

Oracle9i 培训教程

日本事业部

Copyright @ 863soft, 2009.

修改历史

2009年10月08号 作成 V1.0

目 录

- ORACLE系统概述
- 数据和表的创建
- ORACLE体系结构
- SQL语言基础及应用
- 表、视图及其他数据库对象
- PL/SQL编程
- 安全性管理

Oracle系统概述

Oracle介绍

■ 什么是Oracle?

Oracle是一个数据库公司，起初提供的服务很少，或者几乎没有服务，此外没有成熟的应用软件。今天情况就相当不同了。这个数十亿美元的公司拥有大量的产品，大量的服务，很多应用软件。

■ Oracle的今天

今天，座落在加利福尼亚 Redwood Shores的Oracle公司为电子信息管理提供软件产品和服务。Oracle是一个世界范围的软件提供者，1999年的收入将超过80亿美元。Oracle的业务遍及世界90多个国家，其软件在100多种不同的计算机上运行，在信息高速公路中扮演着一个重要的角色。

Oracle9i简介

■ Oracle9i的三个版本

(1) 企业版

企业版主要用于构建安全、可靠、大容量的互联网应用和数据仓库。面向企业级应用。

(2) 标准版

标准版提供了能够通过Web浏览器进行管理的高性能数据库服务，面向部门级应用。

(3) 个人版

个人版是Oracle9i全功能单用户版本，面向开发技术人员。

Oracle9i的安装

■ 1、硬件环境需求

CPU：最低Pentium166，推荐使用Pentium233以上。

内存：最低128MB，推荐使用256MB以上。

硬盘：Oracle9i允许用户选装组件，因此不同的配置对硬盘的容量要求也不同。企业版按照典型配置安装需要3GB容量的硬盘，加上Windows2000Sever安装在同一逻辑盘上共需要4.5GB，因此建议配置8GB容量以上硬盘。

■ 2、软件环境要求

操作系统：可以在Windows2000Sever/XP或
Windows NT 4.0+Service Pack 6.

（具体安装过程参照附件一和附件二）

数据库的基础知识

■ 1、什么是数据库？

Scott Martin是Oracle公司核心开发人员之一，他对什么是数据库这一问题的回答是“数据库就是处理数据文件的一批程序。”数据库就是数据文件以及用于处理这些数据文件的程序的集合。

■ 2、数据文件

数据文件用于存放所有的数据库数据。Oracle数据库由一个或多个数据文件组成，数据文件结合在一起形成表空间。（数据库中所有的数据信息都是存放在数据文件中的。）

■ 3、用户数据和系统数据

在相应数据库的数据文件中存放有两种类型的数据：用户数据和系统数据。

表1 一般用户数据类型

数据类型	存放的有关信息
顾客信息	名字、电话号码
产品信息	产品名，是否有货，价格
医疗信息	诊断结果，医生名，护士名
资产信息	股票数量，期货数量
财务信息	股票价格，利率

用户数据是用户存放在数据库中的信息

表2 系统数据的一般类型

数据类型	存放的有关信息
表	表的字段和表中数据的类型
空间	数据库对象所占的物理空间
用户	名，口令，特权
数据文件	数目，位置，最后使用的时间

系统数据是用来管理用户数据和数据库本身的数据

■ 4、数据库有哪些特点？

- (1) 数据结构化。
- (2) 数据共享。
- (3) 减少数据冗余。
- (4) 优良的存储功能。

■ 5、数据库用户

- (1) 应用开发人员
- (2) 应用程序管理员
- (3) 数据库管理员
- (4) 用户

数据库的由来和发展

■ 1、人工管理阶段

外部存储器只有磁带、卡片和纸带等，还没有磁盘等直接存取存储设备。

特点：

- (1) 数据不保存。计算机用于计算，不需要保存数据。
- (2) 没有专门软件对数据进行管理。
- (3) 只有程序的概念，没有文件的概念。
- (4) 数据面向应用。即一组数据对应一个程序。

■ 2、文件系统阶段

计算机不仅用于科学计算，还用于信息管理。

特点：

(1) 数据可长期保存在外部存储器的磁盘上。

(2) 数据的逻辑结构与物理结构有了区别，但比较简单。程序与数据之间有“设备独立性”，即程序只需用文件名就可与数据打交道，不必关心数据的物理位置。

(3) 文件的组织已多样化。

(4) 数据不再属于某个特定的程序，可以重复使用。

■ 3、数据库阶段

特点：

(1) 数据库的并发控制：避免并发程序之间相互干扰，防止数据库被破坏。

(2) 数据库的恢复：在数据库被破坏或数据不可靠时，系统有能力把数据库恢复到最近某时刻的正确状态。

(3) 数据完整性：保证数据库始终包含正确的数据。

(4) 数据安全性：保证数据的安全，防止数据丢失或被盗窃、破坏。



■4、高级数据库技术阶段

分布式数据库系统

多数处理就地完成。

各地计算机由数据通信网络相联系。

面向对象数据库系统

面向对象数据模型完整地描述现实世界的数据结构，能表达数据间的嵌套、递归的联系。

具有面向对象技术的封装性和继承性

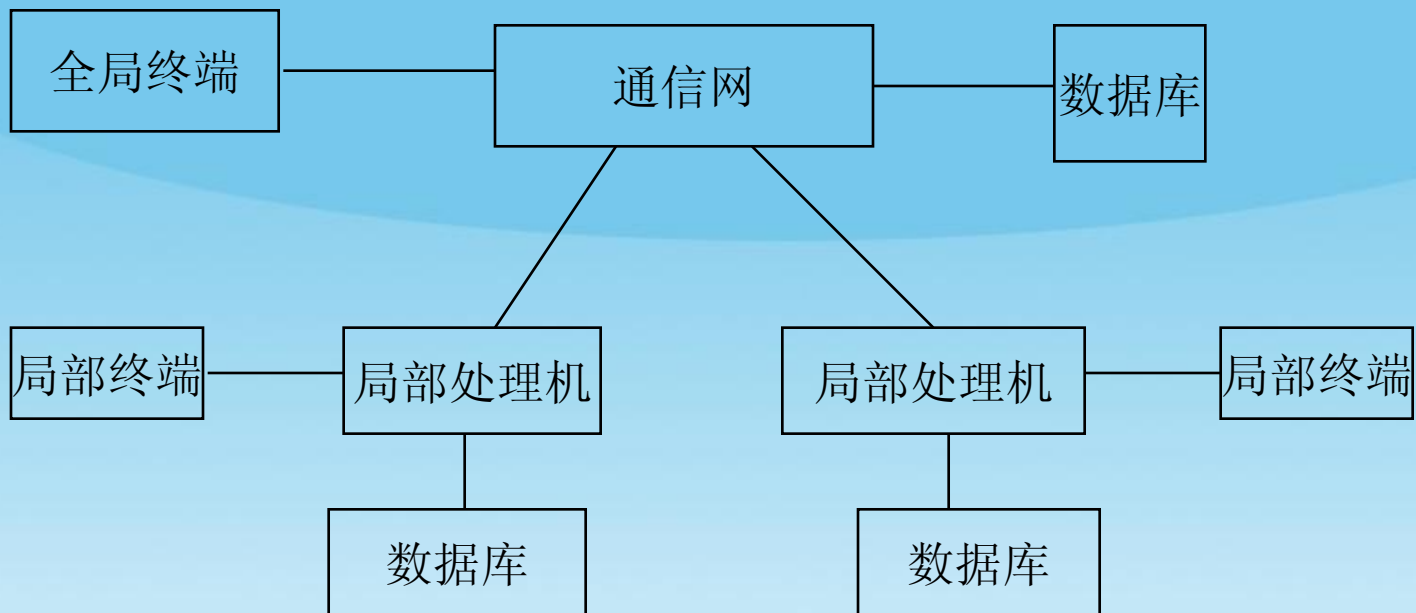
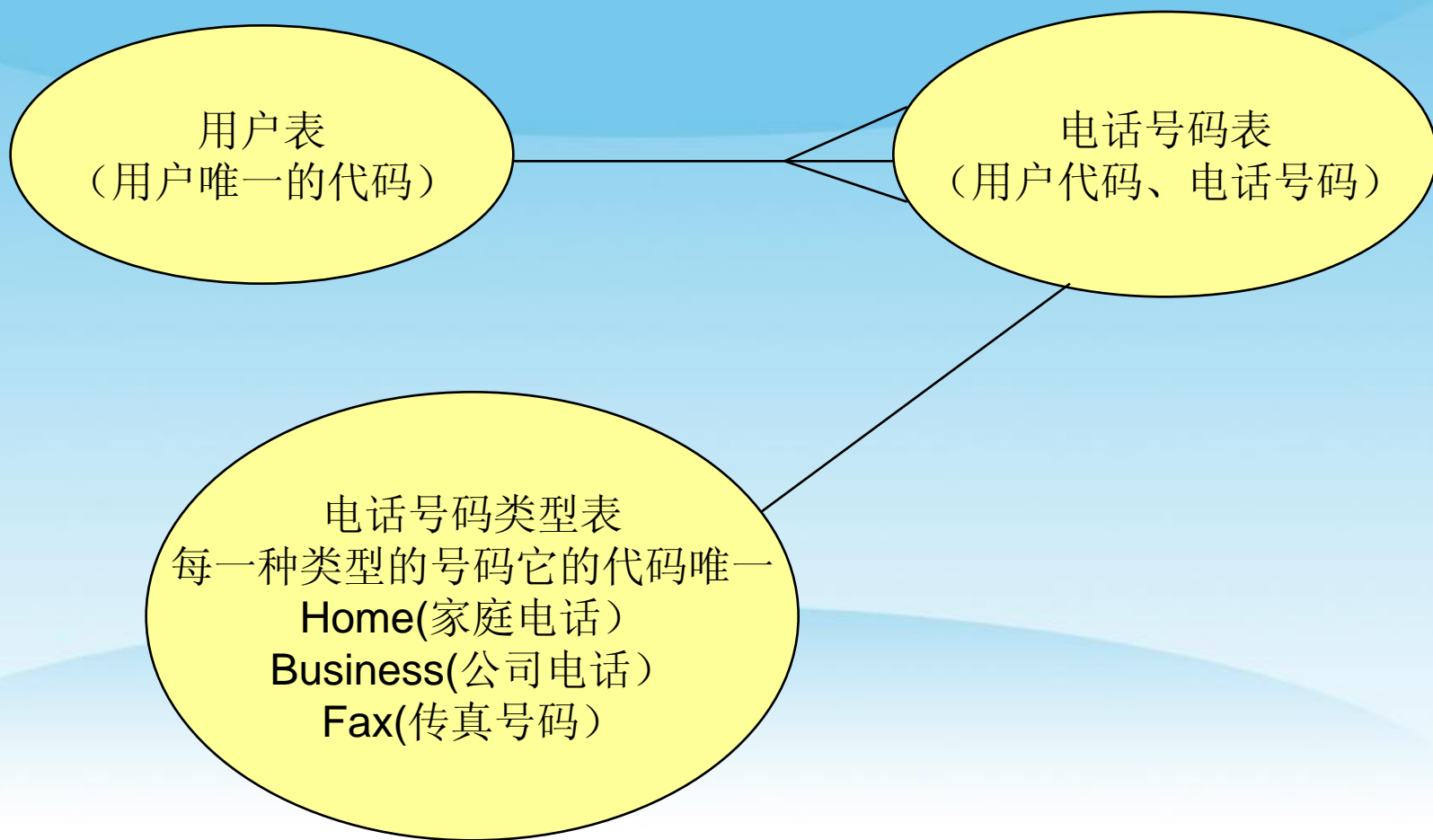


图 5 分布式数据库系统

关系数据库



每个顾客可能有一个或多个电话号码
每个电话号码属于且仅属于一个顾客
任何电话号码必须是且仅是一种类型的（即家庭电话、公司电话、传真号码等）

在关系数据库中，不存在数据更改的不一致问题，
因为所有数据存放且仅存在一个地方

表3 传统模型和关系模型间的差异

任 务	传统的方法	关系的方法
设计应用	找出什么样的应用需要什么样的信息，然后建立一系列的主文件	定义所要收集的数据类型，然后定义它们的关系
实现应用	把数据装入主文件，把每个信息项装入每个主文件的相应槽中	把数据类型装入各自的表中，确保每个项最终放入一个且仅一个位置
更改应用	重新设计数据库并修改所有反映更改的程序。	孤立定义受影响的数据类型的表表中的数据不受影响
修改数据子集	从头到尾读每个主文件。若某行是要更改的一部分，处理它，否则跳到下一个记录	孤立是子集一部分的所有的行的集合，并用一个SQL语言来实现更改

数据库和表的创建

数据库和表的创建

- 创建数据库和表是Oracle9i最基本的工作，数据库和表是Oracle9i用于组织和管理数据的对象。用户使用Oracle设计和实现信息管理系统，首先就是设计和实现数据的表示与存储，即数据库和表的创建。

基本概念

- Oracle是一种关系数据库管理系统（RDBMS）。关系数据库是按照二维表结构方式组织的数据集合，每个表体现了集合理论中定义的数学概念——关系。在创建数据库之前，理解Oracle数据库的基本概念，以确保数据库能很好地运行是很重要的。

数据库

- 数据库是一个数据容器，它包含了表、索引、视图、簇、过程、函数、包等对象，并对其进行统一的管理。用户只有和一个确定的数据库连接，才能使用和管理该数据库中的数据。
- 1、数据库的内部结构
 - (1)表空间。表空间是数据库的逻辑划分，每个数据库至少有一个表空间
 - (2)表。

表是数据库中存放用户数据的对象，它包含一组固定的列。表中的列描述该表所跟踪的实体的属性，每个列都有一个名字和若干个属性。表结构的一个样例如图

列名

WJBH	MC	ZBR_NAME
1001	鼓楼多媒体通信楼	江苏省电信公司 南京分公司
1002	摩托罗拉软件中心	江宁经济技术开 发总公司
1003	王府大厦电梯	南京市安全局

列

数据行

图2.1 表结构样例

(3) 约束条件。

可以为一个表列创建约束条件，此时，表中的每一行都必须满足约束条件定义所规定的条件。

约束条件有以下5种：

1、主键：主键是表中的一列或多个列。为表定义主键有几个作用，主键包含的列不能输入重复的值，以保证一个表的所有行的唯一性；主键包含的列不能输入重复值，以保证一个表的所有行的唯一性；主键也不允许定义此约束的列为NULL值；主键在定义此约束的列中创建了唯一性的索引，利用这个索引可更快地检索表中的行。

2 、默认约束条件：在表中插入一行数据但没有为列指定值时生成一个在定义表时预先指定的值。

3 、检查约束条件：该约束条件确保指定列中的值符合一定的条件。

4 、惟一性约束条件：用于保证不是主键那些列的惟一性。

5 、外键约束条件：该约束条件规定表间的关系性质。一个外键使一个表的一列或多列与已定义为主键的表中的一批相同的列相关联。

(4) 索引。

在关系数据库表中，一个行数据的物理位置无关紧要。为了能够找到数据，表中的每一行都用一个 RowID 来标识。RowID 告诉数据库这一行的准确位置，包括所在的文件、该文件中的块和该块中的行地址。

索引是帮助用户在表中快速地找到记录的数据库结构。它既可以提高数据库性能，又能够保证列值的唯一性。

(5) 簇 (CLUSTER)。经常被同时访问的表在物理位置上可以存储在一起。为了将他们存储在一起，就要创建一个簇来管理这些表。

(6) 用户 (USER)。用户帐号虽然不是数据库中的一个物理结构，但它与数据库中的对象有着重要的关系，这是因为用户拥有数据库的对象。例如，用户SYS拥有数据字典表，这些表中存储了数据库中其他对象的所有信息；用户 SYSTEM拥有访问数据字典表的视图。

为数据库创建对象（例如表）必须在用户帐号下进行。

表

表是用来存储和操作数据的一种逻辑结构。表由行和列组成，因此也称为二维表。

■ 1、表结构

表是在日常工作和生活中经常使用的一种表示数据及其关系的形式，例如下表就是一个学生情况表。

学生情况表

学号	姓名	专业名	性别	出生时间	总学分	备注
001101	王林	计算机	男	1980-02-10	50	
001102	程明	计算机	男	1981-02-01	50	
001103	王燕	计算机	女	1979-10-06	50	

每个表都有一个标识自己的名字

关系数据库使用表来表示实体及其联系。

(1) 表结构：每个数据库包含了若干个表。每个表包含一组固定的列，而列由数据类型和长度两部分组成

(2) 记录：每个表包含了若干行数据，它们是表的“值”，表中的一行称为一个记录，因此，表是记录的有限集合。

(3) 字段：每个记录由若干个数据项构成，构成记录的每个数据项成为字段。

(4) 关键字：关键字可以将表中的不同记录区分开。

。

■ 2关系

主键是确定表记录惟一的列集（一个列或多个列）。

每张表既相互独立，同时相互之间又存在联系。一般将有联系的表放在同一个数据库中。建立关系的目的是把独立存在的表连接起来，以获得有联系的信息。表与表之间有下列关系：

(1) 一对一关系。

有两张表，A表和B表，A表中的一条记录在B表中有一条记录与之对应。反过来，B表中的一条记录在A表中仅有一条记录与之对应。具有这种关系的两张表存在一对一的关系。

(2) 一对多关系。

有两张表，A表和B表，A表中的一条记录在B表中有多条记录与之对应。反过来，B表中的一条记录在A表中仅有一条记录与之对应。具有这种关系的两张表存在一对多的关系。

(3) 多对多关系。

有两张表，A表和B表，A表中的一条记录在B表中有多条记录与之对应。反过来，B表中一条记录在A表中也有多条记录与之对应。

■3、表示实体的表和表示实体联系的表

关系数据库用表来反映数据本身的内容以及反映数据之间的联系。所以在关系数据库中，包含了反映实体信息的表和反映实体之间联系的表。

一般来说，两个反映不同实体信息的表必须通过反映实体之间联系的表才能建立关系。

表 2.2 课程情况表

课程号	课程名	开课学期	学时	学分
101	计算机基础	1	80	5
102	程序设计语言	2	80	4
206	离散数学	4	68	4

表2.3 学生课程成绩表

学号	课程号	成绩
001101	101	80
001101	102	78
001101	206	76
001102	102	78
001102	206	78
001103	101	62
001103	102	70
001103	206	81

界面创建数据库和表

- 数据库的创建和删除

例：使用DBCA创建XSCJ数据库。

启动DBCA， DBCA激活并初始化。

的文档 Microsoft PowerPoint
的电脑 Microsoft Word

上邻居

收藏夹

Windows Catalog
Windows Update
打开 Office 文档
设定程序访问和默认值
新建 Office 文档

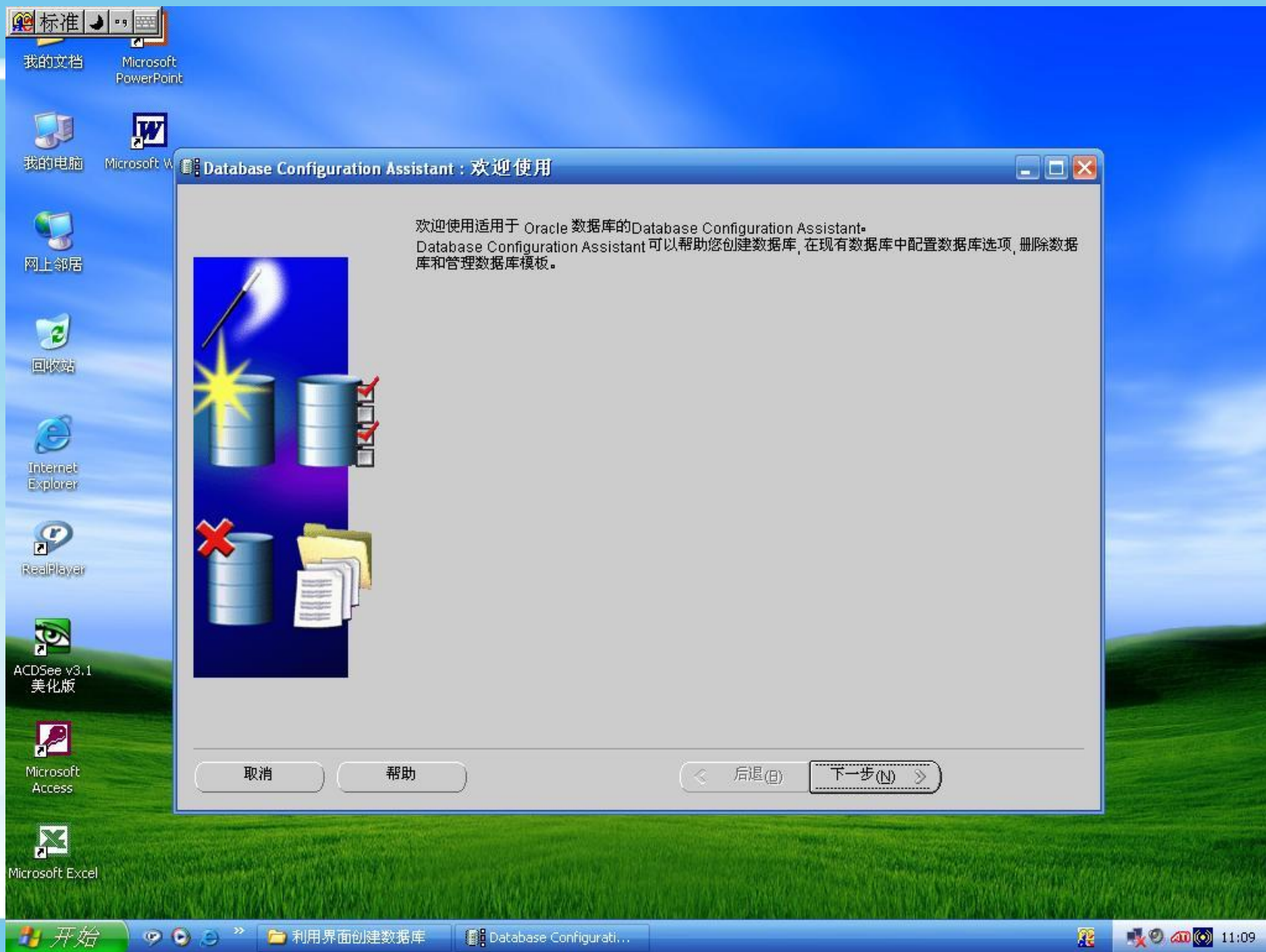
程序(P)
文档(D)
设置(S)
搜索(C)
帮助和支持(H)
运行(R)...

