Oracle 数据库技术

实验任务书

序号 _____

学号 _____

姓名 _____

广州大学计算机学院

2017 年 09 月

前 言

关系数据库管理系统 Oracle 在各行各业都有广泛的应用，学好 Oracle 知识具有重要现实意义，编写《Oracle 数据库技术实验任务书》的目的在于指导学生如何完成实验内容，以掌握 Oracle 的相关技术。

《Oracle 数据库技术实验》课程共 16 课时，共分为 5 个实验（即索引基本操作与存储效率体验；PL /SQL 块中流程控制语句、游标、例外处理；触发器与存储过程基本操作；数据库备份与恢复；数据库应用系统设计与实现），其中，前 4 个实验每个占 2 课时，最后一个综合性实验占 8 课时，每个实验都必须由学生独立完成。

熊伟 綦科

2017 年 09 月 28 日

目录

1 实验前须知

1.1 实验环境介绍

1. **开机时进入装有 Oracle 的操作系统。目前在实验室安装的 Oracle 版本为 11g Express，Oracle 管理员 SYS 和 SYSTEM 的初始密码均为 oracle，若连接时密码有误或者用户被锁定，请参考附录“上机常见问题解决方法”。**

2. **实验中常用的客户端工具有 SQL PLUS 和 SQL DEVELOPER，若用 SQL PLUS 进行实验，可在开始菜单中找到相应菜单。由于在系统中没有安装 SQL DEVELOPER，要使用 SQL DEVELOPER 做实验的同学必须从服务器下载，解压存储到 E 盘（E 盘在重新启动不会还原），供以后上机使用。C 盘和 D 盘在系统重新启动后会自动还原，Oracle 数据库表的数据会丢失，所以上机时把实验时所写的代码保存下来，存放在 E 盘或自带的移动盘上，以便以后使用。**

3. **普通用户的数据库对象通常都存储在与用户对应的模式中，实验中建议同学首先利用管理员登录，然后利用 Create user 命令创建一个新用户，并授予新用户必要的权限，然后利用新建立的用户进行连接，在新创建的用户模式中创建数据库对象。以下代码创建 dblesson 用户（注意，使用 Oracle 12c 的同学不能直接使用下列代码，具体参阅附录中的代码说明）。**

   create user dblesson identified by d1234;/*创建新用户 dblesson，密码为 d1234*/
   grant connect,resource to dblesson;/*把 connect、resource 角色权限授予 dblesson*/
   上述两句可以简写成一句：grant connect, resource to dblesson identified by d1234;

1.2 客户端工具介绍

1.2.1 SQL Plus 的简要使用说明

1. **在操作系统命令模式 cmd 下利用 SQL Plus 登陆 Oracle 的基本语法；**

   SQL PLUS 用户名/密码@IP:port/服务名。

   (1) 示例一：利用 SYS 登陆

   SQLPLUS / AS SYSDBA;
   SQLPLUS sys/oracle as sysdba;

   (2) 示例二：利用 system 用户登录指定 ip 和端口的 Oracle 数据库

   SQLPlus system/oracle@192.168.0.20:1521/orcl;
   实验室登陆本机，可以简写为 SQLPlus system/oracle;

2. **在 SQL Plus 命令符下获取帮助**

   格式为：Help|? Command
   例如：? Set

3. **在 SQL Plus 命令符下切换连接的用户（disconnect 为断开连接）**

   conn a/a; --利用用户 a 进行连接

4. **List 列出缓冲区中的行，执行用/**

5. **利用 Describe 显示表结构**

   Desc student; --显示学生表的结构

6. **设置列的宽度**

   column tablespace format a20；注解：20 个字符宽

7. **设置行宽**

   Set linesize 200

8. **打开控制台输出**

   Set serveroutput on; /*关闭采用 OFF，下同*/

9. **显示当前时间**

   Set time on;

10. **显示代码执行时间**

    Set timing on;

11. **显示环境变量**

    Show all; --显示所有环境变量

    show user; --显示当前用户名

    Show autocommit; -- 是否自动提交

12. **使用替代变量示例**

    Select * from myproject where ptimespan=&x;--x 为临时替换变量。

13. **声明变量**

    Variable v_ptimespan number;

    Begin

        select ptimespan into :v_ptimespan from myproject where pno='p0001';

    End;

    Print v_ptimespan;

14. **常用 SQL Plus 命令如表 1.1 所示**

表 1.1 常用 SQL Plus 命令

SQL Plus 命令	缩写	意义
APPEND text	A text	把字符串增加到当前行的末尾
CHANGE /old/new/	C/old/new/	把当前行的旧字符串替换成新字符串
CHANGE /text/	C/text/	把当前行中字符串删除
CLEAR BUFFER	CL BUFF	从 SQL 缓冲区中删除所有行
DEL		删除当前行
INPUT	I	插入许多行
INPUT text	I text	插入一个包含 text 字符串的行
LIST	L	显示 SQL 缓冲区的所有行
LIST n	L n	显示 SQL 缓冲区中前 n 行
LIST m n	L m n	SQL 缓冲区中从第 m 行显示到第 n 行
RUN	R	显示并运行在缓冲区中当前 SQL 命令

SAVE filename		把 SQL 缓冲区中的内容保存到以 filename 为名字的文件中，默认路径为 orawin\bin
GET filename		把以 filename 为名字的文件内容调入 SQL 缓冲区中
START filename	@ filename	运行以前保存的命令文件
ED filename		用默认的编辑器编辑保存的文件内容
EXIT		退出 SQL Plus
RUNFORM filename		从 SQL Plus 中运行一个 Oracle Forms 应用程序
SPOOL filename		写所有的后面的命令或者输出到一个已经命名的文件中。假脱机输入输出文件的后缀为.LIS
SPO[OL] OFF|OUT		OFF 关闭假脱机输入输出文件；OUT 改变假脱机输入输出，送文件到打印机上
DESCRIBE tablename	DESC STUDENT	显示任何数据库表的数据结构
HELP		激活 Oracle 内部的帮助部件
HOST command		在 SQL Plus 中激活一个操作系统命令
CONNECT userid/password	CONN A/A	在当前的登录下，激活其它的 Oracle 用户
PROMPT text		当运行一个命令文件时，显示文本

1.2.2 SQL DEVELOPER 的简要使用说明

1. 使用 SQL DEVELOPER 连接远程 Oracle 服务器如图 1.1 所示

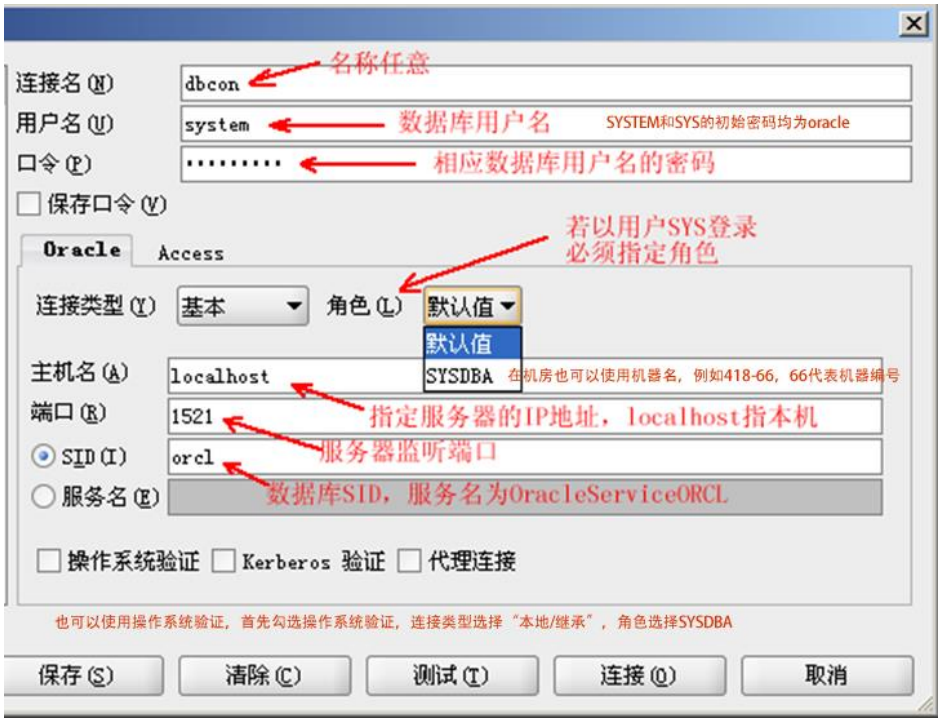

图 1.1 利用 SQL DEVELOPER 连接远程 Oracle 服务器

2. 用管理员 SYS 登录到本地 Oracle 数据库，可以采用“操作系统验证”，连接类型选择“本地/继承”，角色选择 SYSDBA。

1.3 Oracle 常用基本数据类型

Oracle 常用基本数据类型如表 1.2 所示。

表 1.2 Oracle 常用基本数据类型

类型	含义
CHAR(length)	存储固定长度的字符串。参数 length 指定了长度，如果存储的字符串长度小于 length，用空格填充。默认长度是 1，最长不超过 2000 字节。对应的国家字符类型为 NCHAR。
VARCHAR2(length)	存储可变长度的字符串。length 指定了该字符串的最大长度。默认长度是 1，最长不超过 4000 字符。对应的国家字符类型为 NVARCHAR2。
NUMBER(p，s)	既可以存储浮点数，也可以存储整数，p 表示数字的最大位数（如果是小数包括整数部分和小数部分和小数点，p 默认是 38 为），s 是指小数位数。
DATE	存储日期和时间，存储纪元、4 位年、月、日、时、分、秒，存储时间从公元前 4712 年 1 月 1 日到公元后 9999 年 12 月 31 日。
TIMESTAMP	不但存储日期的年月日，时分秒，以及秒后 6 位，同时包含时区。
CLOB	存储大的文本，比如存储非结构化的 XML 文档
BLOB	存储二进制对象，如图形、视频、声音等。

