

Pro*C/C++ for Windows プリコンパイラ

スタート・ガイド

リリース 8.1.6

2000年4月

部品番号 : J01325-01

Pro*C/C++ for Windows プリコンパイラ・スタート・ガイド リリース 8.1.6

部品番号 : J01325-01

原本名 : Pro*C/C++ Precompiler Getting Started, Release 8.1.6 for Windows

原本部品番号 : A73023-01

原本協力者 : Riaz Ahmed, Eric Belden, Sharon Castledine, Joseph Garcia, Lisa Giambruno, Neeraj Gupta, Bernie Harris, Ana Hernandez, Mark Kennedy, Robert Knecht, Shiva Prasad, Ali Shehade, Helen Slattery, Jeff Stein, Nicole Sullivan, Janice Wong, Martha Woo, Ravi Gooty

Copyright © 1994, 2000, Oracle Corporation. All rights reserved.

Printed in Japan.

制限付権利の説明

プログラム（ソフトウェアおよびドキュメントを含む）の使用、複製または開示は、オラクル社との契約に記載された制約条件に従うものとします。著作権、特許権およびその他の知的財産権に関する法律により保護されています。

当プログラムのリバース・エンジニアリング等は禁止されています。

このドキュメントの情報は、予告なしに変更されることがあります。オラクル社は本ドキュメントの無謬性を保証しません。

* オラクル社とは、Oracle Corporation（米国オラクル）または日本オラクル株式会社（日本オラクル）を指します。

危険な用途への使用について

オラクル社製品は、原子力、航空産業、大量輸送、医療あるいはその他の危険が伴うアプリケーションを用途として開発されておりません。オラクル社製品を上述のようなアプリケーションに使用することについての安全確保は、顧客各位の責任と費用により行ってください。万一かかる用途での使用によりクレームや損害が発生いたしましても、日本オラクル株式会社と開発元である Oracle Corporation（米国オラクル）およびその関連会社は一切責任を負いかねます。当プログラムを米国防総省の米国政府機関に提供する際には、『Restricted Rights』と共に提供してください。この場合次の Notice が適用されます。

Restricted Rights Notice

Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

このドキュメントに記載されているその他の会社名および製品名は、あくまでその製品および会社を識別する目的にのみ使用されており、それぞれの所有者の商標または登録商標です。

目次

はじめに	vii
前提条件	viii
対象読者	viii
このドキュメントの構成	viii
規則	viii
ドキュメント・ライブラリ	xi
関連ドキュメント	xii
1 Pro*C/C++ の概要	
Pro*C/C++ の概要	1-2
機能	1-2
制限事項	1-2
ディレクトリ構造	1-2
2 Pro*C/C++ の使用方法	
グラフィカル・ユーザー・インタフェースの使用法	2-2
タイトル・バー	2-2
メニュー・バー	2-3
ツールバー	2-3
情報ペイン	2-4
ステータス・バー	2-5
Pro*C/C++ プロジェクトの作成およびプリコンパイル	2-5
プロジェクトを開く	2-5
出力ファイルのデフォルト拡張子の設定	2-6
既存の入力または出力ファイル名の変更	2-6

プロジェクトへのファイルの追加	2-7
プロジェクトからのファイルの削除	2-8
プリコンパイラ・オプションの設定	2-8
データベース接続情報の指定	2-10
Pro*C/C++ プロジェクトのプリコンパイル	2-11
結果の確認	2-11
エラーの修正	2-12
Pro*C/C++ の終了	2-12
コマンド行での Pro*C/C++ の使用方法	2-13
ヘッダー・ファイル	2-13
ライブラリ・ファイル	2-14
マルチスレッド・アプリケーションの作成	2-14
プリコンパイラ・オプション	2-14
構成ファイル	2-15
CODE	2-15
DBMS	2-15
INCLUDE	2-15
PARSE	2-15
Oracle XA ライブラリと Pro*C/C++ の使用方法	2-15
Pro*C/C++ プログラムの XA とのコンパイルおよびリンク	2-16
XA 動的登録	2-16
現行セッションの環境変数の追加	2-17
すべてのセッションに対するレジストリ変数の追加	2-17
XA および TP モニターに関する情報	2-18

3 サンプル・プログラム

サンプル・プログラムの説明	3-2
サンプル表の作成	3-7
サンプル・プログラムの作成	3-7
サンプルの .pre ファイルのパスの設定	3-8

A Microsoft Visual C++ への Pro*C/C++ の統合

Pro*C/C++ の Microsoft Visual C++ プロジェクトへの統合	A-2
Pro*C/C++ 実行可能プログラムの位置の指定	A-2
Pro*C/C++ ヘッダー・ファイルの位置の指定	A-3

プロジェクトへの .pc ファイルの追加	A-3
プロジェクトへの .c ファイルの参照の追加	A-4
プロジェクトへの Pro*C/C++ のライブラリの追加	A-4
「カスタム ビルド」オプションの指定	A-5
「ツール」メニューへの Pro*C/C++ の追加	A-6

索引

はじめに

このドキュメントでは Microsoft Windows オペレーティング・システム上で実行する Pro*C/C++ プリコンパイラの初歩的な情報を提供します。次の項目について説明します。

- [前提条件](#)
- [対象読者](#)
- [このドキュメントの構成](#)
- [規則](#)
- [ドキュメント・ライブラリ](#)
- [関連ドキュメント](#)

前提条件

このドキュメントでは読者が次の技術に精通していることを前提にしています。

- ファイルのコピーおよび削除のコマンド、検索パス、サブディレクトリおよびパス名の概念
- Microsoft Windows NT または Windows 95/98 オペレーティング・システム
- Microsoft Visual C++ バージョン 6.0 以降

対象読者

このドキュメントは Microsoft Windows で Pro*C/C++ プリコンパイラを使用するユーザーを対象としています。

このドキュメントの構成

このドキュメントは次のように構成されています。

第 1 章「Pro*C/C++ の概要」

Microsoft Windows NT および Windows 95/98 オペレーティング・システムで実行される C および C++ 言語の Oracle プログラム・インタフェースである Pro*C/C++ について説明します。

第 2 章「Pro*C/C++ の使用方法」

プロジェクトの作成およびプリコンパイルの方法を説明します。Windows のメニューやアイコンまたはそれと同等のキーボード操作によりコマンドを実行できる Pro*C/C++ グラフィカル・ユーザー・インタフェースおよびコマンド行での Pro*C/C++ の使用方法についても説明します。

第 3 章「サンプル・プログラム」

このリリースに含まれているサンプル・プログラムを使用した Pro*C/C++ の Oracle データベース・アプリケーションの作成方法を説明し、マルチスレッド・アプリケーションの作成方法の概要についても説明します。

第 4 章「Microsoft Visual C++ への Pro*C/C++ の統合」

Pro*C/C++ を Microsoft Visual C++ 開発環境に統合する方法を説明します。

規則

このドキュメントで使用される表記規則は次のとおりです。

規則	例	意味
大文字	SQL> ALTER DATABASE	コマンド名、SQL 予約語、キーワードを示します。
イタリック	変数を示すために使用 <i>filename</i>	入力が必要な値を示します。たとえば、コマンドで <i>filename</i> を入力するように要求された場合、ファイルの実際の名前を入力します。
大カッコ []	x:¥[pathname]¥oracle¥home_name	オプション項目を示します。たとえば、OFA 準拠の Oracle ホーム・ディレクトリを作成する場合、¥oracle パス名の前にパス名をオプションとして指定できます。 大カッコはファンクション・キーも表します。たとえば [Enter] です。
C:¥>	C:¥ORACLE>	Windows プラットフォームの現行のハード・ディスク・ドライブのコマンド・プロンプトを示します。プロンプトは異なることがあり、現在作業しているサブディレクトリが反映されることもあります。このドキュメントでは "MS-DOS コマンド・プロンプト" として参照されます。
ディレクトリ名の前の円記号 (¥)	¥bin	ディレクトリがルート・ディレクトリのサブディレクトリであることを示します。
oracle_home および oracle_base	oracle_base¥oracle_home¥bin ディレクトリに移動します。	この Optimal Flexible Architecture (OFA) に準拠したリリースでは、すべてのサブディレクトリは最上位の oracle_home ディレクトリの下にはありません。新しい最上位ディレクトリの名前は oracle_base で、このディレクトリのデフォルトは c:¥oracle です。Oracle ホーム・ディレクトリは oracle_base の下にあります。 Oracle8i リリース 8.1.6 を他の Oracle ソフトウェアがインストールされていないコンピュータにインストールする場合、最初の Oracle ホーム・ディレクトリのデフォルト設定は c:¥oracle¥ora81 です。Oracle Universal Installer を再度実行してリリース 8.2.x をインストールする場合、2 番目の Oracle ホーム・ディレクトリは ¥ora82 です。 このドキュメントで例として使用されているディレクトリ・パスは、すべて OFA に準拠しています。OFA の詳細は『Oracle8i for Windows NT 管理者ガイド』を参照してください。

規則	例	意味
HOME_NAME	OracleHOME_NAMETNSListener	Oracle ホーム名を示します。ホーム名は英数字 16 文字までです。ホーム名で使用できる特殊文字は、アンダースコアのみです。
HOMEID	HOME0、HOME1、HOME2	製品をインストールする各 Oracle ホーム・ディレクトリの一意なレジストリ・サブキーを示します。あるコンピュータ上の異なる Oracle ホーム・ディレクトリに製品をインストールするたびに、新しい HOMEID が作成されて番号が増加します。各 HOMEID には、インストールされた Oracle 製品固有の構成パラメータが含まれます。
記号	ピリオド . カンマ , ハイフン - セミコロン ; コロン : 等号 = 円記号 ¥ 一重引用符 ' 二重引用符 " 丸カッコ ()	大カッコと垂直バー以外のコマンド内の記号は表記どおりに入力する必要があります。

ドキュメント・ライブラリ

このドキュメントは Oracle ドキュメント・ライブラリの 1 つです。Oracle ドキュメント・ライブラリは次の 2 種類のドキュメントで構成されています。

