Rocket software

Visual COBOL チュートリアル

OpenESQL アシスタントを利用したデータベースアクセス

1　目的

レガシーな COBOL 言語で開発された企業システムからデータベースのアクセスを行うには繁雑な技術を要するというイメージをお持ちの方も多いかもしれませんが、Visual COBOL を使えば驚くほど簡単に開発が可能です。一般的にリレーショナル・データベースのデータ操作には SQL を利用します。SQL は、COBOL と異なるデータベース言語となるため、COBOL コンパイラは SQL 文を解釈できません。
そこで、COBOL 上で SQL 文を利用する場合、EXEC SQL と END-EXEC で囲んだ部分のみに SQL 文を記述し、COBOL と区別させます。この SQL 文が埋め込まれたソースを Oracle などデータベースベンダが提供するプリコンパイラに渡すことで、SQL 文の部分をデータベースベンダが提供する API コールに展開した COBOL プログラムを生成します。プログラマーが当初作成するソースはこのソースではなくプリコンパイル展開する前のソースになるためデバッグ時のコードと作成時のコードに相違が生まれてしまいます。

Visual COBOL はプログラマーが実際にメンテナンスする埋め込み SQL 文が入ったソースを IDE に認識させるべく、プリコンパイルとコンパイルをシングルステップで処理できるようにしました。プリコンパイル、もしくはそれに相当する処理を Visual COBOL が内部的に処理します。
これにより、プリコンパイル後のソースは扱うことがなくなるため、このソースに関する考慮は不要となりました。

Visual COBOL はこのプリコンパイルを意識させない技術に関して技術や利用する DBMS に応じて柔軟に技術選択いただけるよういくつかのオプションを用意しています。その中でも今回紹介する OpenESQL は、製品搭載のプリプロセッサです。また、OpenESQL によるデータベースアクセスを行う場合、「OpenESQL アシスタント」という開発補助ユーティリティを利用できます。

このチュートリアルでは、OpenESQL アシスタントを使用して OpenESQL によるデータベースアクセスの方法を学びます。

2　前提

本チュートリアルは、下記の環境を前提に作成されています。

OS	Windows 11
COBOL 製品	Visual COBOL 11.0 Patch Update 01 for Eclipse
DBMS 製品	SQL Server 2022

また、SQL 実行やレコード確認のため、SQL Management Studio を使用します。

下記のリンクから事前にチュートリアル用のサンプルデータベースの SQL ファイルをダウンロードして、任意のフォルダに解凍しておいてください。

サンプルデータベースの SQL スクリプトダウンロード

内容

3 チュートリアルについて

3.1 データベースの準備

1) チュートリアル用データベースの作成

チュートリアルで使用するデータベースとサンプルテーブル、データのインポートを行う SQL を実行します。

① SQL Server Management Studio の実行

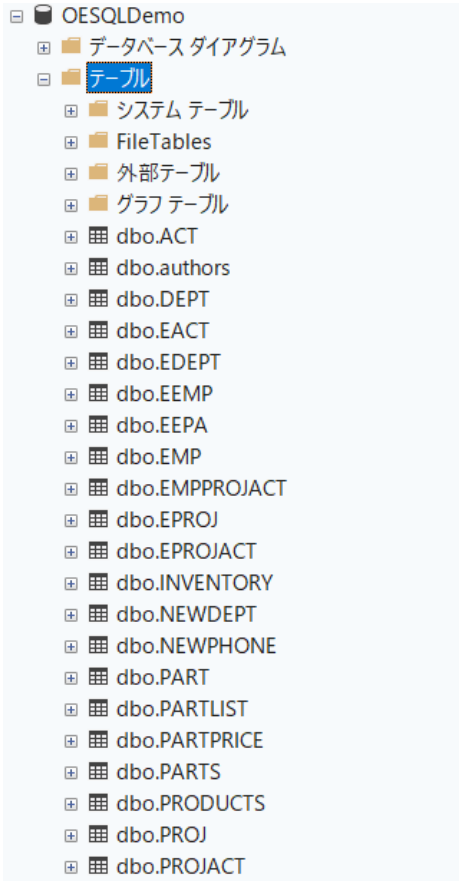
Microsoft SQL Server Management Studio を起動し、管理者でログインを行います。※Microsoft SQL Server Management Studio は別途ダウンロードが必要です。

② SQL スクリプトの実行

ダウンロードした SQL スクリプトの内容をコピー/ペーストしてクエリーを実行します。

③ 実行結果の確認

初めて実行する場合はデータベースが存在しないので drop database 文だけが失敗します。[オブジェクトエクスプローラー] ツリーにてデータベースとして「OESQLDemo」が作成されており、EMP テーブルが存在することを確認します。



④ SQL Server Management Studio の終了

閉じるボタンにて「Microsoft SQL Server Management Studio」を終了します。

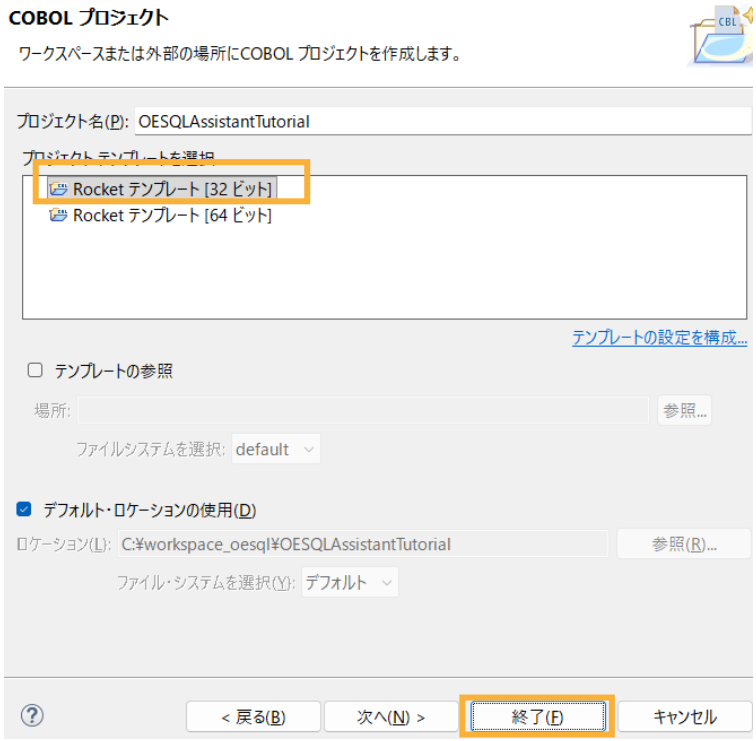
3.2 Eclipse プロジェクトの作成と設定

1) Visual COBOL for Eclipse の起動とプロジェクトの作成

① スタートメニューより [Micro Focus Visual COBOL] > [Visual COBOL for Eclipse] を選択します。

② ワークスペースの選択画面は任意のワークスペースを指定して、[起動(L)] ボタンをクリックします。

③ Eclipse メニューより、[ファイル(F)] > [新規(N)] > [COBOL プロジェクト] を選択し、プロジェクト名に “OESQLAssistantTutorial” を指定して、[終了(F)] ボタンをクリックします。