ANSI SQL Datatype 与 Oracle Datatype 之间的映射关系如表 1.3 所示。

表 1.3 ANSI SQL Datatype 与 Oracle Datatype 的映射关系

ANSI SQL Datatype	Oracle Datatype
CHARACTER(n) CHAR(n)	CHAR(n)
CHARACTER VARYING(n) CHAR VARYING(n)	VARCHAR(n)
NATIONAL CHARACTER(n) NATIONAL CHAR(n) NCHAR(n)	NCHAR(n)
NATIONAL CHARACTER VARYING(n) NATIONAL CHAR VARYING(n) NCHAR VARYING(n)	NVARCHAR2(n)
NUMERIC(p,s) DECIMAL(p,s)	NUMBER(p,s)
INTEGER INT SMALLINT	NUMBER(38)
FLOAT(b) DOUBLE PRECISION REAL	NUMBER

1.4 Oracle 常用函数

本节只给出了函数的简单使用方法，详细使用方法请参阅帮助文档。

LOWER（V）：将字母转成小写。

UPPER（V）：将字母转成大写。

INITCAP（V）：将 V 中各单词首写字母大写。其余小写。

CONCAT（V1，V2）：将 V1 与 V2 相连，相当于“||”。

SUBSTR（V1，N1）：在 V1 中从 N1 开始，取到字符串尾。从左向右取。N1 可为负数，表示从右向左取。

SUBSTR（V1，N1，N2）：从 V1 的 N1 开始取到 N2。

LENGTH（V1）：返回字符串的个数。

LENGTHB（V1）：返回字节数。

INSTR（V1，V2）：判断 V1 中第一次出现 V2 的位置。

INSTR（V1，V2，N1）：从 N1 处开始判断 V2 在 V1 中出现的位置。

INSTR（V1，V2，N1，N2）：从 N1 处开始 V2 在 V1 中第 N2 处出现的位置。

REPLACE（V1，V2）：将 V1 中的 V2 转成空（不是空格，是完全没有）。

REPLAE（V1，V2，V3）：将 V1 中的 V2 换成 V3。

TRIM（V1）：将 V1 两端空格去掉。

TRIM（V1 FROM V2）：从 V1 两端去掉 V2。

LTRIM（V1）：去掉左端的空格。

LTRIM（V1，V2）：去掉 V1 左端的 V2。

RTRIM（V1）：去掉右端的空格。

RTRIM（V1，V2）：去掉 V1 右端的 V2。

LPAD（V1，N1，V2）：用 V2 在 V1 左端补够 N1 位。

RPAD（V1，N1，V2）：用 V2 在 V1 右端补够 N1 位。

ROUND（N1，N2）：N1 四舍五入，保留 N2 位。若省略 N2，则对 N1 取整。

TRUNC（N1，N2）：将 N1 截取，保留 N2 位。省略 N2，四舍五入。

MOD（N1，N2）：返回 N1 除以 N2 的余数。若 N2 为 0,则返回 N1。

CEIL（N）：向上取整。返回大于等于 N 的最小整数。

FLOOR（N）：返回小于 N 的最大整数。

SYSDATE：返回数据库所在机器的系统时间。

MONTHS_BETWEEN(DATE1,DATE2)：两个日期间相差的月份数，要求 DATE1>=DATE2。

ADD_MONTHS（DATE，N1）：向 DATE 中加入 N1 个月后的日期。

LAST_DAY（DATE）：返回 DATE 所月份的最后一天的日期。

NEXT_DAY（DATE，V1）：返回 DATE 的下一个 V1 的日期。V1 在中文集下取值为“星期一~星期日”。

TO_CHAR：对日期和数字进行转换。

TO_CHAR（DATE，FORMATE）：格式参数可省略。

TO_NUMBER（V1，FORMATE）：TO_CHAR 的逆转换函数。

TO_DATE（V1，FORMATE）：也是 TO_CHAR 的逆转换函数。

NVL（P1，P2）：若 P1 为空（NULL），返回 P2。若不为空，则返回 P1。

NVL2（P1，P2，P3）：P1 为空，返回 P3，否则返回 P2。P2 和 P3 的类型必须相同。

NULLIF（P1，P2）：如果 P1=P2，返回 NULL，若不等于，则返回 P1。

COALESCE（P1，P2，P3...PN）：参数不定。返回若干参数中第一个不为空的参数值。参数中至少有一个不为空。

1.5 项目信息管理数据库

项目信息管理数据库涉及的实体有：(1)系别，其属性包括系编号、系名、系主任；(2)教师，其属性包括教工号、姓名、性别、工资、出生日期；(3)项目，其属性包括项目编号、项目名称、经费、开始日期、研究期限、负责人。各实体之间的联系有：

(1) 一个系可以有多个教师，每个教师属于一个系；

(2) 每个教师可以参与 0 到多个项目，每个项目可以被 0 到多个教师参与，某个教师参与某个项目用时间“工作天数”表示；

(3) 有些项目可以设置多个子项目，但每个项目至多有一个父项目；

(4) 每个系的系主任必须是已存在的某个教师；

(5) 每个项目的负责人必须是已存在的某个教师。

项目信息管理数据库的概念模型见图 1.3 所示。

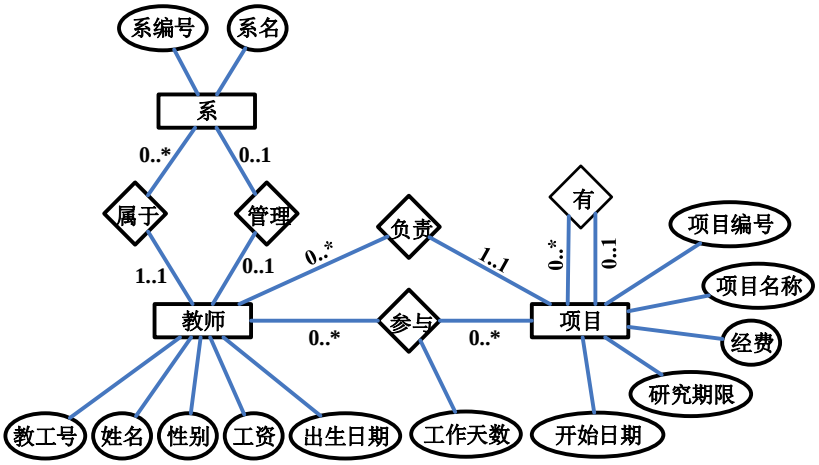

图 1.3 项目信息管理实体联系图

在实体联系图中，实体型“系”和“教师”之间存在两种联系，对于“属于”联系，“属于”联系与实体型“教师”之间的映射基数 1..1 表示一个教师参与到“属于”联系有且仅有 1 次，“属于”联系和实体型“系”之间的映射基数 0..*表示一个系参与到“属于”联系 0 到多次，其他映射基数含义类同（注意，此处的表示方法与教材第五版正好相反）。把实体联系图转换成关系模式时，“属于”联系转换成“教师表”的属性“DNO”，该属性的值并且不能为空，表示一个教师一定属于而且仅仅属于一个系，其他联系的转换描述在此不做详细介绍。

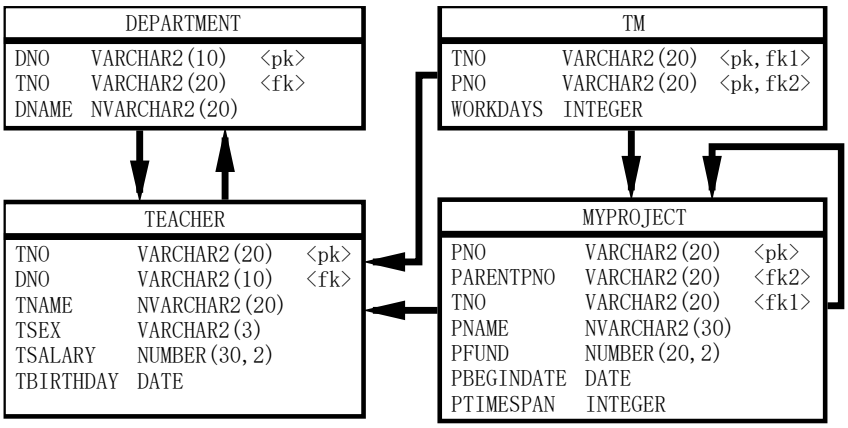

图 1.4 项目信息管理数据库逻辑结构图

图 1.4 为项目信息管理实体联系模型（属性添加数据类型后）等价转换成 Oracle 数据库的逻辑结构图，共有四个关系模式，每个关系模式的具体描述如下：

（1）系表。系表详细结构见表 1.7 所示。

表 1.7 DEPARTMENT（系表）

属性名	类型	长度	约束	含义
DNO	VARCHAR2	10	主码	系编号
DNAME	NVARCHAR2	20	非空	系名称
TNO	VARCHAR2	20	外码，引用 TEACHER(TNO)	系主任编号

（2）参与表。参与表详细结构见表 1.8 所示，其主码为(TNO,PNO)。

表 1.8 TM （参与表）

属性名	类型	长度	约束	含义
TNO	VARCHAR2	20	外码，引用 TEACHER(TNO)	教工号
PNO	VARCHAR2	20	外码，引用 MYPROJECT(PNO)	项目编号
WORKDAYS	INTEGER			工作天数

