

# Содержание

Благодарности . . . . .	xii
Введение . . . . .	xiii

## ЧАСТЬ I РАСШИРЕННЫЕ КОНЦЕПЦИИ, ВНУТРЕННИЕ МЕХАНИЗМЫ И КОНЦЕПЦИИ ОТЛАДКИ

<b>Глава 1. Введение в расширенные концепции, шаблоны и методики . . . . .</b>	<b>2</b>
Что такое <i>эксперт</i> ? . . . . .	3
Новые возможности . . . . .	3
Скрытие исходного кода PL/SQL . . . . .	4
Условная компиляция . . . . .	7
Асинхронная фиксация . . . . .	10
Добывание данных — использование упреждающей аналитики . . . . .	10
Сравнение строк в PL/SQL . . . . .	16
DBMS_OUTPUT.PUT_LINE . . . . .	19
Итоги . . . . .	20
<b>Глава 2. Внутренние механизмы PL/SQL . . . . .</b>	<b>21</b>
Основы архитектуры базы данных . . . . .	21
Пользовательские и серверные процессы . . . . .	22
Глобальная область процессов . . . . .	23
Экземпляр . . . . .	24
База данных . . . . .	27
Обзор архитектуры PL/SQL . . . . .	28
Сравнение раннего и позднего связывания . . . . .	28
Компилятор PL/SQL . . . . .	29
Шаги компиляции . . . . .	29
Сравнение интерпретации и собственной компиляции . . . . .	34
Виртуальная машина PL/SQL . . . . .	36
Итоги . . . . .	37
<b>Глава 3. Отладка приложений PL/SQL . . . . .</b>	<b>38</b>
Введение механизмов обработки ошибок . . . . .	39
Различие между ошибками компиляции и времени выполнения . . . . .	39
Стандартный синтаксис для управления исключительными ситуациями . . . . .	42

Обсуждение стека ошибок и демонстрация управления стеком . . . . .	47
Знакомство со стеком ошибок и демонстрация управления им . . . . .	49
Знакомство с новыми возможностями форматирования стека ошибок и их демонстрация . . . . .	55
Управление ошибками в триггерах баз данных . . . . .	60
Итоги . . . . .	60

## ЧАСТЬ II

### ПРАВА ВЫЗЫВАЮЩЕГО, БИБЛИОТЕКИ JAVA И ОБЪЕКТНЫЕ ШАБЛОНЫ

<b>Глава 4. Архитектура с правами вызывающего</b> . . . . .	62
Знакомство с концепциями прав определяющего и вызывающего . . . . .	63
Концепция прав определяющего . . . . .	64
Концепции прав вызывающего . . . . .	71
Понимание архитектур . . . . .	75
Понимание архитектуры прав определяющего . . . . .	76
Понимание архитектуры прав вызывающего . . . . .	84
Сравнение и противопоставление стратегий реализации . . . . .	87
Права определяющего . . . . .	87
Права вызывающего . . . . .	88
Итоги . . . . .	88
<b>Глава 5. Расширение PL/SQL с помощью библиотек Java</b> . . . . .	89
Архитектура Java в Oracle . . . . .	90
Типы подключений JDBC Oracle . . . . .	94
Драйвер на клиентской стороне, или тонкий драйвер JDBC . . . . .	94
Драйвер интерфейса вызовов Oracle или толстый драйвер промежуточного уровня . . . . .	95
Внутренний серверный драйвер Oracle, или толстый драйвер на уровне сервера . . . . .	96
Построение в Oracle библиотек классов Java . . . . .	96
Построение функций внутреннего сервера Java . . . . .	99
Построение процедур внутреннего сервера Java . . . . .	103
Построение внутренних объектов сервера Java . . . . .	107
Поиск неисправностей при построении, загрузке/удалении и применении . . . . .	116
Отображение типов Oracle . . . . .	121
Итоги . . . . .	123
<b>Глава 6. Реализация объектов PL/SQL</b> . . . . .	124
Знакомство с объектами и объектными типами . . . . .	125
Что такое объект Oracle? . . . . .	125
Приписывание значений объектным типам . . . . .	134
Расширение объектных типов PL/SQL до составных объектных типов . . . . .	137
Использование объектных типов PL/SQL как фасада подсистемы . . . . .	146
Итоги . . . . .	170

### ЧАСТЬ III ОПТИМИЗАЦИЯ РЕШЕНИЙ PL/SQL

<b>Глава 7. Организация поточной обработки выполнения PL/SQL</b>	172
Знакомство с концепциями и компонентами параллельного PL/SQL	173
Знакомство с пакетом DBMS_JOB и его демонстрация	174
Знакомство, сравнение и противопоставление пакетов DBMS_JOB и DBMS_ALERT	180
Знакомство с концепциями организации поточной обработки для параллельных программ и их обсуждение	182
Демонстрация построения параллельных приложений на PL/SQL	184
Демонстрация потока управления	190
Демонстрация потоков выполнения	220
Демонстрация параллельного выполнения	224
Итоги	224
<b>Глава 8. Высокопроизводительный PL/SQL</b>	225
Оптимизация PL/SQL в Oracle	227
Обзор настройки SQL	228
Трассировка и TKPROF	229
Индексы	231
СВО и статистика	233
DBMS_STATS	236
Отыскание проблем с производительностью PL/SQL	236
Предупреждения компилятора	237
Профайлер PL/SQL	242
Собственная компиляция	248
Стратегия настройки производительности	249
Шаги для SQL	250
Шаги PL/SQL	251
Итоги	252
Содержание	