ドキュメントの種類	説明
オペレーティング・システム固有	Windows NT または Windows 95/98 環境での Oracle 製品のインストール、構成および使用方法。オペレーティング・システム固有のドキュメントは共通ドキュメント・セットを参照することがあります。これらのドキュメントのタイトルには固有のオペレーティング・システム名が必ず含まれているため簡単に識別できます。
共通	<p>すべてのオペレーティング・システム・プラットフォームに共通する Oracle データベース、Oracle ネットワークおよびアプリケーション・プログラミング・インタフェース情報を説明。ドキュメント・セットの大部分のドキュメントはこのカテゴリに分類されます。この共通ドキュメント・セットを読んでいくと、Windows NT または Windows 95/98 のオペレーティング・システムに固有の処理について、プラットフォームまたはオペレーティング・システム用のドキュメントを参照するように求められる場合があります。</p> <p>共通ドキュメントの参照先が使用するオペレーティング・システムのドキュメントのどこにあるかを簡単に特定するには、このドキュメントの索引で次の項目を探してください。</p> <p>共通ドキュメントの参照先</p> <p>このドキュメントで説明している共通ドキュメントの参照先がすべて、この索引項目の下に示されます。</p>

関連ドキュメント

詳細は次のドキュメントを参照してください。

- 『Oracle8i for Windows NT インストレーション・ガイド』
- 『Oracle8i for Windows NT リリース・ノート』
- 『Oracle8i for Windows NT 管理者ガイド』
- 『Oracle Enterprise Manager 管理者ガイド』
- 『Net8 管理者ガイド』
- 『Oracle8i Parallel Server 概要』
- 『Oracle Parallel Server for Windows NT 管理者ガイド』
- 『Oracle8i 概要』
- 『Oracle8i リファレンス・マニュアル』
- 『Oracle8i エラー・メッセージ』
- 『Oracle8i Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』

Pro*C/C++ の概要

この章では、Windows オペレーティング・システムで実行される C および C++ 言語のプログラム・インタフェースである Pro*C/C++ について説明します。Pro*C/C++ により Win32 環境の Oracle データベース・アプリケーションを作成できます。次の項目について説明します。

- Pro*C/C++ の概要
- 機能
- 制限事項
- ディレクトリ構造

注意： 追加情報は『Oracle8i Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』を参照してください。

Pro*C/C++ の概要

Pro*C/C++ プリコンパイラを使用することにより Oracle データベースにアクセスするアプリケーションを迅速に、また他のシステムとの互換性を考慮しながら作成できます。

Pro*C/C++ プログラミング・ツールにより C または C++ プログラムに SQL 文を埋め込むことができます。Pro*C/C++ プリコンパイラは埋込み SQL 文を標準の Oracle ランタイム・ライブラリ・コールに変換し、通常の方法でコンパイル、リンクおよび実行できる変更されたソース・プログラムを生成します。

機能

Pro*C/C++ では次の機能がサポートされます。

- Oracle データベースへのローカル・アクセスおよび Net8 によるリモート・アクセス
- 埋込み PL/SQL ブロック
- クライアント / サーバー環境で高いパフォーマンスを提供するデータベース・コールのまとめ
- 埋込み SQL プログラムの ANSI 完全準拠
- PL/SQL バージョン 8.0 および PL/SQL プロシージャ内のホスト言語配列
- マルチスレッド・アプリケーション
- ANSI C 完全準拠
- 32 ビット・アプリケーション用 Microsoft Visual C++ バージョン 6.0 のサポート

注意： Borland C++ はサポートされていません。

制限事項

Pro*C/C++ リリース 8.1.6 では 16 ビット・コードの生成はサポートされていません。

ディレクトリ構造

Oracle ソフトウェアをインストールすると Oracle 製品のハード・ドライブにディレクトリ構造が作成されます。メインの Oracle ディレクトリにはサブディレクトリおよび Pro*C/C++ の実行に必要なファイルが含まれます。

Pro*C/C++ をインストールするとき Oracle Universal Installer によって `precomp` というディレクトリが `oracle_base\oracle_home` ディレクトリに作成されます。このサブディレクトリには表 1-1 「[precomp ディレクトリ構造](#)」にリストされた Pro*C/C++ 実行可能プログラム、ファイル、ライブラリおよびサンプル・プログラムが含まれます。

表 1-1 precomp ディレクトリ構造

ディレクトリ名	内容
¥admin	構成ファイル
¥demo¥proc	Pro*C/C++ 用サンプル・プログラム
¥demo¥sql	サンプル・プログラム用 SQL スクリプト
¥doc¥proc	Pro*C/C++ 用 README ファイル
¥help¥proc	Pro*C/C++ 用ヘルプ・ファイル
¥lib¥msvc	Pro*C/C++ 用ライブラリ・ファイル
¥mesg	メッセージ・ファイル
¥misc¥proc	Pro*C/C++ 用の他のファイル
¥public	ヘッダー・ファイル

注意： ¥precomp ディレクトリには Pro*COBOL などの他の製品のファイルが格納されている場合があります。

Pro*C/C++ の使用方法

この章ではプロジェクトの作成およびプリコンパイルの方法を説明します。また、Windows のメニューやアイコンやキーボード操作によってコマンドを実行できる Pro*C/C++ グラフィカル・ユーザー・インタフェースの説明、およびコマンド行での Pro*C/C++ の使用方法の説明も行います。次の項目について説明します。

- [グラフィカル・ユーザー・インタフェースの使用方法](#)
- [Pro*C/C++ プロジェクトの作成およびプリコンパイル](#)
- [コマンド行での Pro*C/C++ の使用方法](#)
- [ヘッダー・ファイル](#)
- [ライブラリ・ファイル](#)
- [マルチスレッド・アプリケーションの作成](#)
- [プリコンパイラ・オプション](#)
- [Oracle XA ライブラリと Pro*C/C++ の使用方法](#)

注意： Pro*C/C++ の追加情報は『Oracle8i Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』を参照してください。

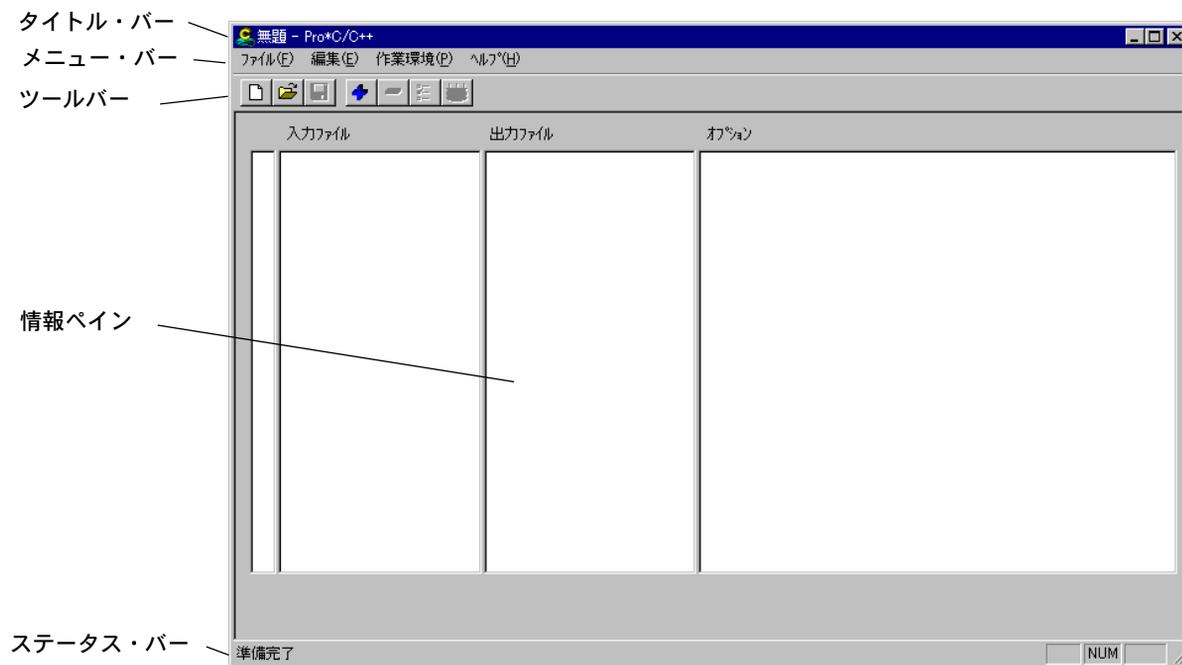
グラフィカル・ユーザー・インタフェースの使用法

Pro*C/C++ プロジェクトの作成およびプリコンパイルの指示に従って作業を開始する前に、Pro*C/C++ グラフィカル・ユーザー・インタフェースの基本的なコマンド、ダイアログ・ボックス、メニューおよびボタンを知っておく必要があります。

Pro*C/C++ グラフィカル・インタフェースの起動

グラフィカル・ユーザー・インタフェースを起動するには、「スタート」→「プログラム」→「Oracle - HOME_NAME」→「Application Development」→「Pro C_C++」を選択します。

Pro*C/C++ プリコンパイル環境には次の図に示された5つの要素があります。



タイトル・バー

タイトル・バーには Pro*C/C++ プロジェクトの名前が表示されます。現行のプロジェクトにまだ名前を付けていない場合は、「名称未設定」という名前が表示されます。

メニュー・バー

メニュー・バーには次のメニューがあります。

メニュー	説明
ファイル	新規 Pro*C/C++ プロジェクトの作成、既存の Pro*C/C++ プロジェクトを開く、アクティブな Pro*C/C++ プロジェクトの同じ名前または別名での保存、Oracle データベースへの接続文字列の指定、Pro*C/C++ プロジェクトのプリコンパイル、およびアプリケーションの終了を行います。
編集	Pro*C/C++ プロジェクトへのファイルの追加、Pro*C/C++ プロジェクトからのファイルの削除、プリコンパイラ・オプションの表示または変更を行います。
作業環境	出力ファイルのデフォルト・ファイル拡張子を設定します。
ヘルプ	Pro*C/C++ アプリケーションのバージョン番号および著作権情報を表示する「Pro*C/C++ のバージョン情報」があります。

