Oracle® Database

2 日で PHP 開発者ガイド

11*g* リリース 1（11.1）

部品番号 : E05696-01

2007 年 9 月

ORACLE®

Oracle Database 2 日で PHP 開発者ガイド , 11*g* リリース 1（11.1）

部品番号 : E05696-01

原本名 : Oracle Database 2 Day Plus PHP Developer's Guide, 11*g* Release 1 (11.1)

原本部品番号 : B28845-01

原本著者 : Simon Watt

原本協力者 : Christopher Jones、Simon Law、Glenn Stokol

目次

はじめに

このマニュアルでは、開発者が PHP を使用して Oracle Database にアクセスする方法について説明します。

ここでは、次の項目について説明します。

対象読者

このマニュアルは、PHP および Oracle Database を使用したアプリケーション開発の概要を示します。

このマニュアルは、対象読者が SQL、PL/SQL、および PHP について大まかに理解していることを想定しています。

ドキュメントのアクセシビリティについて

オラクル社は、障害のあるお客様にもオラクル社の製品、サービスおよびサポート・ドキュメントを簡単にご利用いただけることを目標としています。オラクル社のドキュメントには、ユーザーが障害支援技術を使用して情報を利用できる機能が組み込まれています。HTML 形式のドキュメントで用意されており、障害のあるお客様が簡単にアクセスできるようにマークアップされています。標準規格は改善されつつあります。オラクル社はドキュメントをすべてのお客様がご利用できるように、市場をリードする他の技術ベンダーと積極的に連携して技術的な問題に対応しています。オラクル社のアクセシビリティについての詳細情報は、Oracle Accessibility Program の Web サイト http://www.oracle.com/accessibility/ を参照してください。

ドキュメント内のサンプル・コードのアクセシビリティについて

スクリーン・リーダーは、ドキュメント内のサンプル・コードを正確に読めない場合があります。コード表記規則では閉じ括弧だけを行に記述する必要があります。しかし JAWS は括弧だけの行を読まない場合があります。

外部 Web サイトのドキュメントのアクセシビリティについて

このドキュメントにはオラクル社およびその関連会社が所有または管理しない Web サイトへのリンクが含まれている場合があります。オラクル社およびその関連会社は、それらの Web サイトのアクセシビリティに関しての評価や言及は行っておりません。

Oracle サポート・サービスへの TTY アクセス

アメリカ国内では、Oracle サポート・サービスへ 24 時間年中無休でテキスト電話（TTY）アクセスが提供されています。TTY サポートについては、(800)446-2398 にお電話ください。

関連ドキュメント

詳細は、次の Oracle ドキュメントを参照してください。

- 『Oracle Database 2 日で開発者ガイド』
- 『Oracle Database SQL 言語リファレンス』
- 『Oracle Database PL/SQL 言語リファレンス』
- 『SQL*Plus ユーザーズ・ガイドおよびリファレンス』
- 『Oracle Database グローバリゼーション・サポート・ガイド』

表記規則

このマニュアルでは、次の表記規則を使用します。

規則	意味
太字	太字は、操作に関連付けられている Graphical User Interface あるいは本文中または用語集で定義されている用語を示します。
イタリック体	イタリック体は、特定の値を指定する必要があるプレースホルダや変数を示します。
固定幅フォント	固定幅フォントは、段落内のコマンド、URL、コード例、画面に表示されるテキストまたは入力したテキストを示します。

サポートおよびサービス

次の各項に、各サービスに接続するための URL を記載します。

Oracle サポート・サービス

オラクル製品サポートの購入方法、および Oracle サポート・サービスへの連絡方法の詳細は、次の URL を参照してください。

```
http://www.oracle.co.jp/support/
```

製品マニュアル

製品のマニュアルは、次の URL にあります。

```
http://otn.oracle.co.jp/document/
```

研修およびトレーニング

研修に関する情報とスケジュールは、次の URL で入手できます。

```
http://www.oracle.co.jp/education/
```

その他の情報

オラクル製品やサービスに関するその他の情報については、次の URL から参照してください。

```
http://www.oracle.co.jp
http://otn.oracle.co.jp
```

注意： ドキュメント内に記載されている URL や参照ドキュメントには、Oracle Corporation が提供する英語の情報も含まれています。日本語版の情報については、前述の URL を参照してください。

1

Oracle Database での PHP の導入

PHP は、HTML に埋め込むことができることから、特に Web の開発で有効な人気の高いスクリプト言語です。Zend Core for Oracle によって、Oracle Database で PHP を使用してアプリケーションを開発できます。

この章の内容は次のとおりです。

Zend Core for Oracle

Zend Technologies と共同で開発した Zend Core for Oracle は、安定性が高く高性能でインストールが簡単な PHP の開発環境および本番環境であり、Oracle Database と完全に統合されています。

目的

このマニュアルは、Zend Core for Oracle を使用して Oracle Database に接続する方法および PHP を使用してデータにアクセスし、変更を行う方法を示すチュートリアルです。

Oracle での PHP 開発には、Zend Core 以外の PHP 環境を使用することもできます。

サンプル・アプリケーションの概要

このマニュアルでは、AnyCo Corp という架空の会社で人事管理（HR）サンプル・アプリケーションを開発する手順について説明します。

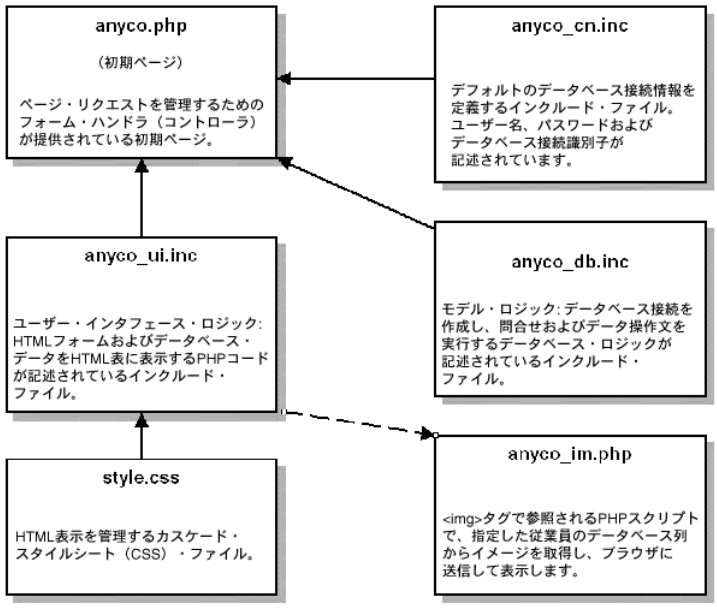
このアプリケーションでは、Oracle Database で用意されている HR スキーマの `DEPARTMENTS` 表および `EMPLOYEES` 表に格納されている部門データおよび従業員データを管理します。このスキーマの詳細は、『Oracle Database サンプル・スキーマ』を参照してください。

完全なサンプル・アプリケーションでは、次のことを行います。

- PHP OCI8 拡張モジュールを使用して、データベースへの接続を確立します。
- 部門データおよび従業員データをデータベースに問い合せます。
- データの表示およびデータ内のナビゲートを行います。
- 従業員レコードを挿入、更新および削除する方法を示します。
- データ例外を処理します。
- 従業員写真をアップロードおよび表示します。

図 1-1 に、このアプリケーション用に開発したファイル間の関係を示します。

図 1-1 人事管理サンプル・アプリケーションのコンポーネント



サンプル・アプリケーションのファイルは次のとおりです。

anyco.php: このファイルには、AnyCo アプリケーションのメイン・ロジックが含まれています。表示するページを判断する制御ロジックも含まれています。このファイルによって、ナビゲーション用のセッション・データが管理されます。また、anyco_cn.inc、anyco_db.inc および anyco_ui.inc の各インクルード・ファイルの関数がコールされます。

anyco_ui.inc: このファイルには、HTML ページでデータおよびフォームを提示するために使用される関数が含まれています。

anyco_cn.inc: このファイルには、データベース接続情報、データベース・ユーザー名、パスワードおよびデータベース接続識別子の定義が含まれています。

anyco_db.inc: このファイルには、データベース接続を作成し、問合せを実行し、データ操作文を実行するデータベース・ロジックが含まれています。

anyco_im.php: このファイルには、JPEG イメージとして表示するために、データベース列からイメージを取り出して Web ブラウザに送信するロジックが含まれています。

style.css: このファイルには、アプリケーションで生成される様々な HTML タグのカスケード・スタイルシート（CSS）定義が含まれています。このファイルによって、アプリケーションのルック・アンド・フィールが管理されます。

接尾辞が .inc のファイルは、他の PHP ファイルにインクルードされる PHP コード・ファイルです。

接尾辞が .php のファイルは、Web ブラウザにロードできます。

PHP アプリケーション・ソース・ファイルは、PHP 開発をサポートするツール（テキスト・エディタなど）で作成および編集できます。

各章で示すコードは、その前章で作成したファイルに基づいています。

リソース

次の OTN（Oracle Technology Network）の Web サイトでは、ユーザーにとって有効な追加情報が提供されています。

- PHP Developer Center

  http://www.oracle.com/technology/tech/php/index.html

- Zend Core for Oracle Developer Center

  http://www.oracle.com/technology/tech/php/zendcore/index.html

- Oracle Database ドキュメント・ライブラリ

  http://www.oracle.com/technology/documentation

- Oracle SQL Developer Center

  http://www.oracle.com/technology/products/database/sql_developer/

2

事前準備

この章では、Oracle Database および PHP 環境をインストールし、テストする方法について説明します。内容は次のとおりです。

必要なもの

Oracle Database および PHP 環境をインストールするには、次のものが必要です。

- Oracle Database Server
- Oracle Database Client
- Zend Core for Oracle
- PHP コードを編集するためのテキスト・エディタ。オプションの PHP 拡張モジュールに対応しているコード・エディタ（Oracle JDeveloper など）も使用できます。

Oracle Database のインストール

Oracle Database Server のコピーをコンピュータにインストールする必要があります。このチュートリアルで使用するサンプル・データは、デフォルトでインストールされます。これがサンプル・スキーマの HR コンポーネントとなります。

このチュートリアルでは、Oracle SQL Developer がデータベース・タスクの実行に使用するグラフィカル・ユーザー・インタフェースとなります。Oracle SQL Developer は、データベース開発用の無料のグラフィカルなツールです。

参照 :

- HR サンプル・スキーマの詳細は、『Oracle Database サンプル・スキーマ』を参照してください。
- Oracle SQL Developer の Web ページ

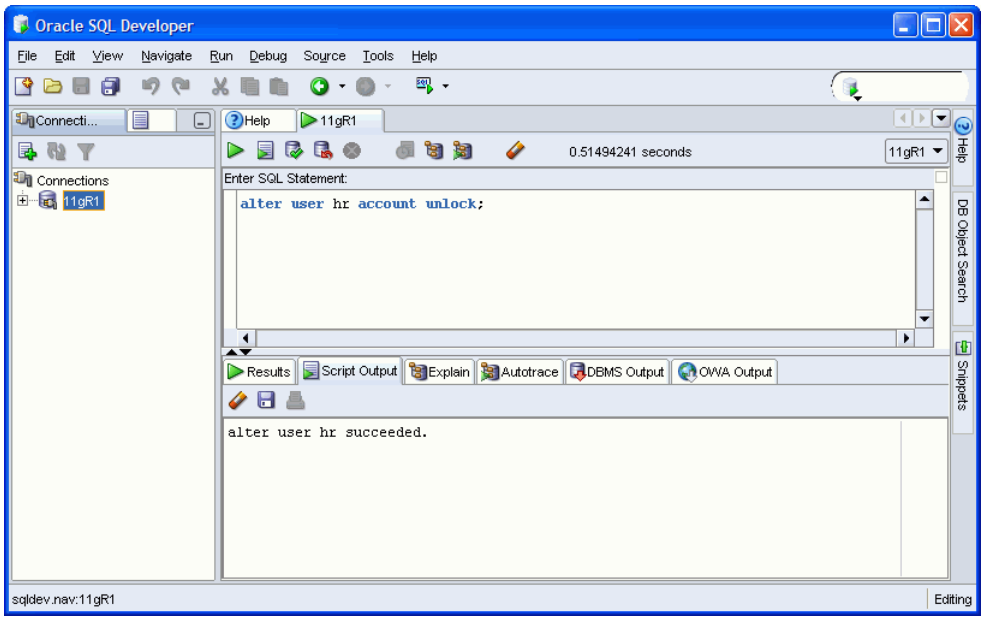
  `http://www.oracle.com/technology/products/database/sql_developer/`

HR ユーザーのロック解除

PHP アプリケーションは、HR ユーザーとしてデータベースに接続します。DBA 権限を持つユーザーとして、HR アカウントをロック解除する必要がある場合があります。HR ユーザーをロック解除するには、次の手順を実行します。

1. SQL Developer を起動し、Oracle データベースへの接続をオープンします。
2. **system** ユーザーとして Oracle データベースにログインします。
3. SQL ワークシートまたは SQL*Plus を開き、次の SQL 文を実行します。

```
alter user hr account unlock;
```



Oracle Database アカウントのロック解除の詳細は、『Oracle Database 2 日でデータベース管理者』の第 6 章「ユーザーおよびセキュリティの管理」を参照してください。

参照 :

- Oracle Database ドキュメント

  `http://www.oracle.com/technology/documentation`

Apache HTTP サーバーのインストール

Zend Core for Oracle には、Apache HTTP サーバーが含まれています。Apache をインストールするには、Zend Core インストール手順を実行します。

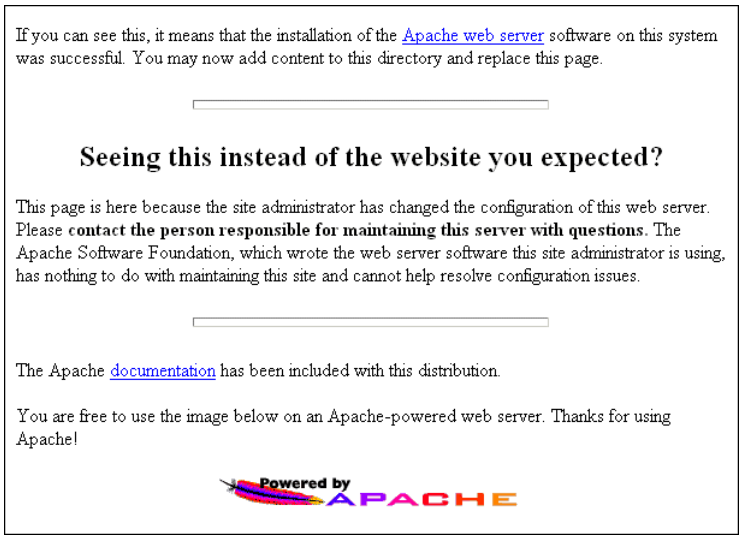
Windows での Apache インストールのテスト

Apache HTTP サーバーのインストールをテストするには、次の手順を実行します。

1. Apache をインストールしたホストで Web ブラウザを起動します。
2. 次の URL を入力します。

   `http://localhost/`

Web ブラウザに、次のようなページが表示されます。



このページが表示されない場合は、Apache 構成を確認します。一般的な問題としては、Apache が実行されていない、Apache がデフォルト以外のポートをリスニングしているなどの問題が考えられます。

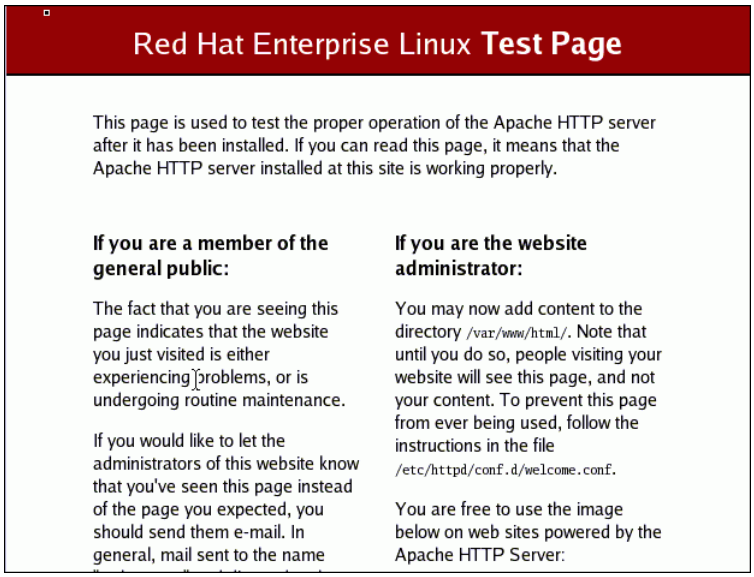
Linux での Apache インストールのテスト

Apache HTTP サーバーのインストールをテストするには、次の手順を実行します。

1. Apache をインストールしたホストで Web ブラウザを起動し、次の URL を入力します。

   ```
   http://localhost/
   ```

   Web ブラウザに、次のようなページが表示されます。



このページが表示されない場合は、Apache 構成を確認します。一般的な問題としては、Apache が実行されていない、Apache がデフォルト以外のポートをリスニングしているなどの問題が考えられます。

2. PHP ファイルにアクセスできるようにするために、デフォルトの Apache HTTP サーバー構成ファイルにパブリック仮想ディレクトリを public_html と設定します。任意のエディタを使用して Apache 構成ファイル /etc/httpd/conf/httpd.conf（Linux にインストールした場合は、ディレクトリが異なることがあります）を開き、次の行の先頭にあるシャープ記号（#）を削除します。

   この例では、Apache httpd.conf ファイルに次の行が含まれています。

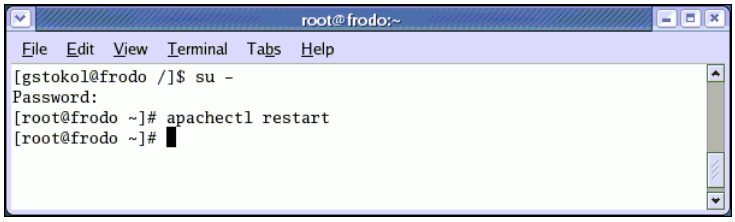
```
<IfModule mod_userdir.c>
    #
    # UserDir is disabled by default since it can confirm the presence
    # of a username on the system (depending on home directory
    # permissions).
    #
    #UserDir disable

    #
    # To enable requests to /~user/ to serve the user's public_html
    # directory, remove the "UserDir disable" line above, and uncomment
    # the following line instead:
    #
    UserDir public_html
</IfModule>
```

   これによって、Web ブラウザで、システムの登録ユーザーを使用して HTTP リクエストを作成し、そのユーザーの $HOME/public_html ディレクトリにあるファイルを処理できるようになります。次に例を示します。

```
http://localhost/~user/
```

3. 新しい Apache 構成ファイルを使用するには、コマンド・ウィンドウに次のコマンドを入力して Apache を再起動します。

```
su -
Password: <enter your su (root) password>
apachectl restart
```



   Apache HTTP サーバーが起動しない場合は、エラー・ログ・ファイルを調べて原因を特定します。構成エラーである可能性があります。

4. コマンド・ウィンドウに次のコマンドを入力して（root 以外で）ログインし、$HOME ディレクトリに public_html サブディレクトリを作成します。

```
mkdir $HOME/public_html
```

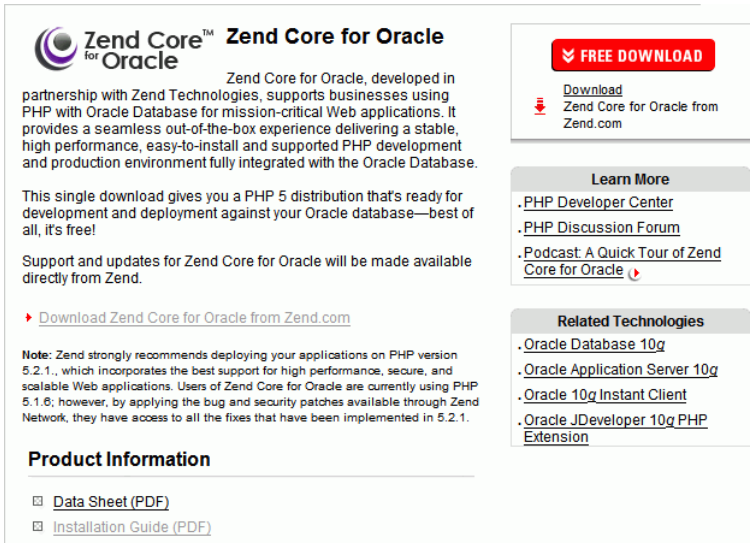
Zend Core for Oracle のインストール

Windows または Linux 用の Zend Core for Oracle を取得するには、次の手順を実行します。

1. Web ブラウザに次の URL を入力します。

   ```
   http://www.oracle.com/technology/tech/php/zendcore/index.html
   ```

2. Zend Core for Oracle の Web ページの右側で、「**Free Download**」ボタンをクリックします。



3. ダウンロードしたファイルを一時ディレクトリ（Windows の場合は `c:¥tmp`、Linux の場合は `\tmp`）に保存します。

Windows での Zend Core for Oracle のインストール

この項では、Windows に Zend Core for Oracle をインストールする方法について説明します。

このチュートリアルは、Zend Core for Oracle での PHP に固有です。

Zend Core for Oracle の詳細な設定情報は、次の URL の Zend Core for Oracle の Web ページの「Product Information」にあるインストレーション・ガイドを参照してください。

```
http://www.oracle.com/technology/tech/php/zendcore/index.html
```

この手順では、Zend Core for Oracle ソフトウェアを `c:\tmp` にダウンロードしたと想定します。そうでない場合は、手順 1 で、ダウンロードしたソフトウェアが格納されているディレクトリに移動する必要があります。

ファイル名および抽出ディレクトリは、現行バージョンに基づいて決定されます。この手順では、インストールしているバージョンのディレクトリ名を使用するようにしてください。

Zend Core for Oracle をインストールするには、管理者ユーザーである必要があります。Zend Core for Oracle をインストールするには、次の手順を実行します。

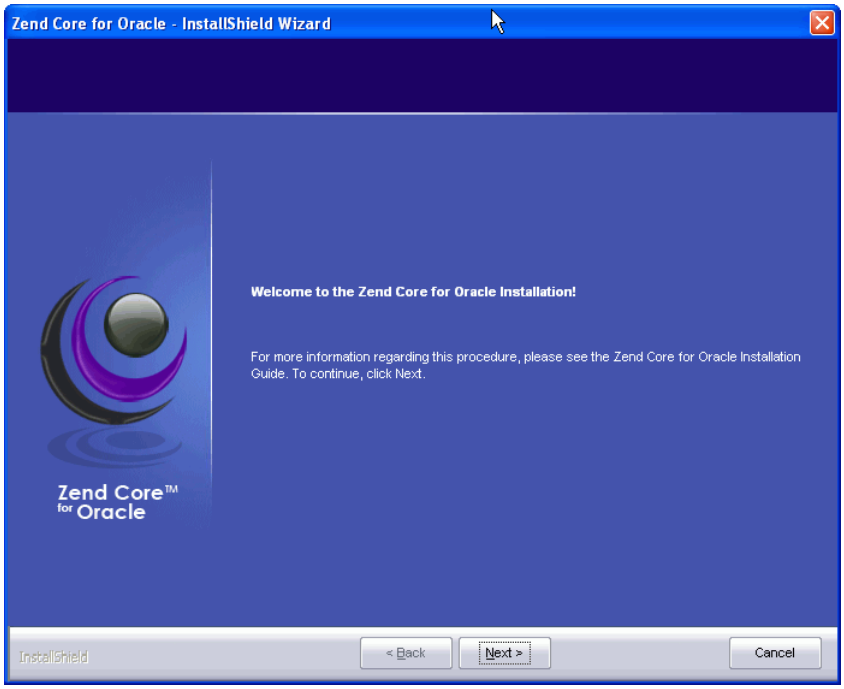
1. Windows のエクスプローラで、Zend Core for Oracle ソフトウェアをダウンロードしたディレクトリに移動します。

2. Zend Core for Oracle インストール・プロセスを開始するには、`.exe` ファイルをダブルクリックします。

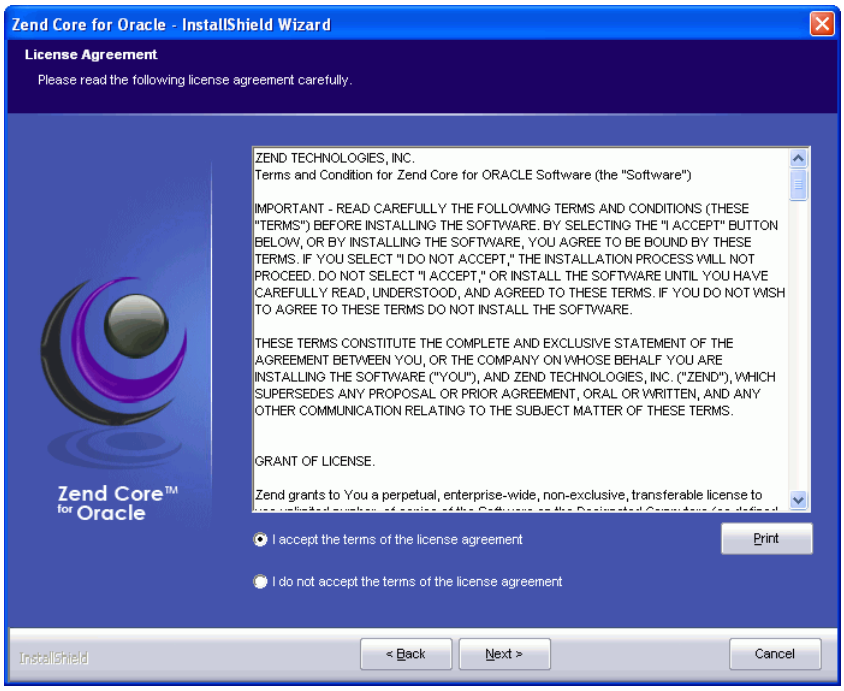
   Zend Core for Oracle に同梱されている README ファイルおよびインストール・ドキュメントを確認してください。

Zend インストーラでは、[Tab] キーまたは矢印キーを使用するか、あるいはマウスを使用して、入力フィールドおよびボタン間を移動します。ボタンを選択するには、[Enter] キーを押すか、またはマウスでボタンをクリックします。

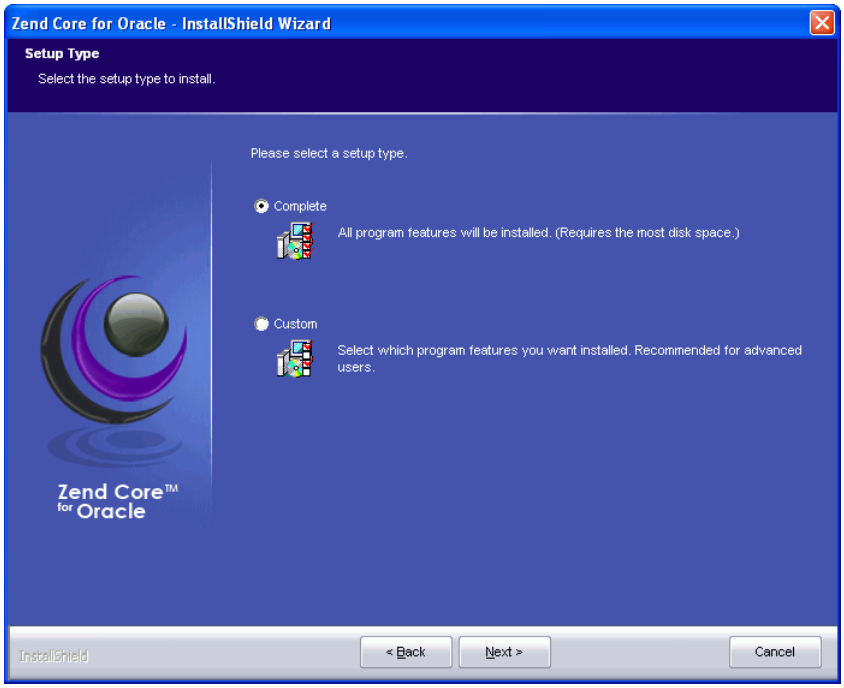
3. 最初の「Zend Core for Oracle Installation」ページで、**「Next」**をクリックします。



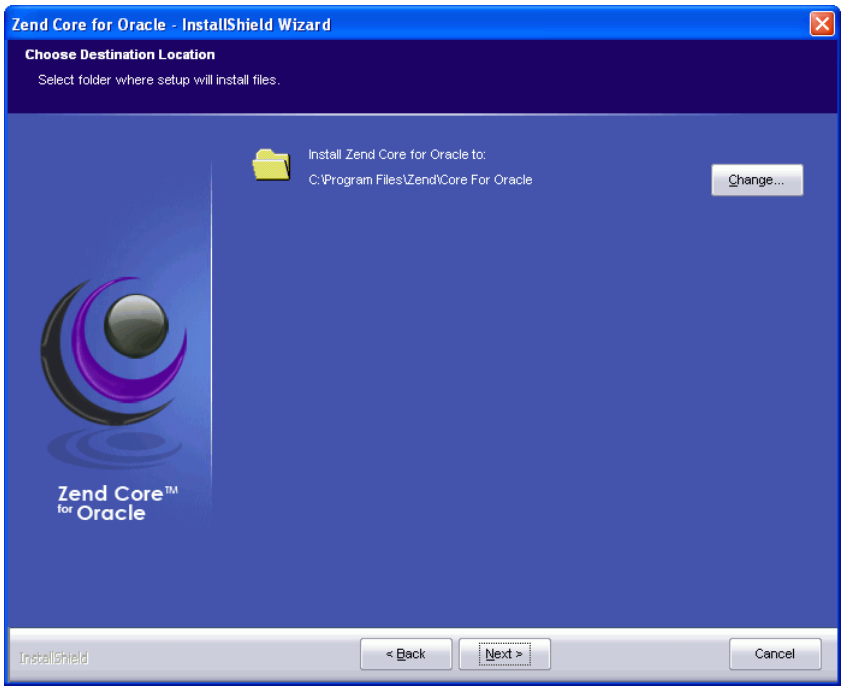
4. 「Zend Core for Oracle License Agreement」ページで、ライセンス契約を確認します。インストールを続行するには、**「I accept the terms of the license agreement」**を選択し、**「Next」**をクリックします。