（3）教师表。教师表详细结构见表 1.9 所示。

表 1.9 TEACHER （教师表）

属性名	类型	长度	约束	含义
TNO	VARCHAR2	20	主码	教工号
DNO	VARCHAR2	10	外码，引用 DEPARTMENT(DNO)	系编号
TNAME	NVARCHAR2	20	非空	教师名称
TSEX	VARCHAR2	3	非空	教师性别
TSALARY	NUMBER(30,2)			教师工资
TBIRTHDAY	DATE		非空	出生日期

（4）项目表。项目表详细结构见表 1.10 所示。

表 1.10 MYPROJECT（项目表）

属性名	类型	长度	约束	含义
PNO	VARCHAR2	20	主码	项目编号
PARENTPNO	VARCHAR2	20	外码，引用 MYPROJECT(PNO)	父项目编号
TNO	VARCHAR2	20	非空，外码，引用 TEACHER(TNO)	教工号
PNAME	NVARCHAR2	30	非空	项目名称
PFUND	NUMBER(20,2)			经费

PBEGINDATE	DATE		非空	开始日期
PTIMESPAN	INTEGER		非空	研究期限

项目信息管理数据库的初始化代码在 3.2 节，实验前可以拷贝到 SQL Developer 中直接运行。

2 实验项目

2.1 实验一 索引基本操作与存储效率体验

预备知识

实验目的和要求

掌握索引创建、修改、使用和删除等操作，设计数据体验索引对查询效率的影响。实验内容要求独立完成。

实验环境

Oracle 11g

实验内容与步骤

本实验的基本内容是通过书写 SQL 的方式，掌握 B+树索引的创建、修改和删除等操作，通过查询性能比较，领会在大数据表中创建索引以提高查询性能的必要性。实验步骤如下：

1. 在基本表 department 中插入 20 个系；
2. 针对 department 表中的每个系，添加 100000 个以上的教师信息，其中教师名称不能重复（建议教师编号加上系编号，防止违反主码约束）；
3. 为教师表的 tname 列创建唯一索引；
4. 查阅帮助文档，修改索引，使刚创建的索引不可见；
5. 分别查询五个教师的姓名，并记录查询时间；
6. 修改索引，使创建的索引可见；
7. 再次分别查询五个教师的姓名，并记录查询时间，并与步骤 5 的记录进行对比分析；

实验总结

总结实验过程中涉及到的知识点、实验过程中遇到的问题及解决方法。

2.2 实验二 PL/SQL 块中流程控制语句、游标、例外处理

预备知识

实验目的和要求

掌握 PL/SQL 块中流程控制语句、游标、例外处理的使用方法。实验内容要求独立完成。

实验环境

Oracle 11g

实验内容与步骤

本实验的基本内容是通过编写 PL/SQL 块，掌握 PL/SQL 块的基本书写方法，具体实验步骤如下：

1. 编写 PL/SQL 块，由用户输入教师的姓名（假设为张三），利用 select…into 语句查询相应的教师的基本信息并输出，异常处理部分有：如果多个教师符合查询条件，输出“发现多个名称为张三的教师”；如果没有符合查询条件的教师信息，输出“不存在名称为张三教师”；其他异常输出异常的代码和文本；
2. 编写 PL/SQL 块，把所有教师的编号、姓名、负责的项目总经费等信息按行输出，如果某个教师没有负责任何项目，则该教师的项目总经费为 0（使用三种循环结构实现）；
3. 编写 PL/SQL 块，给“计算机科学系”的教师的工资提高 5%，给“软件工程系”的教师的工资提高 10%，其他系的所有教师的工资提高 8%；

实验总结

总结实验过程中涉及到的知识点、实验过程中遇到的问题及解决方法。

2.3 实验三 触发器与存储过程基本操作

预备知识

实验目的和要求

掌握触发器、存储过程、存储函数定义、使用和管理方法。实验内容要求独立完成。

实验环境

Oracle 11g

实验内容与步骤

本实验的基本内容是通过编写存储过程、函数和触发器，掌握 Oracle 数据库中存储过程、函数和触发器的定义、使用和管理方法。实验基本步骤如下：

1. 为 Teacher 表创建一个触发器，触发器名称为“学生名字首字母+UpdateTeacher”，当插入新教师时，如果新教师的工资少于 4000，拒绝插入数据；当工资高于 90000 时，把工资改为 9000；当删除教师时，输出被删除的教师基本信息。
2. 编写代码测试第 1 步创建的触发器；
3. 创建视图“学生名字首字母+teacherdept”，视图内容包含教师所有信息及其所在部门信息，即语句 SELECT dname, teacher.* from department, teacher Where department.dno = teacher.dno WITH CHECK OPTION。为该视图创建一个 instead of 触发器，当用户向该视图中插入数据时，数据被插入到 teacher 表中。
4. 编写代码测试第 3 步创建的触发器；
5. 创建一个存储过程，名称设为“学生名字首字母+sumDaysByTno”，存储过程有两个参数，输入参数为教师编号，输出参数为总的工作天数。
6. 编写代码测试第 5 步创建的存储过程；
7. 定义一个函数，名称设为“学生名字首字母+sumFundByTno”，参数为教师编号，返回该教师的所有项目的经费总和；
8. 编写代码测试第 7 步创建的函数；
9. 删除实验中创建的存储过程；
10. 删除实验中创建的函数；
11. 删除实验中创建的触发器。

实验总结

总结实验过程中涉及到的知识点、实验过程中遇到的问题及解决方法。

2.4 实验四 数据库备份与恢复

预备知识

数据库在运行过程中发生故障是不可避免的，例如，断电、死机、磁盘损坏等，都可能破坏数据库的一致性，如何保证在数据库发生故障的情况下，使数据库快速、有效的恢复到正确状态是数据库备份和恢复的主要内容。目前的商用数据库管理系统均提供了较为复制而完善的数据库备份和恢复机制，这些机制能保证在发生事务故障或系统故障的情况下，把数据库恢复到数据一致状态，当发生介质故障时，则需要数据库管理员进行数据库的恢复，因此，对管理员而言，合理制定数据库的备份和恢复策略尤其重要。

Oracle 数据库为数据库的备份与恢复提供了多种技术支持，包括多种备份与恢复工具、多种类型的备份与恢复操作。可以采用数据库恢复管理器 RMAN 进行数据库备份会恢复的管理，也可以采用用户管理方式进行数据库备份与恢复管理；可以基于操作系统文件进行物理备份与恢复，也可以采用数据泵技术进行数据导入、导出的逻辑备份与恢复；可以在数据库关闭状态下进行冷备份，也可以在数据库打开状态下进行热备份；可以进行数据库的完全恢复，也可以将数据库恢复到故障时刻之前的任意状态。

Oracle 逻辑备份是指利用 Oracle 提供的数据泵导出工具，将数据库中选定的记录集或数据字典的逻辑副本以二进制文件的形式存储到操作系统中。逻辑备份的二进制文件称为转储文件，以 dmp 格式存储。逻辑恢复是指利用 Oracle 提供的数据泵导入工具将逻辑备份形成的转储文件导入数据库内部，进行数据库的逻辑恢复。逻辑备份与恢复必须在数据库运行的状态下进行。因此，数据库备份与恢复是以物理备份与恢复为主，以逻辑备份与恢复为辅。

在 Oracle 11g 中，逻辑备份与恢复使用实用程序 Data Pump Export（Expdp）和 Data Pump Import（Impdp）实现，与传统的 EXP/IMP 客户端使用程序不同，EXPDP/IMPDP 是服务器端实用程序。

（1） 查看 EXPDP 帮助文档的命令：EXPDP help=y

（2） 查看 IMPDP 帮助文档的命令：IMPDP help=y

Expdp 介绍

数据泵导出实用程序提供了一种用于在 Oracle 数据库之间传输数据对象的机制。EXPDP 将数据库中的元数据与行数据导出到操作系统的转储文件中。

EXPDP 工具的执行方式有：（1）命令行方式：在命令行中直接指定参数设置；（2）参数文件方式：将参数设置存放到一个参数文件中，在命令行中用 PARFILE 参数指定参数文件；（3）交互方式：通过交互式命令进行导出作业管理。

EXPDP 导出模式决定了所要导出的内容范围。EXPDP 提供了以下 5 种导出模式：

（1） 数据库导出模式

通过参数 FULL 指定，导出整个数据库。用户需要具有 DATAPUMP_EXP_FULL_DATABASE 角色。

（2） 模式导出模式

通过参数 SCHEMAS 指定，是默认的导出模式，导出指定模式中的所有对象。如果用户没有 DATAPUMP_EXP_FULL_DATABASE 角色，则只能导出用户对应的模式。如果没有在导出模式列表中明确指明，参照的模式对象不会被导出。例如，一个作用在表上的触发器，当表所在模式被导出时，如果触发器所在的模式不在导出模式列表中，则该触发器不会被导出。

（3） 表导出模式

通过参数 TABLES 指定，导出指定模式中指定的所有表、分区及其依赖对象。如果指定了参数 TRANSPORTABLE=ALWAYS，则只有对象元数据被导出。可以通过复制数据文件来快速移动数据到目标数据库。如果在不同版本、不同操作系统之间移动数据文件，需要使用 RMAN 进行操作。如果要导出的表不属于当前用户模式，用户需要具有 DATAPUMP_EXP_FULL_DATABASE 角色。需要注意的是，表中列的数据类型的定义不会被导出，参照模式中的对象也不会被导出。