ツールバー

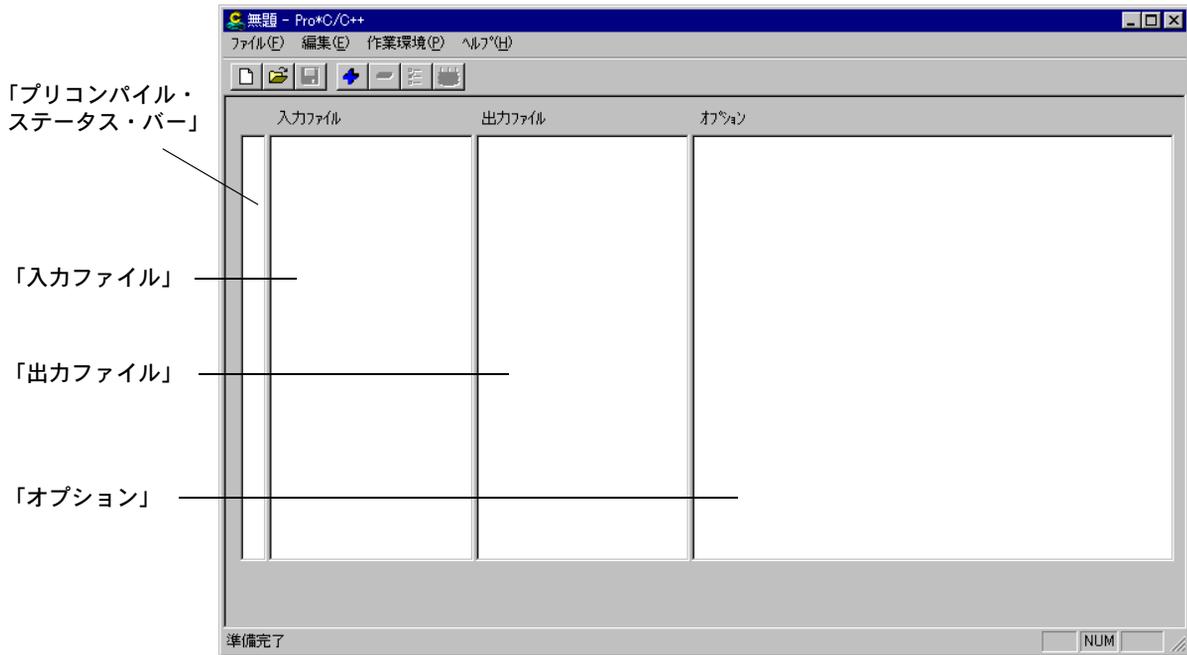
ツールバーの次のボタンをクリックすると対応するコマンドを実行できます。



ボタン	説明
新規作成	新規 Pro*C/C++ プロジェクトの作成
開く	既存の Pro*C/C++ プロジェクトを開く
上書き保存	アクティブな Pro*C/C++ プロジェクトを同じ名前で保存
追加	Pro*C/C++ プロジェクトにファイルを追加
削除	Pro*C/C++ プロジェクトからファイルを削除
オプション	プリコンパイラ・オプションを表示または変更
プリコンパイル	Pro*C/C++ プロジェクトのプリコンパイル

情報ペイン

情報ペインは次の図に示した4つの要素から成ります。



要素	説明
プリコンパイル・ステータス・バー	ファイルのプリコンパイルの成功または失敗を表示
入力ファイル	プリコンパイルされる Pro*C/C++ プロジェクトのファイルを表示
出力ファイル	プリコンパイルされた後の Pro*C/C++ プロジェクトのファイルを表示
オプション	デフォルトのオプションとは異なるプリコンパイラ・オプションを表示

プリコンパイル処理が完了した後、プリコンパイル・ステータス・バーに次の3つのアイコンのうちの1つが表示されます。

- 緑色のチェック・マークはファイルのプリコンパイルが成功したことを示します。
- 黄色のチェック・マークはファイルのプリコンパイルは成功したが1つ以上の警告メッセージがあることを示します。
- 赤色のXマークはファイルのプリコンパイルが失敗したことを示します。

ステータス・アイコンをダブルクリックすると、「プリコンパイル状態」ダイアログ・ボックスが表示されます。このダイアログ・ボックスには警告または失敗の理由に関する詳細情報が表示されます。

ステータス・バー

ウィンドウの一番下のステータス・バーにプリコンパイルの進行状況に関する情報が表示されます。また、ツールバー・ボタンまたはメニュー・コマンドにマウス・カーソルを合わせると、そのボタンやコマンドの用途がステータス・バーに表示されます。

Pro*C/C++ プロジェクトの作成およびプリコンパイル

この項では Pro*C/C++ プロジェクトの作成およびプリコンパイルの手順を説明します。Pro*C/C++ アプリケーションを起動した後、次の手順を実行します。

- プロジェクトを開く
- 出力ファイルのデフォルト拡張子の設定
- 既存の入力または出力ファイル名の変更
- プロジェクトへのファイルの追加
- プロジェクトからのファイルの削除
- プリコンパイラ・オプションの設定
- データベース接続情報の指定
- Pro*C/C++ プロジェクトのプリコンパイル
- 結果の確認
- エラーの修正
- Pro*C/C++ の終了

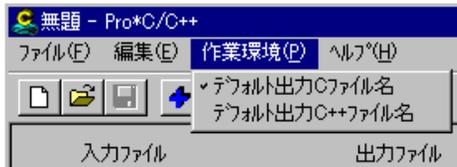
プロジェクトを開く

Pro*C/C++ では一度に1つのプロジェクトのみを開きます。プロジェクトは1つ以上のプリコンパイル可能ファイルで構成されます。プロジェクト・ファイルには拡張子 .PRE が付いています。

- 新規プロジェクトを作成するには「ファイル」→「新規プロジェクト」を選択します。
- 既存のプロジェクトを開くには「ファイル」→「プロジェクトを開く」を選択します。

出力ファイルのデフォルト拡張子の設定

出力ファイルのデフォルト拡張子を設定するには「作業環境」メニューを使用します。



この設定はこの後入力する入力ファイルにのみ影響を与えます。既存の出力ファイル名は変更されません。しかし、既存の出力ファイル名は出力ファイルをダブルクリックして新しい名前を入力すると変更できます。

- 「デフォルト出力C ファイル名」を選択した場合、出力ファイルのデフォルト拡張子は .c になります。
- 「デフォルト出力C++ ファイル名」を選択した場合、出力ファイルのデフォルト拡張子は .cpp になります。
- 「デフォルト出力C ファイル名」および「デフォルト出力C++ ファイル名」を両方とも選択解除した場合は、入力ファイルを追加するときに「出力ファイル名」ダイアログ・ボックスが表示されます。
 - 選択したファイルの出力ファイル名を入力します。ファイル名を選択または入力すると、名前が情報ペインの「出力ファイル」領域に表示されます。

既存の入力または出力ファイル名の変更

既存の入力または出力ファイルの名前を変更する手順は次のとおりです。

1. 情報ペインの「入力ファイル」領域または「出力ファイル」領域に表示されたファイル名をダブルクリックします。「入力ファイル」または「出力ファイル」ダイアログ・ボックスが表示されます。

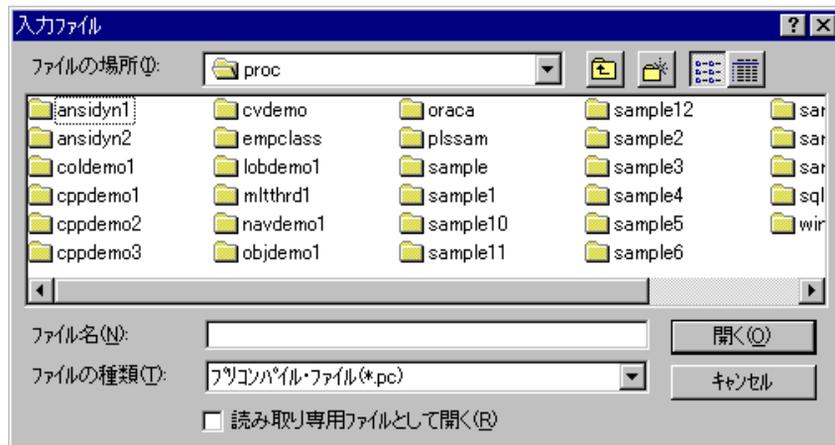


2. 古いファイル名を新しいファイル名で置き換えます。
3. 「開く」をクリックします。

プロジェクトへのファイルの追加

プロジェクトにファイルを追加する手順は次のとおりです。

1. 「編集」 → 「追加」を選択します。「入力ファイル」ダイアログ・ボックスが表示されます。



2. .pc ファイルを1つ以上選択します。隣接していないファイルを複数選択するには、[Ctrl] キーとマウスを使用します。
3. 「開く」をクリックします。選択したファイルが情報ペインに表示されます。

プロジェクトからのファイルの削除

必要であれば、プロジェクトから1つ以上のファイルを簡単に削除できます。

プロジェクトからファイルを削除する手順は次のとおりです。

1. 情報ペインでファイルを選択します。
2. 「編集」 → 「削除」を選択します。
3. 「はい」をクリックします。

プリコンパイラ・オプションの設定

プリコンパイラ・オプションを使用してリソースの使用方法、エラーのレポート方法、入力ファイルおよび出力ファイルの形式設定方法、カーソルの管理方法を制御できます。

プリコンパイル・オプションを設定する手順は次のとおりです。

1. 「入力ファイル」リストからファイルを1つ以上選択します。
2. 「編集」 → 「オプション」を選択します。「オプション」ダイアログ・ボックスが表示されます。



デフォルト・オプションは新たに追加されたすべてのファイルに有効です。オプションのデフォルト値を変更すると「オプション」ダイアログ・ボックスの一番下の「オプション文字列」編集フィールドおよび情報ペインの「オプション」領域に変更内容が表示されます。オプションの追加情報は 2-14 ページの「プリコンパイラ・オプション」を参照してください。

3. プリコンパイラがディスクに書き出す出力リスト・ファイルの形式を変更するには、「リスト / エラー」ボタンをクリックします。「リスト / エラー」ダイアログ・ボックスが表示されます。



設定には生成されるエラー情報の種類およびリスト・ファイルの名前が含まれます。

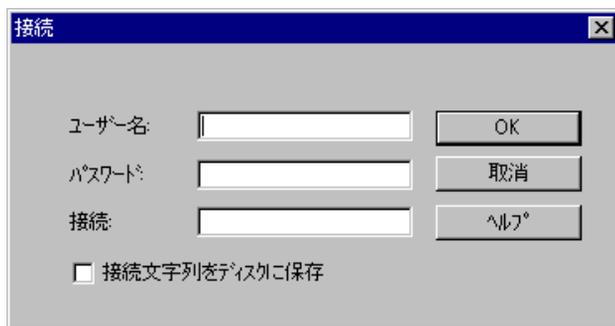
4. 「オプション」ダイアログ・ボックスでオプションを設定した後、「OK」をクリックします。

