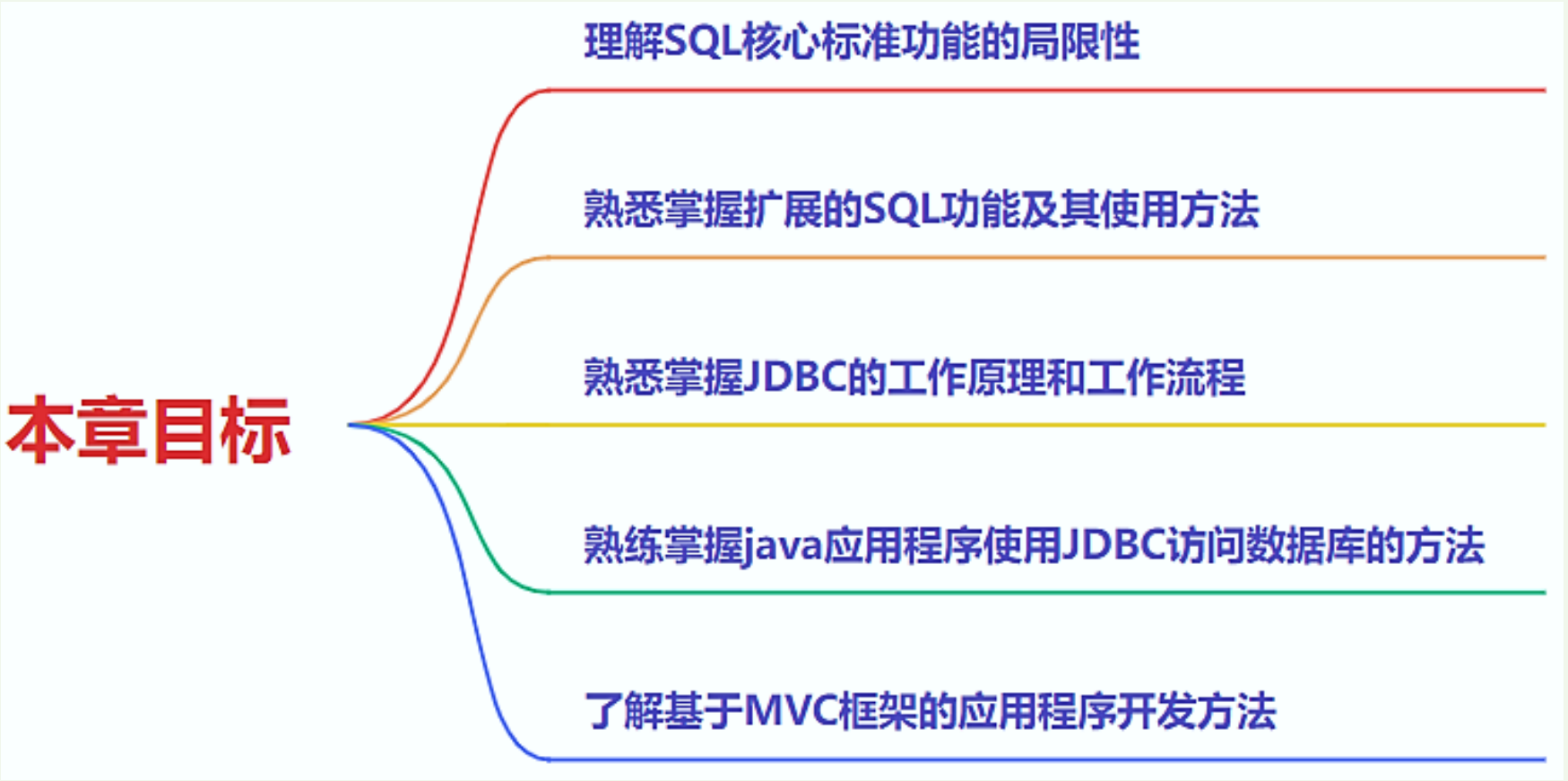
第8章 数据库编程

本章导图


本章目标
理解SQL核心标准功能的局限性
熟悉掌握扩展的SQL功能及其使用方法
熟悉掌握JDBC的工作原理和工作流程
熟练掌握java应用程序使用JDBC访问数据库的方法
了解基于MVC框架的应用程序开发方法

大纲

厦門大學 XIAMEN UNIVERSITY

- 概述
- 过程化SQL
- JDBC编程
- 基于MVC框架的数据库应用
- 本章小结

1.概述

厦門大學 XIAMEN UNIVERSITY

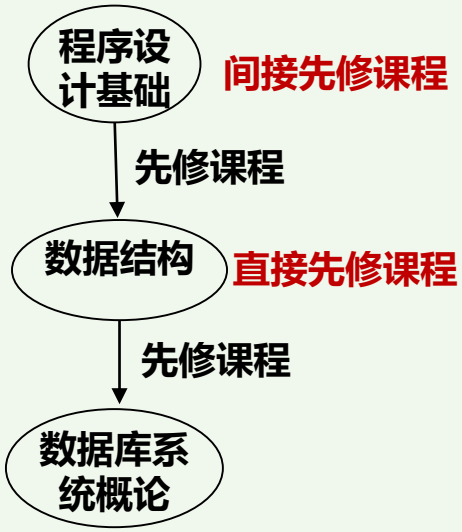
- SQL表达能力的限制
- 扩展SQL的功能
- 通过高级语言实现复杂应用

1.1SQL表达能力的限制

厦門大學 XIAMEN UNIVERSITY

- **[任务一]** 打印“数据库系统概论”课程的所有先修课信息

课程号Cno	课程名Cname	学分Ccredit	先修课程Cpno
81001	程序设计基础与C语言	4	NULL
81002	数据结构	4	81001
81003	数据库系统概论	4	81002
81004	信息系统概论	4	81003
81005	操作系统	4	81001
81006	Python语言	3	81002
81007	离散数学	4	NULL
81008	大数据技术概论	4	81003



- **任务分析：SQL如何表达**
  - **难点：**课程可能同时存在直接先修课和间接先修课
  - **情况一：**如何查询直接先修课：**自身连接查询**
  - **情况二：**如何查询间接先修课：**递归查询（无法表达）**

1.1~直接先修课的SQL语句表达

厦門大學 XIAMEN UNIVERSITY

- **如何查询直接先修课：自连接**
  - 步骤1：找出“数据库系统概论”课程的全部直接先修课：记为L[1]；如果L[1]为空，则任务一结束。

L[1]:

```
select B.Cname
from Course A, Course B
where A.Cname='数据库系统概论'and A.Cpno=B.Cno;
```

1.1~间接先修课的SQL语句表达

厦門大學 XIAMEN UNIVERSITY

- **如何查询间接先修课：递归执行**
  - 步骤i (i>=2)：找出集合L[i-1]中每一门课程的全部直接先修课，并计算它们的并集，记为L[i]
  - 迭代执行步骤i, 直到并集L[i]为空，输出L[1] ∪ … ∪ L[i]

L[2]:

```
select B.Cname
from Course A, Course B
where A.Cname= '数据结构' and
A.Cpno=B.Cno;
```

L[3]:

```
select B.Cname
from Course A, Course B
where A.Cname = `程序设计基础与C语言'
and A.Cpno=B.Cno;
```

表8.1 任务1的输出结果

Cpno	Cname
81002	数据结构
81001	程序设计基础与C语言

1.1SQL表达能力的限制(cont.)