（4） 表空间导出模式

通过参数 TABLESPACES 指定，导出指定表空间中所有表及其依赖对象的元数据和行数据。

（5） 传输表空间导出模式

通过参数 TRANSPORT_TABLESPACES 指定，导出指定表空间中所有表及其依赖对象的元数据。表空间包含的数据文件需要单独进行复制。传输表空间导入的时候需要导入包含表空间元数据的转储文件，并指定相应的数据文件。表空间传输模式的导出一旦停止就无法重新开始，而且不能并行进行表空间传输模式的导出。此外，传输表空间的导出模式不支持加密列的导出。导出的传输表空间只能导入到相同或更高版本的数据库中。

EXPDP 实用程序可以使用以下命令进行调用：

示例: expdp scott/tiger DIRECTORY=dmpdir DUMPFILE=scott.dmp

您可以控制导出的运行方式。具体方法是: 在 'expdp' 命令后输入各种参数。要指定各参数, 请使用关键字：

格式: expdp KEYWORD=value 或 KEYWORD=(value1,value2,...,valueN)

示例: expdp scott/tiger DUMPFILE=scott.dmp DIRECTORY=dmpdir SCHEMAS=scott

或 TABLES=(T1:P1,T1:P2), 如果 T1 是分区表；USERID 必须是命令行中的第一个参数。

以下是可用关键字和它们的说明。方括号中列出的是默认值。

ATTACH：连接到现有作业。例如, ATTACH=job_name。

CLUSTER：利用集群资源并将 worker 进程分布在 Oracle RAC 上。有效的关键字值为: [Y] 和 N。

COMPRESSION：减少转储文件大小。有效的关键字值为: ALL, DATA_ONLY, [METADATA_ONLY] 和 NONE。

COMPRESSION_ALGORITHM：指定应使用的压缩算法。有效的关键字值为: [BASIC], LOW, MEDIUM 和 HIGH。

CONTENT：指定要卸载的数据。有效的关键字值为: [ALL], DATA_ONLY 和 METADATA_ONLY。

DATA_OPTIONS：数据层选项标记。有效的关键字值为: XML_CLOBS。

DIRECTORY：用于转储文件和日志文件的目录对象。

DUMPFILE：指定目标转储文件名的列表 [expdat.dmp]。例如, DUMPFILE=scott1.dmp, scott2.dmp, dmpdir:scott3.dmp。

ENCRYPTION：加密某个转储文件的一部分或全部。有效的关键字值为: ALL, DATA_ONLY, ENCRYPTED_COLUMNS_ONLY, METADATA_ONLY 和 NONE。

ENCRYPTION_ALGORITHM：指定加密的方式。有效的关键字值为: [AES128], AES192 和 AES256。

ENCRYPTION_MODE：生成加密密钥的方法。有效的关键字值为: DUAL, PASSWORD 和 [TRANSPARENT]。

ENCRYPTION_PASSWORD：用于在转储文件中创建加密数据的口令密钥。

ENCRYPTION_PWD_PROMPT：指定是否提示输入加密口令。当标准输入为读取时，将隐藏终端回送。

ESTIMATE：计算作业估计值。有效的关键字值为: [BLOCKS] 和 STATISTICS。

ESTIMATE_ONLY：计算作业估计值而不执行导出。

EXCLUDE：排除特定对象类型。例如, EXCLUDE=SCHEMA:"='HR'"。

FILESIZE：以字节为单位指定每个转储文件的大小。

FLASHBACK_SCN：用于重置会话快照的 SCN。

FLASHBACK_TIME：用于查找最接近的相应 SCN 值的时间。

FULL：导出整个数据库 [N]。

HELP：显示帮助消息 [N]。

INCLUDE：包括特定对象类型。例如, INCLUDE=TABLE_DATA。

JOB_NAME：要创建的导出作业的名称。

LOGFILE：指定日志文件名 [export.log]。

NETWORK_LINK：源系统的远程数据库链接的名称。

NOLOGFILE：不写入日志文件 [N]。

PARALLEL：更改当前作业的活动 worker 的数量。

PARFILE：指定参数文件名。

QUERY：用于导出表的子集的谓词子句。例如, QUERY=employees:"WHERE department_id > 10"。

REMAP_DATA：指定数据转换函数。例如, REMAP_DATA=EMP.EMPNO:REMAPPKG.EMPNO。

REUSE_DUMPFILES：覆盖目标转储文件 (如果文件存在) [N]。

SAMPLE：要导出的数据的百分比。

SCHEMAS：要导出的方案的列表 [登录方案]。

SERVICE_NAME：约束 Oracle RAC 资源的活动服务名和关联资源组。

SOURCE_EDITION：用于提取元数据的版本。

STATUS：监视作业状态的频率, 其中默认值 [0] 表示只要有新状态可用, 就立即显示新状态。

TABLES：标识要导出的表的列表。例如, TABLES=HR.EMPLOYEES,SH.SALES:SALES_1995。

TABLESPACES：标识要导出的表空间的列表。

TRANSPORTABLE：指定是否可以使用可传输方法。有效的关键字值为: ALWAYS 和 [NEVER]。

TRANSPORT_FULL_CHECK：验证所有表的存储段 [N]。

TRANSPORT_TABLESPACES：要从中卸载元数据的表空间的列表。

VERSION：要导出的对象版本。有效的关键字值为: [COMPATIBLE], LATEST 或任何有效的数据库版本。

VIEWS_AS_TABLES：标识要作为表导出的一个或多个视图。例如, VIEWS_AS_TABLES=HR.EMP_DETAILS_VIEW。

下列命令在交互模式下有效。

注: 允许使用缩写。

ADD_FILE：将转储文件添加到转储文件集。

CONTINUE_CLIENT：返回到事件记录模式。如果处于空闲状态, 将重新启动作业。

EXIT_CLIENT：退出客户机会话并使作业保持运行状态。

FILESIZE：用于后续 ADD_FILE 命令的默认文件大小 (字节)。

HELP：汇总交互命令。

KILL_JOB：分离并删除作业。

PARALLEL：更改当前作业的活动 worker 的数量。

REUSE_DUMPFILES：覆盖目标转储文件 (如果文件存在) [N]。

START_JOB：启动或恢复当前作业。有效的关键字值为: SKIP_CURRENT。

STATUS：监视作业状态的频率, 其中默认值 [0] 表示只要有新状态可用, 就立即显示新状态。

STOP_JOB：按顺序关闭作业执行并退出客户机。有效的关键字值为: IMMEDIATE。

使用 EXPDP 导出数据的一般步骤为：

（1） 使用管理员账户登录数据库，使用 CREATE DIRECTORY 语句创建对象。

CREATE OR REPLACE DIRECTORY MyDumpDir AS ‘D:\ORACLEDUMP’;

（2） 使用 GRANT 语句为用户授予目录对象的读写权限。

GRANT READ, WRITE ON DIRECTORY MyDumpDir TO oraclelesson;

（3） 可以在数据字典视图 DBA_DIRECTORIES 中查看所有的目录对象。

SELECT * FROM DBA_DIRECTORIES;

（4） 使用 EXPDP 导出数据，例如导出表：

EXPDP oraclelesson/pwd DIRECTORY= MyDumpDir DUMPFILE=LESSON.DMP LOGFILE=log.log TABLES=department,teacher JOB_NAME=EXP_TABLES PARALLEL=3

Impdp 介绍

IMPDP 是一个用于将转储文件导入目标数据库的工具，可以将转储文件导入到源数据库中，也可以导入到其它平台上运行的不同版本的 Oracle 数据库中。IMPDP 工具的执行也可以采用交互方式、命名行方式以及参数文件方式三种。

与 EXPDP 导出模式相对应，IMPDP 导入模式分为以下 5 种：

（1） 全库导入模式：通过参数 FULL 指定，将源数据库的所有元数据与行数据都导入目标数据库中。如果源数据库与目标数据库不是同一个数据库，用户需要具有 DATAPUMP_IMP_FULL_DATABASE 角色。

（2） 模式导入模式

（3） 表导入模式

（4） 表空间导入模式

（5） 传输表空间导入模式

该实用程序可以使用以下命令进行调用：

示例: impdp scott/tiger DIRECTORY=dmpdir DUMPFILE=scott.dmp

您可以控制导入的运行方式。具体方法是: 在 'impdp' 命令后输入各种参数。要指定各参数, 请使用关键字：

格式: impdp KEYWORD=value 或 KEYWORD=(value1,value2,...,valueN)