### ЧАСТЬ IV УПРАВЛЕНИЕ ТЕКСТАМИ С ИСПОЛЬЗОВАНИЕМ PL/SQL

<b>Глава 9. Введение в информационный поиск</b>	254
Обзор информационного поиска	255
Модели IR	256
Обработка текстов	256
Запросы	258
Введение в Oracle Text	261
Oracle Text и IR	262
Обзор возможностей	262
Создание индекса	263
Обработка запросов	275

Построение поискового приложения с использованием PL/SQL . . . . .	280
Поиск и выборка информации в PL/SQL . . . . .	280
Поиск PSP . . . . .	288
Итоги . . . . .	297
<b>Глава 10. Введение в регулярные выражения . . . . .</b>	<b>298</b>
Фундамент регулярных выражений . . . . .	298
История . . . . .	299
Oracle и регулярные выражения . . . . .	300
Обзор возможностей . . . . .	300
Метасимволы . . . . .	300
Операторы и функции regex . . . . .	301
REGEXP_SUBSTR . . . . .	303
PL/SQL и регулярные выражения . . . . .	310
Общепотребительные применения . . . . .	311
Средства разработки . . . . .	316
Итоги . . . . .	321
<b>ЧАСТЬ V</b>	
<b>PL/SQL SERVER PAGES И УПРАВЛЕНИЕ БАЗОЙ ДАННЫХ</b>	
<b>Глава 11. Усиление использования инструментальных средств PL/SQL . . . . .</b>	<b>323</b>
Критерии выбора между PL/SQL Toolkit и PSP . . . . .	325
Описание и конфигурирование автономного сервера HTTP . . . . .	326
Описание и конфигурирование автономного сервера HTTP Oracle . . . . .	328
Описание и конфигурирование картриджа mod_plsql . . . . .	328
Конфигурирование сервера HTTP Oracle . . . . .	330
Конфигурирование сервера HTTP Oracle 9i . . . . .	331
Конфигурирование сервера HTTP Oracle 10gR1 . . . . .	331
Построение хранимых процедур PL/SQL Toolkit и получение доступа к ним . . . . .	332
Разработка и выполнение процедур без формальных параметров . . . . .	333
Разработка процедур с формальными параметрами . . . . .	336
Получение представления о преимуществах и ограничениях . . . . .	342
Построение PL/SQL Server Pages и получение доступа к ним . . . . .	342
Разработка и выполнение процедур PSP без формальных параметров . . . . .	345
Разработка процедур PSP с формальными параметрами . . . . .	346
Получение представления о преимуществах и ограничениях . . . . .	352
Итоги . . . . .	353
<b>Глава 12. Управление базой данных . . . . .</b>	<b>354</b>
Производительность базы данных . . . . .	354
DBMS_STATS: Статистика для таблиц и индексов . . . . .	354
Перенос файлов . . . . .	357
DBMS_DATAPUMP: Импорт и экспорт данных . . . . .	358
DBMS_FILE_TRANSFER: Перемещение двоичных файлов . . . . .	367
UTL_COMPRESS: Сжатие файлов базы данных . . . . .	368

Защита данных . . . . .	372
Etarvqitcrja . . . . .	372
Инструмент шифрования DBMS_CRYPT0 . . . . .	375
Итоги . . . . .	378

## ЧАСТЬ VI ПРИЛОЖЕНИЯ

<b>Приложение А. Метасимволы и функции регулярных выражений . . . . .</b>	<b>380</b>
Скобочные выражения POSIX. . . . .	380
Метасимволы POSIX . . . . .	381
Метасимволы, не являющиеся метасимволами POSIX . . . . .	384
Функции . . . . .	386
<b>Приложение В. Пакеты, поставляемые с Oracle Text . . . . .</b>	<b>387</b>
Создание пакета . . . . .	387
Пакеты . . . . .	388
CTX_ADM . . . . .	388
CTX_CLS . . . . .	389
CTX_DDL . . . . .	389
CTX_DOC . . . . .	391
CTX_OUTPUT . . . . .	393
CTX_QUERY . . . . .	394
CTX_REPORT . . . . .	395
CTX_THES . . . . .	396
Наиболее часто используемые пакеты . . . . .	398
Примеры применения CTX_DDL . . . . .	398
Примеры CTX_REPORT . . . . .	400
<b>Приложение С. Пакеты инструментальных средств PL/SQL Web Toolkit . . . . .</b>	<b>402</b>
Пакеты . . . . .	402
HTF . . . . .	402
HTP . . . . .	403
OWA_CACHE . . . . .	405
OWA_COOKIE . . . . .	405
OWA_CUSTOM . . . . .	406
OWA_IMAGE . . . . .	406
OWA_OPT_LOCK . . . . .	406
OWA_PATTERN . . . . .	407
OWA_SEC . . . . .	407
OWA_TEXT . . . . .	407
OWA_UTIL . . . . .	407
WPG_DOCLOAD . . . . .	408
<b>Приложение D. Пособие для начинающих по Java и интерфейсу JDBC . . . . .</b>	<b>409</b>
Архитектура Java и JDBC . . . . .	409
Конфигурирование среды Java Oracle . . . . .	410
Тестирование клиентского (или с тонким драйвером) соединения JDBC . . . . .	412