データベース接続情報の指定

「オプション」ダイアログ・ボックスの「SQL チェック」オプションで「semantics」または「full」を選択した場合、Oracle データベースに対するデータベース接続情報の指定が必要になることがあります。データ操作文または PL/SQL ブロックで参照されるすべての表が DECLARE TABLE 文で定義されている場合には、Oracle データベースに接続する必要はありません。

データベース接続情報を指定する手順は次のとおりです。

1. 「ファイル」→「接続」を選択します。「接続」ダイアログ・ボックスが表示されます。



2. このダイアログ・ボックスを使用してプリコンパイルを行う前にデータベース接続情報を指定します。この時点ではデータベース接続は行われません。「semantics」または

「full」のSQLチェックが必要なすべてのファイルに対して指定できるデータベース接続情報は1セットのみです。

3. 前に答えていなければプリコンパイル時に「接続」ダイアログ・ボックスが自動的に表示されます。ユーザー名、パスワードおよびネットワーク・サービス名（データベース別名）を入力します。ネットワーク・サービス名はローカル・データベースには必要ありません。
4. 次回の Pro*C/C++ セッションで使用するために接続情報を保存する場合は、「接続文字列をディスクに保存」チェック・ボックスを選択します。このチェック・ボックスを選択しておかないと、プリコンパイルするたびにこの情報を入力しなければなりません。
5. 「OK」をクリックします。

Pro*C/C++ プロジェクトのプリコンパイル

「入力ファイル」リスト内のファイルは何個でもプリコンパイルできます。

プリコンパイルを行う手順は次のとおりです。

1. 「入力ファイル」リストからファイルを1つ以上選択します。[Ctrl] キーとマウスを使用すると隣接していない複数のファイル（たとえば、リストの1番目と3番目のファイル）を選択できます。
2. 「ファイル」→「プリコンパイル」を選択します。
プリコンパイルが完了するとダイアログ・ボックスに「プリコンパイルが終了しました。」というメッセージが表示され、「取消」ボタンが「OK」に変わります。
3. 「OK」をクリックします。

注意：「取消」をクリックしてもプリコンパイル中のファイルの処理は中断できませんが、残りのファイルのプリコンパイルは停止できます。

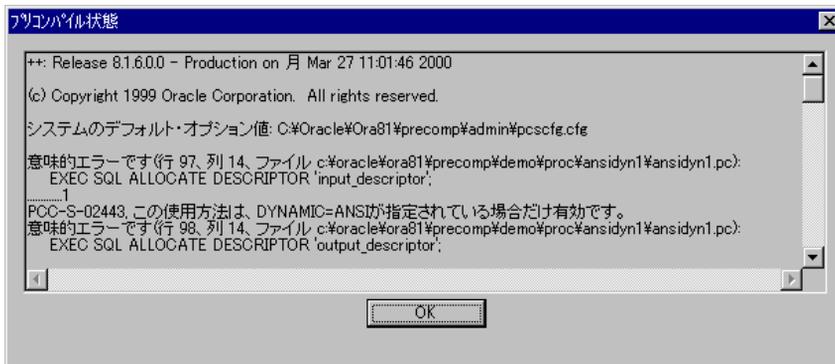
結果の確認

プリコンパイルの結果は成功、警告付きの成功または失敗です。プリコンパイルが終了したときにプリコンパイル・ステータス・バーをチェックします。

- 緑色のチェック・マークはファイルのコンパイルが成功したことを示します。
- 黄色のチェック・マークはファイルのコンパイルは成功したが1つ以上の警告メッセージがあることを示します。
- 赤色の X マークはファイルのコンパイルが失敗したことを示します。

エラーの修正

ステータス・バーに黄色のチェック・マークまたは赤色の X マークが表示された場合は、アイコンをダブルクリックします。「プリコンパイル状態」ダイアログ・ボックスが表示されます。このダイアログ・ボックスは、警告メッセージまたはプリコンパイルが失敗した理由をリストします。例を次に示します。



開発環境に切り替えて問題を修正します。エラーを修正した後、再度プリコンパイルします。

注意: PCC-S-02014 エラー (行 *num*、列 *colnam*、ファイル *name* での構文エラー) を受け取った場合は、次の操作を行います。

- 問題が発生した INCLUDE ファイルがあるディレクトリにバッチ・ファイル `mod_incl.bat` および `add_new1.bat` をコピーします。これらのファイルは `oracle_base\oracle_home\precomp\misc\proc` ディレクトリにあります。
 - `mod_incl.bat` を実行します。
-

Pro*C/C++ の終了

Pro*C/C++ を終了するには、「ファイル」→「終了」を選択します。何らかの形でプロジェクトが変更されていれば、それを保存するようにというプロンプトが出されます。

提案: 変更されたファイルだけでなく元のファイルもそのまま保持する場合は、「別名保存」コマンドを選択します。「上書き保存」コマンドは前のバージョンに上書きします。

コマンド行での Pro*C/C++ の使用方法

コマンド行でファイルをプリコンパイルするには次のコマンドを入力します。

```
C:¥> proc iname=filename.pc
```

filename.pc はファイルの名前です。ファイルが現行の作業ディレクトリ内に存在しない場合は、そのファイルのフル・パスを INAME 引数の後に指定します。

Pro*C/C++ は filename.c を生成します。生成されたファイルを C コンパイラでコンパイルします。

ヘッダー・ファイル

oracle_base¥oracle_home¥precomp¥public ディレクトリには Pro*C/C++ ヘッダー・ファイルが格納されています。

追加情報： oraca.h、sqlca.h および sqllda.h の詳細は『Oracle8i Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』を参照してください。

ヘッダー・ファイル	説明
oraca.h	ランタイム・エラーを診断したり、プログラムによるさまざまな Oracle8 リソースの使用状況をモニターするために役立つ Oracle 通信領域 (ORACA) が含まれています。
sql2oci.h	Oracle Call Interface (OCI) 環境ハンドルおよび OCI サービス・コンテキストを Pro*C/C++ アプリケーション内から取得できるようにする SQLLIB 関数が含まれています。
sqlapr.h	OCI とともに使用できる外部関数の ANSI プロトタイプが含まれています。
sqlca.h	ランタイム・エラーの診断に役立つ、SQL 通信領域 (SQLCA) が含まれています。SQLCA は実行可能な SQL 文が実行されるたびに更新されます。
sqlcpr.h	Pro*C/C++ によって生成される SQLLIB 関数のプラットフォーム固有の ANSI プロトタイプが含まれています。デフォルトでは、Pro*C/C++ は SQL プログラミング・コールの全関数のプロトタイピングをサポートしません。この機能が必要な場合はアプリケーションのソース・ファイル内のどの EXEC SQL 文よりも前に sqlcpr.h をインクルードします。
sqllda.h	動的 SQL のメソッド 4 を使用するプログラムに必要なデータ構造である、SQL 記述子領域 (SQLDA) が含まれています。
sqlkpr.h	OCI とともに使用できる外部関数の KR プロトタイプが含まれています。
sqlproto.h	sqlproto.h ヘッダー・ファイルは Pro*C/C++ リリース 8.0.3 で廃止になりました。sqlproto.h のかわりに sqlcpr.h を使用します。ただし sqlproto.h を使用して作成されたアプリケーションは変更せずに作成できます。sqlcpr.h を含んだダミーの sqlproto.h ファイルが oracle_base¥oracle_home¥precomp¥public ディレクトリにあります。

ライブラリ・ファイル

`oracle_base\oracle_home\precomp\lib\msvc` ディレクトリには Pro*C/C++ アプリケーションをリンクするときに使用するライブラリ・ファイルが格納されています。このライブラリ・ファイルの名前は `orasql8.lib` です。

Pro*C/C++ のアプリケーション・プログラム・インタフェース (API) コールは、使用している Pro*C/C++ ソフトウェアに付属の DLL ファイル内で実装されています。その DLL を使用するには、Pro*C/C++ の DLL に対応するインポート・ライブラリ (.lib ファイル) とアプリケーションをリンクする必要があります。さらに、Pro*C/C++ アプリケーションを実行するコンピュータ上に DLL ファイルがインストールされていることを確認する必要があります。

Microsoft によって、`libc.lib`、`libcmtd.lib` および `msvcrt.lib` の 3 つのライブラリが提供されています。Oracle の DLL は `msvcrt.lib` ランタイム・ライブラリを使用します。他の 2 つの Microsoft ライブラリではなく、必ず `msvcrt.lib` とリンクしてください。

マルチスレッド・アプリケーションの作成

同時データベース操作を実行するのであればマルチスレッド・アプリケーションを作成します。

Windows NT および Windows 95/98 はプロセスに属するスレッドをスケジューリングして割り当てます。スレッドはプログラムの実行のパスです。これはカーネル・スタック、CPU レジスタの状態、スレッド環境ブロックおよびユーザー・スタックから成ります。各スレッドはプロセスのリソースを共有します。マルチスレッド・アプリケーションはプロセスのリソースを使用して個々のスレッドのアクティビティを調整します。

マルチスレッド・アプリケーションを作成する場合、C/C++ コードがリエントラントであることが必要です。つまり静的またはグローバル・データへのアクセスは一度に 1 つのスレッドに限られる必要があります。マルチスレッドと非リエントラント関数が混在すると、スレッドは別のスレッドが必要とする情報を変更する可能性があります。

Pro*C/C++ プリコンパイラは、スレッドのローカル・スタックに変数を自動作成します。これにより確実に Pro*C/C++ 関数を使用するスレッドが変数の一意の集合にアクセスし、リエントラントであることが保証されます。

追加情報： Pro*C/C++ を使用したマルチスレッド・アプリケーションの書き方の追加情報は『Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』を参照してください。

プリコンパイラ・オプション

この項では Windows プラットフォーム用の Pro*C/C++ に関連した問題を取り上げます。

追加情報: プリコンパイラ・オプションの詳細は『Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』の「プリコンパイラ・オプション」の章を参照してください。

構成ファイル

構成ファイルとはプリコンパイラ・オプションを含むテキスト・ファイルのことです。

このリリースでは、システム構成ファイルの名前は `pcscfg.cfg` です。このファイルは `oracle_base\oracle_home\precomp\admin` ディレクトリにあります。