厦門大學 XIAMEN UNIVERSITY

- **[任务二]** 打印一周内将过生日的学生信息

学号Sno	姓名Sname	性别Ssex	生日Sbirthdate	主修专业Smajor
20180001	李勇	男	2000-3-8	信息安全
20180002	刘晨	女	1999-9-1	计算机科学与技术
20180003	王敏	女	2001-8-1	计算机科学与技术
20180004	张立	男	2000-1-8	计算机科学与技术
20180005	陈新奇	男	2001-11-1	信息管理与信息系统
20180006	赵明	男	2000-6-12	数据科学与大数据技术
20180007	王佳佳	女	2001-12-7	数据科学与大数据技术

- **任务分析：SQL如何表达**
  - **任务二**需要数据库系统提供内置函数(本任务主要是日期数)。其求解思路为：扫描每个学生的出生日期，例如 2005-6-12，并执行以下步骤:
    - 把出生日期的年份换成当前日期所在的年份(例如2024)，即2024-6-12
    - 获取当前系统的日期，例如 2024-5-13。
    - 确定过生日的日期范围[2024-5-13，2024-5-20]，即以当前日期为下界，当前日期后的第七天作为上界。
    - 判断出生日期是否在上述日期范围内，如果是，输出该学生信息。
  - 需要使用SQL的**内置函数**完成该任务

1.1SQL表达能力的限制(cont.)

厦門大學 XIAMEN UNIVERSITY

- **[任务三]** 给定学生学号，计算学生的平均学分绩点GPA

表8.2学号为“20180001”的学生的选修课程

学号Sno	课程号Cno	成绩Grade	选课学期Semester	教学班Teachingclass
20180001	81001	85	20192	81001-01
20180001	81002	96	20201	81002-01
20180001	81003	87	20202	81003-01

- **任务分析：SQL如何表达**
  - **难点：**需要用户自主设计业务处理逻辑?
  - 求解思路：给定学生学号，找出该学生所有选修课程的学分、成绩；根据每门课程的成绩，参照“成绩和绩点对照表”，确定该成绩所处的范围，找出该门课程对应的绩点。
  - 使用**存储过程**来完成这些比较复杂的业务逻辑

1.1SQL表达能力的限制(cont.)

厦門大學 XIAMEN UNIVERSITY

- **[任务四]** 教学评价浏览与反馈：学生通过交互界面提交对某一位任课老师的教学评价意见，教师浏览这些评价意见并提供反馈信息

教师教学评价表

学号 Sno	职工号 Tno	教学班号 TCno	意见内容 Assess	意见类型 CAtype	教师反馈 Feedback
20180001	19950018	81001-01	作业难度比较合适	正面	感谢肯定
20180003	19950018	81001-01	老师和助教也很耐心	正面	感谢肯定
20180002	19910101	81001-02	实验框架较为复杂	负面	根据同学们的建议，简化框架

- **任务分析：SQL如何表达**
  - **难点：**需要建立交互功能。一方面，学生需要找到指定的教学班和授课教师，建立如图8.1所示的交互界面并输入课程评价。另一方面，还要建立如图8.2所示的交互界面，教师浏览教学班学生的评价意见，并针对每条评价逐一做出回复
  - **基于JDBC**实现学生选课系统的评教功能

1.1SQL表达能力的限制(cont.)

图8.1 学生输入并提交课程评价

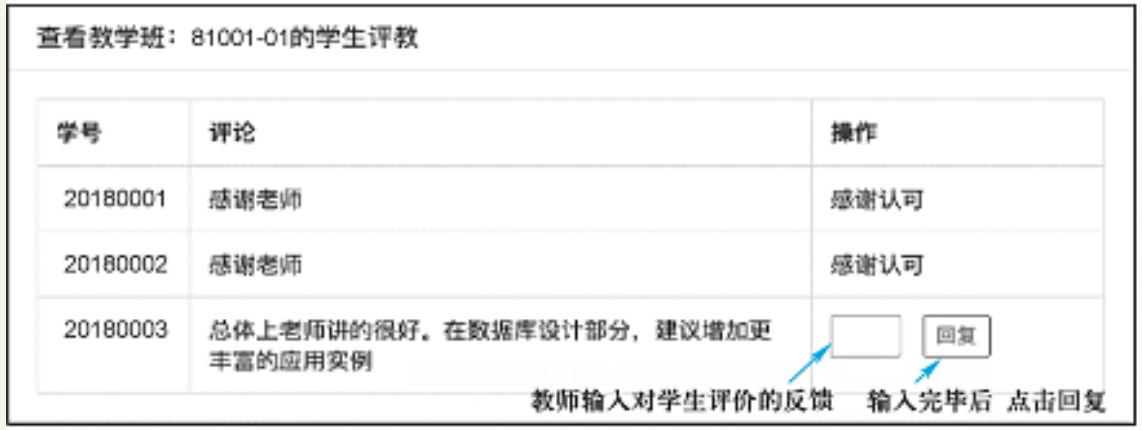


图8.2 教师提交对学生评价的反馈

1.1~小结

厦門大學 XIAMEN UNIVERSITY

- **SQL语言表达能力的限制**
  - 无法表达递归等复杂操作--如间接先修课的查询
  - 无法对数据进行复杂操作—如查询一周内将过生日的同学
  - 无法自主设计业务处理逻辑—如计算学生平均学分绩点
  - 无法进行交互式操作—如教学评价与反馈
- 解决上述问题的两种途径：
  - 扩展SQL的功能
    - 如PL/SQL
  - 通过高级语言实现复杂应用
    - 如python, java, …

1.2扩展SQL的功能

厦門大學 XIAMEN UNIVERSITY

- SQL标准的发展历史可参见：

  https://blog.csdn.net/a13407142317/article/details/134995207

- SQL扩展的方式主要包括：
  - **引入新的SQL子句**
  - **引入新的内置函数**
  - **引入PL/SQL与存储过程/函数**

1.2.1引入新的SQL子句

厦門大學 XIAMEN UNIVERSITY

- 以任务一为例，SQL标准引入了WITH RECURSIVE子句，可执行递归查询。
- 类似于WITH RECURSIVE子句的SQL扩展还有很多，例如面向联机分析处理的窗口子句，面向空间数据管理、文档数据管理的SQL语言扩展等。
- 在介绍WITH RECURSIVE子句之前，先了解WITH子句的用法