示例: impdp scott/tiger DIRECTORY=dmpdir DUMPFILE=scott.dmp

USERID 必须是命令行中的第一个参数。

以下是可用关键字和它们的说明。方括号中列出的是默认值。

ATTACH：连接到现有作业。例如, ATTACH=job_name。

CLUSTER：利用集群资源并将 worker 进程分布在 Oracle RAC 上。有效的关键字值为: [Y] 和 N。

CONTENT：指定要加载的数据。有效的关键字为: [ALL], DATA_ONLY 和

METADATA_ONLY。

DATA_OPTIONS：数据层选项标记。有效的关键字为: DISABLE_APPEND_HINT 和 SKIP_CONSTRAINT_ERRORS。

DIRECTORY：用于转储文件, 日志文件和 SQL 文件的目录对象。

DUMPFILE：要从中导入的转储文件的列表 [expdat.dmp]。

例如, DUMPFILE=scott1.dmp, scott2.dmp, dmpdir:scott3.dmp。

ENCRYPTION_PASSWORD：用于访问转储文件中的加密数据的口令密钥。对于网络导入作业无效。

ENCRYPTION_PWD_PROMPT：指定是否提示输入加密口令。当标准输入为读取时, 将隐藏终端回送。

ESTIMATE：计算作业估计值。有效的关键字为: [BLOCKS] 和 STATISTICS。

EXCLUDE：排除特定对象类型。例如, EXCLUDE=SCHEMA:"='HR'"。

FLASHBACK_SCN：用于重置会话快照的 SCN。

FLASHBACK_TIME：用于查找最接近的相应 SCN 值的时间。

FULL：导入源中的所有对象 [Y]。

HELP：显示帮助消息 [N]。

INCLUDE：包括特定对象类型。例如, INCLUDE=TABLE_DATA。

JOB_NAME：要创建的导入作业的名称。

LOGFILE：日志文件名 [import.log]。

NETWORK_LINK：源系统的远程数据库链接的名称。

NOLOGFILE：不写入日志文件 [N]。

PARALLEL：更改当前作业的活动 worker 的数量。

PARFILE：指定参数文件。

PARTITION_OPTIONS：指定应如何转换分区。有效的关键字为: DEPARTITION, MERGE 和 [NONE]。

QUERY：用于导入表的子集的谓词子句。例如, QUERY=employees:"WHERE department_id > 10"。

REMAP_DATA：指定数据转换函数。例如, REMAP_DATA=EMP.EMPNO:REMAPPKG.EMPNO。

REMAP_DATAFILE：在所有 DDL 语句中重新定义数据文件引用。

REMAP_SCHEMA：将一个方案中的对象加载到另一个方案。

REMAP_TABLE：将表名重新映射到另一个表。例如, REMAP_TABLE=HR.EMPLOYEES:EMPS。

REMAP_TABLESPACE：将表空间对象重新映射到另一个表空间。

REUSE_DATAFILES：如果表空间已存在, 则将其初始化 [N]。

SCHEMAS：要导入的方案的列表。

SERVICE_NAME：约束 Oracle RAC 资源的活动服务名和关联资源组。

SKIP_UNUSABLE_INDEXES：跳过设置为“索引不可用”状态的索引。

SOURCE_EDITION：用于提取元数据的版本。

SQLFILE：将所有的 SQL DDL 写入指定的文件。

STATUS：监视作业状态的频率, 其中默认值 [0] 表示只要有新状态可用, 就立即显示新状态。

STREAMS_CONFIGURATION：启用流元数据的加载

TABLE_EXISTS_ACTION：导入对象已存在时执行的操作。有效的关键字为:

APPEND, REPLACE, [SKIP] 和 TRUNCATE。

TABLES：标识要导入的表的列表。例如, TABLES=HR.EMPLOYEES,SH.SALES:SALES_1995。

TABLESPACES：标识要导入的表空间的列表。

TARGET_EDITION：用于加载元数据的版本。

TRANSFORM：要应用于适用对象的元数据转换。有效的关键字为: DISABLE_ARCHIVE_LOGGING, LOB_STORAGE, OID, PCTSPACE,SEGMENT_ATTRIBUTES, STORAGE 和 TABLE_COMPRESSION_CLAUSE。

TRANSPORTABLE：用于选择可传输数据移动的选项。有效的关键字为: ALWAYS 和 [NEVER]。仅在 NETWORK_LINK 模式导入操作中有效。

TRANSPORT_DATAFILES：按可传输模式导入的数据文件的列表。

TRANSPORT_FULL_CHECK：验证所有表的存储段 [N]。仅在 NETWORK_LINK 模式导入操作中有效。

TRANSPORT_TABLESPACES：要从中加载元数据的表空间的列表。仅在 NETWORK_LINK 模式导入操作中有效。

VERSION：要导入的对象的版本。有效的关键字为: [COMPATIBLE], LATEST 或任何有效的数据库版本。仅对 NETWORK_LINK 和 SQLFILE 有效。

VIEWS_AS_TABLES：标识要作为表导入的一个或多个视图。例如, VIEWS_AS_TABLES=HR.EMP_DETAILS_VIEW.请注意, 在网络导入模式下, 可以将表名附加到视图名。

下列命令在交互模式下有效。

注: 允许使用缩写。

CONTINUE_CLIENT：返回到事件记录模式。如果处于空闲状态, 将重新启动作业。

EXIT_CLIENT：退出客户机会话并使作业保持运行状态。

HELP：汇总交互命令。

KILL_JOB：分离并删除作业。

PARALLEL：更改当前作业的活动 worker 的数量。

START_JOB：启动或恢复当前作业。有效的关键字为: SKIP_CURRENT。

STATUS：监视作业状态的频率, 其中默认值 [0] 表示只要有新状态可用, 就立即显示新状态。

STOP_JOB：按顺序关闭作业执行并退出客户机。有效的关键字为: IMMEDIATE。

实验目的和要求

掌握在 Oracle 中进行备份与恢复的基本方式方法。实验内容要求独立完成。

实验环境

Oracle 11g

实验内容与步骤

1. 在数据库中创建 DIRECTORY 对象，并将该对象的 READ、WRITE 权限授予用户（用户需要先创建）。

```
CREATE OR REPLACE DIRECTORY dumpdir AS 'D:\ORACLE\BACKUP';
GRANT READ,WRITE ON DIRECTORY dumpdir TO ehr;
```

2. 导出或导入非同名模式的对象，赋予用户 EXP_FULL_DATABASE 和 IMP_FULL_DATABASE 权限

GRANT EXP_FULL_DATABASE, IMP_FULL_DATABASE TO ehr;

3. 在控制台 cmd 命令行下，执行 expdp help=y 和 impdp help=y，熟悉 expdp 和 impdp 命令的使用方法；

4. 利用 EXPDP 导出数据

（1）导出 Teacher 表

（2）导出 Teacher 所在的模式

（3）全库导出

5. 利用 IMPDP 导入数据

（1）导入 Teacher 表中的数据

（2）导入模式

（3）导入数据库

6. 在 SQLPLUS 命令符下，执行 set time on，然后设计数据，体验 Oracle 闪回查询的基本功能。

7. 按步骤执行代码，体验 Oracle 闪回表操作。

（1）启动“回收站”：

SQL>SHOW PARAMETER RECYCLEBIN

SQL>ALTER SYSTEM SET RECYCLEBIN=ON;

（2）查看“回收站”：

SQL>DROP TABLE test;

SQL>SELECT OBJECT_NAME,ORIGINAL_NAME,TYPE FROM USER_RECYCLEBIN;

OBJECT_NAME：删除对象在回收站中的名字，唯一；

ORIGINAL_NAME：对象删除前的名字；

（3）闪回删除：

SQL>CREATE TABLE example(ID NUMBER PRIMARY KEY, NAME CHAR(20));

SQL>INSERT INTO example VALUES(1,'BEFORE DROP');

SQL>COMMIT;

SQL>DROP TABLE example;

SQL>FLASHBACK TABLE example TO BEFORE DROP RENAME TO new_example;

SQL>SELECT * FROM new_example;

实验总结

总结实验过程中涉及到的知识点、实验过程中遇到的问题及解决方法。

2.5 实验五 数据库系统设计综合实验

预备知识

本实验的任务是设计并实现一个数据库系统。数据库设计的一般步骤包括：需求分析、概念结构设计、逻辑结构设计、物理结构设计、数据库实施、数据库运行和维护。

(1) 概念结构设计

了解概念结构设计的基本方法，根据需求分析的结果或实验题目给出的要求，能够准确地用实体联系图来描述实体和实体之间的联系。

(2) 逻辑结构设计

理解逻辑结构设计的基本方法，根据实体联系图的设计，转换成合理的关系模式，每个关系模式至少应该满足第三范式的要求。

(3) 物理结构设计

理解物理结构设计的基本方法，选择合理的索引结构和存储结构，优化数据库的存取。

(4) 数据库实施

选择一门熟悉的面向对象程序设计语言，完成应用程序的开发。

实验目的

通过实验，使学生掌握数据库系统设计和开发的一般方法，能够设计并实现简单的数据库系统。

实验要求

熟悉实验室实验环境，掌握实验预备知识，了解实验中故障排除的基本方法。实验中根据实验要求完成相应的任务，并独立完成实验报告。

实验环境

Oracle 10g，windows 2003；

实验内容和步骤

假设有“教师”、“学生”、“课程”三个实体，教师的基本信息包括：工号、姓名、职称、工资，课程的基本信息包括：课程号、课程名、学分数，学生的基本信息包括：学号、姓名、性别、年龄。系统必须满足以下要求：

(1) 一门课程只能有一个教师任课，一个教师可以上多门课程；

(2) 一个学生可以选修多门课程，一门课程可以由多个学生来选修，记录不同学生选修不同课程的成绩；

(3) 设置一个管理员，用于维护（添加、删除和修改等基本任务）学生基本信息、教师基本信息和教师所授课程等工作，此外，管理员添加学生时，为其设置初始密码；当学生选修了某门课程，课程成绩由管理员录入；

