MYSQL1.sas | Local Server の WORK.JAPAN_CITY.DATA

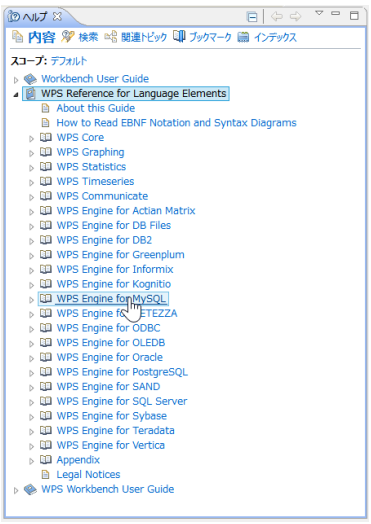
	city_id	city	country_id	last_update
1	10	Akishima	50	15FEB2006:04:45:25
2	172	Fukuyama	50	15FEB2006:04:45:25
3	203	Higashiosaka	50	15FEB2006:04:45:25
4	204	Hino	50	15FEB2006:04:45:25
5	205	Hiroshima	50	15FEB2006:04:45:25
6	224	Isesaki	50	15FEB2006:04:45:25
7	226	Iwaki	50	15FEB2006:04:45:25
8	227	Iwakuni	50	15FEB2006:04:45:25
9	228	Iwatsuki	50	15FEB2006:04:45:25
10	229	Izumisano	50	15FEB2006:04:45:25
11	253	Kakamigahara	50	15FEB2006:04:45:25
12	256	Kamakura	50	15FEB2006:04:45:25
13	260	Kanazawa	50	15FEB2006:04:45:25
14	276	Koriyama	50	15FEB2006:04:45:25
15	284	Kurashiki	50	15FEB2006:04:45:25
16	287	Kuwana	50	15FEB2006:04:45:25
17	331	Matsue	50	15FEB2006:04:45:25
18	338	Miyakonojo	50	15FEB2006:04:45:25
19	355	Nagareyama	50	15FEB2006:04:45:25
20	376	Okayama	50	15FEB2006:04:45:25
21	377	Okinawa	50	15FEB2006:04:45:25
22	380	Omiya	50	15FEB2006:04:45:25
23	382	Onomichi	50	15FEB2006:04:45:25
24	386	Otsu	50	15FEB2006:04:45:25
25	440	Sagamihara	50	15FEB2006:04:45:25
26	463	Sasebo	50	15FEB2006:04:45:25
27	474	Shimonoseki	50	15FEB2006:04:45:25
28	521	Tama	50	15FEB2006:04:45:25
29	547	Tsuyama	50	15FEB2006:04:45:25
30	552	Ueda	50	15FEB2006:04:45:25
31	555	Urawa	50	15FEB2006:04:45:25

※ 注意：MYSQL ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。指定可能なオプションは、以下のように、ヘルプビューで確認できます。

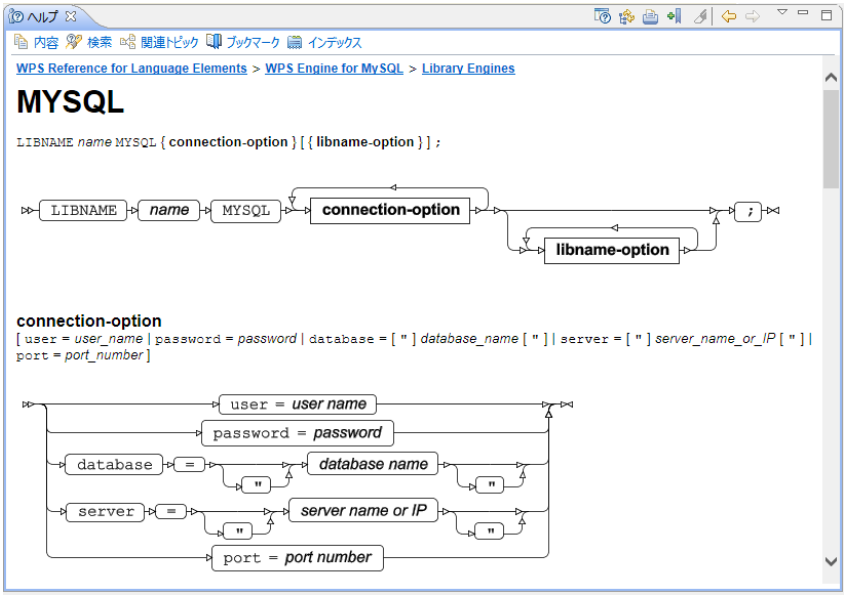
まず、ヘルプビュー[7]の左上にある「内容」をクリックします。ヘルプ全体の目次が表示されますので、WPS Reference for Language Elements をクリックします。



WPS レファレンスを展開し、WPS Engine for MySQL を選択します。



Library Engines → MySQL を選択すると、データベース接続オプション（connection option）が表示されます。



[7] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。

これらの接続オプション（user= password= database= server= port=）は MYSQL ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプション指定としても、また、次に節で説明する libname ステートメントによる DB 接続オプションとしても用いることができ、必要に応じて指定します。

［libname ステートメントによる DB 入力］

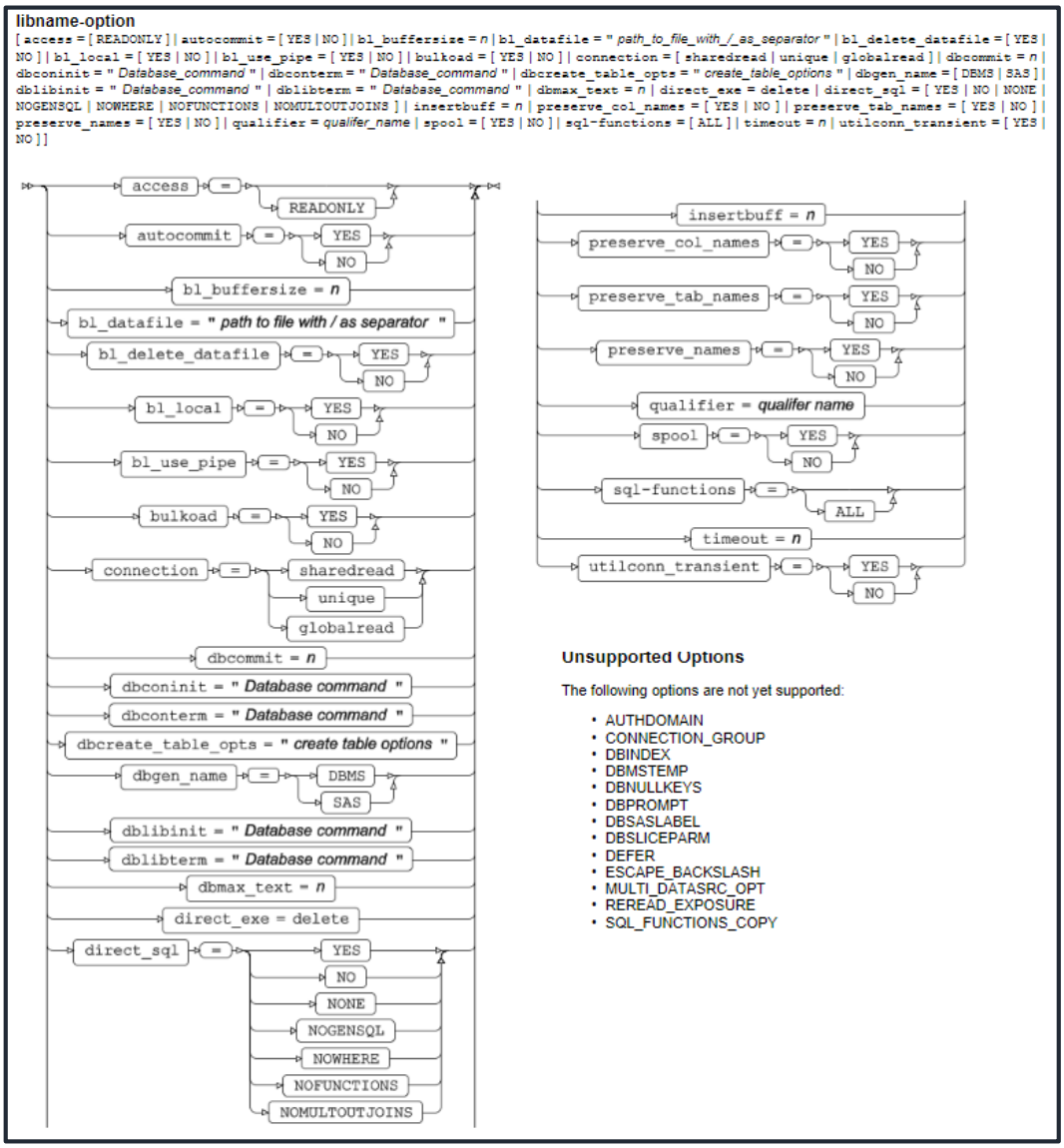
WPS のエディタから以下のプログラムを入力し、実行します。

```
libname in MYSQL database=sakila user=XXX password=password;
```

※ user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

この指定は、WPS がデータベース sakila に MYSQL ネイティブドライバー経由で接続し、sakila データベースのテーブルとビューをライブラリ名 in で参照できるようにする指定です。（入出力両方可能） ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

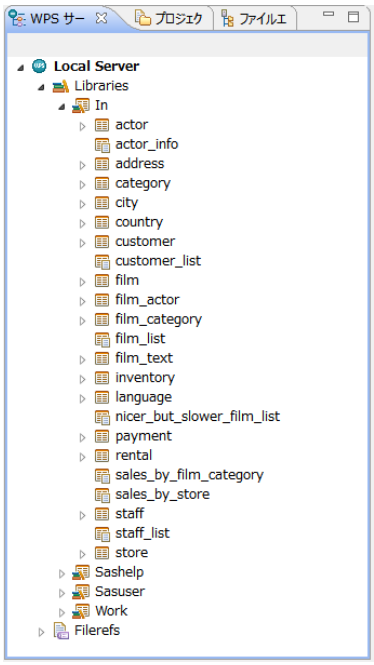
さらに、libname ステートメントには dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを追加指定できます。以下のような libname オプションが指定可能です。（前出の MYSQL エンジンのヘルプ画面から確認できます。）



さて、実行すると、ログに 「ライブラリ名 in が割り当てられた」というメッセージが表示されます。このメッセージは MYSQL エンジン経由で DB と接続できたことを表します。

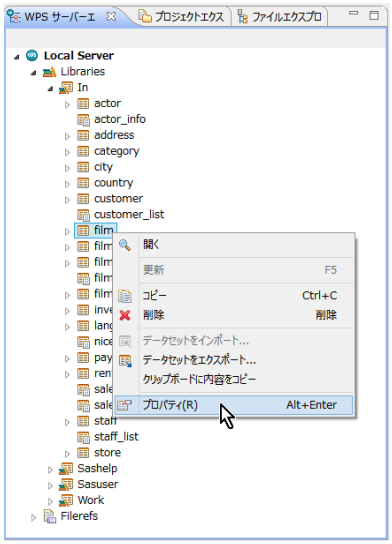
```
29         libname in MYSQL database=sakila user=XXX password=XXXX;
NOTE: Library in assigned as follows:
      Engine:        MYSQL
      Physical Name:
```

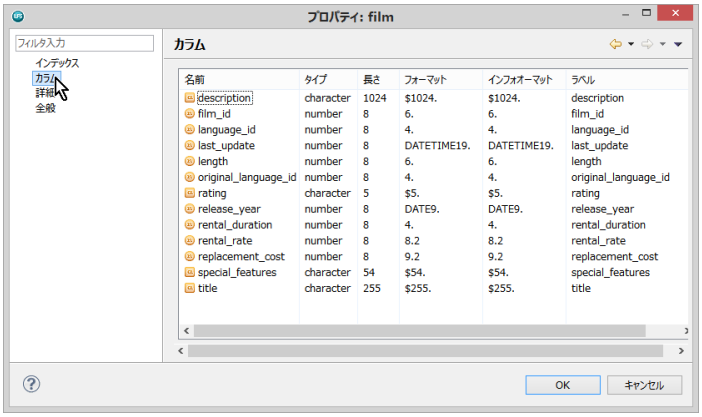
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に sakila データベースのテーブル名およびビュー名が含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、付いていないのはビュー名です。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。



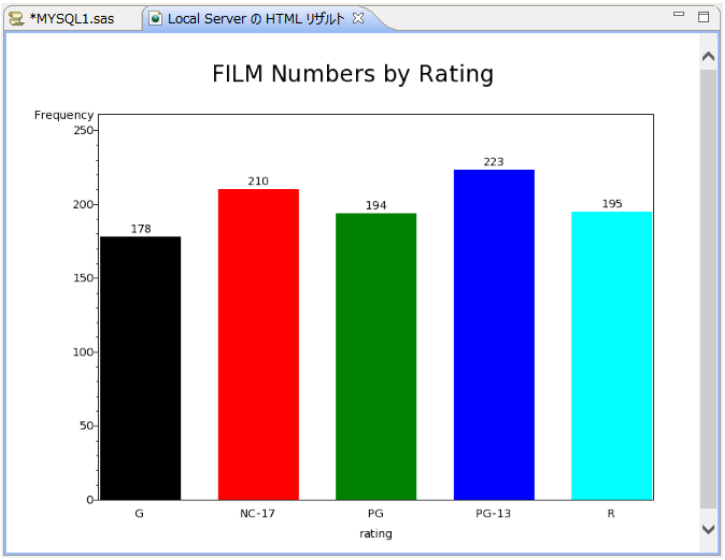


※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS

セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、MYSQL エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、film テーブルを入力として映画鑑賞年齢制限（rating）別の映画の本数の棒グラフを描いています。

```
goptions cback=white;
title "FILM Numbers by Rating";
proc gchart data=in.film;
  vbar rating/patternid=midpoint freq width=20 space=5;
run;quit;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、film テーブルから題名（TITLE）に"BIRD"が含まれるレコードを選択した変数 File_id, title, rental_rate, rating を含む WPS データセット WORK.film_BIRD を DATA ステップで作成しています。

```
data film_BIRD;
```

```
  set in.film(keep=film_id title rental_rate rating);
  where title contains "BIRD";
run;
```

[WPS データセットの DB への出力方法]

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ
③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステートメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名など）を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで film テーブルから題名に"BIRD"を含むレコードを選択した WPS データセット WORK.film_BIRD を MySQL の sakila データベースの film_BIRD テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=film_BIRD dbms=MYSQL;
  server=localhost;
  user=XXX;
  password=password;
  database=sakila;
  table=film_BIRD;
  load;
run;quit;
```

※ 注意： DB にアクセスするため MYSQL エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。また、user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと

同じように扱えるようになります。
以下の例は sakila データベースの city テーブルを読んで、country_id=50 に該当するレコードのみ抽出した全カラムを含むテーブル japan_city を作成しています。

```
libname in MYSQL database=sakila user=XXX password=password;
proc sql;
  create table in.japan_city as
  select *
  from in.city
  where country_id=50;
quit;
```

※ この例では、in ライブラリは MySQL の sakila データベースを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の in ライブラリは WPS データライブラリではなく、sakila DB を表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname in MYSQL database=sakila user=XXX password=password;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=in;
  delete japan_city;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data in.japan_city;
  set in.city;
  where country_id=50;
run;
```

3.2. ODBC ドライバー経由の DB テーブルの入出力

ODBC ドライバーは、Windows 環境や Linux 環境から各種 DB とのインタフェースを共通方式で行うためのソフトウェアです。ほとんどの DB の ODBC ドライバーが DB ベンダやサードパーティから提供されており、ODBC ドライバーの設定を行えば、DB ごとの細かいアクセス条件等の設定を気にせずに、共通の構文で DB アクセスが実現できるという利点があります。

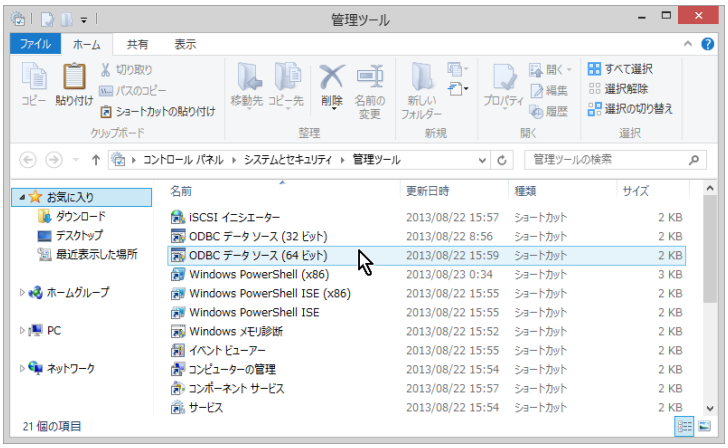
［ODBC ドライバのインストール］

MySQL の ODBC ドライバーは Connector/ODBC を MySQL のバージョンに合わせてインストールします。 もしも、次の［PC 側の ODBC ドライバーの設定］で MySQL の ODBC ドライバーが見当たらない場合は、インストール環境を確認してください。

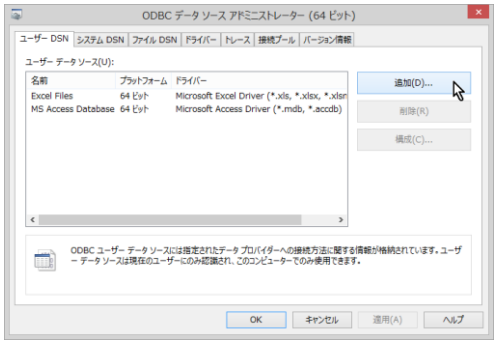
［PC 側の ODBC ドライバーの設定］

WPS が導入されている PC 側では、以下のような手順で、MYSQL の sakila データベースを ODBC データソースとして登録しておきます。

コントロールパネル → システムとセキュリティ → 管理ツールで ODBC データソースを選択します。

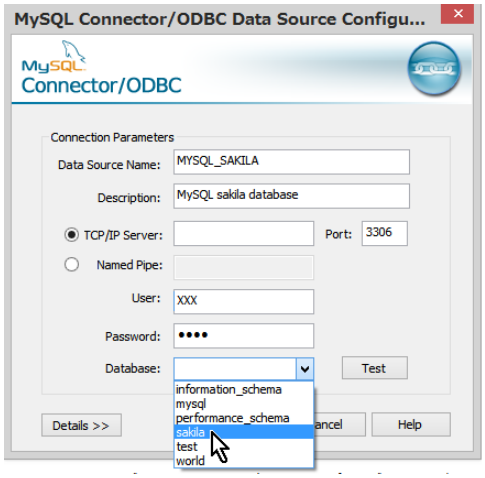


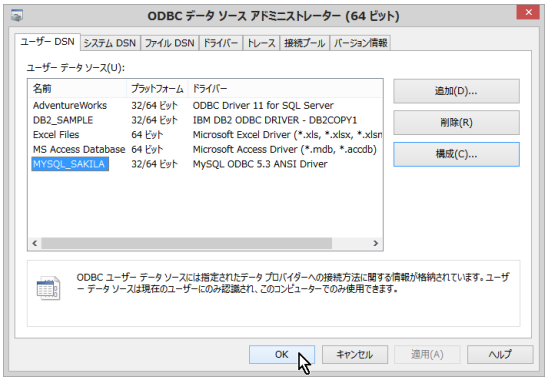
（左図）「追加」をクリックします。
（右図）リストにある MySQL 用 ODBC ドライバーを選択し、「完了」を押します。



（左図）新規データソースの名前と説明にはユーザーが自由に入力できます。ここでは、名

前に「MYSQL_SAKILA」、説明には「MySQL sakila database」と入力します。ユーザー名とパスワードを入力すると、データベース欄のドロップリストが利用できますので、「sakila」を選択します。次に「Test」を押して接続テストを行います。接続テストが成功することを確認すると設定終了です。
（右図）「OK」を押して ODBC データソースリストに MySQL 用 ODBC ドライバー「MYSQL_SAKILA」が追加されたことを確認します。





[SQL 言語を使用した DB 読み取り]

SQL 言語を用いて WPS から ODBC ドライバー経由で MySQL の DB アクセスを行う場合は、以下のように、エンジン名として ODBC を指定し、CONNECT 文のカッコ内に datasrc=ODBC データソース名などの接続オプションや、dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを指定します。

下記は、ODBC ドライバーを用いて、sakila データベースの city テーブルから、country_id=50 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.japan_city を作成する例となっています。