- **WITH子句的一般格式:**

```
WITH  RS1  [(<目标列>,<目标列>)] AS （SELECT 语句1）[,
           RS2  [(<目标列>,<目标列>)] AS （SELECT 语句2）,...]
SQL语句;   /*执行与RS1，RS2,...,相关的查询*/
```

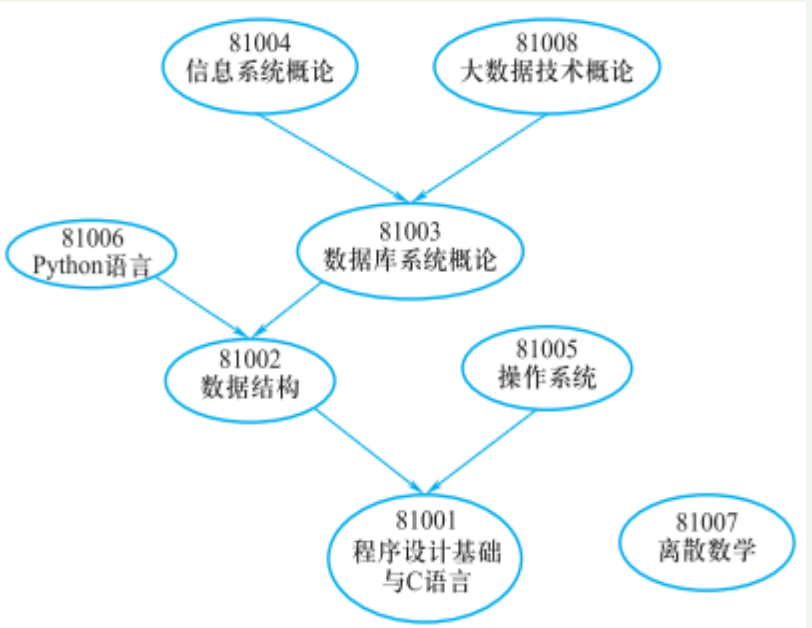
  - 作用：创建一个命名的临时结果集，该结果集仅在SQL语句（SELECT，INSERT或DELETE）执行时有效，不长期存储

1.2.1引入新的SQL子句(cont.)

厦門大學 XIAMEN UNIVERSITY

- **[例8.1]** 求81001-01和81001-02两个教学班之间学生选课平均成绩的差异

```
WITH
RS1(Grade)
    AS
    (SELECT AVG(Grade) FROM SC
     WHERE Teachingclass = '81001-01'),
RS2(Grade)
    AS
    (SELECT AVG(Grade) FROM SC
     WHERE Teachingclass = '81001-02')
SELECT RS1.Grade-RS2.Grade from RS1,RS2;
```



1.2.1引入新的SQL子句(cont.)

厦門大學 XIAMEN UNIVERSITY

- **WITH RECURSIVE 子句的一般格式:**

```
WITH  RECURSIVE RS  AS
      (SEED QUERY          /*初始化查询的临时结果集，记为L[1]*/
       UNION [ALL]         /*是否需要保留重复记录，加ALL为保留*/
       RECURSIVE QUERY     /*执行递归查询，得到全部临时结果集，即L[2]∪…∪L[i]*/
      )
SQL语句;                   /*执行与RS相关的查询*/
```

- **[例8.2](任务一)** 打印“数据库系统概论”课程的所有先修课信息

```
WITH RECURSIVE RS AS （
/*初始化RS，假设结果集为L[1]，即“数据库系统概论”的所有直接先修课*/
SELECT Cpno FROM Course WHERE Cname = '数据库系统概论'
UNION
/*递归查询第i层(i>=1)的数据，即第i-1层数据的直接先修课课程号，并更新RS*/
SELECT Course.Cpno FROM Course,RS WHERE RS.Cpno = Course.Cno ）
/*根据RS中记录的所有先修课程号，通过查找课程表，输出课程号与课程名*/
SELECT Cno, Cname FROM Course WHERE Cno IN (SELECT Cpno FROM RS);
```

1.2.2引入新的内置函数

厦門大學 XIAMEN UNIVERSITY

- SQL常用的内置函数可以分为：
  - 数学函数（如绝对值函数等）
  - 聚合函数（如求和、求平均函数等）
  - 字符串函数（如求字符串长度、求子串函数等）
  - 日期和时间函数（如返回当前日期函数等）
  - 格式化函数（如字符串转IP地址函数等）
  - 控制流函数（如逻辑判断函数等）
  - 加密函数（如使用密钥对字符串加密函数等）
  - 系统信息函数（如返回当前数据库名、服务器版本函数等）

1.2.2引入新的内置函数(cont.)

厦門大學 XIAMEN UNIVERSITY

- **[例8.3](任务二)** 打印一周内将过生日的学生信息

```
SELECT Sno, Sname, Ssex, Sbirthdate, Smajor
FROM Student
WHERE to_date(to_char(current_date,'yyyy')||'-'||to_char(Sbirthdate,'mm-dd'))
BETWEEN CURRENT_DATE AND CURRENT_DATE + INTERVAL '7' DAY;
```

- 内置函数 current_date 返回当前的系统日期，例如 2024-5-13
- 内置函数 to_char(current_data，‘yyyy’)返回当前系统日期的年份，例如 2024; to_char(Sbirthdate，'mm-dd')返回学生出生日期中具体的月份和日期，例如 5-13
- to_date(to_char(current_date,'yyyy') || '-' || to_char(Sbirthdate,'mm-dd'))表示把当前年份与出生日期用'-'连在一起。符号“||”用于把其左右两边的字符串连在一起
- 内置函数 to_date()的作用是将字符类型按一定格式转化为日期类型
- current_date + interval ‘7’ day是对当前的日期调整后的日期。参数interval是年(yyyy)、季度(q)、月(m)、日(d)、时(h)等粒度的时间单位。例如，current_date + interval 7day获得当前日期之后第七天的日期，即返回2024-5-20
- 通过执行此WHERE语句，判断学生表中每位学生转换后的出生日期是否在[2024-5-13，2024-5-20]区间内，如果是，打印该学生的信息

1.2.3引入PL/SQL与存储过程/函数

厦門大學 XIAMEN UNIVERSITY

- 内置函数是RDBMS默认创建的
- 还可以通过安装数据库的扩展插件添加该插件自带的内置函数
- 除了内置函数之外，RDBMS还支持PL/SQL、存储过程、函数等方法，使得用户可以自定义程序逻辑，开发完成业务逻辑复杂的应用系统。
  - 例如，在任务三中，需要用户自定义学分绩点的函数。在该函数中引入逻辑判断和循环控制，逻辑判断用于获取每门课程成绩对应的绩点，循环控制用于计算课程总学分和总学分绩点，最终计算得到平均学分绩点。