注销 user(L)...
关闭计算机(U)...

ACDSee v3.1美化版
FlashGet
Microsoft Office 工具
Themes
附件
启动
游戏
Internet Explorer
Microsoft Access
Microsoft Excel
Microsoft PowerPoint
Microsoft Word
MSN Messenger 6.2
Outlook Express
Windows Media Player
Windows Movie Maker
分段系列号输入工具
远程协助
My MPC 系列·暴风影音
Realtek Sound Manager
Real
RealPlayer
番茄花园 StyleXP 主题
Oracle - OraHome92
Oracle Installation Products

Application Development
Configuration and Migration Tools
Enterprise Manager Quick Tours
Integrated Management Tools
Oracle HTTP Server
Enterprise Manager Console
Release Documentation

Administration Assistant for Windows NT
Database Configuration Assistant
Database Upgrade Assistant
Enterprise Manager Configuration Assistant
Microsoft ODBC Administrator
Net Configuration Assistant
Net Manager



Database Configuration Assistant, 步骤 1(共 8 步): 操作

请选择希望执行的操作

- ☒ 创建数据库
- ☐ 在数据库中配置数据库选项
- ☐ 删除数据库
- ☐ 管理模板



取消 帮助 < 后退(B) 下一步(N) >

Database Configuration Assistant, 步骤 2(共 8 步): 数据库模板

从以下列表中选择模板来创建数据库:



选择	模板名	是否包括数据文件?
<input type="radio"/>	Data Warehouse	是
<input type="radio"/>	General Purpose	是
<input type="radio"/>	Transaction Processing	是
<input checked="" type="radio"/>	New Database	否

显示详细资料...

取消 帮助 < 后退(B) 下一步(N) >

Database Configuration Assistant, 步骤 3(共 8 步): 数据库标识

指定以下数据库信息。

Oracle9i 数据库由全局数据库名唯一标识, 典型的形式是 "name.domain".

全局数据库名:

数据库至少由一个 Oracle9i 例程引用, 此例程由 Oracle 系统标识符 (SID) 唯一标识以区别于该计算机上的任何其他例程。

SID:



取消 帮助 < 后退(B) 下一步(N) > 完成(F)

Database Configuration Assistant, 步骤 5(共 8 步): 数据库连接选项

请选择希望数据库采用的默认操作模式:

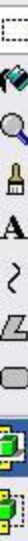


☒ 专用服务器模式
数据库将为每一个客户机连接分配专用资源。当预期客户机连接总数较小, 或客户机向数据库发出的请求持续时间较长, 请使用该模式。

☐ 共享服务器模式
多个客户机连接共享一个已分配数据库的资源池。如果需要并行连接到数据库并有效使用系统资源的用户较多, 请使用此模式。此时将启用 Oracle 共享服务器功能。

[编辑共享连接参数...](#)

[取消](#) [帮助](#) [后退\(B\)](#) [下一步\(N\)](#) [完成\(F\)](#)



我的文档

Micro Power

我的电脑

Microsoft

网上邻居



回收站



Internet Explorer



RealPlayer



ACDSee v3.1
美化版

Database Configuration Assistant, 步骤 6(共 8 步): 初始化参数

内存

字符集

数据库大小

文件位置

归档

☐ 典型

用于 Oracle 的物理内存 (511 MB) 的百分比:

70

数据库类型:

数据仓库

[显示内存的分配情况...](#)

☒ 自定义

共享池:

48

MB

缓冲区高速缓存:

24

MB

Java 池:

32

MB

大型池:

8

MB

PGA:

24

MB

用于 Oracle 的总内存: 176 MB



内存总量包括 40MB 的 Oracle 进程大小以及可能包含的空参数的默认值。

[所有初始化参数...](#)

[文件位置变量...](#)

取消

帮助

< 后退(B)

下一步(N) >

完成(F)

取消

帮助

< 后退(B)

下一步(N) >

完成(F)



Database Configuration Assistant, 步骤 6(共 8 步): 初始化参数

内存 字符集 数据库大小 文件位置 归档

请指定是否希望在归档日志模式下运行数据库。

☒ 归档日志模式

☒ 自动归档

日志归档文件名格式: %t_%s.dbf

建议将归档日志文件写到不同磁盘的多个位置。

归档日志目标位置
{ORACLE_BASE}\oradata\{DB_NAME}\archive

所有初始化参数... 文件位置变量...

取消 帮助 < 后退(B) 下一步(N) > 完成(F)

Database Configuration Assistant, 步骤 6(共 8 步): 初始化参数

内存

字符集

数据库大小

文件位置

归档



数据块是用于分配和 I/O 存储的最小单元。只能在创建数据库时指定数据库的数据块大小。

块大小:

指定排序操作过程中可以使用的最大内存。较大的排序值可以提高大型分类的效率。

排序区域大小:

[所有初始化参数...](#)[文件位置变量...](#)

取消

帮助

< 后退(B)

下一步(N) >

完成(F)



河南省 863 软件孵化器有限公司
<http://www.863soft.cn>

的文档 Microsoft PowerPoint

Microsoft Word

上邻居

网站

Internet Explorer

Player

Free v3.1 化版

Microsoft Access

Microsoft Excel

Database Co

所有初始化参数

名称	值	包含(是/否)	类别
optimizer_features_enable	9.0.1		优化程序
remote_dependencies_m...	TIMESTAMP		PL/SQL
parallel_threads_per_cpu	2		并行执行
logmnr_max_persistent_s...	1		其他
nls_date_language			NLS
workarea_size_policy	MANUAL		排序, 散列联接, 位图索引
O7_DICTIONARY_ACCES...	FALSE		安全性和审计
license_max_sessions	0		许可限制
star_transformation_enabl...	FALSE	✓	优化程序
nls_date_format			NLS
lock_sga	FALSE		SGA 内存
fixed_date			其他
remote_os_roles	FALSE		安全性和审计
nls_comp			NLS
object_cache_max_size_p...	10		对象和 LOB
shared_memory_address	0		SGA 内存
db_recycle_cache_size	0		高速缓存和 I/O
row_locking	always		ANSI 相容性
log_archive_duplex_dest			归档
sql_trace	FALSE		诊断和统计
db_block_buffers	0		高速缓存和 I/O
undo_management	AUTO	✓	系统管理的还原和回退段
oracle_trace_collection_p...	??otrace/admin/cdf		诊断和统计
fast_start_parallel_rollback	LOW		事务处理
global_names	FALSE		分布式, 复制和快照

取消

完成(F)

关闭

显示说明

帮助



Database Configuration Assistant, 步骤 7(共 8 步): 数据库存储

存储

- Controlfile
- 表空间
- 数据文件
- 回退段
- 重做日志组

数据库存储

从**数据库存储**页，可以指定用于创建数据库的存储参数。该页显示树列表和概要视图 (多列列表) 以允许您更改并查看以下对象：

- 控制文件
- 表空间
- 数据文件
- 回退段
- 重做日志组

从任何对象类型文件夹中，单击**添加**创建新对象。要删除某对象，请从对象类型文件夹中选择该对象并单击**删除**。

重要事项：如果选择了种子数据库模板，将无法添加或删除数据文件、表空间或回退段。选择种子模板后，只允许您更改以下内容：

- 数据库名称
- 数据文件的目标位置
- 控制文件或日志组。

添加

删除

文件位置变量...

取消

帮助

后退(B) <

下一步(N) >

完成(F)

取消	undo_management	NO		
	oracle_trace_collection_p...	?/otrace/admin/cdf		诊断和统计







表的创建、修改和删除

表空间就像一个文件夹，是存储数据库对象的容器。

- 1、创建和管理表空间

表空间的创建既可以在Oracle企业管理器中进行也可以使用SQL命令来创建。

- 2数据类型

表是真正存储各种各样数据的对象，由行或列组成。行有时也称为记录，列有时也称为字段或域。设计数据库时，要决定它包括哪些表，每个表中包含哪些列，每列的数据类型等。

- 在表中创建列时，必须为其指定数据类型，列的数据类型决定了数据的取值、范围和存储格式。



数据类型	描述
CHAR	固定长度字符域，最大长度可达2000个字节
NCHAR	多字节字符集的固定长度字符域，长度随字符集而定，最多为2000个字符或20000个字节
VARCHAR2	可变长度字符域，最大长度可达4000个字符
NVARCHAR2	多字节字符集的可变长度字符域，长度随字符而定，最多为40000个字符或40000个字节
DATE	用于存储全部日期的固定长度（7个字节）字符域，时间作为日期的一部分存储在其中。日期以DD-MON-YY格式表示，如13-APR-99表示1999。4。13
NUMBER	可变长度数值列，允许值为0、正数或负数。NUMBER值通常以4个字节或更少的字节存储
LONG	可变长度字符域，最大长度为2GB
RAW	表示二进制数据的可变长度字符域，最长为2000个字节
LONG RAW	表示二进制数据的可变长度字符域，最长为2G
BLOB	二进制大对象，最大长度为4GB
CLOB	字符大对象，最大长度为4GB
NCLOB	多字节字符集的CLOB数据类型，最大长度为4GB
BFILE	外部二进制文件、大小由操作系统决定
ROWID	表示ROWID的二进制数据
UROWID	用于数据寻址的二进制数据，最大长度为4000个字节

■ 3、创建表

通过Oracle企业管理器(OEM)在XSCJ数据库中创建学生情况表。学生情况表名为XS，表结构如表2.5所示。

列名	数据类型	长度	是否允许为空值	默认值	说明	列名含义
XH	CHAR	6	否	无	主键	学号
XM	CHAR	8	否	无		姓名
ZYM	CHAR	10	是	无		专业名
XB	CHAR	2	否	无		性别
CSSJ	DATE		否	无		出生时间
ZYF	NUMBER	2	是	无		总学分
BZ	VARCHAR2	200	是	无		备注

网络

数据库

STUDY - system AS SYSDBA

例程

方案

安全性

存储

控制文件

表空间

CWM Lite

DRSYS

EXAMPLE

INDX

ODM

SYSTEM

TEMP

TOOLS

UNDOTBS1

USERS

XDB

数据文件

回退段

重做日志组

归档日志

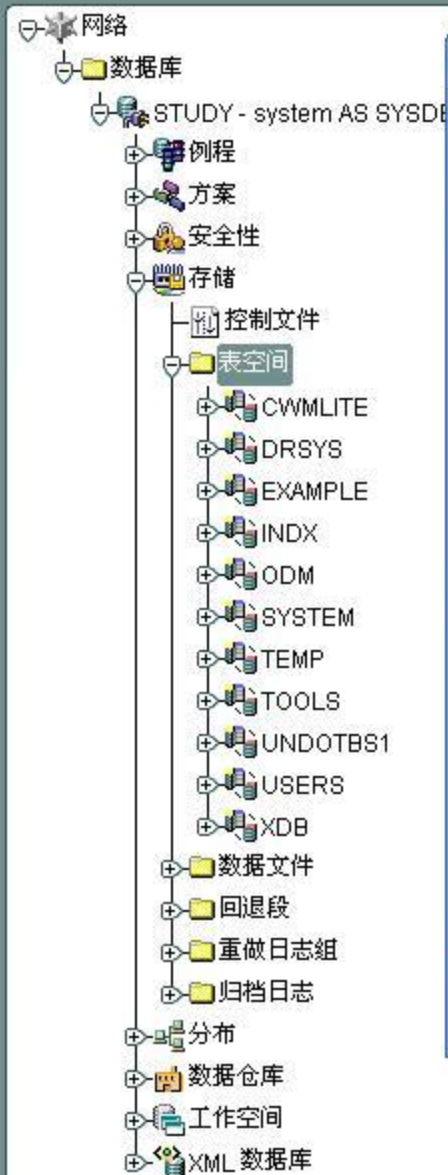
分布

数据仓库

工作空间

XML 数据库

名称	类型	区管理	大小 (M)	已使用 (M)	利用率
CWM Lite	PERMANENT	LOCAL	20.000	9.375	46.88
DRSYS	PERMANENT	LOCAL	20.000	9.688	48.44
EXAMPLE	PERMANENT	LOCAL	149.375	148.875	99.67
INDX	PERMANENT	LOCAL	25.000	0.063	0.25
ODM	PERMANENT	LOCAL	20.000	9.375	46.88
SYSTEM	PERMANENT	LOCAL	400.000	397.438	99.36
TOOLS	PERMANENT	LOCAL	10.000	6.063	60.63
UNDOTBS1	UNDO	LOCAL	200.000	11.313	5.66
USERS	PERMANENT	LOCAL	25.000	0.063	0.25
XDB	PERMANENT	LOCAL	38.125	37.938	99.51
TEMP	TEMPORARY	LOCAL	40.000	0.000	0.00



创建表空间 - system@STUDY

一般信息 存储

名称:

数据文件

文件名	文件目录	大小	
.ora	F:\ORACLE\ORAD...	5 MB	

状态

☒ 联机 ☐ 只读

☐ 脱机

类型

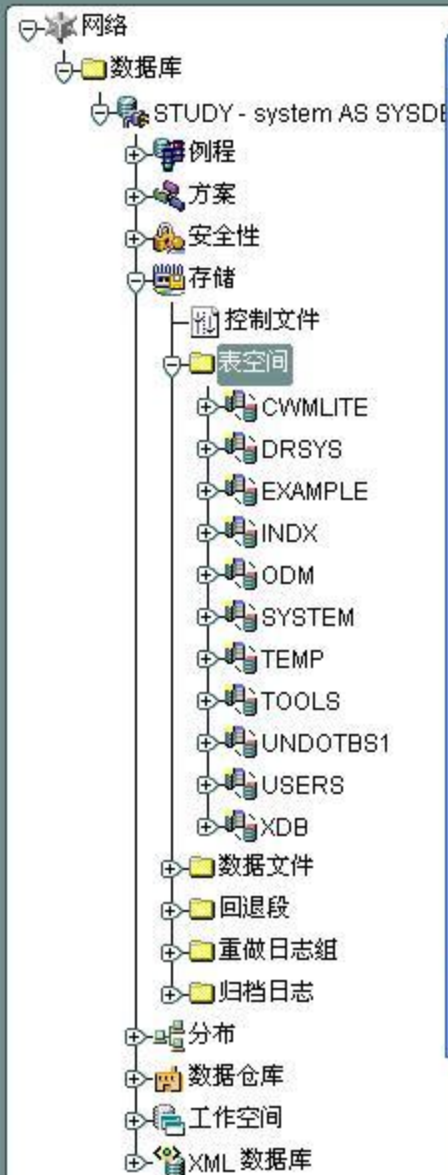
☒ 永久 ☐ 临时

☐ 设置为默认临时表空间

☐ 还原

创建(C) 取消 显示 SQL 帮助

名称	类型	区管理	大小 (M)	已使用 (M)	利用率
					46.88
					48.44
					99.67
					0.25
					46.88
					99.36
					60.63
					5.66
					0.25
					99.51
					0.00



名称	类型	区管理	大小 (M)	已使用 (M)	利用率
					46.88
					48.44
					99.67
					0.25
					46.88
					99.36
					60.63
					5.66
					0.25
					99.51
					0.00

创建 表空间 - system@STUDY

一般信息 存储

区管理: ☒ 本地管理 ☐ 在字典中管理

☒ 自动分配
☐ 统一分配

大小: KB

段空间管理

☒ 自动
表空间中的对象自动管理它们的空闲空间。它提供了高性能的空闲空间管理功能。

☐ 手动
表空间中的对象使用空闲列表来管理它们的空闲空间。这是为向后兼容而提供的功能。

启用事件记录

☒ 是 - 生成重做日志并可恢复
☐ 否 - 快速更新, 不生成重做日志且不可恢复

块大小: 字节

创建(C) 取消 显示 SQL 帮助

网络

数据库

STUDY - system AS SYSDBA

例程

方案

安全性

存储

控制文件

表空间

CWMNLITE

DRSYS

EXAMPLE

INDX

MYTH

ODM

SYSTEM

TEMP

TOOLS

UNDOTBS1

USERS

XDB

数据文件

回退段

重做日志组

归档日志

分布

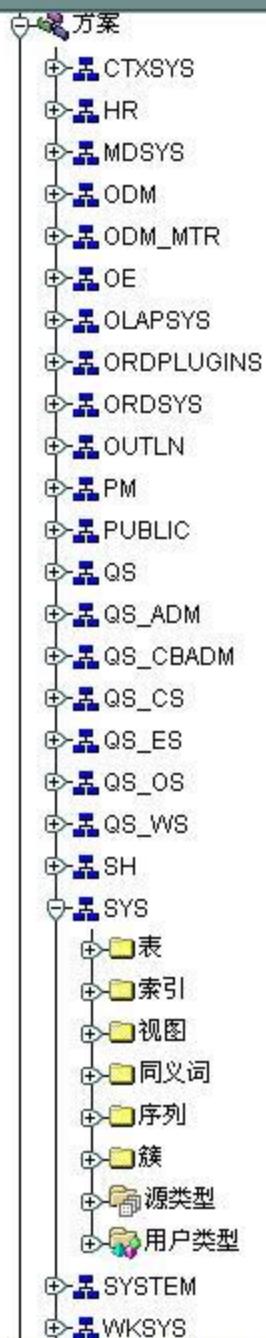
数据仓库

工作空间

XML 数据库

名称	类型	区管理	大小 (M)	已使用 (M)	利用率
CWMNLITE	PERMANENT	LOCAL	20.000	9.375	46.88
DRSYS	PERMANENT	LOCAL	20.000	9.688	48.44
EXAMPLE	PERMANENT	LOCAL	149.375	148.875	99.67
INDX	PERMANENT	LOCAL	25.000	0.063	0.25
MYTH	PERMANENT	LOCAL	5.000	0.063	1.25
ODM	PERMANENT	LOCAL	20.000	9.375	46.88
SYSTEM			400.000	397.438	99.36
TOOLS			10.000	6.063	60.63
UNDOTBS1			200.000	11.313	5.66
USERS			25.000	0.063	0.25
XDB			38.125	37.938	99.51
TEMP			40.000	0.000	0.00





存储管理

使用“存储管理”可通过数据库管理存储对象。

- ▶ 创建表空间、数据文件、回退段和重做日志组存储对象。
- ▶ 将数据文件和回退段添加到表空间中。
- ▶ 删除存储对象。
- ▶ 将对象联机或脱机。
- ▶ 显示对象的相关性。
- ▶ 对数据库文件执行备份、恢复和维护操作。
- ▶ 查看存储布局。此功能只在 Oracle 版本 9.2 和更高版本中可用。



有关存储管理的详细信息，
请单击[速成教学](#)按钮。

方案

- CTXSYS
- HR
- MDSYS
- ODM
- ODM_MTR
- OE
- OLAPSYS
- ORDPLUGINS
- ORDSYS
- OUTLN
- PM
- PUBLIC
- QS
- QS_ADM
- QS_CBADM
- QS_CS
- QS_ES
- QS_OS
- QS_WS
- SH
- SYS
 - 创建(C)...
 - 使用向导创建...
 - 保存列表(S)...
 - 数据管理
 - 分析...
 - 查找数据库对象...
 - 查找(F)...
- SYSTEM
- WKSYS

表

表空间

已分区

行数

上次分析时间

ACCESS\$	SYSTEM	No		
APPLY\$_CONF_HDLR_COLUMNS	SYSTEM	No		
APPLY\$_DEST_OBJ	SYSTEM	No		
APPLY\$_DEST_OBJ_CMAP	SYSTEM	No		
APPLY\$_DEST_OBJ_OPS	SYSTEM	No		
APPLY\$_ERROR	SYSTEM	No		
APPLY\$_ERROR_HANDLER	SYSTEM	No		
APPLY\$_SOURCE_OBJ	SYSTEM	No		
APPLY\$_SOURCE_SCHEMA	SYSTEM	No		
APPROLE\$	SYSTEM	No		
AQ\$_MESSAGE_TYPES	SYSTEM	No		
AQ\$_PENDING_MESSAGES	SYSTEM	No		
AQ\$_PROPAGATION_STATUS	SYSTEM	No		
AQ\$_PUBLISHER	SYSTEM	No		
AQ\$_QUEUE_STATISTICS	SYSTEM	No		
AQ\$_QUEUE_TABLE_AFFINITIES	SYSTEM	No		
AQ\$_REPLAY_INFO	SYSTEM	No		
AQ\$_SCHEDULES	SYSTEM	No		
AQ_EVENT_TABLE	SYSTEM	No		
AQ_SRVNFTN_TABLE	SYSTEM	No		
ARGUMENT\$	SYSTEM	No		
ASSOCIATION\$	SYSTEM	No		
ATEMPTAB\$		No		
ATTRCOL\$	SYSTEM	No		
ATTRIBUTE\$	SYSTEM	No		
ATTRIBUTE_TRANSFORMATIONS\$	SYSTEM	No		
AUD\$	SYSTEM	No		
AUDIT\$	SYSTEM	No		
AUDIT_ACTIONS	SYSTEM	No		
AURORA\$SHUTDOWN\$CLASSES\$	SYSTEM	No		
AURORA\$STARTUP\$CLASSES\$	SYSTEM	No		

表向导, 步骤 1, 共 13 步: 简介

简介

您希望新表使用什么名称?