```
proc sql;
  connect to ODBC (datasrc=MYSQL_SAKILA);
  create table japan_city as
  select *
  from connection to ODBC
  (
    select *
    from city
    where country_id=50
  );
  disconnect from ODBC;
```

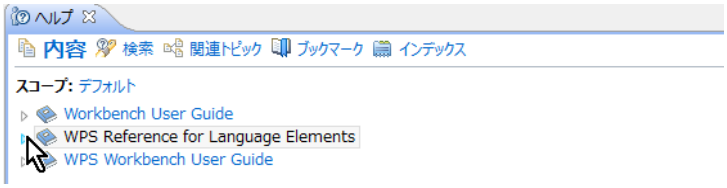
```
quit;
```

(WPS ワークベンチの実行ログ)

```
8         proc sql;
9           connect to ODBC (datasrc=MYSQL_SAKILA);
NOTE: Successfully connected to database ODBC as alias ODBC.
10          create table japan_city as
11          select *
12          from connection to ODBC
13          (
14            select *
15            from city
16            where country_id=50
17          );
NOTE: Data set "WORK.japan_city" has 31 observation(s) and 4 variable(s)
18          disconnect from ODBC;
NOTE: Successfully disconnected from database ODBC.
19        quit;
NOTE: Procedure sql step took :
      real time : 0.031
      cpu time  : 0.015
```

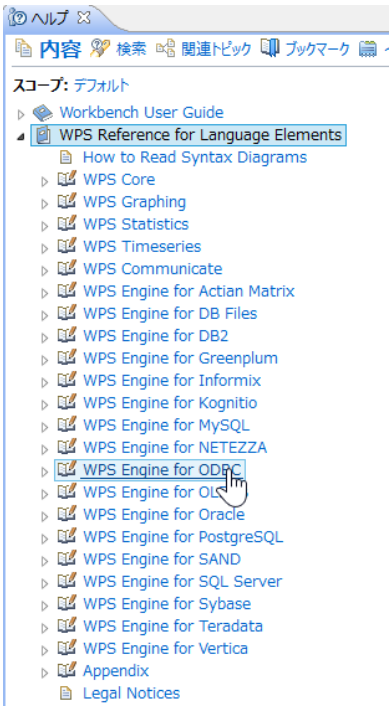
※ ODBC ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。以下のようにヘルプビューで指定可能なオプションが確認できます。

ヘルプビュー[8]の左上にある「内容」をクリックします。
ヘルプ全体の目次が表示されます。WPS Reference for Language Elements をクリックします。

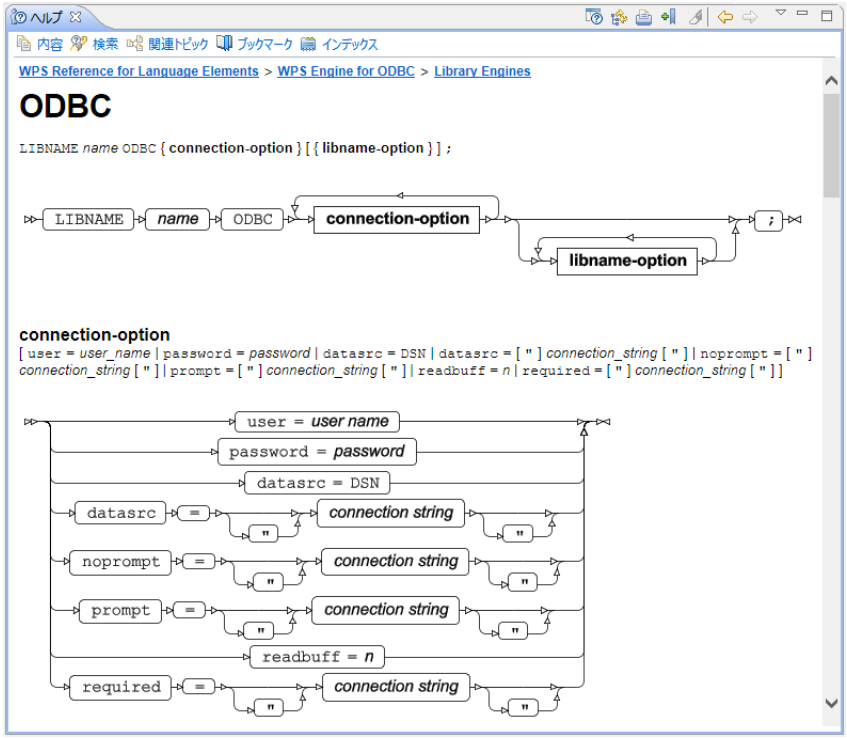


WPS レファレンスを展開し、WPS Engine for ODBC を選択します。

[8] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。



その後、展開を何度か行うと接続オプション（connection option） が表示されます。



これらのオプション（user= password= datasrc= noprompt= prompt= readbuff= required=）は ODBC ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプシ

ョン指定としても、また、次に説明する libname ステートメントによるデータベースアクセスの接続オプションとしても用いることができ、必要に応じて指定します。

[libname ステートメントによる DB 入力]

WPS のエディタから以下のプログラムを実行します。

```
libname in ODBC datasrc=MYSQL_SAKILA;
```

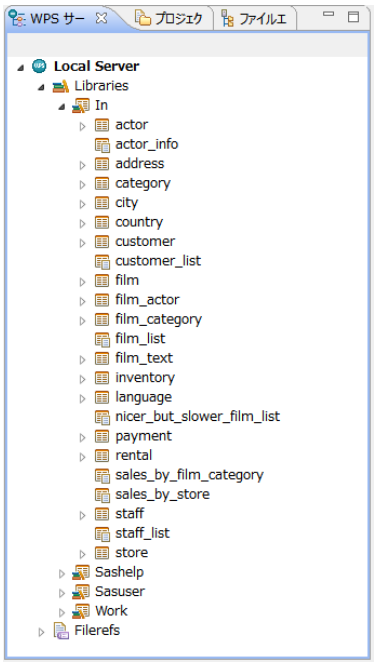
この指定は、WPS がデータソース MYSQL_SAKILA に ODBC ドライバー経由で接続し、ライブラリ名 in で参照できるようにする指定です。（入出力両方可能）
ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

さらに、libname ステートメントには schema=スキーマ名、dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを追加指定できます。指定可能な libname オプションは MYSQL エンジンの libname オプションとほぼ同じですが、詳細は前出の ODBC エンジンのヘルプ画面から確認してください。（Connection-option の下に libname-option が記載されています。）

実行すると、ログに「ライブラリ名 in が割り当てられた」というメッセージが表示されます。このメッセージは ODBC 経由で DB と接続できたことを表します。

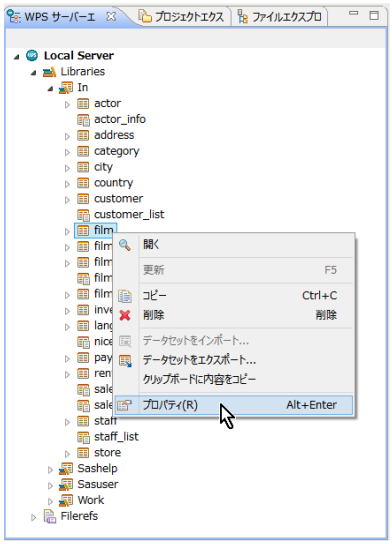
```
345      libname in ODBC datasrc=MYSQL_SAKILA;
NOTE: Library in assigned as follows:
      Engine:       ODBC
      Physical Name: MYSQL_SAKILA
```

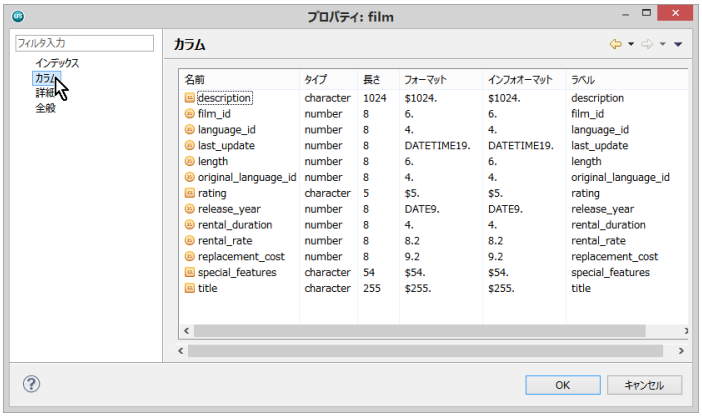
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に sakila データベースのテーブル名およびビュー名が含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、付いていないのはビュー名です。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。



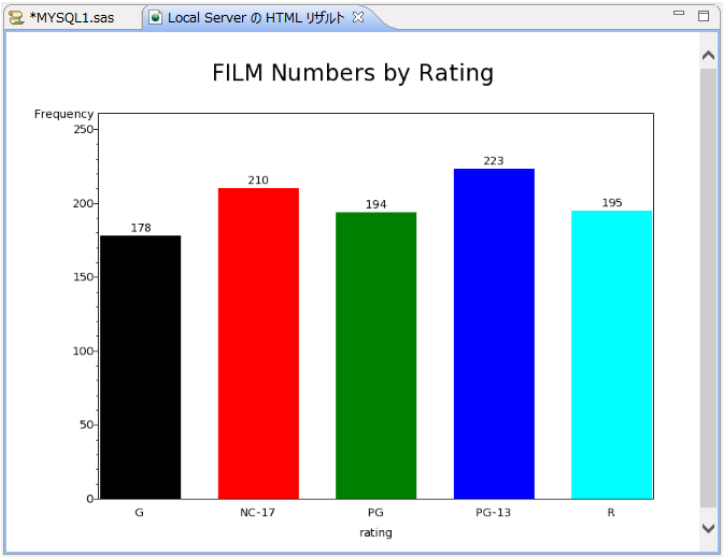


※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できる

だけ避けてください。

さて、ODBC エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、film テーブルを入力として映画鑑賞年齢制限（rating）別の映画の本数の棒グラフを描いています。

```
goptions cback=white;
title "FILM Numbers by Rating";
proc gchart data=in.film;
  vbar rating/patternid=midpoint freq width=20 space=5;
run;quit;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、film テーブルから題名（TITLE）に"BIRD"が含まれるレコードを選択した変数 File_id, title, rental_rate, rating を含む WPS データセット WORK.film_BIRD を DATA ステップで作成しています。

```
data film_BIRD;
  set in.film(keep=film_id title rental_rate rating);
```

```
  where title contains "BIRD";
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ
③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステートメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名やスキーマ名）を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで sakia テーブルから題名（TITLE）に"BIRD"が含まれるレコードを選択した変数 File_id, title, rental_rate, rating を含む WPS データセット WORK.film_BIRD を MySQL の sakila データベースの film_BIRD テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=film_BIRD dbms=ODBC;
  datasrc=MYSQL_SAKIA;
  table=film_BIRD;
  load;
run;quit;
```

※ 注意：DB にアクセスするための権限や ODBC エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は sakila データベースの city テーブルを読んで、country_id=50 に該当するレコードのみ抽出した全カラムを含むテーブル japan_city を作成しています。

```
libname in ODBC datasrc=MYSQL_SAKILA;
proc sql;
  create table in.japan_city as
  select *
  from in.city
  where country_id = 50;
quit;
```

※ この例では、in ライブラリは ODBC 経由で sakila データベースを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の in ライブラリは WPS データライブラリではなく、sakila DB を表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

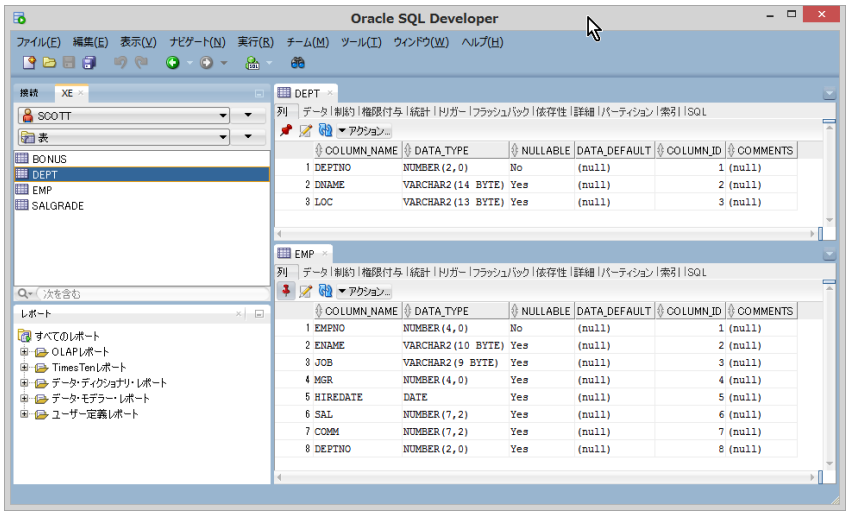
以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname in ODBC datasrc=MYSQL_SAKILA;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=in;
  delete japan_city;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data in.japan_city;
  set in.city;
  where country_id=50;
run;
```

4. ORACLE とのインタフェース

［scott サンプルスキーマ］

$ORACLE_HOME/rdbms/admin/utlsampl.sql を sqlplus から実行して作成される scott スキーマ上の DEPT テーブルと EMP テーブルを例として用います。



4.1. ネイティブドライバー経由の DB テーブルの入出力

まず、コマンドラインインタフェース（CLI ドライバー、ネイティブドライバー）を用いて WPS から DB テーブルの入出力を行ってみましょう。
ネイティブドライバーは、DB と一緒にインストールされます。一般に ODBC ドライバーより高速に DB アクセスができる点でメリットがあると言われています。
WPS の ORACLE エンジンはネイティブドライバーを経由して ORACLE データベースとアクセスを行う仕組みを提供します。

[SQL 言語を使用した DB 読み取り]

DB へのアクセスは SQL 言語を用いて行うことが多く、WPS でも SQL プロシジャを用いて SQL 言語による DB アクセスが可能です。

SQL 言語を用いて WPS からネイティブドライバー経由で ORACLE の DB アクセスを行う場合は、以下のように、エンジン名として ORACLE を指定し、CONNECT 文のカッコ内に user=ユーザー名、password=パスワード、path=パス の接続オプションや schema=、dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを指定します。

以下は scott スキーマ上の EMP テーブルから、SAL>2000 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.EMP_SAL_OVER2000 を作成しています。

```
proc sql;
  connect to ORACLE (user=XXX password=password);
  create table EMP_SAL_OVER2000 as
  select *
  from connection to ORACLE
  (
    select *
    from scott.EMP
    where SAL>2000
   );
  disconnect from ORACLE;
quit;
```

※ 2 行目の user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

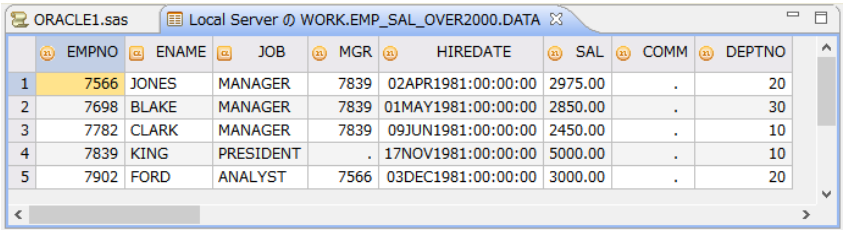
3 行目の create table 文は WPS データセット WORK.EMP_SAL_OVER2000 を作成する指定です。

8 行目の from 節で指定した scott.EMP は scott スキーマ上の EMP テーブルを参照しています。

(WPS ワークベンチの実行ログ)