# Благодарности

Огромное спасибо Лизе МакКлейн и Алексу МакДональду из издательства Osborne MacGraw-Hill за их неустанный труд над этим проектом. Клер Сплэн и Майк МакГи, очень эффективно помогавшие перевести наш текст из стадии редактирования в промышленную стадию, помогли нам не сбиться с пути. Большое вам спасибо за чуткое руководство. Черил Риникер, одна из наиболее ценных сотрудников службы Oracle Worldwide Support, обеспечила неоценимую обратную связь при техническом редактировании и рецензировании текста.

Благодарим Омара Алонсо, Мохаммеда Фейзала, Гаррета Каминага и остальных членов команды разработчиков Oracle Text за их чуткую реакцию. Отдельного упоминания должна быть удостоена Барбара Бёмер, которая любезно разрешила воспользоваться своей функцией Levenshtein Distance для демонстрации новых возможностей Oracle Database 11g Release 2. Спасибо департаменту информационных систем университета Брайхем-Янг и службе ИТ, а также администрации Academy School District 20 за их поддержку при создании этого текста. И, наконец, список благодарностей был бы неполон, если бы мы забыли упомянуть коллектив производственного отдела. Спасибо вам за то, что вы смогли объединить все фрагменты воедино, и за весь ваш упорный труд.

# Введение

В этой книге рассматриваются продвинутое вопросы, относящиеся к PL/SQL, в том числе многие примеры, которые могут быть непосредственно использованы в ваших разработках. Все предлагаемые примеры доступны в онлайн-режиме на одном из сайтов [www.OraclePress.com](http://www.OraclePress.com) или [www.PLSQLBook.com](http://www.PLSQLBook.com). Просто загрузите примеры на свой компьютер и затем выполняйте их в соответствии с предложенными инструкциями.

Текст книги разбит на шесть частей:

- Часть 1: Расширенные концепции, внутренние механизмы и концепции отладки
- Часть 2: Права вызывающего, библиотеки Java и объектные шаблоны
- Часть 3: Оптимизация решений PL/SQL
- Часть 4: Управление текстами с использованием PL/SQL
- Часть 5: PL/SQL Server Pages и управление базой данных
- Часть 6: Приложения

На протяжении всей книги можно будет увидеть разделы, озаглавленные “Рекомендации” и “Как это работает?”. Это те самые вопросы, которые постоянно возникают при чтении многих книг, но которые, как правило, остаются без ответа. В разделах “Почему я должен использовать это?” мы пытаемся предложить практическое применение обсуждаемых характеристик. Если применение опций очевидно, мы не включаем этот раздел. Но там, где их применение не столь явно, мы добавляем его, чтобы высказать некоторые идеи о том, как можно воспользоваться этими возможностями.

Разделы “Как это работает?” включены по той причине, что почти каждый известный нам программист желает знать о некоторых функциях немного больше того, что ему необходимо на практике. Пожалуйста, не стесняйтесь связаться с нами по электронной почте (наш электронный адрес [Feedback@PLSQLBook.com](mailto:Feedback@PLSQLBook.com)), если интересуетесь внутренними механизмами работы прочих опций. Мы рассмотрим вопрос о включении этих механизмов в последующие выпуски книги.

## Чем не является эта книга?

Сразу предупреждаем: эта книга не является пособием для начинающих. В ней не рассматриваются такие вопросы, как циклы FOR, курсоры, типы данных, а также структура процедур, функций или пакетов. Если читателю необходимо освещение вопросов подобного рода, рекомендуем сначала познакомиться с нашей книгой Oracle Database 10g PL/SQL Programming (“Oracle 10g: Программирование на языке PL/SQL”, издательство “Лори”, 2006), а затем вернуться к данной книге. В книге “Oracle 10g. Программирование на языке PL/SQL” освещаются вопросы различных уровней: от начального до весьма продвинутых, и она хорошо подготавливает читателя к темам, которые можно найти в настоящей книге.

## Соглашения

- Примеры программ набраны шрифтом COURIER.
- Ссылки на объекты базы данных или фрагменты программ также набраны шрифтом COURIER.
- Интересующие нас элементы набраны полужирным шрифтом (BOLD) или курсивом (ITALICIZED).
- Разделы “Рекомендации” и “Как это работает?” помещаются в отдельные текстовые окна.