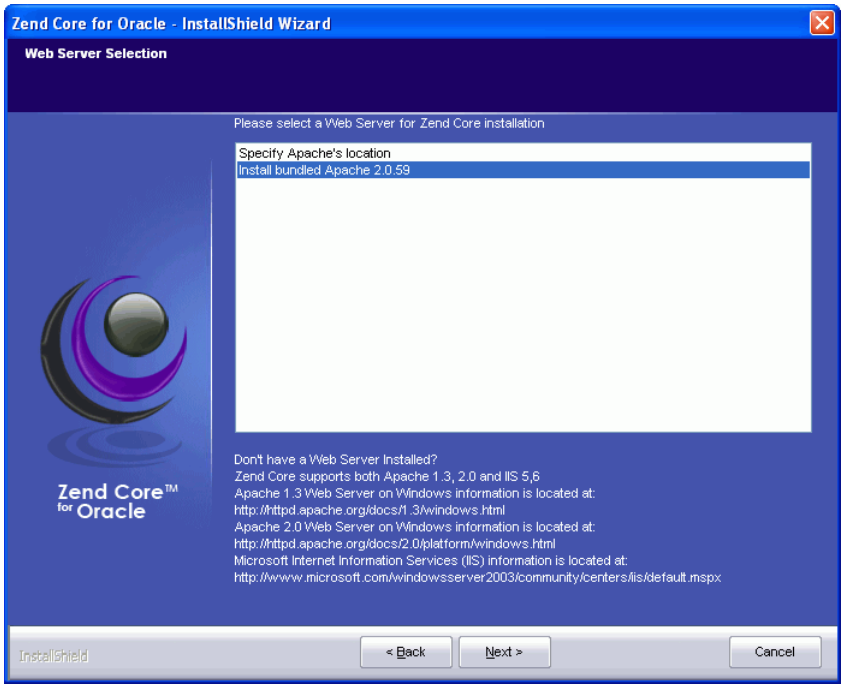
5. インストールのタイプを選択するように要求されます。「**Complete**」を選択し、「**Next**」をクリックします。



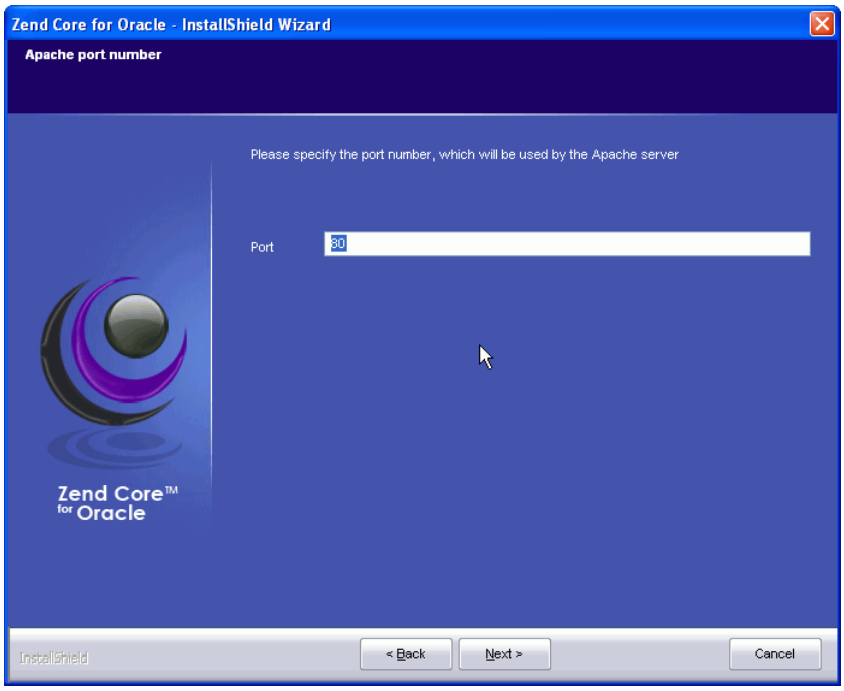
6. Zend Core for Oracle をインストールする場所を指定するように要求された場合は、デフォルトの場所を受け入れ（または任意の場所を入力し）、「**Next**」をクリックします。



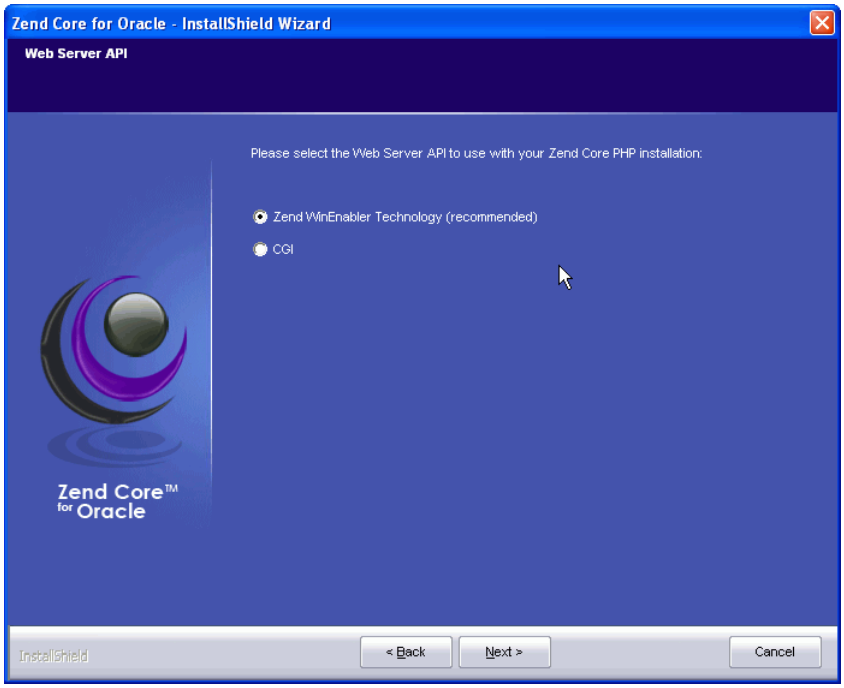
7. 次のページでは、Zend Core インストールの Web サーバーを選択するように要求されます。「Install Bundled Apache」オプションを受け入れ、「**Next**」をクリックします。



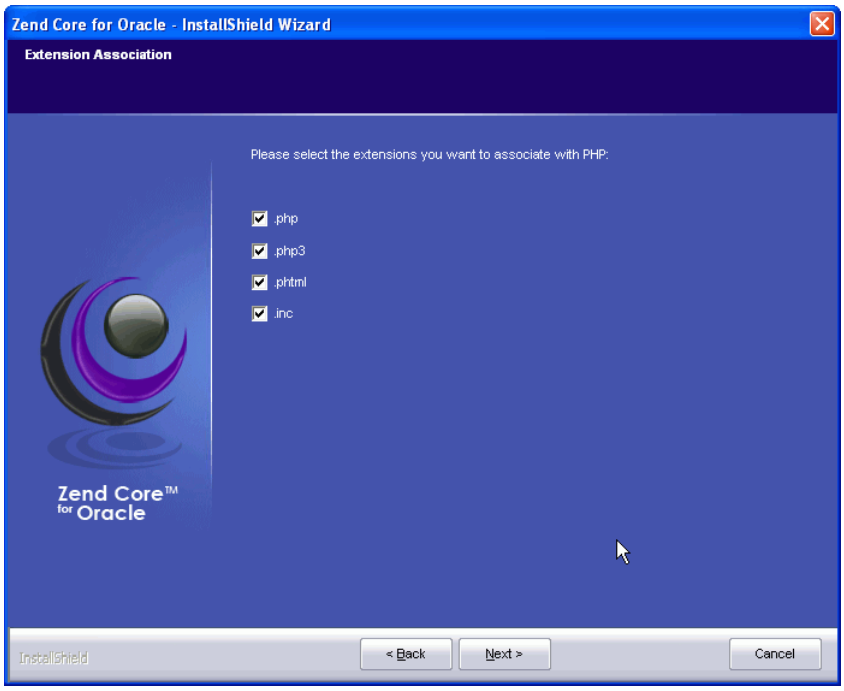
8. 次に、Apache で使用するポート番号を入力するように要求されます。デフォルト値の 80 を受け入れ、「**Next**」をクリックします。



9. 次に、使用する Web サーバー API を入力するように要求されます。「**Apache Module**」を選択し、「**Next**」をクリックします。

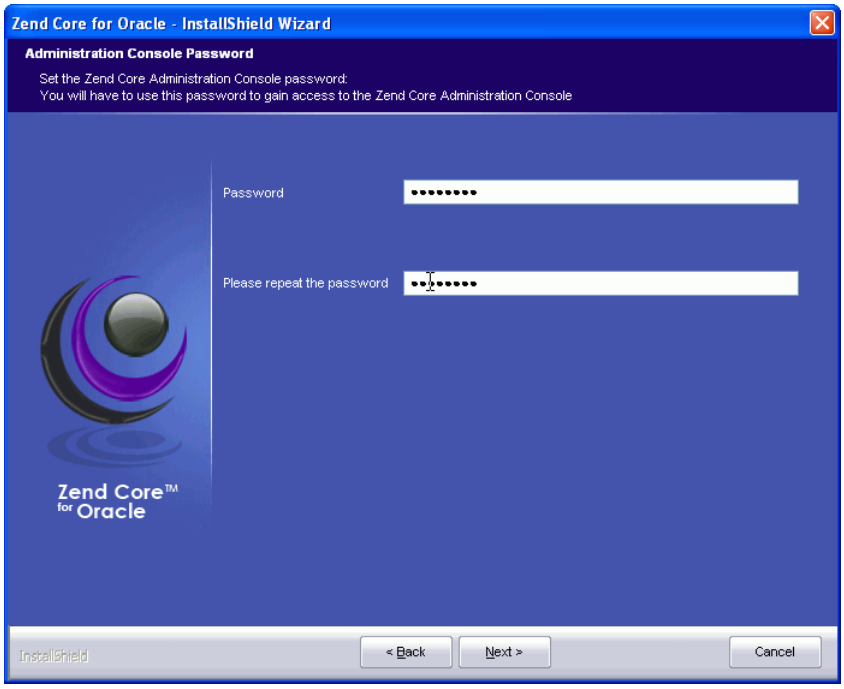


10. Zend Core for Oracle インストールに関連付ける拡張子を選択するように要求された場合は、4 つすべてを選択し、「**Next**」をクリックします。

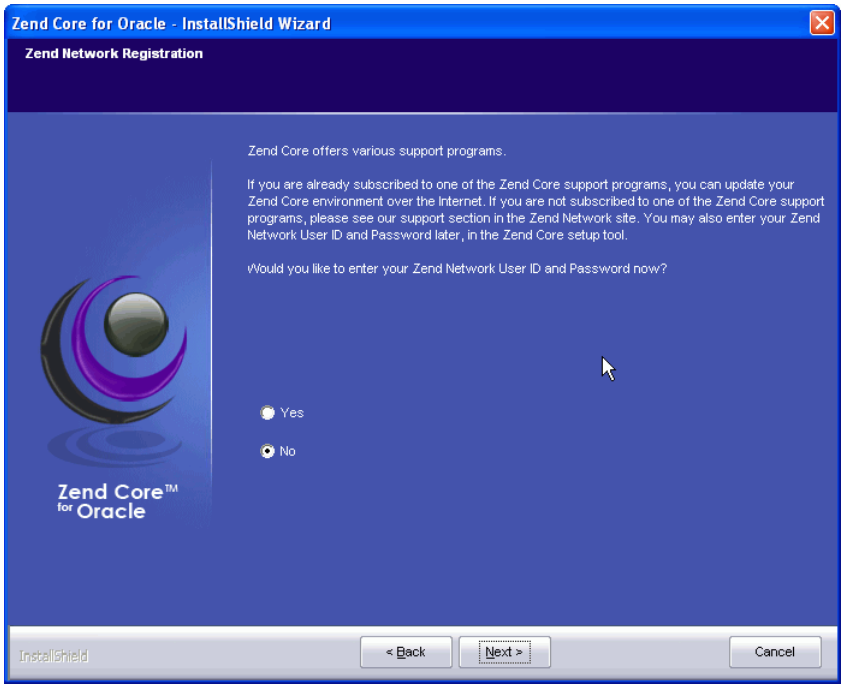


11. ここで、Zend Core GUI のパスワードを入力するように要求されます。このパスワードを入力すると、Zend Core コンソールにログインして、ディレクティブまたはプロパティ値を構成できるようになります。

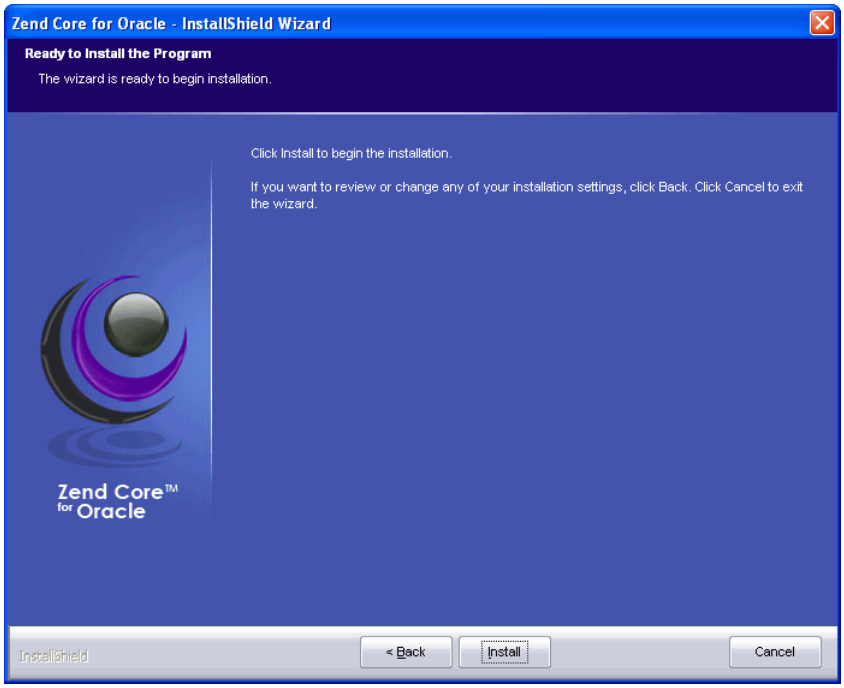
    Zend Core コンソールへのアクセスに使用するパスワードを入力し、「**Next**」をクリックします。



12. オプションで、Zend Network のユーザー ID とパスワードを入力すると、Zend Core コンソールを使用して Zend Core および PHP コンポーネントの更新状況を追跡できます。登録していない場合、または更新状況を追跡しない場合は、「**No**」を選択し、「**Next**」をクリックします。



13. これで、インストール・ウィザードを使用して Zend Core for Oracle をコンピュータにインストールする準備ができました。インストール・ウィザードを起動するには、「**Install**」をクリックします。



14. インストールを完了するために、コンピュータを再起動するように要求されます。「**Yes, I want to restart my computer now**」を選択し、「**Finish**」をクリックします。

インストールが完了し、Zend Core for Oracle を構成する準備ができました。

Linux での Zend Core for Oracle のインストール

この項では、Linux に Zend Core for Oracle をインストールする方法について説明します。

このチュートリアルは、Zend Core for Oracle での PHP に固有です。

Zend Core for Oracle の詳細な設定情報は、次の URL の Zend Core for Oracle の Web ページの「Product Information」にあるインストレーション・ガイドを参照してください。

```
http://www.oracle.com/technology/tech/php/zendcore/index.html
```

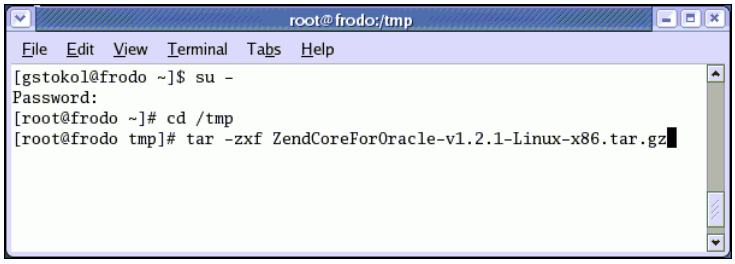
この手順では、Zend Core for Oracle ソフトウェアを `/tmp` にダウンロードしたと想定します。そうでない場合は、手順 1 で、ダウンロードしたソフトウェアが格納されているディレクトリに移動する必要があります。

ファイル名および抽出ディレクトリは、現行バージョンに基づいて決定されます。この手順では、インストールしているバージョンのディレクトリ名を使用するようにしてください。

Zend Core for Oracle をインストールするには、root ユーザーである必要があります。Zend Core for Oracle をインストールするには、次の手順を実行します。

1. コマンド・ウィンドウに次のコマンドを入力して、ダウンロードした Zend Core for Oracle ソフトウェアのコンテンツを抽出します。

```
su -
Password: <enter the root password>
cd /tmp
tar -zxf ZendCoreForOracle-v1.2.1-Linux-x86.tar.gz
```

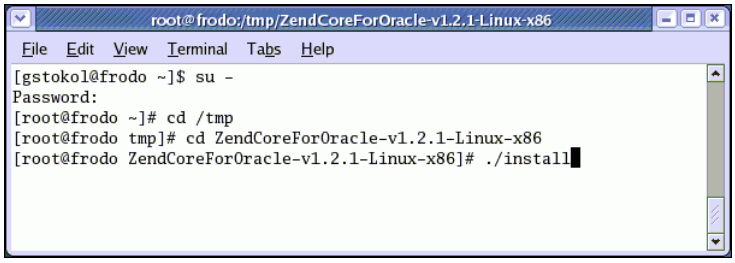


デフォルトでは、ZendCoreForOracle-v1.2.1-Linux-x86 というサブディレクトリにファイルが抽出されます。

Zend Core for Oracle に同梱されている README ファイルおよびインストール・ドキュメントを確認してください。

2. Zend Core for Oracle インストール・プロセスを開始するには、次のコマンドを入力します。

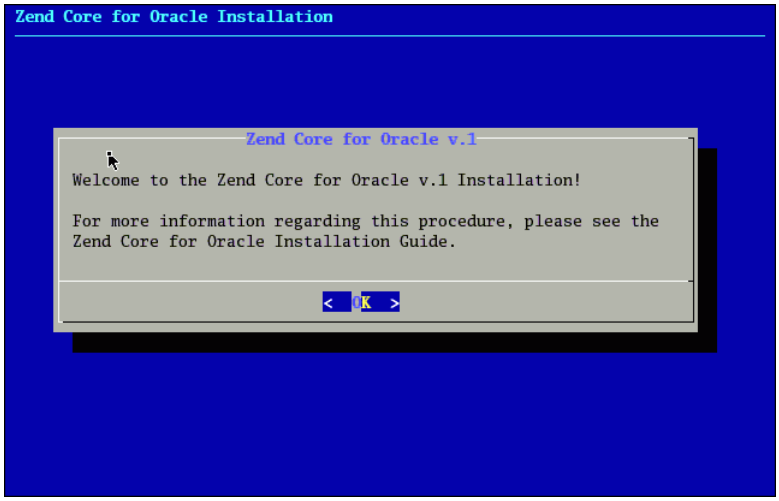
```
cd ZendCoreForOracle-v1.2.1-Linux-x86
./install
```



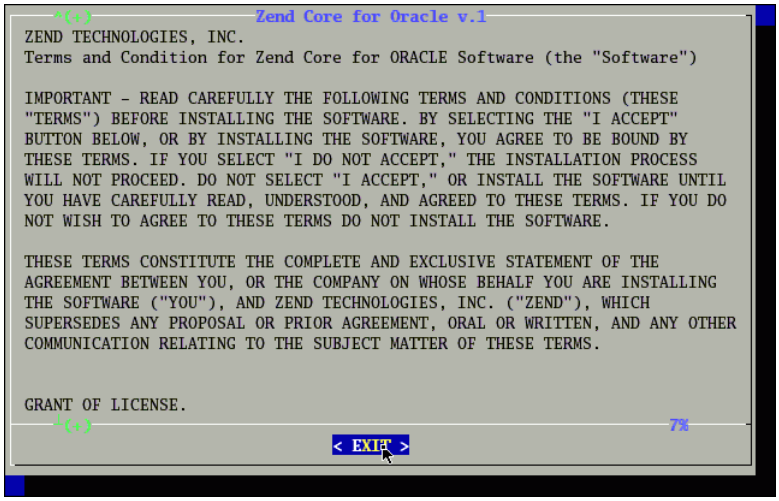
install コマンドは、root ユーザー権限で実行する必要があります。./install コマンドを入力すると、インストール・プロセスが開始されます（以降の手順を参照）。

Zend インストーラでは、[Tab] キーまたは矢印キーを使用するか、あるいはマウスを使用して、入力フィールドおよびボタン間を移動します。ボタンを選択するには、[Enter] キーを押すか、またはマウスでボタンをクリックします。

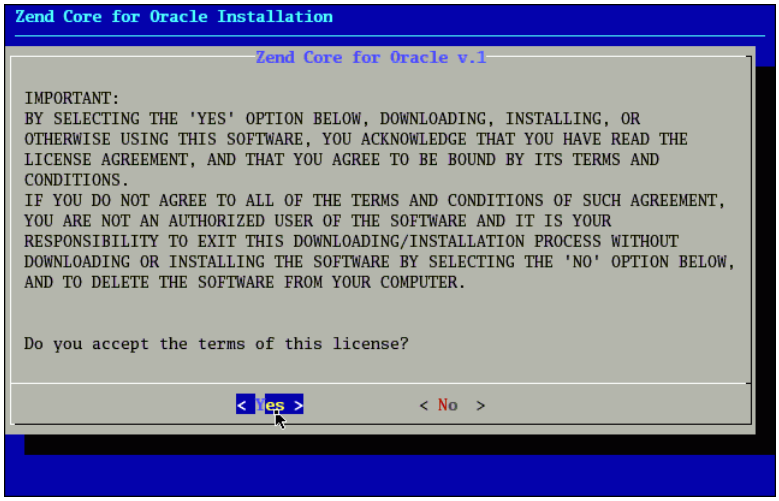
3. 最初の「Zend Core for Oracle Installation」ページで、**「OK」**をクリックします。



4. 「Zend Core for Oracle V.1」ページで、ライセンス契約を確認します。インストールを続行するには、**「Exit」**をクリックします。

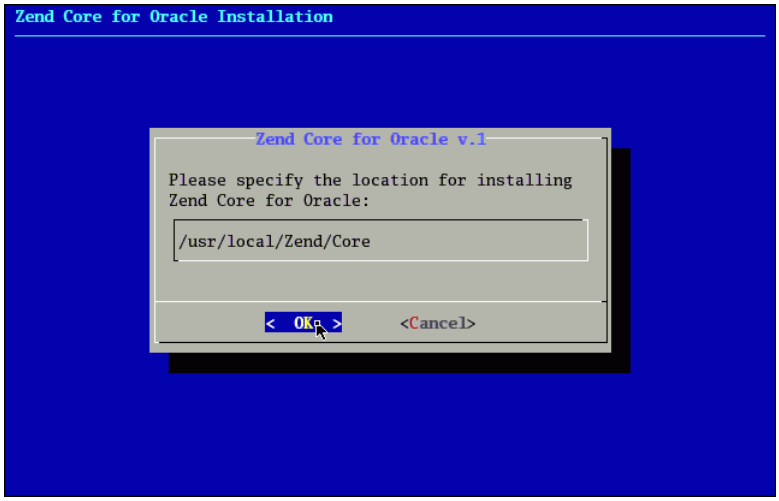


5. ライセンス条項に同意するように要求された場合は、「**Yes**」をクリックします。



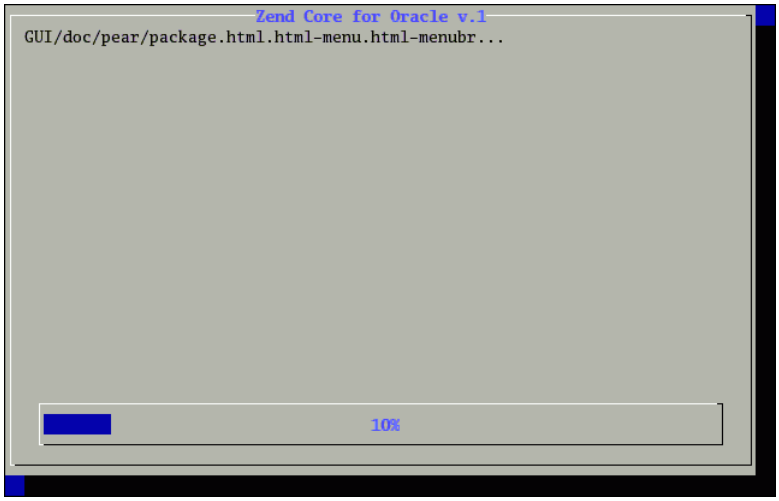
Copyright, 2006, Zend Technologies Ltd.

6. Zend Core for Oracle をインストールする場所を指定するように要求された場合は、デフォルトの場所を受け入れ（または任意の場所を入力し）、「**OK**」をクリックします。

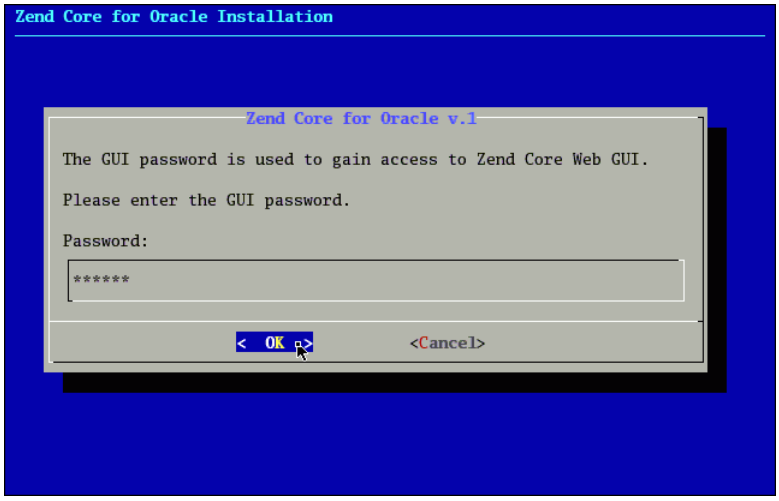


Copyright, 2006, Zend Technologies Ltd.

インストールに必要なファイルの抽出が開始されます。インストールの実行中、次の進捗画面が表示されます。

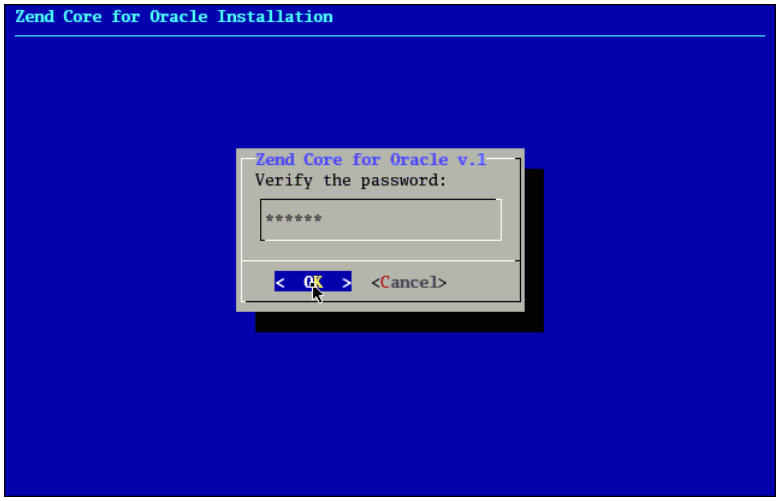


7. ソフトウェアがすべてインストールされたことがこの進捗画面に示されると、GUI パスワードを入力するように要求されます。「Password」フィールドに Zend Core コンソールへのアクセスに使用するパスワードを入力し、**「OK」**をクリックします。

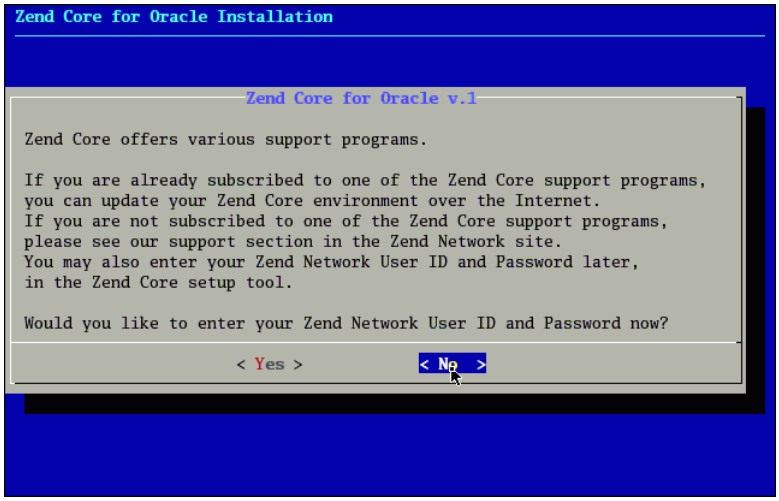


ここでパスワードを指定すると、Zend Core for Oracle の管理用 Web ページにログインできるようになります。これらのページで、Zend Core for Oracle エンジンのディレクティブおよびプロパティ値を構成できます。

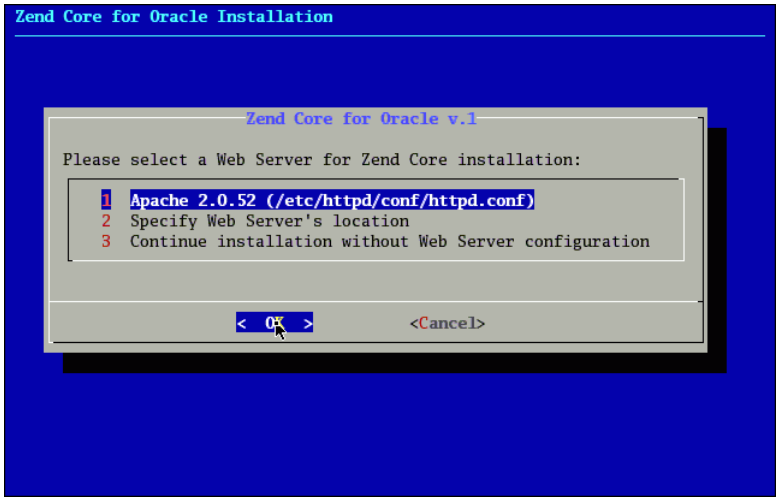
8. パスワードの確認を求められた場合は、手順 7 で指定したパスワードと同じパスワードを入力し、「**OK**」をクリックします。



9. 「Zend Core support」ページで、オプションで、Zend Network のユーザー ID とパスワードを入力すると、Zend Core コンソールを使用して Zend Core および PHP コンポーネントの更新状況を追跡できます。登録していない場合または更新状況を追跡しない場合は、「**No**」をクリックします。