XS

希望该表成为哪个方案的一部分?

SYS

要在哪个表空间中创建表?

MYTS

取消

帮助

< 后退(B)

下一步(N) >

- 方案
 - CTXSYS
 - HR
 - MDSYS
 - ODM
 - ODM_M
 - OE
 - OLAPSY
 - ORDPLU
 - ORDSYS
 - OUTLN
 - PM
 - PUBLIC
 - QS
 - QS_ADM
 - QS_CBA
 - QS_CS
 - QS_ES
 - QS_OS
 - QS_WS
 - SH
 - SYS
 - 表
 - 索引
 - 视图
 - 同义词
 - 序列
 - 簇
 - 源类型
 - 用户类型
 - SYSTEM
 - WKSYS

表	表空间	已分区	行数	上次分析时间
ACCESS\$	SYSTEM	No		
AUDIT\$	SYSTEM	No		
AUDIT_ACTIONS	SYSTEM	No		
AURORA\$SHUTDOWN\$CLASSES\$	SYSTEM	No		
AURORA\$STARTUP\$CLASSES\$	SYSTEM	No		
AUX_STATS\$	SYSTEM	No		
AW\$	SYSTEM	No		
AW\$CWMTOECM	SYSTEM	No		
AW\$EXPRESS	SYSTEM	No		
BOOTSTRAP\$	SYSTEM	No		
CCOL\$	SYSTEM	No		
CDC_CHANGE_COLUMNS\$	SYSTEM	No		

表向导, 步骤 4, 共 13 步: 空约束条件和唯一性约束条件

空约束条件和唯一性约束条件

已定义的列:

XH
XM
ZYM
XB
CSSJ
ZXF
BZ

列的约束条件: BZ

列名: BZ

列数据类型: VARCHAR2(200)

列值是否可以空?

☒ 是, 可以为空☐ 不, 不能为空

列值是否必须唯一?

☐ 是, 必须唯一

约束条件名称: <系统分配的>

☒ 不, 无须唯一

取消

帮助

< 后退(B)

下一步(N) >

完成(F)

AUDIT\$	SYSTEM	No
AUDIT_ACTIONS	SYSTEM	No
AURORA\$SHUTDOWN\$CLASSES\$	SYSTEM	No
AURORA\$STARTUP\$CLASSES\$	SYSTEM	No
AUX_STAT\$	SYSTEM	No
AW\$	SYSTEM	No
AW\$CWMTOECM	SYSTEM	No
AW\$EXPRESS	SYSTEM	No
BOOTSTRAP\$	SYSTEM	No
CCOL\$	SYSTEM	No
CDC_CHANGE_COLUMNS\$	SYSTEM	No

Oracle9i 体系结构

Oracle9i的体系结构

- Oracle9i数据库的逻辑结构

Oracle9i数据库的逻辑结构如下图所示。

数据库

Oracle9i数据库

表空间

表空间

表空间

...

表空间

逻辑对象

表

索引

...

视图

数据段

数据段

索引段

临时段

回滚段

数据区间

数据区间

...

数据区间

数据区间

...

数据区间

数据块

数据块

...

数据块

数据块

...

数据块



- Oracle9i数据库的逻辑结构由6层组成，每一个Oracle9i数据库服务器可以有多个数据库，每一个数据库可以有多个表空间，每一个表空间可以有多个表，每一个表可以有多个段，每一个段可以有多个区间，每一个区间可以有多个数据块。

表空间

- Oracle9i数据库通过表空间来组织数据库数据，表空间是Oracle数据库中数据的逻辑组织单位。数据库逻辑上由一个或多个表空间组成，表空间物理上是由一个或多个数据文件组成。
- Oracle9i数据库首先需要建立表空间才能将数据插入到表空间中的一个对象中。建立数据库中的对象时，必须指定要存放的表空间。Oracle9i使用表空间这个虚拟的逻辑对象，用户可以将不同性质的逻辑对象存放在不同的表空间下。

段、区间和数据块

■ 段

段用于存放数据库中特定逻辑结构的所有数据。正如Oracle为数据库表空间预先分配数据文件作为物理存储区一样，Oracle也为数据库对象（如表，索引等）预先分配段作为其物理存储区。

- Oracle数据库中常用的段有数据段、索引段、临时段和回滚段。

（1）数据段用于存放表中的数据。

（2）索引段用于存放索引数据。

（3）临时段用于存放临时数据。如排序操作所产生的临时数据等。

（4）回滚段用于存储事务的回滚信息。事务用回滚段来记录事务所修改数据的旧值，当需要恢复数据库时，从回滚表空间中提取回滚信息撤销事务。

■ 区间

区间由连续分配的相邻数据块组成。Oracle对段空间的分配是以区间为单位进行的。一个段由一个或多个区间组成。当一个段的所有空间都使用完毕后，Oracle就会为该段分配新的区间，直到表空间的数据文件中没有自由空间或者已达到每个段内部的区间最大数。当用户撤销一个段时，该段所使用的区间就成为自由空间。

■ 数据块

数据块是数据库中最小的、最基本的存储单位。它们是数据库能够分配给对象的最小单元。Oracle数据块与操作系统块有所不同，操作系统块是操作系统能从磁盘读写的最小单元，而Oracle数据块则是Oracle系统从磁盘读写的最小单元。

表及其他逻辑对象

- 表是用于存放数据的数据库对象，是Oracle数据库中最常用的存储机制。按照功能的不同，表分为系统表和用户表。系统表又称数据字典，用于存储管理用户数据和数据库本身的数据，记录数据、口令、数据文件的位置等；用户表就是由用户建立的，用于存放用户数据。

其他的逻辑对象

- 高级队列：队列是一种先进先出的数据结构。Oracle中的高级队列用于存储来自数据库系统的信息，包括数据库如何调度执行各种任务、数据库的状态等。
- 数据类型：Oracle9i提供自定义数组的功能，包括数组元素的个数、元素的类型、长度和精度等。
- 簇：是一种将相互关联、具有相同字段的一些数据表集中存储的机制。其目的是通过数据的集中存放来减少用户查询数据时的输入输出次数。
- 数据库链接：Oracle9i数据库具有访问分布式数据库的能力。

- 函数：指用PL/SQL语言编写的命名程序块，用以完成某些特定的功能。
- 索引：指利用索引关键字进行快速访问数据表中记录的一种机制。
- Java源：Java源是一些Java源代码，这些源代码可以作为Java共享过程相互调用。
- 实体化视图：又称快照，是包含了对一个或多个表的查询结果的表，被查询的这些表可以在不同的数据库中。

- 过程：过程指用PL/SQL编写的、用以完成某些特定功能的命名程序块、与函数不同的是过程没有返回值。
- 序列是一个连续的数字生成器。
- 同义词：同义词其实就是一个别名，使用同义词可以让多个用户访问同一个对象而不用使用拥有者的名称作为前缀。同义词是一个指向Oracle对象的指针。
- 嵌套表：嵌套表又称为表类型，指在定义表时，表中的一列是另外一个表。

- 触发器是一些过程，与表关系密切，用于保护表中的数据，当一个基表被修改时，触发器自动执行，例如通过触发器可以实现多个表间数据的一致性和完整性。触发器与应用程序无关。例如，对于XSCJ数据库有XS表，XS_KC表和KC表，当插入某一学号学生的某一课程成绩时，该学号应是XS表中已存在的，课程号应是KC表中已存在的。
- 视图：视图中并没有存放数据，而仅仅是一条SELECT语句。对用户而言，如果不告诉是在视图中操作的话，还以为是在表中工作。使用视图可以增强数据表的安全性和隐蔽数据的复杂性。

Oracle9i数据库的物理结构

- Oracle9i数据库的物理文件主要有以下几种：
 - (1) 数据文件；
 - (2) 控制文件；
 - (3) 日志文件；
 - (4) 初始化参数文件；
 - (5) 其他Oracle物理文件。

数据文件

- 数据文件就是用来存放数据库数据的物理文件，以.DBF作为后缀。
 - 数据文件存放的主要内容如下：
 - (1) 表中的数据；
 - (2) 索引数据；
 - (3) 数据字典定义；
 - (4) 存储过程、函数和数据包的代码；
 - (5) 用来排序的临时数据。
- Oracle9i数据库系统把这些不同的数据分别存放在一个和多个数据文件中。
- Oracle9i数据库至少包含一个数据文件。表空间的物理组成元素就是数据文件，一个表空间可以包含多个数据文件，并且每个数据文件只能属于一个表空间。

控制文件

- 控制文件用于记录和维护整个数据库的全局物理结构，它是一个二进制文件，以.CTL为文件后缀。控制文件存放了与Oracle9i数据库物理文件有关的关键控制信息，如数据库名和创建时间，物理文件名、大小及存放位置等信息。控制文件在创建数据库时生成，以后当任何数据库发生任何物理变化时都将被自动更新。

日志文件

- 日志文件用于记录对数据库进行的修改操作和事务操作，以.log为文件后缀。在数据库使用过程中，当例程启动失败或介质发生故障时，可以使用日志文件进行恢复。

初始化参数文件

- 初始化参数文件是一个文本文件，定义了要启动的数据库及内存结构的大约200多项参数信息。

Oracle9i数据库服务器的总体结构

■ 总体结构

Oracle9i数据库服务器由数据库管理系统和数据库组成。数据库管理系统由内存结构、后台进程和服务进程组成。

数据库由数据文件、控制文件和日志文件组成。

■ 内存结构

Oracle9i数据库服务器是由内存结构和Oracle进程组成。

Oracle有两种类型的内存结构：系统全局区(System Global Area, SGA) 和程序全局区(Program Global Area, PGA)

■ 1、系统全局区

系统全局区，是运行在客户机上的用户进程和运行在服务器上的服务器进程所使用的内存区域。在Oracle例程中，系统全局区是最重要的存储结构，它是所有通信的中心，包含数据维护、SQL语句分析和重做日志缓存所必须的所有内存结构，所有的用户进程和服务器进程都可以访问这部分内存区域，也就是说系统全局区内的数据是共享的。

- 系统全局区根据其功能的不同，又分成4个部分：数据缓存区、字典缓冲区、日志缓冲区和SQL共享池。
 - (1) 数据缓冲区：用于存储最近从数据库中读取出来的数据块。
 - (2) 字典缓冲区：数据库对象的信息存储在特殊的数据字典表中，包括数据文件名、权限和用户等。当数据库需要这些信息时就从数据字典中读取并存放在字典缓冲区中。

- (3) 日志缓冲区：任何事务在存入日志文件之前都存放在SGA的日志缓冲区内。数据库系统定期将该缓冲区的内容写入日志文件中。
- (4) SQL共享池：SQL共享池是程序的高速缓冲区，存放的是所有通过SQL语法分析并准备执行的SQL语句。

■ 2、程序全局区

程序全局区是单个用户进程所使用的内存区域，每一个连接到Oracle数据库的进程都需要自己私有的内存区，存放单个进程工作时需要的数据和控制信息。

后台进程

- 用户进程和服务进程。

所谓用户进程指在客户机上运行的程序。

服务进程指在服务器上运行的程序，接受用户进程发出的请求，根据请求与数据库通信，完成与数据库的连接操作和I/O访问。

后台进程帮助用户进程和服务进程进行通信，无论是否有用户连接数据库它们都在运行，负责数据库的后台管理工作。

下面是几个Oracle9i数据库的主要后台进程。

- (1) 系统监视进程：在数据库系统启动时执行恢复性工作的强制性进程。
- (2) 进程监视进程：查看进程是否过早断开，并处理任何其他必要的清除任务。
- (3) 数据库书写进程：将修改过的数据块写回到数据文件。
- (4) 日志写入进程：将重做日志写入到日志文件中。
- (5) 恢复进程：在分布式数据库中恢复失败的事务。

Oracle9i数据库的应用构架

- 常见的应用构架：
 - (1) 多磁盘系统；
 - (2) 磁盘映像系统；
 - (3) 客户—服务器系统；
 - (4) 共享服务器系统；
 - (5) 分布式数据库系统；
 - (6) 集群结构。

多磁盘系统

- 在一个多磁盘系统中，可以将数据文件分别存放在不同的磁盘上，这样在数据库进行操作时，就避免了同时从一个磁盘读取多个数据文件的操作，从而提高了数据库的整体性能。

数据库管理系统

数据库

SGA（系统全局区）

后台进程

数据文件1

数据文件2

数据文件3

磁盘1

磁盘2

磁盘3

多磁盘系统



磁盘映像系统

- 磁盘映像系统在逻辑上有一个数据库服务器，一个数据库服务器上有多个磁盘，采用磁盘映像技术，数据库文件在每一个磁盘上都有完整的备份，

SGA（系统全局区）

数据库管理系统

后台进程

数据库

数据库文件1

数据库文件2

数据库文件3

磁盘1

磁盘2

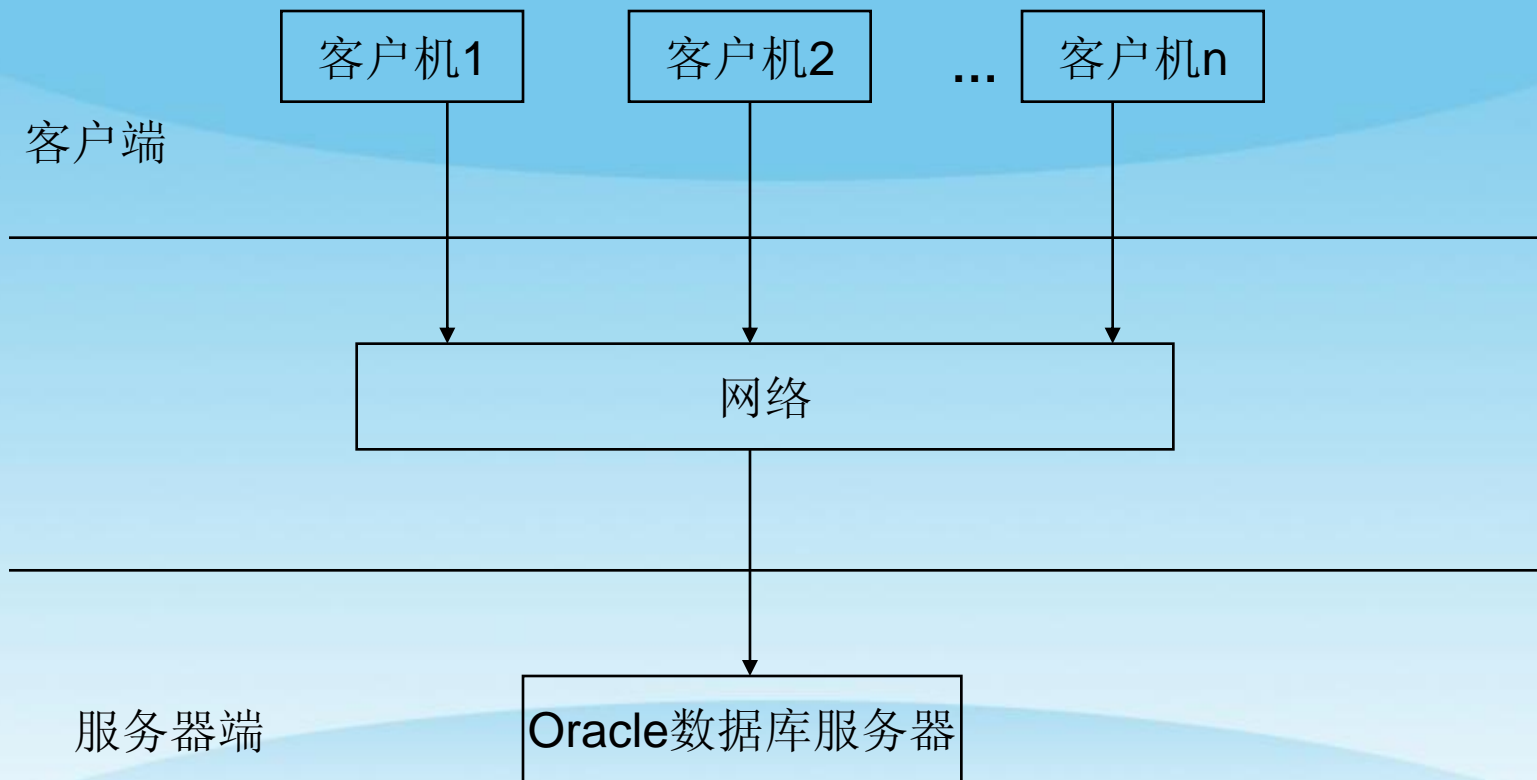
磁盘3

磁盘映像系统

- 优点：磁盘映像可以作为磁盘失效时的备份。在大多数操作系统中，磁盘失效会自动引发映像磁盘来取代失效的磁盘。
- 磁盘映像可以改善系统的性能。

客户—服务器系统

- 在客户—服务器系统中，将数据库服务器的管理和应用分布在两台计算机上，客户机上安装应用程序和连接工具，通过Oracle专用的网络协议建立和服务器的连接，发出数据请求。服务器上运行数据库，接收连接请求，将执行结果返回客户机。



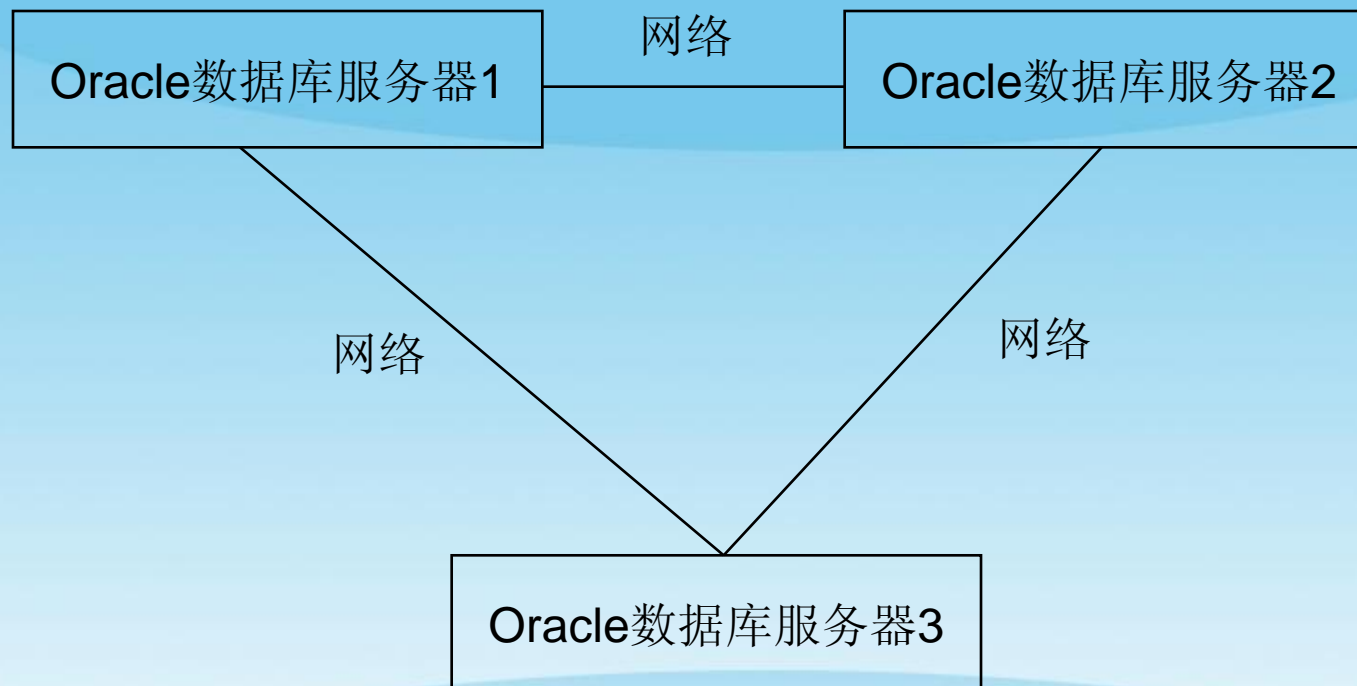
客户—服务器系统

■ 优点：

- （1）客户机和服务器可以选用不同的操作系统。
- （2）应用与服务分离。用户的程序在客户机上运行，只有在需要服务器时才发送请求，减轻了数据库服务器的负担。
- （3）服务器和客户机可以选用不同的硬件平台，服务器选用高档计算机，而大多数客户端使用一般的计算机即可，从而降低了成本。