3.3 COBOL プログラムの作成とオプション設定

1) COBOL プログラムの作成

① COBOL エクスプローラーにて作成した「OESQLAssistantTutorial」プロジェクトを右クリックし、コンテクストメニューから [新規作成(N)] > [COBOL プログラム] を選択します。

② [含まれるプロジェクト] に 「OESQLAssistantTutorial」 が指定されていること、新規ファイル名が「Program1.cbl」になっていることを確認し、[終了(F)] ボタンをクリックします。

2) プロジェクトプロパティの設定

① プロジェクトの構成を変更します。

COBOL エクスプローラーにて作成した「OESQLAssistantTutorial」プロジェクトを右クリックし、コンテクストメニューから [プロパティ(R)] を選択します。

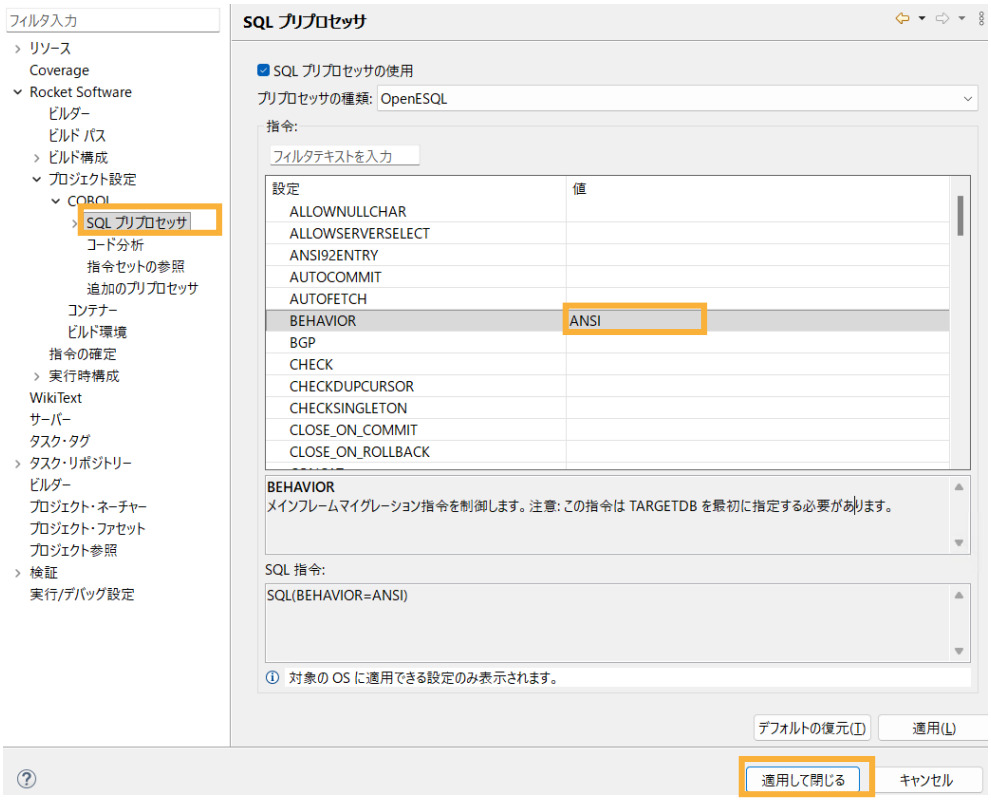
② プロパティ設定ダイアログが表示されます。

[Rocket Software] > [プロジェクト設定] > [COBOL] > [SQL プリプロセッサ] をクリックし、 以下の設定を行ったうえで、[適用して閉じる] をクリックします。

SQL プリプロセッサの使用： チェック

プリプロセッサの種類： “OpenESQL” を選択

BEHAVIOR: “ANSI” を選択



3) OpenESQL アシスタントの設定変更

① Eclipse メニューより [ウィンドウ(W)] > [設定(P)] を選択し、設定ダイアログを開き、[Rocket Software] > [データベース] > [OpenESQL アシスタント] をクリックします。