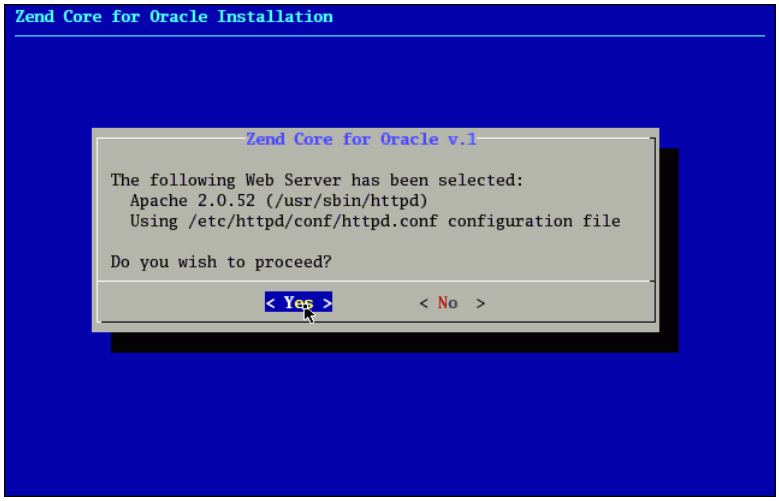


10. 次のページでは、Zend Core インストール用の Web サーバーを選択するように要求されます。デフォルト（Linux にインストールした Apache）を選択し、「**OK**」をクリックします。

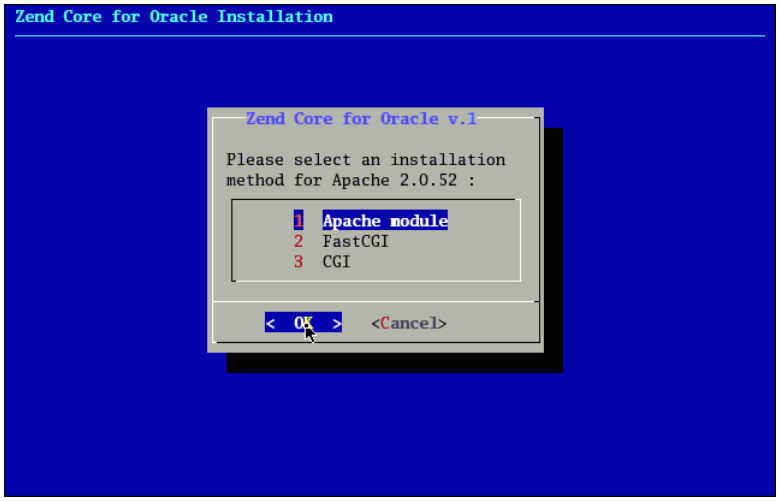


システムにインストールされているサポート対象の別の Web サーバーを使用して、Zend Core for Oracle をインストールすることもできます。

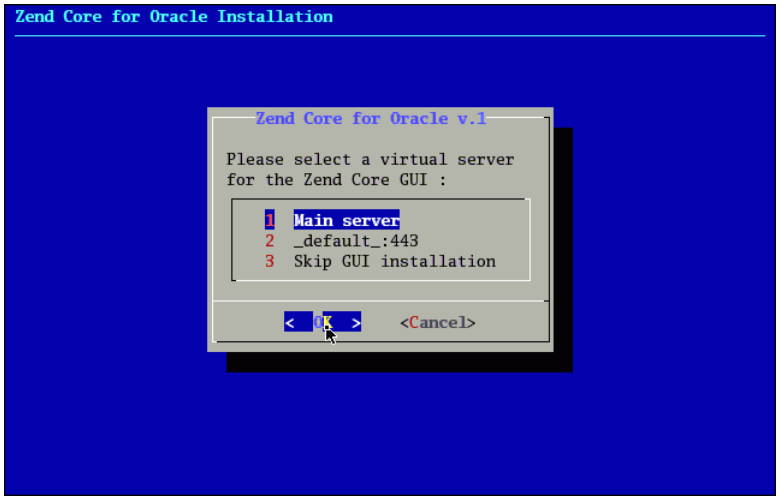
11. Web サーバーの選択を確認するページで、操作を続行するかどうかの確認が求められます。「**Yes**」をクリックします。



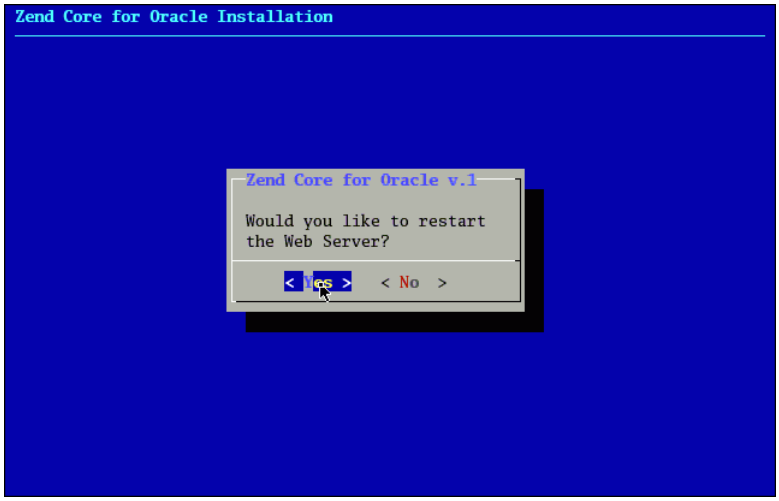
12. 次のインストール・ページでは、Apache2.0.52 用のインストール方法を選択するように要求されます。インストール方法として「**Apache module**」を選択し、「**OK**」をクリックします。



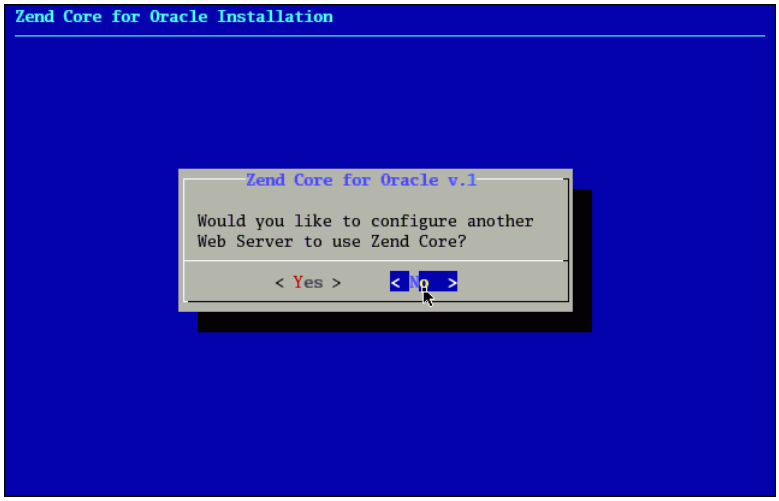
13. 次のインストール・ページでは、Zend Core GUI 用の仮想サーバーを選択するように要求されます。「**Main Server**」を選択し、「**OK**」をクリックします。



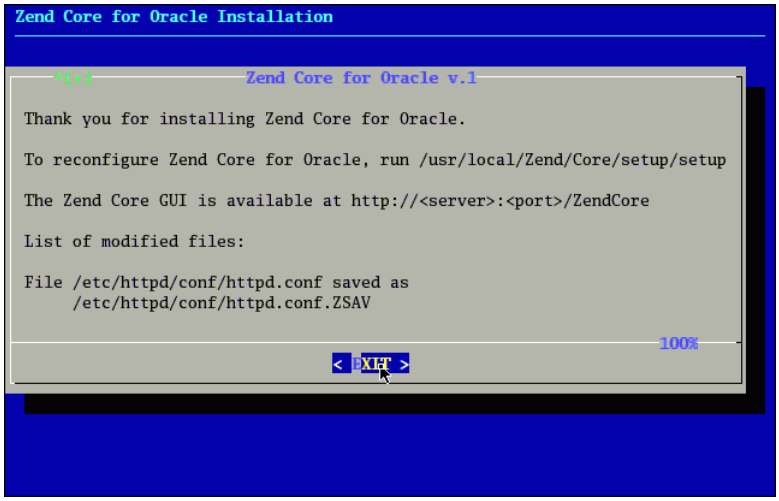
14. 次のインストール・ページでは、Web サーバーを再起動するかどうかの確認が求められます。「**Yes**」をクリックします。



15. 次のインストール・ページでは、Zend Core for Oracle を使用するように別の Web サーバーを構成するかどうかの確認が求められます。「**No**」をクリックします。



16. 最後のインストール・ページには、有効な構成コマンドおよび Zend Core エンジンの管理用 Web ページが表示されます。この情報を書き留め、「**EXIT**」をクリックします。



17. Zend Core インストールが終了すると、インストール最終画面のテキストが端末に表示されます。

```
********************************************************************************

Thank you for installing Zend Core for Oracle.

To reconfigure Zend Core for Oracle, run /usr/local/Zend/Core/setup/setup

The Zend Core GUI is available at http://<server>:<port>/ZendCore

List of modified files:

File /etc/httpd/conf/httpd.conf saved as
     /etc/httpd/conf/httpd.conf.ZSAV

********************************************************************************



[root@frodo ZendCoreForOracle-v1.2.1-Linux-x86]#
```


インストールが完了し、Zend Core for Oracle を構成する準備ができました。

Zend Core for Oracle の構成

この項では、環境変数および Zend Core ディレクティブを構成して、Web ページにレポートされるデフォルト・エラーを制御します。

1. Zend Core の管理ページにアクセスするには、Web ブラウザに次の URL を入力します。

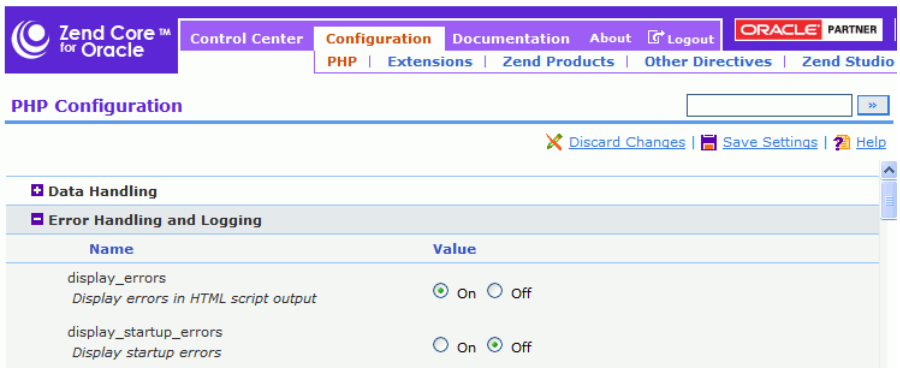
   `http://localhost/ZendCore/`

   Zend Core for Oracle の「Welcome」ページが表示されます。

2. Zend Core for Oracle のインストール時に指定した GUI パスワードを「Password」フィールドに入力します。ログイン（>>>）・アイコンをクリックします。

「Control Center System Overview」ページが表示されます。

3. 構成オプションを表示するには、**「Configuration」**タブをクリックします。
4. **「Error Handling and Logging」**構成エントリを開くには、（+）アイコンをクリックします。
5. 開発時に HTML スクリプトの出力にエラーを表示できるようにするには、`display_errors` ディレクティブを `On` に設定します。



6. 構成の変更を保存するには、**「Save Settings」**をクリックします。

   構成に変更を行ったため、Apache HTTP サーバーを再起動する必要があります。ページ・ヘッダーに「Please Restart Apache」メッセージが表示され、Apache の再起動が必要であることが示されます。

7. **「Logout」**をクリックして、「Zend Core for Oracle Administration」ページを終了します。
8. Apache を再起動します。ApacheMonitor ユーティリティを使用するか、または Windows のサービスを使用できます。

   ApacheMonitor ユーティリティを使用するには、Apache の `bin` ディレクトリに移動し、`ApacheMonitor.exe` をダブルクリックします。デフォルトのインストールでは、Apache の `bin` ディレクトリは `c:¥Program Files¥Zend¥Apache2¥bin` です。

Windows のサービスにアクセスするには、Windows の**「スタート」**メニューで、**「スタート」→「コントロール パネル」→「管理ツール」→「サービス」を選択します。「標準」**タブを選択します。「Apache2 HTTP Server」を右クリックし、**「再起動」**を選択します。

これで、基本的な構成変更は完了です。次の項に進んで、Zend Core for Oracle インストールをテストしてください。

Zend Core for Oracle インストールのテスト

Zend Core for Oracle インストールをテストするには、次の手順を実行します。

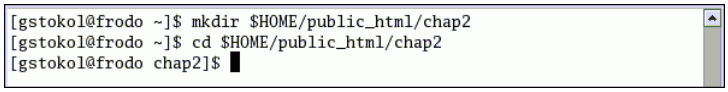
1. chap2 というサブディレクトリを作成します。アプリケーション・ファイル用のディレクトリを作成し、新しく作成したディレクトリに変更するには、コマンド・ウィンドウに次のコマンドを入力します。

   Windows の場合 :

   ```
   mkdir "c:¥program files¥Zend¥Apache2¥htdocs¥chap2"
   cd c:¥program files¥Zend¥Apache2¥htdocs¥chap2
   ```

   Linux の場合 :

   ```
   mkdir $HOME/public_html/chap2
   cd $HOME/public_html/chap2
   ```



   別の場所にファイルを作成する場合は、作業ディレクトリ名および URL と一致するように、ファイルの編集および実行の手順を変更する必要があります。

2. hello.php というファイルを作成し、次の HTML テキストを含めます。

   ```
   <?php
     echo "Hello, world!";
   ?>
   ```

3. Web ブラウザを開き、次の URL を入力します。

   Windows の場合 :

   ```
   http://localhost/chap2/hello.php
   ```

   Linux の場合 :

   ```
   http://localhost/~<username>/chap2/hello.php
   ```

   ブラウザに「Hello, world!」という行が表示されます。

3

接続

この章では、Oracle Database への接続および切断を行う PHP 関数を実装する HR アプリケーション・ファイルを作成します。また、問合せを実行してデータベース接続が正常に確立されたことを確認できる PHP 関数も開発します。

この章では、関数をコールして「Departments」ページのヘッダーおよびフッターを生成する PHP ファイルを作成および変更する手順についても説明します。「Departments」ページのフッター・セクションには日時が含まれています。

この章の内容は次のとおりです。

注意： ここでは、わかりやすくするために、サンプル・アプリケーション・コードにユーザー名およびパスワードが書き込まれています。デプロイするアプリケーションでは、アプリケーションのソース・コードにユーザー名およびパスワードの文字列を直接コーディングしないことをお薦めします。ユーザーにユーザー名およびパスワードの入力を求めるダイアログ・ボックスを実装する方法などのより安全な方法を使用することをお薦めします。

セキュリティ機能およびセキュリティ・プラクティスの詳細は、『Oracle Database セキュリティ・ガイド』および開発環境のドキュメントを参照してください。

「Departments」ページの構築

この項では、アプリケーションの最初の画面用の関数およびスタイルを作成します。

「Departments」ページを構築するには、次の手順を実行します。

1. アプリケーション・ファイル用のディレクトリを作成し、新しく作成したディレクトリに変更するには、コマンド・ウィンドウに次のコマンドを入力します。

   Windows の場合 :

   ```
   mkdir c:¥program files¥Zend¥Apache2¥htdocs¥chap3
   cd c:¥program files¥Zend¥Apache2¥htdocs¥chap3
   ```

   Linux の場合 :

   ```
   mkdir $HOME/public_html/chap3
   cd $HOME/public_html/chap3
   ```

   別の場所にファイルを作成する場合は、作業ディレクトリ名および URL と一致するように、ファイルの編集および実行の手順を変更する必要があります。

2. アプリケーション・ユーザー・インタフェースの開発を開始するには、任意のテキスト・エディタを使用して `anyco_ui.inc` というファイルを作成し、`ui_print_header()` と `ui_print_footer()` の 2 つの関数をそれぞれのパラメータとともにそのファイルに含めて、アプリケーションの Web ページでヘッダー・セクションおよびフッター・セクションの一貫性を保持できるようにします。

   ```
   <?php

   function ui_print_header($title)
   {
     $title = htmlentities($title);
     echo <<<END
     <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">
     <html>
     <head>
       <meta http-equiv="Content-Type"
             content="text/html; charset=ISO-8859-1">
       <link rel="stylesheet" type="text/css" href="style.css">
       <title>Any Co.: $title</title>
     </head>
     <body>
     <h1>$title</h1>
   END;
   }

   function ui_print_footer($date)
   {
     $date = htmlentities($date);
     echo <<<END
     <div class="footer">
       <div class="date">$date</div>
       <div class="company">Any Co.</div>
     </div>
   END;
   }

   ?>
   ```

   - このアプリケーション設計では、PHP 関数定義を使用して、再利用可能なコードをモジュール化します。

- anyco_ui.inc の各関数では、ヒア・ドキュメントという PHP の言語要素を使用します。これによって、次の 2 行間に任意のサイズの HTML 形式のテキストを配置できます。

  ```
  echo <<<END
  END;
  ```

- END; 行の先頭には空白を配置しないでください。配置すると、残りのドキュメントが、出力対象テキストの一部として処理されます。
- ヒア・ドキュメントの本体内に記述されているすべての PHP パラメータ（$title や $date など）は、それぞれの値に置き換えられます。
- PHP 関数 htmlentities() は、ユーザー指定のテキストに誤って HTML マークアップが含まれ、出力形式に影響を与えるのを防ぐために使用します。

3. PHP ファイルでは、ブラウザでの HTML の表示スタイルを指定するために style.css というカスケード・スタイルシート（CSS）・ファイルが使用されます。

   chap3 ディレクトリに、次の CSS テキストを含む style.css ファイルを作成します。

```
body
{ background: #CCCCFF;
  color:      #000000;
  font-family: Arial, sans-serif; }

h1
{ border-bottom: solid #334B66 4px;
  font-size: 160%; }

table
{ padding: 5px; }

td
{ border: solid #000000 1px;
  text-align: left;
  padding: 5px; }

th
{ text-align: left;
  padding: 5px; }

.footer
{ border-top: solid #334B66 4px;
  font-size: 90%; }

.company
{ padding-top: 5px;
  float: right; }

.date
{ padding-top: 5px;
  float: left; }
```

4. ユーザー・インタフェース関数をコールするには、次のテキストを含む anyco.php ファイルを作成します。

```
<?php

require('anyco_ui.inc');

ui_print_header('Departments');
ui_print_footer(date('Y-m-d H:i:s'));

?>
```

   PHP コマンド require() は、anyco_ui.inc をインクルードするために使用します。新しい関数をコールして HTML 出力を生成できます。

5. anyco.php ファイルをテストするには、ブラウザに次の URL を入力します。

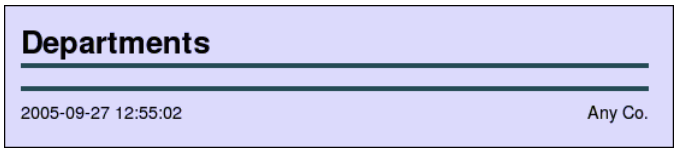
   Windows の場合 :

   ```
   http://localhost/chap3/anyco.php
   ```

   Linux の場合 :

   ```
   http://localhost/~<username>/chap3/anyco.php
   ```

   生成される Web ページは、次のようになります。



   日時がページのフッター・セクションに表示されます。

データベースへの接続

この項では、部門データを表示できるように「Departments」画面にデータベース接続を追加します。

アプリケーションにデータベース接続を追加するには、次の手順を実行します。

データベース接続を確立するには、次の 3 つの文字列パラメータを指定した oci_connect() 関数を使用します。

```
$conn = oci_connect($username, $password, $db)
```

1 つ目および 2 つ目のパラメータはそれぞれデータベース・ユーザー名およびパスワードです。3 つ目のパラメータはデータベース接続識別子です。
oci_connect() 関数は、他の OCI8 コールに必要な接続リソースを戻します。ただし、エラーが発生した場合は FALSE を戻します。戻された接続識別子は、$conn という変数に格納されます。

1. 次のパラメータが指定されているデータベース接続を追加するために anyco.php ファイルを編集します。
   - ユーザー名は hr です。
   - この例のパスワードは hr です。実際には、HR ユーザーの実際のパスワードを使用してください。
   - Oracle 接続識別子は //localhost/orcl です。
2. oci_connect() コールが使用可能なデータベース接続を戻すことを検証するために anyco.php ファイルを編集します。oci_connect() へのコールで取得したデータベース接続識別子および DEPARTMENTS 表のすべての行を選択する問合せ文字列の 2 つのパラメータをとる do_query() 関数を記述します。
3. 問合せを実行するための準備を整えるために anyco.php ファイルを編集します。oci_parse() コールを追加します。oci_parse() 関数には、接続識別子および問合せ文字列の 2 つのパラメータが含まれています。この関数は、問合せを実行し、生成されたデータ行をフェッチするために必要な文識別子を戻します。エラーが発生した場合は、FALSE を戻します。
4. 問合せを実行するために anyco.php ファイルを編集します。oci_execute() 関数へのコールを追加します。oci_execute() 関数は、1 つ目のパラメータに指定された文識別子に関連付けられている文を実行します。2 つ目のパラメータには、実行モードを指定します。OCI_DEFAULT は、文を自動的にコミットしないことを示すために使用します。デフォルトの実行モードは OCI_COMMIT_ON_SUCCESS です。oci_execute() 関数は、正常に完了した場合は TRUE を戻し、そうでない場合は FALSE を戻します。

5. 実行した問合せに対して生成されるすべての行をフェッチするために anyco.php ファイルを編集します。while ループおよび oci_fetch_array() 関数へのコールを追加します。oci_fetch_array() 関数は、結果データの次の行を戻します。次の行が存在しない場合は、FALSE を戻します。oci_fetch_array() 関数の 2 つ目のパラメータ OCI_RETURN_NULLS は、NULL データベース・フィールドが PHP NULL 値として戻されることを示しています。

   データの各行は数値配列として戻されます。コードでは、配列のループに PHP の foreach 構文が使用され、各列値が表の行要素内の HTML 表セルに出力されます。列値が NULL の場合は、改行なしスペースが出力されます。それ以外の場合は、列値が出力されます。

   手順 1 ～ 5 までの編集を終えると、anyco.php ファイルは次のようになります。

```
<?php // File: anyco.php

require('anyco_ui.inc');

// Create a database connection
$conn = oci_connect('hr', 'hr', '//localhost/orcl');

ui_print_header('Departments');
do_query($conn, 'SELECT * FROM DEPARTMENTS');
ui_print_footer(date('Y-m-d H:i:s'));

// Execute query and display results
function do_query($conn, $query)
{
  $stid = oci_parse($conn, $query);
  $r = oci_execute($stid, OCI_DEFAULT);

  print '<table border="1">';
  while ($row = oci_fetch_array($stid, OCI_ASSOC+OCI_RETURN_NULLS)) {
    print '<tr>';
    foreach ($row as $item) {
      print '<td>'.
            ($item ? htmlentities($item) : ' ').'</td>';
    }
    print '</tr>';
  }
  print '</table>';
}

?>
```

6. anyco.php に行った変更をテストするために、変更後の anyco.php ファイルを保存します。ブラウザのウィンドウに、次の URL を入力します。

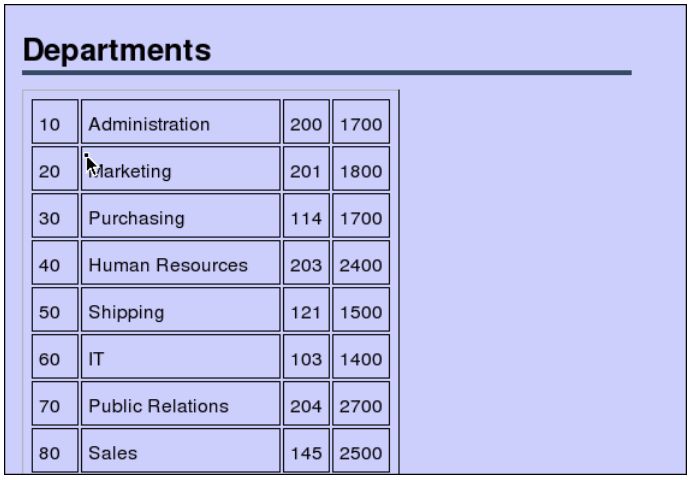
   Windows の場合 :

   http://localhost/chap3/anyco.php

   Linux の場合 :

   http://localhost/~<username>/chap3/anyco.php

ブラウザ・ウィンドウに、次のようなページが戻されます。



EMPLOYEES のデータを問い合せる場合は、`do_query()` 関数コールの問合せを次のように変更することもできます。

```
do_query($conn, 'SELECT * FROM EMPLOYEES');
```

接続で問題が発生した場合

ユーザー名、パスワードおよび接続文字列が有効であることを確認します。接続文字列 `//localhost/orcl` で、Oracle 簡易接続構文を使用します。Oracle Net `tnsnames.ora` ファイルを使用して接続先のデータベースを指定する場合は、`oci_connect()` 関数の 3 つ目のパラメータとしてネットワーク別名を使用します。

エラーが表示されていない場合は、`display_errors` ディレクティブを `ON` に設定し、`error_reporting` ディレクティブを `E_ALL|E_STRICT` に設定します。

PHP コードに問題があり、デバッガを使用していない場合は、PHP の `var_dump()` 関数を使用して変数を調べることができます。次に例を示します。

```
print '<pre>';
var_dump($r);
print '</pre>';
```

他の接続方法

一部のアプリケーションでは、永続接続を使用すると、スクリプトがコールされるたびに再接続する必要がなくなるため、パフォーマンスが向上します。この場合、Apache の構成によっては、多くのデータベース接続が同時にオープンしたままになることがあります。接続のパフォーマンス上のメリットとデータベース・サーバーで発生するオーバーヘッドとのバランスをとる必要があります。

永続接続は、OCI8 の `oci_pconnect()` 関数によって確立されます。PHP 初期化ファイルの設定を使用すると、永続接続の存続時間を制御できます。この設定には、次のものが含まれています。

oci8.max_persistent: プロセスごとの永続接続の数を制御します。

oci8.persistent_timeout: プロセスでアイドル状態の永続接続が保持される時間（秒単位）を指定します。

oci8.ping_interval: 永続接続を ping してその妥当性を確認する前に待機する必要がある時間（秒単位）を指定します。

詳細は、次の PHP のリファレンス・マニュアルを参照してください。

```
http://www.php.net/manual/en/ref.oci8.php
```

共有セッションの詳細は、『Oracle Call Interface プログラマーズ・ガイド』および『Oracle Database Net Services 管理者ガイド』の OCI での接続プーリングに関する項を参照してください。

データベースからの切断

PHP エンジンは、永続接続が確立されていないかぎり、スクリプトの終了時に自動的にデータベース接続をクローズします。明示的に非永続データベース接続をクローズする場合は、`oci_connect()` コールによって戻される接続識別子で `oci_close()` OCI 関数をコールできます。次に例を示します。

```
<?php

$conn = oci_connect('hr', '<your_password>', '//localhost/orcl');
...
oci_close($conn);

...

?>
```

4

データの問合せ

この章では、「Departments」ページに情報を追加して、第 3 章で作成した Anyco HR アプリケーションを拡張します。また、特定の部門の従業員レコードに対して問合せ、挿入、更新および削除を行う機能も実装します。

この章の内容は次のとおりです。

データベース・アプリケーション・ロジックの集中化

この項では、PHP アプリケーションにインクルードできるようにデータベース・アクセス・ロジックを個別のファイルに移動することによって、アプリケーション・コードを変更します。

1. 第 3 章で作成したファイルを chap4 という新しいディレクトリにコピーし、新しく作成したディレクトリに移動します。

   Windows の場合 :

   ```
   mkdir c:¥program files¥Zend¥Apache2¥htdocs¥chap4
   cd c:¥program files¥Zend¥Apache2¥htdocs¥chap4
   copy ..¥chap3¥* .
   ```

   Linux の場合 :

   ```
   mkdir $HOME/public_html/chap4
   cd $HOME/public_html/chap4
   cp ../chap3/* .
   ```

2. 任意のエディタを使用して、anyco_cn.inc というファイルを作成し、データベース接続情報の名前付き定数を定義します。このファイルを使用すると、接続情報を 1 箇所で変更できます。

   ```
   <?php // File: anyco_cn.inc

   define('ORA_CON_UN', 'hr');             // User name
   define('ORA_CON_PW', 'hr');             // Password
   define('ORA_CON_DB', '//localhost/orcl'); // Connection identifier

   ?>
   ```

   ここでは、わかりやすくするために、サンプル・アプリケーション・コードにユーザー名およびパスワードが書き込まれています。デプロイするアプリケーションでは、アプリケーションのソース・コードにユーザー名およびパスワードの文字列を直接コーディングしないことをお薦めします。ユーザーにユーザー名およびパスワードの入力を求めるダイアログ・ボックスを実装する方法などのより安全な方法を使用することをお薦めします。

   セキュリティ機能およびセキュリティ・プラクティスの詳細は、『Oracle Database セキュリティ・ガイド』および開発環境のドキュメントを参照してください。

3. データベース接続の作成、問合せの実行およびデータベースからの切断を実行する関数を宣言する anyco_db.inc というファイルを作成します。db_error() という追加の関数をコールすることによって管理されるエラー処理を含む次のロジックを使用します。

   ```
   <?php  // File: anyco_db.inc

   function db_connect()
   {
     // use constants defined in anyco_cn.inc
     $conn = oci_connect(ORA_CON_UN, ORA_CON_PW, ORA_CON_DB);
     if (!$conn) {
       db_error(null, __FILE__, __LINE__);
     }
     return($conn);
   }

   function db_do_query($conn, $statement)
   {
     $stid = oci_parse($conn, $statement);
     if (!$stid) {
       db_error($conn, __FILE__, __LINE__);
     }

     $r = oci_execute($stid, OCI_DEFAULT);
   ```

```
  if (!$r) {
    db_error($stid, __FILE__, __LINE__);
  }
 $r = oci_fetch_all($stid, $results, null, null,
                    OCI_FETCHSTATEMENT_BY_ROW);
  return($results);
}

// $r is the resource containing the error.
// Pass no argument or false for connection errors

function db_error($r = false, $file, $line)
{
  $err =  $r ? oci_error($r) : oci_error();

  if (isset($err['message'])) {
    $m = htmlentities($err['message']);
  }
  else {
    $m = 'Unknown DB error';
  }

  echo '<p><b>Error</b>: at line '.$line.' of '.$file.'</p>';
  echo '<pre>'.$m.'</pre>';

  exit;
}

?>
```

この例の db_do_query() 関数では、oci_fetch_all() OCI8 関数を使用します。oci_fetch_all() 関数は、次の 5 つのパラメータをとります。

- $stid（実行する文の文識別子）。
- $results（問合せで戻されたデータが格納される出力配列変数）。
- 3 つ目のパラメータは最初にスキップする行数を示し、NULL の場合は無視されます。
- 第 4 パラメータはフェッチする最大行数を示し、NULL の場合は無視されます。この場合、問合せに該当するすべての行が戻されます。この例では、結果セットが大きくないため、それでも問題ありません。
- 最後のパラメータ・フラグ OCI_FETCHSTATEMENT_BY_ROW は、$results 配列内のデータが行ごとに編成され、各行に列値の配列が含まれていることを示します。値が OCI_FETCHSTATEMENT_BY_COLUMN の場合、$results 配列は列ごとに編成され、各列エントリに各行の列値の配列が含まれます。このフラグに選択する値は、ロジックでデータを処理する方法によって異なります。