CODE

CODE オプションのデフォルト値は `ANSI_C` に設定されています。他のオペレーティング・システム用の Pro*C/C++ では、デフォルト値が `KR_C` に設定されている場合もあります。

DBMS

`CHAR_MAP=VARCHAR2` を使用している場合、`DBMS=V6_CHAR` はサポートされません。その場合は `DBMS=V7` を使用してください。

INCLUDE

Pro*C/C++ グラフィカル・ユーザー・インタフェースでは、INCLUDE パス・ディレクトリを入力するには「オプション」ダイアログ・ボックスの「インクルード・ディレクトリ」フィールドを使用します。複数のパスを入力する場合は、それぞれのパスをセミコロンで区切ります。セミコロンの後に空白を入れしないでください。空白を入れると各ディレクトリの前に別個の "INCLUDE=" 文字列が挿入されてしまいます。

`PARSE=PARTIAL` または `PARSE=FULL` を指定してサンプル・プログラムをプリコンパイルした場合、`c:\program files\devstudio\vc\include` のインクルード・パスが追加されます。Microsoft Visual C++ が別の場所にインストールされている場合は、「インクルード・ディレクトリ」フィールドをそれに応じて変更します。これによりサンプル・プログラムを正しくプリコンパイルできます。

PARSE

PARSE オプションのデフォルト値は `NONE` に設定されています。他のオペレーティング・システム用の Pro*C/C++ では、デフォルト値が `FULL` に設定されている場合もあります。

Oracle XA ライブラリと Pro*C/C++ の使用方法

Oracle データベースが次のトランザクション処理 (TP) モニターと対話できるようにするには通常、XA アプリケーション・プログラム・インタフェース (API) を使用します。

- BEA Tuxedo
- IBM Transarc Encina
- IBM CICS

クライアント・プログラムで TP モニターの文を使用することもできます。さらに XA API の使用は Pro*C/C++ および OCI の両方でサポートされています。

Oracle XA ライブラリは Oracle8i Enterprise Edition の一部として自動的にインストールされます。Oracle ホーム・ディレクトリに次のコンポーネントが作成されます。

コンポーネント	位置
oraxa8.lib	oracle_base¥oracle_home¥rdbms¥xa
xa.h	oracle_base¥oracle_home¥rdbms¥xa

Pro*C/C++ プログラムの XA とのコンパイルおよびリンク

Pro*C/C++ プログラムを XA とコンパイルおよびリンクする手順は次のとおりです。

1. Pro*C/C++ を使用した *filename.pc* をプリコンパイルして *filename.c* を生成します。
2. パスに *oracle_base¥oracle_home¥rdbms¥xa* が含まれていることを確認して *filename.c* をコンパイルします。
3. *filename.obj* を次のライブラリとリンクします。

ライブラリ	位置
oraxa8.lib	oracle_base¥oracle_home¥rdbms¥xa
oci.lib	oracle_base¥oracle_home¥oci¥lib¥msvc
orasql8.lib	oracle_base¥oracle_home¥precomp¥lib¥msvc

4. *filename.exe* を実行します。

XA 動的登録

Oracle は XA 動的登録の使用をサポートしています。XA 動的登録により XA 対応 TP モニターとのインタフェースを持つアプリケーションのパフォーマンスが向上します。

Windows NT の Oracle データベースで TP モニターが XA 動的登録を使用するには、環境変数またはレジストリ変数のどちらかを、TP モニターが実行されている Windows NT コンピュータに追加する必要があります。手順は次の項のいずれかを参照してください。

- 現行セッションの環境変数の追加

- すべてのセッションに対するレジストリ変数の追加

現行セッションの環境変数の追加

MS-DOS コマンド・プロンプトで環境変数を追加すると、現行の MS-DOS セッションにのみ影響を与えます。

現行セッションに環境変数を追加する手順は次のとおりです。

1. TP モニターがインストールされているコンピュータに移動します。
2. MS-DOS コマンド・プロンプトで次のように入力します。

```
C:¥> set ORA_XA_REG_DLL = vendor.dll
```

vendor.dll は、ベンダーから提供された TP モニターの DLL です。

すべてのセッションに対するレジストリ変数の追加

レジストリ変数を追加すると Windows NT コンピュータのすべてのセッションに影響します。TP モニターが 1 つだけ実行中のコンピュータでは、この方法が便利です。

すべてのセッションに対してレジストリ変数を追加する手順は次のとおりです。

1. TP モニターがインストールされているコンピュータに移動します。
2. MS-DOS コマンド・プロンプトで次のように入力します。

```
C:¥> regedt32
```

「レジストリ エディタ」ウィンドウが表示されます。

3. HKEY_LOCAL_MACHINE¥SOFTWARE¥ORACLE に移動します。
4. 「編集」メニューから「値の追加」を選択します。
「値の追加」ダイアログ・ボックスが表示されます。
5. 「値の名前」フィールドに ORA_XA_REG_DLL を入力します。
6. 「データの種類」ドロップダウン・リスト・ボックスから REG_EXPAND_SZ を選択します。
7. 「OK」をクリックします。
「文字列エディタ」ダイアログ・ボックスが表示されます。
8. 「文字列」フィールドに *vendor.dll* を入力します。*vendor.dll* はベンダーから提供された TP モニターの DLL です。
9. 「OK」をクリックします。

レジストリ エディタによってパラメータが追加されます。

10. 「レジストリ」メニューから「レジストリ エディタの終了」を選択します。
レジストリ エディタが終了します。

XA および TP モニターに関する情報

XA および TP モニターの詳細は次の情報を参照してください。

- 『Transaction Processing XPG4 X/Open CAE Specification XO/CAE/91/300』または『C193 2/92』
- X/Open Company, Ltd., 1010 El Camino Real, Suite 380, Menlo Park, CA 94025, U.S.A.
- 使用している TP モニター固有のドキュメント

追加情報： Oracle XA ライブラリおよび XA 動的登録の詳細は『Oracle8i アプリケーション開発者ガイド 基礎編』を参照してください。

サンプル・プログラム

この章では、このリリースに含まれているサンプル・プログラムを使用した Pro*C/C++ の Oracle データベース・アプリケーションの作成方法を説明します。次の項目について説明します。

- サンプル・プログラムの説明
- サンプル表の作成
- サンプル・プログラムの作成

サンプル・プログラムの説明

Pro*C/C++ をインストールするときに、Oracle Installer によって Pro*C/C++ の一連のサンプル・プログラムが `oracle_base\oracle_home\precomp\demo\proc` ディレクトリにコピーされます。これらのサンプル・プログラムは表 3-1 「サンプル・プログラム」 にリストされています。SQL スクリプトは `oracle_base\oracle_home\precomp\demo\sql` ディレクトリにあります。

これらのサンプル・プログラムをビルドおよび実行して、Pro*C/C++ が正しくインストールされて正常に動作するか確認されることをお勧めします。使用した後は、プログラムを削除してもかまいません。

Oracle から提供された Pro*C/C++ のサンプル・プログラムをビルドすると、.exe 実行可能ファイルが生成されます。サンプル・プログラムによってはプリコンパイルし実行する前に、サンプル・ディレクトリにある SQL スクリプトを実行する必要があります。SQL スクリプトは、サンプル・プログラムが正常に実行されるように、表およびデータを適切に設定します。

表 3-1 サンプル・プログラム

サンプル・プログラム	ソース・ファイル	Pro*C/C++ GUI プロジェクト・ ファイル	MSVC コンパイラ・ プロジェクト・ ファイル	注意
ANSIDYN1	ansidyn1.pc	ansidyn1.pre	ansidyn1.dsp	
ANSIDYN2	ansidyn2.pc	ansidyn2.pre	ansidyn2.dsp	
COLDEMO1	coldemo1.h coldemo1.pc coldemo1.sql coldemo1.typ	coldemo1.pre	coldemo1.dsp	coldemo1 を作成する前に coldemo1.sql および Object Type Translator を実行します。
CPPDEMO1	cppdemo1.pc	cppdemo1.pre	cppdemo1.dsp	
CPPDEMO2	cppdemo2.pc empclass.pc cppdemo2.sql empclass.h	cppdemo2.pre	cppdemo2.dsp	cppdemo2 を作成する前に cppdemo2.sql を実行します。
CPPDEMO3	cppdemo3.pc	cppdemo3.pre	cppdemo3.dsp	
CVDEMO	cv_demo.pc cv_demo.sql	cv_demo.pre	cv_demo.dsp	cv_demo を作成する前に cv_demo.sql を実行します。
EMPCLASS	cppdemo2.pc empclass.pc cppdemo2.sql empclass.h	empclass.pre	empclass.dsp	empclass を作成する前に cppdemo2.sql を実行します。

サンプル・プログラム	ソース・ファイル	Pro*C/C++ GUI プロジェクト・ ファイル	Msvc コンパイラ・ プロジェクト・ ファイル	注意
LOBDEMO1	lobdemo1.h lobdemo1.pc lobdemo1.sql	lobdemo1.pre	lobdemo1.dsp	lobdemo1 を作成する前に lobdemo1.sql を実行します。
MLTTHRD1	mltthrd1.pc mltthrd1.sql	mltthrd1.pre	mltthrd1.dsp	mltthrd1 を作成する前に mltthrd1.sql を実行します。
NAVDEMO1	navdemo1.h navdemo1.pc navdemo1.sql navdemo1.typ	navdemo1.pre	navdemo1.dsp	navdemo1 を作成する前に navdemo1.sql および Object Type Translator を実行します。
OBJDEMO1	objdemo1.h objdemo1.pc objdemo1.sql objdemo1.typ	objdemo1.pre	objdemo1.dsp	objdemo1 を作成する前に objdemo1.sql および Object Type Translator を実行します。
ORACA	oraca.pc oracatst.sql	oraca.pre	oraca.dsp	oraca を作成する前に oracatst.sql を実行します。
PLSSAM	plssam.pc	plssam.pre	plssam.dsp	
SAMPLE	sample.pc	sample.pre	sample.dsp	
SAMPLE1	sample1.pc	sample1.pre	sample1.dsp	
SAMPLE2	sample2.pc	sample2.pre	sample2.dsp	
SAMPLE3	sample3.pc	sample3.pre	sample3.dsp	
SAMPLE4	sample4.pc	sample4.pre	sample4.dsp	
SAMPLE5	sample5.pc exampbld.sql exemplod.sql	sample5.pre	sample5.dsp	sample5 を作成する前に exampbld.sql そして exemplod.sql を実行します。
SAMPLE6	sample6.pc	sample6.pre	sample6.dsp	
SAMPLE7	sample7.pc	sample7.pre	sample7.dsp	
SAMPLE8	sample8.pc	sample8.pre	sample8.dsp	
SAMPLE9	sample9.pc calldemo.sql	sample9.pre	sample9.dsp	sample9 を作成する前に calldemo.sql を実行します。
SAMPLE10	sample10.pc	sample10.pre	sample10.dsp	
SAMPLE11	sample11.pc sample11.sql	sample11.pre	sample11.dsp	sample11 を作成する前に sample11.sql を実行します。
SAMPLE12	sample12.pc	sample12.pre	sample12.dsp	