(4) 学生可以利用学号和密码登录系统，登陆系统后，可以进行选课、修改密码和个人基本信息、查询自己的选课及总学分等操作；

(5) 能够统计不同职称的教师的数量、不同职称的教师的平均工资，可以统计每门课程的平均成绩、最高分、最低分，统计每个学生选修课程的总学分；

根据上述描述，解答下列问题：

(1) 设计并画出 E-R 图，要求标注连通词（即联系类型）；

(2) 将 E-R 图转化为关系模型，并指出各关系的主码和外码；

(3) 在 MySql、SQL Server、Oracle 中选择一个数据库管理系统，并设计数据库的物理结构；

(4) 选择一门熟悉的面向对象程序设计语言，完成系统开发。

实验总结

总结实验过程中涉及到的知识点、实验过程中遇到的问题及解决方法。

3 附录

3.1 项目信息管理数据库 DDL(Oracle SQL)

```
alter table Department
    drop constraint FK_DEPARTME_MANAGE_TEACHER;
alter table MyProject
    drop constraint FK_MYPROJEC_PP_MYPROJEC;
alter table MyProject
    drop constraint FK_MYPROJEC_PROJECTMA_TEACHER;
alter table TM
    drop constraint FK_TM_TM_TEACHER;
alter table TM
    drop constraint FK_TM_TM2_MYPROJEC;
alter table Teacher
    drop constraint FK_TEACHER_BELONGTO_DEPARTME;
drop table Department cascade constraints;
drop table MyProject cascade constraints;
drop table TM cascade constraints;
drop table Teacher cascade constraints;
/*==============================================================*/
/* Table: Department                                            */
/*==============================================================*/
create table Department
(
    DNO                  VARCHAR2(10)          not null,
    TNO                  VARCHAR2(20),
    DName                NVARCHAR2(20)         not null,
    constraint PK_DEPARTMENT primary key (DNO)
);
/*==============================================================*/
/* Table: MyProject                                             */
/*==============================================================*/
create table MyProject
(
    PNO                  VARCHAR2(20)          not null,
    ParentPno            VARCHAR2(20),
    TNO                  VARCHAR2(20)          not null,
    PName                NVARCHAR2(30)         not null,
    PFund                NUMBER(20,2),
    PBeginDate           DATE                  not null,
    PTimeSpan            INTEGER               not null,
```

```
    constraint PK_MYPROJECT primary key (PNO)
);
/*==============================================================*/
/* Table: TM                                                    */
/*==============================================================*/
create table TM
(
    TNO                 VARCHAR2(20)        not null,
    PNO                 VARCHAR2(20)        not null,
    WorkDays            INTEGER,
    constraint PK_TM primary key (TNO, PNO)
);
/*==============================================================*/
/* Table: Teacher                                               */
/*==============================================================*/
create table Teacher
(
    TNO                 VARCHAR2(20)        not null,
    DNO                 VARCHAR2(10)        not null,
    TName               NVARCHAR2(20)       not null,
    TSex                VARCHAR2(3)         not null,
    TSalary             NUMBER(30,2),
    TBirthday           DATE                not null,
    constraint PK_TEACHER primary key (TNO)
);
alter table Department
    add constraint FK_DEPARTME_MANAGE_TEACHER foreign key (TNO)
        references Teacher (TNO);
alter table MyProject
    add constraint FK_MYPROJEC_PP_MYPROJEC foreign key (ParentPno)
        references MyProject (PNO);
alter table MyProject
    add constraint FK_MYPROJEC_PROJECTMA_TEACHER foreign key (TNO)
        references Teacher (TNO);
alter table TM
    add constraint FK_TM_TM_TEACHER foreign key (TNO)
        references Teacher (TNO);
alter table TM
    add constraint FK_TM_TM2_MYPROJEC foreign key (PNO)
        references MyProject (PNO);
alter table Teacher
    add constraint FK_TEACHER_BELONGTO_DEPARTME foreign key (DNO)
        references Department (DNO);
```

3.2 项目信息管理数据库初始化数据

--注意：拷贝的日期，例如“7-7 月-1977”，拷贝后，在“7 月”之间可能有空格，必须删除以后再执行数据插入。

--系表数据

```
insert into department(dno,dname) values('d001', '计算机科学系');
insert into department(dno,dname) values('d002', '网络工程系');
insert into department(dno,dname) values('d003', '软件工程系');
```

--教师表数据（注意，复制粘贴后，“7 月”可能有空格，请删除空格再执行，后同）

```
insert into teacher(tno, tname, tsex, tsalary, tbirthday, dno) values('t001', '张三', '男', 3000, To_date('7-7 月-1977', 'DD-mon-yyyy'), 'd001');
insert into teacher(tno, tname, tsex,  tsalary, tbirthday, dno) values('t002', '李四', '女', 3600, To_date('21-10 月-1979', 'DD-mon-yyyy'), 'd001');
insert into teacher(tno, tname, tsex,  tsalary, tbirthday, dno) values('t003', '王五', '女', 5600, To_date('7-7 月-1981', 'DD-mon-yyyy'), 'd002');
insert into teacher(tno, tname, tsex,  tsalary, tbirthday, dno) values('t004', '刘晨', '女', 5800, To_date('7-7 月-1985', 'DD-mon-yyyy'), 'd002');
insert into teacher(tno, tname, tsex,  tsalary, tbirthday, dno) values('t005', '王二小', '男', 3500, To_date('7-7 月-1981', 'DD-mon-yyyy'), 'd003');
insert into teacher(tno, tname, tsex,  tsalary, tbirthday, dno) values('t006', '李小龙', '男', 5687, To_date('7-12 月-1990', 'DD-mon-yyyy'), 'd003');
insert into teacher(tno, tname, tsex,  tsalary, tbirthday, dno) values('t007', '熊猫', '男', 6000, To_date('27-11 月-1980', 'DD-mon-yyyy'), 'd003');
insert into teacher(tno, tname, tsex,  tsalary, tbirthday, dno) values('t008', '李小小', '女', 5687, To_date('17-10 月-1985', 'DD-mon-yyyy'), 'd001');
```

--myproject 数据

```
insert into myproject(PNO, pname, pfund, pbegindate, ptimespan, parentpno, tno)
values('p0001', '信息安全技术研究', 30, To_date('7-12 月-2012', 'DD-mon-yyyy'), 3, null, 't001');
insert into myproject(PNO, pname, pfund, pbegindate, ptimespan, parentpno, tno)
values('p0002', '云计算研究', 40, To_date('7-12 月-2008', 'DD-mon-yyyy'), 4, null, 't003');
insert into myproject(PNO, pname, pfund, pbegindate, ptimespan, parentpno, tno)
values('p0003', '信息中心网络研究', 60, To_date('7-12 月-2012', 'DD-mon-yyyy'), 6, null, 't006');
insert into myproject(PNO, pname, pfund, pbegindate, ptimespan, parentpno, tno)
values('p0004', '对等网络研究', 30, To_date('7-12 月-2010', 'DD-mon-yyyy'), 3, 'p0002', 't006');
```

--tm 数据

```
insert into tm(tno, pno, workdays) values('t001', 'p0001', 180);
insert into tm(tno, pno, workdays) values('t001', 'p0002', 180);
insert into tm(tno, pno, workdays) values('t001', 'p0004', 360);
insert into tm(tno, pno, workdays) values('t001', 'p0003', 540);
insert into tm(tno, pno, workdays) values('t002', 'p0001', 720);
insert into tm(tno, pno, workdays) values('t002', 'p0002', 90);
```

```
insert into tm(tno, pno, workdays) values('t002', 'p0004', 90);
insert into tm(tno, pno, workdays) values('t003', 'p0001', 180);
insert into tm(tno, pno, workdays) values('t003', 'p0002', 360);
insert into tm(tno, pno, workdays) values('t003', 'p0003', 360);
insert into tm(tno, pno, workdays) values('t003', 'p0004', 360);
insert into tm(tno, pno, workdays) values('t004', 'p0001', 180);
insert into tm(tno, pno, workdays) values('t004', 'p0002', 180);
insert into tm(tno, pno, workdays) values('t004', 'p0003', 360);
insert into tm(tno, pno, workdays) values('t005', 'p0004', 180);
insert into tm(tno, pno, workdays) values('t005', 'p0002', 720);
insert into tm(tno, pno, workdays) values('t005', 'p0003', 180);
insert into tm(tno, pno, workdays) values('t006', 'p0004', 360);
insert into tm(tno, pno, workdays) values('t006', 'p0002', 180);
insert into tm(tno, pno, workdays) values('t006', 'p0003', 720);
update department set tno='t006' where dno='d003';
commit;
```

3.3 在 Oracle 12c 中创建新用户并分配表空间

```
create temporary tablespace oraclelesson_temp
tempfile 'H:oraclespace\oraclelesson_temp01.dbf'
size 2m
autoextend on
next 2m maxsize 1024m
extent management local;

create tablespace oraclelesson_data
logging
datafile 'H:oraclespace\oraclelesson_data01.dbf'
size 2m
autoextend on
next 2m maxsize 2048m
extent management local;

create user c##oraclelesson identified by o123
profile default
default tablespace oraclelesson_data
temporary tablespace oraclelesson_temp;
grant resource,connect to c##oraclelesson;
grant unlimited tablespace to c##oraclelesson;
```

3.4 上机建议及常见问题解决方法

3.4.1 实验建议

(1) 实验开始时，请首先创建一个用户并授予必要的权限，然后利用新创建的用户登录，在新用户模式下进行实验。创建用户和授权的示例代码如下：

create user dblesson identified by d1234;/*创建新用户 dblesson，密码为 d1234*/
grant connect,resource to dblesson;/*把 connect、resource 角色权限授予 dblesson*/
上述两句可以简写成一句：grant connect, resource to dblesson identified by d1234;

(2) 实验时，建议先把实验的实验内容和步骤拷贝到 Oracle SQL Developer 中，然后按照内容编号有序完成实验内容。

(3) 如果要在个人主机上安装 Oracle，建议安装 Oracle 11g Express，并安装管理工具 Oracle SQL Developer(两个软件均可在 www.oracle.com 下载)。

3.4.2 SQL Developer 无法启动

(1) Unable to create an instance of the Java Virtual Machine Located at path: <SQLDEVELOPER>/jdk/jre/bin/client/jvm.dll

解决方法：右键单击“我的电脑”，选择“属性”，点击高级选项卡，设置环境变量 path，把 Java 的 bin 目录路径添加到 path 变量中。

(2) 没有找到 msvcr71.dll，因此这个应用程序未能启动。

找到 msvcr71.dll，拷贝到%windir%\system32 目录下或其他环境变量 path 的目录。

3.4.3 用户被锁定或密码错误

(1) system 账户被锁定（其他用户解锁方式方法相同）

使用超级管理员 sys 登录来解锁 system 账户。

方法一：启动 CMD 进入命令提示符，执行命令：SQLPLUS / as SYSDBA，然后执行 Alter user system account unlock 解锁 system 账户。