1.3通过高级语言实现复杂应用

厦門大學 XIAMEN UNIVERSITY

- 通过动态链接库调用的方式
  - RDBMS的功能被包装成一个子程序，由应用程序通过动态链接库调用来获得数据管理的功能
- 基于嵌入式SQL的方式
  - 将SQL嵌入到高级语言中混合编程，SQL语句负责操纵数据库，高级语言语句负责控制逻辑流程
- 基于ODBC/JDBC的中间件方式
  - 建立了连接不同数据库的一组规范。无论使用什么数据库，都采用同样的一组API来访问数据库

2.过程化SQL

厦門大學 XIAMEN UNIVERSITY

- SQL 99标准支持存储过程和函数的概念，商用RDBMS大多提供过程化的SQL，如Oracle的PL/SQL、Microsoft SQL Sever的Transact-SQL、IBM DB2的SQL PL等。
- 本节主要内容：
  - **过程化SQL的块结构**
  - **变量和常量的定义**
  - **流程控制**
  - **游标的定义与使用**
  - **存储过程**
  - **存储函数**

2.1过程化SQL的块结构

厦門大學 XIAMEN UNIVERSITY

- 过程化SQL程序的基本结构是块(block)。
  - 每个块完成一个逻辑操作
  - 块与块之间可以相互嵌套

可选

```
DECLARE
        /* 声明部分: 变量、常量、游标、异常等，定义的变量、常量等*/
        /*只能在该基本块中使用。当基本块执行结束时，定义就不再存在 */
```

必选

```
BEGIN
        /* 执行部分: SQL语句、过程化SQL的流程控制语句 */
EXCEPTION
        /* 异常处理部分: 错误处理 */
END;
```

```
declare
  v_ernp emp loyees%rowtype;
begin
  select * into v_emp from employees where employee_id = 1;
  dbms_output.put_line(v_emp.last_name || '-' || v_emp.department_id);
end;
```

2.2变量和常量的定义

厦門大學 XIAMEN UNIVERSITY

- 变量定义
  - 变量名 数据类型 [[NOT NULL]:=初值表达式]或
  - 变量名 数据类型 [[NOT NULL] 初值表达式]
- 常量定义
  - 常量名 数据类型 CONSTANT :=常量表达式
  - 常量必须要赋予一个值，并且该值在存在期间或常量的作用域内不能改变。如果试图修改它，过程化SQL将返回一个异常
- 赋值语句
  - 变量名称 :=表达式

2.3流程控制

厦門大學 XIAMEN UNIVERSITY

- 过程化SQL提供了流程控制语句，主要有：
  - 条件控制语句
  - 循环控制语句
- 语句的语法、语义和一般的高级语言类似。
- 主要介绍：
  - **条件控制语句**
  - **循环控制语句**
  - **错误处理**

2.3.1条件控制语句

厦門大學 XIAMEN UNIVERSITY

- 条件控制语句一般包含三种形式的IF语句：
  - IF-THEN

```
1. IF condition THEN
       Sequence_of_statements;
   END IF;
```

  - IF-THEN-ELSE

```
2. IF condition THEN
       Sequence_of_statements1;
   ELSE
       Sequence_of_statements2;
   END IF;
```

  - 嵌套的IF语句
    - 即，在THEN和ELSE子句中还可以再包含IF语句

2.3.2循环控制语句

厦門大學 XIAMEN UNIVERSITY

- 过程化SQL有三种循环控制语句：
  - LOOP语句、WHILE-LOOP语句和FOR-LOOP语句

```
LOOP
    Sequnce_of_stmts;
END LOOP;
```

- 最简单的循环语句LOOP格式
- 多数数据库服务器的过程化SQL都提供EXIT、BREAK或LEAVE等循环结束语句
- 保证LOOP语句块能够在适当的条件下提前结束

```
WHILE cond LOOP
    Sequnce_of_stmts;
END LOOP;
```

- 每次执行循环体语句之前，首先对条件进行求值
- 如果条件为真，则执行循环体内的语句序列
- 如果条件为假，则跳过循环并把控制传递给下一个语句

```
FOR count IN [REVERSE] bound1..bound2 LOOP
    Sequnce_of_stmts;
END LOOP;
```

- 将count设置为循环的下界bound1，检查它是否小于上界bound2。当指定REVERSE时则将count设置为循环的上界bound2，检查count是否大于下界bound1。如果越界则执行跳出循环，否则执行循环体，然后按步长(+1或-1)更新count的值，重新判断条件。

2.3.3错误处理

厦門大學 XIAMEN UNIVERSITY

- 如果过程化SQL在执行时出现异常，则应该让程序在产生异常的语句处停下来，根据异常的类型去执行异常处理语句
- SQL标准对数据库服务器提供什么样的异常处理做出了建议，要求过程化SQL管理器提供完善的异常处理机制

2.4游标的定义与使用

厦門大學 XIAMEN UNIVERSITY

- 在过程化SQL中，如果SELECT语句只返回一条记录，可以将该结果存放到变量中
- 当查询返回多条记录时，就要使用游标对结果集进行处理。一个游标与一个SQL语句相关联
- 游标的使用步骤：
  1. **声明游标**
  2. **打开游标**
  3. **使用游标**
  4. **关闭游标**

2.4游标的定义与使用(cont.)

厦門大學 XIAMEN UNIVERSITY

1.声明游标

DECLARE 游标名 [(参数1 数据类型, 参数2 数据类型, …)]
CURSOR FOR
SELECT语句;

- 定义游标仅仅是一条说明性语句，这时RDBMS并不执行SELECT语句

2.打开游标

OPEN 游标名[(参数1 数据类型, 参数2 数据类型, …)];

- 打开游标实际上是执行相应的SELECT语句，把查询结果取到缓冲区中。这时游标处于活动状态，指针指向查询结果集中的第一条记录

3.使用游标

FETCH 游标名 INTO 变量1[, 变量2,…];

- 变量必须与SELECT语句中的目标列表达式一一对应
- 用FETCH语句把游标指针向前推进一条记录，同时将缓冲区中的当前记录取出来送至变量供过程化SQL进一步处理
- 循环执行FETCH语句，逐条取出结果集中的行进行处理

4.关闭游标

CLOSE 游标名;

- 游标被关闭后就不再和原来的查询结果集相联系,但被关闭的游标可以再次被打开，与新的查询结果相联系

2.4游标的定义与使用(cont.)