分布式数据库系统

- 分布式数据库系统指一个单独的数据库位于不同物理位置的主机上。不同的物理位置可以是相邻的办公室，也可以是地球的另一边。在一个分布式环境中，不同服务器上的数据库虽然在物理上是分开的，但在逻辑上却是一个整体。



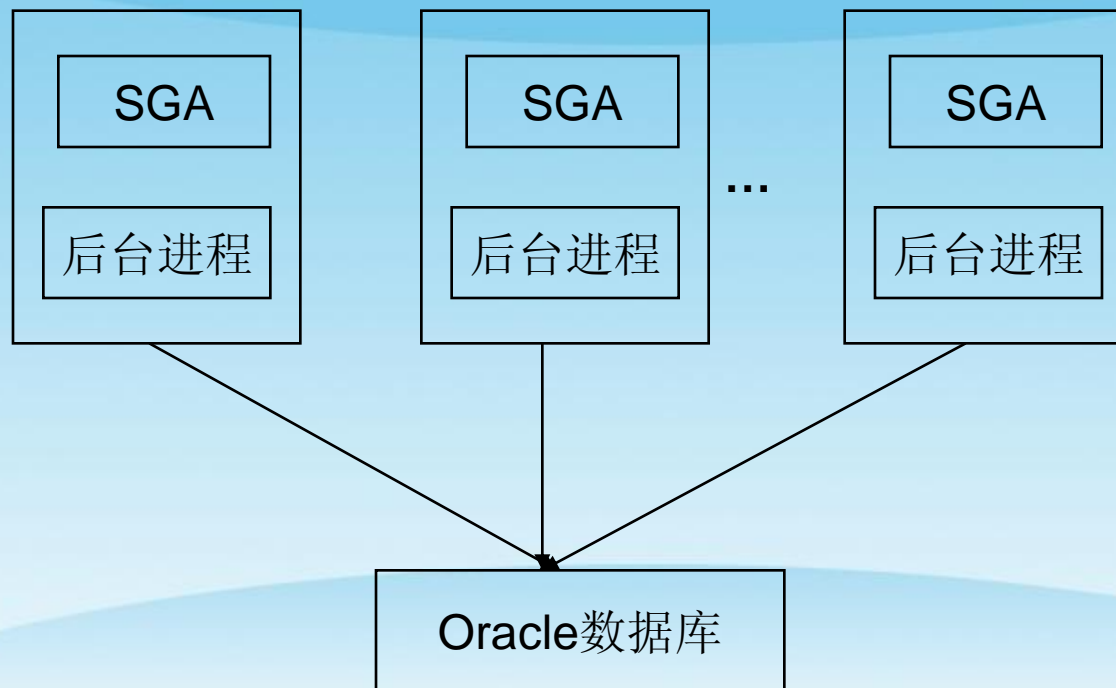
分布式数据库系统

■ 特点:

- (1) 局部自治。本地机器上拥有数据，并能对数据进行维护和管理。
- (2) 分布式查询。用户可以在一个节点上查询另一个节点上的数据库。
- (3) 分布式事务处理。用户可以执行分布式更新、插入和删除操作。

群集结构

- 群集结构指使用多个数据库例程来访问同一个数据库



■ 优点:

- (1) 采用并行结构的服务器，增加了系统资源，如更多的SGA、后台进程等。
- (2) 提供了一种对灾难进程恢复的有效手段，如果一个服务器停止工作，其他服务器仍可以提供服务。
- (3) 根据执行进程的类型对用户分组，让占用大量CPU时间的用户单独驻留在一台主机上。

SQL语言基础及应用

SQL语言

- SQL (Structure Query Language, 结构化查询语言) 于1975年由Boyce和Chamberlin提出, 用来实现关系运算中查询、选择等操作, 是一种综合的、功能极强的同时又简单易学的语言。作为关系数据库操作的标准语言, 目前已被ANSI正式批准为数据库的工业标准。
- SQL语言功能极强, 可以完成数据库整个生命周期的全部活动。SQL是一种面向集合的操作语言, 即操作对象和操作结果都是元组的集合。

- SQL语言按照功能可以分为3类：DDL、DML和DCL。

1、DDL（Data Definition Language，数据定义语言）：用于建立、修改、删除数据库对象，下面是一些DDL语句。

CREATE Database：创建数据库。

CREATE Tablespace：创建表空间。

CREATE Table：创建表。

CREATE View：创建视图。

CREATE Index：创建索引。

ALTER Database: 修改数据库。

ALTER Tablespace: 修改表空间。

ALTER Table: 修改表。

DROP Database: 删除数据库。

DROP Tablespace: 删除表空间。

DROP View: 删除视图。

DROP Index: 删除索引。

■ 2、DML

DML (Data Manipulation Language, 数据操作语言)：用于对数据库中的数据进行操作，DML包含的SQL语句如下。

SELECT：查询数据库中的数据。

INSERT：把数据插入到数据库。

■ 3、DCL

DCL (Data Control Language, 数据控制语言) : 用于控制对数据库的访问。DCL包含的一些语句如下:

GRANT: 授权用户对数据库对象的操作权限。

REVOKE: 撤销对用户的授权。

COMMIT: 提交事务, 用于事务处理。

ROLLBACK: 回滚事务, 用于事务处理。

LOCK: 锁定数据库的某一部分, 直到某一个事务完成, 用于并发控制。

oracle9i中的SQL环境

- 在Oracle9i中，进行SQL语句操作的最主要工具是SQL*PLUS。

SQL*Plus用于执行大多数的SQL命令和语句，是数据库管理员操作数据库中数据最直接和有效的工具。在使用SQL*PLUS之前，首先要登陆SQL*Plus工具，有两种登陆方式：DOS方式和Windows方式。

- 1、DOS方式登录SQL*Plus

在DOS方式下使用SQL*Plus，需要在DOS方式下输入“sqlplus.exe”命令，接着再输入用户名和口令，当出现SQL>提示符后，就可以使用SQL*Plus工具了。

- 2、Windows方式登录SQL*Plus

选择[开始] | [程序] | Oracle-
OraHome92 | Application Development | SQL*PLUS

CREATE语句

- 该语句可以创建表、索引、视图、同义词、过程、函数等。

1、表的建立

```
CREATE TABLE table(field1 type[(size)][index1], field2  
type [(size)])
```

字段	类型	说明
NO	CHAR(8)	编号
TITLE	VARCHAR2(50)	书名
AUTHOR	VARCHAR2(20)	作者
PUBLISH	VARCHAR2(20)	出版社
PUB_DATE	DATE	出版日期
PRICE	NUMBER(6, 2)	价格

■ CREATE TABLE BOOK (
 NO CHAR(8) PRIMARY KEY,
 TITLE VARCHAR2(50) NOT NULL,
 AUTHOR VARCHAR2(20),
 PUBLISH VARCHAR2(20),
 PUB_DATE DATE,
 PRICE NUMBER(6, 2));

2、约束条件

约束条件是数据库系统提供的对数据的完整性进行制约的机制。通俗的讲就是将不合法的数据摒弃在数据库外的机制。Oracle9i 允许创建5种约束条件：

非空(NOT NULL)：用来防止NULL值数据进入数据库，这种类型的约束条件定义在具体的数据列上。

主关键字：关键字是指数据库种惟一标识每个记录的属性或者属性的组合。主关键字约束条件可以用来惟一区分表中的每个记录，因此一个数据表只能建立一个主关键字约束条件。

惟一：惟一约束条件可以用来确保表中的每个值或者值的集合没有重复值。

外关键字：通过使用公共列在两个表之间建立关联，一个数据表的某些列为另外一个数据表的外键。

检查：用来检查是否满足约束条件中指定的条件，从而确保值在指定的范围内。

■ CREATE TABLE MYFRIND
(NAME VARCHAR2(10),
ADDRESS VARCHAR2(50) NOT NULL,
TELEPHONE VARCHAR2(20) NOT NULL,
AGE NUMBER(2) NOT NULL,
BIRTHDAY VARCHAR2(10) NOT NULL)
TABLESPACE USERS;

非空约束
条件的标
志

名称	数据类型	大小	小数位数	说明
NAME	VARCHAR2	10		姓名
ADDRESS	VARCHAR2	50		地址
TELEPHONE	VARCHAR2	20		电话
AGE	NUMBER	2		年龄
BIRTHDAY	DATE			生日

例：对AGE实行检查约束条件。

在创建数据表的【约束条件】选项卡里面。age>=10
and age<=99.

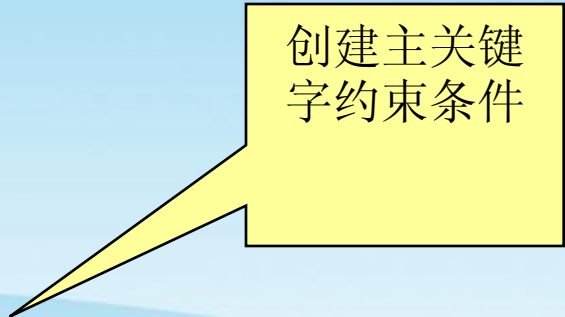
```
CREATE TABLE MYFRIEND  
( NAME    VARCHAR2(10),  
  ADDRESS  VARCHAR2(50) NOT NULL,  
  TELEPHONE  VARHCAR2(20) NOT NULL,  
  AGE      NUMBER(2) NOT NULL,  
  BIRTHDAY  VARCHAR2(10) NOT NULL,  
  CONSTRAINT AGECHECK CHECK(age>=10 and age<=99))  
TABLESPACE USERS;
```

检查约束条件
的代码

3、创建主关键字

Oracle9i数据表只允许有一个关键字，Oracle为关键字中的每个列创建一个惟一索引和非空约束条件。

```
CREATETABLE MYFRIEND(  
    NAME VARCHAR2(10),  
    TELEPHONE VARCHAR2(20) NOT NULL,  
    AGE NUMBER(2) NOT NULL,  
    BIRTHDAY DATE,  
    CONSTRAINT NAMEPRIMARY PRIMARYKEY (NAME) )  
TABLESPACE USERS;
```



创建主关键字约束条件

■ 4、创建外关键字约束条件

数据表，包含学生的信息和考试成绩的信息。设计的数据表包含的字段有学号、姓名、班级、语文成绩、数学成绩等。假如我们要删除学生的成绩，则有关该学生的所有信息都会丢失。

常用的方法是将其设计成两个数据表，一个名为STUDENT，存储学生的信息，以学号为主关键字，另外一个名为SCORE，存储考试成绩的信息，也是以学号为主关键字。很显然，这两个数据表之间存在关联关系。这种关联关系可以这样描述：如果STUDENT数据表中没有某个学号的学生纪录，则在SCORE数据表

里就不应该有该学生的学习成绩；如果STUDENT数据表中存在某个学号的学生纪录，则在SCORE数据表里就一定存在该学生的学生成绩。

表述上述关系的数据库实现方法就是外关键字约束条件。将STUDENT数据表的主关键字STUDENTNO（学号），设计为SCORE数据表的外关键字。

使用SELECT语句查询数据

- SQL语言使用SELECT语句实现对数据表的任何查询，包括选择符合条件的行或列及其其他操作等。常用的SELECT语法格式如下：

SELECT 字段1， 字段2， ……

FROM 表1[, 表2]……

WHERE 查询条件

GROUP BY 分组字段1[, 分组字段2]……HAVING
分组条件

ORDER BY 列1[, 列2]……

SELECT中的条件语句

(1) FROM条件子句

FROM条件子句指定表或视图的名称，其中包含列在SELECT语句中的字段数据。语句格式如下：

```
SELECE fieldlist
```

```
FROM tableexpression
```

例：从EMPLOYEE下，查询出所有NAME字段的数据，
可以用下面的语句来实现；

```
SELECT NAME FROM EMPLOYEE;
```

(2) WHERE条件子句。WHERE条件子句指定查询的条件与限制，语句格式如下：

```
SELECT fieldlist  
FROM tableexpression  
WHERE criteria
```

运算符

- Oracle提供了3类运算符：算术运算符、关系运算符和逻辑运算符。

1、算术运算符

算术运算符执行算术运算。算术运算符有6种，它们是：

+ (加)，- (减)，* (乘)，/ (除)，** (指数)、|| (连接)，其中+ (加)和- (减)运算符也可用于对DATE数据类型的值进行算术运算。

2、关系运算符

关系运算符（又称比较运算符）有以下几种：

- (1) = 等于、<> 或 != (不等于)、< (小于)、> (大于)、>= (大于或等于)、<= (小于或等于)；
- (2) BETWEEN... AND... (检索两值之间的内容)；
- (3) IN (检索匹配列表中的值)；
- (4) LIKE (检索匹配字符样式的数据).；
- (5) IS NULL (检索空数据).

- 例：查询总学分在40~50的学生的学号和姓名。

```
SELECT XH, XM, ZXF
```

```
FROM XS
```

```
WHERE ZXF BETWEEN 40 AND 50;
```

- 例：使用>=和<=代替BETWEEN实现与上例相同的功能。

```
SELECT XH, XM, ZXF
```

```
FROM XS
```

```
WHERE ZXF>=40 AND ZXF<=50;
```

■ 3、逻辑运算符

逻辑运算符用于对某个条件进行测试，运算结果位TRUE和FALSE。Oracle提供的逻辑运算符有：

- (1) AND (两个表达式位真则结果位真)；
- (2) OR (只要有一个为真则结果为真)；
- (3) NOT (取相反的逻辑值)。

■ 例：查询总学分不在40~50的学生的学号和姓名。

```
SELECT XH, XM, ZXF  
FROM XS  
WHERE ZXF NOT BETWEEN 40 AND 50;
```

- 例：查询总学分大于45的计算机系学生。

```
SELECT XH, XM, ZXF
```

```
FROM XS
```

```
WHERE ZXF>45 AND ZXM= '计算机' ;
```

- 例：查询计算机系和通信工程系学生的基本情况。

```
SELECT XH, XM, ZXM, ZXF
```

```
FROM XS
```

```
WHERE ZYM= '计算机' OR ZYM= '通信工程' ;
```

- (3) DISTINCT

格式如下：

```
SELECT DISTINCT FROM table
```

例：查询顾客表中的公司名称，要求公司名称不重复，

```
SELECT DISTINCT company FROM CUSTOMER;
```

- (4) ORDER BY条件子句。ORDER BY 条件子句通常与SELECT语句合并使用，目的是将查询的结果依照字段加以排序。语句格式如下：

例1、对表5中的salary字段排序。

```
SELECT name , salary FROM consult  
ORDER BY salary ;
```

- (5) GROUP BY条件子句。GROUP BY条件子句的功能是依据指定的字段，将具有相同数值的记录合并成一条
- (6) HAVING条件子句。指定一特定的分组记录，并满足HAVING所指定的条件或状态，

多表查询

- 多表查询指从多个有关联的表中查询数据，其基本语法跟单表查询类似。

例：查询雇员姓名和所在部门名称

```
SELECT ENAME, DNAME  
FROM SCOTT.EMP A, SCOTT.DEPT B  
WHERE A.DEPTNO=B.DEPTNO;
```


嵌套查询

■ 嵌套查询

嵌套查询指一个SELECT查询中包含一个以上的子查询，所谓子查询指嵌套在另一个SELECT、INSERT、UPDATE或DELETE语句中的SELECT查询语句。子查询的语法与SELECT语法类似，但有所限制，如子查询不能含有ORDER BY、COMPUT和INTO关键字等。

SELECT语句不仅仅能从表或视图中检索数据，它还能够从其他查询语句所返回的结果集合中查询数据。

在查询中的SELECT子句中建立表达式

- SELECT语句中位于SELECT关键词之后的列名用来决定哪些列将作为查询结果返回。用户可以按照自己的需要选择任意列，还可以使用通配符*来设定返回表格中的所有列，也可以是一组列名列表、星号、表达式、变量（包括局部变量和全局变量）

```
SELECT NAME, SALARY FROM CONSULT;
```

■ 1、数学函数

数学函数返回需要运算的数据的值。下面列出Oracle9i所支持的数学函数。

SQL标准函数	Oracle	注解
绝对值	ABS	绝对值
Arc cosine	ACOS	反余弦
Arc sine	ASIN	反正弦
Arc tangent of n	ATAN	反正切
Arc tangent of n and m	ATAN2	反余切
Smallest integer \geq value	CEIL	大于或等于指定值的最小整数
Cosine	COS	余弦
Exponential value	EXP	给定数据的指数值
Largest integer \leq value	FLOOR	小于或等于指定值的最大整数
Natural logarithm	LN	自然对数
Logarith, any base	LOG (N)	以N为底底对数
Logarithm, base 10	LOG (10)	以10为底的对数
Modulus (remainder)	MOD	模
Power	POWER	数据整数次幂
Random number	N/A	随机数

■ 2、字符串函数

Oracle中有一些专门处理字符数据的函数称为字符串函数，它们提供了丰富的字符处理功能。这些字符数据由字母、符号和数字组成。大多数字符串函数只能用于CHAR、VARCHAR和VARCHAR2数据类型。

函数	Oracle
把字符转换为ASCII	ASCII
字符串连接	CONCA
把ASCII转换为字符	CHR
返回字符串中的开始字符（左起）	INSTR
把字符转换为小写	LOWER
把字符转换为大写	UPPER
填充字符串的左边	LPAD
清除开始的空白	LTRIM
清除尾部的空白	RTRIM
从数字数据转换为字符数据	TO_CHAR
列表中最大的字符串	GREATEST
列表中最小的字符串	LEAST

■ 3、日期函数

SQL本身具有日期的知识，它能知道每年的天数、月份的名称和缩写、每月的天数、每星期各天的名称和缩写以及从公元前4712年到公元4712年之间的任意一天在星期中的排行。SQL语句主要通过日期函数操纵这些日期和时间数据。日期和时间数据应该用单引号括起来。

函数	Oracle
日期相加	(date column+/-value)or ADD_MONTHS
两个日期的差	(date colum+/- value)or MONTHS_BETWEEN
当前日期和时间	SYSDATE
一个月的最后一天	LAST_DAY
时区转换	NEW_TIME
日期后的第一个周日	NEXT_DAY
代表日期的字符串	TO_CHAR
代表日期的整数	TO_NUMBER (TO_CHAR)
日期舍入	ROUND
日期截断	TRUNC
字符串转换为日期	TO_DATE
如果为NULL则转换日期	NVL

■ 转换函数

转换函数允许用户把表达式从一种数据类型转换成另外一种数据类型，还允许把日期转化为不同的样式。

函数	Oracle
数字转换为字符	TO_CHAR
字符转换为数字	TO_NUMBER
日期转换为字符	TO_CHAR
字符转换为日期	TO_DATE
十六进制转换为二进制	HEX_TO_RAW
二进制转换为十六进制	RAW_TO_HEX

从表中检索特定行

- 在实际的大型应用中，一些表可能有上百万行数据，因此在一次查询中全部处理这些数据是不现实的。在数据库的实际运用中，往往需要对查询加以条件限制。在SQL中可以通过WHERE子句设置查询条件，过滤掉不需要的数据行。

- 在WHERE子句中支持子查询。子查询是指在WHERE子句中也包括查询语句。子查询主要是将自己的结果用于建立主查询的查找条件。

在下例中，将在AUTHORINFO表中查找籍贯为湖北省英山县的作者的作品。

```
SELECT bookname FROM bookinfo
```

```
WHERE authorname=
```

```
    (SELECT authorname FROM authorinfo
```

```
WHERE hometown= ‘湖北省英山县’ )
```



指定子查询要注意以下几点：

子查询中不能有ORDER BY子句。

当子查询返回值不是一个值而是一个集合时，不能用比较运算符。

- 例：查询作者为李国文的图书的书名和ISBN号码。

```
SELECT BOOKNAME, ISBN FROM BOOKINFO
WHERE BOOKNAME IN(
    SELECE BOOKNAME
    FROM BOOKINFO
    WHERE AUTHORNAME= ‘李国文’
);
```

排序查询

- 一列没有特定顺序的名字读起来很不方便。如果把
这些名字按某种顺序排列，读起来就比较容易。通
过使用ORDER BY 子句，用户可以强制一个查询结果
按升序排列。。

如果想把查询结果按降序排列，可以使用关键字
DESC。

函数查询

SELECT查询语句可以使用函数。

- 集合函数：

MIN: 计算最小值;