## Примеры

Каждая глава сконструирована так, чтобы быть независимой от других глав. В результате примеры для каждой из глав полностью автономны. Если требуется выполнить примеры для главы 10, загрузите в компьютер ZIP-файл с сайта [www.OraclePress.com](http://www.OraclePress.com) или получите программный код с сайта [www.PLSQLBook.com](http://www.PLSQLBook.com) и начните с создания пользователя. Чтобы быть уверенным, что в среде компьютера работают все примеры, выполните сценарий создания пользователя именно для этой главы. Не пытайтесь повторно использовать схему, созданную ранее для какой-то другой главы. Ведь полномочия пользователя изменяются от сценария к сценарию. Кроме того, если есть файл README, ознакомьтесь с ним, прежде чем начнете выполнять любой из сценариев.

Если при выполнении какого-либо из сценариев возникнут проблемы, пожалуйста, сообщите о них, отправив нам письмо по адресу [Feedback@PLSQLBook.com](mailto:Feedback@PLSQLBook.com). Мы сделаем все, чтобы как можно быстрее разобраться с проблемой. Если возникла ошибка, пожалуйста, пришлите вместе с письмом журнальный файл с сообщением об ошибке, а также следующие сведения о своей операционной среде:

- Версия базы данных
- Операционная система
- Глава
- Версия сервера приложений (если проблема связана с PSP)



# **Часть I**

**Расширенные концепции,  
внутренние механизмы  
и концепции отладки**

# ГЛАВА 1

## ВВЕДЕНИЕ В РАСШИРЕННЫЕ КОНЦЕПЦИИ, ШАБЛОНЫ И МЕТОДИКИ

### **экс-перт (ex-pert)**

*Лицо, имеющее, использующее или показывающее особые навыки или знания, полученные путем обучения или на собственном опыте.*

— Онлайн-словарь Miriam-Webster

Пять лет занятий программированием требуется большинству программистов, чтобы почувствовать, что они становятся профессионалами. Традиционные концепции программирования оказываются второй натурой, подобно вождению автомобиля или набору на клавиатуре. Это хорошо известно и нанимателям, которые часто в качестве одного из условий приема на работу требуют наличия не менее чем *пятилетнего* опыта работы в программировании. Однако за это время обычно что-то происходит. Страсть к ремеслу или вспыхивает еще сильнее, направляя его обладателя к новым высотам, или угасает, в результате чего пятый год становится высшей точкой многих карьер.

То, что помогает пройти через эту связанную с пятым годом *критическую точку*, часто проявляется вместе с возобновившимся желанием учиться. Если вы читаете эту книгу, есть шансы, что у вас не пропало желание пробиться наверх и преуспеть и что вы ищете знания, которые помогут более эффективно выполнять свою работу.

Экспертный PL/SQL предназначен для тех, кто желает упрочить свои навыки и выйти за рамки традиционного программирования на PL/SQL. Возможно, вы только что переключились с другого языка программирования и хотите изучить PL/SQL глубже, чем это удастся с помощью других книг, а может быть, вы собираетесь расширить имеющиеся знания в области PL/SQL. Каждая глава этой книги непосредственно посвящена исследованию трудных вопросов, и в ней вводятся новые концепции, которые можно сразу же применить во вновь разрабатываемых проектах.

Темы данной главы:

- Что значит быть *экспертом*
- Кому следует прочесть эту книгу
- Как структурированы примеры
- Новые возможности PL/SQL в версии Oracle 10g Release 2

## Что такое *эксперт*?

Программист-эксперт — это программист, к которому обращаются в надежде получить ответы на свои вопросы. Эксперты используют все возможности для того, чтобы делать нечто большее, чем построение программы по спецификации. Они воздействуют на каждый аспект процесса разработки, помогая тем, кто еще только вступает на путь, который со временем позволит и им добиться успеха. Признание кого-то экспертом отчасти определяется навыками, а отчасти позицией.

Хотите стать экспертом? Помимо совета прочесть предлагаемую книгу, есть еще несколько советов, позволяющих расширить ваши знания:

- Помогайте другим программистам вашей компании. Предлагайте короткие курсы занятий и нацеливайте их на конкретные представляющие вам необходимыми темы.
- Ведите мониторинг форумов пользователей и отвечайте на поставленные в них вопросы. Полезно начать с форума корпорации Oracle в сети OTN (<http://otn.oracle.com/forums>). Неплохо посетить форумы Metalink и форумы на сайте [www.OraFAQ.com](http://www.OraFAQ.com)., чтобы набраться опыта, пытаясь ответить на вопросы, задаваемые другими людьми, и помочь кому-то разрешить имеющиеся у них проблемы.
- Обратите внимание на проблемы, оказывающие влияние на вашу организацию, и найдите их решение. Общая проблема, стоящая практически перед любой организацией, — это перегруженность информацией. Терабайты данных — и ни малейшей возможности использовать большую часть этой информации — вот проблема, которую можно попытаться разрешить. Хранилища документов на бумажных носителях, которые могут быть архивированы в цифровом виде, — вот еще одна проблема, которую можно помочь решить (см. об этом в главе 9). Используйте эти возможности для расширения навыков и одновременного увеличения размеров бизнеса.

Вы заметили в этих советах какую-то общую тему, помимо построения собственного успеха? Все они помогают преуспеть и другим.