結果配列の構造を調べるには、問合せの実行後、PHP の var_dump() 関数を使用します。これはデバッグに有効です。次に例を示します。

```
print '<pre>';
var_dump($results);
print '</pre>';
```

db_error() 関数は、3 つの引数をとります。$r パラメータには、接続エラーを取得する場合は false または NULL を指定し、接続リソースまたは文リソースに関連するエラーを取得する場合はそのリソースを指定できます。$file 値および $line 値には __FILE__ および __LINE__ を実際のパラメータとして使用してそれぞれ値が移入されるため、データベース・エラーのレポート元のソース・ファイルおよび行をエラー・メッセージに表示できます。これによって、エラーの原因を簡単に追跡できるようになります。

db_error() 関数は、oci_error() 関数をコールしてデータベース・エラー・メッセージを取得します。

db_error() 関数は、メッセージを出力する前に isset() 関数をコールします。isset() 関数は、データベース・エラー構造のメッセージ・コンポーネントが設定されているかどうか、またはエラーが不明なものであるかどうかを確認します。

4. anyco_ui.inc を編集します。DEPARTMENTS 表の問合せから 1 行取得した結果を HTML 表形式に設定するには、次の関数を挿入します。

```
function ui_print_department($dept)
{
  if (!$dept) {
    echo '<p>No Department found</p>';
  }
  else {
    echo <<<END
  <table>
  <tr>
    <th>Department<br>ID</th>
    <th>Department<br>Name</th>
    <th>Manager<br>Id</th>
    <th>Location ID</th>
  </tr>
  <tr>
END;
    echo '<td>'.htmlentities($dept['DEPARTMENT_ID']).'</td>';
    echo '<td>'.htmlentities($dept['DEPARTMENT_NAME']).'</td>';
    echo '<td>'.htmlentities($dept['MANAGER_ID']).'</td>';
    echo '<td>'.htmlentities($dept['LOCATION_ID']).'</td>';
    echo <<<END
  </tr>
  </table>
END;
  }
}
```

第 3 章で説明したように、END; 行の先頭には空白を配置しないでください。配置すると、残りのドキュメントが、出力対象テキストの一部として処理されます。

5. anyco.php ファイルを編集します。anyco_ui.inc および anyco_db.inc ファイルをインクルードし、次のコードを使用して、department_id が 80 の部門の情報を問い合せて表示するデータベース関数をコールします。ファイルは、次のようになります。

```
<?php // File: anyco.php

require('anyco_cn.inc');
require('anyco_db.inc');
require('anyco_ui.inc');

$query =
  'SELECT   department_id, department_name, manager_id, location_id
   FROM     departments
   WHERE    department_id = 80';

$conn = db_connect();

$dept = db_do_query($conn, $query);
ui_print_header('Departments');
ui_print_department($dept[0]);
ui_print_footer(date('Y-m-d H:i:s'));

?>
```

6. アプリケーションに対して行った変更をテストするには、ブラウザのウィンドウに次のURL を入力します。

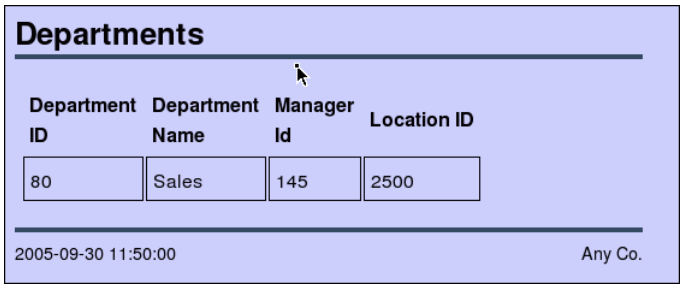
   Windows の場合 :

   ```
   http://localhost/chap4/anyco.php
   ```

   Linux の場合 :

   ```
   http://localhost/~<username>/chap4/anyco.php
   ```

   ブラウザ・ウィンドウに、次のようなページが戻されます。



バインド変数を使用した問合せの作成

WHERE 句に値を含めた問合せを使用すると有効な場合があります。ただし、問合せの条件値が変更される可能性が高い場合は、問合せに値をエンコードするのは適切ではありません。バインド変数を使用することをお薦めします。

バインド変数は、問合せ内でコロンの後に指定するシンボリック名で、リテラル値のプレースホルダとして動作します。たとえば、anyco.php ファイルに作成した問合せ文字列は、バインド変数 :did を使用して次のように記述しなおすことができます。

```
$query =
  'SELECT   department_id, department_name, manager_id, location_id
   FROM     departments
   WHERE    department_id = :did';
```

バインド変数を使用して SQL 文をパラメータ化すると、次のメリットがあります。

- コードを変更せずに様々な入力値で文を再利用できます。
- Oracle データベースでは、以前に同じ問合せ文字列を呼出したときの解析情報を再利用できるため、サーバーの問合せ解析時間が短縮され、問合せのパフォーマンスが向上します。
- SQL インジェクションというセキュリティの問題に対する保護機能があります。
- ユーザー入力で、引用符を特別に処理する必要はありません。

問合せでバインド変数を使用する場合は、問合せを実行する前に、PHP コードで、問合せで使用している各バインド変数（プレースホルダ）に実際の値を関連付ける必要があります。このプロセスは、ランタイム・バインディングと呼ばれます。

PHP アプリケーションで問合せのバインド変数を使用できるようにするには、PHP アプリケーション・コードに対して次の変更を行います。

1. anyco.php ファイルを編集します。バインド変数が使用されるように問合せを変更し、バインド変数に関連付ける値を格納するための配列を作成して、db_do_query() 関数に $bindargs 配列を渡します。

   ```
   <?php // File: anyco.php
   ...

   $query =
   'SELECT   department_id, department_name, manager_id, location_id
    FROM     departments
   ```

```
 WHERE    department_id = :did';

$bindargs = array();
// In the $bindargs array add an array containing
// the bind variable name used in the query, its value, a length
array_push($bindargs, array('DID', 80, -1));

$conn = db_connect();
$dept = db_do_query($conn, $query, $bindargs);

...
?>
```

この例では、DID というバインド変数は、パラメータ化した問合せの入力引数であり、値 80 に関連付けられています。このバインド変数の値は、後で動的に決定されます。また、OCI8 レイヤーで長さを決定できるように、長さ構成要素を -1 として渡します。バインド変数を使用してデータベースから出力を戻す場合は、サイズを明示的に指定する必要があります。

2. anyco_db.inc ファイルを編集します。$bindvars 配列変数を 3 つ目のパラメータとしてとるように db_do_query() 関数を変更します。oci_bind_by_name() OCI8 コールをコールして、$bindvars パラメータに指定した PHP 値を問合せのバインド変数に関連付けます。この関数は、次のようになります。

```
function db_do_query($conn, $statement, $bindvars = array())
{
  $stid = oci_parse($conn, $statement);
  if (!$stid) {
    db_error($conn, __FILE__, __LINE__);
  }

  // Bind the PHP values to query bind parameters
  foreach ($bindvars as $b) {
    // create local variable with caller specified bind value
    $$b[0] = $b[1];
    // oci_bind_by_name(resource, bv_name, php_variable, length)
    $r = oci_bind_by_name($stid, ":$b[0]", $$b[0], $b[2]);
    if (!$r) {
      db_error($stid, __FILE__, __LINE__);
    }
  }
  $r = oci_execute($stid, OCI_DEFAULT);
  if (!$r) {
    db_error($stid, __FILE__, __LINE__);
  }
  $r = oci_fetch_all($stid, $results, null, null,
       OCI_FETCHSTATEMENT_BY_ROW);
  return($results);
}
```

バインドは、oci_execute() が実行される前に foreach ループで実行されます。

$bindvars 配列の各エントリでは、最初の要素に問合せのバインド変数名が含まれており、この名前を使用して同じ名前の PHP 変数が作成されます。つまり、$$b[0] は $b[0] の値 DID をとり、$DID という PHP 変数を作成します。この PHP 変数の値は、エントリの 2 つ目の要素から割り当てられます。

oci_bind_by_name() 関数は、4 つのパラメータをとります。リソースとしての $stid、配列エントリの最初の要素から導出された問合せ内のバインド変数名を表す文字列、そのバインド変数に関連付けられる値が格納されている PHP 変数、および入力値の長さの 4 つです。

3. ここまでの変更の結果をテストするには、anyco.php ファイルおよび anyco_db.inc ファイルを保存し、次の URL を入力します。

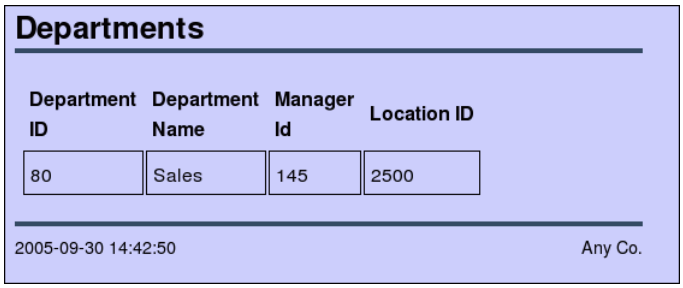
   Windows の場合 :

   http://localhost/chap4/anyco.php

   Linux の場合 :

   http://localhost/~<username>/chap4/anyco.php

   ブラウザ・ウィンドウに、次のようなページが戻されます。



データベース・レコードのナビゲート

データベース・レコードのナビゲーションを追加するには、アプリケーション・ロジックに対していくつかの重要な変更を行う必要があります。この変更を行うには、次の操作を組み合せる必要があります。

- データベース・レコードを 1 つずつ移動する「**Next**」および「**Previous**」ナビゲーション・ボタンを提供する HTML フォームをインクルードします。
- ページの HTTP リクエストが、「**Next**」または「**Previous**」ボタンをクリックしてポストされたかどうかを検出します。
- HTTP セッション・ステートを使用して最後に問い合せた行を追跡します。HTTP リクエスト間で特定のクライアントの状態情報を保持するために PHP セッションを開始します。最初の HTTP リクエストが最初のデータ行を取り出し、セッション・ステートを初期化します。ナビゲーション・ボタンで開始した後続リクエストでは、前の HTTP リクエストのセッション・ステートが組み合されるため、問合せで次に取り出すレコードを制御する変数をアプリケーションに設定できます。
- アプリケーションの状態によって値が決まる一連の条件に基づいて、行のサブセットを戻す問合せを作成します。

データベース行のナビゲーションを追加するには、次の手順を実行します。

1. anyco_ui.inc ファイルを編集します。「Departments」ページに「**Next**」および「**Previous**」ナビゲーション・ボタンを追加します。ui_print_department() 関数に、フォーム属性 action の値を指定する $posturl という 2 つ目のパラメータを追加します。</table> タグを出力した後、「**Next**」および「**Previous**」ボタンの HTML フォーム・タグをインクルードします。

```
<?php // File: anyco_ui.inc
...
function ui_print_department($dept, $posturl)
{
  ...
    echo <<<END
  </tr>
  </table>
  <form method="post" action="$posturl">
  <input type="submit" value="< Previous" name="prevdept">
  <input type="submit" value="Next >"     name="nextdept">
```

```
  </form>
END;
  }
}

?>
```

2. anyco.php ファイルを編集します。「**Next**」または「**Previous**」ボタンを使用してページを起動したかどうかを検出し、セッション・ステートを追跡するには、PHP 関数 session_start() をコールし、construct_departments() という関数を作成します。

   データベース・アクセス・ロジックを新しい construct_departments() 関数に移動して変更します。この関数は、ナビゲーションが実行されたかどうかを検出し、セッション・ステートを管理し、処理するデータベース・アクセス・レイヤーの副問合せを定義し、db_get_page_data() 関数に接続してコールします。
   ファイルは、次のようになります。

```
<?php // File: anyco.php

require('anyco_cn.inc');
require('anyco_db.inc');
require('anyco_ui.inc');

session_start();
construct_departments();

function construct_departments()
{
  if (isset($_SESSION['currentdept']) &&
      isset($_POST['prevdept']) &&
      $_SESSION['currentdept'] > 1) {
    $current = $_SESSION['currentdept'] - 1;
  }
  elseif (isset($_SESSION['currentdept']) &&
          isset($_POST['nextdept'])) {
    $current = $_SESSION['currentdept'] + 1;
  }
  elseif (isset($_POST['showdept']) &&
          isset($_SESSION['currentdept'])) {
    $current = $_SESSION['currentdept'];
  }
  else {
    $current = 1;
  }

  $query = 'SELECT department_id, department_name,
                   manager_id, location_id
            FROM   departments
            ORDER BY department_id asc';

  $conn = db_connect();

  $dept = db_get_page_data($conn, $query, $current, 1);
  $deptid = $dept[0]['DEPARTMENT_ID'];

  $_SESSION['currentdept'] = $current;

  ui_print_header('Department');
  ui_print_department($dept[0], $_SERVER['SCRIPT_NAME']);
  ui_print_footer(date('Y-m-d H:i:s'));
}

?>
```

construct_departments() 関数の先頭にある if および elseif 構文は、HTTP ポスト・リクエストでナビゲーション・ボタンを使用してページを処理したかどうかを検出し、セッション・ステートに currentdept の数値が設定されているかどうかを追跡するために使用されています。変数 $current は状況によって異なり、「Previous」ボタンをクリックすると 1 つ減分され、「**Next**」ボタンをクリックすると 1 つ増分されます。それ以外の場合は、現行の部門に設定されるか、または初めて処理に対して 1 に初期化されます。

department_id のすべての部門行を昇順で取得するように問合せが形成されています。ORDER BY 句は、ナビゲーション・ロジックに不可欠な部分です。この問合せは、多くの行で構成されているページを取得する db_get_page_data() 関数内で副問合せとして使用されています。取得するページのページ当たりの行数は、db_get_page_data() 関数の 4 つ目の引数として指定されています。データベースに接続した後、指定した問合せで取得した行セットを取り出すために db_get_page_data() がコールされています。db_get_page_data() 関数には、接続リソース、問合せ文字列、必要なデータ行の次ページの最初の行を示す $current の値およびページ当たりの行数（この例では 1 ページ当たり 1 行）が指定されています。

1 ページ分の行を取得するために db_get_page_data() がコールされた後、$current() の値がアプリケーションのセッション・ステートに保存されています。

ページのヘッダーとフッターを出力する間に、最近フェッチした部門行を表示するために ui_print_department() 関数がコールされています。ui_print_department() 関数は、$_SERVER['SCRIPT_NAME'] を使用して、$posturl パラメータに現行の PHP スクリプト名を指定します。これで、HTML フォームにアクション属性が設定され、「**Next**」または「**Previous**」ボタンをクリックするたびに、anyco.php ファイルがコールされます。

3. anyco_db.inc ファイルを編集します。行のサブセットを問い合せるように db_get_page_data() 関数を実装します。

```
// Return subset of records
function db_get_page_data($conn, $q1, $current = 1,
                $rowsperpage = 1, $bindvars = array())
{
  // This query wraps the supplied query, and is used
  // to retrieve a subset of rows from $q1
  $query = 'SELECT *
            FROM (SELECT A.*, ROWNUM AS RNUM
                  FROM ('.$q1.') A
                  WHERE ROWNUM <= :LAST)
            WHERE :FIRST <= RNUM';

  // Set up bind variables.
  array_push($bindvars, array('FIRST', $current, -1));
  array_push($bindvars,
             array('LAST', $current+$rowsperpage-1, -1));

  $r = db_do_query($conn, $query, $bindvars);
  return($r);
}
```

db_get_page_data() 関数で問合せの構造を使用すると、一連の（1 ページ分の）データベース行をナビゲートできます。

$q1 に指定した問合せは、次の副問合せ内に副問合せとしてネストされています。

```
SELECT A.*, ROWNUM AS RNUM FROM $q1 WHERE ROWNUM <= :LAST
```

$q1 に指定した問合せは、順序づけられた一連の行を取り出します。これらの行は、最初の行から次ページのサイズ（$rowsperpage）までのすべての行を戻す囲み問合せでフィルタ処理されています。Oracle の ROWNUM 関数（または疑似列）が、$q1 に指定した問合せによって戻される各行に対して 1 で始まる整数を戻すため、この操作を実行できます。

副問合せを囲む問合せ $q1 によって戻される行セットは、次の最も外側の問合せの条件で再度フィルタ処理されています。

```
WHERE :FIRST <= RNUM
```

この条件によって、:FIRST の値（$current の値）より前にある行が最終的な行セットから除外されます。この問合せを使用すると、最初の行が $current 値で決まり、ページ・サイズが $rowsperpage 値で決まる行セットをナビゲートできます。

$current 値は、:FIRST というバインド変数に関連付けられています。式 $current+$rowsperpage-1 によって、:LAST バインド変数に関連付けられている値が設定されます。

4. アプリケーションに対して行った変更をテストするには、変更後のファイルを保存し、Web ブラウザに次の URL を入力します。

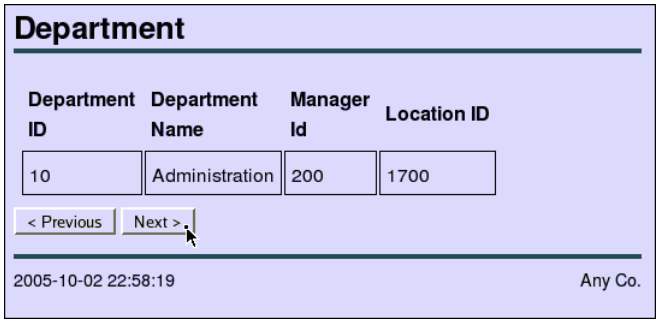
   Windows の場合 :

   ```
   http://localhost/chap4/anyco.php
   ```

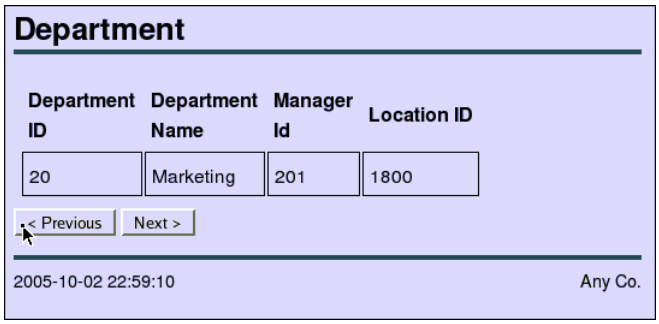
   Linux の場合 :

   ```
   http://localhost/~<username>/chap4/anyco.php
   ```

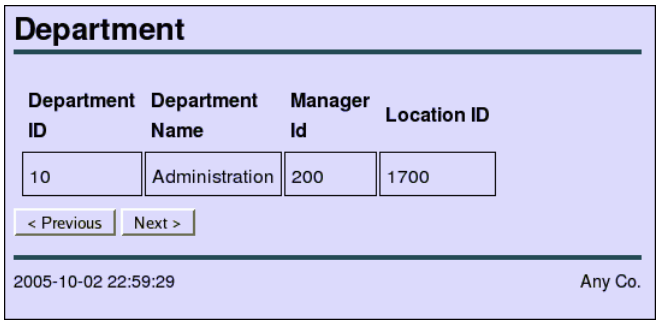
   anyco.php ページをリクエストすると、DEPARTMENT 表の最初のレコードである「Administration」部門が表示されます。



5. 次の部門レコード（「Marketing」）にナビゲートするには、「**Next**」をクリックします。



6. 最初の部門レコード（「Administration」）に戻るには、**「Previous」**をクリックします。



必要に応じて、**「Next」**および**「Previous」**をクリックして DEPARTMENTS 表の他のレコードにナビゲートして、アプリケーションのテストおよび試用を継続して行うことができます。

注意： DEPARTMENTS 表の最後のレコードを越えてナビゲートすると、エラーが発生します。エラー処理の詳細は、第 5 章の「エラー・リカバリの追加」を参照してください。

ROWNUM と ROW_NUMBER()

ハードコードされた問合せで PHP 関数を記述する場合、戻される行数を制限する方法としては ROW_NUMBER() 関数の方が簡単なことがあります。たとえば、すべての従業員の姓を戻す問合せがあるとします。

```
SELECT last_name FROM employees ORDER BY last_name;
```

次のように記述すると、51 ～ 100 までの行を選択できます。

```
SELECT last_name FROM
  SELECT last_name, ROW_NUMBER() OVER (ORDER BY last_name R FROM employees)
  where R BETWEEN 51 AND 100;
```

基本的な「Departments」ページの拡張

次の追加情報が含まれるように、「Departments」ページを拡張します。

- 部門のマネージャの名前
- 部門に配属されている従業員の数
- 部門の場所を識別する国名

これらの追加情報は、DEPARTMENTS、EMPLOYEES、LOCATIONS、COUNTRIES の各表間で結合操作を実行するように問合せを変更することによって取得できます。

「Departments」ページを拡張するには、次の手順を実行します。

1. anyco_ui.inc ファイルを編集します。「Manager ID」および「Location ID」への参照をそれぞれ「Manager Name」および「Location」に置き換え、「Department Name」の後に「Number of Employees」フィールドを挿入して、ui_print_department() 関数を変更します。表のヘッダーおよびデータ・フィールドに必要な変更を行います。この関数は、次のようになります。

```
function ui_print_department($dept, $posturl)
{
  if (!$dept) {
    echo '<p>No Department found</p>';
  }
  else {
    echo <<<END
```

```
  <table>
  <tr>
    <th>Department<br>ID</th>
    <th>Department<br>Name</th>
    <th>Number of<br>Employees</th>
    <th>Manager<br>Name</th>
    <th>Location</th>
  </tr>
  <tr>
END;
  echo '<td>'.htmlentities($dept['DEPARTMENT_ID']).'</td>';
  echo '<td>'.htmlentities($dept['DEPARTMENT_NAME']).'</td>';
  echo '<td>'.htmlentities($dept['NUMBER_OF_EMPLOYEES']).'</td>';
  echo '<td>'.htmlentities($dept['MANAGER_NAME']).'</td>';
  echo '<td>'.htmlentities($dept['COUNTRY_NAME']).'</td>';
  echo <<<END
  </tr>
  </table>
  <form method="post" action="$posturl">
  <input type="submit" value="< Previous" name="prevdept">
  <input type="submit" value="Next >"     name="nextdept">
  </form>
END;
  }
}
```

2. anyco.php ファイルを編集します。construct_departments() の問合せ文字列を次のように置き換えます。

```
$query =
  "SELECT d.department_id, d.department_name,
       substr(e.first_name,1,1)||'. '|| e.last_name as manager_name,
       c.country_name, count(e2.employee_id) as number_of_employees
   FROM  departments d, employees e, locations l,
         countries c, employees e2
   WHERE d.manager_id = e.employee_id
   AND d.location_id = l.location_id
   AND d.department_id = e2.department_id
   AND l.country_id = c.country_id
   GROUP BY d.department_id, d.department_name,
         substr(e.first_name,1,1)||'. '||e.last_name,
         c.country_name
   ORDER BY d.department_id ASC";
```

この文では SQL リテラル文字列が一重引用符で囲まれているため、文の記述を簡略化するために問合せ文字列は二重引用符で囲まれています。

3. ファイルへの変更を保存し、Web ブラウザに次の URL を入力して、変更をテストします。

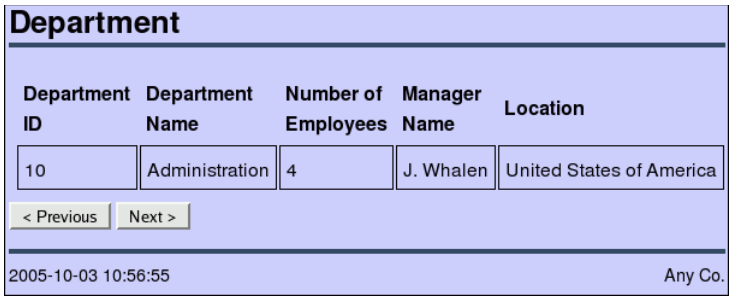
Windows の場合 :

```
http://localhost/chap4/anyco.php
```

Linux の場合 :

```
http://localhost/~<username>/chap4/anyco.php
```

Web ページの結果は、次のような出力になります。



5

データの更新

この章では、従業員レコードを挿入、更新および削除できるフォームを使用して Anyco HR アプリケーションを拡張します。

- 基本的な「Employees」ページの構築
- 基本的な「Employees」ページの拡張
- 「Departments」と「Employees」の結合
- エラー・リカバリの追加
- 追加のエラー処理

基本的な「Employees」ページの構築

この項では、基本的な「Employees」ページが含まれるようにアプリケーションを拡張します。

従業員レコードを表示するには、次の手順を実行します。

1. chap5 ディレクトリを作成し、chap4 からアプリケーション・ファイルをコピーし、新しく作成したディレクトリに移動します。

   Windows の場合 :

   ```
   mkdir c:¥program files¥Zend¥Apache2¥htdocs¥chap5
   cd c:¥program files¥Zend¥Apache2¥htdocs¥chap5
   copy ..¥chap4¥* .
   ```

   Linux の場合 :

   ```
   mkdir $HOME/public_html/chap5
   cd $HOME/public_html/chap5
   cp ../chap4/* .
   ```

2. anyco.php ファイルを編集します。construct_employees() 関数を追加します。この関数は、従業員問合せを作成し、db_do_query() 関数をコールしてその問合せを実行し、ui_print_employees() 関数を使用してその結果を出力します。

   ```
   function construct_employees()
   {
     $query =
     "SELECT employee_id,
       substr(first_name,1,1) || '.  '|| last_name as employee_name,
       hire_date,
       to_char(salary, '9999G999D99') as salary,
       nvl(commission_pct,0) as commission_pct
      FROM   employees
      ORDER BY employee_id asc";

     $conn = db_connect();
     $emp = db_do_query($conn, $query);

     ui_print_header('Employees');
     ui_print_employees($emp);
     ui_print_footer(date('Y-m-d H:i:s'));
   }
   ```

   この問合せではバインド変数を使用しないため、db_do_query() コールに $bindargs パラメータを渡す必要はありません。db_do_query() を宣言すると、デフォルト値（空配列）が自動的に指定されます。PHP では、関数に様々な数のパラメータを含めることができます。

3. anyco.php ファイルを編集します。construct_departments() へのコールを construct_employees() へのコールに置き換えます。

   ```
   <?php // File: anyco.php

   require('anyco_cn.inc');
   require('anyco_db.inc');
   require('anyco_ui.inc');

   session_start();
   construct_employees();
   ...
   ?>
   ```

4. anyco_ui.inc ファイルを編集します。ui_print_employees() 関数を追加して、HTML 表の従業員データの表示を実装します。

```
function ui_print_employees($employeerecords)
{
  if (!$employeerecords) {
    echo '<p>No Employee found</p>';
  }
  else {
    echo <<<END
  <table>
  <tr>
    <th>Employee<br>ID</th>
    <th>Employee<br>Name</th>
    <th>Hiredate</th>
    <th>Salary</th>
    <th>Commission<br>(%)</th>
  </tr>
END;
    // Write one row per employee
    foreach ($employeerecords as $emp) {
      echo '<tr>';
      echo '<td align="right">'.
            htmlentities($emp['EMPLOYEE_ID']).'</td>';
      echo '<td>'.htmlentities($emp['EMPLOYEE_NAME']).'</td>';
      echo '<td>'.htmlentities($emp['HIRE_DATE']).'</td>';
      echo '<td align="right">'.
            htmlentities($emp['SALARY']).'</td>';
      echo '<td align="right">'.
            htmlentities($emp['COMMISSION_PCT']).'</td>';
      echo '</tr>';
    }
    echo <<<END
  </table>
END;
  }
}
```

5. anyco.php ファイルおよび anyco_ui.inc ファイルへの変更を保存します。Web ブラウザに次の URL を入力して、これらの変更の結果をテストします。

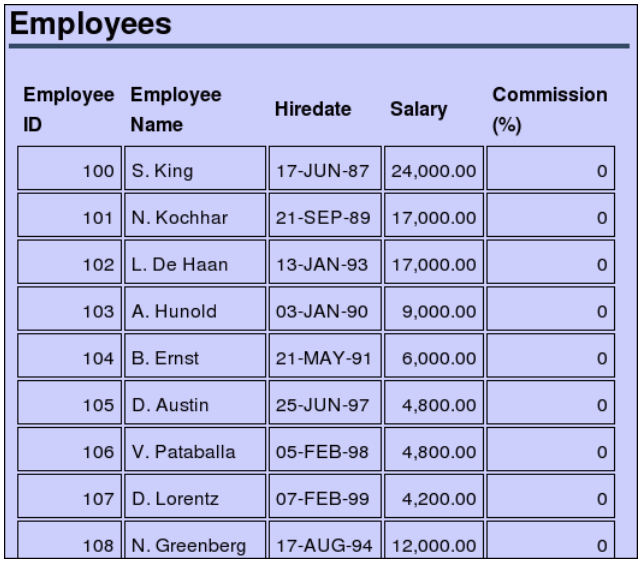
   Windows の場合 :

   http://localhost/chap5/anyco.php

   Linux の場合 :

   http://localhost/~<username>/chap5/anyco.php

結果ページを調べ、下へスクロールして、ページに表示されているすべての従業員レコードを参照します。



Employees

Employee ID	Employee Name	Hiredate	Salary	Commission (%)
100	S. King	17-JUN-87	24,000.00	0
101	N. Kochhar	21-SEP-89	17,000.00	0
102	L. De Haan	13-JAN-93	17,000.00	0
103	A. Hunold	03-JAN-90	9,000.00	0
104	B. Ernst	21-MAY-91	6,000.00	0
105	D. Austin	25-JUN-97	4,800.00	0
106	V. Pataballa	05-FEB-98	4,800.00	0
107	D. Lorentz	07-FEB-99	4,200.00	0
108	N. Greenberg	17-AUG-94	12,000.00	0

基本的な「Employees」ページの拡張

この項では、従業員レコードを操作できるように基本的な「Employees」ページを拡張します。

従業員レコードを操作できるようにするには、次の手順を実行します。

1. anyco.php ファイルを編集します。construct_employees() コールを、従業員レコードの表示、挿入、更新および削除のリクエストを管理するフォーム・ハンドラ制御ロジックに置き換えます。

```
<?php // File: anyco.php

require('anyco_cn.inc');
require('anyco_db.inc');
require('anyco_ui.inc');

session_start();
// Start form handler code
if (isset($_POST['insertemp'])) {
  construct_insert_emp();
}
elseif (isset($_POST['saveinsertemp'])) {
  insert_new_emp();
}
elseif (isset($_POST['modifyemp'])) {
  construct_modify_emp();
}
elseif (isset($_POST['savemodifiedemp'])) {
  modify_emp();
}
elseif (isset($_POST['deleteemp'])) {
  delete_emp();
}
else {
  construct_employees();
}

...
```