MAX: 计算最大值;

AVG: 计算平均值;

SUM: 求和;

COUNT: 计算符合条件的记录总数

■ 数值函数：

ABS： 计算绝对值

MOD： 计算余数

CEIL： 计算大于等于参数N的最小整数

FLOOR： 计算小于等于参数N的最大整数

POWER： 计算幂

SORT： 计算平方根

ROUND： 四舍五入

SIGN： 符号函数

- 字符串函数：

LENGTH： 获取字符串的长度

CONCAT： 字符串连接符号

LOWER： 转化为小写字符

UPPER： 转化为大写字符

SUBSTR： 截取子串

INSTR： 获取选定字符在字符串综合首次出现的位置

- 日期函数：

SYSDATE：获取日期和时间

MONTHS_BETWEEN：获取两个日期之间的月份间隔

ADD_MONTHS：在指定日期上添加月份

NEXT_DAY：指定日期的下一天

LAST_DAY：每月的最后一天

- 例：查询雇员总人数

```
SELECT COUNT(*) FROM EMP
```

- 把查询出来的部门名称以小写字符显示出来。

```
SELECT LOWER(DNAME) AS “部门名称” FROM  
DEPT;
```

- 求和

```
SELECT TO_NUMBER(‘100’)+TO_NUMBER(‘200’ )  
AS ‘求和’ FROM DUAL;
```

使用INSERT语句插入数据

- SQL语言用INSERT语句在数据表中插入数据。INSERT语句的语法一般有如下两种：

INSERT INTO 表名[字段1, 字段2, ……]

VALUES (值1, 值2, ……);

INSERT INTO 表名[字段1, 字段2, ……] SELECT
(字段1, 字段2, ……) FROM其他表名。

其中，INSERT INTO 指明要插入的表以及表中的字段，VALUES指明要插入相应字段的值。

注意：要插入的值与要插入的字段相互对应。

- 多行记录的插入

例：新建NEWEMP表，它的结构和EMP表一样，执行下面的语句。

```
INSERT INTO NEWEMP SELECT *FROM EMP;
```

使用UPDATE语句更新数据

- SQL使用UPDATE语句对数据表中的符合更新条件的记录进行更新。UPDATE语句的一般语法如下：

UPDATE 表名 SET 字段1=值1

[字段2=值2]……WHERE条件表达式；

例：为雇员BLAKE加薪10%。

UPDATE EMP SET SAL=SAL*1.1

WHERE ENAME= 'BLAKE'

例：为EMP表中的所有雇员加薪10%。

UPDATE EMP SET SAL=SAL*1.1;



使用DELETE语句删除数据

语法：

DELETE FROM 表名 [where 条件] ；

例：删除NEWEMP表中雇员姓名为MARY的记录。

DELETE FROM NEWEMP WHERE ENAME= 'MARY'

删除所有记录：

DELETE FROM NEWEMP;

表、视图及其他数据库对象

Oracle9i 中表的分类

- 关系表：默认的表类型，存储永久性的数据。
- 临时表：存储私有数据或一个会话中特定的数据，数据库中的其他用户不能使用这些数据。
- 索引表：按照结构化主关键字进行排序

创建表

要想创建表，必须首先以具有创建权限的用户名登录数据库。

- 创建关系表的命令是CREATE TABLE，以用户SCOTT创建数据表为例。

(1) 选择以【独立启动】的方式登录到数据库。

(2) 出现创建表的【一般信息】选项卡。

(3) 若需要调整列的次序，可以用鼠标列前方的方框，出现编辑列的菜单。

■ 5.2.2约束条件的配置

约束条件是数据库系统提供的对数据的完整性进行制约的机制。通俗的讲，就是将不合法的数据摒弃在数据库外的机制。Oracle9i允许创建五种约束条件。

- 非空(not null):用来防止null值数据进入数据库，这种类型的约束条件定义在具体的数据列上。
- 主关键字(PRIMARY):关键字是数据表中可以惟一标识每个记录的属性或者属性的组合。主关键字约束条件可以用来惟一区分表中的每个记录，因此自动隐含了非空约束条件，一个数据表只能建立一个主关键字约束条件。

- 惟一(unique):惟一约束条件可以用来确保表中的每个值或者值的集合没有重复的值。
- 外关键字(foreign):通过使用公共列在两个表之间建立关联,一个数据表的某些列为另外一个数据表的外键。
- 检查(check):用来检查是否满足约束条件中指定的条件,从而确保值在指定的范围内。

- 1、创建非空约束条件

在创建数据表的【一般信息】选项卡里可以创建非空约束条件。

- 2、创建检查约束条件

在数据列或数据表可以进行检查约束条件的设置，在同一列上可以定义多个检查约束条件，且该列可以有空值。

例：为AGE创建检查约束条件。

在【约束条件】选项卡里进行检查约束条件的配置。在【名称】中输入“AGECHECK”，在【类型】下拉框里选择‘CHECK’，在【检查条件】单元格输入“age>=10 and age<=99”。

- 3、创建主关键字约束条件

Oracle9i数据表只允许有一个主关键字，Oracle为关键字中的每个列创建一个惟一索引和非空约束条件。

例：为MYFRIEND数据表的” NAME” 数据列建立主关键字约束条件。

- 4、创建惟一约束条件

Oracle9i允许对多个列的组合值定义惟一约束条件。

例：为MYFRIDEND数据表的” NAME” 和” ADDRESS” 数据列创建惟一约束条件。

- 5、创建外关键字约束条件

例： STUDENT表和SCORE表

■ 存储参数的配置

Oracle9i的若干表空间构成了逻辑上的数据库。一个服务器上可以有多个数据库，一个数据库可以有多个表空间，一个表空间可以有多个表，一个表可以有多个段，一个段可以有多个区间，一个区间可以有多个数据块。

在创建数据表时，如果没有指定存储参数和表空间，数据表将创建在默认的用户表空间中，使用的存储参数就是默认表空间指定的默认存储参数，合理组织这些存储参数，可以提高数据表的性能。

■ 1、存储参数与性能的关系

在创建数据表时，系统将为数据表分配段，每个段里又包括若干区间。在需要时，系统将给表分配新的区间。

例子：一个数据表有10000条记录，分配区间的时候，如果每10个记录占用一个区间，完成一次全表扫描就需要磁盘输入/输出1000个区间。如果每100个记录占用一个区间，则同样的操作只需要输入/输出100个区间。很显然，由于系统的性能主要取决于磁盘的I/O次数，因此，两种划分存储参数的办法将直接影响数据库的性能的好坏。对于大型数据库尤其如此。

■ 2、用户配置存储参数

在【存储】选项卡中进行。

- 【初始大小】：对象的第一个物理存储区的大小，可以输入一个值，至少为一个数据块的大小，默认值为5个数据块的大小。
- 【下一个大小】：可分配给对象的下一个分区的大小，默认值为5个数据块的大小，允许的最小值为一个数据块的大小。
- 【增量】：每个区相对于上一个区的增长百分比（在第二个区之后）。

- **【最小数量】**：创建段时已分配的总区数，默认值为1，可以输入1或大于1的值。
- **【最大数量】**：Oracle数据库可以分配给该对象的总区数（包括第一个区）。选择**【无限制】**单选钮：创建的区数将只受表空间中连续空闲空间数量的限制。

例：在**【初始大小】**输入10，**【下一个大小】**输入10，在**【增量】**输入50，在**【最小数量】**输入1，在**【最大数量】/【值】**文本框里输入200。

这个参数的配置表明区的大小顺序为：10kb，10kb，15kb（比第二块增长50%），22.5kb（比第3块增长50%）

创建有簇列的表

1、 簇在一些有关的数据库书籍中也被称为聚簇，是一种数据表记录的存储方法。

例：两张数据表STUDENT和SCORE。

簇是包含多个数据表的对象，特点是簇内的所有表共有一个以上的公共列，叫做簇列，将逻辑上经常一起查询的数据表组织成的簇的形式可以提高查询速度

- 手动创建簇。
- 创建有簇的表。

将SCORE数据表的STUDENTNO列定义为簇列，这样在同一个数据块中数据的存储情况如下

102080 王海 200203班

102080 99.5 67.8

101981 李东 200204班

101981 98.5 89.0

- 5.2.6利用查询创建关系表
- 如果用户已经创建了一些数据表，且新创建的数据表可以由从已创建的数据表中提取数据或字段组成。

什么是视图？

- 视图是由一个或若干个基表产生的数据的集合，它不占用存储空间。
- 视图犹如数据表的窗户，管理员定义这些“窗户”的位置后，用户就只能查看他可以看到的数据库。视图不是数据表，它仅是一些SQL查询语句的集合，作用是按照不同的要求从数据表中提取不同的数据。

- 视图与表不同，视图是一个虚表，即视图所对应的数据不进行实际存储，数据库中只存储视图的定义，对视图的数据进行操作时，系统根据视图的定义去操作与视图相关联的表。

视图与表的比较

- 相同之处:

- 1、视图和表一样由列组成，其查询方式与表完全相同。

- 2、和表一样，用户也可以在视图中插入、更新或删除数据，在视图中做这些操作时与在表中是一样的。

■ 区别:

1、与表不同，视图没有数据，而仅仅是一条SQL查询语句。按此查询语句检索出的数据以表的形式表示。视图中的列可以在一个或多个基本表中找到。视图不使用物理存储位置来存储数据。

2、视图的定义（列的安排、授予的权限等等）存储在数据字典中。对一个视图进行查询时，视图将查询其基于的表，并且以视图定义所规定的格式和顺序返回值。

3、由于视图没有直接相关的物理数据，所以不能像表那样被索引。



创建一视图

可以在CREATE VIEW 中嵌入一子查询.

```
CREATE [OR REPLACE] [FORCE|NOFORCE] VIEW view  
    [(alias[, alias]...)]  
    AS subquery  
    [WITH CHECK OPTION [CONSTRAINT constraint]]  
    [WITH READ ONLY];
```

子查询中可以包含复杂的SELECT语句. 子查询不能包含ORDER BY语句.

例:创建一视图, EMPVU10, 包含部门10中所有员工的详细信息.

```
CREATE VIEW      empvu10
  AS SELECT empno,  ename,  job
  FROM          emp
  WHERE                  deptno = 10;
```

通过使用DESCRIBE命令来描述视图的结构.

```
DESCRIBE empvu10
```


在一视图使用DML语句的规则

在一些简单视图中, 你可以使用DML操作.

如果一视图中包含下面的内容, 你就不能删除一视图:

组函数

GROUP BY 语句

DISTINCT 关键字

移去一视图

移去一视图并不会丢失数据, 因为一视图是基于数据库中隐含的一张表的.

```
DROP VIEW view;
```

更新视图

- 通过更新视图数据可以修改基表数据。但不是所有的视图都可以更新，只有对满足可更新条件的视图，才能进行更新。
 - 1、没有使用连接函数、集合运算函数和组函数。
 - 2、创建视图的SELECT语句中没有集合函数且没有GROUP BY, DISTICT关键字。
 - 3、创建视图的SELECT语句中不包含从基表列通过计算所得的列。
 - 4、创建视图没有包含只读属性。

- 插入数据

使用INSERT语句通过视图向基本表中插入数据。

注意：当视图所依赖的基本表有多个时，不能向该视图中插数据。

- 修改、删除数据

使用UPDATE语句可以通过视图修改基本表的数据。

使用DELETE语句可以通过视图删除基本表的数据。

注意：对于依赖多个基本表的视图，不能使用DELETE语句。

数据库对象

对象	描述
表	基本的存储单元，由行和列组成
视图	将来自于一张或多张表中的数据子集逻辑地表示出来
序列	用于产生主键
索引	增进一些查询操作的性能
同义词	一个对象的别名

什么是序列?

是一种可被多个用户使用的用于自动产生一系列惟一数字的数据库对象。序列是一个共享的对象。通常被用来产生主键值。

创建序列的语句 CREATE SEQUENCE

定义一个序列来自动地产生有序的数字。

```
CREATE SEQUENCE sequence  
[INCREMENT BY n]  
[START WITH n]  
[ {MAXVALUE n | NOMAXVALUE} ]  
[ {MINVALUE n | NOMINVALUE} ]  
[ {CYCLE | NOCYCLE} ]  
[ {CACHE n | NOCACHE} ] ;
```

■ 例：

```
CREATE SEQUENCE dept_deptno  
INCREMENT BY 1  
START WITH 91  
MAXVALUE 100  
NOCACHE  
NOCYCLE;
```

用以确定在内存高速缓冲区预先装入的一组序列数目

确保在序列达到最大值（增序列）或最小值（减序列）之后不能产生更多的值，用以防止序列回转

查看你的序列

可以在USER_SEQUENCES数据字典表中查看你的序列a dictionary table.

```
SELECT      sequence_name, min_value, max_value,  
            increment_by, last_numbe  
            FROM      user_sequences;
```

NEXTVAL 和 CURRVAL 伪列

NEXTVAL返回下一个序列的值。

它一次返回一个值。

CURRVAL当前的序列值。

每引用一次序列的伪列NEXTVAL，就会按照序列的定义产生一个新的序列码；而通过序列伪列CURRVAL可以反复利用当前的序列码。

使用序列

- ◆插入一个新的部门” MARKETING” 地址在San Diego.

```
INSERT INTO dept(deptno, dname, loc)
VALUES (dept_deptno.NEXTVAL,
        'MARKETING', 'SAN DIEGO');
```

- ◆查看DEPT_DEPTNO序列当前的值。

```
SELECT dept_deptno.CURRVAL
FROM dual;
```

在以下场合可以使用两个伪列

- 1、INSERT语句的VALUES子句中
- 2、SELECT语句中的前面选择的表列名中
- 3、UPDATE语句中的SET子句

在以下场合不可以使用两个伪列

- 1、子查询中。
- 2、视图定义的查询中。
- 3、带有DISTINCT操作符的SELECT语句
- 4、带有GROUP BY 或ORDER BY子句的SELECT语句。
- 5、SELECT语句的WHERE子句中
- 6、检查约束条件中

- 修改一个序列

```
ALTER SEQUENCE dept_deptno  
    INCREMENT BY 1  
    MAXVALUE 999999  
    NOCACHE  
    NOCYCLE;
```

■ 删除一个序列

- 从数据字典中删除一个序列可以使用DROP SEQUENCE 语句。
- 一旦删除序列就不再能使用了。

```
DROP SEQUENCE dept_deptno;
```


索引如何被创建?

- 自动创建索引: 当你为一张表定义主键或唯一性约束条件时一个惟一的索引就已经被创建了。
- 手动创建索引: 用户可以自己创建索引。

创建一个索引

对一列或多列创建索引

```
CREATE INDEX index  
ON table (column[, column]...);
```

例：

```
CREATE INDEX emp_ename_idx  
ON emp (ename);
```

- 何时需要创建一个索引

在WHERE语句中该列被经常用到。

该列所包含的值有很大取值范围。

在WHERE语句中有两个或两个以上的列被经常在一起使用的时候。

表很大。

- 何时不需要创建一个索引

表很小时。

这列在查询中不经常用到。

表不断地被更新。

察看索引

USER_INDEXES 数据字典视图包含索引名和它的uniqueness.

USER_IND_COLUMNS 视图包含索引名、表名、列名。

```
SELECT      ic.index_name, ic.column_name,
2          ic.column_position col_pos, ix.uniqueness
3  FROM      user_indexes ix, user_ind_columns ic
4  WHERE     ic.index_name = ix.index_name
5  AND              ic.table_name = 'EMP' ;
```

基于函数的索引

基于函数的索引是一个基于表达式的索引。
索引表达式是建立于表的列，SQL函数等。

例：

```
CREATE TABLE test (col1 NUMBER);  
CREATE INDEX test_index on test(col1,col1+10);  
SELECT col1+10 FROM test;
```

删除一个索引

从数据字典中删除索引

```
DROP INDEX index;
```

从数据字典中删除EMP_ENAME_IDX索引。

```
DROP INDEX emp_ename_idx;
```

要删除一个索引你必须是这个索引的拥有者或拥有删除索引的权限。

同义词

同义词是一个对象的别名，使用同义词可以让多个用户访问同一个对象而不用添加拥有者的名称作为前缀。同义词是一个指向Oracle对象的指针，本身不包含自己的数据。

```
CREATE [PUBLIC] SYNONYM synonym  
FOR      object;
```


创建和删除一个同义词

为DEPT_SUM_VU 视图创建一个短一点的同义词.

```
CREATE SYNONYM d_sum  
FOR dept_sum_vu;
```

删除一个同义词。

```
DROP SYNONYM d_sum;
```

PL/SQL编程

PL/SQL编程基础

■ 什么是PL/SQL程序

对于标准化的SQL语言对数据库进行各种操作，每次只能执行一条语句，语句以英文的分号“；”为结束标识。这是因为Oracle数据库系统不像VB，VC这样的程序设计语言，侧重于后台数据库的管理，因此提供的变成能力较弱，而结构化编程语言对数据库的支持能力又较弱。

在这种要求的驱使下，Oracle公司在标准SQL语言的基础上发展了自己的PL/SQL语言，将变量、控制结构、过程和函数等结构化程序设计的要素引入了SQL语言中，这样就能够编制比较复杂的SQL程序了，利用PL/SQL语言编写的程序也称为PL/SQL程序块。

PL/SQL程序的结构

■ 实例:

建立名为testtable的数据表，在该表中有recordnumber整数型字段和currentdate时间型字段，编制一个PL/SQL程序完成向该表中自动输入100个记录的操作，要求recordnumber字段从1到100，currentdate字段为当前系统时间。

```
create table testtable(  
    recordnumber int,  
    currentdate date);  
set serveroutput on  
declare  
    maxrecords constant int:=100;  
    i int:=1;  
begin  
    for i in 1.. maxrecords loop  
        insert into      testtable(recordnumber,currentdate)  
            values(i, sysdate);  
    end loop;  
    dbms_output.put_line( '成功录入数据! ' );  
    commit;  
end;
```

程序代码	说明
Set serveroutput on	允许服务器输出
declare	定义部分标识
maxrecords constant int:=100;	定义maxrecords为整型常量100
i int:=1;	定义i为整型值变量，初值为1
begin	执行部分标识
for i in 1..maxrecords loop	i 从1循环到maxrecords
insert into testtable (recordnumber, currentdate) values(i,sysdate);	向数据表中插入数据
end loop;	结束循环
dbms_output.put_line('成功录入数据!');	显示成功录入数据信息
commit;	提交结果
end;	结束执行

PL/SQL程序的总体结构

该部分包含在这行部分里面，以exception为标识，对程序执行中产生的异常情况进行处理（有的程序比较简单，往往省略异常处理部分。

```
declare  
    定义语句段  
begin  
    执行语句段  
exception  
    异常处理语句段  
end
```

以declare为标识，在该部分定义程序中要使用的常量、变量、游标等

以begin为开始标识。该部分是每个pl/sql程序所必备的，包含了对数据库的操作语句和各种流程控制语句

常 量

- 语法格式：常量名 constant 类型标识符 [not null]:=值；
- 要求：常量名与后面的变量名都必须以字母开头，不能有空格，不能超过30个字符长度，同时不能和保留字同名，常（变）量名称不分大小写，在字母后面可以带数字或特殊字符。括号内的not null为可选参数，若选用，表明该常（变）量不能为空值。

- 例：定义名为pi的数字型常量，长度为9。

```
declare
```

```
    pi  constant number(9) := 3.1415926;
```

```
begin
```

```
    commit;
```

```
end;
```

基本数据类型变量

- 语法格式：变量名 类型标识符 [notnull]:=值；
- 实例：定义名为age的数字型变量，长度为3，初始值为26。

```
declare  
    age number(3) := 26;  
begin  
    commit;  
end;
```

类型标识符	说明
number	数字型
int	整数型
Pls_integer	整数型，产生溢出时出现错误
Binary_integer	整数型，表示带符号的整数
Char	定长字符型，最大255个字符
varchar2	变长字符型，最大2000个字符
Long	变长字符型，最大2GB
Date	日期型
Boolean	布尔型（TRUE，FALSE，NULL）

复合数据类型变量

■ 1、使用%type定义变量

为了让PL/SQ中变量的类型和数据表中的字段的数据类型一致，Oracle9i提供了%type定义方法。这样当数据表的字段类型修改后，PL/SQL程序中相应变量的类型也自动修改。

- 例：定义名为mydate的变量，其类型和testtable数据表中的currentdate字段类型是一致的。

Declare

```
mydate testtable.currentdate%type;
```

Begin

```
commit;
```

End;

■ 2、定义记录类型变量

在结构化程序中，都提供了记录类型的数据类型，在PL/SQL中，也支持将多个基本数据类型捆绑在一起的记录数据类型。

例：定义名为myrecord的记录类型，该记录类型由整数型的myrecordnumber和日期型的mycurrentdate基本类型变量组成，srecord是该类型的变量。引用记录型变量的方法是“记录变量名.基本类型变量名”。

程序的执行部分从testtable数据表中提取recordnumber字段为68的记录的内容，存放在srecord复合变量里，然后输出srecord.mycurrentdate的值，实际上就是数据表中相应记录的currentdate的值。