厦門大學 XIAMEN UNIVERSITY

- **[例8.4]** 根据给定学号20180001，使用游标输出该学生的全部选课记录。

```
DECLARE
    CnoOfStudent CHAR(10);
    GradeOfStudent INT;
    mycursor CURSOR FOR
    SELECT Cno,Grade FROM SC WHERE Sno = '20180001';
BEGIN
    OPEN mycursor;                          /*打开游标*/
    LOOP                                    /*循环遍历游标*/
        FETCH mycursor INTO CnoOfStudent, GradeOfStudent;   /*检索游标*/
        EXIT WHEN mycursor%NOTFOUND;
        RAISE NOTICE 'Sno:20180001, Cno:%, Grade:%', CnoOfStudent, GradeOfStudent;
    END LOOP;
CLOSE mycursor;                             /*关闭游标*/
END;
```

2.5存储过程

厦門大學 XIAMEN UNIVERSITY

- 类似于高级语言程序，过程化SQL程序也可以被命名和编译，并保存在数据库中，称为存储过程(stored procedure)或存储函数(stored function)，供其他过程化SQL调用
- 存储过程或存储函数也是一类数据库的对象，需要有创建、删除等语句。这里的存储函数指自定义函数
- 存储过程的用户接口
  1. **创建存储过程**
  2. **执行存储过程**
  3. **修改存储过程**
  4. **删除存储过程**

2.5存储过程(cont.)

厦門大學 XIAMEN UNIVERSITY

1.创建存储过程

```
CREATE OR REPLACE PROCEDURE 过程名(
[[IN|OUT|INOUT] 参数1 数据类型,
[IN|OUT|INOUT] 参数2 数据类型, ...]
)                                    /*存储过程首部*/
AS <过程化SQL块>;                    /*存储过程体，描述该存储过程的操作*/
```

- 过程名：数据库服务器合法的对象标识
- 参数列表：存储过程提供了 IN、OUT、INOUT 三种参数模式，分别对应输入、输出、输入输出三种语义，不声明参数模式时，缺省为 IN 类型。输入参数在被调用时需要指定参数值，输出参数调用时不传入参数值，而是作为返回值返回。输入输出参数调用时需要传入初始值，并会返回操作后的最终值。参数列表中需要指定参数模式、参数名、以及参数的数据类型
- 过程体：是一个<过程化SQL块>，包括声明部分和可执行语句部分

2.5存储过程(cont.)

厦門大學 XIAMEN UNIVERSITY

- **[例8.5]** 给定学生学号，计算学生的平均学分绩点GPA
- 求解思路为：
  - 给定学生学号，找出该学生所有选修课程的学分、成绩
  - 根据每门课程的成绩，参照成绩和绩点对照表，确定该成绩所处的范围，找出该门课程对应的学分绩点

学号 Sno	课程号 Cno	成绩 Grade	选课学期 Semester	教学班 Teachingclass
20180001	81001	85	20192	81001-01
20180001	81002	96	20201	81002-01
20180001	81003	87	20202	81003-01

编码	成绩下限	成绩上限	绩点
1	0	59	0
2	60	69	1
3	70	79	2
4	80	89	3
5	90	100	4

  - “81001”课程考试成绩为85分，85分对应成绩范围为[80,89]，该门课程对应的学分绩点为3
  - 类似地，课程号为“81002”对应的学分绩点为4；课程号为“81003”对应的学分绩点为3
  - 81001-81003三门课程的学分都是4。
  - 根据平均学分绩点GPA的计算公式=总学分绩/总学分=（每门课程的学分*对应课程的绩点）的总和/12 = (3*4 + 4*4 + 3*4)/12 = 3.33

```
CREATE OR REPLACE PROCEDURE compGPA(     /*定义存储过compGPA*/
                IN inSno CHAR(10),                  /*输入参数：学生学号inSno*/
                OUT outGPA FLOAT)                   /*输出参数：平均学分绩outGPA*/
                AS
                DECLARE
                     courseGPA INT;      /*声明变量courseGPA，临时存储课程学分绩 */
                     totalGPA INT;       /*声明变量totalGPA，临时存储总学分绩 */
                    totalCredit INT;     /*声明变量totalCredit，临时存储总学分*/
                    grade INT;           /*声明变量grade，临时存储学生成绩 */
                    credit INT;          /*声明变量credit ，临时存储课程学分 */
                    mycursor CURSOR FOR      /*声明游标mycursor */
                     SELECT Ccredit, grade FROM SC, Course   WHERE Sno = inSno and SC.Cno = Course.Cno;
               BEGIN
                      totalGPA := 0;
                      totalCredit := 0;
                      OPEN mycursor;                        /*打开游标mycursor */
               LOOP                                         /*循环遍历游标*/
                      FETCH mycursor INTO credit, grade;    /*检索游标*/
                      EXIT WHEN mycursor%NOTFOUND;
                      IF grade BETWEEN 90 AND 100 THEN courseGPA := 4.0;
                          ELSIF grade BETWEEN 80 AND 89 THEN courseGPA := 3.0;
                          ELSIF grade BETWEEN 70 AND 72 THEN courseGPA := 2.0;
                          ELSIF grade BETWEEN 60 AND 69 THEN courseGPA := 1.0;
                          ELSE courseGPA := 0;
                     END IF;    /*参照表8.2，根据成绩找出某门课程对应的学分绩点*/
                    totalGPA := totalGPA + courseGPA * credit;
                     totalCredit := totalCredit + credit;
                END LOOP;
                CLOSE mycursor;          /*关闭游标mycursor */
                outGPA:= 1.0 * totalGPA / totalCredit;
                END;
```

2.5存储过程(cont.)

厦門大學 XIAMEN UNIVERSITY

- **2.执行存储过程**

  CALL/PERFORM [PROCEDURE] 过程名([参数1,参数2,...]);

  - 使用CALL或者PERFORM等方式激活存储过程的执行
  - 在过程化SQL中，数据库服务器支持在过程体中调用其他存储过程

- **[例8.6]** 查询学号为“20180001”学生的课程GPA

```
DECLARE outGPA FLOAT;
BEGIN
    CALLcompGPA('20180001',outGPA);
    RAISE NOTICE 'GPA: %', outGPA;
END;
```

  - 在调用含有输入参数和输入输出参数的存储过程时，需要指定具体的参数值。在调用含有输出参数的存储过程时，对应位置不需要传入参数值，但需要事先定义输出变量

2.5存储过程(cont.)