サンプル・プログラム	ソース・ファイル	Pro*C/C++ GUI プロジェクト・ ファイル	MSVC コンパイラ・ プロジェクト・ ファイル	注意
SQLVCP	sqlvcp.pc	sqlvcp.pre	sqlvcp.dsp	
WINSAM	resource.h winsam.h winsam.ico winsam.pc winsam.rc	winsam.pre	winsam.dsp	

ここではサンプル・プログラムの機能について説明します。

ANSIDYN1

実行時まで認識されない SQL 文を ANSI 動的 SQL を使用して処理する方法を示します。このプログラムは、ANSI 動的 SQL を使用した最も簡単な（ただし最も効率的というわけではない）アプローチの例です。

ANSIDYN2

実行時まで認識されない SQL 文を ANSI 動的 SQL を使用して処理する方法を示します。このプログラムでは、Oracle によるバッチ処理および参照用の拡張が使用されています。

COLDEMO1

カリフォルニア州の複数の郡についての調査情報をフェッチします。このプログラムはコレクション型のデータベース列をナビゲートする様々な方法の例です。

CPPDEMO1

ユーザーが従業員番号を入力すると EMP 表に問合せが発行され、従業員の名前、給与および歩合を取得します。このプログラムでは（標識構造体内の）標識変数を使用して歩合が NULL でないかどうかを判別します。

CPPDEMO2

指定した部門の全従業員の名前を EMP 表から取り出します（動的 SQL 方法 3）。

CPPDEMO3

このプログラムではすべての営業担当者を検索して名前と総所得（歩合を含む）を出力します。このプログラムは C++ の継承の例です。

CVDEMO

このプログラムでは参照カーソルを宣言し開きます。

EMPCLASS

EMPCLASS および CPPDEMO2 ファイルは C++ フレームワーク内での Pro*C/C++ プログラムのコーディング方法の例です。EMPCLASS は emp 表に関する特定の問合せをカプセル化しており、カーソル変数を使用してインプリメントされています。EMPCLASS は問合せの

インスタンスをインスタンス化し、emp クラスに属す C++ メンバー関数によりカーソル変数の機能（開く、フェッチ、閉じるなど）を提供します。empclass.pc ファイルはスタンドアロンのデモ・プログラムではありません。このファイルは cppdemo2 デモ・プログラムによって使用されます。emp クラスを使用するには、emp クラスのインスタンスを宣言してこのクラスのメンバー関数をコールするドライバ (cppdemo2.pc) を作成する必要があります。

LOBDEMO1

個人の社会保障番号に基づいて、データベースに対する犯罪記録のフェッチを行います。このプログラムは、表にアクセスしてラージ・オブジェクト (LOB) を格納するメカニズム、および DBMS_LOB パッケージにより使用可能となるストアド・プロシージャで LOB を操作するメカニズムの例です。

MLTTHRD1

スレッドをプリコンパイラとともに使用する方法を示しています。このプログラムではスレッドと同じ数のセッションが作成されます。2-14 ページの「[マルチスレッド・アプリケーションの作成](#)」を参照してください。

NAVDEMO1

オブジェクト・キャッシュ内でのオブジェクト間のナビゲーション・アクセスの例です。

OBJDEMO1

オブジェクトの使用法の例です。このプログラムではオブジェクト型 "person" および "address" を操作します。

ORACA

実行時にさまざまなパフォーマンス・パラメータを決定するために ORACA を使用する方法を示します。

PLSSAM

埋込み PL/SQL ブロックの使用法を示します。このプログラムではデータベースに登録されている従業員名を入力するように求められます。名前を入力すると PL/SQL ブロックが実行され、4つの SELECT 文の結果が戻されます。

SAMPLE

人事データベースに新しい従業員レコードを追加しデータベースの整合性をチェックします。データベース内での従業員番号としては現在の従業員番号の最大値 +10 の値が自動的に選択されます。

SAMPLE1

Oracle データベースにログオンしてユーザーに従業員番号を入力すると、データベースに問合せを発行して従業員の名前、給与および歩合を取得します。ユーザーが従業員番号として 0 (ゼロ) を入力するまでこの処理を繰り返します。

SAMPLE2

Oracle データベースにログオンし、カーソルを宣言して開き、すべての営業担当者の名前、給与および歩合をフェッチして結果を表示します。最後にカーソルを閉じます。

SAMPLE3

Oracle データベースにログオンし、カーソルを宣言して開き、配列を使用してバッチでフェッチを実行し `print_rows()` 関数を使用して結果を出力します。

SAMPLE4

LONG VARRAW 外部データ型を使用して型を同値化する使用例です。

SAMPLE5

口座番号と引出金額を入力するようにユーザーに求めます。プログラムは口座から借方を差し引く前に口座番号が正しいことおよび引出金額を取り出せるだけの預金があることを確認します。このプログラムは埋込み SQL の使用例です。

SAMPLE6

表を作成して行を挿入し、挿入をコミットして表を削除します（動的 SQL 方法 1）。

SAMPLE7

EMP 表に 2 つの行を挿入し、挿入した行を削除します（動的 SQL 方法 2）。

SAMPLE8

指定した部門の全従業員の名前を EMP 表から取り出します（動的 SQL 方法 3）。

SAMPLE9

SCOTT/TIGER アカウントを使用して Oracle データベースに接続します。いくつかのホスト配列を宣言し PL/SQL ストアド・プロシージャ（CALLDEMO パッケージの `GET_EMPLOYEES`）をコールします。PL/SQL プロシージャは最大 ASIZE 個の値を戻します。このプログラムはすべての行を取り出すまで、`GET_EMPLOYEES` をコールし、毎回 ASIZE 個分の配列を取得し、値を出力し続けます。

SAMPLE10

ユーザー名とパスワードを使用して Oracle データベースに接続し、SQL 文を入力するように求めます。有効な任意の SQL 文を入力できますが、埋込み SQL ではなく通常の SQL 構文を使用する必要があります。入力した文が処理されます。それが問合せであればフェッチされた行が表示されます（動的 SQL 方法 4）。

SAMPLE11

カーソル変数を使用して EMP 表からデータをフェッチします。カーソルを開く操作には `EMP_DEMO_PKG` パッケージ内の PL/SQL ストアド・プロシージャ `open_cur` を使用しています。

SAMPLE12

動的 SQL 方法 4 を使用して配列をフェッチする方法の例です。

SQLVCP

sqlvcp() 関数を使用して VARCHAR 構造体の実際のサイズを判断する方法の例です。この例では、サイズはポインタに加算して VARCHAR の配列を移動するためのオフセットとして使用されています。

このプログラムはさらに SQLStmtGetText() 関数を使用して最後に実行された SQL 文のテキストを取得する方法も示しています。

WINSAM

人事データベースに新しい従業員レコードを追加し、データベースの整合性をチェックします。必要な数の従業員名を入力できます。「Employee レコード」ダイアログ・ボックスの適切なボタンを選択して SQL コマンドを実行できます。これはサンプル・プログラムの GUI バージョンです。

サンプル表の作成

サンプル・プログラムを実行するには、ユーザー名が SCOTT でパスワードが TIGER のデータベース・アカウントが必要です。サンプル表 EMP および DEPT が入ったデータベースも必要です。このアカウントは Oracle8i Server の初期データベースに含まれています。このアカウントがデータベースにない場合はサンプル・プログラムを実行する前に作成します。

詳細は、『Oracle8i for Windows NT 管理者ガイド』を参照してください。データベースに EMP 表および DEPT 表がない場合は demobld.sql スクリプトを使用して作成します。

サンプル表の作成手順は次のとおりです。

1. SQL*Plus を起動します。
2. ユーザー名 SCOTT、パスワード TIGER で接続します。
3. demobld.sql スクリプトを実行します。

```
SQL> @ORACLE_BASE\ORACLE_HOME\SQLPLUS\DEMO\DEMOBLD.SQL;
```

サンプル・プログラムの作成

Microsoft Visual C++ 6.0 のプロジェクト・ファイルには拡張子 .dsp が付いています。 .dsp ファイルは oracle_base\oracle_home\precomp\demo\proc ディレクトリに存在し、サンプル・プログラムをプリコンパイル、コンパイルおよびリンクするために必要な手順のガイドおよび制御を行います。

Pro*C/C++、SQL*Plus および Object Type Translator は、Microsoft Visual C++ のサンプル・プロジェクト・ファイルに統合されています。コンパイル前に Pro*C/C++、SQL*Plus および Object Type Translator を個別に実行する必要はありません。詳細は [付録 A 「Microsoft Visual C++ への Pro*C/C++ の統合」](#) を参照してください。

追加情報： OTTの詳細は『Oracle8i Pro*C/C++ プリコンパイラ・プログラマーズ・ガイド』を参照してください。

サンプル・プログラムの作成手順は次のとおりです。

1. sample1.dsp などの Visual C++ のプロジェクト・ファイルを開きます。
2. プロジェクト・ファイルに指定されているパスを調べ、使用しているシステムの構成に対応していることを確認します。対応していない場合は正しくパスを変更します。コンポーネントへのパスに間違いがあるとシステムはエラー・メッセージを生成します。2-8 ページの「[プリコンパイラ・オプションの設定](#)」および 3-8 ページの「[サンプルの .pre ファイルのパスの設定](#)」を参照してください。