以下の設定を行ったのち、[適用して閉じる] をクリックします。

[DCLGEN] > [COBOL 変数の型] ： “SQL TYPE” を選択

[一般] > [モード] ： “ODBC” を選択

3.4 ODBC データソースの設定

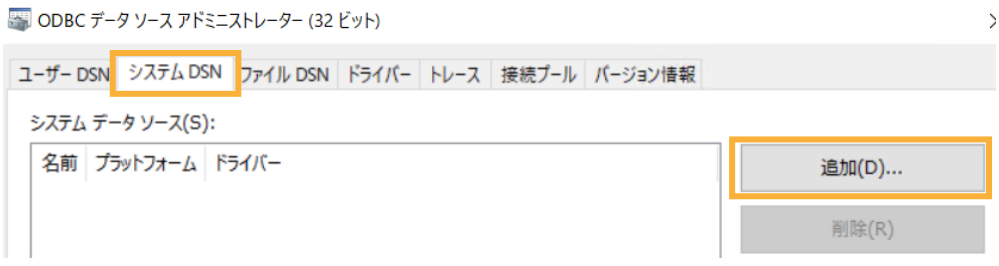
1) ODBC データソースアドミニストレーター（32 ビット）の起動

① スタートメニュー ＞ 検索にて ”ODBC” とタイプし、検索候補から「ODBC Data Sources（32 ビット）」を起動します。

2) データソースの設定と接続確認

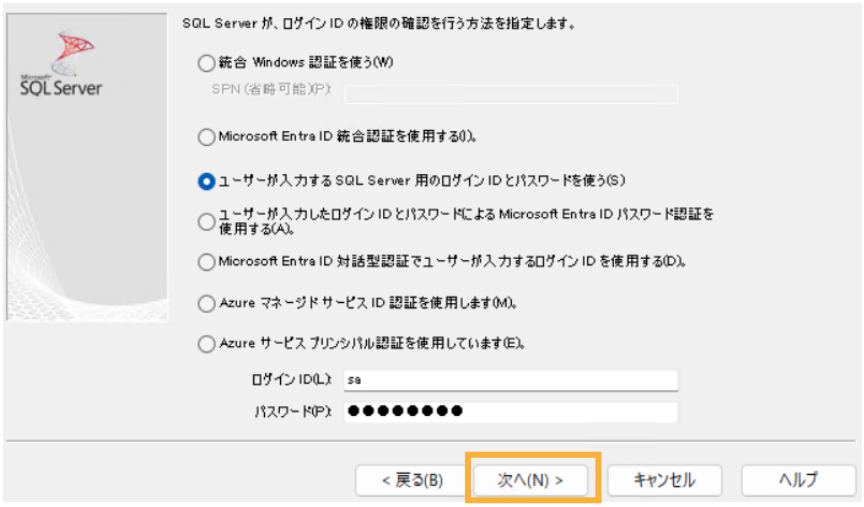
① ODBC データソースアドミニストレーター（32 ビット）画面より [システム DSN] タブをクリックし、[追加(D)…] ボタンをクリックします。

② [データソースの新規作成] ダイアログが表示されるので [ODBC Driver 18 for SQL Server] を選択し、[完了] ボタンをクリックします。

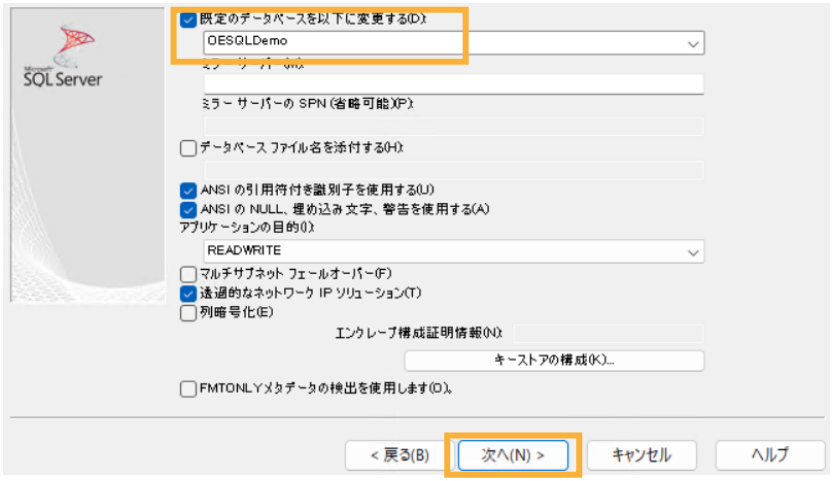


③ [Name] に “OESQLDemo” を入力し、[Server] に接続する SQL Server アドレスを指定し、[次へ(N)]をクリックします。もし、不明な場合は、自社の DBMS 管理者に確認してください。

SQL Server 接続のための認証方式を選択し [次へ(N)] ボタンをクリックします。不明な場合は、自社の DBMS 管理者に確認してください。（以下は SQL Server 認証の例です）



④ [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、作成したデータベース「OESQLDemo」を指定し、[次へ(N)] ボタンをクリックします。



⑤ [完了] ボタンをクリックして設定を保存します。

次の画面はそのまま [完了] をおします。

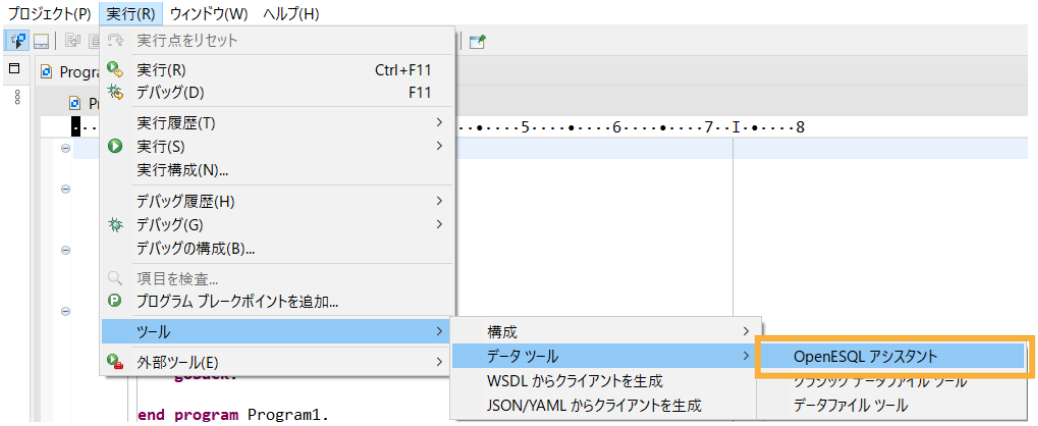
以下の画面では、[データソースのテスト] をクリックして接続確認を行ってください。