## Новые возможности версии

Один из способов продвижения к статусу эксперта — это всегда быть “на уровне” новых возможностей. Довольно трудно воспользоваться преимуществами усовершенствований, если не знать об их существовании.

В этой главе будут рассмотрены:

- WRAP для динамически генерируемых PL/SQL
- Условная компиляция
- Асинхронная фиксация
- Упреждающий анализ
- Использование UTL\_MATCH для различения программных кодов
- Модификации DBMS\_OUTPUT

Эти новые возможности в деталях обсуждаются в последующих разделах.

## Соккрытие исходного кода PL/SQL

В книге “Oracle 10g: Программирование на языке PL/SQL” (“Лори”, 2006) рассматривается утилита PL/SQL Wrapper, показывающая, как можно “закрыть” исходный текст, конвертируя его в шестнадцатеричные числа. Утилита, найти которую можно в каталоге \$ORACLE\_HOME/bin, называется WRAP. Недостаток при использовании утилиты командной строки состоит в том, что нельзя скрыть динамически сгенерированные PL/SQL.

В Oracle 10g Release 2 к поставляемому пакету DBMS\_DDL добавляется перегруженная функция и процедура. Их описание приводится в таблице 1.1.

Для иллюстрации их использования создается таблица, в которой содержатся коды аэропортов, регионы и стран:

- ```

❑ -- Доступно в онлайн-режиме как часть сценария wrap.sql
CREATE TABLE airport_list (
    airport_id NUMBER(10) PRIMARY KEY,
    airport_code VARCHAR2(10 CHAR) NOT NULL,
    region VARCHAR2(30 CHAR) NOT NULL,
    country VARCHAR2(30 CHAR) NOT NULL);

```

Ниже приводится отредактированный список данных:

- ```

❑ -- Доступно в онлайн-режиме как часть сценария wrap.sql
INSERT INTO airport_list (airport_id, airport_code, region, country)
VALUES (1, 'AKL', 'Auckland', 'NEW ZEALAND');
INSERT INTO airport_list (airport_id, airport_code, region, country)
VALUES (2, 'BHE', 'Blenheim', 'NEW ZEALAND');
INSERT INTO airport_list (airport_id, airport_code, region, country)
VALUES (3, 'CHC', 'Christchurch', 'NEW ZEALAND');
INSERT INTO airport_list (airport_id, airport_code, region, country)
VALUES (4, 'CHT', 'Chatham Islands', 'NEW ZEALAND');
INSERT INTO airport_list (airport_id, airport_code, region, country)
VALUES (5, 'DUD', 'Dunedin', 'NEW ZEALAND');
...

```

Таблица 1.1. WRAP

Процедура/функция	Описание
WRAP	Перегруженная функция, возвращающая упакованный исходный код PL/SQL вместо полученного на входе исходного кода.
CREATE_WRAPPED	Процедура, упаковывающая исходный код, предлагаемый на входе. Это быстрее, чем использовать WRAP.

## WRAP

Перегруженная функция WRAP используется с оператором EXECUTE IMMEDIATE для создания упакованного кода:

```

❑ -- Доступно в онлайнном режиме как часть сценария wrap.sql
DECLARE
    v_procedure VARCHAR2 (32767);
BEGIN
    v_procedure := 'CREATE OR REPLACE PROCEDURE wrap_test '
    || 'IS '
    || '    v_airport_codes AIRPORT_LIST.AIRPORT_CODE%TYPE; '
    || '    '
    || '    CURSOR airport_cur IS '
    || '        SELECT airport_code'
    || '        FROM airport_list'
    || '        ORDER_BY airport_code; '
    || '    '
    || 'BEGIN '
    || '    FOR y IN airport_cur LOOP '
    || '        DBMS_OUTPUT.PUT_LINE( ''Airport Code: ''||y.airport_code); '
    || '    END LOOP; '
    || 'END;';

    EXECUTE IMMEDIATE DBMS_DDL.WRAP(v_procedure);
END;
/

```

Чтобы увидеть упакованную процедуру, следует выбрать текст из представления USER\_SOURCE.

```

❑ -- Доступно в онлайнном режиме как часть сценария wrap.sql
SELECT text
FROM user_source
WHERE name = 'WRAP_TEST';

```

В результате будет показана упакованная процедура, как отображено ниже:

```

❑ TEXT

PROCEDURE wrap_test wrapped
a000000

```

```

369
abcd
abcd
abcd
abcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
adcd
7
126 103
7ocFripnJzPMGnie8IG1P3gyt0Ywg1z0r54VfHRAEI1xkSMxkhbTrPGA+fgyhXm0wh9KP3mV6
ue7N1Bu6yshPJJosBnUh7N3nK0Jx09LhPITiEuVW2uCh7HQjVFwL7Ym6Hhaza+wwUDcACKoq
2WxWnMY5Dd2ncXZTZQ2Y5D3K34VpxqRjDtQTzc18LG3Iwc2VQIViNCa6TxbjmQmL/zGkQRj8
AM+S7p1FqrzRaV8LaHPHQ+V/0c1xSj8pR1dZi6z