注意： サンプル・プログラムはすべて c:\oracle\ora81 をデフォルト・ドライブとして作成されています。

3. 「ビルド」→「リビルド」を選択します。Visual C++ によって実行ファイルが作成されます。

サンプルの .pre ファイルのパスの設定

デフォルトでは、サンプルの .pre ファイルは対応する .pc ファイルを c:\oracle\ora81 ディレクトリ内で検索します。c:\ は使用しているドライブで、oracle\ora81 は Oracle ホームの位置を表します。使用しているコンピュータで Oracle ベース・ディレクトリと Oracle ホーム・ディレクトリが異なる場合は、ディレクトリ・パスを正しいパスに変更する必要があります。

サンプルの .pre ファイルのパスの変更手順は次のとおりです。

1. Pro*C/C++ で .pre ファイルを開きます。
2. 「入力ファイル」領域でファイル名をダブルクリックして「入力ファイル」ダイアログ・ボックスを表示します。
3. ディレクトリ・パスを正しいパスに変更します。
4. 「開く」をクリックします。

Microsoft Visual C++ への Pro*C/C++ の統合

この付録では Pro*C/C++ を Microsoft Visual C++ 開発環境に統合する方法を説明します。
次の項目について説明します。

- Pro*C/C++ の Microsoft Visual C++ プロジェクトへの統合
- 「ツール」メニューへの Pro*C/C++ の追加

Pro*C/C++ の Microsoft Visual C++ プロジェクトへの統合

この項では Microsoft Visual C++ 6.0 プロジェクトに Pro*C/C++ を完全に統合する方法について説明します。

プリコンパイラのすべてのエラーと警告は Microsoft Visual C++ がコンパイラおよびリンカーのメッセージを表示する出力ボックスに表示されます。Microsoft Visual C++ の作成環境と別にファイルをプリコンパイルする必要はありません。さらに重要なこととして、Microsoft Visual C++ が .C ファイルと .PC ファイルの間の依存性をメンテナンスすることが挙げられます。必要であれば Microsoft Visual C++ によって依存性とプリコンパイル・ファイルがメンテナンスされます。

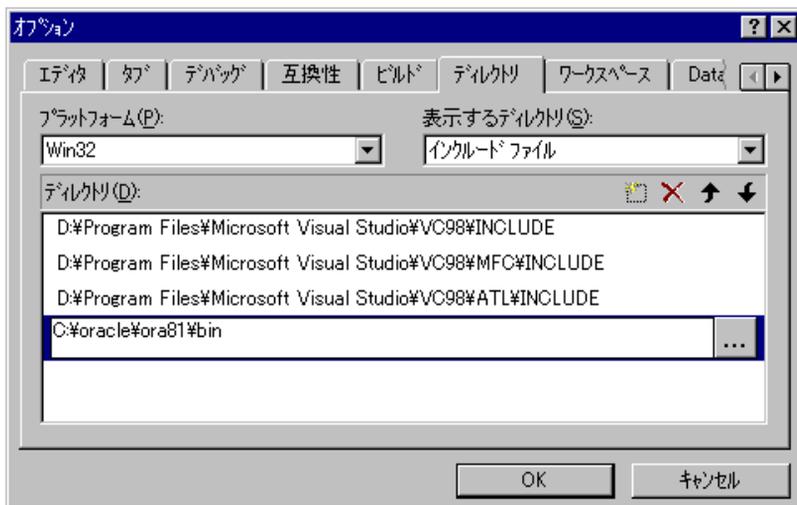
この項で説明する手順はすべて Microsoft Visual C++ の内部で実行します。

Pro*C/C++ 実行可能プログラムの位置の指定

Microsoft Visual C++ から Pro*C/C++ を実行するには、Pro*C/C++ 実行可能プログラムの位置を Microsoft Visual C++ に指定する必要があります。Microsoft Visual C++ をインストールした後で Oracle 8.1 製品をインストールした場合は、ディレクトリ・パスを追加してください。

Pro*C/C++ 実行可能プログラムの位置の指定手順は次のとおりです。

1. 「ツール」メニューから「オプション」を選択します。「オプション」ダイアログ・ボックスが表示されます。



2. 「ディレクトリ」タブをクリックします。
3. 「表示するディレクトリ」リスト・ボックスから実行可能ファイルを選択します。

4. 「ディレクトリ」フィールドの一番下までスクロールして点線の長方形をクリックします。
5. `oracle_base¥oracle_home¥bin` ディレクトリを入力します。例を次に示します。
`c:¥oracle¥ora81¥bin`
6. 「OK」をクリックします。

Pro*C/C++ ヘッダー・ファイルの位置の指定

Pro*C/C++ ヘッダー・ファイルの位置の指定手順は次のとおりです。

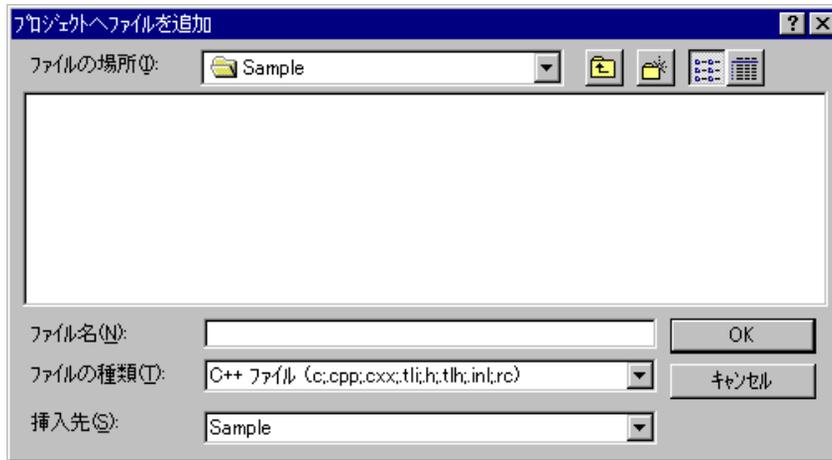
1. 「ツール」メニューから「オプション」を選択します。「オプション」ダイアログ・ボックスが表示されます。
2. 「ディレクトリ」タブをクリックします。
3. 「表示するディレクトリ」リスト・ボックスからインクルード・ファイルを選択します。
4. 「ディレクトリ」フィールドの一番下までスクロールして点線の長方形をクリックします。
5. `oracle_base¥oracle_home¥precomp¥public` ディレクトリを入力します。例を次に示します。
`c:¥oracle¥ora81¥precomp¥public`
6. 「OK」をクリックします。

プロジェクトへの .pc ファイルの追加

プロジェクトの作成後、.pc ファイルを追加する必要があります。

プロジェクトに .pc ファイルを追加する手順は次のとおりです。

1. 「プロジェクト」メニューから「プロジェクトへ追加」→「ファイル」を選択します。「プロジェクトへファイルを追加」ダイアログ・ボックスが表示されます。



2. 「ファイルの種類」リスト・ボックスで「すべてのファイル」を選択します。
3. .pc ファイルを選択します。
4. 「OK」をクリックします。

プロジェクトへの .c ファイルの参照の追加

各 .PC ファイルについて、プリコンパイルの結果作成される .C ファイルへの参照を追加する必要があります。

.c ファイルへの参照をプロジェクトに追加する手順は次のとおりです。

1. 「プロジェクト」メニューから「プロジェクトへ追加」→「ファイル」を選択します。「プロジェクトへファイルを追加」ダイアログ・ボックスが表示されます。
2. 「ファイル名」フィールドに .c ファイルの名前を入力します。
3. 「OK」をクリックします。.c ファイルがまだ作成されていないため、Microsoft Visual C++ により「指定されたファイルは存在しません。プロジェクトに対するファイルの参照を追加しますか?」というメッセージが表示されます。
4. 「はい」をクリックします。

プロジェクトへの Pro*C/C++ のライブラリの追加

Pro*C/C++ アプリケーションはライブラリ・ファイル orasql8.lib とリンクする必要があります。

プロジェクトへの Pro*C/C++ のライブラリの追加手順は次のとおりです。

1. 「プロジェクト」メニューから「プロジェクトへ追加」→「ファイル」を選択します。
「プロジェクトへファイルを追加」ダイアログ・ボックスが表示されます。
2. 「ファイルの種類」リスト・ボックスで「すべてのファイル」を選択します。
3. `oracle_base¥oracle_home¥precomp¥lib¥msvc` ディレクトリから `orasql8.lib` を選択します。
4. 「OK」をクリックします。

「カスタムビルド」オプションの指定

「カスタムビルド」オプションを指定する手順は次のとおりです。

1. FileView で .pc ファイルを右クリックして「設定」を選択します。「プロジェクト設定」ダイアログ・ボックスが現れ「カスタムビルド」タブが表示されます。



2. 「ビルドのコマンド」フィールドに次のコマンドを 1 行で入力します。

```
$(ProjDir)¥.¥.¥.¥.¥bin¥proc $(ProjDir)¥$(InputName) .pc
include=$(ProjDir)¥.¥.¥.¥public include="$(MSDEVDIR)¥.¥vc¥include"
```

\$(ProjDir) および \$MSDEVDIR は Microsoft Visual C++ の「カスタムビルド」コマンドのマクロです。詳細は Microsoft Visual C++ のドキュメントを参照してください。

3. 「出力ファイル」フィールドに次のどちらかを入力します。

生成するファイル	入力するファイル名
.C ファイル	\$(ProjDir)¥\$(InputName).c
.CPP ファイル	\$(ProjDir)¥\$(InputName).cpp

プロジェクトが作成される時 Microsoft Visual C++ は出力ファイルの日付をチェックしてソース・コードが変更されていないかを調べ、出力ファイルを再度作成する必要があるかどうかを判断します。詳細は Microsoft Visual C++ のドキュメントを参照してください。