厦門大學 XIAMEN UNIVERSITY

- **3.修改存储过程**
  - 重命名

    ALTER PROCEDURE 过程名1 RENAME TO 过程名2;
  - 重新编译

    ALTER PROCEDURE 过程名 COMPILE;
- **4.删除存储过程**

  DROP PROCEDURE 过程名;
- 存储过程的优点
  1. 运行效率高
  2. 降低了客户机和服务器之间的通信量
  3. 方便实施企业规则

2.6存储函数

厦門大學 XIAMEN UNIVERSITY

- 存储函数也称为自定义函数，区别于RDBMS的内置函数。
- **存储函数和存储过程的异同**
  - 同：都是持久性存储模块
  - 异：函数必须指定返回的类型

1. 函数的定义语句格式

CREATE OR REPLACE FUNCTION 函数名([参数1 数据类型, 参数2数据类型, ...]) RETURNS<类型>
AS <过程化SQL块>;

2. 函数的执行语句格式

CALL/SELECT 函数名 ([参数1,参数2,...]);

3. 修改函数

- 重命名

  ALTER FUNCTION 函数名1 RENAME TO 函数名2;

- 重新编译

  ALTER FUNCTION 函数名 COMPILE;

3.JDBC编程

厦門大學 XIAMEN UNIVERSITY

- 本节介绍如何进行JDBC编程，即如何使用Java语言调用JDBC来进行数据库应用程序的设计。

- 主要内容：
  - **JDBC工作原理概述**
  - **JDBC APIs基础**
  - **使用JDBC操纵数据库的工作流程**

3.1JDBC工作原理概述

厦門大學 XIAMEN UNIVERSITY

- **JDBC** (Java Database Connection)
  - 由于不同的数据库管理系统的存在，在某个RDBMS下编写的应用程序就不能在另一个RDBMS下运行
  - 许多应用程序需要共享多个部门的数据资源，访问不同的RDBMS
  - JDBC是面向Java 语言的软件开发工具包(JDK)中有关数据库的一个组成部分
    - 提供了一组访问数据库的应用程序编程接口（Application Programming Interface，API）
      - JDBC接口提供的功能必须完备
      - 使用JDBC提供的接口而不是RDBMS的接口来进行应用开发
- **JDBC约束力**
  - 规范应用开发
  - 规范关系数据库管理系统应用接口
    - RDBMS必须实现所有JDBC接口

3.1JDBC工作原理概述(cont.)

厦門大學 XIAMEN UNIVERSITY

- JDBC应用系统的体系结构
  1. 用户应用程序
  2. JDBC驱动程序管理器
  3. 数据源

图8.5 JDBC应用系统的体系结构

3.2JDBC APIs基础

厦門大學 XIAMEN UNIVERSITY

- JDBC APIs是各个数据库厂商提供的JDBC应用程序开发接口都要遵循的标准。
- **1.JDBC中的常用类**
  - JDBC进行应用程序开发涉及到的所有类都包含在java.sql包中。不同JDBC版本接口名和使用略有差异

类名	路径	备注
驱动程序类	java.sql.Driver	由各数据库厂商提供
驱动程序管理类	java.sql.DriverManager	作用于应用程序与驱动程序之间
数据库连接类	java.sql.Connection	用于建立与指定数据库的连接
静态SQL语句执行类	java.sql.Statement	用于执行静态SQL语句并返回结果
动态SQL语句执行类	java.sql.PreparedStatement	用于执行含参SQL语句并返回结果
存储过程语句执行类	java.sql.CallableStatement	用于执行存储过程语句并返回结果
结果集处理类	java.sql.ResultSet	用于检索结果集中的数据

- **2.数据类型**

  SQL数据类型：用于数据源

  - VARCHAR、 CHAR 、 BIT 、NUMERIC、DATE等

  Java数据类型：用于应用程序的Java代码

  - String、boolean、BigDecimal、byte、short、int等

3.2JDBC APIs基础(cont.)

厦門大學 XIAMEN UNIVERSITY

SQL数据类型 java.sql.types	Java数据类型 java.sql或者java.lang	Oracle	SQL Server
CHAR	java.lang.character	CHAR	CHAR
VARCHAR	java.lang.String	VARCHAR2	VARCHAR
DATE	java.sql.Date	DATE	DATETIME
TIMESTAMP	java.sql.Timestamp	TIMESTAMP(9)	DATETIME
CLOB	java.lang.string	CLOB	NTEXT
BLOB	byte[] / Serializable	BLOB	IMAGE
BIT	boolean	SMALLINT	BIT
SMALLINT	short	SMALLINT	SMALLINT
INTEGER	int	INTEGER	INTEGER
BIGINT	long	NUMBER	NUMERIC
REAL	REAL	REAL	REAL
DOUBLE	double	DOUBLE PRECISION	FLOAT
DECIMAL(p,s)	java.math.BigDecimal	NUMBER(p,s)	DECIMAL(p,s)

3.2JDBC APIs基础(cont.)

- 由DBMS的驱动程序完成自身数据类型和JDBC标准数据类型的映射.
  - 当用户的应用程序通过JDBC APIs从数据库中存取数据时，java程序所使用的变量类型需要与数据库中关系模式属性的类型建立映射关系。
    - 如，通过结果集处理对象中的get方法从数据库中获得结果集，并把记录中的属性值赋给应用程序变量；
    - 为动态SQL语句执行对象绑定应用程序变量。

- SQL数据类型和java数据类型之间的转换规则

数据类型	SQL数据类型	Java数据类型
SQL数据类型	数据源之间转换	应用程序变量传递给语句参数
Java数据类型	从数据库中读取数据赋值给应用程序变量	应用程序变量之间转换