2. anyco.php ファイルを編集します。construct_insert_emp() 関数を追加します。

```
function construct_insert_emp()
{
  $conn = db_connect();

  $query = "SELECT job_id, job_title
            FROM jobs
            ORDER BY job_title ASC";
  $jobs = db_do_query($conn, $query,
                      OCI_FETCHSTATEMENT_BY_COLUMN);

  $query = "SELECT sysdate FROM dual";
  $date = db_do_query($conn, $query,
                      OCI_FETCHSTATEMENT_BY_COLUMN);
  $emp = array(
    'DEPARTMENT_ID' => 10,      // Default to department 10
    'HIRE_DATE' => $date['SYSDATE'][0],
    'ALLJOBIDS' => $jobs['JOB_ID'],
    'ALLJOBTITLES' => $jobs['JOB_TITLE']
    );

  ui_print_header('Insert New Employee');
  ui_print_insert_employee($emp, $_SERVER['SCRIPT_NAME']);
  // Note: The two kinds of date used:
  // 1) SYSDATE for current date of the database system, and
  // 2) The PHP date for display in the footer of each page
  ui_print_footer(date('Y-m-d H:i:s'));
}
```

construct_insert_emp() 関数は、2 つの問合せを実行して、「Insert New Employee」フォームへの移入に使用するデフォルト・データを取得します。このフォームは、ui_print_insert_employee() 関数によって表示されます。

JOBS 表の $query は、既存のすべてのジョブ ID およびその説明のリストを取得して、ui_print_insert_employee() 関数によって生成された HTML フォームにジョブ・タイプ選択用のリストを作成します。

SYSDATE が使用されている $query は、新しい従業員のデフォルトの雇用日を設定するために、現在のデータベース日時を取得します。

このアプリケーション・コードでは、2 種類の日付が使用されています。PHP の date() 関数は、ページ・フッターに日時を出力します。Oracle の SYSDATE 関数は、「Employees」ページの雇用日フィールドに表示するデフォルトの日時を取得し、テキストが正しいデータベース形式で入力されるようにします。

2 つの db_do_query() 関数コールには、問合せの戻り型が列値の配列になるように指定する追加のパラメータ値 OCI_FETCHSTATEMENT_BY_COLUMN が含まれています。

3. anyco.php ファイルを編集します。insert_new_emp() 関数を追加して、EMPLOYEES 表に従業員レコードを挿入します。

```
function insert_new_emp()
{
  $newemp = $_POST;
  $statement =
    "INSERT INTO employees
        (employee_id, first_name, last_name, email, hire_date,
         job_id, salary, commission_pct, department_id)
     VALUES (employees_seq.nextval, :fnm, :lnm, :eml, :hdt, :jid,
             :sal, :cpt, :did)";

  $conn = db_connect();
  $emailid = $newemp['firstname'].$newemp['lastname'];
```

```
  $bindargs = array();
  array_push($bindargs, array('FNM', $newemp['firstname'], -1));
  array_push($bindargs, array('LNM', $newemp['lastname'], -1));
  array_push($bindargs, array('EML', $emailid, -1));
  array_push($bindargs, array('HDT', $newemp['hiredate'], -1));
  array_push($bindargs, array('JID', $newemp['jobid'], -1));
  array_push($bindargs, array('SAL', $newemp['salary'], -1));
  array_push($bindargs, array('CPT', $newemp['commpct'], -1));
  array_push($bindargs, array('DID', $newemp['deptid'], -1));

  $r = db_execute_statement($conn, $statement, $bindargs);
  construct_employees();
}
```

db_execute_statement() 関数の戻り値は無視され、変数にも割り当てられません。関数の結果に対してアクションが実行されないためです。

4. anyco.php ファイルを編集します。construct_modify_emp() 関数を追加して、従業員レコードを更新するための HTML フォームを構築します。

```
function construct_modify_emp()
{
  $empid = $_POST['emprec'];
  $query =
    "SELECT employee_id, first_name, last_name, email, hire_date,
            salary, nvl(commission_pct,0) as commission_pct
     FROM   employees
     WHERE  employee_id = :empid";

  $conn = db_connect();
  $bindargs = array();
  array_push($bindargs, array('EMPID', $empid, -1));

  $emp = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_ROW,
                                    $bindargs);

  ui_print_header('Modify Employee ');
  ui_print_modify_employee($emp[0], $_SERVER['SCRIPT_NAME']);
  ui_print_footer(date('Y-m-d H:i:s'));
}
```

5. anyco.php ファイルを編集します。modify_emp() 関数を追加して、更新フォーム・フィールドの値で EMPLOYEES 表の従業員レコードを更新します。

```
function modify_emp()
{
  $newemp = $_POST;
  $statement =
    "UPDATE employees
     SET   first_name = :fnm, last_name = :lnm, email = :eml,
           salary = :sal, commission_pct = :cpt
     WHERE employee_id = :eid";

  $conn = db_connect();
  $bindargs = array();
  array_push($bindargs, array('EID', $newemp['empid'], -1));
  array_push($bindargs, array('FNM', $newemp['firstname'], -1));
  array_push($bindargs, array('LNM', $newemp['lastname'], -1));
  array_push($bindargs, array('EML', $newemp['email'], -1));
  array_push($bindargs, array('SAL', $newemp['salary'], -1));
  array_push($bindargs, array('CPT', $newemp['commpct'], -1));

  $r = db_execute_statement($conn, $statement, $bindargs);
  construct_employees();
}
```

6. anyco.php ファイルを編集します。delete_emp() 関数を追加して、EMPLOYEES 表から従業員レコードを削除します。

```
function delete_emp()
{
  $empid = $_POST['emprec'];
  $statement = "DELETE FROM employees
                WHERE employee_id = :empid";

  $conn = db_connect();
  $bindargs = array();
  array_push($bindargs, array('EMPID', $empid, 10));
  $r = db_execute_statement($conn, $statement, $bindargs);

  construct_employees();
}
```

7. anyco.php ファイルを編集します。construct_employees() 関数で、db_do_query() コールの最後のパラメータとして OCI_FETCHSTATEMENT_BY_ROW を指定し、ui_print_employees() コールの 2 つ目のパラメータとして $_SERVER['SCRIPT_NAME'] を指定します。
ファイルは、次のようになります。

```
function construct_employees()
{
  $query =
  "SELECT employee_id,
    substr(first_name,1,1) || '.  '|| last_name as employee_name,
    hire_date,
    to_char(salary, '9999G999D99') as salary,
    nvl(commission_pct,0) as commission_pct
   FROM   employees
   ORDER BY employee_id asc";

  $conn = db_connect();
  $emp = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_ROW);

  ui_print_header('Employees');
  ui_print_employees($emp, $_SERVER['SCRIPT_NAME']);
  ui_print_footer(date('Y-m-d H:i:s'));
}
```

8. anyco_db.inc ファイルを編集します。
db_do_query() 関数に 3 つ目のパラメータとして $resulttype を追加します。コール元が出力タイプを選択できるように、oci_fetch_all() コールの最後のパラメータ値 OCI_FETCHSTATEMENT_BY_ROW を変数に置き換えます。

```
function db_do_query($conn, $statement, $resulttype,
                     $bindvars = array())
{
  $stid = oci_parse($conn, $statement);

  ...

  $r = oci_fetch_all($stid, $results, null, null, $resulttype);
  return($results);
}
```

9. anyco_db.inc ファイルを編集します。db_get_page_data() 関数内で、db_do_query() コールの 3 つ目のパラメータ値として OCI_FETCHSTATEMENT_BY_ROW を挿入します。

```
function db_get_page_data($conn, $q1, $current = 1,
                          $rowsperpage = 1, $bindvars = array())
{

  ...

  $r = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_ROW, $bindvars);
  return($r);
}
```

10. anyco_db.inc ファイルを編集します。db_execute_statement() 関数を追加して、INSERT 文などデータ操作文を実行します。

```
function db_execute_statement($conn, $statement, $bindvars = array())
{
  $stid = oci_parse($conn, $statement);
  if (!$stid) {
    db_error($conn, __FILE__, __LINE__);
  }

  // Bind parameters
  foreach ($bindvars as $b) {
    // create local variable with caller specified bind value
    $$b[0] = $b[1];
    $r = oci_bind_by_name($stid, ":$b[0]", $$b[0], $b[2]);
    if (!$r) {
      db_error($stid, __FILE__, __LINE__);
    }
  }

  $r = oci_execute($stid);
  if (!$r) {
    db_error($stid, __FILE__, __LINE__);
  }
  return($r);
}
```

11. anyco_ui.inc ファイルを編集します。従業員行を含む HTML フォームが生成されるように、ui_print_employees() 関数を変更します。この関数は、次のようになります。

```
function ui_print_employees($employeerecords, $posturl)
{
  if (!$employeerecords) {
    echo '<p>No Employee found</p>';
  }
  else {
    echo <<<END
  <form method="post" action="$posturl">
  <table>
  <tr>
    <th> </th>
    <th>Employee<br>ID</th>
    <th>Employee<br>Name</th>
    <th>Hiredate</th>
    <th>Salary</th>
    <th>Commission<br>(%)</th>
  </tr>
END;
    // Write one row per employee
    foreach ($employeerecords as $emp) {
```

```
      echo '<tr>';
      echo '<td><input type="radio" name="emprec" value="'.
        htmlentities($emp['EMPLOYEE_ID']).'"></td>';
      echo '<td align="right">'.
        htmlentities($emp['EMPLOYEE_ID']).'</td>';
      echo '<td>'.htmlentities($emp['EMPLOYEE_NAME']).'</td>';
      echo '<td>'.htmlentities($emp['HIRE_DATE']).'</td>';
      echo '<td align="right">'.
        htmlentities($emp['SALARY']).'</td>';
      echo '<td align="right">'.
        htmlentities($emp['COMMISSION_PCT']).'</td>';
      echo '</tr>';
    }
    echo <<<END
  </table>
  <input type="submit" value="Modify" name="modifyemp">
  <input type="submit" value="Delete" name="deleteemp">
    
  <input type="submit" value="Insert new employee"
         name="insertemp">
  </form>
END;
  }
}
```

変更または削除するレコードを選択できるように、各行の最初の列にラジオ・ボタンが表示されています。

12. anyco_ui.inc ファイルを編集します。ui_print_insert_employee() 関数を追加して、新しい従業員データを入力するためのフォームを生成します。

```
function ui_print_insert_employee($emp, $posturl)
{
  if (!$emp) {
    echo "<p>No employee details found</p>";
  }
  else {
    $deptid = htmlentities($emp['DEPARTMENT_ID']);
    $hiredate = htmlentities($emp['HIRE_DATE']);

    echo <<<END
  <form method="post" action="$posturl">
  <table>
    <tr>
      <td>Department ID</td>
      <td><input type="text" name="deptid" value="$deptid"
                 size="20"></td>
    </tr>
    <tr>
      <td>First Name</td>
      <td><input type="text" name="firstname" size="20"></td>
    </tr>
    <tr>
      <td>Last Name</td>
      <td><input type="text" name="lastname" size="20"></td>
    </tr>
    <tr>
      <td>Hiredate</td>
      <td><input type="text" name="hiredate" value="$hiredate"
                 size="20"></td>
    </tr>
    <tr>
      <td>Job</td>
       <td><select name="jobid">
```

```
END;
    // Write the list of jobs
    for ($i = 0; $i < count($emp['ALLJOBIDS']); $i++)
    {
      echo '<option
             label="'.htmlentities($emp['ALLJOBTITLES'][$i]).'"'.
           ' value="'.htmlentities($emp['ALLJOBIDS'][$i]).'">'.
           htmlentities($emp['ALLJOBTITLES'][$i]).'</option>';
    }
    echo <<<END
      </select>
      </td>
    </tr>
    <tr>
      <td>Salary</td>
      <td><input type="text" name="salary" value="1"
                 size="20"></td>
    </tr>
    <tr>
      <td>Commission (%)</td>
      <td><input type="text" name="commpct" value="0"
                 size="20"></td>
    </tr>
  </table>
    <input type="submit" value="Save" name="saveinsertemp">
    <input type="submit" value="Cancel" name="cancel">
  </form>
END;
  }
}
```

13. anyco_ui.inc ファイルを編集します。ui_print_modify_employee() 関数を追加して、従業員レコードを更新するためのフォームを生成します。

```
function ui_print_modify_employee($empdetails, $posturl)
{
  if (!$empdetails) {
    echo '<p>No Employee record selected</p>';
  }
  else {
    $fnm = htmlentities($empdetails['FIRST_NAME']);
    $lnm = htmlentities($empdetails['LAST_NAME']);
    $eml = htmlentities($empdetails['EMAIL']);
    $sal = htmlentities($empdetails['SALARY']);
    $cpt = htmlentities($empdetails['COMMISSION_PCT']);
    $eid = htmlentities($empdetails['EMPLOYEE_ID']);

    echo <<<END
  <form method="post" action="$posturl">
  <table>
    <tr>
      <td>Employee ID</td>
      <td>$eid</td></tr>
    <tr>
      <td>First Name</td>
      <td><input type="text" name="firstname" value="$fnm"></td>
    </tr>
    <tr>
      <td>Last Name</td>
      <td><input type="text" name="lastname" value="$lnm"></td>
    </tr>
    <tr>
      <td>Email Address</td>
```

```
      <td><input type="text" name="email" value="$eml"></td>
    </tr>
    <tr>
      <td>Salary</td>
      <td><input type="text" name="salary" value="$sal"></td>
    </tr>
    <tr>
      <td>Commission (%)</td>
      <td><input type="text" name="commpct" value="$cpt"></td>
    </tr>
  </table>
  <input type="hidden" value="{$empdetails['EMPLOYEE_ID']}"
         name="empid">
  <input type="submit" value="Save" name="savemodifiedemp">
  <input type="submit" value="Cancel" name="cancel">
  </form>
END;
  }
}
```

14. Anyco アプリケーション・ファイルへの変更を保存し、Web ブラウザに次の URL を入力して、変更をテストします。

    Windows の場合 :

    `http://localhost/chap5/anyco.php`

    Linux の場合 :

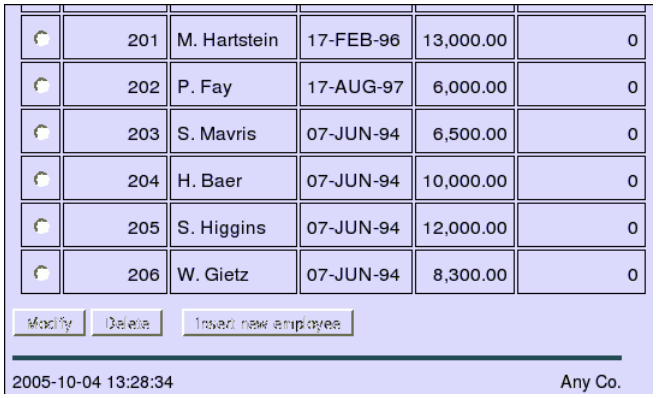
    `http://localhost/~<username>/chap5/anyco.php`

    各行にラジオ・ボタンが付いたすべての従業員のリストが表示されます。

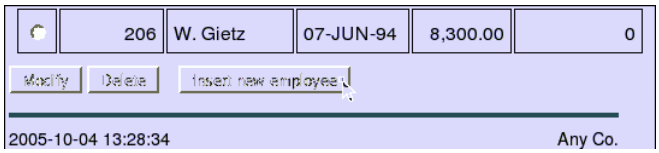
Employees

	Employee ID	Employee Name	Hiredate	Salary	Commission (%)
○	100	S. King	17-JUN-87	24,000.00	0
○	101	N. Kochhar	21-SEP-89	17,000.00	0
○	102	L. De Haan	13-JAN-93	17,000.00	0
○	103	A. Hunold	03-JAN-90	9,000.00	0
○	104	B. Ernst	21-MAY-91	6,000.00	0
○	105	D. Austin	25-JUN-97	4,800.00	0
○	106	V. Pataballa	05-FEB-98	4,800.00	0
○	107	D. Lorentz	07-FEB-99	4,200.00	0

「Employees」ページの下部にスクロールして、**「Modify」**、**「Delete」**、**「Insert new employee」**の各ボタンを表示します。



15. 新しい従業員レコードを挿入するには、**「Insert new employee」**をクリックします。



従業員レコードを作成または変更する場合は、データベース定義に従って、給与を 0（ゼロ）より大きい値にし、コミッションを 1 より小さい値にする必要があります。コミッションは、小数点以下 2 桁に丸められます。「Insert New Employee」ページでは、「Department ID」フィールドは 10（デフォルト）、「Hiredate」は現在の日付（デフォルトのデータベース日付書式）、「Salary」は 1、「Commission (%)」は 0（ゼロ）になっています。次のフィールド値を入力します。

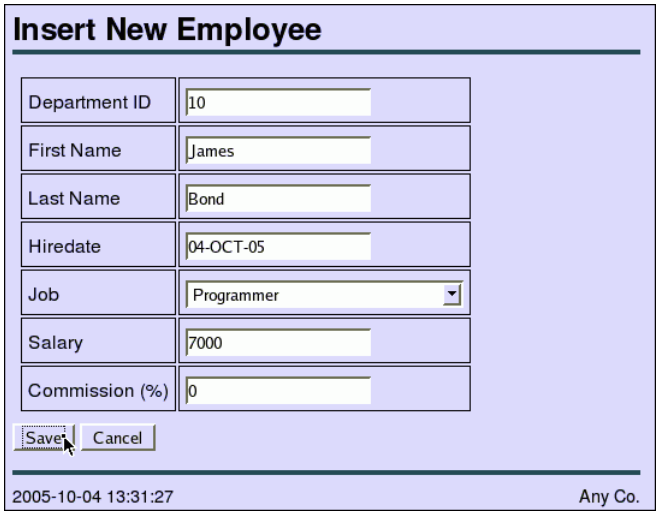
First Name: James

Last Name: Bond

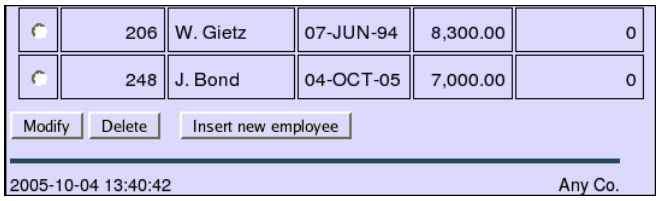
Job: リストから「Programmer」を選択します。

Salary: 1 を 7000 に置き換えます。

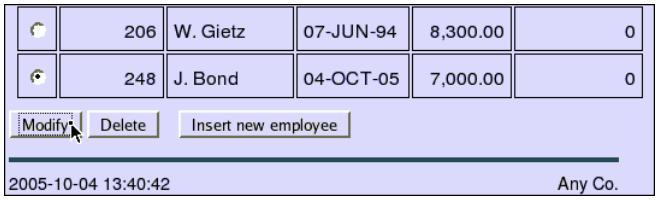
「Save」をクリックします。



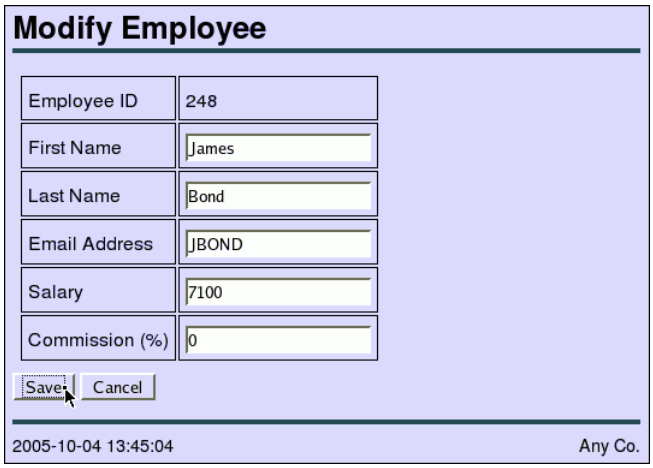
16. 新しい従業員レコードが正常に挿入されると、Web ページがリフレッシュされ、フォームにすべての従業員が表示されます。Web ページを最後のレコードまでスクロールし、新しい従業員レコードが存在していることを確認します。システムの新しいレコードに割り当てられている従業員 ID は、次の例に示す従業員 ID とは異なる場合があります。



17. 新しい従業員レコードを変更するには、そのレコードの横にあるラジオ・ボタンを選択し、**「Modify」**をクリックします。



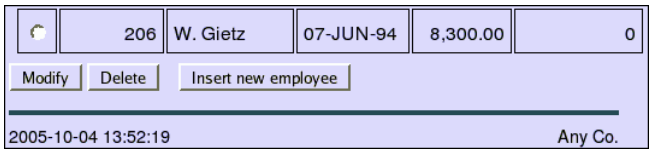
18. 「Modify Employee」ページで、「Email Address」フィールドを JBOND に変更し、「Salary」を 7100 に増やし、**「Save」**をクリックします。



19. 従業員レコードが正常に更新されると、「Employees」ページが再表示されます。最後の従業員レコードまでスクロールし、James Bond の給与が 7,100 になっていることを確認します。

20. 新しい従業員レコードを削除するには、そのレコードの横にあるラジオ・ボタンを選択し、**「Delete」**をクリックします。

削除が正常に完了すると、削除した行は「Employees」ページに再表示された従業員レコードのリストに表示されません。



「Departments」と「Employees」の結合

この項では、「Employees」と「Departments」の両方のページにアクセスできるようにアプリケーションを変更します。

「Departments」ページと「Employees」ページを結合するには、次の手順を実行します。

1. anyco.php ファイルを編集します。construct_employees() 関数の問合せに、department_id を :did というバインド変数の値と比較する WHERE 句を含めます。これによって、一度に 1 つの部門の従業員がページに表示されます。deptid セッション・パラメータ値を取得してバインド変数に移入します。

```
$query =
 "SELECT employee_id,
        substr(first_name,1,1) || '.  '|| last_name as employee_name,
        hire_date,
        to_char(salary, '9999G999D99') as salary,
        nvl(commission_pct,0) as commission_pct
  FROM   employees
  WHERE  department_id = :did
  ORDER BY employee_id asc";

$deptid = $_SESSION['deptid'];
```

2. anyco.php ファイルを編集します。construct_employees() 関数で、バインド情報が渡されるように db_do_query() 関数のコールを更新します。

```
$conn = db_connect();

$bindargs = array();
array_push($bindargs, array('DID', $deptid, -1));

$emp = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_ROW, $bindargs);
```

3. anyco.php ファイルを編集します。construct_departments() 関数で、部門 ID をセッション・パラメータに保存します。

```
$_SESSION['currentdept'] = $current;
$_SESSION['deptid'] = $deptid;
```

「Departments」ページの現行の部門 ID がセッション・パラメータとして保存され、「Employees」ページで使用されます。

4. anyco.php ファイルを編集します。「Departments」ページおよび「Employees」ページのタイトルに出力する部門名を問い合せる関数 get_dept_name() を作成します。

```
function get_dept_name($conn, $deptid)
{
  $query =
    'SELECT department_name
     FROM   departments
     WHERE  department_id = :did';

  $conn = db_connect();
  $bindargs = array();
  array_push($bindargs, array('DID', $deptid, -1));
  $dn = db_do_query($conn, $query,OCI_FETCHSTATEMENT_BY_COLUMN, $bindargs);

  return($dn['DEPARTMENT_NAME'][0]);
}
```

5. anyco.php ファイルを編集します。「Employees」ページのヘッダーに部門名が出力されるように construct_employees() 関数を変更します。

```
$deptname = get_dept_name($conn, $deptid);
ui_print_header('Employees: '.$deptname);
```

6. anyco.php ファイルを編集します。「Departments」ページのヘッダーに部門名が出力されるように construct_departments() 関数を変更します。

```
$deptname = get_dept_name($conn, $deptid);
ui_print_header('Department: '.$deptname);
```

7. anyco.php ファイルを編集します。ui_print_insert_employee() 関数の $emp 配列に渡されたセッション・パラメータからデフォルトの部門が取得されるように、construct_insert_emp() 関数を変更します。この関数は、次のようになります。

```
function construct_insert_emp()
{
  $deptid = $_SESSION['deptid'];

  $conn = db_connect();
  $query = "SELECT job_id, job_title FROM jobs ORDER BY job_title ASC";
  $jobs = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_COLUMN);
  $query = "SELECT sysdate FROM dual";
  $date = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_COLUMN);
  $emp = array(
    'DEPARTMENT_ID' => $deptid,
    'HIRE_DATE' => $date['SYSDATE'][0],
    'ALLJOBIDS' => $jobs['JOB_ID'],
    'ALLJOBTITLES' => $jobs['JOB_TITLE']
    );
  ui_print_header('Insert New Employee');
  ui_print_insert_employee($emp, $_SERVER['SCRIPT_NAME']);
  ui_print_footer(date('Y-m-d H:i:s'));
}
```

8. anyco.php ファイルを編集します。HTML フォーム・ハンドラの最後の else 文を変更します。ハンドラは、次のようになります。

```
// Start form handler code
if (isset($_POST['insertemp'])) {
  construct_insert_emp();
}
elseif (isset($_POST['saveinsertemp'])) {
  insert_new_emp();
}
elseif (isset($_POST['modifyemp'])) {
```

```
  construct_modify_emp();
}
elseif (isset($_POST['savemodifiedemp'])) {
  modify_emp();
}
elseif (isset($_POST['deleteemp'])) {
  delete_emp();
}
elseif (   isset($_POST['showemp'])) {
  construct_employees();
}
elseif (   isset($_POST['nextdept'])
        || isset($_POST['prevdept'])
        || isset($_POST['firstdept'])
        || isset($_POST['showdept'])) {
  construct_departments();
}
else {
  construct_departments();
}
```

9. anyco_ui.inc ファイルを編集します。ui_print_department() 関数で、「Employees」ページをコールできるように HTML フォームを変更します。

```
...
<form method="post" action="$posturl">
<input type="submit" value="First" name="firstdept">
<input type="submit" value="< Previous" name="prevdept">
<input type="submit" value="Next >" name="nextdept">
   
<input type="submit" value="Show Employees" name="showemp">
</form>
...
```

10. anyco_ui.inc ファイルを編集します。ui_print_employees() 関数で、「Departments」ページをコールできるように HTML フォームを変更します。

```
...
</table>
<input type="submit" value="Modify" name="modifyemp">
<input type="submit" value="Delete" name="deleteemp">
  
<input type="submit" value="Insert new employee" name="insertemp">
  
<input type="submit" value="Return to Departments" name="showdept">
</form>
...
```

11. PHP ファイルへの変更を保存します。ブラウザで、次の URL を入力して、変更をテストします。

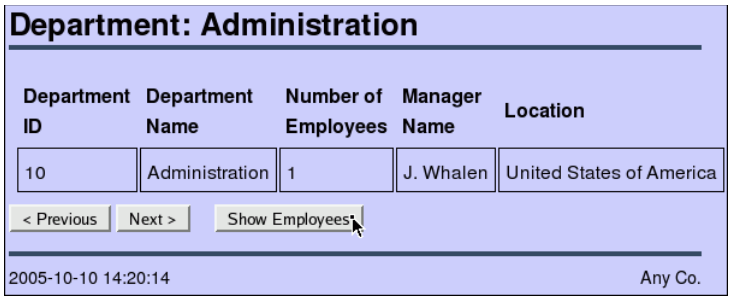
Windows の場合 :

```
http://localhost/chap5/anyco.php
```

Linux の場合 :

```
http://localhost/~<username>/chap5/anyco.php
```

「Departments」ページが表示されます。



部門の従業員のリストを表示するには、**「Show Employees」**ボタンをクリックします。

「Return to Departments」ボタンをクリックすると、「Departments」ページに戻ることができます。別の部門にナビゲートし、その従業員を表示して、「Departments」ページと「Employees」ページを切り替えるプロセスを確認します。

エラー・リカバリの追加

エラー管理は、常に設計上の重要な決定事項です。本番システムでは、エラーを分類し、様々な方法で処理する必要がある場合があります。致命的エラーは、標準の「site not available」ページまたはホームページにリダイレクトできます。新しいレコードの作成で発生したデータ・エラーは、無効なフィールドがハイライト表示された該当するフォームに戻すことができます。

ほとんどの本番システムでは、`php.ini` ファイルの `display_errors` 構成オプションを `off` に、`log_errors` 構成オプションを `on` に設定します。

PHP 出力バッファリング機能を使用して、関数の実行中にエラー・テキストをトラップできます。`ob_start()` を使用すると、エラー・テキストが画面に表示されないようにすることができます。エラーが発生した場合は、`ob_get_contents()` 関数を使用して、以前に生成されたエラー・メッセージを後で表示または分析するために文字列に格納できます。

ここで、カスタム・エラー処理関数を使用して新しいページにエラー・メッセージおよびデータベース・エラーを表示するようにアプリケーションを変更します。これによって、エラーが `db*` 関数から戻され、暗黙的に保持されます。

1. `anyco_db.inc` ファイルを編集します。出力して終了するのではなく、エラー情報を配列構造に戻すように `db_error()` 関数を変更します。この関数は、次のようになります。

```
function db_error($r = false, $file, $line)
{
  $err =  $r ? oci_error($r) : oci_error();

  if (isset($err['message'])) {
    $m = htmlentities($err['message']);
    $c = $err['code'];
  }
  else {
    $m = 'Unknown DB error';
    $c = null;
```

```
  }

  $rc = array(
    'MESSAGE' => $m,
    'CODE'    => $c,
    'FILE'    => $file,
    'LINE'    => $line
    );
  return $rc;
}
```

2. anyco_db.inc ファイルを編集します。db_error() 関数のすべてのコールに対して、$e という変数に戻り値を割り当て、各コールの後に return false; 文を追加します。

```
if (<error test>)
{
  $e = db_error(<handle>, __FILE__, __LINE__);
  return false;
}
```

<error test> パラメータおよび <handle> パラメータは、各コールに現在指定されているものと同じにしてください。__FILE__ 定数および __LINE__ 定数を使用すると、開発時に障害の場所を特定するのに役立ちます。この情報は、アプリケーションの本番デプロイメントで致命的エラーが発生した場合に備えてログに記録しておくと有効な情報です。

3. anyco_db.inc ファイルを編集します。すべての関数に $e パラメータを追加して、エラー情報を戻すことができるようにします。& 参照接頭辞を使用して、結果がコール元関数に戻されるようにします。各関数の宣言は、次のようになります。

```
function db_connect(&$e) {...}

function db_get_page_data($conn, $q1, $currrownum = 1, $rowsperpage = 1,
                          &$e, $bindvars = array()) {...}

function db_do_query($conn, $statement, $resulttype, &$e,
                     $bindvars = array()) {...}

function db_execute_statement($conn, $statement, &$e,
                              $bindvars = array()) {...}
```

4. anyco_db.inc ファイルを編集します。db_get_page_data() 関数で、エラー・パラメータ $e が渡されるように db_do_query() 関数へのコールを変更します。

```
$r = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_ROW, $e, $bindvars);
```

5. anyco_db.inc ファイルを編集します。すべての oci_* 関数コールに @ 接頭辞を追加します。次に例を示します。

```
@ $r = @oci_execute($stid);
```