⑥ [OK] をクリックしたのち、ODBC データソースアドミニストレーター（32 ビット）画面をクローズします。

3.5 クエリーの組み立てとテスト

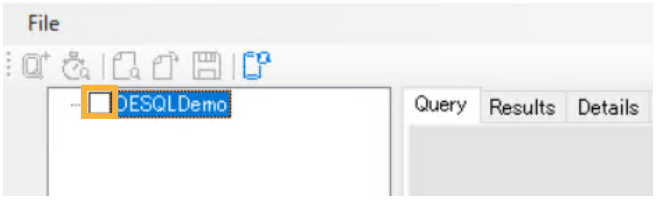
1) OpenESQL アシスタントの起動

Eclipse メニューより、[実行(R)] > [ツール] > [データツール] > [OpenESQL アシスタント] をクリックします。

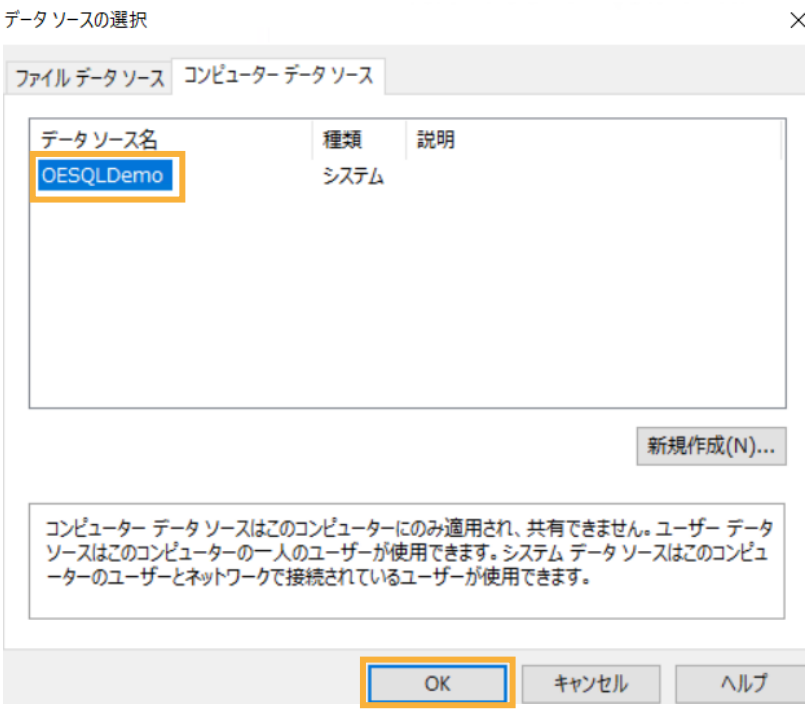


2) テーブルの選択

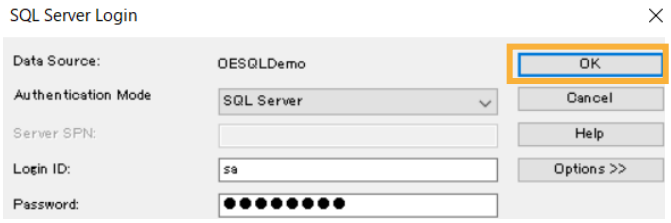
① OpenESQL アシスタントの画面より、表示されている [OESQLDemo] にチェックします。



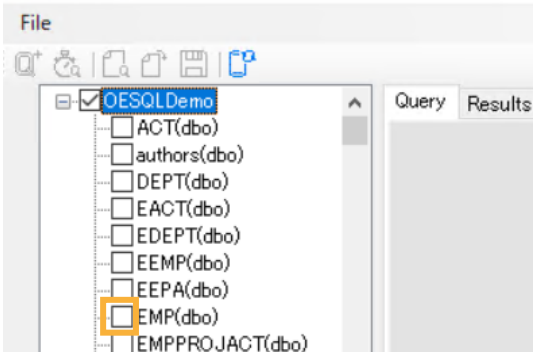
② データソースの選択画面が表示されるので 3.4 で設定した ODBC データソースを選択し、[OK] をクリックします。



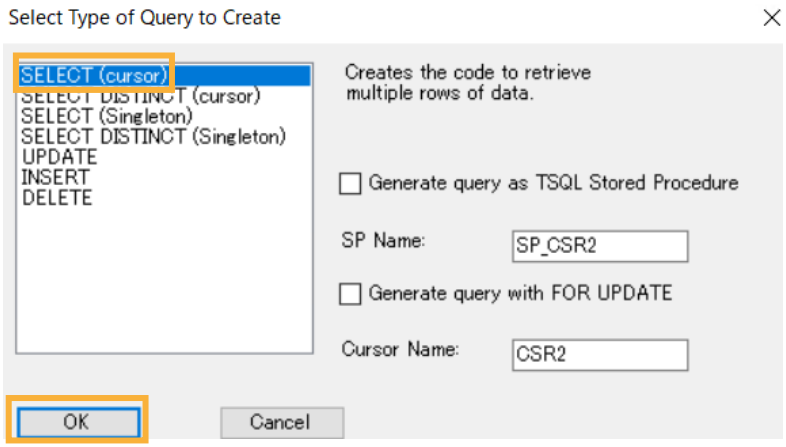
下記ダイアログが表示された場合は、データベースのログイン情報を入力した上で、[OK] をクリックします。



③ テーブルの一覧が表示されるので「EMP」にチェックします。



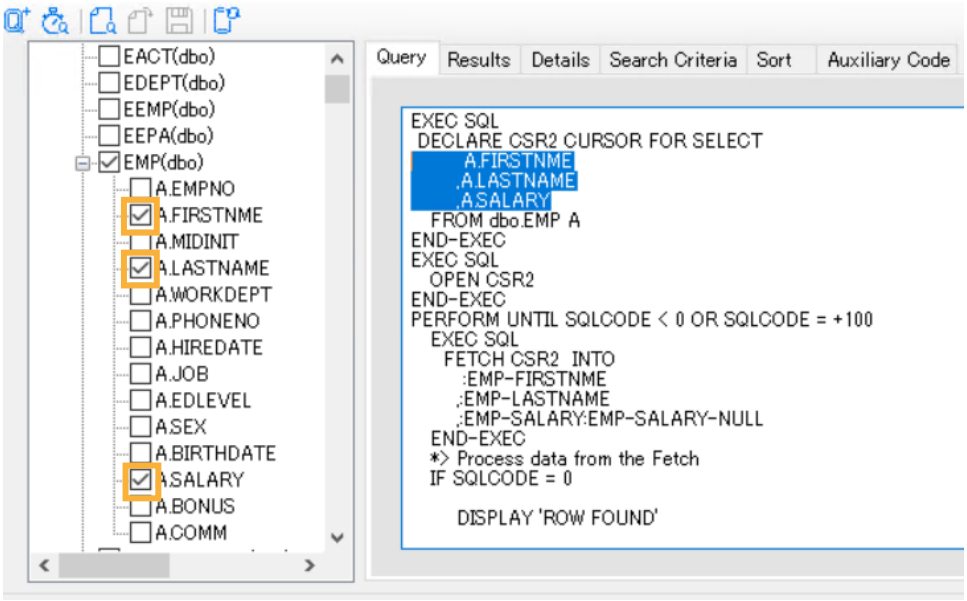
④ [生成するクエリーのタイプを選択] ダイアログでは、[SELECT (cursor)] を選択し、[OK] をクリックします。



[Query] タブにサンプルとなるコードが生成されます。

3) カラムの選択

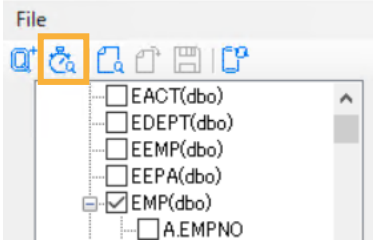
① EMP テーブルツリーから [A.FIRSTNME]、[A.LASTNAME]、[A.SALARY] にチェックします。[Query] タブのサンプルコードに選択したフィールドが付加されます。