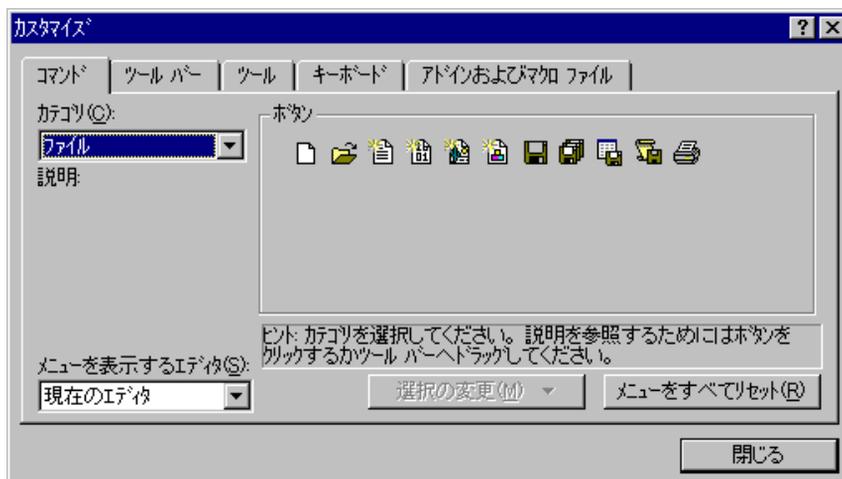
4. 「OK」をクリックします。

「ツール」メニューへの Pro*C/C++ の追加

Microsoft Visual C++ 6.0 の「ツール」メニューに選択項目として「Pro*C/C++」を追加できます。

「ツール」メニューに「Pro*C/C++」を追加する手順は次のとおりです。

1. Microsoft Visual C++ の中で、「ツール」メニューから「カスタマイズ」を選択します。「カスタマイズ」ダイアログ・ボックスが表示されます。



2. 「ツール」タブをクリックします。
3. 「メニュー項目」フィールドの一番下までスクロールして点線の長方形をクリックします。
4. 次のテキストを入力します。

Pro*C/C++

- 「コマンド」フィールドにグラフィカル Pro*C/C++ 実行可能ファイルのパスとファイル名を入力するか、フィールドの右側のブラウズ・ボタンを使用してファイル名を選択します。例を次に示します。

c:\oracle\ora81\bin\procui.exe

- 「引数」フィールドに次のテキストを入力します。

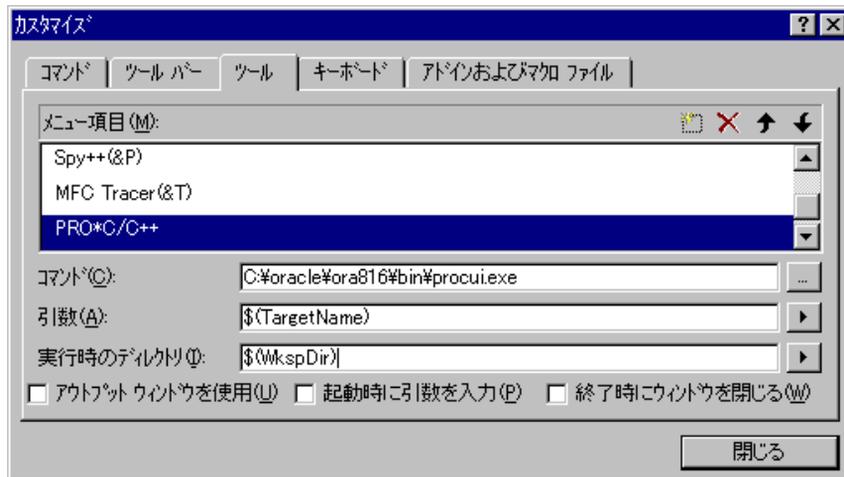
\$(TargetName)

「ツール」メニューから「Pro*C/C++ 8.1」を選択すると、Microsoft Visual C++ によって \$(TargetName) 引数が使用されて現行の開発プロジェクトの名前が Pro*C/C++ に渡されます。これにより Pro*C/C++ は開いているプロジェクトと同じ名前で拡張子が .PRE のプリコンパイル・プロジェクト（プロジェクト・ディレクトリにある）を開きます。

- 「実行時のディレクトリ」フィールドに次のテキストを入力します。

\$(WkspDir)

この時点で「カスタマイズ」ダイアログ・ボックスの内容は次の図のようになります（ただし使用するコンピュータによっては Oracle ホーム・ディレクトリが異なる場合があります）。



- 「閉じる」をクリックします。Microsoft Visual C++ の「ツール」メニューに「Pro*C/C++」が追加されます。

索引

数字

16 ビット・コードの非サポート, 1-2

A

add_newl.bat, 2-12
ANSI 準拠, 1-2
ANSI 動的 SQL, 3-4

C

CODE オプション, 2-15

D

DBMS オプション, 2-15
.DSP ファイル, 3-7

I

INCLUDE オプション, 2-15

L

LOB, 3-5

M

Microsoft Visual C++
Pro*C/C++ の統合, A-1 ~ A-7
mod_incl.bat, 2-12
msvcrt.lib ランタイム・ライブラリ, 2-14

N

Net8, 1-2

O

Object Type Translator (OTT), 3-7
oraca.h ヘッダー・ファイル, 2-13
Oracle XA, 2-15
Oracle XA ライブラリ
補足ドキュメント, 2-18
Oracle ベース
説明, ix
Oracle ホーム
説明, ix
orasql8.lib ライブラリ・ファイル, 2-14
OTT (Object Type Translator), 3-7

P

PARSE オプション, 2-15
PCC-S-02014 エラー, 2-12
pcscfg.cfg 構成ファイル, 2-15
.pre ファイル, 2-5
パスのチェック, 3-8
Pro*C/C++
Microsoft Visual C++ への統合, A-1 ~ A-7
概要, 1-1, 1-2
起動, 2-2
機能, 1-2
グラフィカル・ユーザー・インタフェース,
2-2 ~ 2-5
構成ファイル, 2-15
コマンド行インタフェース, 2-13
ライブラリ・ファイル, A-4

リンク, 2-14

S

sql2oci.h ヘッダー・ファイル, 2-13
sqlapr.h ヘッダー・ファイル, 2-13
sqlca.h ヘッダー・ファイル, 2-13
sqlcpr.h ヘッダー・ファイル, 2-13
sqlda.h ヘッダー・ファイル, 2-13
sqlkpr.h ヘッダー・ファイル, 2-13
sqllib80.lib, A-5
sqlproto.h ヘッダー・ファイル, 2-13
SQLStmtGetText() 関数, 3-7
sqlvcp() 関数, 3-7
SQL (Structured Query Language), 1-2
Structured Query Language (SQL), 1-2

う

埋込み SQL, 3-5

お

オブジェクト, 3-5
「オプション」ダイアログ・ボックス, 2-8

き

起動、Pro*C/C++, 2-2
共通ドキュメントの参照先
 demo ディレクトリ, 1-3
 Oracle XA, 2-15
 オプションのデフォルト値, 2-14
 構成ファイルの位置, 2-15
 ヘッダー・ファイルの位置, 2-13
 リンク, 2-14

く

グラフィカル・ユーザー・インタフェース, 2-2 ~ 2-5

こ

構成ファイル, 2-15
コマンド行からのプリコンパイル, 2-13

さ

「作業環境」メニュー, 2-3, 2-6

削除、ファイル, 2-8

サンプル表

 作成, 3-7

サンプル・プログラム

 ANSIDYN1, 3-2, 3-4

 ANSIDYN2, 3-2, 3-4

 COLDEMO1, 3-2, 3-4

 CPPDEMO1, 3-2, 3-4

 CPPDEMO2, 3-2, 3-4

 CPPDEMO3, 3-2, 3-4

 CV_DEMO, 3-2, 3-4

 EMPCLASS, 3-2, 3-4

 INCLUDE パス, 2-15

 LOBDEMO1, 3-3, 3-5

 MLTTHRD1, 3-3, 3-5

 NAVDEMO1, 3-3, 3-5

 OBJDEMO1, 3-3, 3-5

 ORACA, 3-3, 3-5

 PLSSAM, 3-3, 3-5

 .pre ファイルのパスの設定, 3-8

 SAMPLE, 3-3, 3-5

 SAMPLE1, 3-3, 3-5

 SAMPLE10, 3-3, 3-6

 SAMPLE11, 3-3, 3-6

 SAMPLE12, 3-3, 3-6

 SAMPLE2, 3-3, 3-6

 SAMPLE3, 3-3, 3-6

 SAMPLE4, 3-3, 3-6

 SAMPLE5, 3-3, 3-6

 SAMPLE6, 3-3, 3-6

 SAMPLE7, 3-3, 3-6

 SAMPLE8, 3-3, 3-6

 SAMPLE9, 3-3, 3-6

 SQLVCP, 3-4, 3-7

 WINSAM, 3-4, 3-7

 位置, 1-2, 3-2

 作成, 3-7

 説明, 3-4 ~ 3-7

 デフォルト・ドライブ, 3-8

 パスの設定, 3-8

し

出力ファイル名, 2-6

「新規」 ツールバー・ボタン, 2-5

す

ステータス・バー, 2-5
スレッドの定義, 2-14

せ

「接続」 ダイアログ・ボックス, 2-11
接続文字列, 2-10

た

タイトル・バー, 2-2

つ

追加、ファイル, 2-7
ツールバー・ボタン
「新規」, 2-5
「開く」, 2-5

て

ディレクトリ構造, 1-2
データベース接続文字列, 2-10
デフォルト出力
「C++ ファイル名」 コマンド, 2-6
「C ファイル名」 コマンド, 2-6

と

動的 SQL
方法 1, 3-6
方法 2, 3-6
方法 3, 3-4, 3-6
方法 4, 3-6
動的リンク・ライブラリ (DLL), 2-14
ドキュメント
共通, xi
トランザクション・プロセッシング・モニター
補足ドキュメント, 2-18

に

「入力ファイル」 ダイアログ・ボックス, 2-7

は

パス
.pre ファイルのチェック, 3-8
チェック, 3-8

ひ

「開く」 ツールバー・ボタン, 2-5

ふ

「ファイル」 メニュー, 2-3
プリコンパイルの手順, 2-5 ~ 2-12
プロジェクト・ファイル, 2-5, 3-7

へ

ヘッダー・ファイル
oraca.h, 2-13
sql2oci.h, 2-13
sqlapr.h, 2-13
sqlca.h, 2-13
sqlcpr.h, 2-13
sqlda.h, 2-13
sqlkpr.h, 2-13
sqlproto.h, 2-13
位置, 2-13
「別名保存」 コマンド, 2-12
「ヘルプ」 メニュー, 2-3
「編集」 メニュー, 2-3

ま

マルチスレッド・アプリケーション, 2-14, 3-5

め

メニュー・バー, 2-3

ら

ラージ・オブジェクト, 3-5

り

リエントラント関数, 2-14

「リスト / エラー」ダイアログ・ボックス, 2-9
リンク, 2-14