```
8         proc sql;
9           connect to ORACLE (user=XXX password=XXXXXXXX);
NOTE: Successfully connected to database ORACLE as alias ORACLE.
10          create table EMP_SAL_OVER2000 as
11          select *
12          from connection to ORACLE
13          (
14            select *
15            from scott.EMP
16            where SAL>2000
17           );
NOTE: Data set "WORK.EMP_SAL_OVER2000" has 5 observation(s) and 8 variable(s)
18          disconnect from ORACLE;
NOTE: Successfully disconnected from database ORACLE.
19        quit;
NOTE: Procedure sql step took :
      real time : 0.078
      cpu time  : 0.031
```

作成された WPS データセット WORK. EMP_SAL_OVER2000 データセットの内容を確認します。

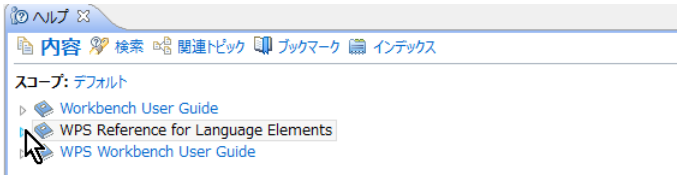


	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7566	JONES	MANAGER	7839	02APR1981:00:00:00	2975.00	.	20
2	7698	BLAKE	MANAGER	7839	01MAY1981:00:00:00	2850.00	.	30
3	7782	CLARK	MANAGER	7839	09JUN1981:00:00:00	2450.00	.	10
4	7839	KING	PRESIDENT	.	17NOV1981:00:00:00	5000.00	.	10
5	7902	FORD	ANALYST	7566	03DEC1981:00:00:00	3000.00	.	20

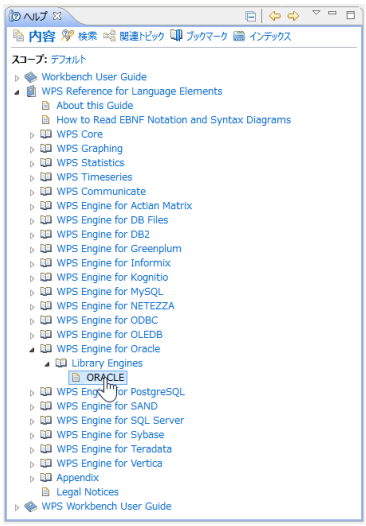
※ 注意：ORACLE ドライバーによる DB へのアクセス制限等の設定によっては、SQL プ

ロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。指定可能なオプションは、以下のように、ヘルプビューで確認できます。

まず、ヘルプビュー[9]の左上にある「内容」をクリックします。ヘルプ全体の目次が表示されますので、WPS Reference for Language Elements をクリックします。

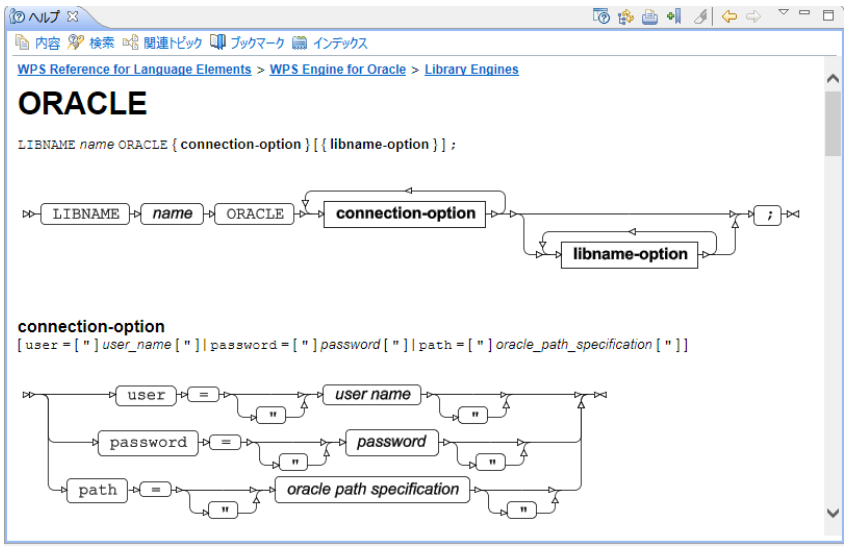


WPS レファレンスを展開し、WPS Engine for ORACLE を選択します。



Library Engines → ORACLE を選択すると、データベース接続オプション（connection option）が表示されます。

[9] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。



これらの接続オプション（user= password= path=）は ORACLE ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプション指定としても、また、次に節で説明する libname ステートメントによる DB 接続オプションとしても用いることができ、必要に応じて指定します。

［libname ステートメントによる DB 入力］

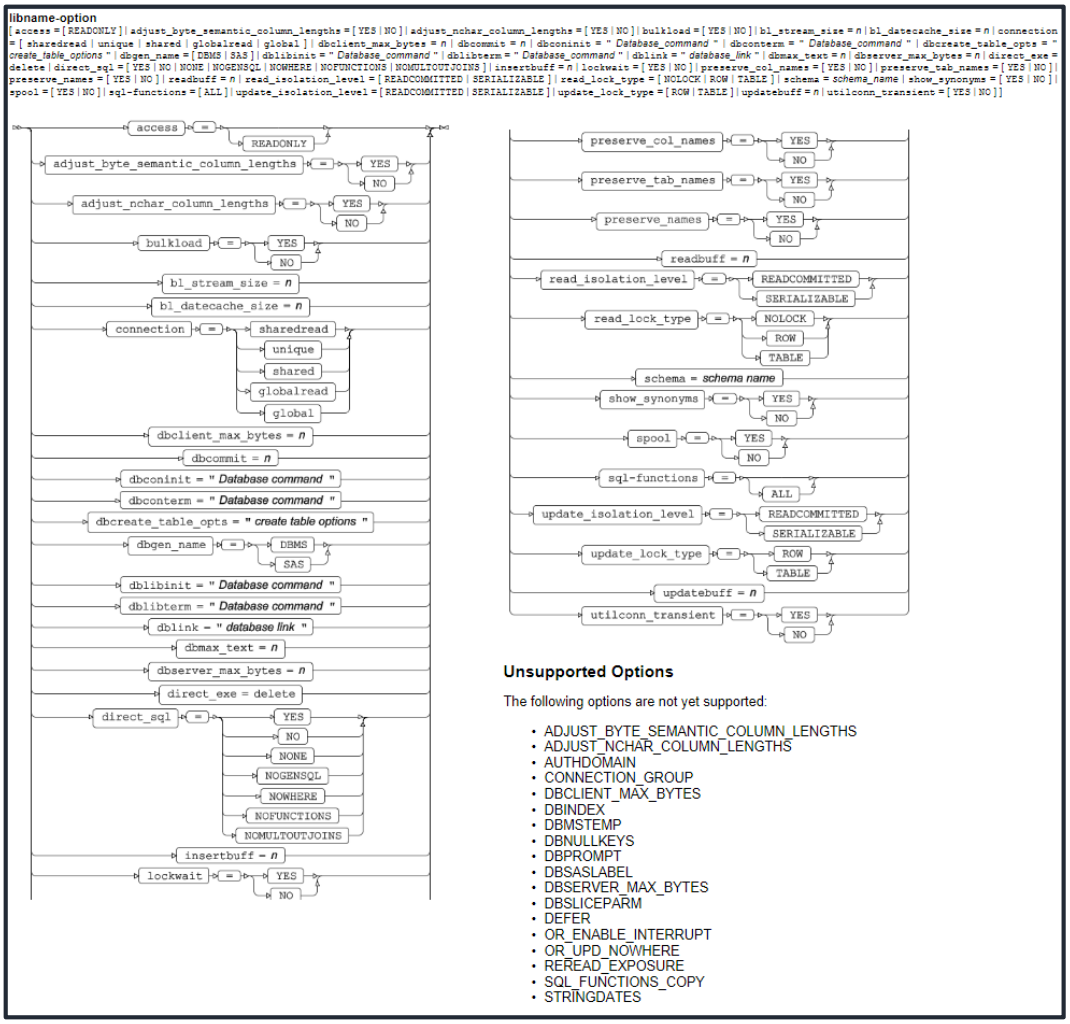
WPS のエディタから以下のプログラムを入力し、実行します。

```
libname in ORACLE user=XXX password=password schema=scott;
```

※ user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

この指定は、WPS がスキーマ scott に ORACLE ネイティブドライバー経由で接続し、scott スキーマ上のテーブルとビューをライブラリ名 in で参照できるようにする指定です。（入出力両方可能） ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

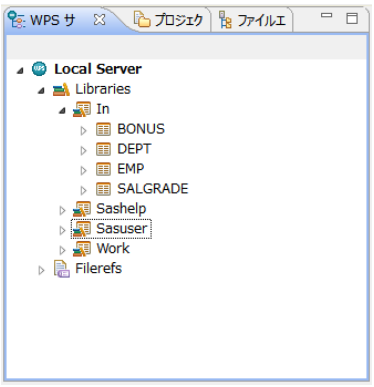
さらに、libname ステートメントには shema=スキーマ名、 dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを追加指定できます。以下のような libname オプションが指定可能です。（前出の ORACLE エンジンのヘルプ画面から確認できます。）



さて、実行すると、ログに 「ライブラリ名 in が割り当てられた」というメッセージが表示されます。このメッセージは ORACLE エンジン経由で DB と接続できたことを表します。

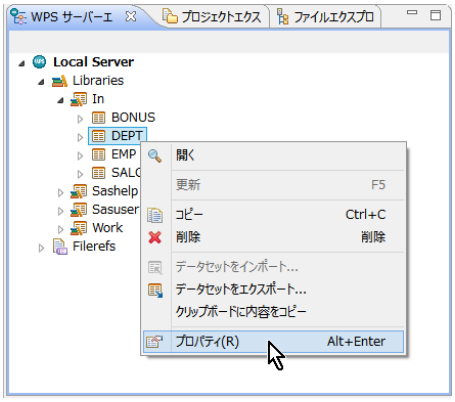
```
49       libname in ORACLE user=XXX password=XXXXXX schema=scott;
NOTE: Library in assigned as follows:
      Engine:        ORACLE
      Physical Name:
```

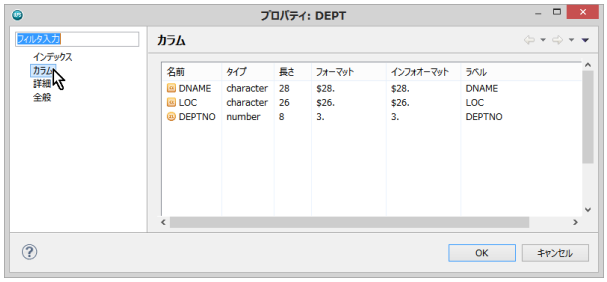
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に scott スキーマ上のテーブル名が含まれていることがわかります。



※ 右の白抜き矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、この scott スキーマにはビューが存在しませんが、ビュー名の場合には矢印が付きません。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。



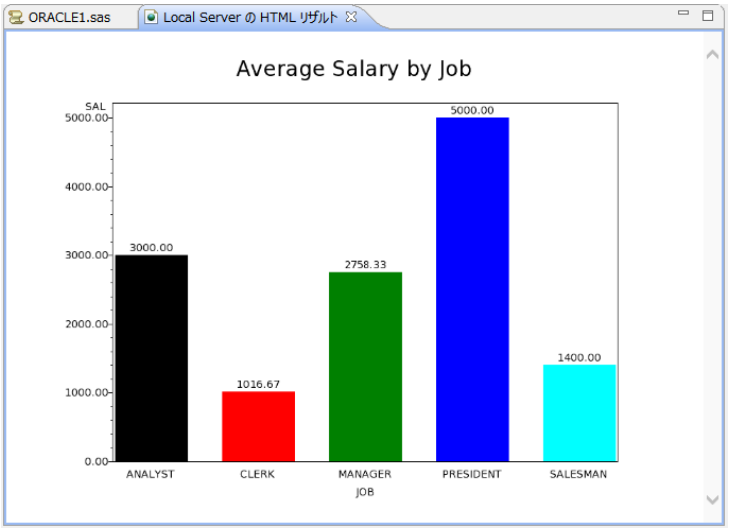


※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、ORACLE エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、EMP テーブルを入力として職種（JOB）別給与（SAL）平均を示す棒グラフを描いています。

```
goptions cback=white;
title "Average Salary by Job";
```

```
proc gchart data=in.emp;
  vbar job/sumvar=SAL type=mean patternid=midpoint mean width=20 space=5;
run;quit;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、EMP テーブルの中で直属の上司（MGR）が社員名（ENAME）"BLAKE" の社員番号（EMPNO）を持つ社員のみを抽出した WPS データセット WORK.BLAKE_grp を DATA ステップで作成しています。

```
data BLAKE_grp;
  if _n_=1 then set in.emp(keep=ENAME EMPNO where=(ENAME="BLAKE")
rename=(EMPNO=SEARCHNO));
  set in.emp;
  if MGR=SEARCHNO;
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ

③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステートメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名など）を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで EMP テーブルから"BLAKE"を直属の上司とする社員を選択した WPS データセット WORK.BLAKE_grp を ORACLE の scott スキーマ上の BLAKE_GRP テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=BLAKE_grp dbms=ORACLE;
  user=XXX;
  password=password;
  schema=scott;
  table=BLAKE_GRP;
  load;
run;quit;
```

※ 注意： DB にアクセスするため ORACLE エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。また、user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は scott スキーマ上の dept テーブルを読んで、LOC="CHICAGO"に該当するレコードのみ抽出した全カラムを含む 1 件のレコードのみ含むテーブル CHICAGO を作成しています。

```
libname in ORACLE user=XXX password=password schema=scott;
proc sql;
  create table in.CHICAGO as
  select *
  from in.dept
  where LOC="CHICAGO";
quit;
```

※ この例では、in ライブラリは ORACLE データベースの scott スキーマを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の in ライブラリは WPS データライブラリではなく、scott スキーマを表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname in ORACLE user=XXX password=password schema=scott;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=in;
  delete CHICAGO;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data in.CHICAGO;
  set in.DEPT;
  where LOC="CHICAGO";
run;
```

4.2. ODBC ドライバー経由の DB テーブルの入出力

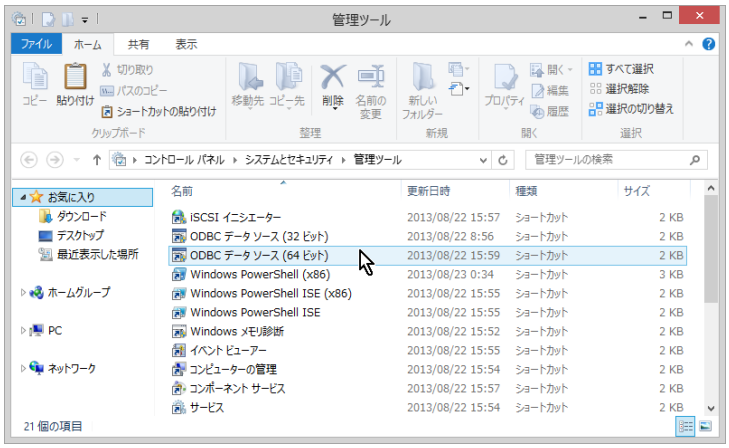
ODBC ドライバーは、Windows 環境や Linux 環境から各種 DB とのインタフェースを共通方式で行うためのソフトウェアです。ほとんどの DB の ODBC ドライバーが DB ベンダやサードパーティから提供されており、ODBC ドライバーの設定を行えば、DB ごとの細かいアクセス条件等の設定を気にせずに、共通の構文で DB アクセスが実現できるという利点があります。

[ODBC ドライバのインストール]

[PC 側の ODBC ドライバーの設定]

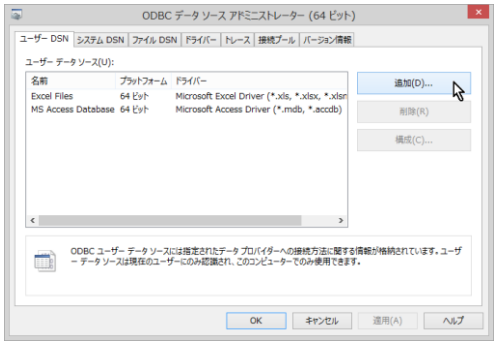
WPS が導入されている PC 側では、以下のような手順で、ORACLE データベースを ODBC データソースとして登録しておきます。

コントロールパネル → システムとセキュリティ → 管理ツールで ODBC データソースを選択します。



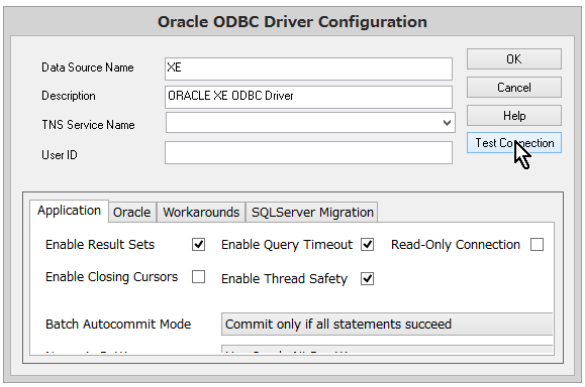
（左図）「追加」をクリックします。

（右図）リストにある ORACLE 用 ODBC ドライバーを選択し、「完了」を押します。



（左図）新規データソースの名前と説明にはユーザーが自由に入力できます。ここでは、Data Source Name に「XE」、Description には「ORACLE XE ODBC Driver」と入力します。「Test Connection」ボタンを押して User Name と Password を入力し、「OK」ボタンを押して「Connection Successful」のメッセージが出て接続テストが成功することを確認すると設定終了です。

（右図）「OK」を押して ODBC データソースリストに ORACLE 用 ODBC データソース「XE」が追加されたことを確認します。



[SQL 言語を使用した DB 読み取り]

SQL 言語を用いて WPS から ODBC ドライバー経由で ORACLE の DB アクセスを行う場合は、以下のように、エンジン名として ODBC を指定し、CONNECT 文のカッコ内に user=ユーザー名、password=パスワード、datasrc=ODBC データソース名などの接続オプションや dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを指定します。

下記は、ODBC ドライバーを用いて、scott スキーマ上の EMP テーブルから、SAL>2000 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.EMP_SAL_OVER2000 を作成しています。

```
proc sql;
  connect to ODBC (user=XXX password=password datasrc=XE);
  create table EMP_SAL_OVER2000 as
  select *
  from connection to ODBC
  (
    select *
    from scott.EMP
    where SAL>2000
   );
  disconnect from ODBC;
quit;
```

※ 2 行目の user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

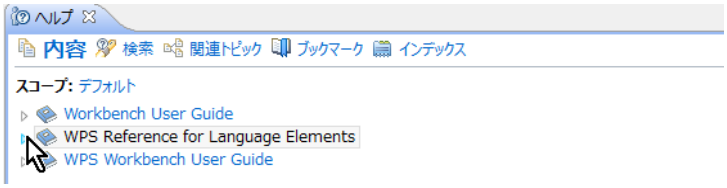
(WPS ワークベンチの実行ログ)

```
146       proc sql;
147         connect to ODBC (user=XXX password=XXXXXX datasrc=XE);
NOTE: Successfully connected to database ODBC as alias ODBC.
148         create table EMP_SAL_OVER2000 as
```

```
149         select *
150         from connection to ODBC
151         (
152           select *
153           from scott.EMP
154           where SAL>2000
155          );
NOTE: Data set "WORK.EMP_SAL_OVER2000" has 5 observation(s) and 8 variable(s)
156         disconnect from ODBC;
NOTE: Successfully disconnected from database ODBC.
157       quit;
NOTE: Procedure sql step took :
      real time : 0.015
      cpu time  : 0.000
```

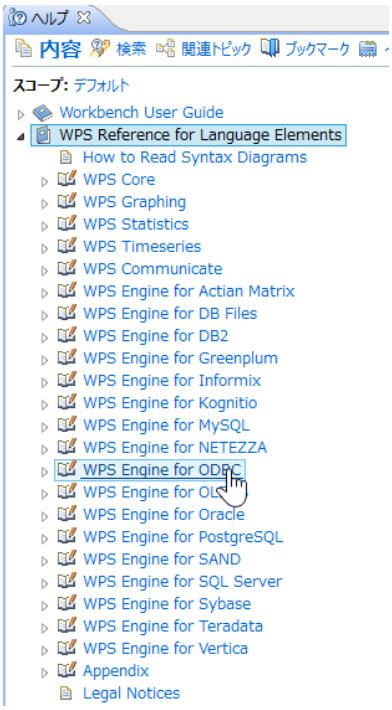
※ ODBC ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。以下のようにヘルプビューで指定可能なオプションが確認できます。

ヘルプビュー[10]の左上にある「内容」をクリックします。
ヘルプ全体の目次が表示されます。WPS Reference for Language Elements をクリックします。

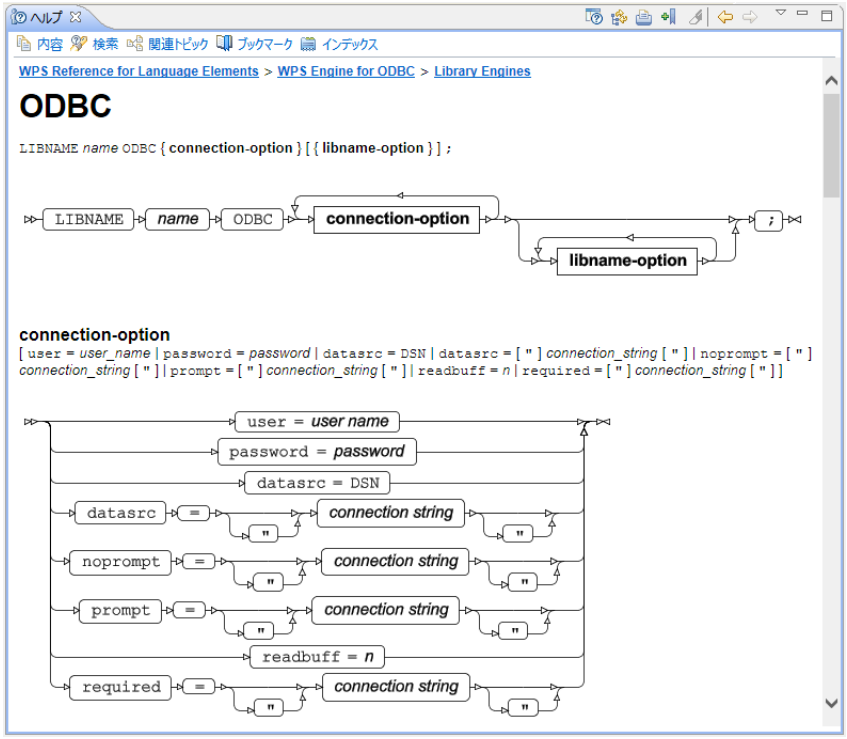


WPS レファレンスを展開し、WPS Engine for ODBC を選択します。

[10] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。



その後、展開を何度か行うと接続オプション（connection option） が表示されます。



これらのオプション（user= password= datasrc= noprompt= prompt= readbuff= required=）は ODBC ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプシ

ョン指定としても、また、次に説明する libname ステートメントによるデータベースアクセスの接続オプションとしても用いることができ、必要に応じて指定します。

[libname ステートメントによる DB 入力]

WPS のエディタから以下のプログラムを実行します。