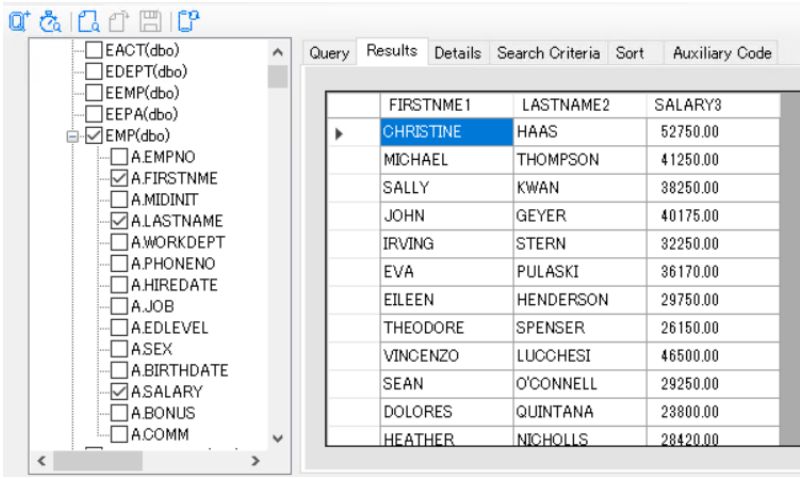


カラム名に “A.” が付与されていますが、これは上記画像の 6 行目にて EMP テーブルのエイリアスを “A” としているためです。複数テーブルを JOIN する場合、エイリアスが “B”, “C”, … と設定することができ、異なるテーブルに同名の項目が存在した場合でも、エイリアスを利用することで正しく取得先を識別することができます。

4) クエリーのテスト

① [クエリーの実行] アイコン をクリックすると、[Results] タブに実行結果が表示されます。

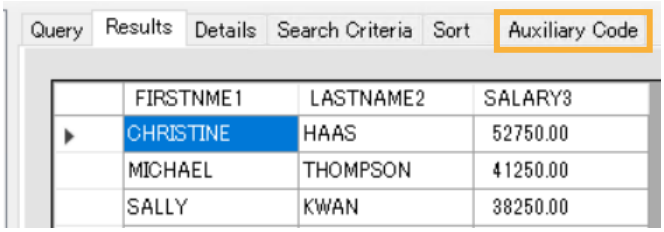




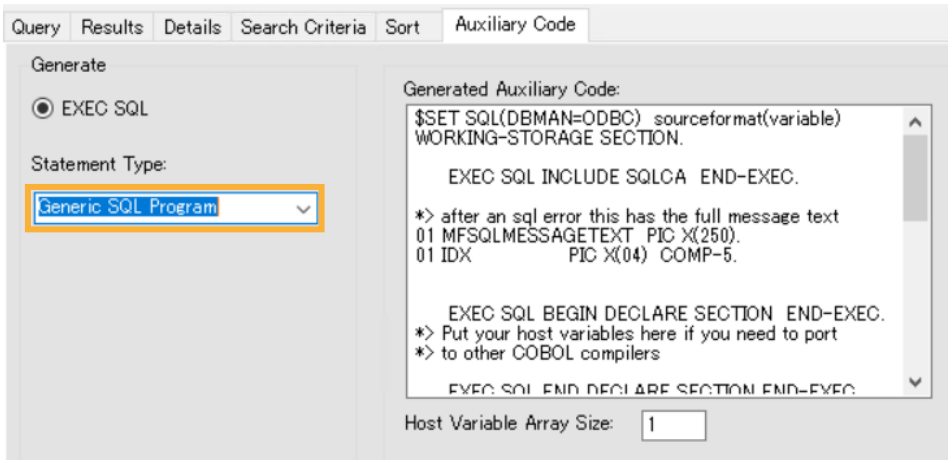
3.6 SQL プログラムの生成と埋め込み

1) SQL プログラムの生成

① OpenESQL アシスタントにて [Auxiliary Code] タブをクリックします。



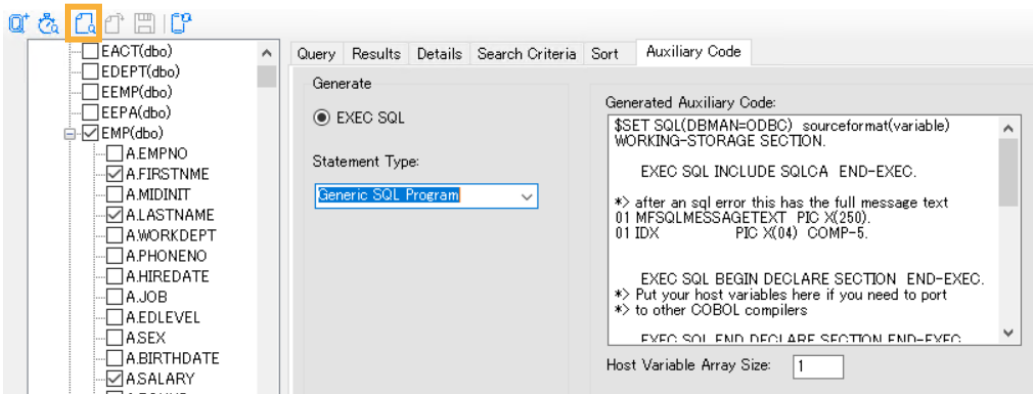
② [Statement Type] にて「Generic SQL Program」を選択します。生成された補助コード欄に生成されたソースコードが表示されます。



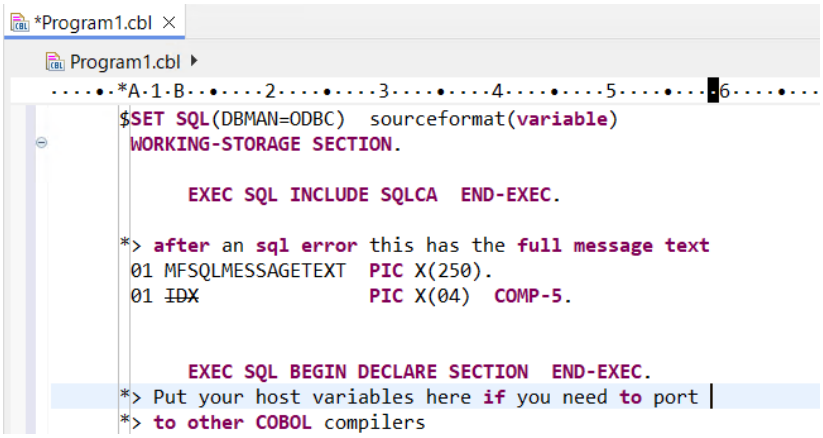
2) OpenESQL アシスタントが生成するプログラムコードを利用したデータベース接続プログラムの作成