```

## CREATE\_WRAPPED

Аналогичным образом работает и процедура DBMS\_DDL.CREATE\_WRAPPED. В приведенном ниже примере показывается, чем она отличается от функции WRAP.

```

❑ -- Доступно в онлайн-режиме как часть сценария wrap.sql
DECLARE
    v_procedure VARCHAR2(32767);
BEGIN
    v_procedure := 'CREATE OR REPLACE PROCEDURE create_wrapped_test '
        || 'IS '
        || '    v_airport_codes AIRPORT_LIST.AIRPORT_CODE%TYPE; '
        || ' '
        || '    CURSOR airport_cur IS '
        || '        SELECT airport_code'
        || '            FROM airport_list'
        || '            ORDER BY airport-code;'
        || ' '
        || 'BEGIN '
        || '    FOR y IN airport_cur LOOP '
        || '        DBMS_OUTPUT.PUT_LINE(''Airport Code: '' || y.airport_code);'
        || '    END LOOP;'
        || 'END;';
    SYS.DBMS_DDL.CREATE_WRAPPED(v_procedure);
END;
/

```

Не рекомендуется использовать оператор `EXECUTE IMMEDIATE`. Представление `USER_SOURCE` снова покажет упакованный исходный код.

❑ TEXT

```
-----
PROCEDURE create_wrapped_test wrapped
a000000
369
abcd
abcd
...
```

Выполните упакованную процедуру, чтобы проверить, что все работает, как и ожидалось:

- ❑ -- Доступно в онлайнном режиме как часть сценария `wrap.sql`  
`EXEC create_wrapped_test`

При этом будут возвращены следующие результаты:

- ❑ Airport Code: AKL  
 Airport Code: BHE  
 Airport Code: CHC  
 Airport Code: CHT  
 Airport Code: DUD  
 Airport Code: GIS  
 ...

Упаковка исходного кода PL/SQL с использованием этого метода не слишком отличается от двоичного использования `WRAP`, однако, она является более гибкой.

## Условная компиляция

Условная компиляция позволяет записывать программные коды, содержащие зависящие от версии возможности. В результате оказывается, что в получаемый после обработки код включены только программные секции, относящиеся к той конкретной версии базы данных, для которой выполняется PL/SQL. При условной компиляции используется оператор `DBMS_VERSION` и следующие символы управления препроцессором:

- \$ERROR
- \$IF
- \$THEN
- \$ELSIF
- \$ELSE
- \$END

## Рекомендации

Теперь новые возможности версии 10g Release 2 можно включить в проект приложения, не дожидаясь того момента, когда будет прекращена поддержка более старых версий сервера! Хотя и раньше можно было сделать нечто похожее с помощью операторов IF-THEN, проверяющих версию базы данных, соответствующие операторы IF-THEN всегда включались в исполняемый код. Теперь же откомпилированная версия содержит только программный код, соответствующий версии, в среде которой он будет выполняться.

Лучше всего пояснить эту возможность с помощью примера. В приведенной ниже процедуре `COMPILE_BY_VERSION` предварительно определенные управляющие символы используются для того, чтобы указать Oracle, какая часть программного кода должна быть выбрана перед компиляцией:

```

❑ -- Доступно в онлайнном режиме как часть сценария compile_by_version.sql
CREATE OR REPLACE PROCEDURE compile_by_version
IS
BEGIN
    $IF DBMS_DB_VERSION.VER_LE_10_2
    $THEN
        DBMS_OUTPUT.PUT_LINE('10.2 and under');
    $ELIF DBMS_DB_VERSION.VER_LE_10_1
    $THEN
        DBMS_OUTPUT.PUT_LINE('10.1 and under');
    $ELSE
        DBMS_OUTPUT.PUT_LINE('Not 10g');
    $END;
END;
/

```

Обратите внимание на то, что для оператора `$IF` нет соответствующего ему оператора `$END IF`. Это является очевидным отклонением от традиционного синтаксиса операторов IF-THEN-END IF. В чем дело? В записи `END IF` содержится пробел, т. е. один из тех символов, которые не должны содержаться в управляющем символе. Поэтому вместо него используется `$END`.

После создания процедуры взглянем на представление `USER_SOURCE`. Приводимый ниже оператор `SELECT` позволяет получить исходный текст процедуры:

```

❑ -- Доступно в онлайнном режиме как часть сценария compile_by_version.sql
SET PAGES 9999
SELECT TEXT
FROM USER_SOURCE
WHERE NAME = 'COMPILE_BY_VERSION';

```



При этом будет возвращен следующий результат:

```

❑ TEXT
-----
PROCEDURE compile_by_version
IS
BEGIN
  $IF DBMS_DB_VERSION.VER_LE_10_2
  $THEN
    DBMS_OUTPUT.PUT_LINE('10.2 and under');
  $ELSIF DBMS_DB_VERSION.VER_LE_10_1
  $THEN
    DBMS_OUTPUT.PUT_LINE('10.1 and under');
  $ELSE
    DBMS_OUTPUT.PUT_LINE('Not 10g');
  $END;
END;
```