```
libname in ODBC datasrc=XE user=XXX password=password schema=scott;
```

※ user=xxx password=*password* の xxx と *password* にはそれぞれユーザ名とパスワードを指定します。

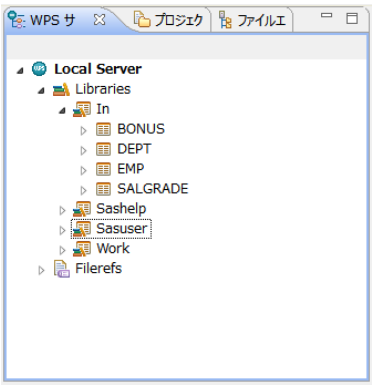
この指定は、WPS がデータソース XE の scott スキーマ上のテーブルに ODBC ドライバー経由で接続し、ライブラリ名 in で参照できるようにする指定です。（入出力両方可能）
ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプション（path=）が必要となる場合があります。

さらに、libname ステートメントには schema=スキーマ名、dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを追加指定できます。指定可能な libname オプションは ORACLE エンジンの libname オプションとほぼ同じですが、詳細は前出の ODBC エンジンのヘルプ画面から確認してください。（Connection-option の下に libname-option が記載されています。）

実行すると、ログに「ライブラリ名 in が割り当てられた」というメッセージが表示されます。このメッセージは ODBC 経由で DB と接続できたことを表します。

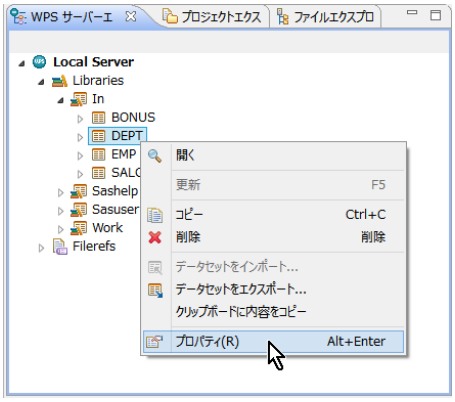
```
167      libname in ODBC datasrc=XE user=XXX password=XXXXXX schema=scott;
NOTE: Library in assigned as follows:
      Engine:        ODBC
      Physical Name: XE
```

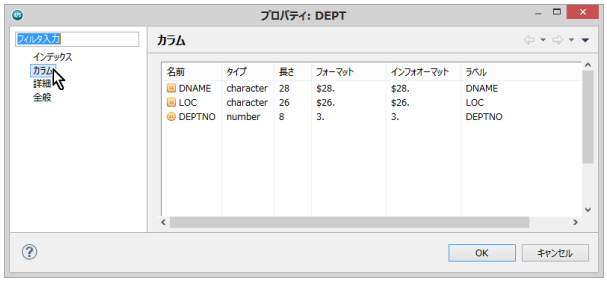
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に scott スキーマ上のテーブル名が含まれていることがわかります。



※ 右の白抜き矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、ビュー名の場合には矢印が付いていません。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。



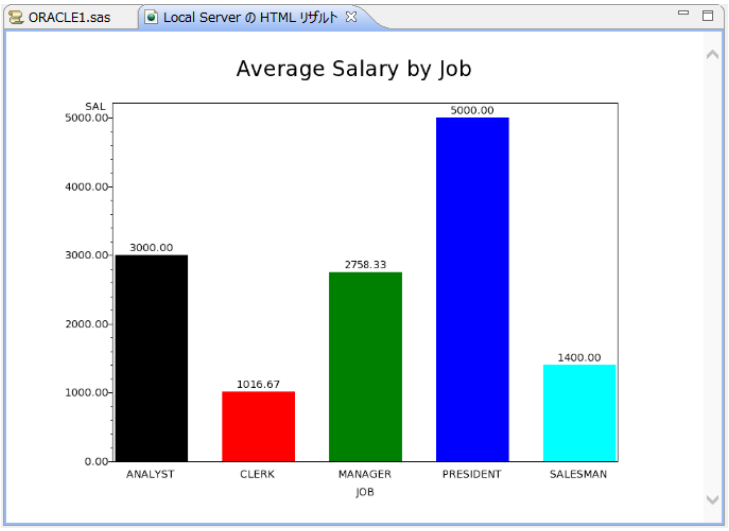


※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、ODBC エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、EMP テーブルを入力として職種（JOB）別給与（SAL）平均を示す棒グラフを描いています。

```
goptions cback=white;
title "Average Salary by Job";
```

```
proc gchart data=in.emp;
  vbar job/sumvar=SAL type=mean patternid=midpoint mean width=20 space=5;
run;quit;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、EMP テーブルの中で直属の上司（MGR）が社員名（ENAME）"BLAKE" の社員番号（EMPNO）を持つ社員のみを抽出した WPS データセット WORK.BLAKE_grp を DATA ステップで作成しています。

```
data BLAKE_grp;
  if _n_=1 then set in.emp(keep=ENAME EMPNO where=(ENAME="BLAKE")
rename=(EMPNO=SEARCHNO));
  set in.emp;
  if MGR=SEARCHNO;
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ

③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステートメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名やスキーマ名）を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで EMP テーブルの中で直属の上司（MGR）が社員名（ENAME）"BLAKE" の社員番号（EMPNO）を持つ社員のみを抽出した WPS データセット WORK.BLAKE_grp を ODBC ドライバー経由で XE データソースの scott スキーマ上の BLAKE_GRP テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=BLAKE_grp dbms=ODBC;
  datasrc=XE;
  user=XXX;
  password=password;
  schema=scott;
  table=BLAKE_GRP;
  load;
run;quit;
```

※ 注意： user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。また、DB にアクセスするための権限や ODBC エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は scott スキーマ上の dept テーブルを読んで、LOC="CHICAGO"に該当するレコードのみ抽出した全カラムを含む 1 件のレコードのみ含むテーブル CHICAGO を作成しています。

```
libname in ODBC datasrc=XE user=XXX password=password schema=scott;
proc sql;
```

```
  create table in.CHICAGO as
  select *
  from in.dept
  where LOC="CHICAGO";
quit;
```

※ この例では、in ライブラリは ODBC 経由で ORACLE データベースの scott スキーマを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の in ライブラリは WPS データライブラリではなく、ORACLE DB を表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

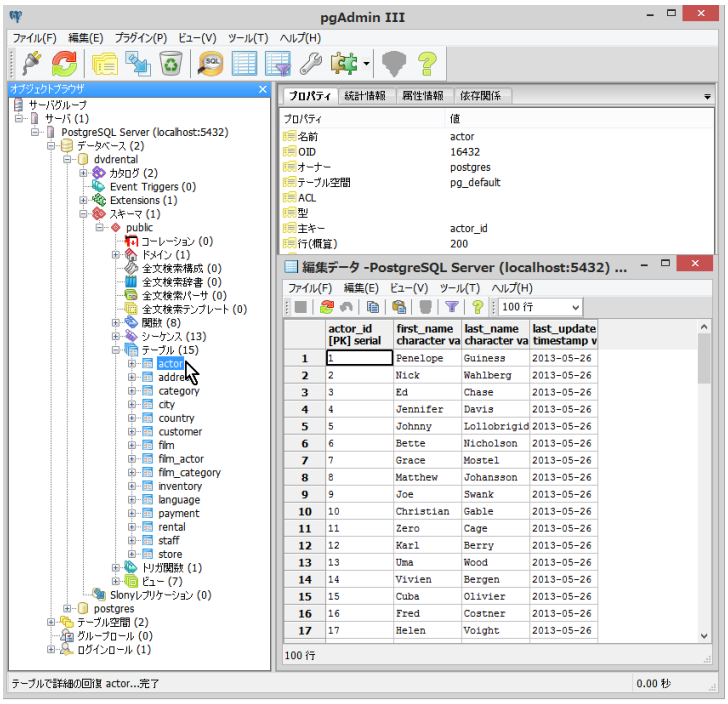
```
libname in ODBC datasrc=XE user=XXX password=password schema=scott;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=in;
  delete CHICAGO;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data in.CHICAGO;
  set in.DEPT;
  where LOC="CHICAGO";
run;
```

5. PostregSQL とのインタフェース

［dvdrental サンプル DB］

dvdrental データベース[11]を例として用います。

[11] http://www.postgresqltutorial.com/load-postgresql-sample-database/ からダウンロードできます。



5.1. ネイティブドライバー経由の DB テーブルの入出力

まず、コマンドラインインタフェース（CLI ドライバー、ネイティブドライバー）を用いて WPS から DB テーブルの入出力を行ってみましょう。

ネイティブドライバーは、一般に ODBC ドライバーより高速に DB アクセスができる点でメリットがあると言われています。
WPS の POSTGSQL エンジンはネイティブドライバーを経由して PostgreSQL データベースとアクセスを行う仕組みを提供します。

【重要な注意】PostgreSQL のネイティブドライバー経由のインタフェースの記述部分については、WPS3.1 のビルド 678 (03.01.01.00.000678) 以降の対応状況を反映しています。

［SQL 言語を使用した DB 読み取り］

DB へのアクセスは SQL 言語を用いて行うことが多く、WPS でも SQL プロシジャを用いて SQL 言語による DB アクセスが可能です。

SQL 言語を用いて WPS からネイティブドライバー経由で PostgreSQL の DB アクセスを

行う場合は、以下のように、エンジン名として POSTGSQL を指定し、CONNECT 文のカッコ内に database=データベース名、user=ユーザー名、password=パスワード などの接続オプションや shema=スキーマ名、dbmax_text=n（文字テキストの読み取り最大長さの設定）などの後述する libname オプションを必要に応じて指定します。

以下は dvdrental データベースの public スキーマ上の city テーブルから、country_id=50 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.japan_city を作成しています。

```
proc sql;
  connect to POSTGSQL (database=dvdrental user=XXX password=password);
  create table japan_city as
  select *
  from connection to POSTGSQL
  (
    select *
    from public.city
    where country_id=50
  );
  disconnect from POSTGSQL;
quit;
```

※ 2 行目の user=xxx password=*password* の xxx と *password* にはそれぞれユーザ名とパスワードを指定します。

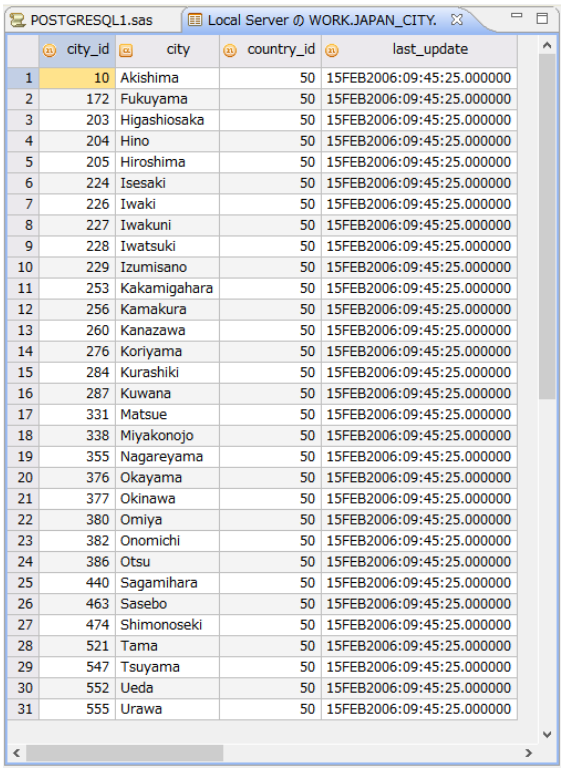
3 行目の create table 文は WPS データセット WORK.japan_city を作成する指定です。

8 行目の from 節で指定した public.city は dvdrental データベースの public スキーマ上の city テーブルを参照しています。

(WPS ワークベンチの実行ログ)

```
8         proc sql;
9           connect to POSTGSQL (database=dvdrental user=XXX password=XXXXXXX);
NOTE: Successfully connected to database POSTGSQL as alias POSTGSQL.
10          create table japan_city as
11          select *
12          from connection to POSTGSQL
13          (
14            select *
15            from public.city
16            where country_id=50
17          );
NOTE: Data set "WORK.japan_city" has 31 observation(s) and 4 variable(s)
18          disconnect from POSTGSQL;
NOTE: Successfully disconnected from database POSTGSQL.
19        quit;
NOTE: Procedure sql step took :
      real time : 0.0093
      cpu time  : 0.000
```

作成された WPS データセット WORK.staff データセットの内容を確認します。

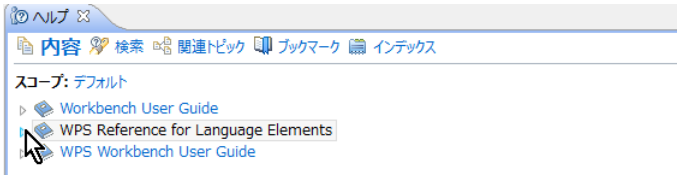

POSTGRESQL1.sas　Local Server の WORK.JAPAN_CITY.

	city_id	city	country_id	last_update
1	10	Akishima	50	15FEB2006:09:45:25.000000
2	172	Fukuyama	50	15FEB2006:09:45:25.000000
3	203	Higashiosaka	50	15FEB2006:09:45:25.000000
4	204	Hino	50	15FEB2006:09:45:25.000000
5	205	Hiroshima	50	15FEB2006:09:45:25.000000
6	224	Isesaki	50	15FEB2006:09:45:25.000000
7	226	Iwaki	50	15FEB2006:09:45:25.000000
8	227	Iwakuni	50	15FEB2006:09:45:25.000000
9	228	Iwatsuki	50	15FEB2006:09:45:25.000000
10	229	Izumisano	50	15FEB2006:09:45:25.000000
11	253	Kakamigahara	50	15FEB2006:09:45:25.000000
12	256	Kamakura	50	15FEB2006:09:45:25.000000
13	260	Kanazawa	50	15FEB2006:09:45:25.000000
14	276	Koriyama	50	15FEB2006:09:45:25.000000
15	284	Kurashiki	50	15FEB2006:09:45:25.000000
16	287	Kuwana	50	15FEB2006:09:45:25.000000
17	331	Matsue	50	15FEB2006:09:45:25.000000
18	338	Miyakonojo	50	15FEB2006:09:45:25.000000
19	355	Nagareyama	50	15FEB2006:09:45:25.000000
20	376	Okayama	50	15FEB2006:09:45:25.000000
21	377	Okinawa	50	15FEB2006:09:45:25.000000
22	380	Omiya	50	15FEB2006:09:45:25.000000
23	382	Onomichi	50	15FEB2006:09:45:25.000000
24	386	Otsu	50	15FEB2006:09:45:25.000000
25	440	Sagamihara	50	15FEB2006:09:45:25.000000
26	463	Sasebo	50	15FEB2006:09:45:25.000000
27	474	Shimonoseki	50	15FEB2006:09:45:25.000000
28	521	Tama	50	15FEB2006:09:45:25.000000
29	547	Tsuyama	50	15FEB2006:09:45:25.000000
30	552	Ueda	50	15FEB2006:09:45:25.000000
31	555	Urawa	50	15FEB2006:09:45:25.000000

※ 【重要な注意】WPS3.1.1 の POSTGSQL エンジンでは WPS データセットのデータ型に変換できないデータベースカラムのデータ型（ENUM 型,mpaa_rating 型, text[]型, tsvector 型）があり、これらのデータ型のカラムは除いて読み取る必要があります。対応としては後述の ODBC ドライバーを用いる方法もあります。

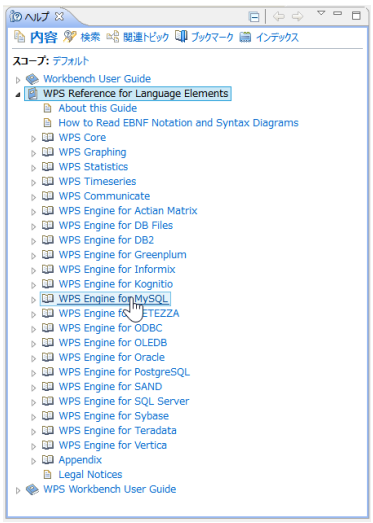
※ 注意：POSTGSQL ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。指定可能なオプションは、以下のように、ヘルプビューで確認できます。

まず、ヘルプビュー[12]の左上にある「内容」をクリックします。ヘルプ全体の目次が表示されますので、WPS Reference for Language Elements をクリックします。

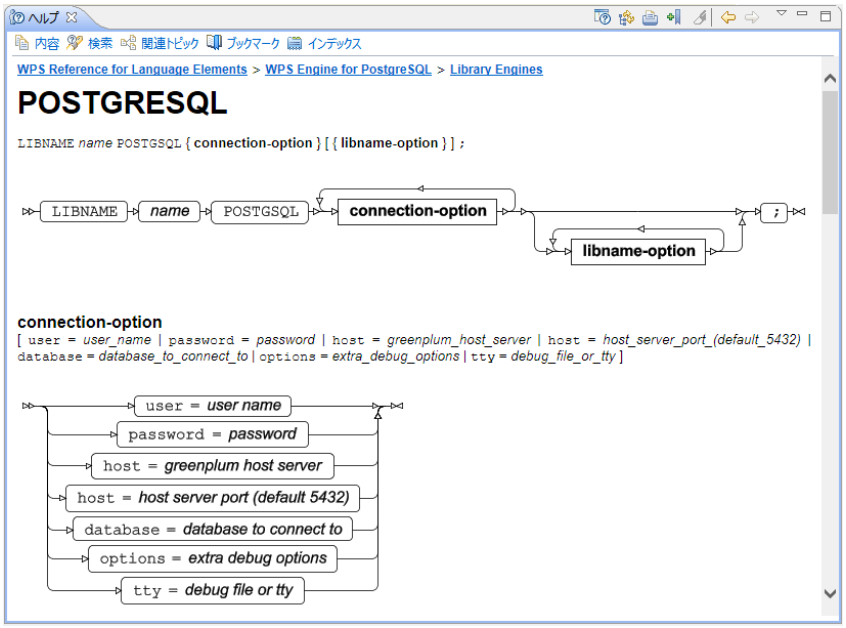


WPS レファレンスを展開し、WPS Engine for PostgreSQL を選択します。

[12] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。



Library Engines → POSTGRESQL を選択すると、データベース接続オプション（connection option）が表示されます。



これらの接続オプション（user= password= database= server= port=）は POSTGSQL ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプション指定としても、また、次に節で説明する libname ステートメントによる DB 接続オプションとしても用いることができ、必要に応じて指定します。