① [Program1.cbl] をエディターで開き、現在のコードをすべて削除したうえで保存します。

② OpenESQL アシスタントにて [Auxiliary Code] タブをクリック選択し、[Add query to temporary file] アイコン をクリックし、クリップボードにソースコードをコピーします。



③ 「Program1.cbl」をエディター上で開き、Ctrl+V で生成された補助コードを貼りつけます。



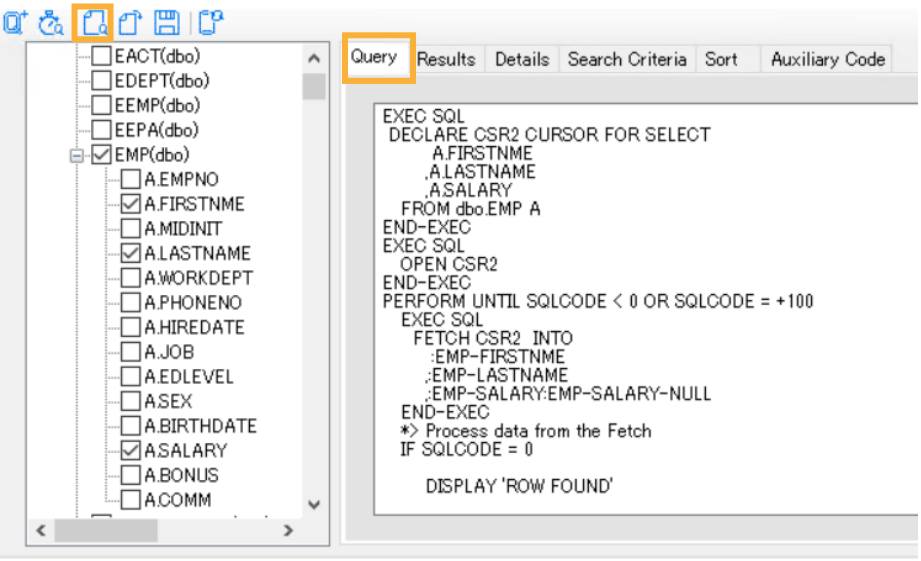
④ “CONNECT TO ‘OESQLDemo’ の末尾に “USER ‘データベースユーザーID.パスワード’ “ 句を記述します。以下の例では、データベースユーザーID に “sa”, パスワードに “HogeHoge” を指定しています。

```
EXEC SQL
    CONNECT TO 'OESQLDemo' USER 'sa.HogeHoge'
END-EXEC
```

補足）

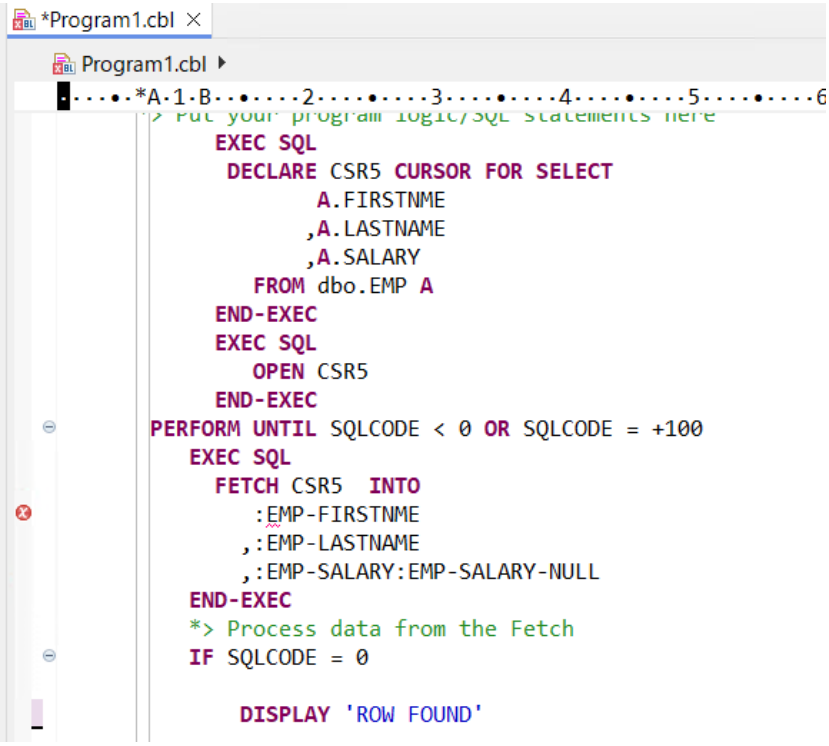
上記例は、SQL Server 認証が有効な場合の記述例です。環境に合わせ、データベースユーザーID にはデータベースにアクセス可能なログインユーザー、 パスワードにはデータベースユーザーID のパスワードを設定してください。なお、認証方式や情報については、自社のデータベース管理者にお問い合わせください。

⑤ OpenESQL アシスタントにて [Query] タブをクリックします。[Add query to temporary file] アイコンをクリックし、クリップボードにソースコードをコピーします。



⑥ Eclipse の COBOL エディターにて下記のコメント部分の下部に CTRL+V でコードを張り付けます。

“*> Put your program logic/SQL statements here”