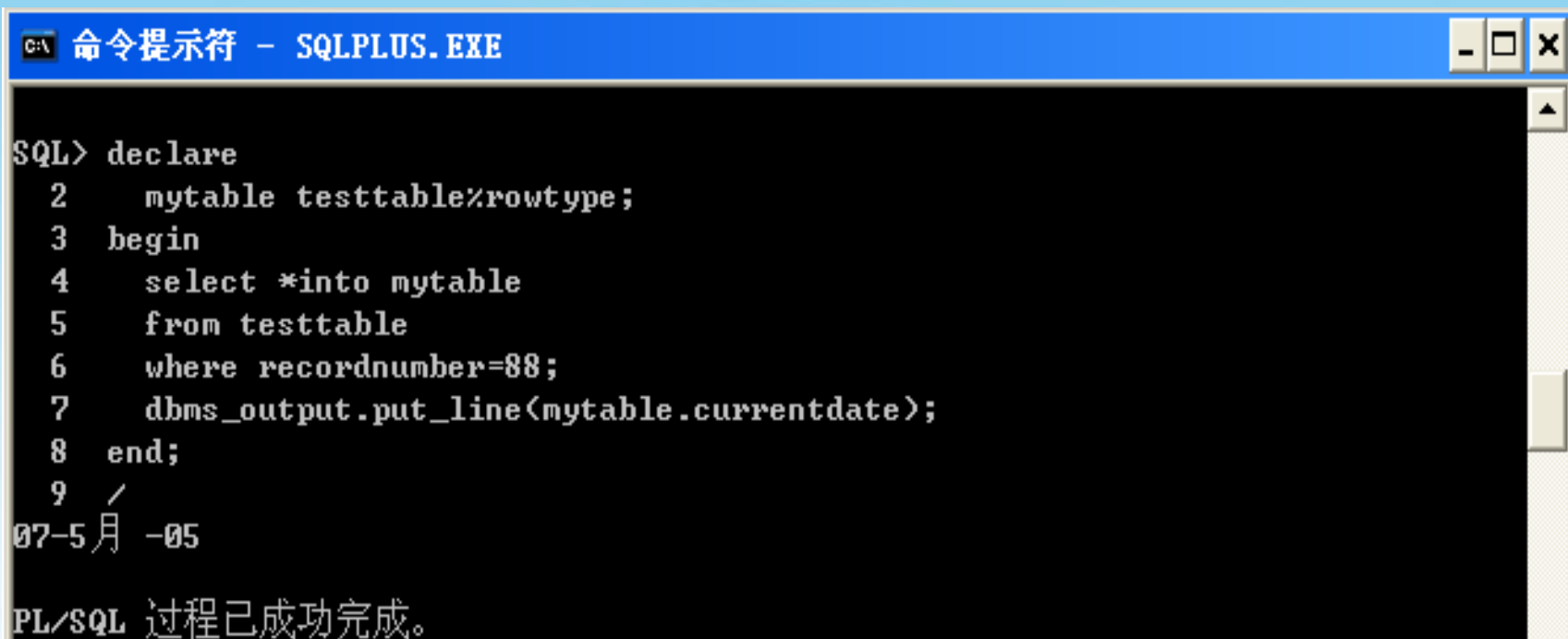
```
命令提示符 - SQLPLUS.EXE
SQL> set serveroutput on
SQL> declare
  2   type myrecord is record(
  3       myrecordnumber int,
  4       mycurrentdate date);
  5   srecord myrecord;
  6   begin
  7       select * into srecord from testtable where recordnumber=68;
  8       dbms_output.put_line(srecord.mycurrentdate);
  9   end;
10 /
07-5月 -05
```

在PL/SQL程序中，select语句总是和into配合使用的，into子句后面就是要被赋值的变量。

■ 3. 使用%rowtype定义变量

使用%type可以使变量获得字段的数据类型，使用%rowtype可以使变量获得整个记录的数据类型。比较两者定义的不同： 变量名 数据表.列名%type， 变量名 数据表%rowtype。

- 例：定义名为mytable的复合类型变量，与testtable数据表结构相同。



```
命令提示符 - SQLPLUS.EXE

SQL> declare
  2   mytable testtable%rowtype;
  3 begin
  4   select *into mytable
  5   from testtable
  6   where recordnumber=88;
  7   dbms_output.put_line(mytable.currentdate);
  8 end;
  9 /
07-5月 -05

PL/SQL 过程已成功完成。
```

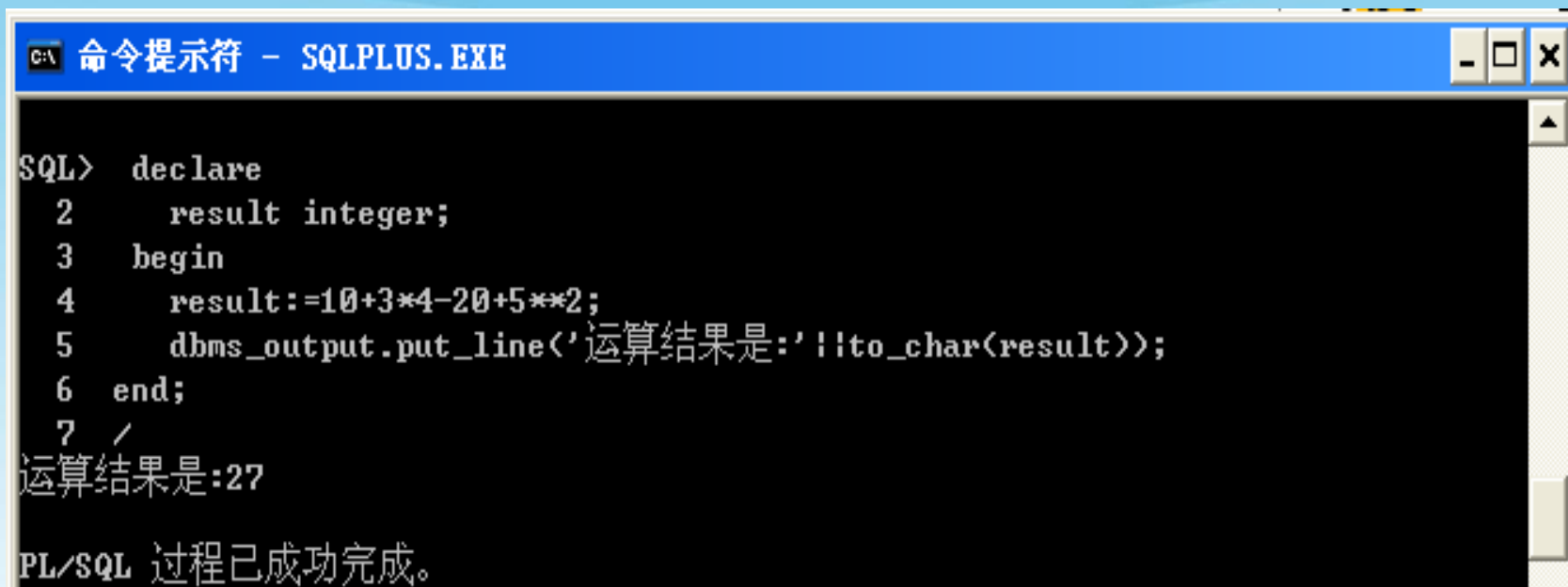
表达式

- 变量、常量经常需要组成各种表达式来进行运算，下面介绍PL/SQL中常见表达式的运算规则。

1、数值表达式

PL/SQL程序中的数值表达式是由数值型常数、变量、函数和算术运算符组成的，可以使用的算术运算符包括+（加法）、-（减法）、*（乘法）、/（除法）和**（乘方）等。

- 例，定义名为result的整数型变量，计算的是 $10+3*4-20+5**2$ 的值，理论结果应该是27。



```
命令提示符 - SQLPLUS.EXE

SQL> declare
  2     result integer;
  3  begin
  4     result:=10+3*4-20+5**2;
  5     dbms_output.put_line('运算结果是:'||to_char(result));
  6  end;
  7  /
运算结果是:27

PL/SQL 过程已成功完成。
```

Dbms_output.put_line函数输出只能是字符串，因此利用To_char函数将数值型结果转换为字符型。

■ 2、字符表达式

字符表达式由字符型、变量、函数和字符运算符组成，惟一可以使用的字符运算符就是连接运算符“||”。

■ 3、关系表达式

关系表达式由字符表达式或数值表达式与关系运算符组成，关系型表达式运算符两边的表达式的数据类型必须一致。可以使用的关系运算符包括以下9种。

- =等于（不是赋值运算符:=）
- like类似于
- in在. . . 之中
- <=小于等于
- >=大于等于
- !=不等于
- Between在. . . 之间

■ 4、逻辑表达式

逻辑表达式由逻辑常数、变量、函数和逻辑运算符组成，常见的逻辑运算符包括以下3种：

NOT：逻辑非

OR：逻辑或

AND：逻辑与

■ 5、重要的几个函数

PL/SQL程序 中提供了很多函数供扩展功能，除了标准SQL语言的函数可以使用外，最常见的数据类型转换函数有以下3个。

To_char:将其他类型数据转换为字符型.

To_date:将其他类型的数据转换为日期型.

To_number:将其他类型数据转换为数值型.

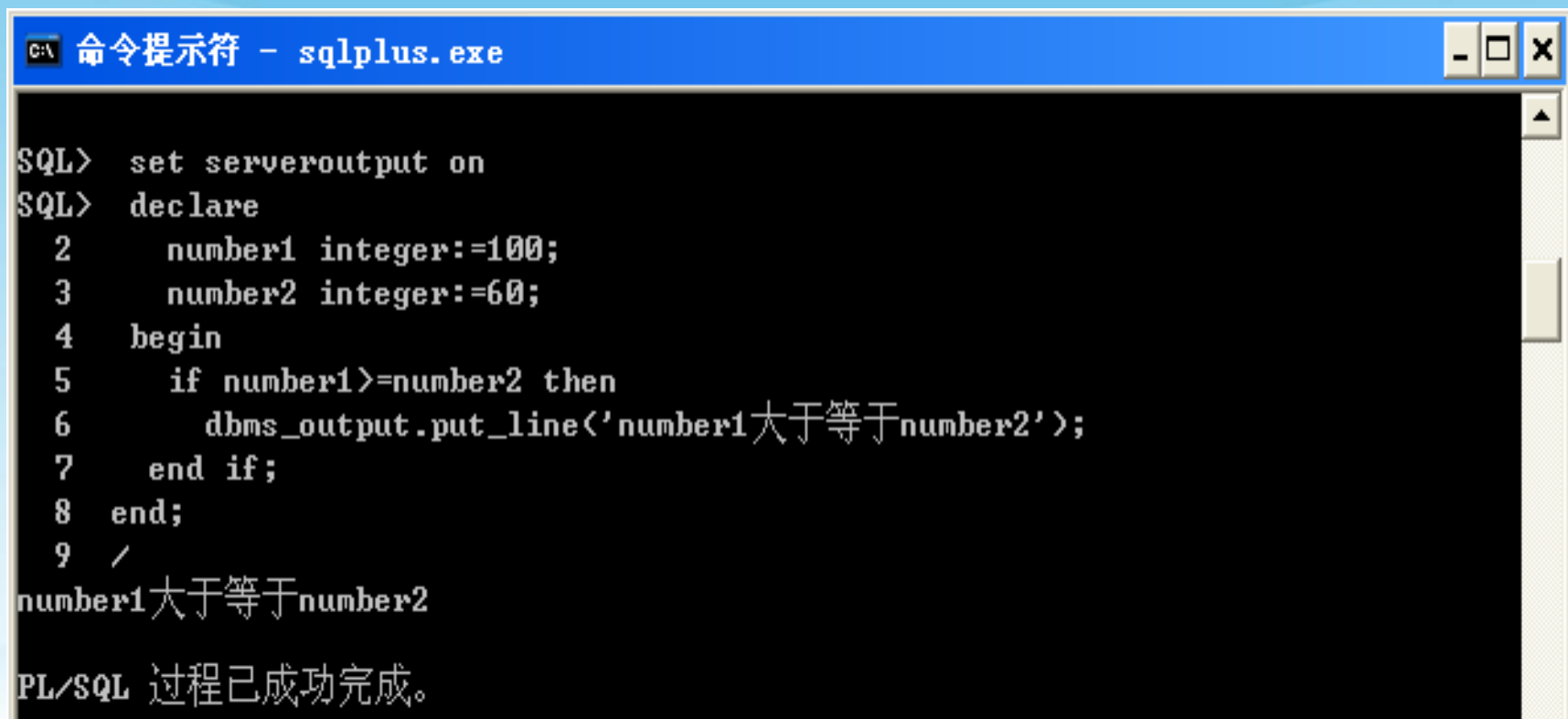
条件控制语句

- 1、if..then..end if 条件控制

采用if..then..end if条件控制的语法结构如图：

```
If 条件 then  
    语句段;  
end if;
```

- 实例：利用if..then..endif条件控制语句判断两个整数变量的大小。



```
命令提示符 - sqlplus.exe

SQL> set serveroutput on
SQL> declare
2     number1 integer:=100;
3     number2 integer:=60;
4     begin
5         if number1>=number2 then
6             dbms_output.put_line('number1大于等于number2');
7         end if;
8     end;
9     /
number1大于等于number2

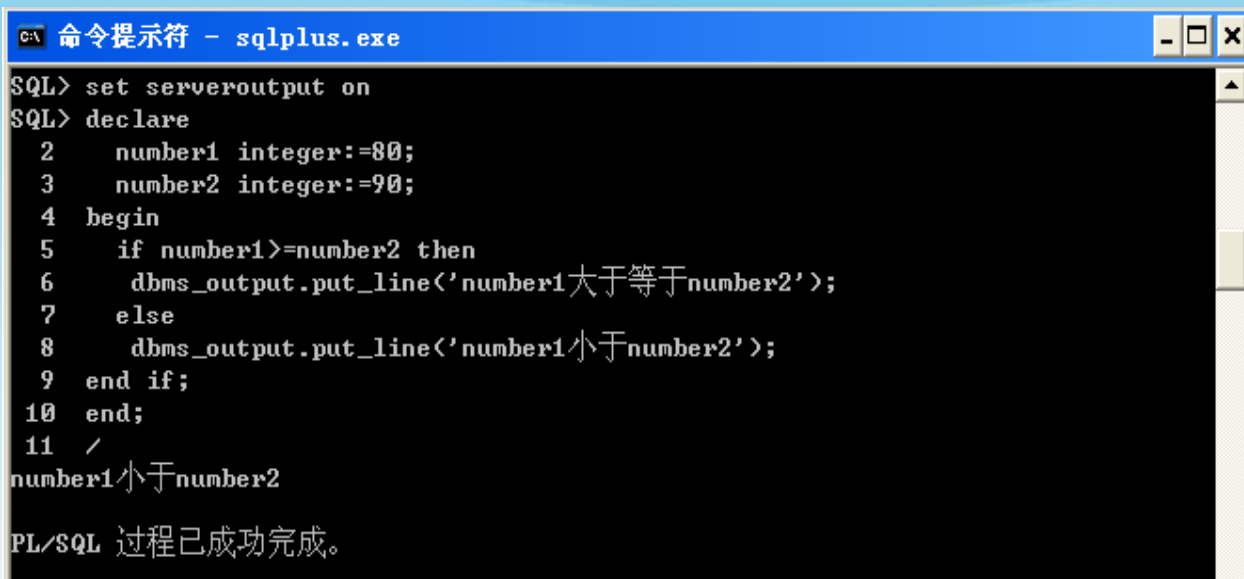
PL/SQL 过程已成功完成。
```

■ 2、if..then..else..end if条件控制

采用if..then..else..end if条件控制的语法结构如下：

```
if 条件 then  
    语句段 1 ;  
else  
    语句段 2 ;  
end if;
```

- 例：采用if..then..else..end if条件控制语句判断两个整数变量的大小，输出不同的结果。



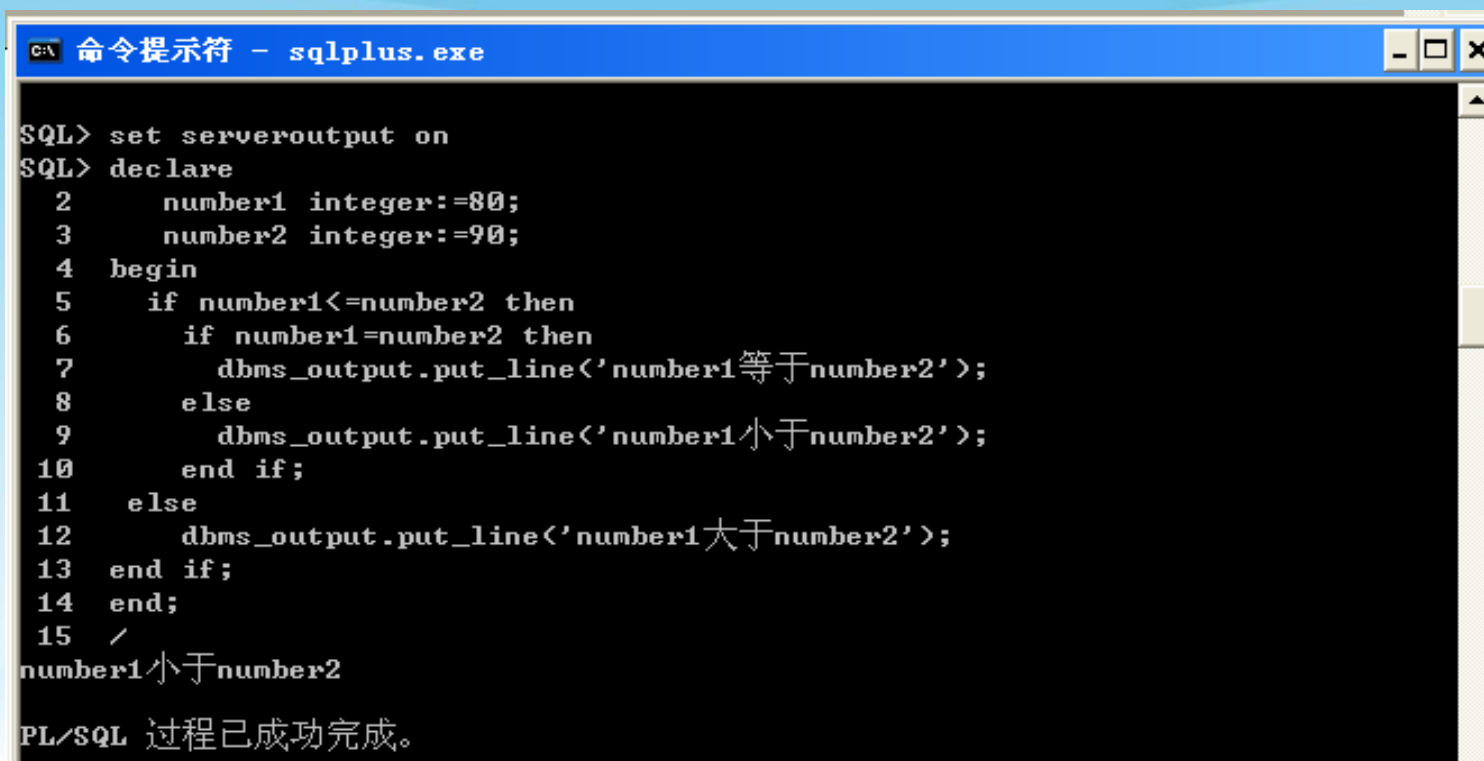
```
命令提示符 - sqlplus.exe
SQL> set serveroutput on
SQL> declare
2   number1 integer:=80;
3   number2 integer:=90;
4   begin
5   if number1>=number2 then
6   dbms_output.put_line('number1大于等于number2');
7   else
8   dbms_output.put_line('number1小于number2');
9   end if;
10  end;
11  /
number1小于number2
PL/SQL 过程已成功完成。
```


■ 3、if嵌套条件控制

采用if嵌套条件控制的语法结构如下：

```
if  条件1  then
    if  条件2  then
        语句段1;
    else
        语句段2;
    end if;
else
    语句段3;
end if;
```

- 例：采用if嵌套条件控制语句判断两个整数变量的大小，输出不同的结果。



```
命令提示符 - sqlplus.exe

SQL> set serveroutput on
SQL> declare
  2     number1 integer:=80;
  3     number2 integer:=90;
  4 begin
  5     if number1<=number2 then
  6         if number1=number2 then
  7             dbms_output.put_line('number1等于number2');
  8         else
  9             dbms_output.put_line('number1小于number2');
 10         end if;
 11     else
 12         dbms_output.put_line('number1大于number2');
 13     end if;
 14 end;
 15 /
number1小于number2

PL/SQL 过程已成功完成。
```