［libname ステートメントによる DB 入力］

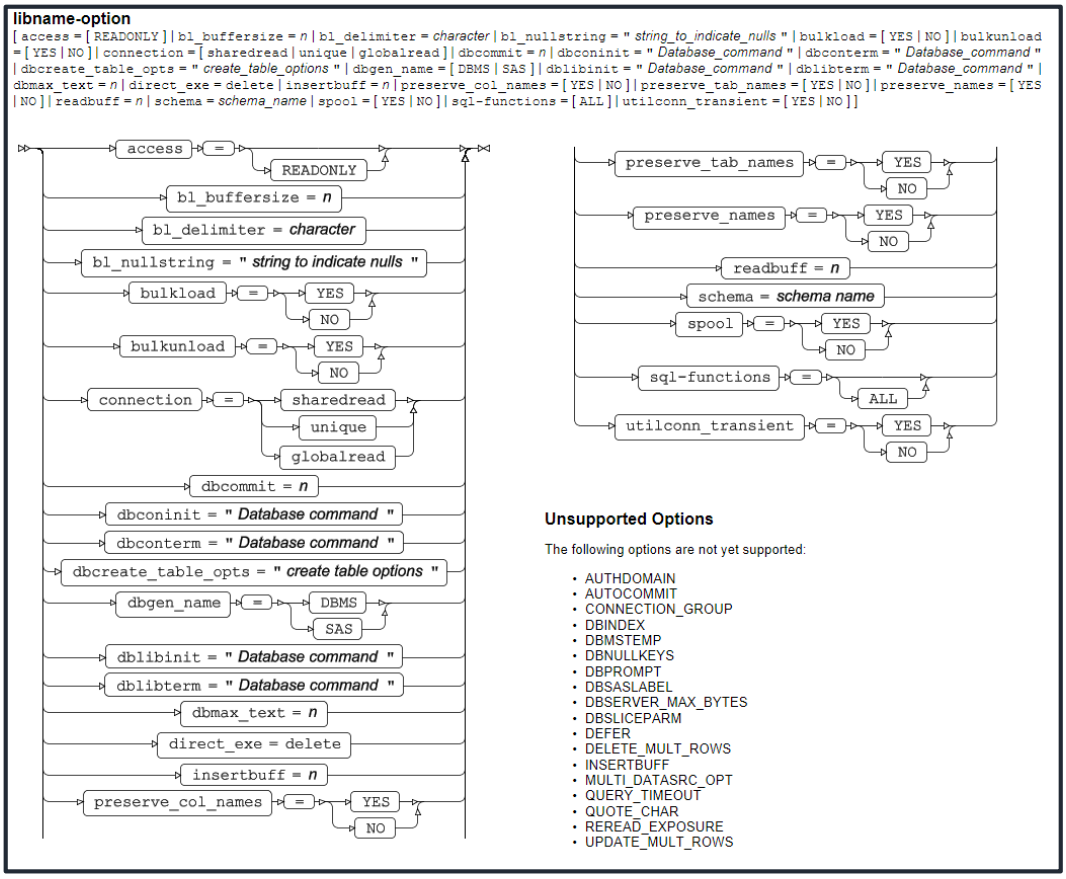
WPS のエディタから以下のプログラムを入力し、実行します。

```
libname in POSTGSQL database=dvdrental user=XXX password=password
schema=public;
```

※ user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

この指定は、WPS がデータベース dvdrental の public スキーマに POSTGSQL ネイティブドライバー経由で接続し、dvdrental データベースの public スキーマ上のテーブルとビューをライブラリ名 in で参照できるようにする指定です。（入出力両方可能） ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

さらに、libname ステートメントには dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを追加指定できます。以下のような libname オプションが指定可能です。（前出の POSTGSQL エンジンのヘルプ画面から確認できます。）

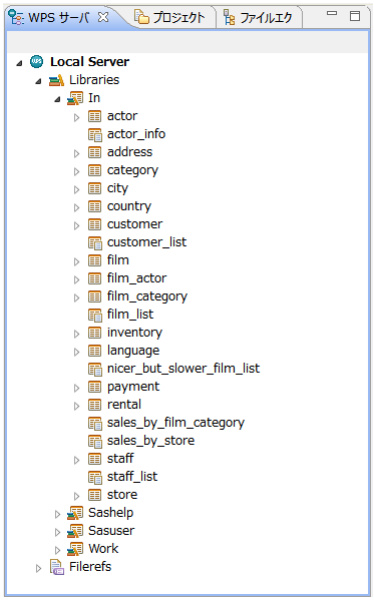


さて、実行すると、ログに 「ライブラリ名 in が割り当てられた」というメッセージが表

示されます。このメッセージは POSTGSQL エンジン経由で DB と接続できたことを表します。

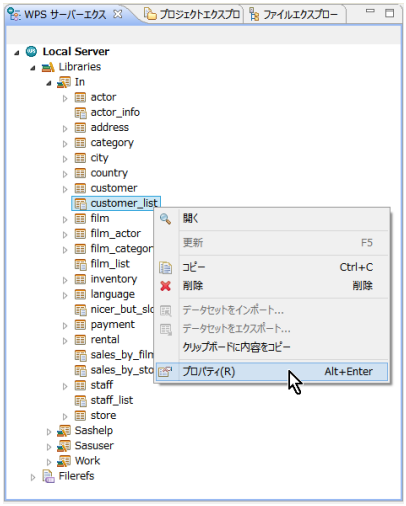
```
29         libname in POSTGSQL database=dvdrental user=XXX password=XXXXXXX schema=public;
NOTE: Library in assigned as follows:
      Engine:        POSTGRESQL
      Physical Name:
```

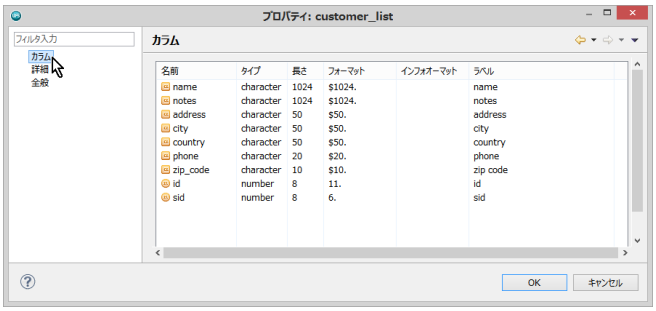
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に dvdrental データベースの public スキーマ上のテーブル名およびビュー名が含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、付いていないのはビュー名です。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。



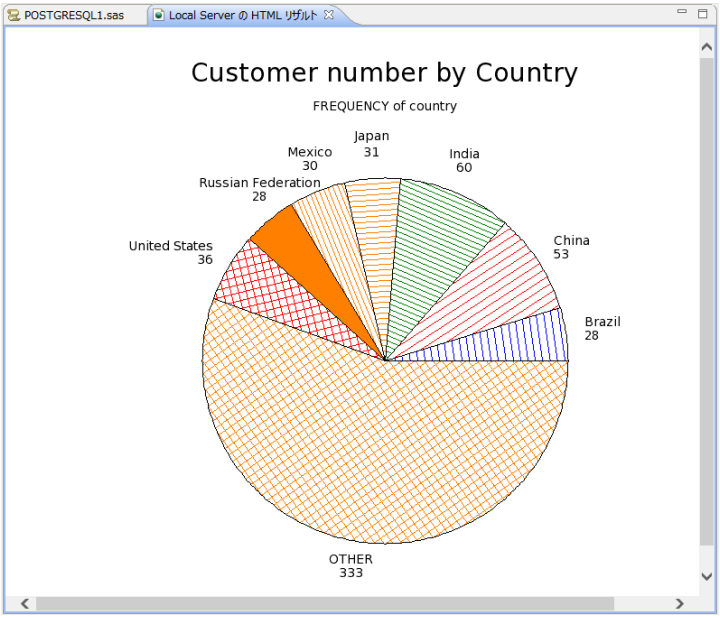


※【重要な注意 1】WPS3.1.1 の POSTGSQL エンジンでは WPS データセットのデータ型に変換できないデータ型（ENUM 型,mpaa_rating 型, text[]型, tsvector 型）があり、これらのデータ型のカラムを含むテーブルやビューは、アクセスエラーが発生し読み取りはできません。対応としては後述の ODBC ドライバーを用いてください。

※【重要な注意 2】:テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、POSTGSQL エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、customer_list ビューを入力として国（country）別の登録顧客数の棒グラフを描いています。

```
goptions cback=white;
title "Customer number by Country";
proc gchart data=in.customer_list;
  pie country;
run;quit;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、customer_list ビューから国名（country）を大文字化した値が"GREECE"のレコードを選択した変数 id, name, address, city, country を含む WPS データセット WORK.custromer_greece を DATA ステップで作成しています。

```
data customer_greece;
  set in.customer_list(keep=id name address city country);
  where upcase(country)="GREECE";
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ

前節の最後のプログラムで customer_list ビューから国名（country）を大文字化した値が"GREECE"のレコードを選択した変数 id, name, address, city, country を含む WPS データ

セット WORK.custromer_greece を POSTGRESQL の dvdrental データベースの public スキームの customer_greece テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=customer_greece dbms=POSTGSQL;
  user=XXX;
  password=password;
  database=dvdrental;
  host=localhost;
  port=5432;
  table=public.customer_greece;
  load;
run;quit;
```

※ 注意： DB にアクセスするため POSTGSQL エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。また、user=XXX password=*password* の XXX と *password* にはそれぞれユーザ名とパスワードを指定します。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は dvdrental データベースの public スキーム上の city テーブルを読んで、country_id=50 に該当するレコードのみ抽出した全カラムを含むテーブル japan_city を作成しています。

```
libname in POSTGSQL database=dvdrental user=XXX password=password
schema=public;
proc sql;
  create table in.japan_city as
  select *
  from in.city
  where country_id=50;
quit;
```

※ この例では、in ライブラリは dvdrental データベースの public スキームを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の in ライブラリは WPS データライブラリではなく、dvdrental データベースの public スキームを表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。
【重要な注意】現時点では、実行後、再度 libname ステートメントを実行してください。そうでないと、データセットの内容やプロパティを確認できないという問題があります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname in POSTGSQL database=dvdrental user=XXX password=password
schema=public;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=in;
  delete japan_city;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data in.japan_city;
  set in.city;
  where country_id=50;
run;
```

【重要な注意】現時点では、実行後、再度 libname ステートメントを実行してください。そうでないと、データセットの内容やプロパティを確認できないという問題があります。

5.2. ODBC ドライバー経由の DB テーブルの入出力

ODBC ドライバーは、Windows 環境や Linux 環境から各種 DB とのインタフェースを共通方式で行うためのソフトウェアです。ほとんどの DB の ODBC ドライバーが DB ベンダやサードパーティから提供されており、ODBC ドライバーの設定を行えば、DB ごとの細かいアクセス条件等の設定を気にせずに、共通の構文で DB アクセスが実現できるという利点があります。

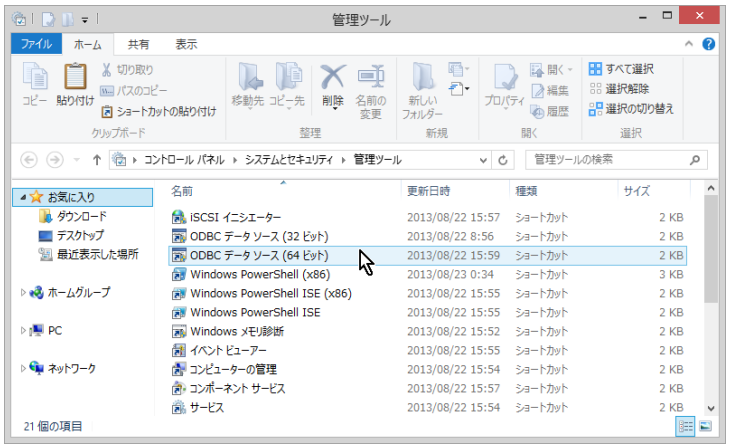
［ODBC ドライバのインストール］

PostgreSQL の ODBC ドライバーは PostgreSQL の動作環境とバージョンに合わせてインストールします。 もしも、次の［PC 側の ODBC ドライバーの設定］で PostgreSQL の ODBC ドライバーが見当たらない場合は、インストール環境を確認してください。

［PC 側の ODBC ドライバーの設定］

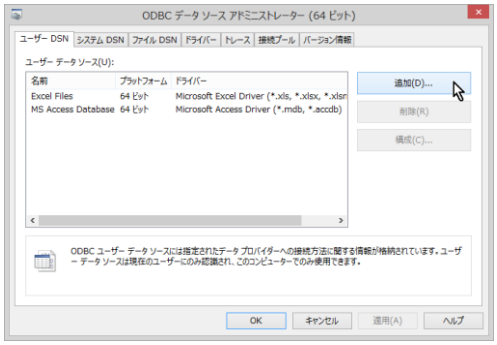
WPS が導入されている PC 側では、以下のような手順で、PostgreSQL の dvdrental データベースを ODBC データソースとして登録しておきます。

コントロールパネル → システムとセキュリティ → 管理ツールで ODBC データソースを選択します。



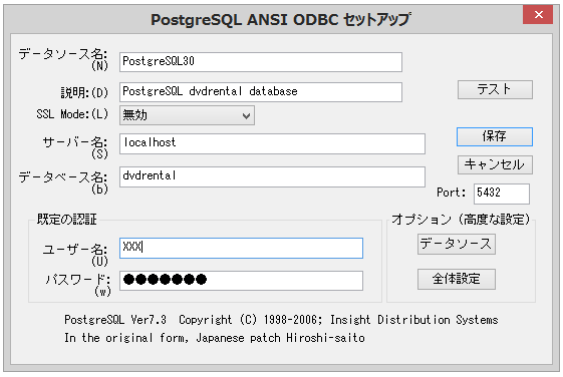
（左図）「追加」をクリックします。

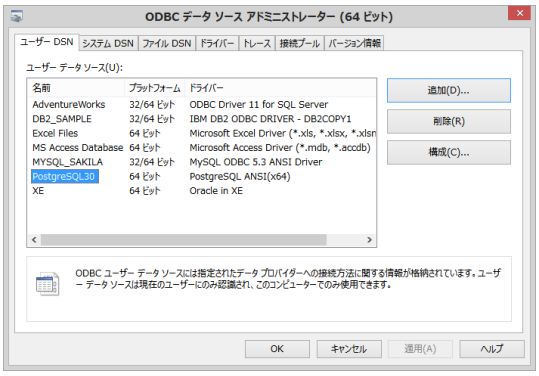
（右図）リストにある PostgreSQL 用 ODBC ドライバーを選択し、「完了」を押します。



（左図）データソース名と説明にはユーザーが自由に入力できます。ここでは、データソース名は「PostgreSQL30」（既定のまま）、説明には「PostgreSQL dvdrental database」と入力します。データベース名には「dvdrental」、ユーザー名とパスワードを入力します。次に「テスト」ボタンを押し接続テストを行います。接続テストが成功することを確認すると「保存」ボタンを押し設定終了です。

（右図）ODBC データソースリストに PostgreSQL 用 ODBC ドライバーが「PostgreSQL30」というデータソース名で追加されたことを確認します。





［SQL 言語を使用した DB 読み取り］

SQL 言語を用いて WPS から ODBC ドライバー経由で PostgreSQL の DB アクセスを行う場合は、以下のように、エンジン名として ODBC を指定し、CONNECT 文のカッコ内に datasrc=ODBC データソース名などの接続オプションや dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを必要に応じて指定します。

下記は、ODBC ドライバーを用いて、dvdrental データベースの public スキーマ上の city テーブルから、country_id=50 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.japan_city を作成しています。

```
proc sql;
  connect to ODBC (datasrc=PostgreSQL30);
  create table japan_city as
  select *
  from connection to ODBC
  (
    select *
    from city
    where country_id=50
  );
  disconnect from ODBC;
quit;
```

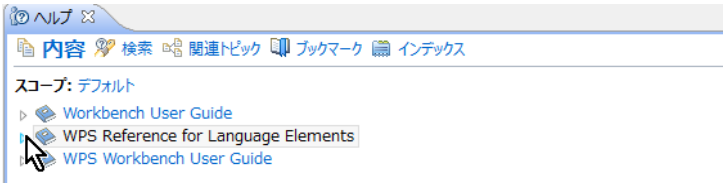
(WPS ワークベンチの実行ログ)

```
8         proc sql;
9           connect to ODBC (datasrc=PostgreSQL30);
NOTE: Successfully connected to database ODBC as alias ODBC.
10          create table japan_city as
11          select *
12          from connection to ODBC
13          (
14            select *
15            from public.city
16            where country_id=50
17          );
NOTE: Data set "WORK.japan_city" has 31 observation(s) and 4 variable(s)
```

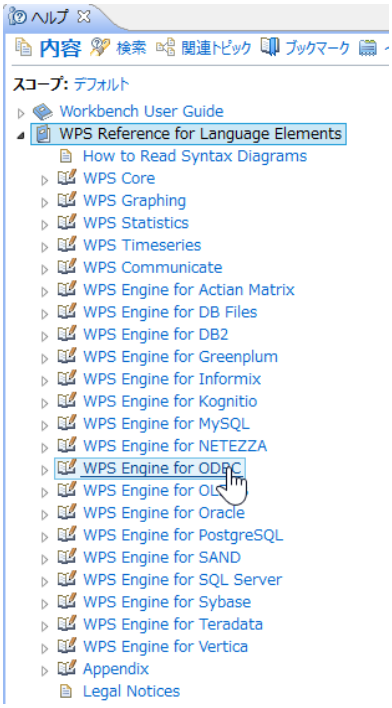
```
18          disconnect from ODBC;
NOTE: Successfully disconnected from database ODBC.
19        quit;
NOTE: Procedure sql step took :
      real time : 0.093
      cpu time  : 0.015
```

※ ODBC ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。以下のようにヘルプビューで指定可能なオプションが確認できます。

ヘルプビュー[13]の左上にある「内容」をクリックします。
ヘルプ全体の目次が表示されます。WPS Reference for Language Elements をクリックします。

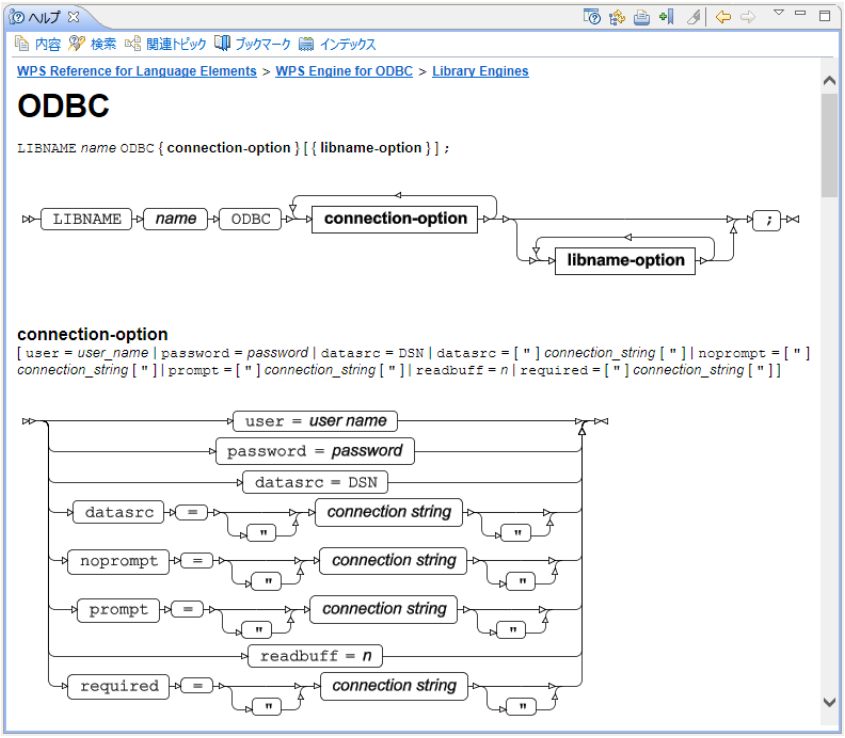


WPS レファレンスを展開し、WPS Engine for ODBC を選択します。



その後、展開を何度か行うと接続オプション（connection option） が表示されます。

[13] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。



これらのオプション（user= password= datasrc= noprompt= prompt= readbuff= required=）は ODBC ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプション指定としても、また、次に説明する libname ステートメントによるデータベースアクセスの接続オプションとしても用いることができ、必要に応じて指定します。