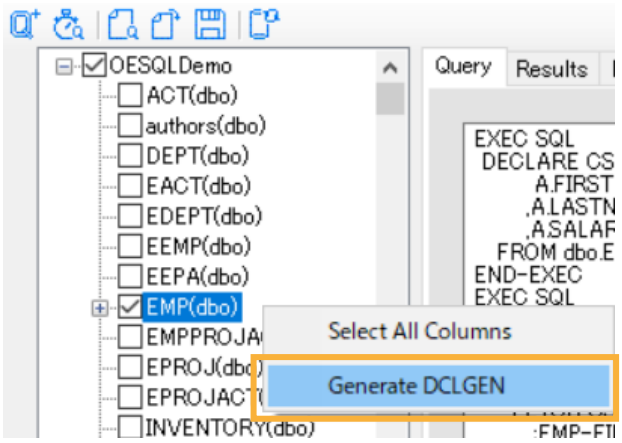
エラーが報告されますが、ここでは無視してください。

⑦ ファイルの変更を保存します。

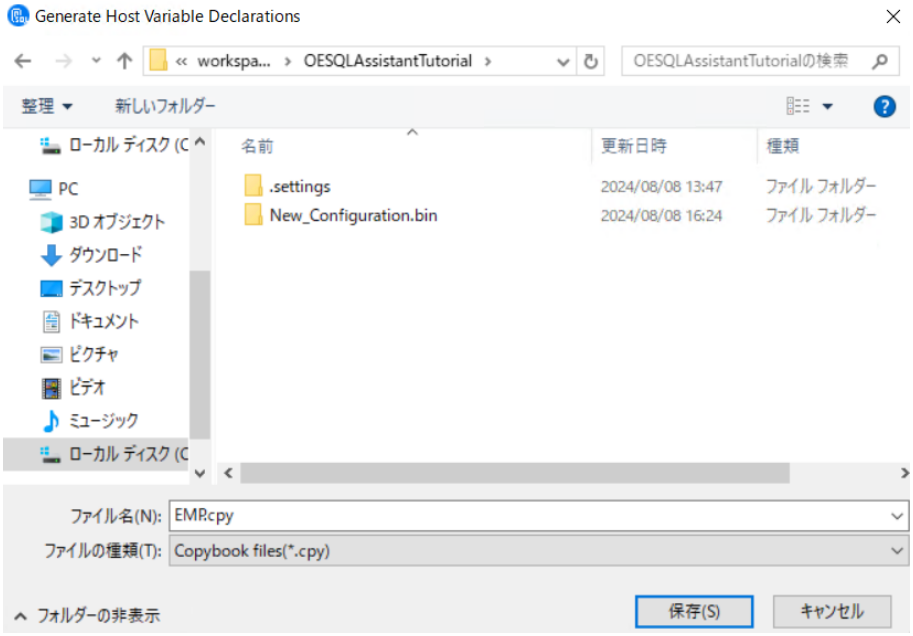
3.7 追加のコードを記述

1) コピーブックの生成

① OpenESQL アシスタントの「EMP」テーブルツリー内で選択した上で右クリックし、コンテクストメニューから [Generate DCLGEN] を選択します。

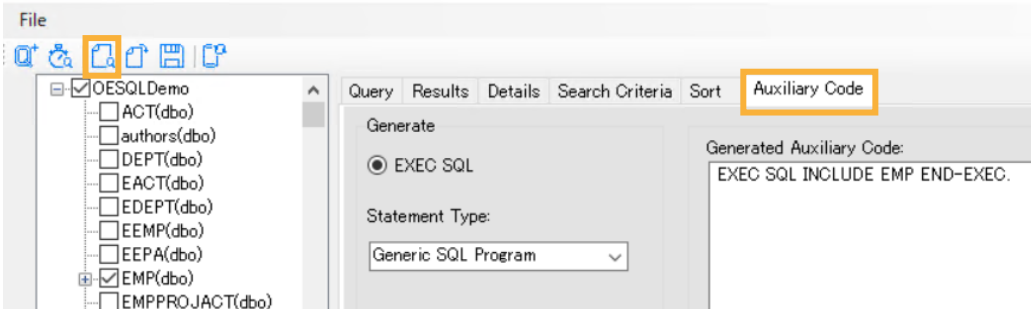


② [Generate Host Variable Declarations] ダイアログが表示されるので OESQLAssistantTutorial プロジェクトフォルダ配下まで移動した上で、"EMP.cpy" という名称でコピーブックを保存します。



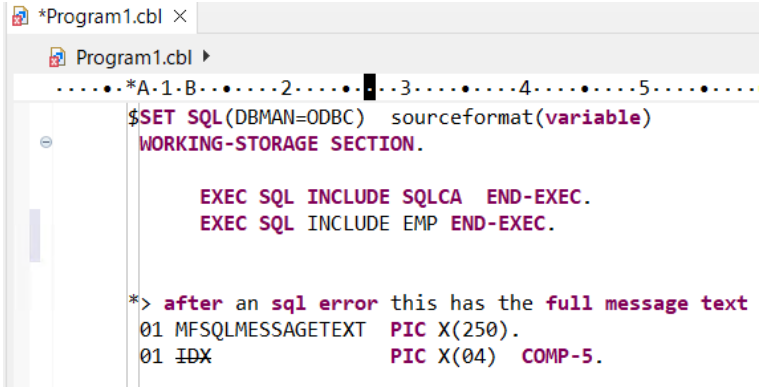
2) INCLUDE 命令の追加

① OpenESQL アシスタントにて [Auxiliary Code] タブをクリックします。[EXEC SQLINCLUDE] 文が生成されているので [Add file to temporary file] アイコン をクリックし、クリップボードにソースコードをコピーします。



② Eclipse の COBOL エディターにて EXEC SQL INCLUDE SQLCA に下に CTRL+V で貼り付けます。

③ Eclipse メニューより [ファイル(F)] > [保存(S)] を選択し、プログラムを保存します。



3) 表示フォーマットの追加

① Eclipse の COBOL エディターにて「EXEC SQL INCLUDE EMP END-EXEC」の下に下記の命令を追加します。

```
       01 EDIT-PAY      PIC Z,ZZ99.99.
```

② コードを下の方にスクロールして「DISPLAY 'ROW FOUND'」命令を以下の２行で置き換えます。

```
                MOVE EMP-SALARY TO EDIT-PAY
                DISPLAY 'NAME: ' EMP-LASTNAME ', ' EMP-FIRSTNME '    SALARY: ' EDIT-PAY
UPON CONSOLE
```

修正前）

```
PERFORM UNTIL SQLCODE < 0 OR SQLCODE = +100
   EXEC SQL
     FETCH CSR2  INTO
        :EMP-FIRSTNME
       ,:EMP-LASTNAME
       ,:EMP-SALARY:EMP-SALARY-NULL
   END-EXEC
   *> Process data from the Fetch
   IF SQLCODE = 0

       DISPLAY 'ROW FOUND'

   *> for array fetches, field sqlerrd(3) contains
   *> the number of rows returned
   *>   PERFORM VARYING IDX FROM 1 BY 1
   *>     UNTIL IDX > SQLERRD(3)
```