Это не отличается от первоначального текста! Вместо того чтобы просматривать представление `USER | ALL | DBA_SOURCE`, используем процедуру `DBMS_PREPROCESSOR.PRINT_POST_PROCESSED_SOURCE`, чтобы увидеть влияние условной компиляции.

```

❑ -- Доступно в онлайнном режиме как часть сценария compile_by_version.sql
SET SERVEROUTPUT ON
BEGIN
  DBMS_PREPROCESSOR.PRINT_POST_PROCESSED_SOURCE (
    'PROCEDURE',
    'PLSQL',
    'COMPILE_BY_VERSION');
END;
/
```

При этом в данном случае будет возвращено:

```

❑ PROCEDURE compile_by_version
IS
BEGIN
  DBMS_OUTPUT.PUT_LINE('10.2 and under');
END;
```

---

**Примечание** Выходные данные для среды пользователя могут получиться совсем другими, если там не эксплуатируется Oracle 10g Release 2.

---

Символ `$` на самом деле является символом срабатывания (триггером) препроцессора, который указывает Oracle, что именно должно быть обработано до завершения компиляции. При этом `ERROR`, `IF`, `THEN`, `ELSIF`, `ELSE` и `END` уже определены как управляющие символы.

## Асинхронная фиксация

При фиксации (commit) транзакции Oracle записывает элемент повтора (буфер повтора) из памяти в файл журнала повтора (Redo Log File). Управление не передается обратно клиенту до тех пор, пока этот процесс не будет завершен. Асинхронная фиксация позволяет вернуть управление клиенту, не дожидаясь, пока элемент повтора будет записан на диск.

При выполнении оператора COMMIT могут быть использованы опции, приведенные в таблице 1.2.

Для того чтобы их можно было использовать в операторе COMMIT, следует либо установить параметр COMMIT\_WRITE, либо указать эту опцию при выполнении COMMIT. В следующем примере устанавливается параметр COMMIT\_WRITE:

- ❑ SQL> conn / as sysdba
- SQL> ALTER SYSTEM SET COMMIT\_WRITE = NOWAIT;

В качестве альтернативы можно указать эту опцию при задании оператора COMMIT:

- ❑ SQL> COMMIT NOWAIT;

Значениями по умолчанию являются WAIT и IMMEDIATE, так что только что показанная модификация опции COMMIT приводит к тому, что транзакция будет немедленно записана на диск, вместо того, чтобы ожидать подтверждения, прежде чем вернуть управление клиенту.

## Добывание данных — использование упреждающей аналитики

Добывание данных — это отнюдь не новый момент. Для упреждающего анализа используется пакет DBMS\_PREDICTIVE\_ANALYTICS. Пакет очень прост и содержит всего две процедуры: для проверки наличия трендов в существующих данных и предсказания будущих результатов на основании моделей, идентифицируемых при выполнении пакета.

**Таблица 1.2.** Опции оператора фиксации

Опция	Описание
BATCH	Записывает транзакцию из буфера повтора (Redo Buffer) на диск, когда это возможно, а не только при выполнении оператора фиксации (commit).
IMMEDIATE	Записывает транзакцию из буфера повтора (Redo Buffer) при выполнении оператора фиксации.
NOWAIT	Не ожидает, пока будет записан элемент повтора. Происходит немедленный возврат управления к клиенту.
WAIT	Не возвращает управления к клиенту, пока элемент повтора не будет записан на диск.

## Исследование текущих данных

Структура процедуры `EXPLAIN` приведена в таблице 1.3.

Для выполнения следует передать в параметр `DATA_TABLE_NAME` имя исходной таблицы, а в параметр `EXPLAIN_COLUMN_NAME` — имя подлежащего анализу столбца. Процедура `EXPLAIN` просматривает все столбцы исходной таблицы и анализирует их содержимое, чтобы определить, существует ли для этого столбца модель влияния. По завершении анализа в таблице результатов печатаются результаты для каждого столбца. Эта таблица не должна существовать до выполнения `DBMS_PREDICTIVE_ANALYTICS`.

---

**Примечание** Для того чтобы можно было выполнить эти примеры, необходимо установить опцию `Data Mining`.

---

Используемый в этом разделе пример начинается с таблицы, содержащей данные об оценках учащихся:

```

❑ -- Доступно в онлайнном режиме как часть сценария predictive_analytics.sql
CREATE TABLE assessment (
    student_id NUMBER(10) PRIMARY KEY,
    performance NUMBER(10) NOT NULL,
    gender VARCHAR2(10) NOT NULL,
    ethnicity NUMBER(10) NOT NULL,
    age NUMBER(10) NOT NULL);

```

**Таблица 1.3.** Параметры процедуры `EXPLAIN`

<code>DATA_TABLE_NAME</code>	Имя таблицы, которая содержит столбец, указанный в <code>EXPLAIN_COLUMN_NAME</code> .
<code>EXPLAIN_COLUMN_NAME</code>	Имя подлежащего анализу столбца.
<code>RESULT_TABLE_NAME</code>	Имя таблицы для хранения результатов выполнения процедуры. Это имя должно быть уникальным.
<code>DATA_SCHEMA_NAME</code>	По умолчанию используется имя текущей схемы. Если подвергаемые анализу таблица и столбец находятся в другой схеме, здесь необходимо указать имя схемы.