［libname ステートメントによる DB 入力］

WPS のエディタから以下のプログラムを実行します。

```
libname in ODBC datasrc=PostgreSQL30 schema=public;
```

この指定は、WPS がデータソース PostgreSQL30 に ODBC ドライバー経由で接続し、ライブラリ名 in で参照できるようにする指定です。（入出力両方可能）
ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

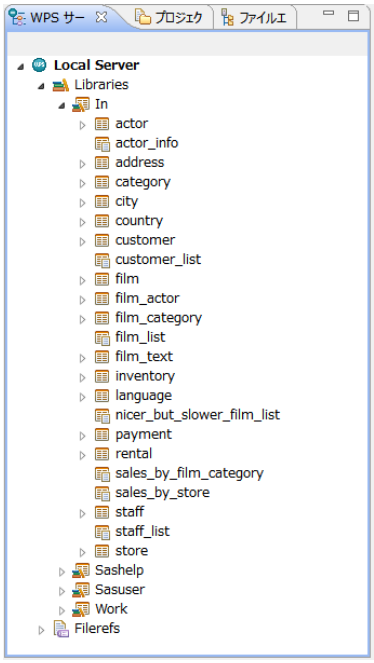
さらに、libname ステートメントには schema=スキーマ名、dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを追加指定できます。指定可能な libname オプションは POSTGSQL エンジンの libname オプションとほぼ同じですが、詳細

は前出の ODBC エンジンのヘルプ画面から確認してください。（Connection-option の下に libname-option が記載されています。）

実行すると、ログに「ライブラリ名 in が割り当てられた」というメッセージが表示されます。このメッセージは ODBC 経由で DB と接続できたことを表します。

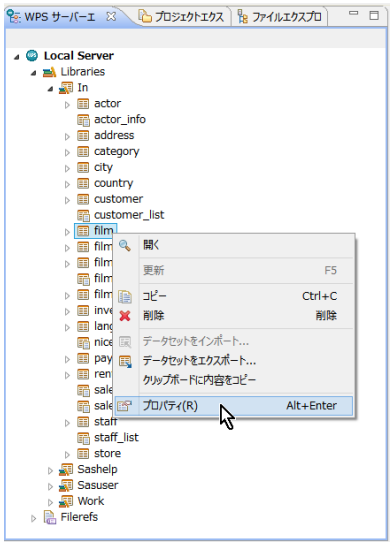
```
345       libname in ODBC datasrc=PostgreSQL30 schema=public;
NOTE: Library in assigned as follows:
      Engine:        ODBC
      Physical Name: PostgreSQL30
```

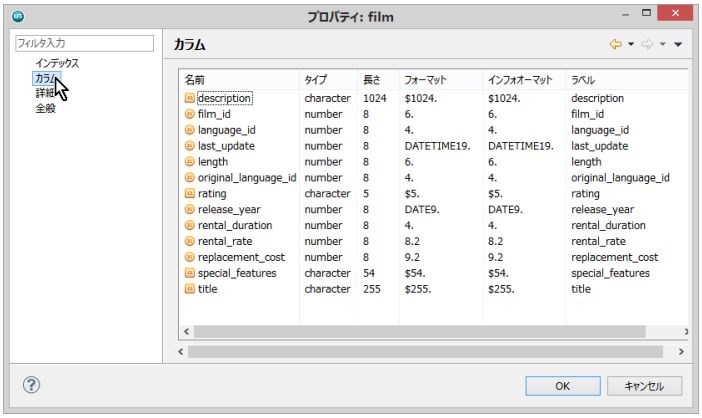
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、in ライブラリがあることを確認してください。この中に dvdrental データベースの public スキーマ上のテーブル名およびビュー名が含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、付いていないのはビュー名です。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。

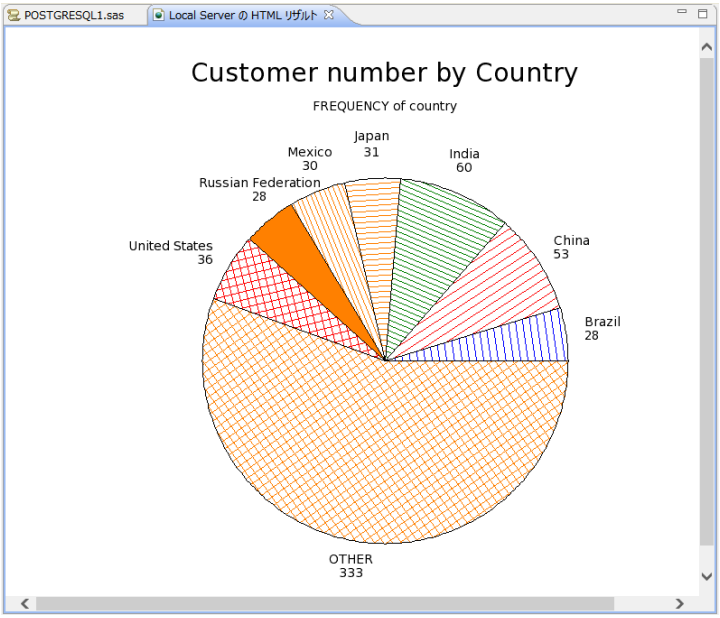




※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、ODBC エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、customer_list ビューを入力として国（country）別の登録顧客数の棒グラフを描いています。

```
goptions cback=white;
title "Customer number by Country";
proc gchart data=in.customer_list;
  pie country;
run;quit;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、customer_list ビューから国名（country）を大文字化した値が"GREECE"のレコードを選択した変数 id, name, address, city, country を含む WPS データセット WORK.custromer_greece を DATA ステップで作成しています。

```
data customer_greece;
  set in.customer_list(keep=id name address city country);
  where upcase(country)="GREECE";
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ
③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステートメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名やスキーマ名）を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで sakia テーブルから題名（TITLE）に"BIRD"が含まれるレコードを選択した変数 File_id, title, rental_rate, rating を含む WPS データセット WORK.film_BIRD を MySQL の film_BIRD テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=customer_greece dbms=ODBC;
  datasrc=PostgreSQL30;
  table=public.customer_greece;
  load;
run;quit;
```

※ 注意：DB にアクセスするための権限や ODBC エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は sakila データベースの city テーブルを読んで、country_id=50 に該当するレコードのみ抽出した全カラムを含むテーブル japan_city を作成しています。

```
libname in ODBC datasrc=PostgreSQL30 schema=public;
proc sql;
  create table in.japan_city as
  select *
  from in.city
  where country_id = 50;
quit;
```

※ この例では、in ライブラリは ODBC 経由で dvdrental データベースの public スキーマを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の in ライブラリは WPS データライブラリではなく、DB を表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname in ODBC datasrc=PostgreSQL30 schema=public;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=in;
  delete japan_city;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data in.japan_city;
  set in.city;
  where country_id=50;
run;
```

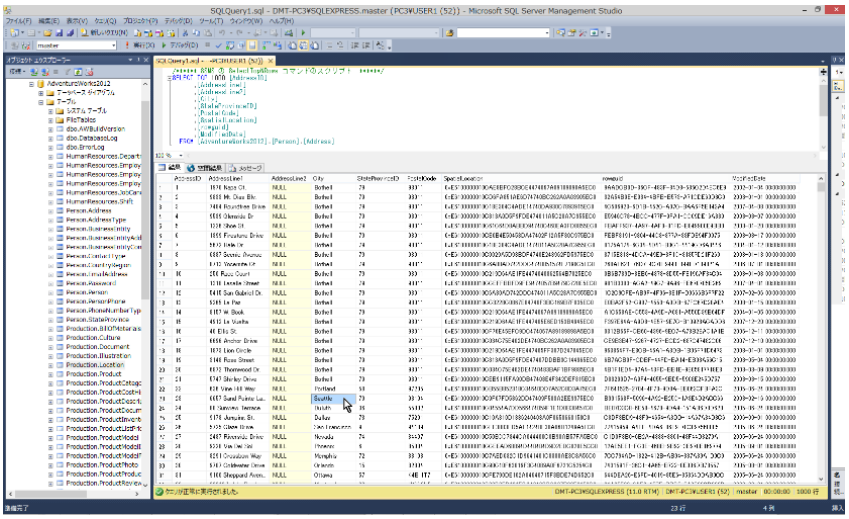
6. SQL Server とのインタフェース

［AdventureWorks2012 サンプル DB］

SQL Server 2012 Express Edition に アタッチした Microsoft 提供の AdventureWorks2012 サンプル DB[14]を例として用います。
SQL Server Managenent Studio で AdventureWorks2012 DB をアタッチし、内容を確認してください。

(AdventureWorks2012 データベースのテーブル一覧と Person.Address テーブルの内容)

[14] http://msftdbprodsamples.codeplex.com/releases/view/55330 の DOWNLOADS タブから AdventureWorks2012 Data File をダウンロードします。



6.1. ネイティブドライバー経由の DB テーブルの入出力

まず、コマンドラインインタフェース（CLI ドライバー、ネイティブドライバー）を用いて WPS から DB テーブルの入出力を行ってみましょう。
ネイティブドライバーは DB と一緒にインストールされており、一般に ODBC ドライバーより高速に DB アクセスができる点でメリットがあると言われています。
WPS の SQLSERVR エンジンはネイティブドライバーを経由して SQL Server データベースとアクセスを行う仕組みを提供します。

[SQL 言語を使用した DB 読み取り]

DB へのアクセスは SQL 言語を用いて行うことが多く、WPS でも SQL プロシジャを用いて SQL 言語による DB アクセスが可能です。

SQL 言語を用いて WPS からネイティブドライバー経由で SQL Server の DB アクセスを行う場合は、以下のように、エンジン名として SQLSERVR を指定し、CONNECT 文のカッコ内に server=サーバー名、database=DB 名などの接続オプションや後述する dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを指定します。

以下は AdventureWorks2012 データベースの Sales.Customer テーブルから、TerritoryID=1 に該当するレコードのみ抽出した全カラムを含む WPS データセット WORK.Customer を作成しています。

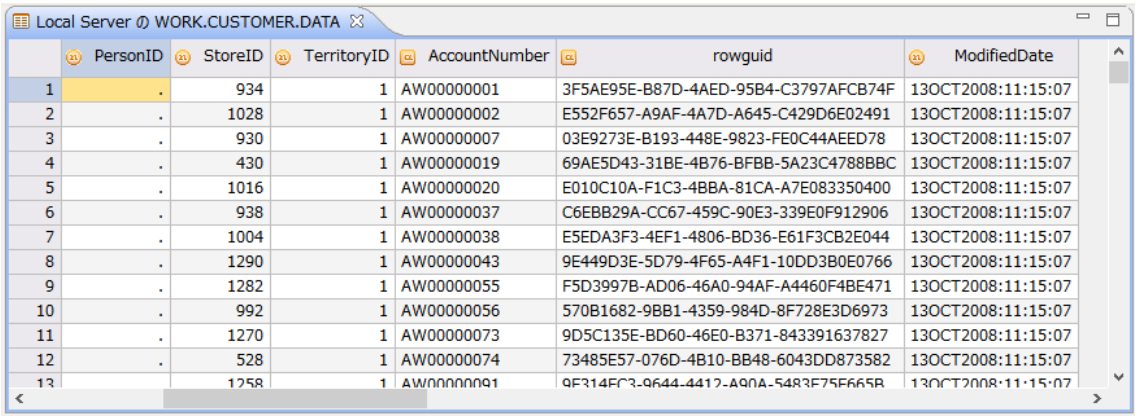
```
proc sql;
  connect to SQLSERVR(server="PC3¥SQLEXPRESS" database=AdventureWorks2012);
  create table Customer as
  select *
  from connection to SQLSERVR
  (
    select *
    from Sales.Customer
    where TerritoryID = 1;
  );
  disconnect from SQLSERVR;
quit;
```

※ 3 行目の create table 文は WPS データセット WORK.Customer を作成する指定です。8 行目の from 節で指定した Sales.Customer は AdventureWorks2012 データベースの sales スキーマ上の Customer テーブルを参照しています。

(WPS ワークベンチの実行ログ)

```
8         proc sql;
9           connect to SQLSERVR(server="DMT-PC3¥SQLEXPRESS" database=AdventureWorks2012);
NOTE: Successfully connected to database SQLSERVR as alias SQLSERVR.
10          create table Customer as
11          select *
12          from connection to SQLSERVR
13          (
14            select *
15            from Sales.Customer
16            where TerritoryID = 1;
17          );
NOTE: Data set "WORK.Customer" has 3520 observation(s) and 7 variable(s)
18          disconnect from SQLSERVR;
NOTE: Successfully disconnected from database SQLSERVR.
19        quit;
NOTE: Procedure sql step took :
      real time : 0.062
      cpu time  : 0.031
```

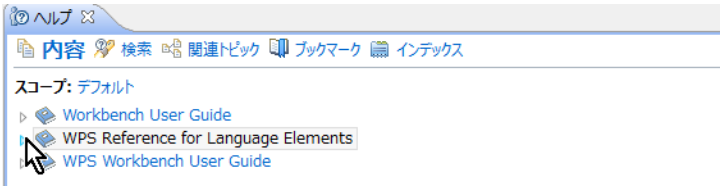
作成された WPS データセット WORK.Customer データセットの内容を確認します。


Local Server の WORK.CUSTOMER.DATA

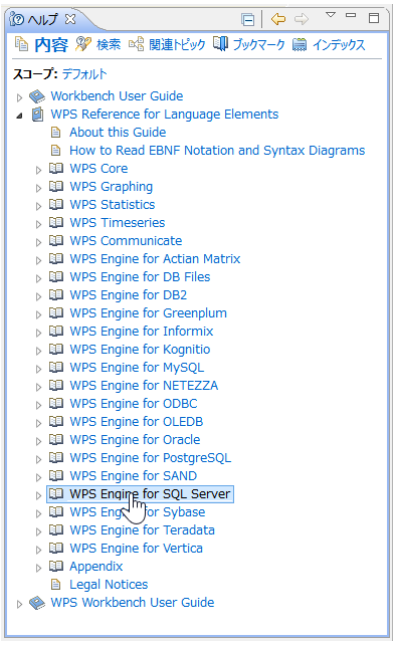
	PersonID	StoreID	TerritoryID	AccountNumber	rowguid	ModifiedDate
1	.	934	1	AW00000001	3F5AE95E-B87D-4AED-95B4-C3797AFCB74F	13OCT2008:11:15:07
2	.	1028	1	AW00000002	E552F657-A9AF-4A7D-A645-C429D6E02491	13OCT2008:11:15:07
3	.	930	1	AW00000007	03E9273E-B193-448E-9823-FE0C44AEED78	13OCT2008:11:15:07
4	.	430	1	AW00000019	69AE5D43-31BE-4B76-BFBB-5A23C4788BBC	13OCT2008:11:15:07
5	.	1016	1	AW00000020	E010C10A-F1C3-4BBA-81CA-A7E083350400	13OCT2008:11:15:07
6	.	938	1	AW00000037	C6EBB29A-CC67-459C-90E3-339E0F912906	13OCT2008:11:15:07
7	.	1004	1	AW00000038	E5EDA3F3-4EF1-4806-BD36-E61F3CB2E044	13OCT2008:11:15:07
8	.	1290	1	AW00000043	9E449D3E-5D79-4F65-A4F1-10DD3B0E0766	13OCT2008:11:15:07
9	.	1282	1	AW00000055	F5D3997B-AD06-46A0-94AF-A4460F4BE471	13OCT2008:11:15:07
10	.	992	1	AW00000056	570B1682-9BB1-4359-984D-8F728E3D6973	13OCT2008:11:15:07
11	.	1270	1	AW00000073	9D5C135E-BD60-46E0-B371-843391637827	13OCT2008:11:15:07
12	.	528	1	AW00000074	73485E57-076D-4B10-BB48-6043DD873582	13OCT2008:11:15:07
13		1258	1	AW00000091	9F314EC3-9644-4412-A90A-5483E75E665B	13OCT2008:11:15:07

※ 注意：SQLSERVR ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の（）内のオプションに、その他の指定が必要となる場合があります。指定可能なオプションは、以下のように、ヘルプビューで確認できます。

まず、ヘルプビュー[15]の左上にある「内容」をクリックします。ヘルプ全体の目次が表示されますので、WPS Reference for Language Elements をクリックします。

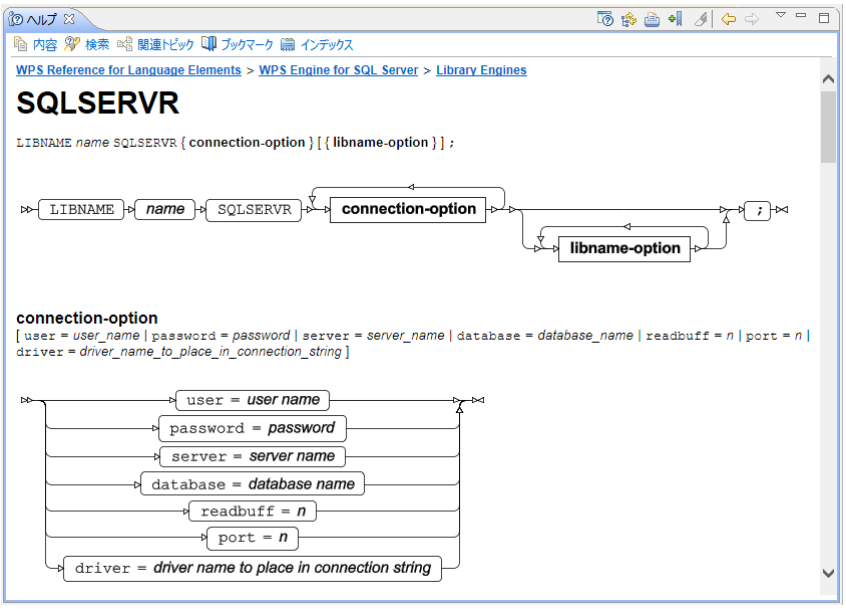


WPS レファレンスを展開し、WPS Engine for SQL Server を選択します。



Library Engines → SQLSERVR を選択すると、データベース接続オプション（connection-option）が表示されます。

[15] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。



これらの接続オプション（user= password= server= database= readbuff= port= driver=）は SQLSERVR ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプション指定としても、また、次に節で説明する libname ステートメントによる DB 接続オプションとしても用いることができ、必要に応じて指定します。