修正後）

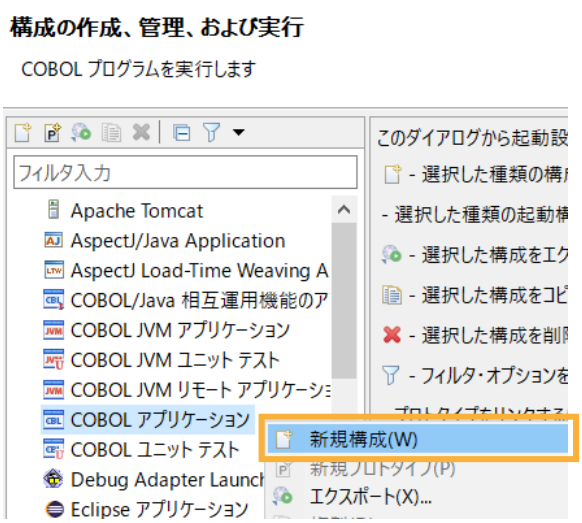
```
PERFORM UNTIL SQLCODE < 0 OR SQLCODE = +100
   EXEC SQL
     FETCH CSR2  INTO
        :EMP-FIRSTNME
       ,:EMP-LASTNAME
       ,:EMP-SALARY:EMP-SALARY-NULL
   END-EXEC
   *> Process data from the Fetch
   IF SQLCODE = 0

        MOVE EMP-SALARY TO EDIT-PAY
        DISPLAY 'NAME: ' EMP-LASTNAME ', ' EMP-FIRSTNME '   SALARY: ' EDIT-PAY UPON CONSOLE

   *> for array fetches, field sqlerrd(3) contains
   *> the number of rows returned
   *>   PERFORM VARYING IDX FROM 1 BY 1
   *>     UNTIL IDX > SQLERRD(3)
```

3.8 プログラムの実行

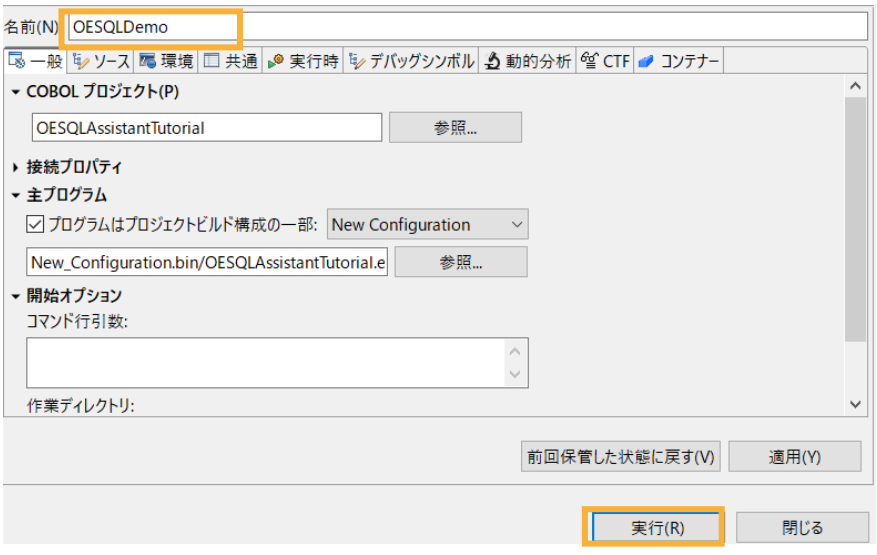
1) プログラムの実行

① Eclipse メニューより [実行(R)] > [実行構成(W)] をクリックします。

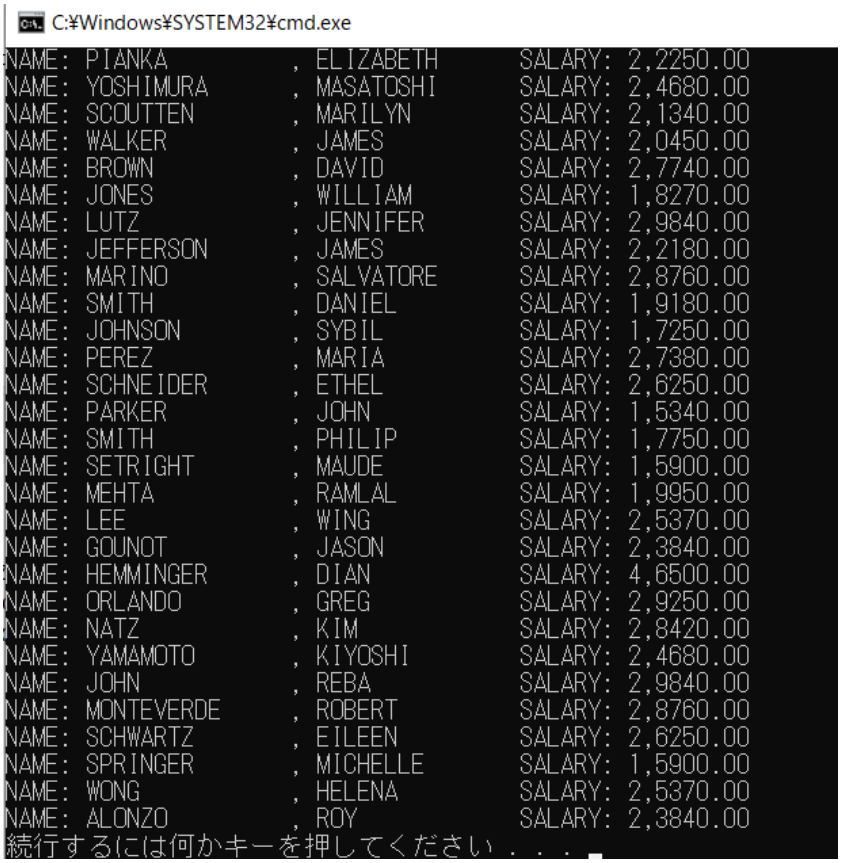
② [COBOL アプリケーション] を選択し、マウスの右クリックでコンテクストメニューを開き、 [新規構成(W)] を選択します。



③ [名前] に “OESQLDemo” を入力し、[実行(R)] をクリックします。



④ プログラムが実行され EMP テーブルの内容が出力されることを確認します。



何かキーをタイプすると、実行画面がクローズします。

免責事項

ここで紹介したソースコードは、機能説明のためのサンプルであり、製品の一部ではございません。ソースコードが実際に動作するか、御社業務に適合するかなどに関しまして、一切の保証はございません。 ソースコード、説明、その他すべてについて、無謬性は保障されません。
ここで紹介するソースコードの一部、もしくは全部について、弊社に断りなく、御社の内部に組み込み、そのままご利用頂いても構いません。
本ソースコードの一部もしくは全部を二次的著作物に対して引用する場合、著作権法の精神に基づき、適切な扱いを行ってください。