Выборочные (а не реальные!) данные включают уровни выполнения теста для учеников третьего класса с типичным дезагрегированием (детализацией) по полу (`gender`), возрасту (`age`) и этнической принадлежности (`ethnicity`). Этническая принадлежность была разбита на 5 групп, которым присвоены номера от 1 до 5. Возраст испытуемых находится в диапазоне от 8 до 10 лет.

```

❑ -- Доступно в онлайнном режиме как часть сценария predictive_analytics.sql
INSERT INTO assessment
VALUES(NULL, 3, 'M', 5, 10);
INSERT INTO assessment
VALUES(NULL, 3, 'F', 5, 9);

```

```

INSERT INTO assessment
  VALUES(NULL, 2, 'F', 5, 9);
INSERT INTO assessment
  VALUES(NULL, 4, 'M', 5, 9);
INSERT INTO assessment
  VALUES(NULL, 4, 'M', 5, 9);
INSERT INTO assessment
  VALUES(NULL, 3, 'F', 5, 8);
INSERT INTO assessment
  VALUES(NULL, 3, 'M', 5, 9);
...
COMMIT;

```

Всего в таблицу при выполнении сценария `predictive_analytics.sql` вставляется 1024 записи. Можно ожидать, что чем больше данных подвергнуто анализу, тем более точной будет корреляция. При наличии большего числа записей выбросы (резко выделяющиеся значения) будут оказывать меньшее влияние. Например, если бы в таблице оценок было всего 64 записи, существенные модели было бы очень трудно определить, но стоит расширить базовый набор данных — и соответствующие модели сразу же начинают проявляться.

Для исследования столбца `ASSESSMENT.PERFORMANCE` выполните следующие действия:

```

❑ -- Доступно в онлайн-режиме как часть сценария predictive_analytics.sql
BEGIN
  DBMS_PREDICTIVE_ANALYTICS.EXPLAIN (
    'ASSESSMENT',
    'PERFORMANCE',
    'ASSESSMENT_ANALYSIS');
END;
/

```

Процедура `DBMS_PREDICTIVE_ANALYTICS.EXPLAIN` создает таблицу для хранения выходных данных. Мы назвали ее `ASSESSMENT_ANALYSIS`, но, конечно, будет работать и любое другое допустимое имя таблицы. Вне зависимости от предложенного имени таблица имеет одинаковую структуру (см. ниже):

```

❑ Name                               Null?      Type
-----
ATTRIBUTE_NAME                       VARCHAR2 (40)
EXPLANATORY_VALUE                     NUMBER
RANK                                   NUMBER

```

В столбец `ATTRIBUTE_NAME` включено по одной записи для каждого столбца анализируемой таблицы за исключением столбца, с которыми они сравниваются. В столбце `EXPLANATORY_VALUE` содержится число в диапазоне от 0 до 1, которое ранжирует возможность этого столбца предсказать значение целевого столбца. Значение 0 означает, что корреляция не обнаружена, в то время как значение 1 указывает на полную корреляцию. Чем

больше значение EXPLANATION\_VALUE, тем в большей степени можно полагаться на значение соответствующего столбца для предсказания значения целевого столбца.

Для знакомства с результатами выберите из таблицы ASSESSMENT\_ANALYSIS все три столбца.

```
❑ -- Доступно в онлайнном режиме как часть сценария predictive_analytics.sql
SET PAGES 9999
SELECT *
  FROM ASSESSMENT_ANALYSIS;
```

Это приведет к следующим результатам:

```
❑ ATTRIBUTE_NAME          EXPLANATORY-VALUE          RANK
-----
GENDER                    ,071302739                 1
AGE                       ,059764092                 2
ETHNICITY                 ,037652922                 3
STUDENT_ID                0                           4
```

Таким образом, для этого примера набора данных достигнутые учеником результаты можно лучше всего объяснить, используя значение столбца gender, хотя он никоим образом не является совершенным индикатором. На самом деле степень объяснения достаточно низкая, на что указывает довольно малое значение столбца EXPLANATORY\_VALUE (всего .07).

**Примечание** Если повторить выполнение этого примера, не удалив предварительно таблицу ASSESSMENT\_ANALYSIS, это закончится аварийно с выдачей следующего сообщения:

```
❑ BEGIN
*
ERROR at line 1:
ORA-00955: name is already used by an existing object
ORA-06512: at "DMSYS.DBMS_PREDICTIVE_ANALYTICS", line 1100
ORA-06512: at line 2
(ОШИБКА в строке 1:
ORA-00955: имя уже используется существующим объектом
ORA-06512: в строке 1100 "DMSYS.DBMS_PREDICTIVE_ANALYTICS"
ORA-06512: в строке 2)
```

Для того чтобы избежать ошибки, следует перед повторным выполнением удалить таблицу.

## Предсказание будущих результатов

Довольно легко найти корреляции между различными типами данных, помогающие понимать результаты, но с помощью того же самого пакета можно предсказывать значения. Процедура DBMS\_PREDICTIVE\_ANALYTICS.PREDICT имеет показанные в таблице 1.4 параметры.