［libname ステートメントによる DB 入力］

WPS のエディタから以下のプログラムを入力し、実行します。

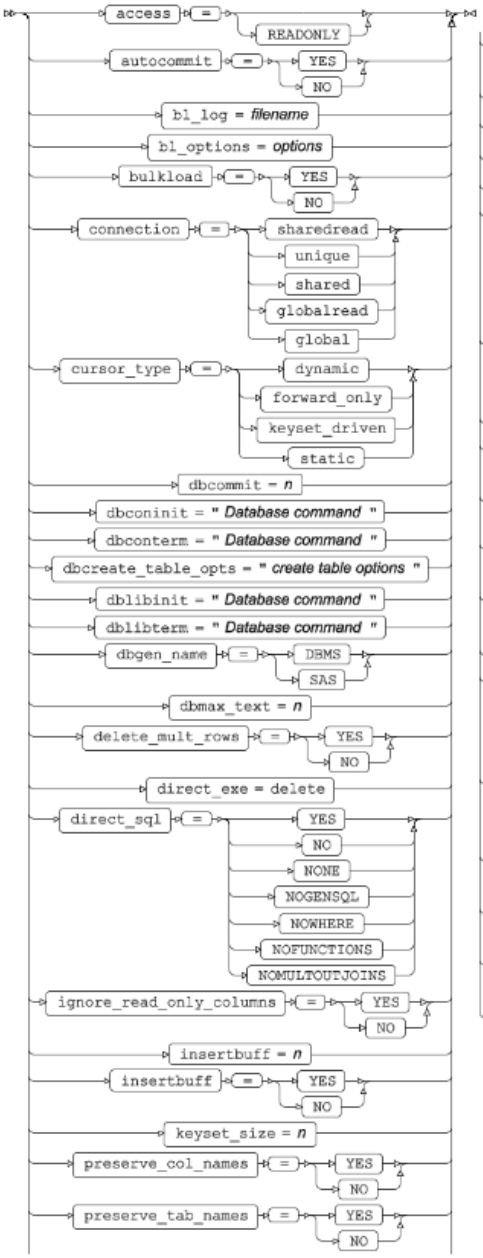
```
libname person SQLSERVR server="PC3¥SQLEXPRESS" database=AdventureWorks2012
schema=Person;
libname sales SQLSERVR server="PC3¥SQLEXPRESS" database=AdventureWorks2012
schema=Sales;
```

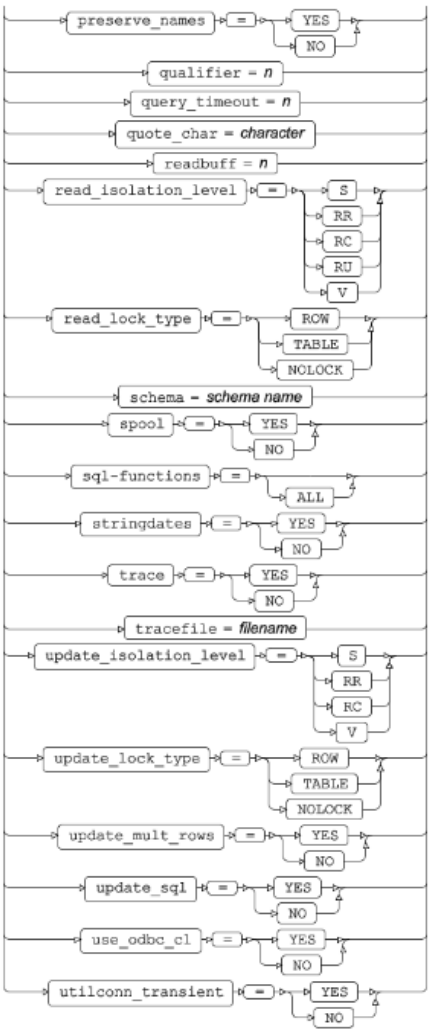
この指定は、WPS がデータベース AdventureWorks2012 に SQLSERVR ネイティブドライバー経由で接続し、Person スキーマ上のテーブルとビューをライブラリ名 person で、Sales スキーマ上のテーブルとビューをライブラリ名 sales で、それぞれ参照できるようにする指定です。（入出力両方可能） ただし、DB への接続やデータテーブルの検索に必要な権限の設定や、他の接続オプションが必要となる場合があります。

なお、server=サーバー名、database=DB 名指定は接続オプション、schema=指定は libname オプションです。libname オプションには他に、dbmax_text=n（文字テキストの読み取り最大長さの設定）などがあり、以下のような libname オプションが指定可能です。

libname-option

[access = [READONLY] | autocommit = [YES | NO] | bl_log = *filename* | bl_options = *options* | bulkload = [YES | NO] | connection = [sharedread | unique | shared | globalread | global] | cursor_type = [dynamic | forward_only | keyset_driven | static] | dbcommit = *n* | dbconinit = " *Database_command* " | dbconterm = " *Database_command* " | dbcreate_table_opts = " *create_table_options* " | dblibinit = " *Database_command* " | dblibterm = " *Database_command* " | dbgen_name = [DBMS | SAS] | dbmax_text = *n* | delete_mult_rows = [YES | NO] | direct_exe = delete | direct_sql = [YES | NO | NONE | NOGENSQL | NOWHERE | NOFUNCTIONS | NOMULTOUTJOINS] | ignore_read_only_columns = [YES | NO] | insertbuff = *n* | insertbuff = [YES | NO] | keyset_size = *n* | preserve_col_names = [YES | NO] | preserve_tab_names = [YES | NO] | preserve_names = [YES | NO] | qualifier = *n* | query_timeout = *n* | quote_char = *character* | readbuff = *n* | read_isolation_level = [S | RR | RC | RU | V] | read_lock_type = [ROW | TABLE | NOLOCK] | schema = *schema_name* | spool = [YES | NO] | sql-functions = [ALL] | stringdates = [YES | NO] | trace = [YES | NO] | tracefile = *filename* | update_isolation_level = [S | RR | RC | V] | update_lock_type = [ROW | TABLE | NOLOCK] | update_mult_rows = [YES | NO] | update_sql = [YES | NO] | use_odbc_cl = [YES | NO] | utilconn_transient = [YES | NO]]





Unsupported Options

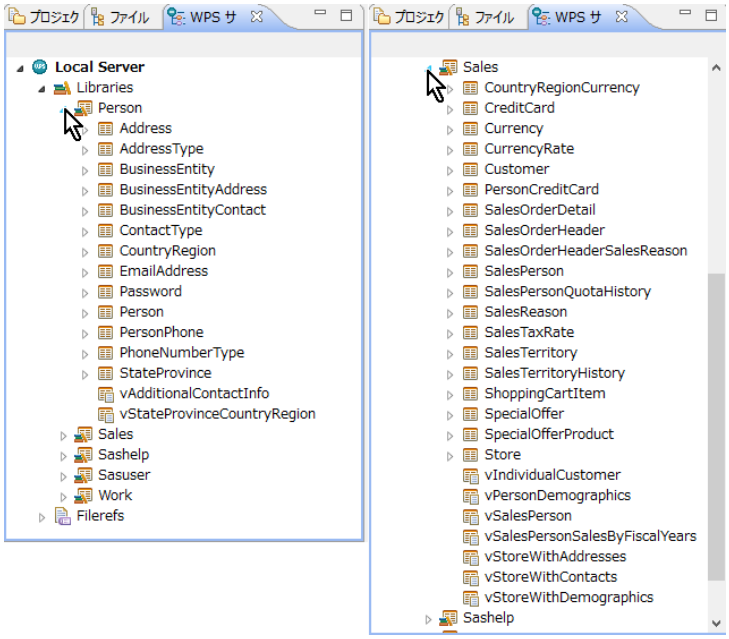
The following options are not yet supported:

- AUTHDOMAIN
- CONNECTION_GROUP
- DBINDEX
- DBMSTEMP
- DBNULLKEYS
- DBPROMPT
- DBSASLABEL
- DBSLICEPARM
- DEFER
- MULTI_DATASRC_OPT
- REREAD_EXPOSURE
- SQL_FUNCTIONS_COPY

さて、実行すると、ログに person と sales の 2 個の「ライブラリ名が割り当てられた」というメッセージが表示されます。このメッセージは SQLSERVR エンジン経由で DB と接続できたことを表します。

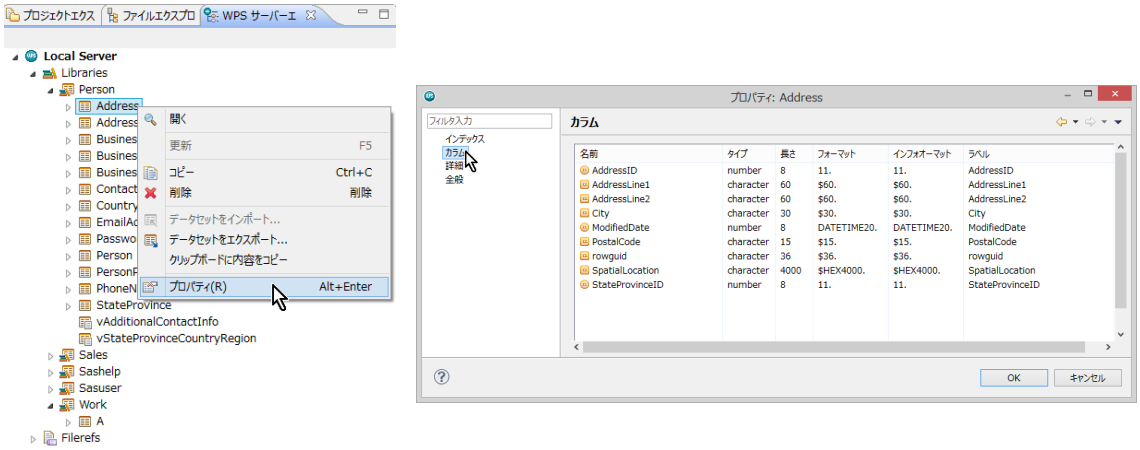
```
29       libname person SQLSERVR server="DMT-PC3¥SQLEXPRESS" database=AdventureWorks2012 schema
29     ! =Person;
NOTE: Library person assigned as follows:
      Engine:        SQLSERVR
      Physical Name: AdventureWorks2012

30       libname sales SQLSERVR server="DMT-PC3¥SQLEXPRESS" database=AdventureWorks2012 schema=
30     ! Sales;
NOTE: Library sales assigned as follows:
      Engine:        SQLSERVR
      Physical Name: AdventureWorks2012
```

実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、Person ライブラリと Sales ライブラリがあることを確認してください。これらの中に AdventureWorks2012 データベースの Person スキーマ、Sales スキーマ上のテーブル名およびビュー名がそれぞれ含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、付いていないのはビュー名です。

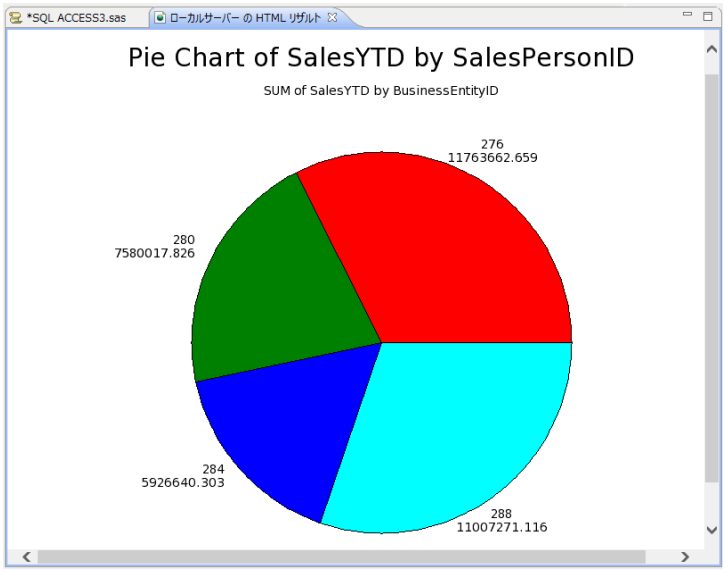
これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。



※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、SQLSERVR エンジンを libname ステートメントで指定した後は、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例では、Sales.Salesperson テーブルを入力としてビジネスエリア別の売り上げ高の円グラフを描いています。

```
goptions cback=white;
title "Pie Chart of SalesYTD by SalesPersonID";
proc gchart data=sales.salesperson;
  pie businessentityid/sumvar=salesYTD;
run;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを読むことができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、Person.Address テーブルから 3 つの都市名のいずれかに該当するレコードを選択した WPS データセット WORK.address_subset を DATA ステップで作成しています。

```
data address_subset;
  set person.address(keep=AddressID AddressLine1 City);
  where City in ('San Francisco', 'Los Angeles', 'Seattle');
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ
③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステートメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続に用いるエンジン名と接続先（DB 名やスキーマ名）を libname ステートメン

トで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで Person.Address テーブルから 3 つの都市名のいずれかに該当するレコードを選択した WPS データセット WORK.address_subset を SQL Server の Person スキーマ上の Address_Subset テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=address_subset dbms=SQLSERVR;
  server="PC3¥SQLEXPRESS";
  database=AdventureWorks2012;
  table=Person.Address_Subset;
  load;
run;quit;
```

※ 注意： DB にアクセスするため SQLSERVR エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で指定を追加してください。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントでエンジン名を指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は AdventureWorks2012 データベースの Sales.Customer テーブルを読んで、TerritoryID=1 に該当するレコードのみ抽出した全カラムを含むテーブル Sales.Customer_Territory_1 を作成しています。

```
libname sales SQLSERVR server="PC3¥SQLEXPRESS" database=AdventureWorks2012
schema=Sales;
proc sql;
  create table Sales.Customer_Territory_1 as
  select *
  from Sales.Customer
  where TerritoryID = 1;
quit;
```

※ この例では、Sales ライブラリは SQL Server DB の Sales スキーマを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の Sales ライブラリは WPS データライブラリではなく、AdventureWorks2012 DB の Sales スキーマを表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

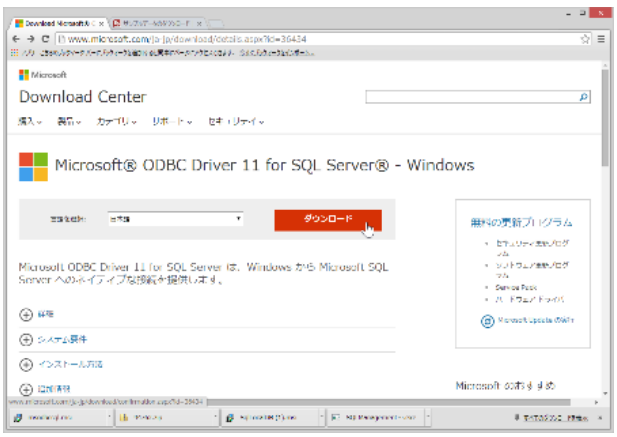
```
libname sales SQLSERVR server="PC3¥SQLEXPRESS" database=AdventureWorks2012
schema=Sales;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=sales;
  delete Customer_Territory_1;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data Sales.Customer_Territory_1;
  set Sales.Customer;
  where TerritoryID = 1;
run;
```

6.2. ODBC ドライバー経由の DB テーブルの入出力

ODBC ドライバーは、Windows 環境や Linux 環境から各種 DB とのインタフェースを共通方式で行うためのソフトウェアです。ほとんどの DB の ODBC ドライバーが DB ベンダやサードパーティから提供されており、ODBC ドライバーの設定を行えば、DB ごとの細かいアクセス条件等の設定を気にせずに、共通の構文で DB アクセスが実現できるという利点があります。

［ODBC ドライバのインストール］

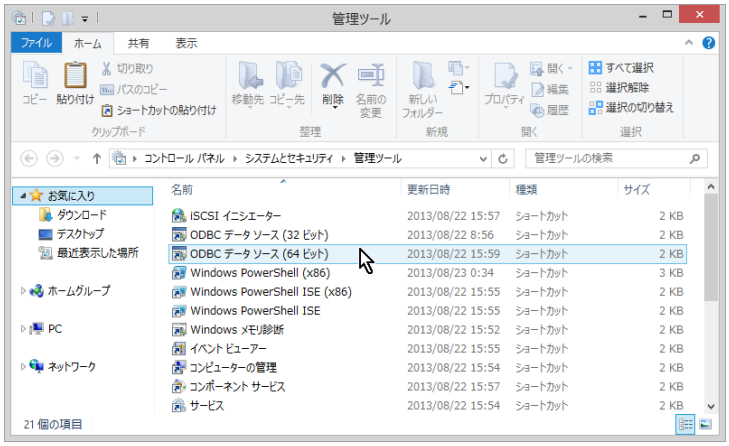
DB ベンダーのサイトから ODBC ドライバーをダウンロードし、インストールしておきます。ここでは、Microsoft のサイトから SQL Server の ODBC ドライバーをダウンロードしています。



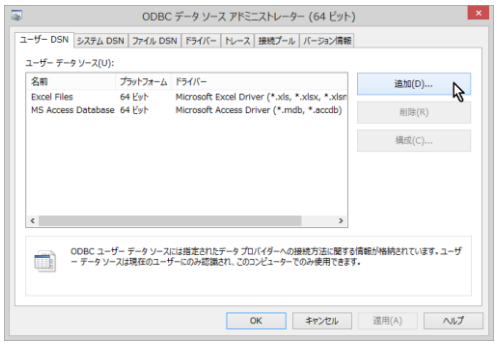
［PC 側の ODBC ドライバーの設定］

WPS が導入されている PC 側では、以下のような手順で、上記 SQL Server 2012 の AdventureWorks2012 データベースを ODBC データソースとして登録しておきます。

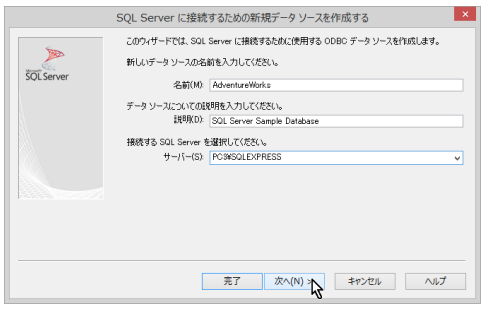
コントロールパネル → システムとセキュリティ → 管理ツールで ODBC データソースを選択します。

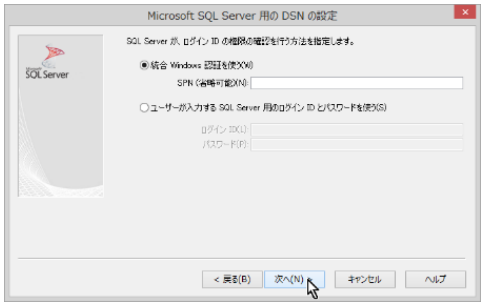


（左図）「追加」をクリックします。
（右図）ダウンロードした SQL Server 用 ODBC ドライバーを選択し、「完了」を押します。

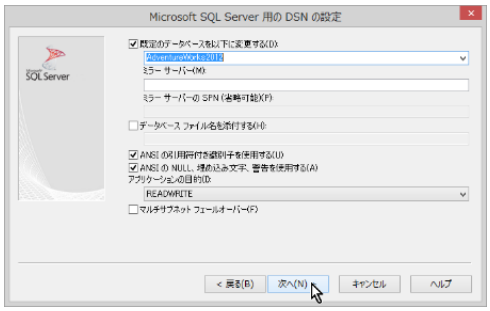


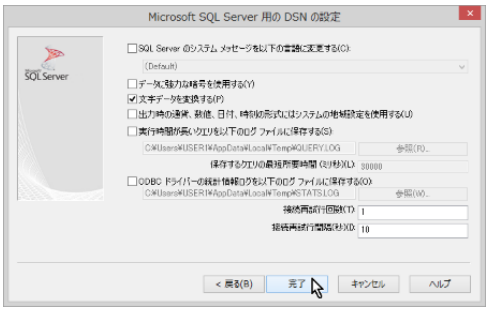
（左図）新規データソースの名前と説明にはユーザーが自由に入力できます。ここではわかりやすく、名前に「AdventureWorks」、説明には「SQL Server Sample Database」と入力を行いました。「接続する SQL Server を選択してください」には、SQL Server のサーバー名を入力します。「次へ(N)」をクリックします。
（右図）ログイン方法の設定を確認して「次へ(N)」をクリックします。



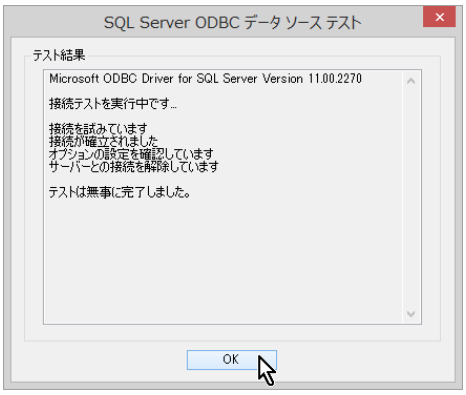


（左図）この画面で「既定のデータベースを以下に変更する」にチェックを入れて、アクセスしたいデータベース名（ここでは AdventureWorks2012）をプルダウンリストから選択し、「次へ(N)」をクリックします。
（右図）そのまま「完了」をクリックします。

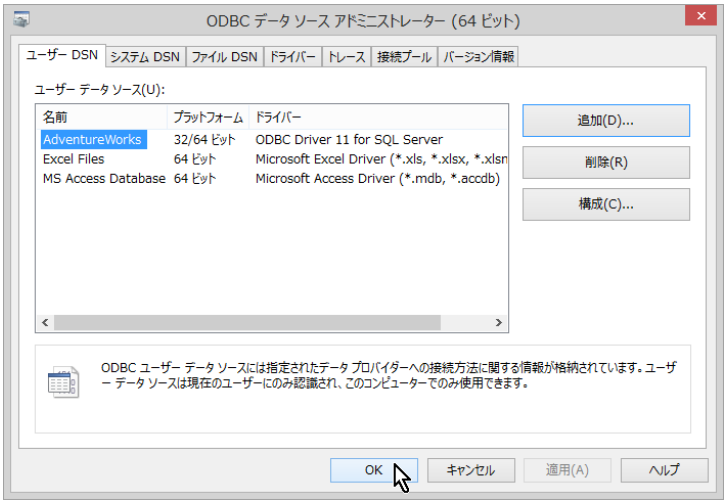




（左図）「データソースのテスト」をクリックし接続テストを行います。
（右図）テスト結果を確認し、「OK」をクリックし、戻ります。



AdventureWorks が ODBC データソースに追加されたことを確認し、「OK」を押して設定終了です。



[SQL 言語を使用した DB 読み取り]

SQL 言語を用いて WPS から ODBC ドライバー経由で SQL Server の DB アクセスを行う場合は、以下のように、エンジン名として ODBC を指定し、CONNECT 文のカッコ内に datasrc=ODBC データ ソース名などの接続オプションや dbmax_text=n（文字テキストの読み取り最大長さの設定）などの libname オプションを指定します。

以下は、ODBC ドライバーを用いて、AdventureWorks2012 データベースの Person.Address テーブルの 3 つのカラム（AddressID, AddressLine1, City）を City の値が 'San Francisco' か 'Los Angeles' か 'Seattle' のいずれかに該当するレコードを抽出し、AddressID の小さい方から並べた WPS データセット WORK.A を作成する例となっています。