3.1使用JDBC操纵数据库的工作流程

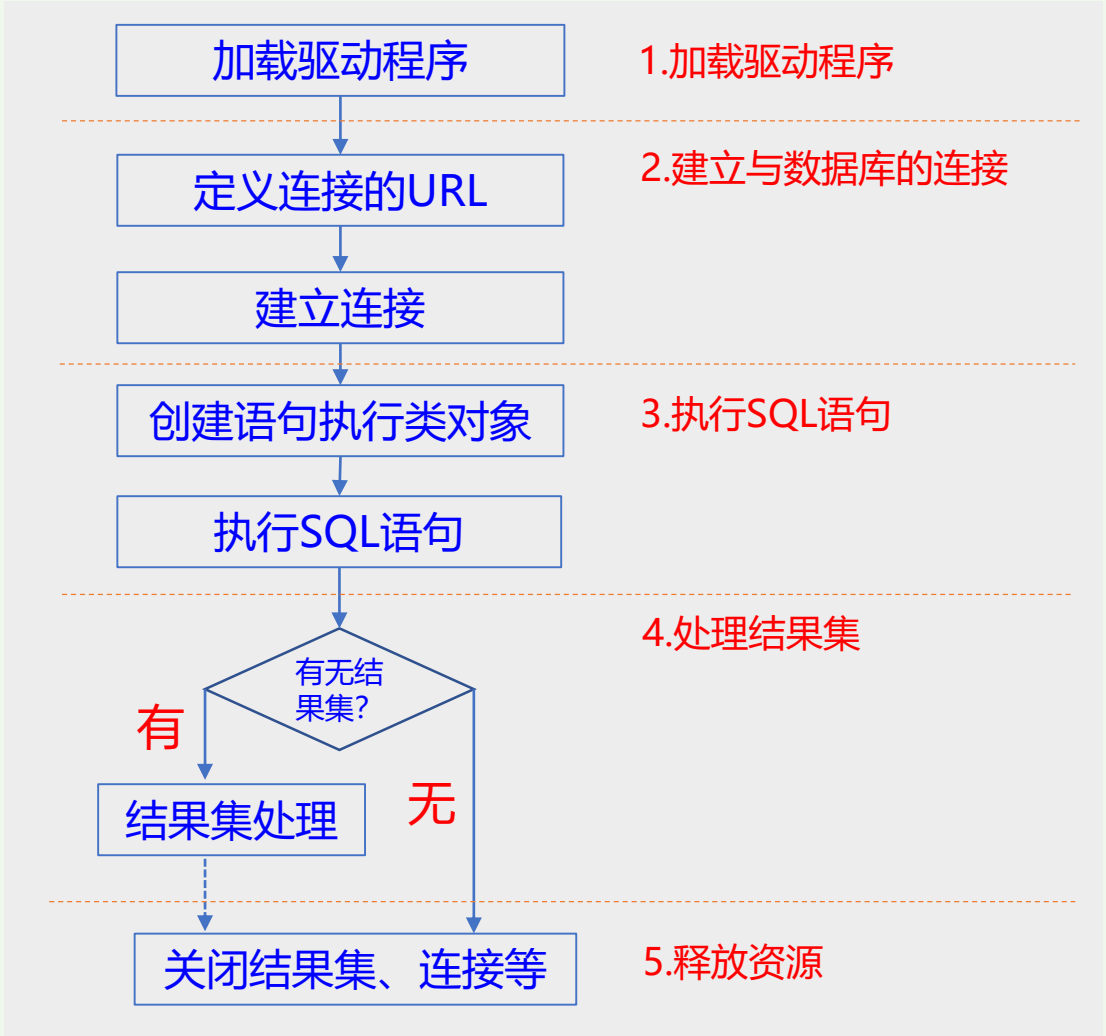


图8.6 使用JDBC操纵数据库的工作流程

3.1使用JDBC操纵数据库的工作流程(cont.)

- **步骤1：加载驱动程序**
  - 驱动程序在JDBC API中实现定义数据交互的接口

  [例8.7] 对Kingbase、Oracle、SQL Server加载数据库驱动

```
Class.forName("com.kingbase.Driver");        /* Kingbase */
Class.forName("oracle.jdbc.OracleDriver");   /* Oracle */
Class.forName("com.microsoft.jdbc.sqlserver.SQLServerDriver"); /* SQL Server */
```

  - JDBC 4.0及以后版本不再需要使用Class.forName() 显式地加载JDBC 驱动程序

- **步骤2：设置待连接数据库的URL，并建立与数据库连接**
  - 加载驱动后，可以通过URL地址与数据库建立连接
  - URL包含了连接数据库所需的协议、子协议和数据库名称，定义格式为：

    <协议名>:<子协议名>:<数据库名称>
  - JDBC连接中的协议名称总是jdbc，子协议名和数据库名称由具体连接的DBMS决定

3.1使用JDBC操纵数据库的工作流程(cont.)

- **[例8.8]** 定义与Kingbase、Oracle、SQL Server数据库连接的URL

```
strURL = "jdbc:kingbase://" + 服务器名 +  ":" + 端口号 + "/" + 数据库名;
strURL = "jdbc:oracle:thin:@" + 服务器名 + ":" + 端口号 + ":" + 数据库名
strURL = "jdbc:microsoft:sqlserver://" + 服务器名 + ":" + 端口号 + ":" + 数据库名
```

  - Kingbase、Oracle、SQL Server的默认端口号分别为54321、1521、1433

- 利用生成的URL建立与数据库的连接。连接数据库时需要向java.sql.DriverManager类请求并创建一个数据库连接类(Connection)对象。使用java.sql.DriverManager类提供的静态方法getConnection，传入指定的连接URL、数据库用户名和密码建立连接。

- **[例8.9]** 建立与Kingbase数据库的连接，假定服务器地址为192.168.0.118，端口为54321，数据库名为DB-Student，用户名为Info001，密码为123456

```
String strURL ="jdbc:kingbase:// 192.168.0.118:54321/DB-Student";
Connection conn= DriverManager.getConnection(strURL," Info001","123456");
```

3.1使用JDBC操纵数据库的工作流程(cont.)

- **步骤3：执行SQL语句**

  1.创建语句执行类对象

  – 建立数据库连接后，就可以开始操纵数据库了。JDBC提供了语句执行类对象用来执行用户提交的SQL语句，并接受SQL查询的结果集。

  – 利用数据库连接类的createStatement方法创建静态语句执行类对象(Statement)，静态语句执行类对象只能用于执行静态的SQL语句。

  – 静态语句执行类还派生出：

    - 动态语句执行类(PreparedStatement)：执行动态的SQL语句
      - 经过预编译，可有效防止SQL注入攻击，提高数据库的安全性
    - 和存储过程执行类(CallableStatement)：专门执行数据库的存储过程

3.1使用JDBC操纵数据库的工作流程(cont.)

- **步骤3：执行SQL语句**

  2.三种执行SQL语句的方法

  - ResultSet executeQuery():
    - 执行数据库查询语句，成功执行后返回一个ResultSet类(结果集)对象
  - int executeUpdate(): 处理增、删、改以及定义语句
    - 用来处理增、删、改及数据库定义语句，成功执行后返回一个int类型值
  - boolean execute(): 处理存储过程或动态SQL语句
    - 用来处理存储过程或动态SQL语句，成功执行后返回一个boolean类型值

3.1使用JDBC操纵数据库的工作流程(cont.)

- **步骤3：执行SQL语句**
  - **[例8.10]** 使用JDBC向课堂评价表中插入一条记录。课程评价关系模式为ClassAssess(Sno,Tno,Tcno,Assess,Catpye,Feedback)

```
/*创建PreparedStatement类对象*/
PreparedStatement stmt = conn.prepareStatement("INSERT INTO ClassAssess
VALUES(?,?,?,?,?,?)");
/* 生成PreparedStatement类对象中的动态参数，注意第六个字段Feedback，未设置输入值 */
stmt.setString(1, "20180001");          /*设置学生学号*/
stmt.setString(2, "19950018");          /*设置职工号*/
stmt.setString(3, "81001-01");          /*设置教学班号*/
stmt.setString(4, “老师讲得很出色");    /*设置学生评价意见*/
stmt.setBoolean(5,true);                /*设置学生评价意见类型*/
stmt.executeUpdate();
```

3.1使用JDBC操纵数据库的工作流程(cont.)

- **步骤3：执行SQL语句**
  - **[例8.11]** 利用例8.5创建的compGPA存储过程，使用JDBC查询学号为20180001同学的平均学分绩点。

```
CallableStatement stmt=conn.prepareCall( "{CALL compGPA(?,?)}" );
/*创建存储过程执行类对象*/
/* 生成CallableStatement类对象中的动态参数 */
stmt.setString(1, "20180001" );              /*设置输入参数*/
stmt.registerOutParameter(2, Types.REAL);    /*设置输出参数*/
stmt.execute( );                             /*执行存储过程*/
float fGPA = stmt.getFloat(2);               /*获取输出参数fGPA*/
```

3.1使用JDBC操纵数据库的工作流程(cont.)