方法二：在 SQL Developer 中，使用本地/继承、sysdba、操作系统验证连接，连接后执行 Alter user system account unlock 解锁 system 账户。

(2) system 账户密码错误（其他用户处理方式方法相同）

同样，采用 SYS 用户登陆数据库，执行 Alter user SYSTEM identified by 新密码;

3.4.4 SQL Developer 无法连接数据库

(1) 机器没有安装 oracle

解决方法：连接旁边的机器做实验。

(2) 服务没有启动

解决方法：启动 OracleServiceOrcl 和监听器 listener 服务。

(3) 参数配置错误

解决方法：利用 net manager 网络配置工具配置监听程序的监听地址。

(4) 其他

如果使用了很多方法仍然无法连接数据库，试试使用 SQLPlus 连接，如果 SQL Plus 也无法连接，则试试连接旁边的机器。

3.5 Power designer 使用说明

(1) 如何把生成的脚本中的对象的双引号去掉？

打开 cdm 的情况下，进入 Tools－Model Options－Naming Convention，把 Name 和 Code 的卷标的 Charcter case 选项设置成 Uppercase 或者 Lowercase，只要不是 Mixed Case 就行！或者选择 Database->Edit current database->Script->Sql->Format，有一项 CaseSensitivityUsingQuote，它的 comment 为“Determines if the case sensitivity for identifiers is managed using double quotes”，表示是否适用双引号来规定标识符的大小写，可以看到右边的 values 默认值为“YES”,改为“No”即可！

或者在打开 pdm 的情况下，进入 Tools－Model Options－Naming Convention，把 Name 和 Code 的卷标的 Character case 选项设置成 Uppercase 就可以！

(2) 由 pdm 生成建表脚本时，字段超过 15 字符发生错误（oracle）

解决办法是打开 PDM 后，会出现 Database 的菜单栏，进入 Database －Edit Current DBMS －script－objects－column－maxlen，把 value 值调大，比如改成 60。

表或者其它对象的长度也出现这种错误的话，可以选择对应的 objects 照此方法更改！

(3) 建立一个表后，为何检测出现 Existence of index 的警告

A table should contain at least one column, one index, one key, and one reference.

可以不检查 Existence of index 这项，也就没有这个警告错误！意思是说没有给表建立索引，这只是一个警告。

4 PL/SQL 程序设计

4.1 基本语法

4.1.1 PL/SQL 块结构

```
[DECLARE
    --declaration statements]   ①声明部分
BEGIN
    --executable statements    ②可执行部分
[EXCEPTION
    --exception statements]    ③异常处理部分
END;
```

（1）声明部分：主要用于声明变量、常量、数据类型、游标、异常处理名称以及本地（局部）子程序定义等。

（2）可执行部分：执行部分是 PL/SQL 块的功能实现部分。该部分通过变量赋值、流程控制、数据查询、数据操纵、数据定义、事务控制、游标处理等实现块的功能。

（3）异常处理部分：异常处理部分用于处理该块执行过程中产生的异常。

4.1.2 PL/SQL 常用数据类型

字符类型：CHAR、NCHAR、VARCHAR2、NVARCHAR2、VARCHAR
数字类型：NUMBER、BINARY_INTEGER、PLS_INTEGER
日期/区间类型：DATE、TIMESTAMP、INTERVAL
布尔类型：BOOLEAN
行标识类型：ROWID、UROWID
原始类型：LONG、LONG RAW
LOB 类型：CLOB、BLOB、NCLOB、BFILE
引用类型：REF CURSOR、REF OBJECT_TYPE
记录类型：RECORD
集合类型：TABLE

4.1.3 PL/SQL 变量声明语法

identifier [CONSTANT] datatype [NOT NULL [:= | DEFAULT expr];
语法说明如下：
（1）每行只能定义一个变量；
（2）关键字 CONSTANT 表示所定义的是一个常量，必须为它赋初值；
（3）如果定义变量时使用 NOT NULL 关键字，则必须为变量赋初值；
（4）如果变量没有赋初值，则默认为 NULL；
（5）使用 DEFAULT 或“:=”运算符为变量初始化。
变量声明示例：

```
Declare
   v_date          DATE;
   v_dno           NUMBER(2) NOT NULL := 10;
   c_pi            CONSTANT NUMBER := 3.14;
   v_regdate       DATE := SYSDATE + 7;
   v_row           TEACHER%ROWTYPE;-- TEACHER%ROWTYPE 表示引用 Teacher 表的行记录类型
   v_tno           TEACHER.tno%TYPE;-- TEACHER.tno%TYPE 表示引用 teacher 表的 tno 列的数据类型
```

4.1.4 记录类型的定义语法

```
TYPE record_type IS RECORD(
field1 datatype1 [NOT NULL][DEFAULT|:=expr1],
field2 datatype2 [NOT NULL][ DEFAULT|:=expr2],
```

……

fieldn datatypen [NOT NULL][DEFAULT|:=exprn]);

4.2 控制结构

4.2.1 if 语句基本格式

IF condition1 THEN statements1;

[ELSIF condition2 THEN statements2;]

……

[ELSE else_statements];

END IF;

注意：条件是一个布尔型变量或表达式，取值只能是 TRUE，FALSE，NULL。

4.2.2 CASE 语句基本格式

```
CASE
        WHEN condition1 THEN statements1;
        WHEN condition2 THEN statements2;
        ……
        WHEN conditionn THEN statementsn;
        [ELSE       else_statements;]
END CASE;
```

在 CASE 语句中，当第一个 WHEN 条件为真时，执行其后的操作，操作完后结束 CASE 语句。其他的 WHEN 条件不再判断，其后的操作也不执行。

CASE 语句的另外一种形式如下：

```
CASE test_value
      WHEN value1 THEN statements1;
      WHEN value2 THEN statements2;
      ……
      WHEN valuen THEN statementsn;
      [ELSE       else_ statements;]
END CASE;
```

4.2.3 简单循环语句格式

```
LOOP
    sequence_of_statement;
    EXIT [WHEN condition] ;
END LOOP;
```

注意，在循环体中一定要包含 EXIT 语句，否则程序进入死循环。

4.2.4 While 循环语句格式

```
WHILE condition LOOP
    sequence_of_statement;
END LOOP;
```

4.2.5 For 循环语句格式

```
FOR  循环变量  IN [REVERSE]  循环下限..循环上限  LOOP
    --循环体
END LOOP;
```

注意，循环变量不需要显式定义，系统隐含地将它声明为 BINARY_INTEGER 变量；系统默认时，循环变量从下界往上界递增计数，如果使用 REVERSE 关键字，则表示循环变量从上界向下界递减计数；循环变量只能在循环体中使用，不能在循环体外使用。

4.2.6 PL/SQL 块结构示例

```
set serveroutput on;
DECLARE
    v_tname VARCHAR2(10);
BEGIN
    SELECT tname into v_tname from teacher where tno = 't001';
    DBMS_OUTPUT.PUT_LINE(v_tname);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE(没有找到记录');
END;
```

4.3 Oracle 数据库对象

4.3.1 序列

（1）序列创建语法

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

（2）删除序列

DROP SEQUENCE sequence_name；

4.3.2 同义词

（1）同义词创建语法

CREATE [PUBLIC] SYNONYM synonym_name FOR [schema.] object[@db_link];

（2）同义词

DROP SYNONYM synonym_name;

4.3.3 存储过程

（1）存储过程创建语法

```
CREATE [OR REPLACE] PROCEDURE procedure_name
(parameter1_name [mode] datatype [DEFAULT|:=value]
[, parameter2_name [mode] datatype [DEFAULT|:=value],…])
AS|IS
    /*Declarative section is here */
BEGIN
    /*Executable section is here*/
EXCEPTION
    /*Exception section is here*/
END[procedure_name];
```

参数的模式说明如下：

IN（默认参数模式）表示当过程被调用时，实参值被传递给形参；在过程内，形参起常量作用，只能读该参数，而不能修改该参数

OUT 表示当过程被调用时，实参值被忽略；在过程内，形参初始值为 NULL，可以进行读/写操作；当子程序调用结束后返回调用环境时，形参值被赋给实参。OUT 模式参数只能是变量，不能是常量或表达式。

IN OUT 表示当过程被调用时，实参值被传递给形参；在过程内，形参起已初始化的 PL/SQL 变量的作用，可读可写；当子程序调用结束返回调用环境时，形参值被赋给实参。IN OUT 模式参数只能是变量，不能是常量或表达式。

（2）删除存储过程

DROP PROCEDURE 过程名;

（3）存储过程创建示例

```
create or replace procedure findTeacherByTno
(v_tno varchar2 )
as
    v_tname VARCHAR2(10);
BEGIN
    SELECT tname into v_tname from teacher where tno = v_tno;
    DBMS_OUTPUT.PUT_LINE(v_tname);
EXCEPTION
    WHEN NO_DATA_FOUND THEN
```

```
            DBMS_OUTPUT.PUT_LINE('没有找到记录');
END findTeacherByTno;
```

（4）调用存储过程

```
exec pro_name(参数 1..);
call pro_name(参数 1..);
```

4.3.4 函数

（1）创建函数的基本语法

```
CREATE [OR REPLACE] FUNCTION function_name
(parameter1_name [mode] datatype [DEFAULT|:=value]
[, parameter2_name [mode] datatype [DEFAULT|:=value],…])
RETURN return_datatype
AS|IS
        /*Declarative section is here */
BEGIN
      /*Executable section is here*/
EXCEPTION
      /*Exception section is here*/
END [function_name];
```