```
proc sql;
  connect to ODBC(datasrc=AdventureWorks);
  create table A as
  select *
  from connection to ODBC
  (
    select AddressID, AddressLine1, City
    from Person.Address
    where City in ('San Francisco', 'Los Angeles', 'Seattle')
    order by AddressID
  );
  disconnect from ODBC;
quit;
```

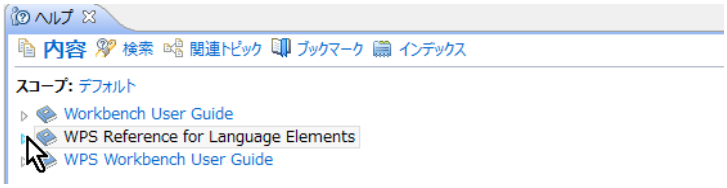
(WPS ワークベンチの実行ログ)

```
95        proc sql;
96          connect to ODBC(datasrc=AdventureWorks);
NOTE: Successfully connected to database ODBC as alias ODBC.
```

```
97          create table A as
98          select *
99          from connection to ODBC
100         (
101           select AddressID, AddressLine1, City
102           from Person.Address
103           where City in ('San Francisco', 'Los Angeles', 'Seattle')
104           order by AddressID
105         );
NOTE: Data set "WORK.A" has 285 observation(s) and 3 variable(s)
106         disconnect from ODBC;
NOTE: Successfully disconnected from database ODBC.
107       quit;
NOTE: Procedure sql step took :
      real time : 0.046
      cpu time  : 0.015
```

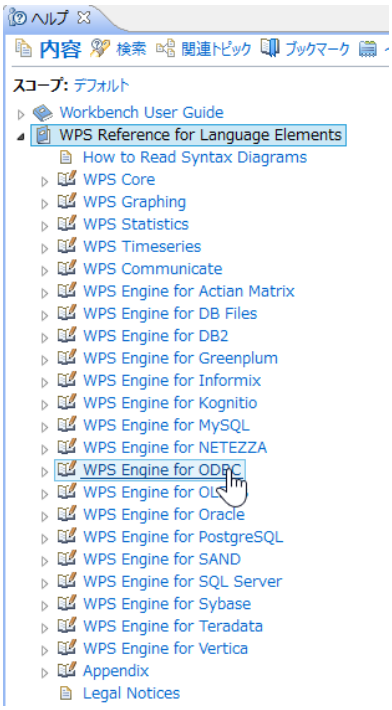
※ ODBC ドライバーによる DB へのアクセス制限等の設定によっては、SQL プロシジャの CONNECT 文の () 内のオプションに、その他の指定が必要となる場合があります。以下のようにヘルプビューで指定可能なオプションが確認できます。

ヘルプビュー[16]の左上にある「内容」をクリックします。
ヘルプ全体の目次が表示されます。WPS Reference for Language Elements をクリックします。

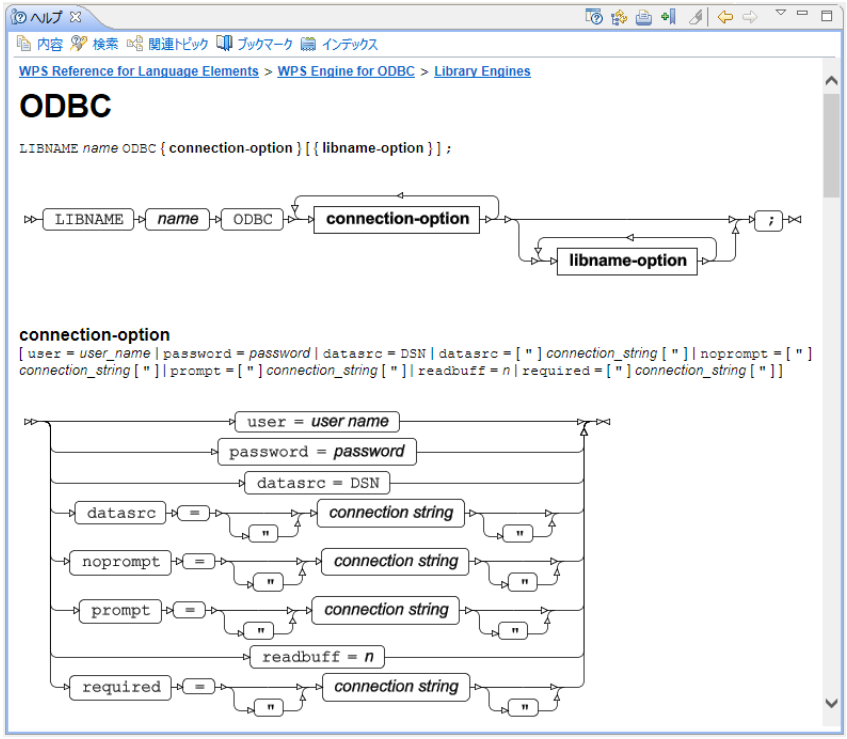


WPS レファレンスを展開し、WPS Engine for ODBC を選択します。

[16] ヘルプビューが存在しない場合は、「ウィンドウ(W)」→「ビューの表示(V)」→ 「その他(O)」→「ヘルプ」フォルダー → 「ヘルプ」からビューを出現させます。また、「ヘルプ(H)」→「ヘルプ目次(H)」から別ウィンドウで表示されるメニューでも同じです。



その後、展開を何度か行うと接続オプション（connection option） が表示されます。



これらのオプション（user= password= datasrc= noprompt= prompt= readbuff= required=）は ODBC ドライバー経由でのデータベースアクセスの際の CONNECT 文の DBMS オプシ

ョン指定としても、また、次に説明する libname ステートメントによるデータベースアクセスの接続オプションとしても用いることができ、必要に応じて指定します。

[libname ステートメントによる DB 入力]

WPS のエディタから以下のプログラムを実行します。

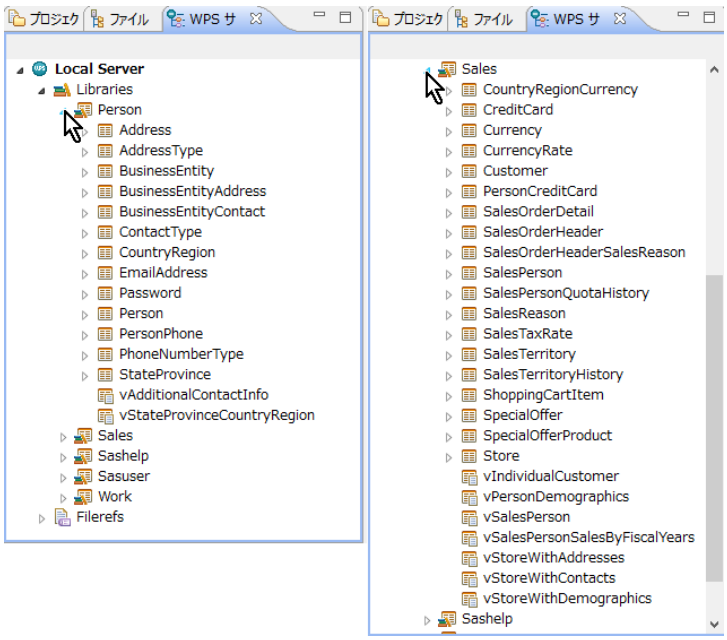
```
libname person ODBC datasrc=AdventureWorks schema=Person;
libname sales ODBC datasrc=AdventureWorks schema=Sales;
```

この指定は、WPS がデータソース AdventureWorks に ODBC ドライバー経由で接続し、Person スキーマをライブラリ名 person で、Sales スキーマのテーブルをライブラリ名 sales で、それぞれ参照できるようにする指定です。(入出力両方可能)
なお、datasrc=指定は DB 接続オプション、shema=指定は libname オプションです。
libname オプションには他に、dbmax_text=n (文字テキストの読み取り最大長さの設定) などがあります。指定可能な libname オプションは SQLSERVR エンジンの libname オプションとほぼ同じですが、詳細は前出の ODBC エンジンのヘルプ画面から確認してください。(Connection-option の下に libname-option が記載されています。)

実行すると、ログに person と sales 2 個の「ライブラリ名が割り当てられた」というメッセージが表示されます。このメッセージは ODBC 経由で DB と接続できたことを表します。

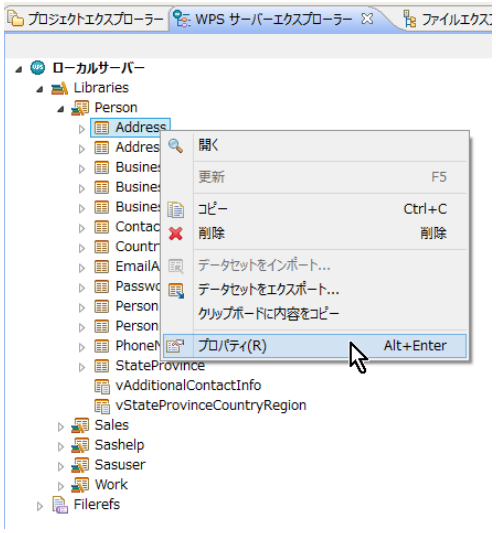
```
128      libname person ODBC datasrc=AdventureWorks schema=Person;
NOTE: Library person assigned as follows:
      Engine:        ODBC
      Physical Name: AdventureWorks

129      libname sales ODBC datasrc=AdventureWorks schema=Sales;
NOTE: Library sales assigned as follows:
      Engine:        ODBC
      Physical Name: AdventureWorks
```

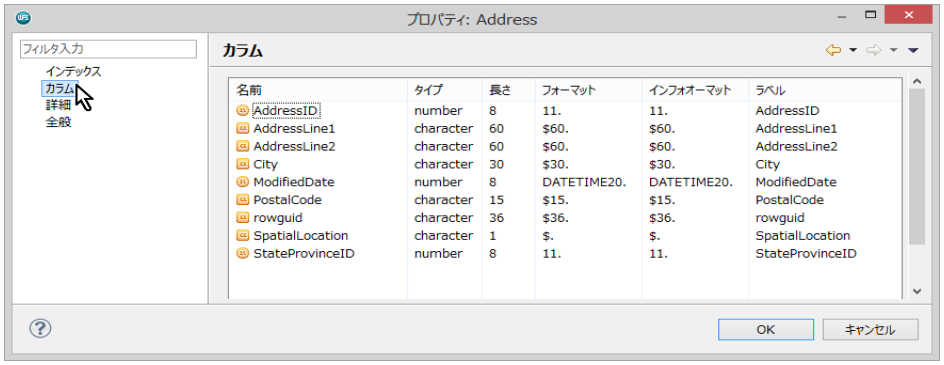
実行後、WPS サーバーエクスプローラービュー内の ローカルサーバー の下の Libraries を展開し、Person ライブラリと Sales ライブラリがあることを確認してください。これらの中に AdventureWorks2012 データベースの Person スキーマ、Sales スキーマ上のテーブル名およびビュー名がそれぞれ含まれていることがわかります。



※ 右矢印がついたアイテムはテーブル名（展開すると、テーブルに含まれる項目名と項目タイプを表すアイコンが表示されます。）、付いていないのはビュー名です。

これらのテーブルやビューは、WPS データセットに対する操作と同じファイル操作が可能です。例えば、以下のように、テーブル名を右クリックして出現するメニューの「プロパティ」を選択すると、カラム名などを確認することができます。

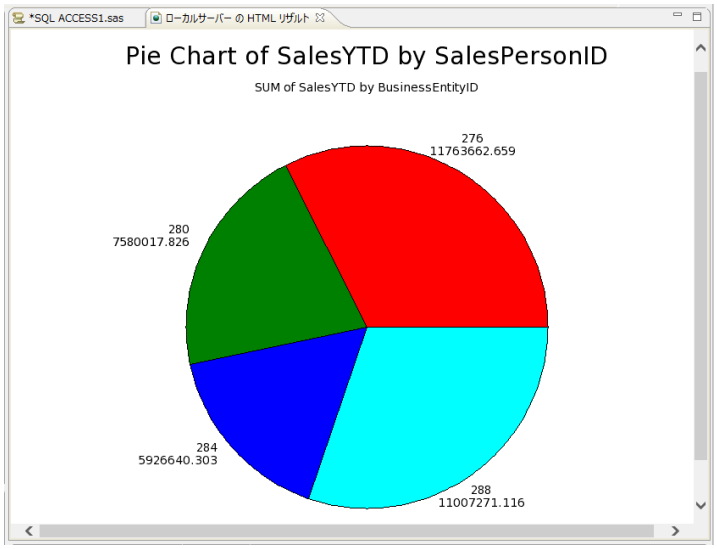




※ 【重要な注意】: テーブル名をダブルクリックすると、データセットビューアが起動し、テーブルの中身を表示しようとします。この操作はテーブルのサイズが大きいときは WPS セッションが動かなくなったり不意に終了してしまうなどの危険を伴いますので、できるだけ避けてください。

さて、ODBC アクセスエンジンを libname ステートメントで指定すると、以下のプログラムのように、DB テーブルやビューを WPS データセットに変換することなく、直接プロシジャ入力に用いることもできます。この例は、sales.salesperson テーブルを入力としてビジネスエリア別の売り上げ高の円グラフを描いています。

```
goptions cback=white;
title "Pie Chart of SalesYTD by SalesPersonID";
proc gchart data=sales.salesperson;
  pie businessentityid/sumvar=salesYTD;
run;
```



※ 注意：グラフィック表示は WPS3.1.1 では日本語の表示はサポートされていません。

また、以下のように、SAS 言語の DATA ステップでも DB テーブルを処理することができるようになります。また、SQL プロシジャを使用して処理することも可能ですので、libname ステートメントを使って DB アクセスを行うときは、SAS 言語と SQL 言語いずれも使用可能になるというメリットがあります。

下の例は、前の節で SQL プロシジャで作成した WORK.A と同じ内容のデータセット WORK.B を DATA ステップで作成しています。

```
data B;
  set person.address(keep=AddressID AddressLine1 City);
  where City in ('San Francisco', 'Los Angeles', 'Seattle');
run;
```

［WPS データセットの DB への出力方法］

以下のいずれかの方法で WPS データセットを DB のテーブルへ出力できます。

① DBLOAD プロシジャ
② SQL プロシジャ
③ DATA ステップ

①の方法では DB への接続方法を DBLOAD プロシジャで指定しますので、libname ステートメントで DB エンジンを指定する必要はありません。しかし、②と③の方法では、事前に DB への接続方法を libname ステートメントで指定しておく必要があります。

① DBLOAD プロシジャ

前節の最後のプログラムで Person.Address テーブルから 3 つの都市名のいずれかに該当するレコードを選択した WPS データセット WORK.A を SQL Server の Person スキーマ上の Address_Subset_A テーブルへ出力してみましょう。以下のようにプログラムを書いて実行します。

```
proc dbload data=A dbms=ODBC;
  datasrc=AdventureWorks;
  table=Person.Address_Subset_A;
  load;
run;
```

※ 注意：DB にアクセスするための権限や ODBC エンジンの接続オプションや libname オプションがさらに必要な場合は、load ステートメントの前に、オプション名=値; の形式で

指定を追加してください。

② SQL プロシジャによるテーブルへの読み書き

Libname ステートメントで ODBC エンジンを指定すると、DB テーブルを WPS データセットと同じように扱えるようになります。
以下の例は AdventureWorks2012 データベースの Sales.Customer テーブルを読んで、TerritoryID=1 に該当するレコードのみ抽出した全カラムを含むテーブル Sales.Customer_Territory_1 を作成しています。

```
libname sales ODBC datasrc=AdventureWorks schema=Sales;
proc sql;
  create table Sales.Customer_Territory_1 as
  select *
  from Sales.Customer
  where TerritoryID = 1;
quit;
```

※ この例では、Sales ライブラリは SQL Server DB の Sales スキーマを参照するよう最初の libname ステートメントで定義しています。したがって、SQL プロシジャの中の Sales ライブラリは WPS データライブラリではなく、AdventureWorks2012 DB の Sales スキーマを表すものと解釈され、読み取りも書き出しも DB テーブルが対象になります。

③ DATA ステップ

以下の例は②の SQL プロシジャを使った処理と同じことを SAS 言語の DATA ステッププログラムで行っています。

```
libname sales ODBC datasrc=AdventureWorks schema=Sales;
/* 同じ名前のテーブルを一旦削除 */
proc datasets lib=sales;
  delete Customer_Territory_1;
run;quit;
/* DATAステップでDBテーブルを読み書き */
data Sales.Customer_Territory_1;
  set Sales.Customer;
  where TerritoryID = 1;
run;
```

以上