@ 接頭辞を使用すると、戻された結果がそれぞれテストされるため、エラーが表示されなくなります。エラーが表示されないと、誤ったパラメータの使用が隠されてしまう可能性があるため、この項で行った変更をテストできない場合があります。@ 接頭辞は追加する必要はありませんが、追加すると、エラーが表示された場合の今後の結果に影響を及ぼす可能性があります。

6. anyco.php ファイルを編集します。エラー情報を処理する関数を作成します。

```
function handle_error($message, $err)
{
  ui_print_header($message);
  ui_print_error($err, $_SERVER['SCRIPT_NAME']);
  ui_print_footer(date('Y-m-d H:i:s'));
}
```

7. anyco.php ファイルを編集します。追加のエラー・パラメータが含まれるように、db_* 関数へのすべてのコールを変更します。

   手順 8 ～ 15 で新しい関数を完成させるため、この手順で行うコード変更はスキップできます。

   - すべての db_connect() コールを db_connect($err) に変更します。
   - すべての db_do_query() コールを変更し、4 つ目のパラメータとして $err パラメータを挿入します。たとえば、construct_employees() のコールは次のようになります。

     ```
     $emp = db_do_query($conn, $query,
                        OCI_FETCHSTATEMENT_BY_ROW, $err, $bindargs);
     ```

     コールごとに既存のパラメータ値が保持されるように、anyco.php の他の 4 つの db_do_query() コールを変更します。
   - すべての db_get_page_data() コールを変更し、5 つ目のパラメータとして $err パラメータを挿入します。

     ```
     $dept = db_get_page_data($conn, $query, $current, 1, $err);
     ```

   - すべての db_execute_statement() コールを変更し、3 つ目のパラメータとして $err パラメータを挿入します。次に例を示します。

     ```
     $r = db_execute_statement($conn, $statement, $err, $bindargs);
     ```

8. anyco.php ファイルを編集します。戻されたエラーが処理されるように construct_departments() 関数を変更します。ファイルは、次のようになります。

```
function construct_departments()
{
  if (isset($_SESSION['currentdept']) && isset($_POST['prevdept']) &&
          $_SESSION['currentdept'] > 1)
     $current = $_SESSION['currentdept'] - 1;
  elseif (isset($_SESSION['currentdept']) && isset($_POST['nextdept']))
     $current = $_SESSION['currentdept'] + 1;
  elseif (isset($_POST['showdept']) && isset($_SESSION['currentdept']))
     $current = $_SESSION['currentdept'];
  else
     $current = 1;

  $query =
    "SELECT d.department_id, d.department_name,
          substr(e.first_name,1,1)||'. '|| e.last_name as manager_name,
          c.country_name, count(e2.employee_id) as number_of_employees
     FROM   departments d, employees e, locations l,
          countries c, employees e2
     WHERE  d.manager_id    = e.employee_id
     AND    d.location_id   = l.location_id
     AND    d.department_id = e2.department_id
     AND    l.country_id    = c.country_id
     GROUP BY d.department_id, d.department_name,
            substr(e.first_name,1,1)||'. '||e.last_name, c.country_name
     ORDER BY d.department_id ASC";

  $conn = db_connect($err);

  if (!$conn) {
    handle_error('Connection Error', $err);
  }
  else {
    $dept = db_get_page_data($conn, $query, $current, 1, $err);
    if ($dept === false) {
      // Use === so empty array at end of fetch is not matched
```

```
      handle_error('Cannot fetch Departments', $err);
    } else {

      if (!isset($dept[0]['DEPARTMENT_ID']) && $current > 1) {
        // no more records so go back one

        $current--;
        $dept = db_get_page_data($conn, $query, $current, 1, $err);
      }

      $deptid = $dept[0]['DEPARTMENT_ID'];

      $_SESSION['deptid'] = $deptid;
      $_SESSION['currentdept'] = $current;

      $deptname = get_dept_name($conn, $deptid);
      ui_print_header('Department: '.$deptname);
      ui_print_department($dept[0], $_SERVER['SCRIPT_NAME']);
      ui_print_footer(date('Y-m-d H:i:s'));
    }
  }
}
```

9. anyco.php ファイルを編集します。エラーが処理されるように construct_employees() 関数を変更します。ファイルは、次のようになります。

```
function construct_employees()
{
  $query =
    "SELECT employee_id,
            substr(first_name,1,1) || '.  '|| last_name as employee_name,
            hire_date,
            to_char(salary, '9999G999D99') as salary,
            nvl(commission_pct,0) as commission_pct
     FROM   employees
     WHERE  department_id = :did
     ORDER BY employee_id asc";

  $deptid = $_SESSION['deptid'];

  $conn = db_connect($err);

  if (!$conn) {
    handle_error('Connection Error', $err);
  }
  else {
    $bindargs = array();
    array_push($bindargs, array('DID', $deptid, -1));
    $emp = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_ROW, $err,
    $bindargs);

    if (!$emp) {
      handle_error('Cannot fetch Employees', $err);
    }
    else {
      $deptname = get_dept_name($conn, $deptid);
      ui_print_header('Employees: '.$deptname);
      ui_print_employees($emp, $_SERVER['SCRIPT_NAME']);
      ui_print_footer(date('Y-m-d H:i:s'));
    }
  }
}
```

10. anyco.php ファイルを編集します。エラーが処理されるように construct_insert_emp() 関数を変更します。ファイルは、次のようになります。

```
function construct_insert_emp()
{
  $deptid = $_SESSION['deptid'];
  $conn = db_connect($err);
  if (!$conn) {
    handle_error('Connection Error', $err);
  }
  else {
    $query = "SELECT job_id, job_title FROM jobs ORDER BY job_title ASC";
    $jobs = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_COLUMN, $err);
    $query = "SELECT sysdate FROM dual";
    $date = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_COLUMN, $err);

    $emp = array(
      'DEPARTMENT_ID' => $deptid,
      'HIRE_DATE' => $date['SYSDATE'][0],
      'ALLJOBIDS' => $jobs['JOB_ID'],
      'ALLJOBTITLES' => $jobs['JOB_TITLE']
      );

    ui_print_header('Insert New Employee');
    ui_print_insert_employee($emp, $_SERVER['SCRIPT_NAME']);
    ui_print_footer(date('Y-m-d H:i:s'));
  }
}
```

11. anyco.php ファイルを編集します。エラーが処理されるように insert_new_emp() 関数を変更します。ファイルは、次のようになります。

```
function insert_new_emp()
{
  $statement =
    'INSERT INTO employees
                (employee_id, first_name, last_name, email, hire_date,
                job_id, salary, commission_pct, department_id)
     VALUES (employees_seq.nextval, :fnm, :lnm, :eml, :hdt,
            :jid, :sal, :cpt, :did)';

  $newemp = $_POST;

  $conn = db_connect($err);
  if (!$conn) {
    handle_error('Connect Error', $err);
  }
  else {
    $emailid = $newemp['firstname'].$newemp['lastname'];

    $bindargs = array();
    array_push($bindargs, array('FNM', $newemp['firstname'], -1));
    array_push($bindargs, array('LNM', $newemp['lastname'], -1));
    array_push($bindargs, array('EML', $emailid, -1));
    array_push($bindargs, array('HDT', $newemp['hiredate'], -1));
    array_push($bindargs, array('JID', $newemp['jobid'], -1));
    array_push($bindargs, array('SAL', $newemp['salary'], -1));
    array_push($bindargs, array('CPT', $newemp['commpct'], -1));
    array_push($bindargs, array('DID', $newemp['deptid'], -1));

    $r = db_execute_statement($conn, $statement, $err, $bindargs);
    if ($r) {
      construct_employees();
    }
```

```
    else {
      handle_error('Cannot insert employee', $err);
    }
  }
}
```

12. anyco.php 関数を編集します。エラーが処理されるように construct_modify_emp() 関数を変更します。ファイルは、次のようになります。

```
function construct_modify_emp()
{
  if (!isset($_POST['emprec'])) { // User did not select a record
    construct_employees();
  }
  else {
    $empid = $_POST['emprec'];

    $query =
      "SELECT employee_id, first_name, last_name, email, hire_date,
              salary, nvl(commission_pct,0) as commission_pct
       FROM   employees
       WHERE  employee_id = :empid";

    $conn = db_connect($err);
    if (!$conn) {
      handle_error('Connect Error', $err);
    }
    else {
      $bindargs = array();
      array_push($bindargs, array('EMPID', $empid, -1));

      $emp = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_ROW, $err,
             $bindargs);

      if (!$emp) {
        handle_error('Cannot find details for employee '.$empid, $err);
      }
      else {
        ui_print_header('Modify Employee ');
        ui_print_modify_employee($emp[0], $_SERVER['SCRIPT_NAME']);
        ui_print_footer(date('Y-m-d H:i:s'));
      }
    }
  }
}
```

13. anyco.php ファイルを編集します。エラーが処理されるように modify_emp() 関数を変更します。ファイルは、次のようになります。

```
function modify_emp()
{
  $newemp = $_POST;

  $statement =
    "UPDATE employees
     SET    first_name = :fnm, last_name = :lnm, email = :eml,
            salary = :sal, commission_pct = :cpt
     WHERE  employee_id = :eid";

  $conn = db_connect($err);
  if (!$conn) {
    handle_error('Connect Error', $err);
  }
  else {
```

```
    $bindargs = array();
    array_push($bindargs, array('EID', $newemp['empid'], -1));
    array_push($bindargs, array('FNM', $newemp['firstname'], -1));
    array_push($bindargs, array('LNM', $newemp['lastname'], -1));
    array_push($bindargs, array('EML', $newemp['email'], -1));
    array_push($bindargs, array('SAL', $newemp['salary'], -1));
    array_push($bindargs, array('CPT', $newemp['commpct'], -1));

    $r = db_execute_statement($conn, $statement, $err, $bindargs);

    if (!$r) {
      handle_error('Cannot update employee '.$newemp['empid'], $err);
    }
    else {
      construct_employees();
    }
  }
}
```

14. anyco.php ファイルを編集します。エラーが処理されるように delete_emp() 関数を変更します。ファイルは、次のようになります。

```
function delete_emp()
{
  if (!isset($_POST['emprec'])) { // User did not select a record
    construct_employees();
  }
  else {
    $empid = $_POST['emprec'];

    $conn = db_connect($err);
    if (!$conn) {
      handle_error('Connection Error', $err);
    }
    else {
      $statement = "DELETE FROM employees WHERE employee_id = :empid";
      $bindargs = array();
      array_push($bindargs, array('EMPID', $empid, -1));
      $r = db_execute_statement($conn, $statement, $err, $bindargs);

      if (!$r) {
        handle_error("Error deleting employee $empid", $err);
      }
      else {
        construct_employees();
      }
    }
  }
}
```

15. anyco.php ファイルを編集します。エラーが処理されるように get_dept_name() 関数を変更します。ファイルは、次のようになります。

```
function get_dept_name($conn, $deptid)
{
  $query =
    'SELECT department_name
     FROM   departments
     WHERE  department_id = :did';

  $conn = db_connect($err);
  if (!$conn) {
    return ('Unknown');
  }
```

```
  else {
    $bindargs = array();
    array_push($bindargs, array('DID', $deptid, -1));
    $dn = db_do_query($conn, $query, OCI_FETCHSTATEMENT_BY_COLUMN,
                      $err, $bindargs);
    if ($dn == false)
      return ('Unknown');
    else
      return($dn['DEPARTMENT_NAME'][0]);
  }
}
```

16. anyco_ui.inc ファイルを編集します。新しい関数 ui_print_error() を追加します。

```
function ui_print_error($message, $posturl)
{
  if (!$message) {
    echo '<p>Unknown error</p>';
  }
  else {
    echo "<p>Error at line {$message['LINE']} of "
         ."{$message['FILE']}</p>";  // Uncomment for debugging
    echo "<p>{$message['MESSAGE']}</p>";
  }
  echo <<<END
  <form method="post" action="$posturl">
  <input type="submit" value="Return to Departments" name="showdept">
END;
}
```

END; 行の先頭には空白を配置しないでください。END; 行の先頭に空白を配置すると、残りのドキュメントが出力対象テキストの一部として処理されます。

17. アプリケーション・ファイルへの変更を保存します。ブラウザに次の URL を入力して、変更をテストします。

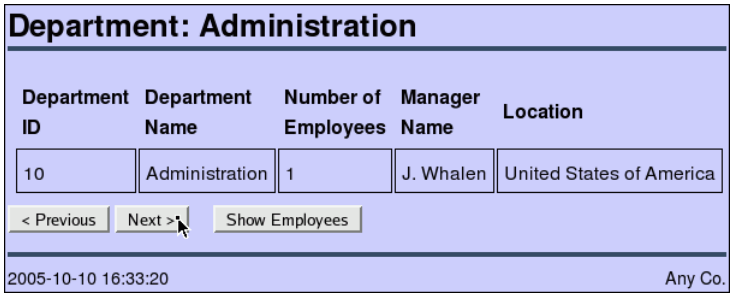
    Windows の場合 :

    http://localhost/chap5/anyco.php

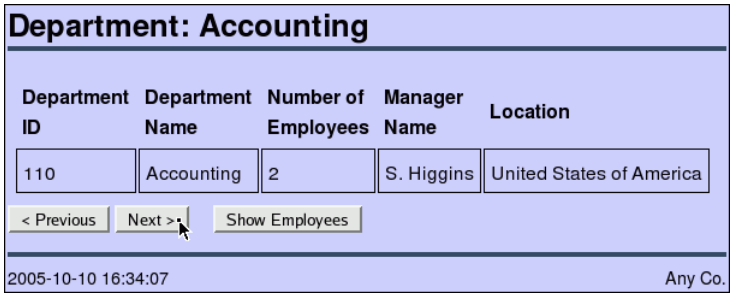
    Linux の場合 :

    http://localhost/~<username>/chap5/anyco.php

    「Departments」ページが表示されます。



18. 「**Next**」をクリックして、最後の部門レコード（ID が 110 の「Accounting」部門）にナビゲートします。「**Next**」をクリックして、最後の部門レコードを越えてナビゲートしてみます。



エラー処理によって、最後の部門レコードを越えてナビゲートすることはできません。

19. 給与 0（ゼロ）の新しい従業員を挿入した場合、または部門 ID を存在しない ID に変更した場合は、「Cannot insert employee」というヘッダーの新しいエラー・ページが表示されます。

追加のエラー処理

特定の Oracle エラーを個々に処理できます。たとえば、「Employees」ページで「**Insert new employee**」ボタンをクリックして新しい従業員レコードを作成し、部門 ID を存在しない部門に変更した場合は、このエラーをトラップし、よりわかりやすいメッセージを表示できます。

1. `anyco.php` ファイルを編集します。`insert_new_emp()` 関数のエラー処理を変更します。

```
$r = db_execute_statement($conn, $statement, $err, $bindargs);
if ($r) {
  construct_employees();
}
else {
  if ($err['CODE'] == 2291) {  // Foreign key violated
    handle_error("Department {$newemp['deptid']} does not yet exist",
    $err);
  }
  else {
    handle_error('Cannot insert employee', $err);
  }
}
```

2. アプリケーション・ファイルへの変更を保存します。次の URL を入力して、変更をテストします。

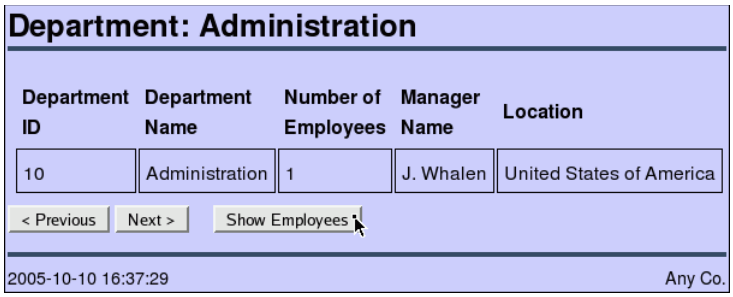
   Windows の場合 :

   `http://localhost/chap5/anyco.php`

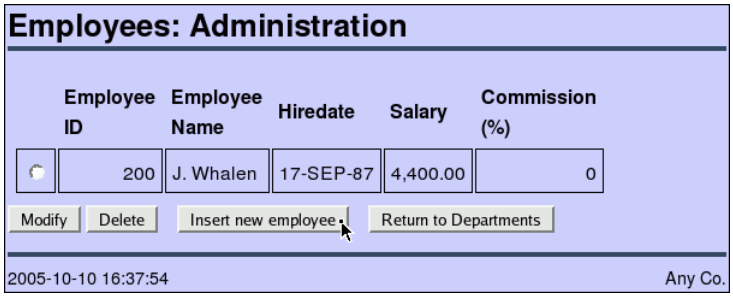
   Linux の場合 :

   `http://localhost/~<username>/chap5/anyco.php`

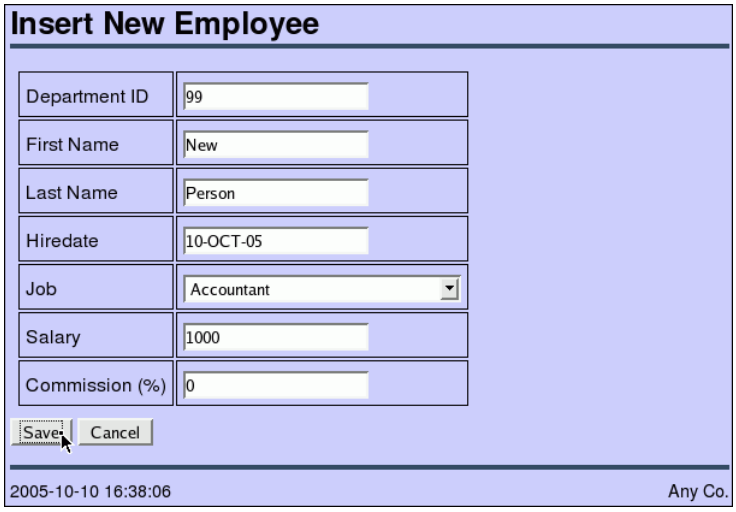
3. 「Departments」ページで、**「Show Employees」**をクリックします。



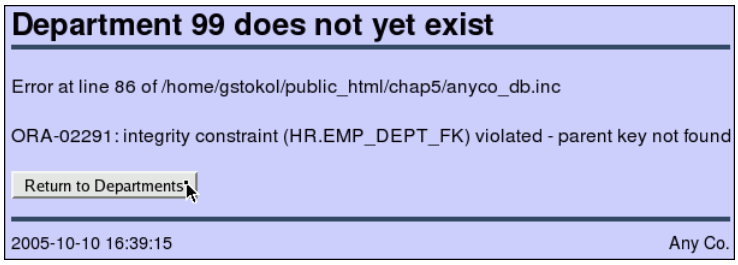
4. 「Employees」ページで、**「Insert new employee」**をクリックします。



5. 「Insert New Employee」ページで、次に示すように従業員詳細を入力し、「Department ID」を 99 に設定して、**「Save」**をクリックします。



次のエラー・ページが表示されます。



「Return to Departments」をクリックして「Departments」ページに戻り、**「Show Employees」**をクリックして新しい従業員レコードが「Administration」部門に追加されていないことを確認できます。

6

ストアド・プロシージャおよびストアド・ファンクションの実行

この章では、PHP および Oracle Database を使用してストアド・プロシージャおよびストアド・ファンクションを実行する方法について説明します。内容は次のとおりです。

- PL/SQL を使用したビジネス・ロジックの取得
- PL/SQL 参照カーソルを使用した結果セットの戻し

各従業員の報酬を計算する PL/SQL ファンクションおよび従業員レコードの参照カーソルを戻す PL/SQL プロシージャで Anyco アプリケーションを拡張します。

PL/SQL を使用したビジネス・ロジックの取得

Oracle PL/SQL プロシージャおよびファンクションを使用すると、すべてのクライアント・プログラムで使用できるビジネス・ロジックをデータベースに保存できます。これによって、データベースと PHP 間で転送するデータ量を減らし、パフォーマンスを向上させることができます。

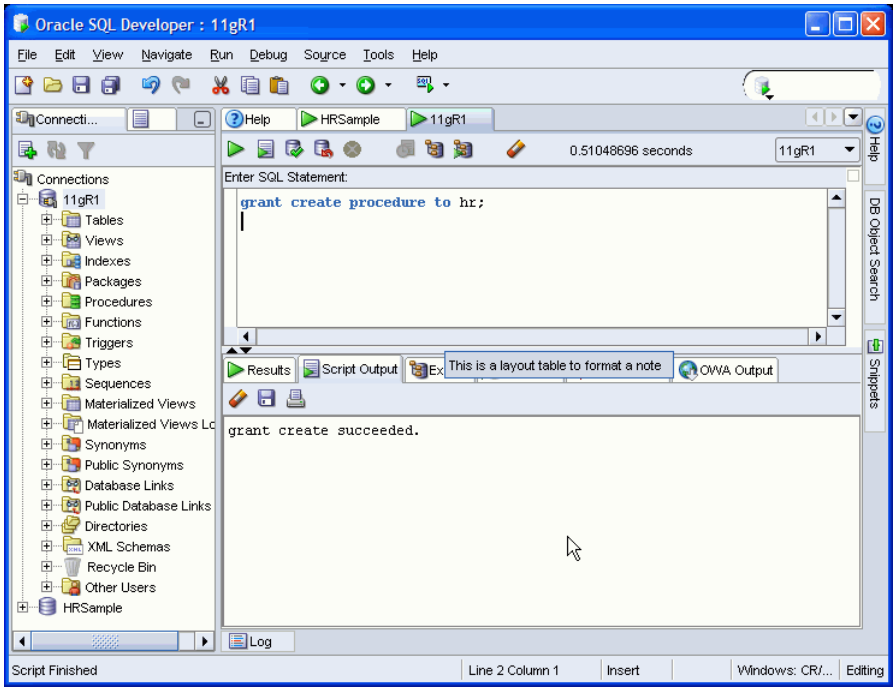
この項では、各従業員の合計報酬を計算して表示する PL/SQL ストアド・ファンクションを作成します。

各従業員の合計報酬を表示するには、次の手順を実行します。

PHP アプリケーションは、HR ユーザーとしてデータベースに接続します。DBA 権限を持つユーザーとして、HR アカウントをロック解除する必要がある場合があります。HR ユーザーをロック解除するには、次の手順を実行します。

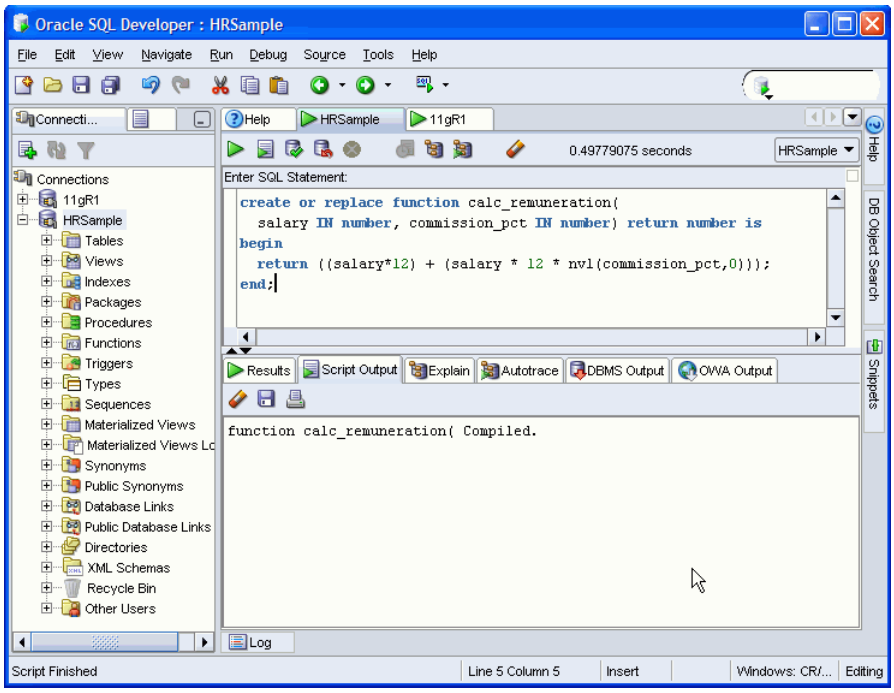
1. SQL Developer を起動し、Oracle データベースへの接続をオープンします。
2. **system** ユーザーとして Oracle データベースにログインします。
3. SQL ワークシートまたは SQL*Plus を開き、次の `grant` 文を実行して HR ユーザーに `create procedure` 権限を割り当てます。

```
grant create procedure to hr;
```



4. `hr` ユーザーとして HR サンプル・スキーマにログインします。
5. SQL ワークシートまたは SQL*Plus を開き、次のテキストを入力して `calc_remuneration()` 関数を作成します。

```
create or replace function calc_remuneration(
  salary IN number, commission_pct IN number) return number is
begin
  return ((salary*12) + (salary * 12 * nvl(commission_pct,0)));
end;
```



6. chap6 ディレクトリを作成し、chap5 からアプリケーション・ファイルをコピーし、新しく作成したディレクトリに移動します。

   Windows の場合 :

   ```
   mkdir c:¥program files¥Zend¥Apache2¥htdocs¥chap6
   cd c:¥program files¥Zend¥Apache2¥htdocs¥chap6
   copy ..¥chap5¥* .
   ```

   Linux の場合 :

   ```
   mkdir $HOME/public_html/chap6
   cd $HOME/public_html/chap6
   cp ../chap5/* .
   ```

7. anyco.php ファイルを編集します。行が戻されるたびに PL/SQL ファンクションがコールされるように construct_employees() 関数の問合せを変更します。

   ```
   $query =
    "SELECT employee_id,
            substr(first_name,1,1) || '. '|| last_name as employee_name,
            hire_date,
            to_char(salary, '9999G999D99') as salary,
            nvl(commission_pct,0) as commission_pct,
            to_char(calc_remuneration(salary, commission_pct),'9999G999D99')
              as remuneration
      FROM employees
      WHERE department_id = :did
      ORDER BY employee_id ASC";
   ```

8. anyco_ui.inc ファイルを編集します。ui_print_employees() 関数で、表に「Remuneration」列を追加し、従業員ごとに「remuneration」フィールドが表示されるように foreach ループを変更します。

   ```
   echo <<<END
      <form method="post" action="$posturl">
      <table>
      <tr>
        <th> </th>
   ```

```
      <th>Employee<br>ID</th>
      <th>Employee<br>Name</th>
      <th>Hiredate</th>
      <th>Salary</th>
      <th>Commission<br>(%)</th>
      <th>Remuneration</th>
    </tr>
END;

    // Write one row per employee
    foreach ($employeerecords as $emp) {
      echo '<tr>';
      echo '<td><input type="radio" name="emprec"
                value="'.htmlentities($emp['EMPLOYEE_ID']).'"></td>';
      echo '<td align="right">'.htmlentities($emp['EMPLOYEE_ID']).'</td>';
      echo '<td>'.htmlentities($emp['EMPLOYEE_NAME']).'</td>';
      echo '<td>'.htmlentities($emp['HIRE_DATE']).'</td>';
      echo '<td align="right">'.htmlentities($emp['SALARY']).'</td>';
      echo '<td align="right">'.htmlentities($emp['COMMISSION_PCT']).'</td>';
      echo '<td align="right">'.htmlentities($emp['REMUNERATION']).'</td>';
      echo '</tr>';
    }
```

9. アプリケーション・ファイルへの変更を保存します。ブラウザで、次の URL を入力して、アプリケーションをテストします。

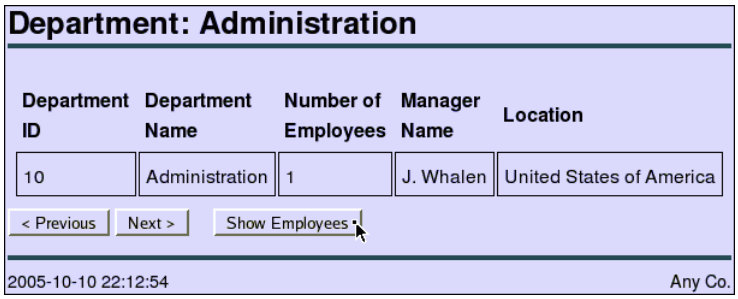
Windows の場合 :

```
http://localhost/chap6/anyco.php
```

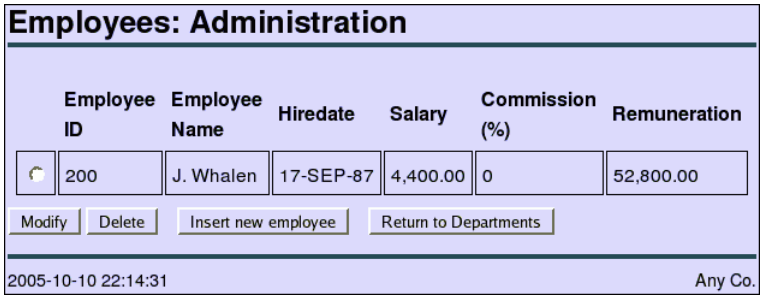
Linux の場合 :

```
http://localhost/~<username>/chap6/anyco.php
```

10. 「Departments」ページで、**「Show Employees」**をクリックします。



「Employees」ページで、従業員の報酬が最後の列に表示されます。



PL/SQL 参照カーソルを使用した結果セットの戻し

PL/SQL ブロックから参照カーソルとして問合せデータを戻し、PHP に表示できます。この操作は、データセットで複雑な機能が必要な場合または複数のアプリケーション・プログラムで同じ問合せを使用する場合に有効です。

PL/SQL の参照カーソルは、カーソル変数に割り当てられているタイプ定義です。パッケージ仕様部内に PL/SQL タイプを宣言して、パッケージ本体などの他の PL/SQL 構造で再利用することがよく行われます。

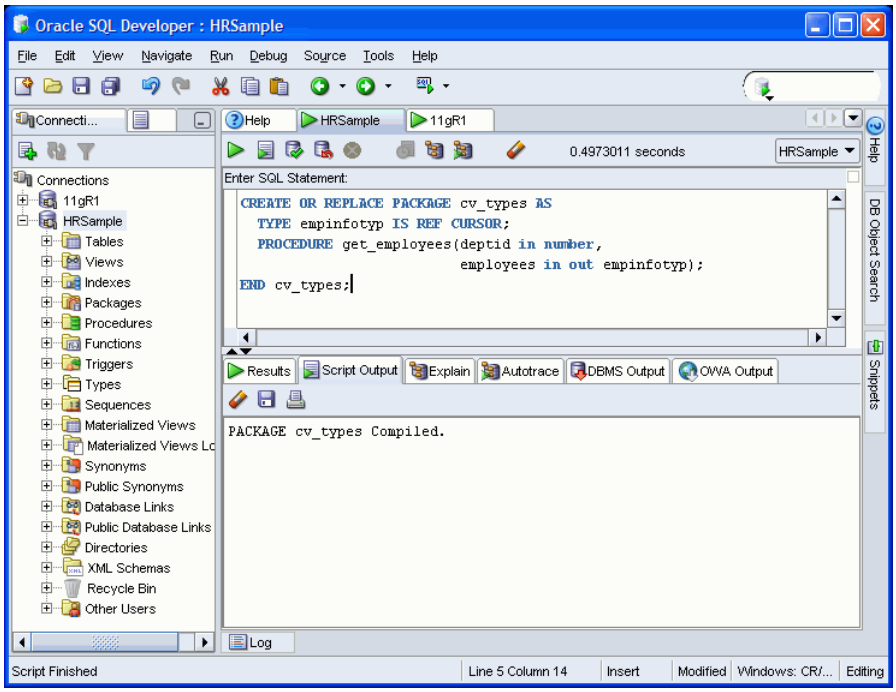
この項では、参照カーソルを使用して、特定の部門の従業員を取り出します。

PL/SQL パッケージ仕様部およびパッケージ本体を作成し、参照カーソルを使用して特定の部門の従業員を取り出すには、次の手順を実行します。

1. SQL Developer を起動し、hr ユーザーとして HR サンプル・スキーマにログインします。
2. SQL ワークシートまたは SQL*Plus を開き、次のテキストを入力して PL/SQL パッケージ cv_types を作成します。

```
CREATE OR REPLACE PACKAGE cv_types AS
  TYPE empinfotyp IS REF CURSOR;
  PROCEDURE get_employees(deptid in number,
                          employees in out empinfotyp);
END cv_types;
```