循环控制语句

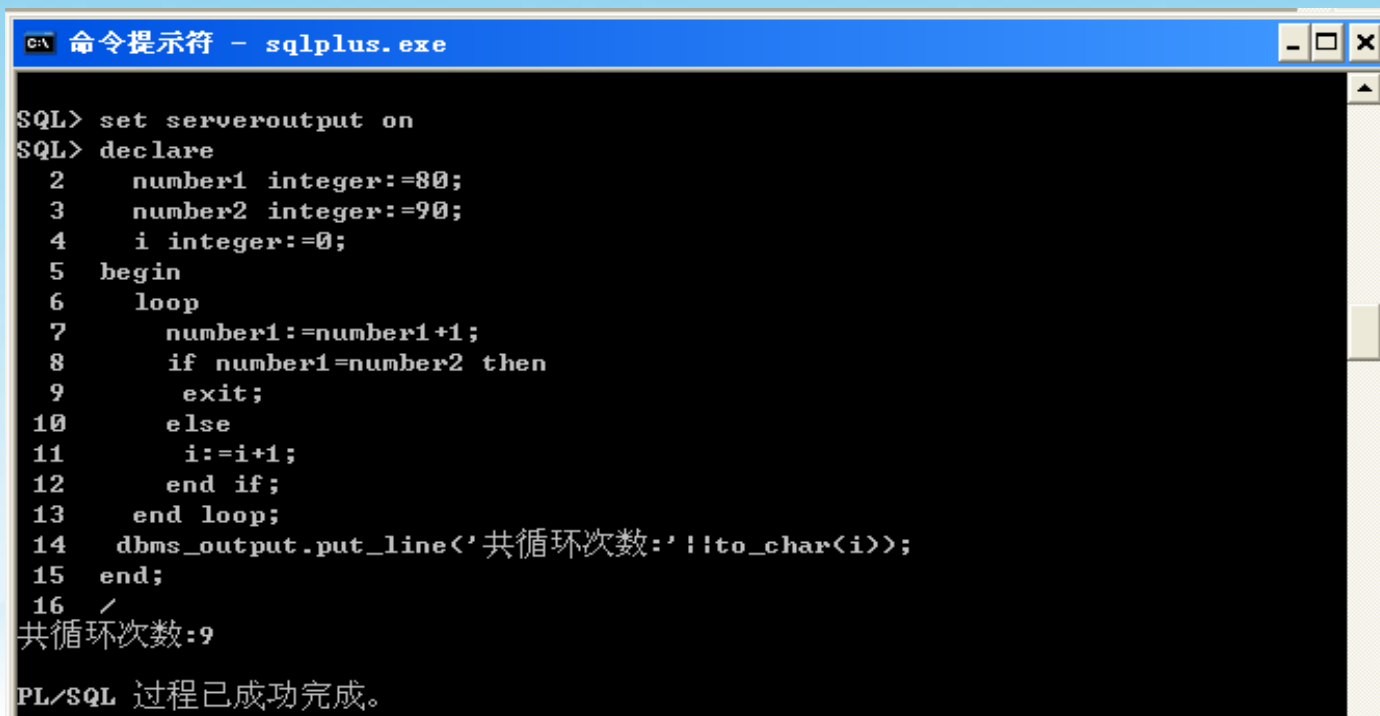
- 循环结构是按照一定逻辑条件执行一组命令，PL/SQL中可以有4种基本循环结构，在它们基础上又可以演变出许多嵌套循环控制，这里介绍最基本的循环控制语句。

■ 1、loop..exit..end loop循环控制

采用loop..exit..end loop循环控制的语法结构如下：

```
loop
    循环语句段;
    if 条件语句 then
        exit;
    else
        退出循环的处理语句段;
    end if;
end loop;
```

- 例：采用loop..exit..end loop循环控制语句将number1变量每次加1，一直到等于number2为止，统计输出循环次数。



```
C:\ 命令提示符 - sqlplus.exe

SQL> set serveroutput on
SQL> declare
  2   number1 integer:=80;
  3   number2 integer:=90;
  4   i integer:=0;
  5   begin
  6   loop
  7     number1:=number1+1;
  8     if number1=number2 then
  9       exit;
 10     else
 11       i:=i+1;
 12     end if;
 13   end loop;
 14   dbms_output.put_line('共循环次数:'||to_char(i));
 15 end;
 16 /
共循环次数:9

PL/SQL 过程已成功完成。
```

■ 2、loop..exit..when..end loop循环控制

采用loop..exit..when..end loop循环控制的语法结构与上图所示结构类似

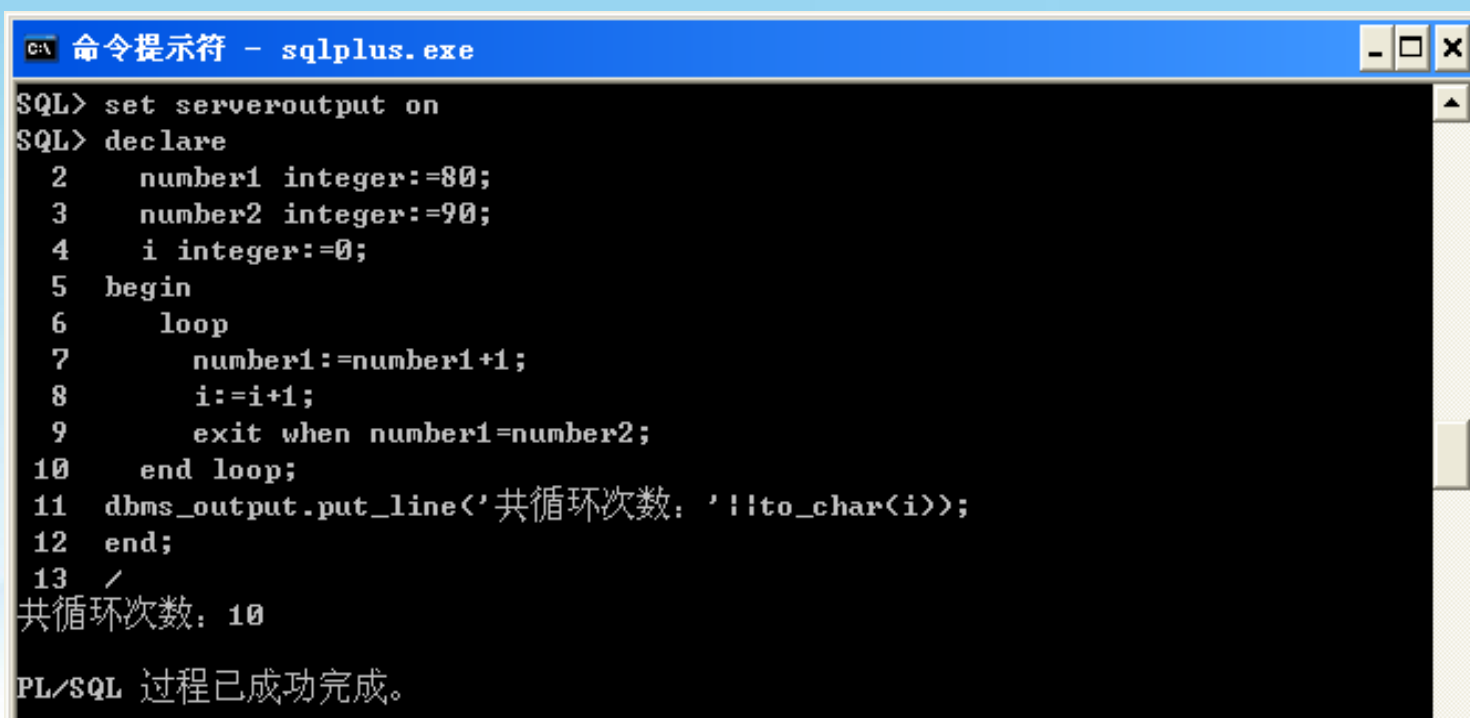
exit when实际上就相当于

if 条件 then

exit;

end if;

- 例：采用loop..exit..when..end loop循环控制语句将number1变量每次加1，一直到等于number2为止，统计输出循环次数。采用when循环控制结束条件比采用if条件控制结束条件循环次数多1次。



```
C:\> 命令提示符 - sqlplus.exe

SQL> set serveroutput on
SQL> declare
  2   number1 integer:=80;
  3   number2 integer:=90;
  4   i integer:=0;
  5   begin
  6     loop
  7       number1:=number1+1;
  8       i:=i+1;
  9       exit when number1=number2;
 10   end loop;
 11   dbms_output.put_line('共循环次数: '||to_char(i));
 12 end;
 13 /
共循环次数: 10

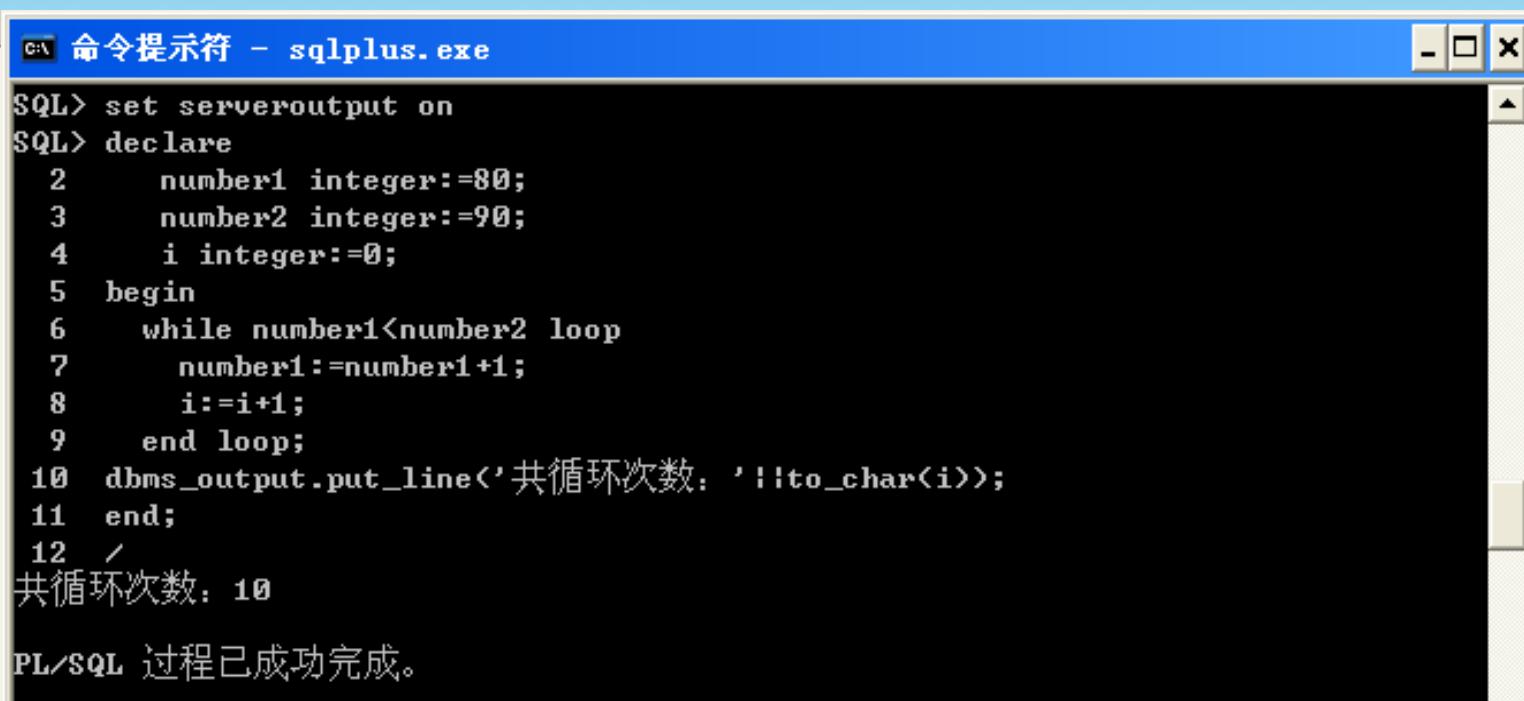
PL/SQL 过程已成功完成。
```


■ 3、while..loop..end loop循环控制

采用while..loop..end loop循环控制的语法如下。

```
while 条件 loop  
    执行语句段;  
end loop;
```

- 例：采用while..loop..end loop循环控制语句将number1变量每次加1，一直到等于number2为止，统计输出循环次数。



```
命令提示符 - sqlplus.exe
SQL> set serveroutput on
SQL> declare
  2   number1 integer:=80;
  3   number2 integer:=90;
  4   i integer:=0;
  5   begin
  6   while number1<number2 loop
  7     number1:=number1+1;
  8     i:=i+1;
  9   end loop;
 10   dbms_output.put_line('共循环次数: '||to_char(i));
 11 end;
 12 /
共循环次数: 10
PL/SQL 过程已成功完成。
```

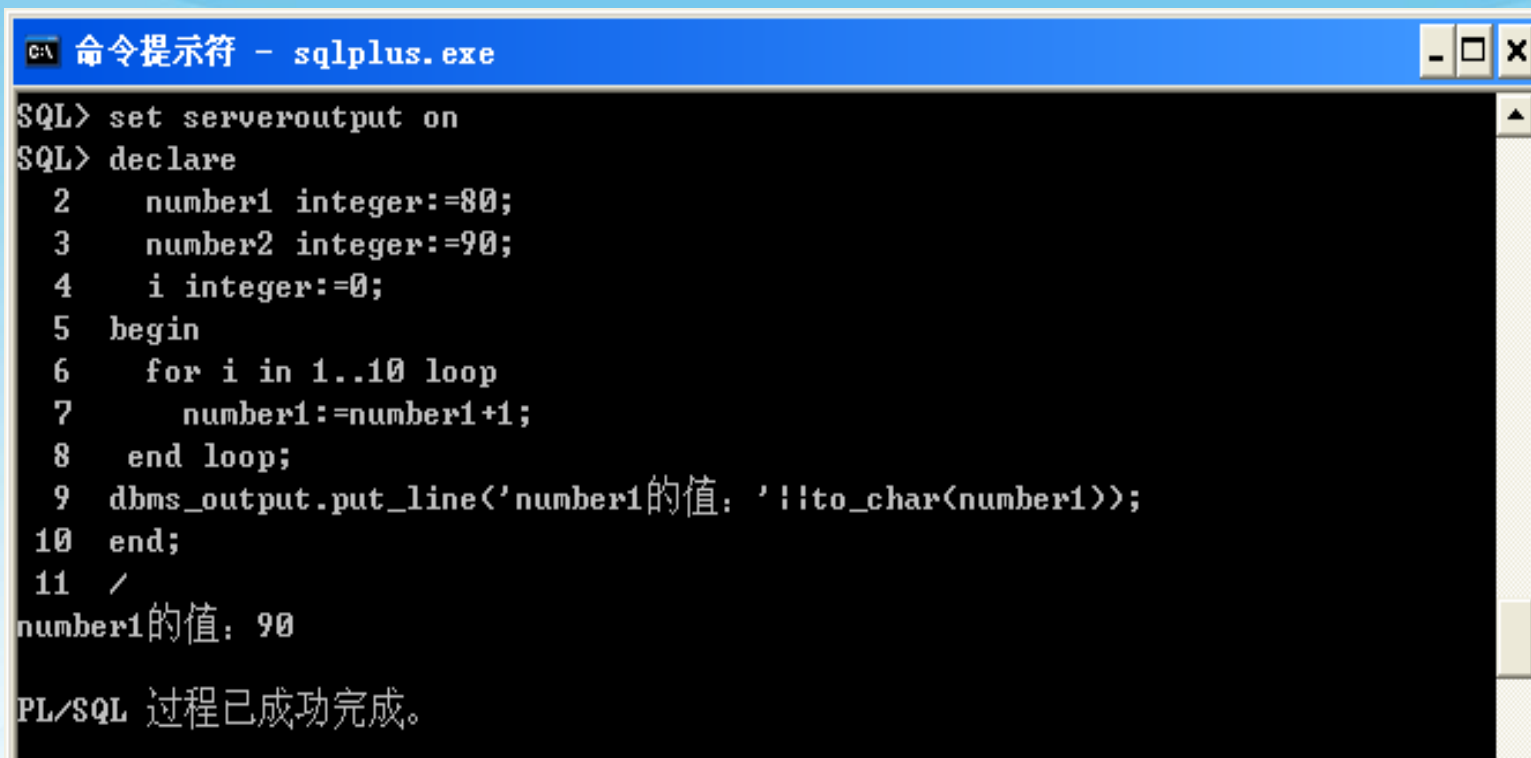
■ 4、for..in..loop..end循环控制

采用for..in..loop..end循环控制的语法如下：

```
for  循环变量  in [reverse] 循环下界..循环上界 loop  
    循环处理语句段;  
end loop;
```

Reverse选项强制循环变量从终止值开始，每次循环减1，直到起始值。

- 例：采用for..in..loop..end循环控制语句，通过循环变量i来控制number1增加次数并输出结果。



```
C:\ 命令提示符 - sqlplus.exe
SQL> set serveroutput on
SQL> declare
2   number1 integer:=80;
3   number2 integer:=90;
4   i integer:=0;
5   begin
6   for i in 1..10 loop
7       number1:=number1+1;
8   end loop;
9   dbms_output.put_line('number1的值: '||to_char(number1));
10  end;
11  /
number1的值: 90

PL/SQL 过程已成功完成。
```

游 标

- 游标是从数据表中提取出来的数据，以临时表的形式存放在内存中，在游标中有一个数据指针，在初始状态下指向的是首记录，利用fetch语句可以移动该指针，从而对游标中的数据进行各种操作，然后将操作结果写回数据表中。

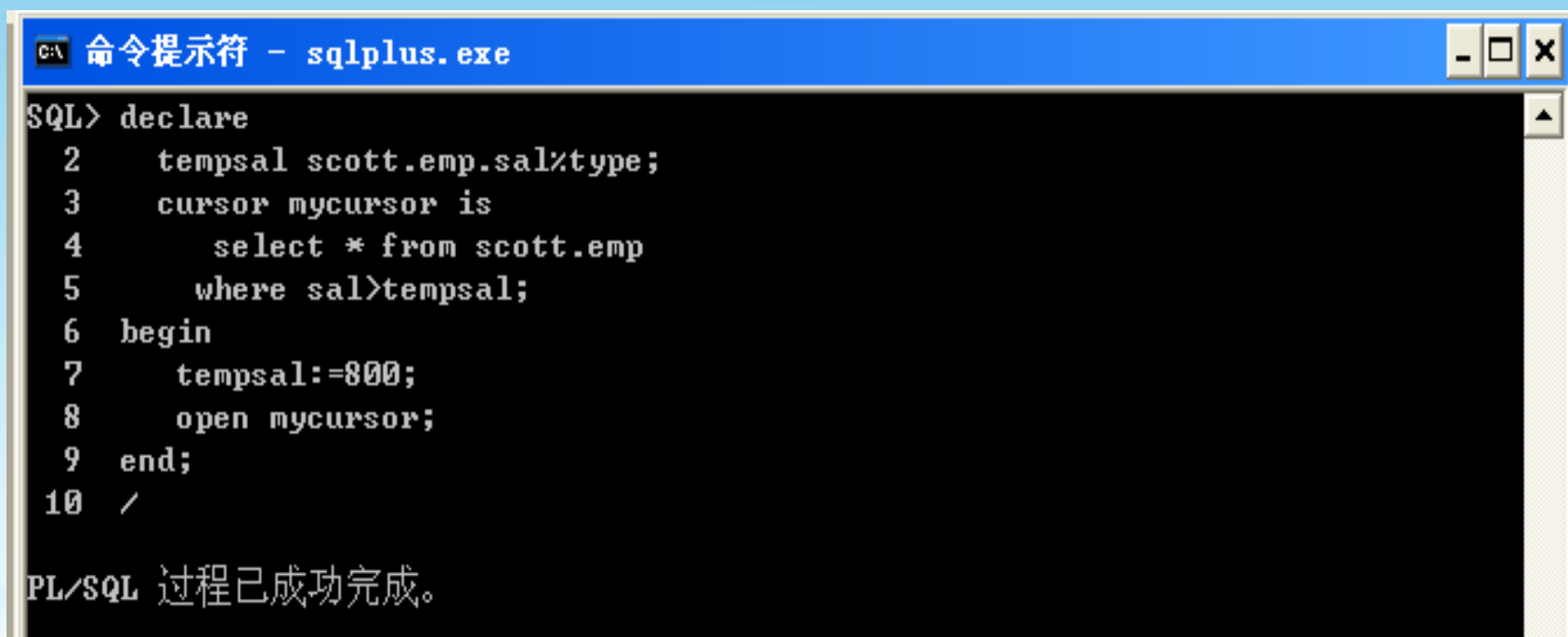
■ 1. 游标的定义

游标作为一种数据类型，首先必须进行定义，其语法如下。

`cursor` 游标名 `is select` 语句:

`cursor`是定义游标的关键字，`select`是建立游标的数据表查询命令。

- 例：定义tempсал为与scott.emp数据表中的sal字段类型相同的变量，mycursor为从scott.emp数据表中提取的sal的值大于tempсал的数据构成的游标。



```
命令提示符 - sqlplus.exe

SQL> declare
  2   tempсал scott.emp.sal%type;
  3   cursor mycursor is
  4       select * from scott.emp
  5       where sal>tempсал;
  6   begin
  7       tempсал:=800;
  8       open mycursor;
  9   end;
10   /

PL/SQL 过程已成功完成。
```