（2）删除函数

```
DROP FUNCTION  函数名；
```

（3）函数创建示例

```
create or replace function sumdaysByTno(v_tno varchar2)
   return number
as
   sumdays number;
begin
   select sum(workdays) into sumdays from tm where tno=v_tno;
   return sumdays;
   --exception  省略
end;
```

（4）函数调用示例

```
Select teacher.*, sumdaysByTno(tno) from teacher;
```

4.3.5 触发器

（1）创建 DML 触发器的语法

```
CREATE [OR REPLACE] TRIGGER trigger_name
BEFORE|AFTER triggering_event [OF column_name]
ON table_name]
[FOR EACH ROW]
[WHEN trigger_condition]
```

```
DECLARE
    /*Declarative section is here */
BEGIN
    /*Executable section is here*/
EXCEPTION
    /*Exception section is here*/
END [trigger_name];
```

语句级触发器：在默认情况下创建的 DML 触发器为语句级触发器，即触发事件发生后，触发器只执行一次。在语句级触发器不能对列值进行访问和操作，也不能获取当前行的信息。如果触发器响应多个 DML 事件，而且需要根据事件的不同进行不同的操作，则可以在触发器体中使用 3 个条件谓词，即 INSERTING、UPDATING、DELETING。

行级触发器：行级触发器是指执行 DML 操作时，每操作一个记录，触发器就执行一次，一个 DML 操作涉及多少个记录，触发器就执行多少次。在行级触发器中可以使用 WHEN 条件，进一步控制触发器的执行。在行级触发器中引入了:old 和:new 两个标识符，来访问和操作当前被处理记录中的数据。:old 和:new 是 DML 语句对应表的行记录类型。在不同触发事件中，:old 和:new 的意义不同。在执行部分引用使用:old.field 和:new.field，在 WHEN 条件中使用则不带冒号。注意，:old 和:new 是伪记录，不能作为整个记录进行赋值或引用，不能传递给带 triggering_table%ROWTYPE 参数的过程和函数。

（2）创建 INSTEAD OF 触发器的**语法**

```
CREATE [OR REPLACE] TRIGGER trigger_name
INSTEAD OF triggering_event [OF column_name]
ON view_name
FOR EACH ROW
[WHEN trigger_condition]
DECLARE
    /*Declarative section is here */
BEGIN
    /*Executable section is here*/
EXCEPTION
    /*Exception section is here*/
END [trigger_name];
```

（3）创建系统触发器的语法

```
CREATE [OR REPLACE] TRIGGER trigger_name
BEFORE|AFTER ddl_event_list|database_event_list
ON DATABASE|SCHEMA
[WHEN trigger_condition]
DECLARE
    /*Declarative section is here */
BEGIN
   /*Executable section is here*/
EXCEPTION
   /*Exception section is here*/
END [trigger_name];
```

（4）删除触发器

DROP TRIGGER 触发器名称;

（5）行级触发器示例

修改研究期限时，只能增大研究期限，否则拒绝执行。

```
create or replace trigger myprojecttrigger
before update of ptimespan on myproject
for each row
when(new.ptimespan<old.ptimespan)
begin
   raise_application_error(-20000, '研究期限只能增大');
end;
```

（6）语句级触发器示例

为 myproject 表建立一个触发器，当执行删除和插入操作时，统计项目负责人负责的项目数量，当执行更新操作时，统计科研项目的总经费。

```
create or replace trigger triggertest
after insert or delete or update of pfund
on myproject
declare
   cursor c_project is select tno,count(*) pnum from myproject group by tno;
   sumfund number;
begin
   if(updating) then
      select sum(pfund) into sumfund from myproject;
      dbms_output.put_line('总经费数量为：'||sumfund);
   else
      for tmp in c_project loop
         dbms_output.put_line(tmp.tno || ' ' || tmp.pnum);
      end loop;
   end if;
end;
```

4.4 集合类型

（1）索引表类型的定义语法

TYPE index_table IS TABLE OF element_type INDEX BY BINARY_INTERGER| PLS_INTEGER| VARCHAR2(n);

（2）嵌套表定义的语法

TYPE nested_table IS TABLE OF element_type [NOT NULL];

（3）可变数组类型定义的基本语法

TYPE varray_name IS VARRAY|VARYING ARRAY(maximum_size) OF element_type [NOT NULL];

4.5 批绑定

（1）FORALL 语句语法

FORALL index IN lower_bound..upper_bound [SAVE EXCEPTIONS] sql_statement;

FORALL index IN INDICES OF COLLECT [BETWEEN lower_bound AND upper_bound] sql_statement;

FORALL index IN VALUES OF index_COLLECT sql_statement

（2）批查询语法

SELECT column_name BULK COLLECT INTO collect_name;

FETCH cursor_name BULK COLLECT INTO collect_name;

RETURNING column_name BULK COLLECT INTO collect_name;

4.6 动态 SQL

（1）处理动态非查询语句及单行查询语句的动态 SQL 使用 Execute Immediate 命令

EXECUTE IMMEDIATE ‘SQL statement’

[INTO (var1,var2,…|record)]

[USING [IN|OUT|IN OUT] bindvar1,bindvar2,…]

[RETURNING| RETURN INTO outvar1,outvar2,…];

INTO 子句：将查询的返回值赋给 PL/SQL 变量，该变量的模式为 OUT。

USING 子句：将 SQL 语句中的绑定变量与 PL/SQL 变量或参数相关联。如果动态 SQL 语句中有 RETURNING 子句，则该变量可以是 IN 模式给绑定变量传值，也可以是 OUT 模式接收值。否则，只能是 IN 模式。

RETURNING：RETURNING 中的变量是 out 模式，接收动态 SQL 语句中 RETURNING 子句的操作返回值。

（2）动态批绑定的 DML 基本语法

FORALL index IN lower_bound..upper_bound

EXECUTE IMMEDIATE ‘v_sql’

[USING collection_name]

[RETURNING BULK COLLECTION INTO output_collection_name]

（3）Select 语句中使用动态批绑定的基本语法

EXECUTE IMMEDIATE ‘dyn_select_SQL’

BULK COLLECT INTO collection_name

USING [IN|OUT|IN OUT] bindvar1, bindvar2,… ;

（4）在 FETCH 语句中使用动态批绑定的基本语法

OPEN cursor_variable FOR ‘dyn_SQL’

USING [IN|OUT|IN OUT] bindvar1, bindvar2,… ;

FETCH cursor_variable BULK COLLECT INTO collection_name

5 课堂练习题

5.1 基本语法

1. 定义一个记录类型，名称为 projectType，包括项目编号、项目名称、科研经费等信息。编写一个匿名块，把编号为’p0001’的项目的项目编号、项目名称、科研经费填充到一个 projectType 记录类型的变量，打印输出该变量的信息。

2. 编写一个匿名块，输出编号为‘d001’的系的全部信息，然后在系名前面加上“计算机学院”并更新系表。

3. 使用循环结构向教师表添加 100000 个教师信息，每个教师的所在系均为’d001’，教师编号为 t1、t2…t100000，奇数编号的教师性别为女，偶数编号的教师性别为男。
   - DBMS_RANDOM.RANDOM：返回一个整数
   - DBMS_RANDOM.VALUE：默认返回 0 到 1 之间的随机数
   - DBMS_RANDOM.VALUE(m,n)：生成 m 和 n 之间的随机数
   - DBMS_RANDOM.STRING(‘类型代码’, 长度)：类型代码可以是 U、L、A，U 用来生成大写字符，L 用来生成小写字符，A 用来生成大小写混合的字符；长度指定生成的随机字符串的长度

4. 使用循环结构为每个系添加 100000 个教师信息，奇数编号的教师性别为女，偶数编号的教师性别为男。

5. 编写程序，把所有教师的编号、姓名、负责的项目总经费等信息一行一行输出,如果某个教师没有负责任何项目，则该教师的项目总经费为 0

5.2 数据库对象

1. 创建一个存储过程，如果项目名称包含“云计算”，则项目经费增加 10，包含“对等网络”则增加 5，包含“网格计算”则增加 2，其他项目的经费保持不变，存储过程的参数必须返回更新的记录数目。

2. 编写一个函数，计算教师表中所有教师的平均年龄。

3. 为 myproject 表编写一个行级触发器，当插入数据时，如果新数据的 pfund<ptimespan*10，禁止插入数据，否则把 pfund 设置为 ptimespan*10，然后插入数据

4. 假设有视图 testview 定义如下，编写一个 instead of 触发器，当向该视图中插入数据时，使数据插入到 tm 表中（Create view testview as select teacher.tno, tname, workdays from teacher.tno=tm.tno）。

5.3 集合类型

1. 创建一个以 BINARY_INTERGER 为键值的索引表，用于存储教师号为 t001 的教师负责的项目信息，然后在索引表中删除倒数第二项，最后输出索引表中的项目信息。

2. 创建一个全局嵌套表类型，用于存储 myproject 表的行信息。创建一个教师表 teacher2，其列包括 tno、tname 和嵌套表类型列，该列用于存储教工 tno 所负责的所有项目信息。把 teacher 表和 myproject 表中的相关信息填入表 teacher2。

3. 创建一个以 BINARY_INTERGER 为键值的索引表，用于存储教师号为 t001 的教师负责的项目信息，然后在索引表中删除倒数第二项，最后输出索引表中的项目信息。

5.4 批绑定与动态 SQL

1. 利用批查询把“软件工程系”的教师负责的项目的项目编号、名称、经费存入一个嵌套表，并输出嵌套表的内容。

2. 创建一个过程，根据指定某列名称和字符串值来查询项目信息并输出。