「Run」をクリックします。

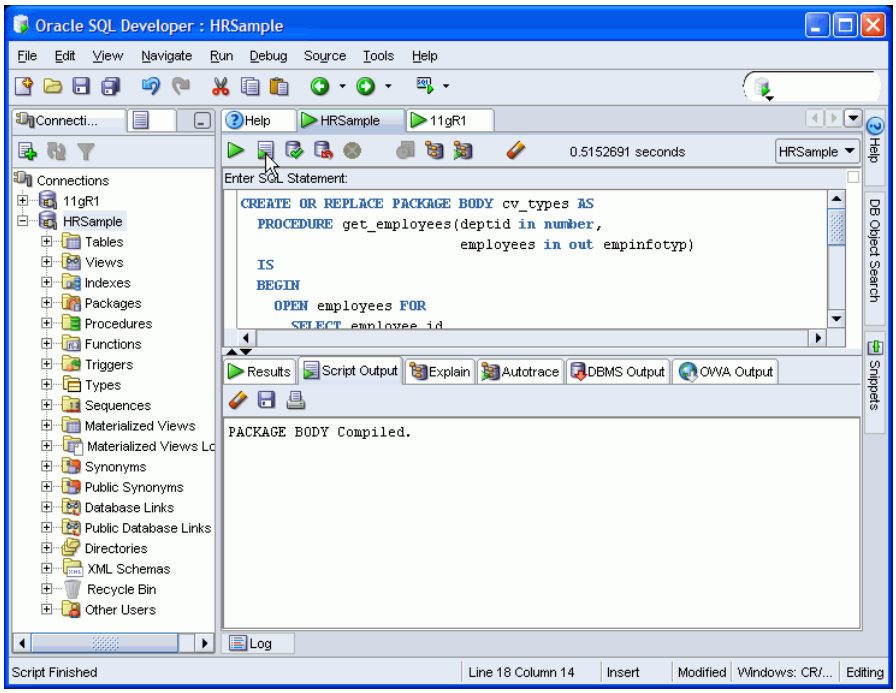


3. SQL ワークシートで、次のテキストを入力して PL/SQL パッケージ本体 cv_types を作成します。

```
CREATE OR REPLACE PACKAGE BODY cv_types AS
  PROCEDURE get_employees(deptid in number,
                          employees in out empinfotyp)
  IS
  BEGIN
    OPEN employees FOR
      SELECT employee_id,
        substr(first_name,1,1) || '. '|| last_name as employee_name,
        hire_date,
```

```
        to_char(salary, '999G999D99') as salary,
        NVL(commission_pct,0) as commission_pct,
        to_char(calc_remuneration(salary, commission_pct),
                '9999G999D99') as remuneration
    FROM employees
    WHERE department_id = deptid
    ORDER BY employee_id ASC;
  END get_employees;
END cv_types;
```

「**Run**」をクリックします。



4. anyco_db.inc ファイルを編集します。PL/SQL パッケージ・プロシージャをコールする新しい PHP 関数を作成します。

```
// Use ref cursor to fetch employee records
// All records are retrieved - there is no paging in this example
function db_get_employees_rc($conn, $deptid, &$e)
{
  // Execute the call to the stored procedure
  $stmt = "BEGIN cv_types.get_employees($deptid, :rc); END;";
  $stid = @oci_parse($conn, $stmt);
  if (!$stid) {
    $e = db_error($conn, __FILE__, __LINE__);
    return false;
  }
  $refcur = oci_new_cursor($conn);
  if (!$stid) {
    $e = db_error($conn, __FILE__, __LINE__);
    return false;
  }
  $r = @oci_bind_by_name($stid, ':RC', $refcur, -1, OCI_B_CURSOR);
  if (!$r) {
    $e = db_error($stid, __FILE__, __LINE__);
    return false;
  }
  $r = @oci_execute($stid);
  if (!$r) {
```

```
    $e = db_error($stid, __FILE__, __LINE__);
    return false;
  }
  // Now treat the ref cursor as a statement resource
  $r = @oci_execute($refcur, OCI_DEFAULT);
  if (!$r) {
    $e = db_error($refcur, __FILE__, __LINE__);
    return false;
  }
  $r = @oci_fetch_all($refcur, $employeerecords, null, null,
                      OCI_FETCHSTATEMENT_BY_ROW);
  if (!$r) {
    $e = db_error($refcur, __FILE__, __LINE__);
    return false;
  }
  return ($employeerecords);
}
```

db_get_employees_rc() 関数は、次の無名（名前が指定されていない）PL/SQL ブロックを実行します。

```
BEGIN cv_types.get_employees($deptid, :rc); END;
```

BEGIN END ブロック内の PL/SQL 文は、ストアド PL/SQL パッケージ・プロシージャ cv_types.et_employees() をコールします。これによって、PHP 変数 $refcur に OCI_B_CURSOR 参照カーソル・バインド変数が戻されます。

$refcur 変数は、oci_parse() によって戻される文ハンドルと同様に処理されます。PHP で SQL 問合せを実行した場合と同様に、実行操作およびフェッチ操作に使用されます。

5. anyco.php ファイルを編集します。construct_employees() 関数で、問合せテキストおよびバインド引数を削除します。ファイルは、次のようになります。

```
function construct_employees()
{
  $deptid = $_SESSION['deptid'];
  $conn = db_connect($err);
  if (!$conn) {
    handle_error('Connection Error', $err);
  }
  else {
    $emp = db_get_employees_rc($conn, $deptid, $err);

    if (!$emp) {
      handle_error('Cannot fetch Employees', $err);
    }
    else {
      $deptname = get_dept_name($conn, $deptid);

      ui_print_header('Employees: '.$deptname);
      ui_print_employees($emp, $_SERVER['SCRIPT_NAME']);
      ui_print_footer(date('Y-m-d H:i:s'));
    }
  }
}
```

6. アプリケーション・ファイルへの変更を保存します。ブラウザで、次の URL を入力して、アプリケーションをテストします。

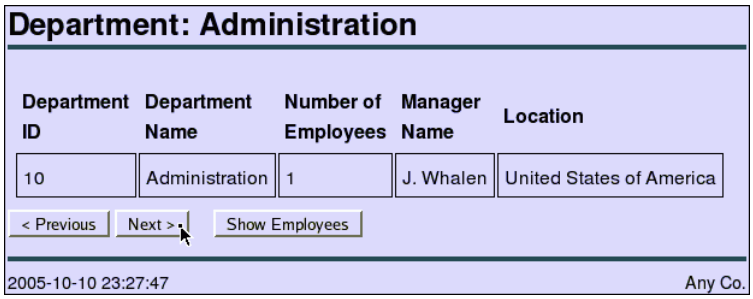
   Windows の場合 :

   ```
   http://localhost/chap6/anyco.php
   ```

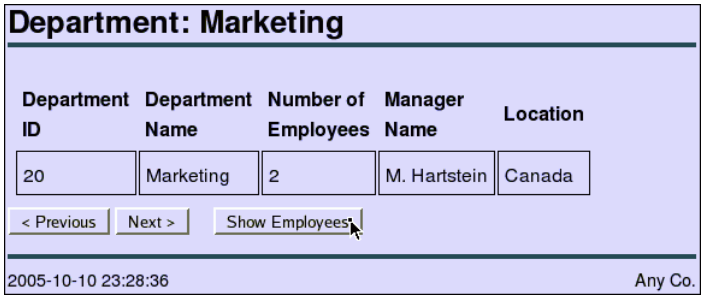
   Linux の場合 :

   ```
   http://localhost/~<username>/chap6/anyco.php
   ```

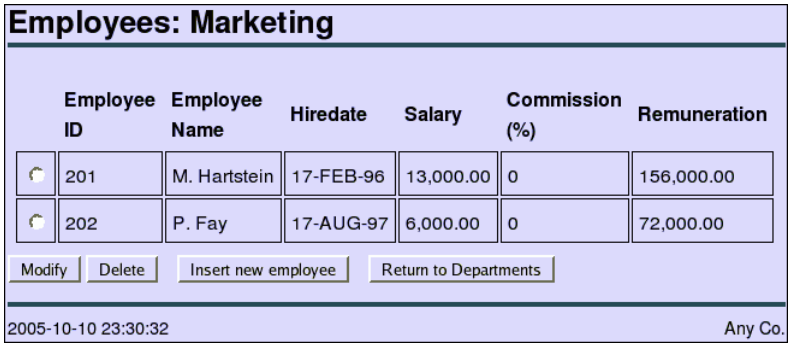
7. 「Departments」ページで、**「Next」**をクリックして「Marketing」部門ページにナビゲートします。



8. 「Marketing」部門ページで、**「Show Employees」**をクリックします。



「Employees」ページに、従業員のページが以前と同様に表示されます。



7

イメージのロード

この章では、新しい従業員レコードの JPEG イメージをアップロードして「Employees」ページに表示するためにアプリケーションを変更する方法について説明します。内容は次のとおりです。

- BLOB を使用した従業員イメージの保存およびロード
- イメージのサイズ変更

BLOB を使用した従業員イメージの保存およびロード

この項では、従業員のレコードに写真を格納できるようにアプリケーション・コードを変更します。

従業員レコードに従業員のイメージを格納できるようにするには、次の手順を実行します。

1. chap7 ディレクトリを作成し、chap6 からアプリケーション・ファイルをコピーし、新しく作成したディレクトリに移動します。

   Windows の場合 :

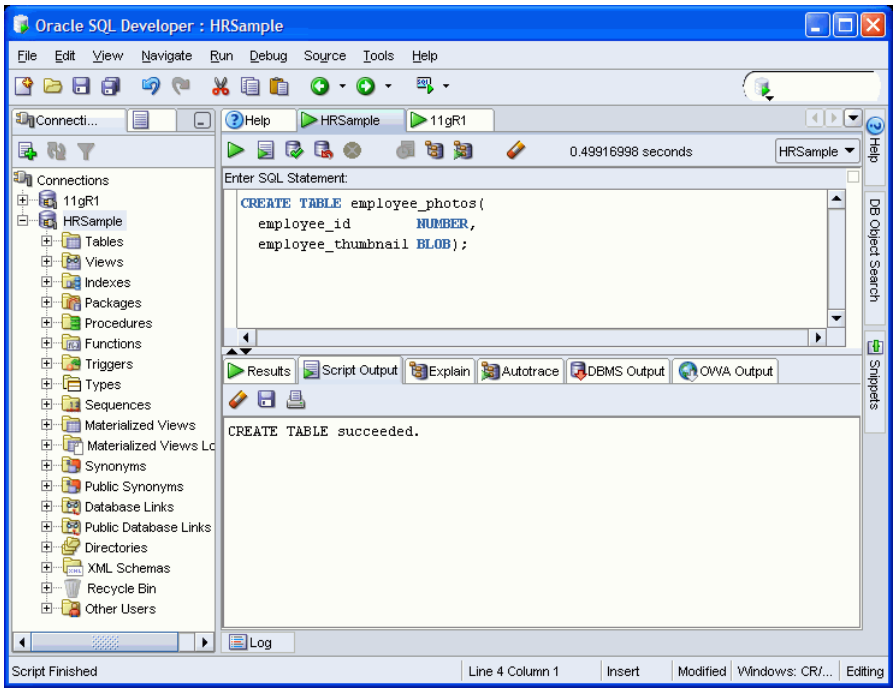
   ```
   mkdir c:¥program files¥Zend¥Apache2¥htdocs¥chap7
   cd c:¥program files¥Zend¥Apache2¥htdocs¥chap7
   copy ..¥chap6¥* .
   ```

   Linux の場合 :

   ```
   mkdir $HOME/public_html/chap7
   cd $HOME/public_html/chap7
   cp ../chap6/* .
   ```

2. SQL Developer を起動し、HR サンプル・スキーマへの接続をオープンします。
3. **hr** ユーザーとして HR サンプル・スキーマにログインします。
4. SQL ワークシートを開き、次の CREATE TABLE 文を入力して、従業員イメージを格納するための新しい表を作成します。

   ```
   CREATE TABLE employee_photos(
     employee_id        NUMBER,
     employee_thumbnail BLOB);
   ```



5. HR ユーザーがこのコマンドを実行するには、CREATE TABLE 権限が必要です。「insufficient privileges」エラー・メッセージが戻された場合は、HR ユーザーとしてログアウトしてから、system としてログインし、次の GRANT コマンドを実行します。

   ```
   GRANT create table TO hr;
   ```

   次に、再度 HR としてログインして、CREATE TABLE 文を実行します。

6. anyco_ui.inc ファイルを編集します。ui_print_employees() 関数の EMPLOYEES 表に「Photograph」列を追加します。

```
<th>Commission<br>(%)</th>
<th>Remuneration</th>
<th>Photograph</th>
```

「Photograph」列のデータは、<img> タグ付きで移入されます。このタグでは、src 属性が新しい anyco_im.php ファイルへの URL 参照として定義されているため、従業員レコードごとにイメージが表示されます。

7. anyco_ui.inc ファイルを編集します。ui_print_employees() 関数に、従業員 ID をパラメータとして指定して、anyco_im.php ファイルを参照する <img> タグを生成するコードを追加します。

```
echo '<td align="right">'
     .htmlentities($emp['REMUNERATION']).'</td>';
echo '<td><img src="anyco_im.php?showempphoto='.$emp['EMPLOYEE_ID']
     .'" alt="Employee photo"></td>';
```

8. anyco_ui.inc ファイルを編集します。新しい従業員レコードの作成時にイメージをアップロードできるようにするには、ui_print_insert_employee() 関数の <form> タグに enctype 属性を追加します。

```
<form method="post" action="$posturl" enctype="multipart/form-data">
```

フォームの下部にアップロード用のフィールドを追加し、その入力タイプを file に設定します。

```
<tr>
  <td>Commission (%)</td>
  <td><input type="text" name="commpct" value="0" size="20"></td>
</tr>
<tr>
  <td>Photo</td>
  <td><input type="file" name="empphoto"></td>
</tr>
```

9. anyco_im.php ファイルを作成します。このファイルは、URL パラメータとして従業員 ID をとり、その従業員レコードの「Photograph」列からイメージを読み取り、表示するサムネイル・イメージを戻します。

```
<?php    // anyco_im.php

require('anyco_cn.inc');
require('anyco_db.inc');
construct_image();

function construct_image()
{
  if (!isset($_GET['showempphoto'])) {
    return;
  }

  $empid = $_GET['showempphoto'];

  $conn = db_connect($err);

  if (!$conn) {
    return;
  }

  $query =
    'SELECT employee_thumbnail
     FROM   employee_photos
```

```
    WHERE  employee_id = :eid';

  $stid = oci_parse($conn, $query);
  $r = oci_bind_by_name($stid, ":eid", $empid, -1);
  if (!$r) {
    return;
  }
  $r = oci_execute($stid, OCI_DEFAULT);
  if (!$r) {
    return;
  }

  $arr = oci_fetch_row($stid);
  if (!$arr) {
    return;                     // photo not found
  }

  $result = $arr[0]->load();

  // If any text (or whitespace!) is printed before this header is sent,
  // the text is not displayed. The image also is not displayed properly.
  // Comment out the "header" line to see the text and debug.
  header("Content-type: image/JPEG");
  echo $result;
}

?>
```

construct_image() 関数は、OCI-Lob->load() 関数を使用して、イメージ・データである Oracle LOB データを取り出します。PHP の header() 関数は、HTTP レスポンス・ヘッダーに MIME タイプを設定して、ブラウザがデータを JPEG イメージとして認識できるようにします。

他のイメージ・タイプを表示する場合は、そのタイプに応じて Content-type を変更する必要があります。

10. anyco_db.inc ファイルを編集します。EMPLOYEE_PHOTOS 表にイメージを挿入する新しい関数 db_insert_thumbnail() を追加します。

```
function db_insert_thumbnail($conn, $empid, $imgfile, &$e)
{
  $lob = oci_new_descriptor($conn, OCI_D_LOB);
  if (!$lob) {
    $e = db_error($conn, __FILE__, __LINE__);
    return false;
  }

  $insstmt =
    'INSERT INTO employee_photos (employee_id, employee_thumbnail)
     VALUES(:eid, empty_blob())
     RETURNING employee_thumbnail into :etn';

  $stmt = oci_parse($conn, $insstmt);
  $r = oci_bind_by_name($stmt, ':etn', $lob, -1, OCI_B_BLOB);
  if (!$r) {
    $e = db_error($stid, __FILE__, __LINE__);
    return false;
  }
  $r = oci_bind_by_name($stmt, ':eid', $empid, -1);
  if (!$r) {
    $e = db_error($stid, __FILE__, __LINE__);
    return false;
  }
  $r = oci_execute($stmt, OCI_DEFAULT);
```

```
  if (!$r) {
    $e = db_error($stid, __FILE__, __LINE__);
    return false;
  }

  if (!$lob->savefile($imgfile)) {
    $e = db_error($stid, __FILE__, __LINE__);
    return false;
  }
  $lob->free();

  return true;
}
```

新しい EMPLOYEE_PHOTOS 表と EMPLOYEES 表を結び付けるには、両方の表で同じ従業員 ID を使用する必要があります。

11. anyco_db.inc ファイルを編集します。OUT バインド変数値がデータベースから戻されるように、db_execute_statement() 関数の $bindvars パラメータを &$bindvars に変更します。この関数の下部に、戻されたバインド変数値を設定するループを追加します。

```
function db_execute_statement($conn, $statement, &$e, &$bindvars = array())
{
  ...
  $r = @oci_execute($stid);
  if (!$r) {
    $e = db_error($stid, __FILE__, __LINE__);
    return false;
  }
  $outbinds = array();
  foreach ($bindvars as $b) {
    $outbinds[$b[0]] = $$b[0];
  }
   $bindvars = $outbinds;
   return true;
}
```

12. anyco.php ファイルを編集します。バインド変数 :neweid に新しい従業員 ID が戻されるように、insert_new_emp() 関数の INSERT 文を変更します。この値がイメージとともに新しい EMPLOYEE_PHOTOS 表に挿入されます。

```
$statement =
  'INSERT INTO employees
              (employee_id, first_name, last_name, email, hire_date,
              job_id, salary, commission_pct, department_id)
   VALUES (employees_seq.nextval, :fnm, :lnm, :eml, :hdt,
          :jid, :sal, :cpt, :did)
   RETURNING employee_id into :neweid';
```

また、insert_new_emp() 関数で、array_push() 関数へのコールを追加して、新しいバインド変数 NEWEID を array_push() コールのリストの最後に設定します。

```
array_push($bindargs, array('CPT', $newemp['commpct'], -1));
array_push($bindargs, array('DID', $newemp['deptid'], -1));
array_push($bindargs, array('NEWEID', null, 10));
```

NEWID の値は INSERT 文の RETURNING 句で取り出されるため、その初期値は NULL に設定します。長さは、戻り値に十分な桁を確保するために 10 に設定します。

13. anyco.php ファイルを編集します。insert_new_emp() 関数で、db_execute_statement() コールと construct_employees() コールの間にサムネイル・イメージを挿入するコールを追加します。

```
$r = db_execute_statement($conn, $statement, $err, $bindargs);
if ($r) {
  $r = db_insert_thumbnail($conn, $bindargs['NEWEID'],
                           $_FILES['empphoto']['tmp_name'], $e);
  construct_employees();
}
```

14. ブラウザに、次のアプリケーション URL を入力します。

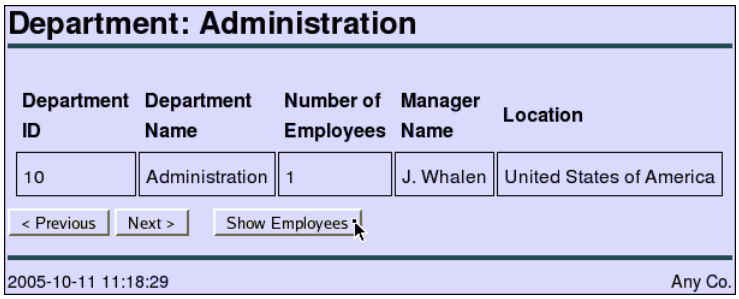
    Windows の場合 :

    ```
    http://localhost/chap7/anyco.php
    ```

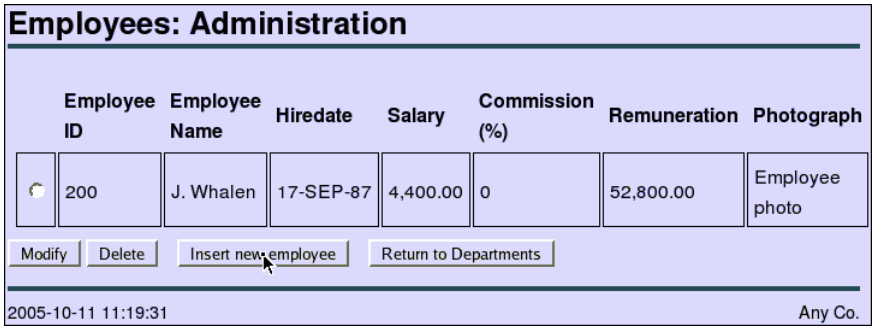
    Linux の場合 :

    ```
    http://localhost/~<username>/chap7/anyco.php
    ```

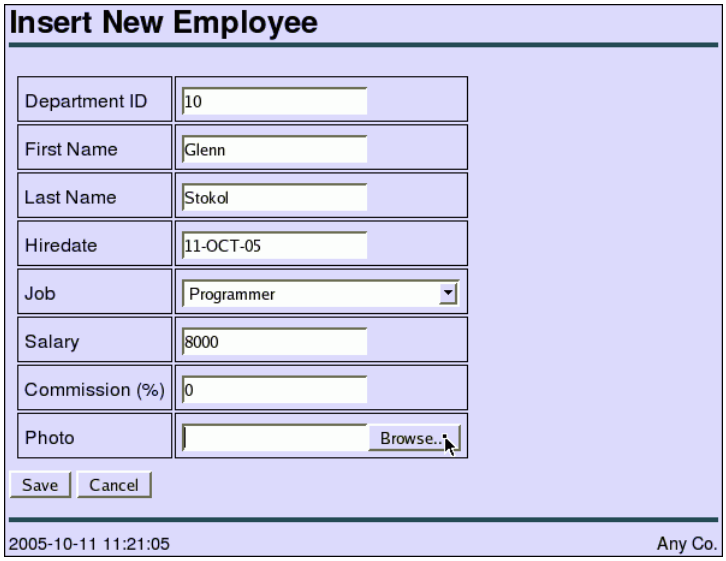
15. 「Departments」ページで、**「Show Employees」**をクリックして「Employees」ページにナビゲートします。



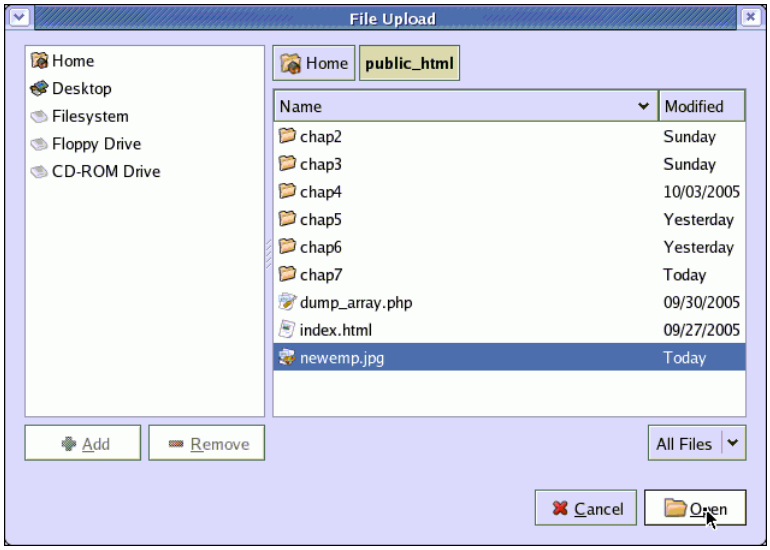
16. 「Employees」ページで、新しい従業員レコードを挿入するには、**「Insert new employee」**をクリックします。



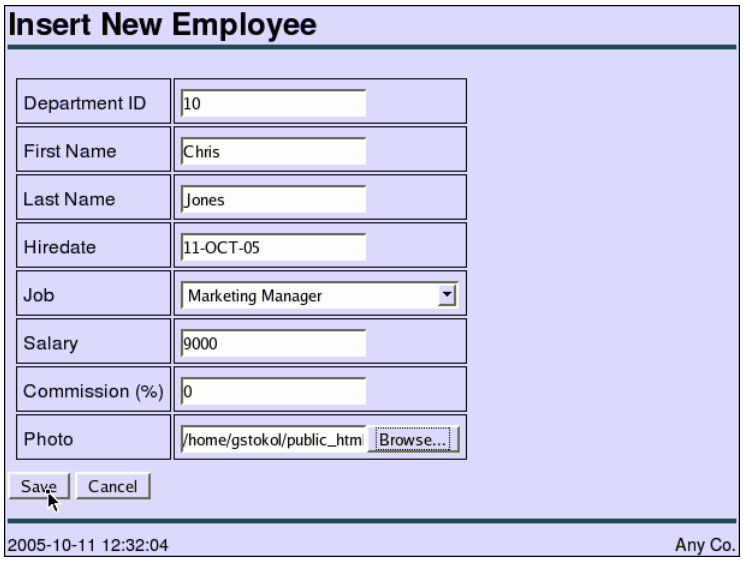
17. 「Insert New Employee」フォームでは、データベースにアップロードするシステム上のサムネイル・イメージを選択できます。次の各フィールドに独自の値を入力するか、または表示されている値を使用します。**「Browse」**をクリックします。



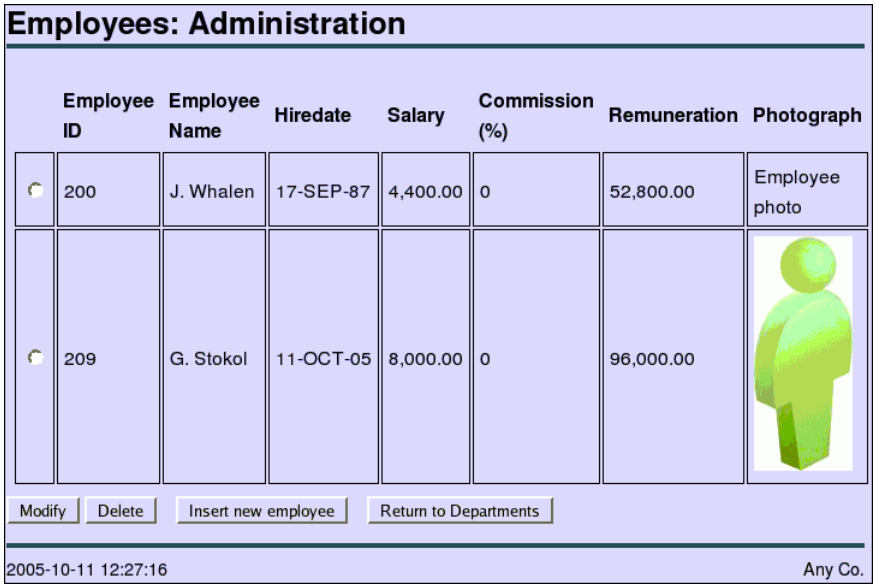
18. 「File Upload」ウィンドウで、JPEG イメージ・ファイルを参照して選択し、**「Open」**をクリックします。



19. 「Insert New Employee」ページで、「**Save**」をクリックします。



「Employees」ページに、元のサイズのイメージを含む新しい従業員レコードが表示されます。



イメージのサイズ変更

この項では、アプリケーション・コードをさらに変更して、指定したイメージからサムネイル・イメージを作成し、従業員のレコードにそのサムネイル・イメージを格納します。

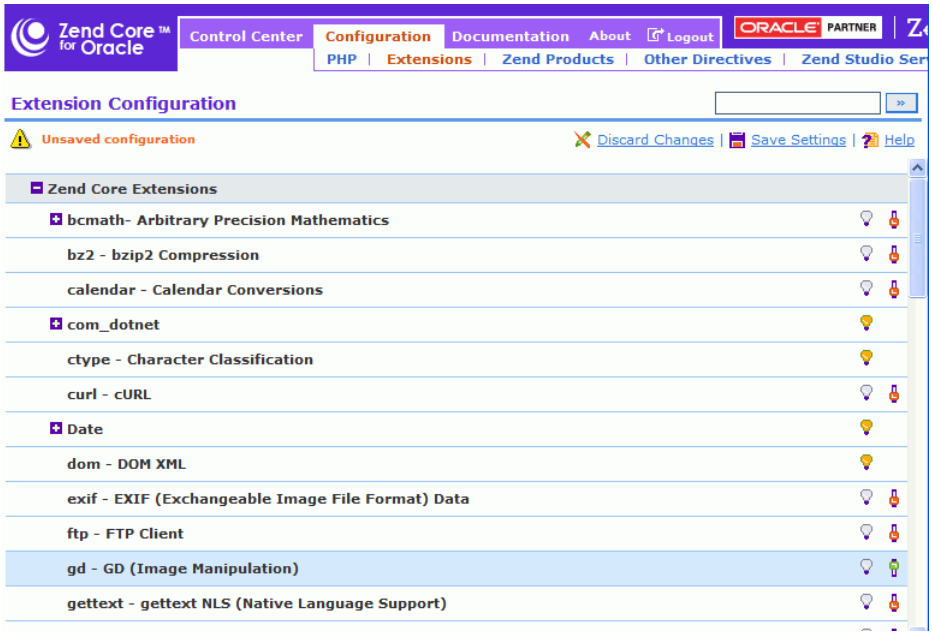
PHP の GD グラフィックス拡張モジュールを使用して、従業員イメージをサイズ変更します。