■ 2、游标的打开

要使用创建好的游标就要打开游标，语法形式如下。

open 游标名；

打开游标的过程有以下两个步骤。

- 1)、将符合条件的记录送入内存。
- 2)、将指针指向第一条记录

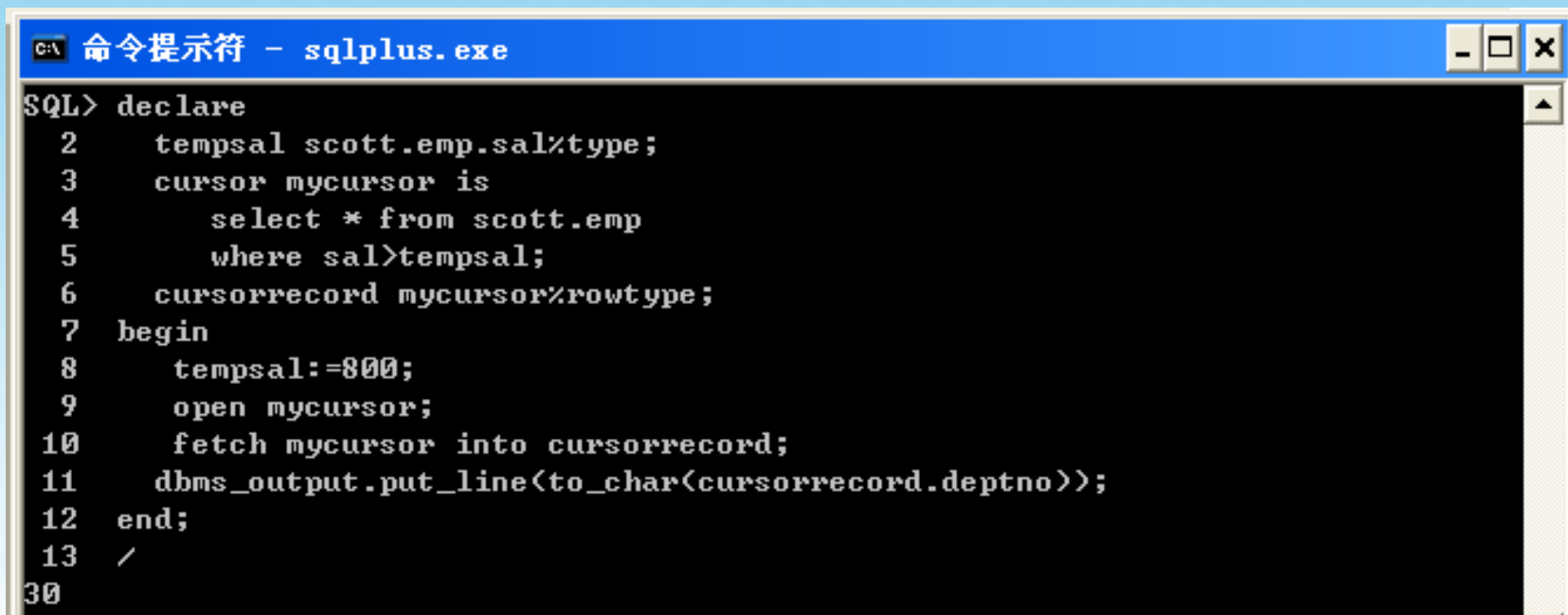
■ 3、游标数据的提取

要提取游标中的数据，需要使用fetch命令，语法形式如下。

fetch 游标名 into 变量名1, 变量名2, ……;
或

fetch 游标名 into 记录型变量名;

- 例：定义cursorrecord变量是游标mycursor的记录行变量，在游标mycursor的结果中找到sal字段大于800的第一个记录，显示deptno字段的内容。



```
SQL> declare
  2   tempsal scott.emp.sal%type;
  3   cursor mycursor is
  4     select * from scott.emp
  5     where sal>tempsal;
  6   cursorrecord mycursor%rowtype;
  7 begin
  8   tempsal:=800;
  9   open mycursor;
 10   fetch mycursor into cursorrecord;
 11   dbms_output.put_line(to_char(cursorrecord.deptno));
 12 end;
 13 /
30
```

■ 4、游标的关闭

使用完游标后，需要关闭游标，使用close命令，语法形式如下。

close 游标名

■ 5、游标的重要属性

游标提供的一些属性可以帮助编写PL/SQL程序，游标属性的使用方法为：游标名[属性]。

■ %isopen属性

属性功能：测试游标是否打开，如果没有打开游标就是用fetch语句并提示错误。 例：

```
declare
    tempsal emp.sal%tpye;
    cursor mycursor is select *from emp where sal>tempsal;
    cursorrecord mycursor%rowtpye;
Begin
    tempsal:=800;
    if mycursor%isopen then
        fetch mycursor into cursorrecord;
        dbms_output.putline(to_char(cursorrecord.deptno));
    else
        dbms_output.putline( '游标没有打开!' );
    end if;
end;
```

■ %FOUND

逻辑值，游标是否找到一条记录。如游标找到记录其值为True，反之为False. 例：

```
Declare
    tempsal emp.sal%type;
    cursor mycursor is select * from emp where sal>tempsal;
    cursorrecord mycursor%rowtype;
Begin
    tempsal:=800;
    open mycursor;
        fetch mycursor into cursorrecord;
    if mycursor%found then
        dbms_output.put_line(to_char(cursorrecord.deptno));
    else
        dbms_output.put_line('没有数据');
    endif;
End;
```


- **%NOTFOUND**

逻辑值，游标没有找到记录，是%FOUND属性的逻辑非。

■ %ROWCOUNT

返回提取游标记录的行数。例：

```
declare
    tempsal emp.sal%type;
    cursor mycursor is
        select * from emp where sal>tempsal;
    cursorrecord mycursor%rowtype;
begin
    tempsal:=800;
    open mycursor;
    fetch mycursor into cursorrecord;
    dbms_output.putline(to_char(mycursor%rowcount));
End;
```

若返回值为0，表明游标已经打开，但没有提取出数据。

异常处理

- 在设计PL/SQL程序时，经常会发生这样或那样的错误，异常处理就是针对错误进行处理的程序段，Oracle9i中的异常处理分为系统定义异常处理和自定义异常处理两部分。

■ 系统定义的异常

ACCESS_INTO_NULL	ORA-06530	-6530
CASE_NOT_FOUND	ORA-06592	-6592
COLLECTION_IS_NULL	ORA-06531	-6531
CURSOR_ALREADY_OPEN	ORA-06511	-6511
DUP_VAL_ON_INDEX	ORA-00001	-1
INVALID_CURSOR	ORA-01001	-1001
INVALID_NUMBER	ORA-01722	-1722
LOGIN_DENIED	ORA-01017	-1017
NO_DATA_FOUND	ORA-01403	+100
NOT_LOGGED_ON	ORA-01012	-1012
PROGRAM_ERROR	ORA-06501	-6501
ROWTYPE_MISMATCH	ORA-06504	-6504
SELF_IS_NULL	ORA-30625	-30625
STORAGE_ERROR	ORA-06500	-6500
SUBSCRIPT_BEYOND_COUNT	ORA-06533	-6533
SUBSCRIPT_OUTSIDE_LIMIT	ORA-06532	-6532
SYS_INVALID_ROWID	ORA-01410	-1410
TIMEOUT_ON_RESOURCE	ORA-00051	-51
TOO_MANY_ROWS	ORA-01422	-1422
VALUE_ERROR	ORA-06502	-6502
ZERO_DIVIDE	ORA-01476	-1476

■ 2、用户定义异常处理

异常不一定必须是Oracle返回的系统错误，用户可以在自己的应用程序中创建可触发及可处理的异常。

语法

declare

异常名 exception;

触发异常处理的语法是：

raise 异常名;



- 触发异常处理后，可以定义异常处理部分，语法如下：

Exception

when异常名1 then

异常处理语句段1;

when 异常名2 then

异常处理语句段2;

- 例：下面的PL/SQL程序包含了完整的异常处理定义、触发、处理的过程。定义名为salaryerror的异常名，在emp数据表中查找empno=8099的记录，将其值放入变量tempsal中，判断tempsal值若不在1000—2600之间，说明该员工的薪水有问题，将激活异常处理提示信息。


```
■ Declare
    salaryerror exception;
    tempsal emp.sal%type;
begin
    select sal into tempsal
        from emp
        where empno=8099;
    if tempsal<1000 or tempsal>2600 then
        raise salaryerror;
    end if;
exception
when salaryerror then
    dbms_output.put_line( '薪水超出范围' );
End;
```

过 程

- 过程也叫存储过程，是由SQL语句和PL/SQL语句组合在一起为执行某一个任务的一个可执行单位。我们在使用时可以调用，过程没有返回值。

无参数过程的创建

- 语法结构

```
Create or replace procedure 过程名 as  
    声明语句段;  
begin  
    执行语句段;  
exception  
    异常处理语句段;  
End;
```

```
Create procedure tempprocedure as
    tempdate hr.employees.hire_date%type;
Begin
    select hire_date
        into tempdate
    from hr.employees
        where employee_id=200;
    dbms_output.putline_line(to_char(tempdate));
End;
```

带参数过程的创建

■ 1、参数类型

在PL/SQL过程中，可以有3中类型的参数。

in 参数：读入参数，主程序向过程传递参数值。

Out 参数：读出参数，过程向主程序传递参数值。

in out参数：双向参数，过程与主程序双向交流数据。

■ 2、定义带参数过程的语法

```
create or replace procedure 过程名 (  
    参数1[in|out|in out]数据类型  
    [, 参数2[in|out|in out] 数据类型]  
    .....  
    [, 参数n[in|out|in out] 数据类型]  
)  
(is|as)  
    声明语句段;  
Begin  
    执行语句段;  
Exception  
    异常处理语句段;  
End;
```

```
Create or replace procedure tempprocedure(  
    tempdeptno in scott.dept.deptno%type,  
    tempdname out scott.dept.dname%type,  
    temploc in out scott.dept.loc%type) as  
    loc1 scott.dept.loc%type;  
    dname1 scott.dept.dname%type;  
Begin  
    select loc into loc1  
        from scott.dept  
where deptno=tempdeptno;  
    select dname into dname1  
        from scott.dept  
where deptno=tempdeptno;  
    temploc:= '地址:' || loc1;  
    tempdname:= '姓名' || dname1;  
End;
```


过程的执行

- 1、利用begin..end执行
begin
 过程名;
end;
- 2、利用execute执行
execute 过程名;

过程的调用

- 例：主程序调用带参数的temprocedure过程完成数据的查询输出。

```
Declare
    myno   scott.dept.deptno%type;
    mydname scott.dept.dname%type;
    myloc   scott.dept.loc%type;
Begin
    myno:=10;
    mydname:= ' ' ;
    myloc:= " ";
    temprocedure(myno, mydname, muloc);
    dbms_output.put_line(myno);
    dbms_output.put_line(mydname);
    dbms_output.put_line(myloc);
End;
```

过程的删除

- Drop procedure tempprocedure;

函 数

- 函数和过程的结构类似，也是由SQL语句和PL/SQL语句组合在一起，执行某一个任务的一个可执行单位。过程和函数的差别在于，函数总返回单个值给调用者，而过程没有值返回给调用者。

函数的创建

- Create or replace function 函数名 (
 参数1[in|out|in out] 数据类型
 [, 参数2[in|out|in out|] 数据类型

 [, 参数n[in|out|in out| 数据类型]
)
 return 数据类型
(is|as)
 声明语句段;
Begin
 执行语句段;
Exception
 异常处理语句段;
End;

例：创建函数tempfunction

```
Create function tempfunction (personname in varchar2)
return number
is
    tempid number(6,0);
Begin
    select employee_id
    into tempid
    from employees
    where first_name=personname;
    return tempid;
End;
```

函数的执行

```
set serveroutput on  
declare  
    name1 varchar2(10) := 'shelley' ;  
Begin  
    dbms_output.put_line(tempfunction(name1));  
End;
```

触发器

- 触发器是一种特殊类型的存储过程，由一些SQL语句组成，主要用于执行强制性的业务规则或要求，但不返回结果。当对数据库的操作触发了触发器的条件时，系统将自动执行触发器里设置的SQL语句，完成某些特定的功能。当数据表被修改时，与其相关的触发器隐式地被激发执行。对表的INSERT、UPDATE或DELETE操作都可以定义触发器。

触发器的结构

- 触发事件或语句:引起触发器被激发的SQL语句，是对指定表执行的INSERT、UPDATE或DELETE语句。
- 触发器限制: 一个布尔表达式，当触发器激发时该条件必须为TRUE。触发器的限制是用WHEN子句来指定。
- 触发器的动作: 一个PL/SQL块（过程），由SQL语句和PL/SQL语句组成。当触发语句发出，触发器的限制计算为TRUE时，它被执行。在触发器动作的语句中，可使用触发器的处理的当前行的列值（包括新值和旧值），使用的形式为：NEW. 列名（引用新值）或者OLD. 列名（引用旧值）。

触发器的分类

■ 1、按照激活触发的操作分类

激活触发的操作是指对数据表实行什么样的操作时激活触发器，分为3类：

INSERT触发器：对数据表插入数据时执行触发器动作。

UPDATE触发器：对数据表更新数据时执行触发器动作。

DELETE触发器：对数据表删除数据时执行触发器动作。

触发器的状态

- 一个触发器可以处于两种不同的状态：启用和停用状态。
- 启用状态：当触发语句发出，这种类型的触发器执行其触发动作。
- 停用状态：即使触发语句被发出，这种类型的触发器也不执行触发动作。

触发器的创建

- 可以使用CREATE TRIGGER来创建一个触发器。与过程不同，每个触发器都必须和某个特定的表相关，而且每一个触发器都只和作用于该表上的一个或多个动作有关。
- 可以使用任何想用的名称来命名触发器，但是每个触发器都必须由惟一的名称。建议在触发器的名称内包含与触发器相关联的表和动作。

- 触发器的创建

在企业管理器下面创建触发器。

- 触发器的修改

在企业管理器下面用鼠标右键点击触发器名称选择
‘查看/编辑详细资料

- 触发器的删除

在企业管理器下面用鼠标右键选择移去

■ 触发器的启用

```
ALTER TRIGGER “HR” .” TEMPTRIGGER”  
ENABLE
```

如果要启用数据表的所有触发器，可以使用下面的SQL代码。

```
ALTER TABLE “HR” .” EMPLOYEES”  
ENABLE ALL TRIGGERS;
```

■ 触发器的停用

触发器的停用可以执行下面的SQL语句来完成。

```
ALTER TRIGGER “HR” .” TEMPTRIGGER”  
DISABLE
```

安全性管理

Oracle9i的安全性机制

- Oracle数据库系统的安全性机制可以分为两种：系统安全性机制和数据安全性机制。
- 1、系统安全性机制

系统安全性机制是指在系统级控制数据库的存取和使用的机制。Oracle9i提供的系统安全性机制的作用包括以下几个方面：

- 1) 防止未授权的数据库存取：必须是有有效的用户名/口令的组合并授权的用户才能连接数据库。
- 2) 防止未授权的对方案对象的存取：对合法用户必须授予相应的方案对象的各种权限才能进行存取操作。
- 3) 控制对磁盘的使用：用户可用的表空间及分配的空间数量。
- 4) 控制对系统资源对使用：控制用户使用CPU时间、磁盘I/o等。
- 5) 对用户的动作进行审计：将用户对数据库实施的操作记录下来供管理员分析使用。

■ 2、数据安全性机制

数据安全性机制是指在对象级控制数据库的存取和使用的机制，Oracle9i提供的数据库安全性机制的作用包括以下几个方面。

- 1) 哪些用户可存取一指定的方案对象。
- 2) 在对象上允许进行哪些操作。

用户的管理

- Oracle9i数据库提供了一组合法的用户，要访问Oracle9i数据库，必须以用户名和口令登录，经过数据库服务器验证后，才能按照系统管理员给用户的授权访问数据库对象。

Oracle9i默认的用户

用户名	口令	登录身份及说明
sys	change_on_install	SYSDBA或SYSOPER ,但不能以normal登录，可作为默认系统管理员
system	manager	SYSDBA或NORMAL ,但不能以SYSORER登录，可作为默认的系统管理员
scott	tiger	NORMAL 普通用户
Aqadm	aqadm	SYSDBA或NORMAL ，高级队列管理员
Dbsnmp	dbsnmp	SYSDBA或NORMAL ，复制管理员

通过查询数据字典视图SYS.DBA_USERS和SYS.USER_USERS可以获得有关数据库的用户信息，包括用户名、加密后的口令、表空间、配置文件名、创建日期和锁定状态等。

系统权限配置

■ 1、什么是系统权限

在Oracle数据库中，权限用于控制对数据的访问并限制用户可以执行的操作。使用适当的权限，用户可以创建、修改和删除特定的数据库对象和能够访问的数据。有两种分配权限的方法：

直接给用户授予权限。

将权限授予角色，然后将角色分配给用户。

- Oracle中的权限分为系统权限和对象权限两类。所谓系统权限是指用户可以执行某个操作的权限，它与对象权限最大的区别是在于系统权限不属于某个具体的方案对象。Oracle提供有135种不同的系统权限，每一种系统权限允许用户执行一种特殊的数据库操作或一类数据库操作。系统权限一般被授给数据库管理人员和应用开发人员，终端用户不需要这些相关功能。

- 2、授予系统权限

授予用户系统权限的语法结构为：

GRANT 系统权限名 TO 用户名

- 3、撤销系统权限

REVOKE 系统权限 FROM 用户名

- 4、查询系统权限

在SYS.DBA_SYS_PRIVS视图中列出了已经授予给所有用户的系统权限。

对象权限

■ 1、什么是对象权限

对象权限是对特定的方案对象进行操作的权限。管理员可以为方案对象加上读、写、修改、删除、添加或引用等对象权限。

- 2、授予对象权限

例：将HR. EMPLOYEES数据表的对象权限SELECT授予用户TEMPUSER。

- 3、撤销对象权限

如果需要撤销已经授予用户的对象权限，其语法结构为：

REVOKE 对象权限 ON 方案对象名 FROM 用户名

角色的管理

- 如果数据库有很多用户，且这些用户的权限各不相同，那么单独授权给每个用户的话，不便于集中管理，当权限变化，管理员可能需要逐个修改用户的权限，非常繁琐。
- 角色就是对权限的集中管理机制，当若干个用户都被赋予同一个角色时，它们都继承了该角色拥有的权限，若角色的权限变更了，这些相关的用户权限都会发生变更。因此，角色可以方便管理员对用户权限的集中管理。角色不属于任何用户，它只属于数据库。

- 角色的创建
在企业管理器中创建

概要文件

- Oracle9i系统为了合理分配和使用系统的资源提出了概要文件的概念。所谓概要文件，就是一份描述如何使用系统的资源（主要是CPU资源）的配置文件。将概要文件赋予某个数据库用户，在用户连接并访问数据库服务器时，系统就按照概要文件给他分配资源。

审计

- 前面介绍的用户的识别和口令的验证，系统权限和对象权限的分级设定，采用资源配置文件和视图等技术屏蔽数据表等措施都属于强制性的安全性机制，目的是将用户对数据库的操作局限在其许可的范围之内。如何系统的安全性都是相对的，Oracle数据库系统也不例外。对数据的安全性要求较高的需求，Oracle提供了通过审计追踪来记录用户对数据库实施的某些操作。利用在数据库中记录的审计信息，管理员可以分析出数据库的安全性以及有那些非法用户执行了对数据库的操作，从而采取积极的应对措施。

- 审计对于数据库犹如黑匣子对于飞机，航空专家们通过对记录的飞机的飞行数据进行分析来帮助查找事故的原因，管理员也可以通过对审计信息的分析获取数据库系统的非法操作的来源。

审计的作用

- 审查可疑活动

当管理员发现数据库有可能被非法操作后，比如数据被非授权用户所删除，就可以决定对该数据库的所有连接进行审计，以及对数据库的所有表的成功或不成功的删除进行审计。这样可以发现非法用户的来源、使用的终端、会话的时间等。

- 监视和收集关于指定数据库活动的数据库

管理员可以收集哪些数据被修改、执行了多少次逻辑的I/O等统计数据。

审计的类型

- 语句审计

对某种类型的SQL语句进行审计，不指定结构和对象。比如对SCOTT用户执行的所有SELECT语句进行审计（AUDIT SELECT BY SCOTT）

- 权限审计

对执行相应活动的系统权限的审计。比如审计哪些用户使用了CREATE TRIGGER系统权限（AUDIT CREATE TRIGGER）

- 对象审计

对特定的方案对象上的指定语句的审计，实际上就是对对象权限的审计。比如审计对HR. EMPLOYEES数据表执行的SELECT操作的审计（AUDIT SELECT ON HR. EMPLOYEES）。

审计的启动

- Oracle数据库在安装时并没有激活审计功能，因此无法进行审计，管理员需要手工更改一个名为“AUDIT_TRIAL”的初始化参数值来激活审计功能。

结束！ 谢谢！