1. グラフィックス拡張モジュールを有効にするには、ブラウザに次の URL を入力して、Zend Core for Oracle コンソールにアクセスします。

   ```
   http://localhost/ZendCore
   ```

2. ログイン画面で、「Password」フィールドに Zend Core for Oracle のインストール時に指定したパスワードを入力し、ログイン (>>>)・アイコンをクリックします。

3. コンソール・ページで、「**Configuration**」タブをクリックします。
4. 「Configuration」タブ・ページで、「**Extensions**」サブタブをクリックします。
5. 「Extensions」サブタブ・ページで、「Zend Core Extensions」ツリー・コントロールを開きます。「**gd - GD (Image Manipulation)**」エントリを検索し、そのスイッチを on または `enabled` に変更します。



6. 「Extensions」サブタブ・ページで、構成変更を保存するには、「**Save Setting**」をクリックします。
7. 「**Logout**」をクリックして、Zend Core for Oracle コンソールからログアウトします。

8. Apache を再起動します。ApacheMonitor ユーティリティを使用するか、または Windows のサービスを使用できます。

   ApacheMonitor ユーティリティを使用するには、Apache の bin ディレクトリに移動し、ApacheMonitor.exe をダブルクリックします。デフォルトのインストールでは、Apache の bin ディレクトリは c:¥Program Files¥Zend¥Apache2¥bin です。

   Windows のサービスにアクセスするには、Windows の**「スタート」**メニューで、**「スタート」→「コントロール パネル」→「管理ツール」→「サービス」を選択します。「標準」**タブを選択します。「Apache2 HTTP Server」を右クリックし、**「再起動」**を選択します。

9. anyco_db.inc ファイルを編集します。イメージをサイズ変更してサムネイル・イメージを作成するには、db_insert_thumbnail() 関数で $lob->savefile($imgfile) へのコールの前に次のコードを追加します。

```
$r = oci_execute($stmt, OCI_DEFAULT);
if (!$r) {
  $e = db_error($stid, __FILE__, __LINE__);
  return false;
}

// Resize the image to a thumbnail
define('MAX_THUMBNAIL_DIMENSION', 100);
$src_img = imagecreatefromjpeg($imgfile);
list($w, $h) = getimagesize($imgfile);
if ($w > MAX_THUMBNAIL_DIMENSION || $h > MAX_THUMBNAIL_DIMENSION)
{
  $scale =  MAX_THUMBNAIL_DIMENSION / (($h > $w) ? $h : $w);
  $nw = $w * $scale;
  $nh = $h * $scale;

  $dest_img = imagecreatetruecolor($nw, $nh);
  imagecopyresampled($dest_img, $src_img, 0, 0, 0, 0, $nw, $nh, $w, $h);

  imagejpeg($dest_img, $imgfile);  // overwrite file with new thumbnail

  imagedestroy($src_img);
  imagedestroy($dest_img);
}

if (!$lob->savefile($imgfile)) {
...
```

   imagecreatefromjpeg() 関数を使用して、JPEG ファイルを読み取り、後続の GD 関数で使用される内部表現を作成しています。次に、最長側面が 100 ピクセル以下の新しい次元を計算しています。imagecreatetruecolor() 関数を使用して、その新しいサイズのテンプレート・イメージを作成しています。imagecopyresampled() 関数を使用して元のイメージのデータをそのテンプレート・イメージにサンプリングして、サムネイル・イメージを作成しています。サムネイル・イメージを元のファイルに書き戻し、イメージの内部表現を解放しています。

   db_insert_thumbnail() 関数の既存のコードは、以前の実装と同様に、データベースにイメージ・ファイルをアップロードします。

10. ブラウザに次の URL を入力して、アプリケーションの変更をテストします。

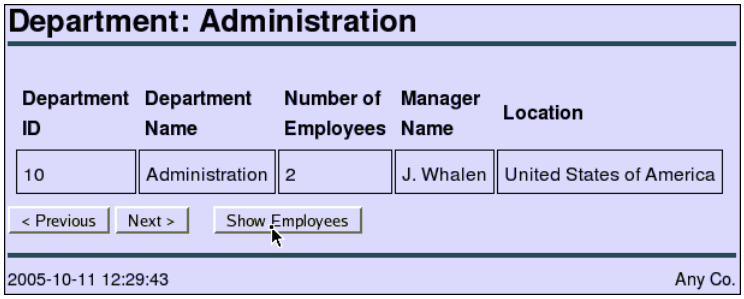
    Windows の場合 :

    http://localhost/chap7/anyco.php

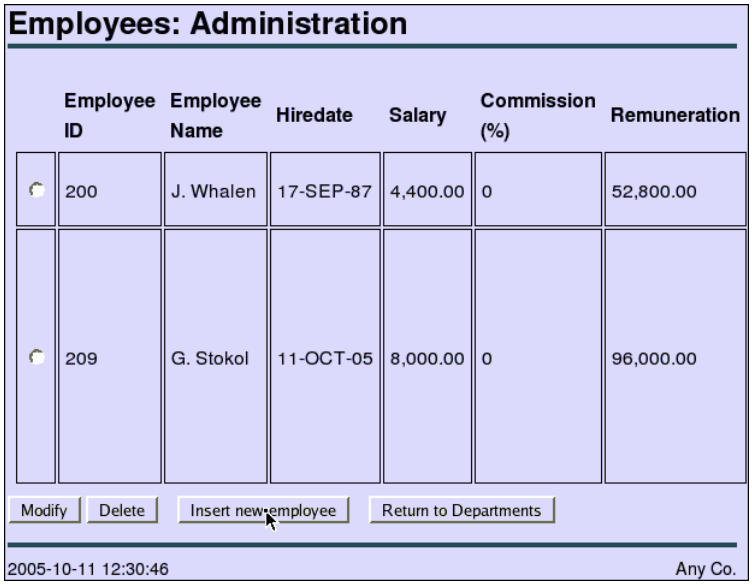
    Linux の場合 :

    http://localhost/~<username>/chap7/anyco.php

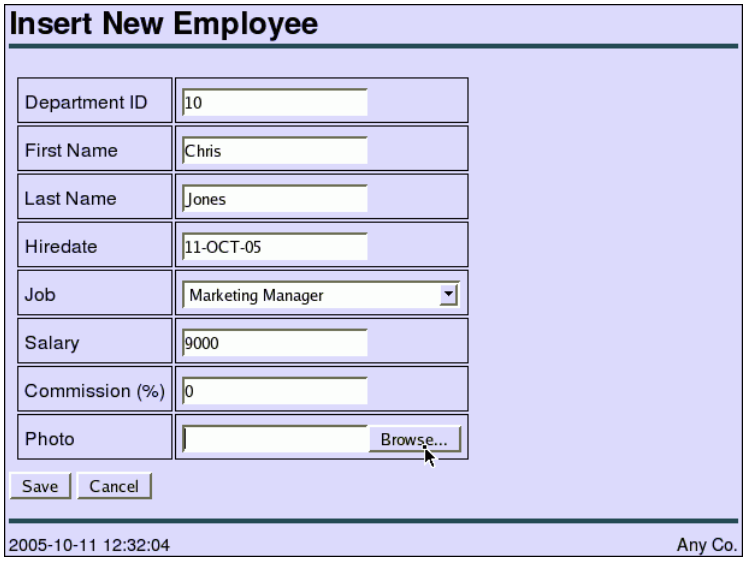
11. 「Departments」ページで、**「Show Employees」**をクリックして、「Employees」ページにナビゲートします。



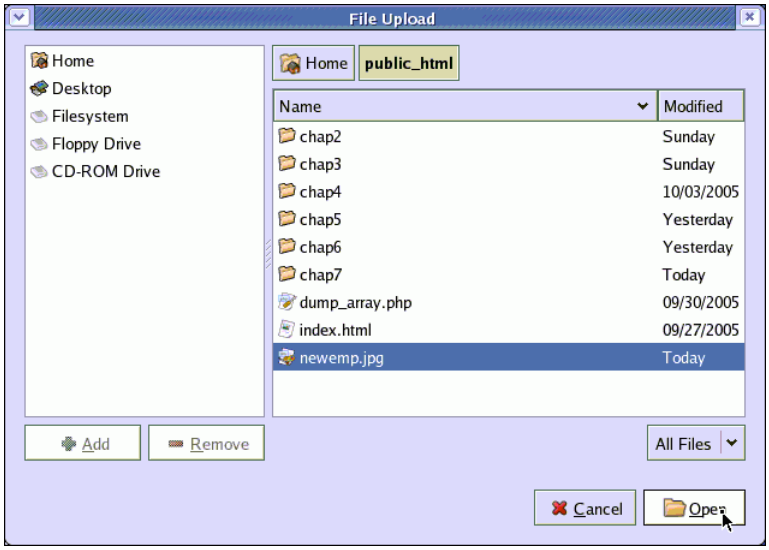
12. 「Employees」ページで、新しい従業員レコードを挿入するには、**「Insert new employee」**をクリックします。



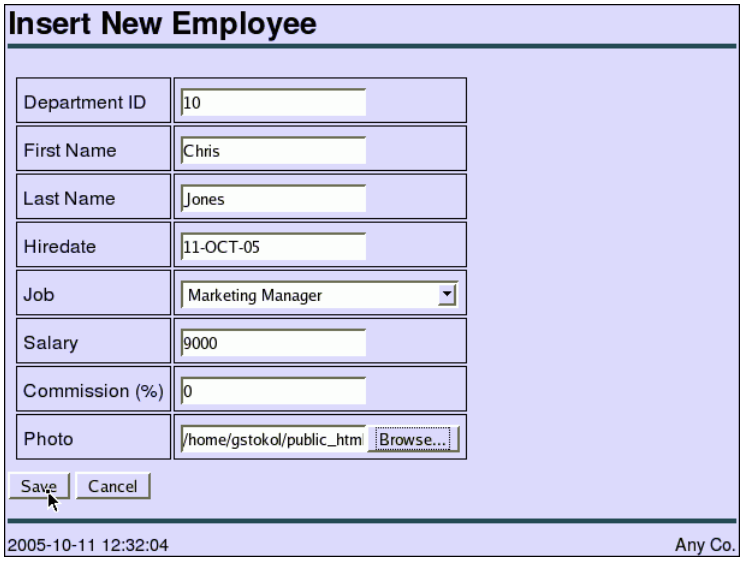
13. 新しく従業員の詳細を入力するか、または表示されている値を使用します。従業員イメージを参照するには、**「Browse」**をクリックします。



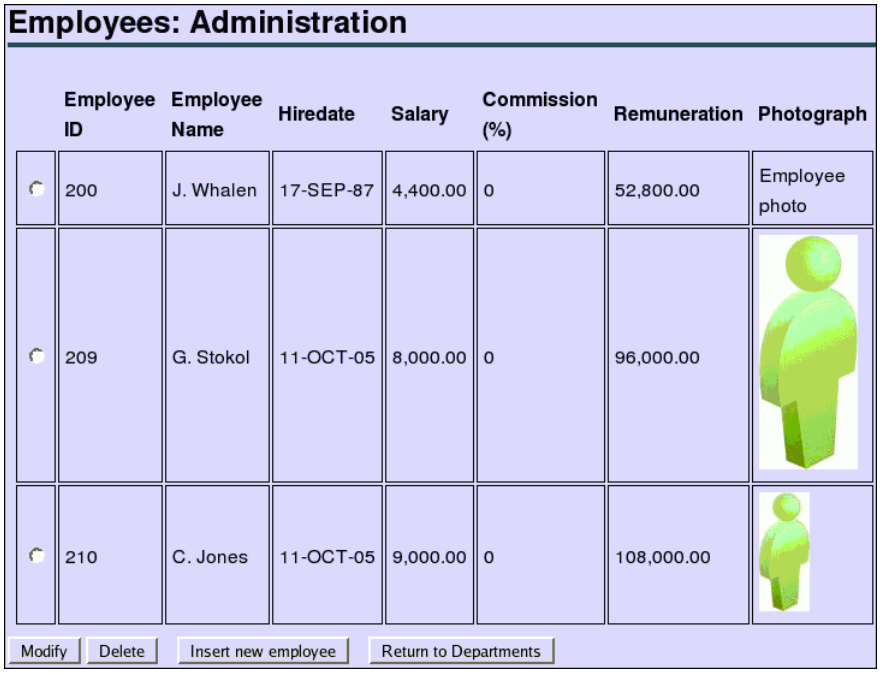
14. サイズが 100 ピクセルより大きい JPEG イメージを検索して選択し、「**Open**」をクリックします。



15. 「Insert New Image」ページで、「**Save**」をクリックします。



「Employees」ページに、新しくアップロードした JPEG イメージが表示されます。そのイメージ・サイズは、イメージ・サイズ変更コードを含める前にロードしたイメージと比較すると小さくなります。



Employees: Administration

	Employee ID	Employee Name	Hiredate	Salary	Commission (%)	Remuneration	Photograph
○	200	J. Whalen	17-SEP-87	4,400.00	0	52,800.00	Employee photo
○	209	G. Stokol	11-OCT-05	8,000.00	0	96,000.00	
○	210	C. Jones	11-OCT-05	9,000.00	0	108,000.00	

Modify | Delete | Insert new employee | Return to Departments

8

グローバル・アプリケーションの構築

この章では、PHP および Oracle Database 環境でのグローバル・アプリケーションの開発について説明します。グローバルなインターネット・アプリケーションの開発およびデプロイに関連付けられている基本的なタスク（ロケール認識の開発、ユーザー選択言語での HTML コンテンツの構築、ユーザーのロケールの表記規則に従ったデータの表示など）について説明します。

様々なロケールがサポートされているグローバルなインターネット・アプリケーションを構築するには、優れた開発プラクティスが必要です。ロケールとは、各国語およびその言語が話されている地域のことです。アプリケーション自体がユーザーのロケール・プリファレンスを認識し、ユーザーが考えているとおりの表記規則に従ってコンテンツを表示できる必要があります。適切なロケール特性（正しい日付書式や数値書式など）でデータを表示することが重要です。Oracle Database は、完全な国際化を実現しており、グローバル・アプリケーションを開発およびデプロイするためのグローバルなプラットフォームを提供します。

この章の内容は次のとおりです。

Oracle と PHP 間の環境の確立

グローバル・アプリケーションを構築するための最初の手順は、PHP エンジンと Oracle データベースの間に接続を正しく設定する手順です。これで、すべての層にわたってデータ整合性が保証されます。インターネット・ベースのほとんどの標準規格で、文字エンコーディングとして Unicode がサポートされているため、この章ではデータ交換用のキャラクタ・セットとして Unicode を使用します。

Zend Core for Oracle は Oracle OCI アプリケーションであり、OCI に適用されるルールは PHP にも適用されます。Oracle ロケールの動作（OCI アプリケーションで使用されるクライアント・キャラクタ・セットを含む）は、NLS_LANG 環境変数で定義します。この環境変数の形式は次のとおりです。

```
<language>_<territory>.<character set>
```

たとえば、ドイツのドイツ語ユーザーが Unicode でアプリケーションを実行している場合、NLS_LANG は次のように設定されています。

```
GERMAN_GERMANY.AL32UTF8
```

言語および地域の設定によって、Oracle 日付書式、エラー・メッセージ言語、ソート順序に使用されるルールなどの Oracle の動作が制御されます。キャラクタ・セット AL32UTF8 は、UTF-8 の Oracle 名です。

NLS_LANG 環境変数の詳細は、Oracle Database のインストレーション・ガイドを参照してください。

Zend Core for Oracle が Apache にインストールされている場合は、/etc/profile に NLS_LANG を設定できます。

```
export NLS_LANG GERMAN_GERMANY.AL32UTF8
```

Zend Core for Oracle が Oracle HTTP Server にインストールされている場合は、$ORACLE_HOME/opmn/conf/opmn.xml に環境変数として NLS_LANG を設定する必要があります

```
<ias-component id="HTTP_Server">
  <process-type id="HTTP_Server" module-id="OHS">
    <environment>
      <variable id="PERL5LIB"
       value="D:\oracle\1012J2EE\Apache\Apache\mod_perl\site\5.6.1\lib"/>
      <variable id="PHPRC" value="D:\oracle\1012J2EE\Apache\Apache\conf"/>
      <variable id="NLS_LANG" value="german_germany.al32utf8"/>
    </environment>
    <module-data>
      <category id="start-parameters">
        <data id="start-mode" value="ssl-disabled"/>
      </category>
    </module-data>
    <process-set id="HTTP_Server" numprocs="1"/>
  </process-type>
</ias-component>
```

この変更を実装するには、Web リスナーを再起動する必要があります。

文字列の操作

PHP は、ISO-8859-1 キャラクタ・セットで動作するように設計されています。それ以外のキャラクタ・セット（特にマルチバイト・キャラクタ・セット）を処理する場合は、一連のマルチバイト文字列関数を使用できます。これらの関数を有効にするには、Zend Core for Oracle コンソールを開き、「Configuration」タブに移動します。

「Extensions」サブタブにナビゲートし、「Zend Core Extensions」ツリー・コントロールを開きます。

アプリケーション・コードで `mb_strlen()` などの関数を使用して、文字列内の文字数を計算する必要があります。これによって、文字列内のバイト数を戻す `strlen()` とは異なる値が戻される場合があります。

mbstring 拡張モジュールを有効にし、Web サーバーを再起動すると、いくつかの構成オプションを使用できるようになります。`mbstring.func_overload` を Overload 設定のいずれかに設定して、標準の PHP 文字列関数の動作を変更できます。

詳細は、PHP の mbstring に関する次のリファレンス・マニュアルを参照してください。

`http://www.php.net/mbstring`

ユーザーのロケールの決定

グローバル環境のアプリケーションでは、ロケール・プリファレンスが異なるユーザーに対応する必要があります。ユーザーの優先ロケールを決定した後は、そのロケールの言語で HTML コンテンツを構築し、そのロケールの表記規則に従います。

ユーザーのロケールを決定する一般的な方法として、ブラウザのデフォルトの ISO ロケール設定を使用する方法があります。通常、ブラウザは、Accept Language HTTP ヘッダーで HTTP サーバーにそのロケール・プリファレンス設定を送信します。Accept Language HTTP ヘッダーが NULL の場合は、使用可能なロケール・プリファレンス情報がなく、事前定義のデフォルトのロケールにフォールバックされます。

次の PHP コードは、`$_SERVER` サーバー変数を使用して、Accept Language HTTP ヘッダーから ISO ロケールを取り出します。

```
$s = $_SERVER["HTTP_ACCEPT_LANGUAGE"]
```

ロケール認識の開発

ユーザーのロケール・プリファレンスを決定すると、日付、時間、通貨の書式などのロケール依存の関数をコールして、ロケールの表記規則に従って HTML ページを書式設定できます。

様々なプログラミング環境で実装されるグローバル・アプリケーションを構築する場合は、異なる環境間でユーザー・ロケール設定を同期できるようにする必要があります。たとえば、PL/SQL プロシージャをコールする PHP アプリケーションでは、PL/SQL プロシージャをコールする前に、ISO ロケールを対応する `NLS_LANGUAGE` 値および `NLS_TERRITORY` 値にマップし、ユーザーのロケールに合わせてパラメータ値を変更する必要があります。PL/SQL UTL_I18N パッケージには、ISO ロケールと Oracle ロケール間でマップできるマッピング関数が含まれています。

表 8-1 に、よく使用されるロケールを ISO 環境および Oracle 環境に定義する方法を示します。

表 8-1 ISO、SQL、PL/SQL の各プログラミング環境でのロケールの表現

ロケール	ロケール ID	NLS_LANGUAGE	NLS_TERRITORY
中国語（中華人民共和国）	zh-CN	SIMPLIFIED CHINESE	CHINA
中国語（台湾）	zh-TW	TRADITIONAL CHINESE	TAIWAN
英語（アメリカ合衆国）	en-US	AMERICAN	AMERICA
英語（イギリス）	en-GB	ENGLISH	UNITED KINGDOM

表 8-1 ISO、SQL、PL/SQL の各プログラミング環境でのロケールの表現（続き）

ロケール	ロケール ID	NLS_LANGUAGE	NLS_TERRITORY
フランス語（カナダ）	fr-CA	CANADIAN FRENCH	CANADA
フランス語（フランス）	fr-FR	FRENCH	FRANCE
ドイツ語	de	GERMAN	GERMANY
イタリア語	it	ITALIAN	ITALY
日本語	ja	JAPANESE	JAPAN
韓国語	ko	KOREAN	KOREA
ポルトガル語（ブラジル）	pt-BR	BRAZILIAN PORTUGUESE	BRAZIL
ポルトガル語	pt	PORTUGUESE	PORTUGAL
スペイン語	es	SPANISH	SPAIN

HTML ページのエンコーディング

HTML ページのエンコーディングは、ブラウザおよびインターネット・アプリケーションに関する重要な情報です。ページ・エンコーディングは、インターネット・アプリケーションがサービスを提供しているロケールで使用されるキャラクタ・セットとみなすことができます。ブラウザは、正しいフォントおよびキャラクタ・セットのマッピング表を使用して HTML ページを表示できるように、ページ・エンコーディングを認識している必要があります。インターネット・アプリケーションは、HTML フォームに入力されたデータを処理できるように、HTML ページ・エンコーディングを認識している必要があります。

ロケールごとに異なるネイティブ・エンコーディングを使用するのではなく、すべてのページ・エンコードで UTF-8（Unicode エンコーディング）を使用することをお薦めします。このエンコーディングを使用すると、グローバル・アプリケーションのコーディングを簡略化できるのみでなく、単一ページに多言語コンテンツを配置できます。

HTML ページ用のページ・エンコーディングの指定

HTML ページのエンコーディングを HTTP ヘッダーまたは HTML ページ・ヘッダーに指定できます。

HTTP ヘッダーへのエンコーディングの指定

HTML ページ・エンコーディングを HTTP ヘッダーに指定するには、HTTP 仕様に Content-Type HTTP ヘッダーを含めます。このヘッダーで、コンテンツ・タイプおよびキャラクタ・セットを指定します。Content-Type HTTP ヘッダーの形式は次のとおりです。

```
Content-Type: text/html; charset=utf-8
```

charset パラメータで、HTML ページのエンコーディングを指定します。charset パラメータに指定可能な値は、ブラウザでサポートされている文字エンコーディングの IANA 名です。

HTML ページ・ヘッダーへのエンコーディングの指定

この方法は、主に静的な HTML ページに使用します。HTML ページ・エンコーディングを HTML ページ・ヘッダーに指定するには、文字エンコーディングを次のように HTML ヘッダーに指定します。

```
<meta http-equiv="Content-Type" content="text/html;charset=utf-8">
```

charset パラメータで、HTML ページのエンコーディングを指定します。Content-Type HTTP の場合と同様に、charset パラメータに指定可能な値は、ブラウザでサポートされている文字エンコーディングの IANA 名です。

PHP へのページ・エンコーディングの指定

default_charset 構成変数を次のように設定して、Content-Type HTTP ヘッダーの HTML ページのエンコーディングを PHP に指定できます。

```
default_charset = UTF-8
```

これは、Zend Core for Oracle コンソールの「Configuration」タブにあります。「PHP」サブタブを選択し、「Data Handling」ツリー・コントロールを開きます。値を入力した後、構成設定を保存し、Web サーバーを再起動します。

この設定によって送信ページは変換されません。アプリケーションでは、サーバー生成のページを UTF-8 でエンコードする必要があります。

翻訳のための HTML ページのコンテンツの編成

ユーザー・インタフェースをユーザーのローカル言語で使用できるようにすることは、アプリケーションのグローバル化の基本的なタスクです。HTML ページのコンテンツの翻訳可能なソースは、次のように分類されます。

- アプリケーション・コードに含まれているテキスト文字列
- 静的な HTML ファイル、イメージ・ファイル、および CSS などのテンプレート・ファイル
- データベースに格納されている動的なデータ

PHP の文字列

PHP アプリケーション・ロジック内の翻訳可能な文字列を外部化して、そのテキストを簡単に翻訳できるようにする必要があります。これらのテキスト・メッセージは、翻訳対象データのタイプおよびボリュームに応じて、フラット・ファイルまたはデータベース表に格納できます。

静的ファイル

HTML ファイルや GIF ファイルなどの静的ファイルは簡単に翻訳できます。これらのファイルを翻訳する場合は、UTF-8 をファイル・エンコーディングとして指定して、対応する言語に翻訳する必要があります。翻訳したファイルの言語を区別するには、静的ファイルを言語ごとに異なるディレクトリにステージングするか、または異なるファイル名でステージングします。

データベースのデータ

通常、製品名や製品説明などの動的な情報がデータベースに格納されています。様々な翻訳を区別するには、この情報を保持しているデータベース・スキーマに言語を示すための列を含める必要があります。目的の言語を選択するには、問合せに WHERE 句を含める必要があります。

ユーザーが考えているとおりの表記規則を使用したデータの表示

アプリケーションのデータは、ユーザーが考えているとおりに表示される必要があります。そうでない場合、データの意味が誤って解釈される可能性があります。たとえば、12/11/05 という日付は、アメリカ合衆国では 2005 年 12 月 11 日を意味するのに対して、イギリスでは 2005 年 11 月 12 日を意味します。ユーザーの数値書式および通貨書式にも同様の違いが存在します。たとえば、. という記号はアメリカ合衆国では小数点セパレータですが、ドイツでは千単位セパレータです。

各言語に独自のソート・ルールがあります。アルファベットの文字の順序に従う言語、文字の画数に従う言語、単語の発音に従う言語などがあります。ユーザーが慣れている言語順序でソートされていないデータを表示すると、情報の検索が難しくなり、時間がかかることがあります。

アプリケーション・ロジックおよびデータベースから取り出すデータの量によっては、アプリケーション・レベルではなくデータベース・レベルでデータを書式設定した方がより適切な場合があります。Oracle Database には、ユーザーのロケール・プリファレンスがわかっている場

合にデータの表示を調整するのに役立つ機能が多く用意されています。以降の項では、SQL でのロケール依存の操作の例を示します。

Oracle の日付書式

Oracle Database の日付表示書式には、標準の日付、短い日付、長い日付の 3 種類があります。次の例では、アメリカ合衆国およびドイツでの短い日付書式と長い日付書式の違いを示します。

```
SQL> alter session set nls_territory=america nls_language=american;

Session altered.


SQL> select employee_id EmpID,
  2  substr(first_name,1,1)||'.'||last_name "EmpName",
  3  to_char(hire_date,'DS') "Hiredate",
  4  to_char(hire_date,'DL') "Long HireDate"
  5  from employees
  6* where employee_id <105;

     EMPID EmpName                   Hiredate   Long HireDate
---------- ------------------------- ---------- ----------------------------
       100 S.King                    06/17/1987 Wednesday, June 17, 1987
       101 N.Kochhar                 09/21/1989 Thursday, September 21, 1989
       102 L.De Haan                 01/13/1993 Wednesday, January 13, 1993
       103 A.Hunold                  01/03/1990 Wednesday, January 3, 1990
       104 B.Ernst                   05/21/1991 Tuesday, May 21, 1991

SQL> alter session set nls_territory=germany nls_language=german;

Session altered.

SQL> select employee_id EmpID,
  2  substr(first_name,1,1)||'.'||last_name "EmpName",
  3  to_char(hire_date,'DS') "Hiredate",
  4  to_char(hire_date,'DL') "Long HireDate"
  5  from employees
  6* where employee_id <105;

     EMPID EmpName                   Hiredate Long HireDate
---------- ------------------------- -------- ------------------------------
       100 S.King                    17.06.87 Mittwoch, 17. Juni 1987
       101 N.Kochhar                 21.09.89 Donnerstag, 21. September 1989
       102 L.De Haan                 13.01.93 Mittwoch, 13. Januar 1993
       103 A.Hunold                  03.01.90 Mittwoch, 3. Januar 1990
       104 B.Ernst                   21.05.91 Dienstag, 21. Mai 1991
```

Oracle の数値書式

次の例では、アメリカ合衆国とドイツでの小数点文字およびグループ・セパレータの違いを示します。

```
SQL> alter session set nls_territory=america;

Session altered.

SQL> select employee_id EmpID,
  2  substr(first_name,1,1)||'.'||last_name "EmpName",
  3  to_char(salary, '99G999D99') "Salary"
  4  from employees
  5* where employee_id <105
```

```
     EMPID EmpName                   Salary
---------- ------------------------- ----------
       100 S.King                     24,000.00
       101 N.Kochhar                  17,000.00
       102 L.De Haan                  17,000.00
       103 A.Hunold                    9,000.00
       104 B.Ernst                     6,000.00

SQL> alter session set nls_territory=germany;

Session altered.

SQL> select employee_id EmpID,
  2  substr(first_name,1,1)||'.'||last_name "EmpName",
  3  to_char(salary, '99G999D99') "Salary"
  4  from employees
  5* where employee_id <105

     EMPID EmpName                   Salary
---------- ------------------------- ----------
       100 S.King                     24.000,00
       101 N.Kochhar                  17.000,00
       102 L.De Haan                  17.000,00
       103 A.Hunold                    9.000,00
       104 B.Ernst                     6.000,00
```

Oracle の言語ソート

スペイン語では、伝統的に *ch*、*ll* および *Ò* が一意の文字として扱われ、それぞれ *c*、*l* および *n* の後に順序付けられます。次の例では、Chen と Chung という従業員名にスペイン語のソートを使用した場合の結果を示します。

```
SQL> alter session set nls_sort=binary;

Session altered.

SQL> select employee_id EmpID,
  2         last_name "Last Name"
  3  from employees
  4  where last_name like 'C%'
  5* order by last_name

     EMPID Last Name
---------- -------------------------
       187 Cabrio
       148 Cambrault
       154 Cambrault
       110 Chen
       188 Chung
       119 Colmenares

6 rows selected.

SQL> alter session set nls_sort=spanish_m;

Session altered.

SQL> select employee_id EmpID,
  2         last_name "Last Name"
  3  from employees
  4  where last_name like 'C%'
  5* order by last_name
```

```
     EMPID Last Name
---------- -------------------------
       187 Cabrio
       148 Cambrault
       154 Cambrault
       119 Colmenares
       110 Chen
       188 Chung

6 rows selected.
```

Oracle のエラー・メッセージ

NLS_LANGUAGE パラメータは、データベースから戻されるデータベース・エラー・メッセージの言語も制御します。SQL 文を発行する前にこのパラメータを設定すると、言語固有のデータベース・エラー・メッセージがアプリケーションに戻されるようになります。

次のサーバー・メッセージについて考えてみます。

```
ORA-00942: table or view does not exist
```

NLS_LANGUAGE パラメータをフランス語に設定すると、このサーバー・メッセージは次のように表示されます。

```
ORA-00942: table ou vue inexistante
```

Oracle Database のグローバリゼーション・サポート機能の詳細は、『Oracle Database 2 日で開発者ガイド』のグローバル環境での作業に関する項を参照してください。

索引

記号

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

S

T

U

V

W

Z

あ

い

え

か

き

く

け

こ

さ

し

せ

そ

て

は

ひ

ふ

へ

ゆ

ろ