- **步骤4：处理结果集**
  - 使用executeQuery方法执行数据库查询语句，成功执行后返回一个结果集类对象(ResultSet)。
    - 通过遍历结果集类对象中的每一个元组，执行用户定义的应用逻辑。
    - 通过ResultSet类提供的getXXX()方法来获得结果集中元组的属性值。XXX代表某种数据类型
    - 当用户想要获取某种数据类型的某列值时，可以指定参数为列号(JDBC的列从1开始)或列名
  - JDBC使用游标来处理结果集数据。三种游标类型：
    - TYPE_FORWARD_ONLY：只能向下滚动(默认类型)
    - TYPE_SCROLL_IN(SEN)SENSITIVE：双向滚动，区别是前者不会同步更新数据库中的操作，后者会在结果集类对象中及时跟踪数据库的更新操作
  - JDBC游标的打开方式不同于嵌入式SQL，不是显示声明而是由系统自动产生。当结果集刚刚生成时，游标指向第一行数据之前，通过next()、previous()等操作来移动游标获取结果集中的每一行数据。

3.1使用JDBC操纵数据库的工作流程(cont.)

- **步骤4：处理结果集**

[例8.12]遍历教师职工号为19950018，教学班号为81001-01的学生课程评价详情。

```
String SQL = "SELECT Sno, Assess, CAtype, Feedback FROM ClassAssess
                    WHERE Tno='19950018' AND TCo = '81001-01'";
ResultSet rs = stmt.executeQuery(SQL);
while(rs.next()){
        String Sno = rs.getString("Sno");                    /*等价于rs.getString(1)*/
        String strAssess = rs.getString("Assess");           /*等价于rs.getString(2) */
        Boolean bCAtype= rs.getBoolean ("CAtype");    /*等价于rs.getBoolean (3) */
        String strFeedback = rs.getString("Feedback"); /*等价于rs.getString(4) */
         /*打印课程评价详情，注意隐藏学生的信息，如学生的学号*/
        System.out.printf("[%s,%b,%s]%n", strAssess, bCAtype, strFeedback);
    }
```

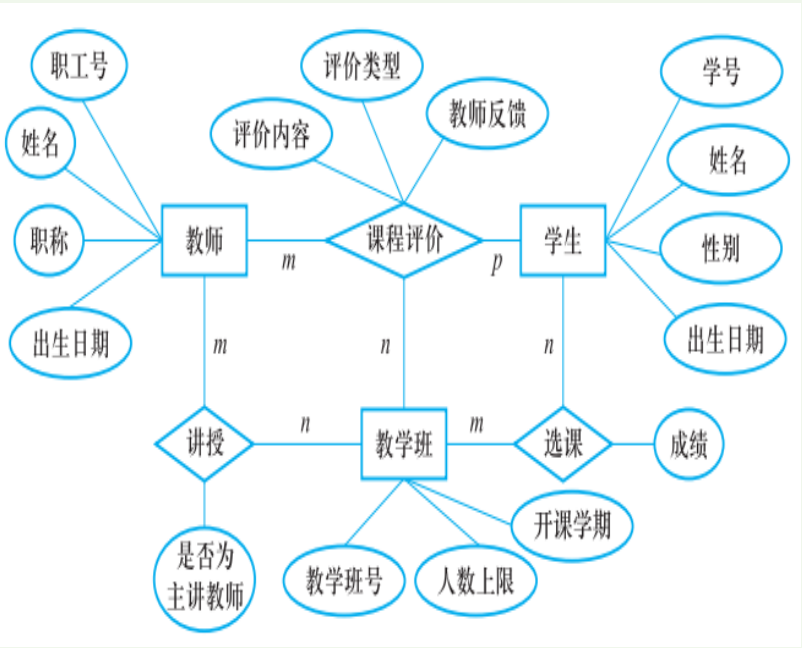
3.1使用JDBC操纵数据库的工作流程(cont.)

厦門大學 XIAMEN UNIVERSITY

- **步骤5：释放资源**
  - 处理结束后，应用程序将使用close方法释放与数据库交互的对象资源。
  - 释放资源的标准顺序：
    1. 关闭结果集：ResultSet.close()
    2. 关闭语句执行类对象：Statement.close()
    3. 释放数据库连接对象：Connection.close()

[例8.13] 基于JDBC实现任务4

- 选课关系模式：SC(Sno,TCno, Grade)
- 教师关系模式：Teacher(Tno, Tname, Ttitle, Tbirthdate, Dno)
- 教学班关系模式：TeachingClass(TCno, Capacity, Semester, Cno)
- 讲授关系模式为：Teaching(TCno, Tno, isLeading)
- 课堂评价关系模式为：ClassAssess(Sno, Tno, TCno, Assess,CAtype, CRfeedback)



3.1使用JDBC操纵数据库的工作流程(cont.)


```
public class ConnectionManager{
    static final String jdbcUrl ="jdbc:kingbase://192.168.0.118:54321/DB-Student";
    static final String jdbcUsername = "Info001";
    static final String jdbcPassword = "123456";
    private static Connection connection = null;
    /*静态代码段,加载 Kingbase 数据库驱动程序*/
    static{
        Class.forName("com.kingbase.Driver");
    }
    public static Connection createConnection(){
        connection=DriverManager.getConnection(jdbcUrl,jdbcUsername,jdbcPassword);
        System.out.println("数据库连接成功...");
        return connection;
    }
    /*释放资源*/
    public static void release(){
        if (connection!=null) connection.close();
        System.out.println("释放资源成功...");
    }
}
```

1.数据库连接的管理

- 以金仓数据库为例
- 设置连接数据库的URL
- 创建连接数据库的静态函数
- 创建释放数据库连接对象的静态函数

3.1使用JDBC操纵数据库的工作流程(cont.)

- **2.学生浏览指定的教学班和授课教师，并输入课程评价。**

```
Connection conn = ConnectionManager.createConnection( );
Statement stmnt = conn.createStatement( );
String SQL4ClassAssess = “SELECT Tcno, Tno FROM Student, Teacher, Teaching, SC WHERE Teacher.Tno = Teaching.Tno
AND Teaching.TCno = SC.Tcno AND SC.Sno = ” +strSno;
ResultSet rs = stmnt.executeQuery(SQL4ClassAssess);
String insertSQL = “INSERT INTO ClassAssess(Sno, Tno, Tcno, Assess, CAtype) VALUES(?, ?, ?, ?, ?)”
PreparedStatement ps = conn.prepareStatement(insertSQL);
while(rs.next( )){
        String strTno = rs.getString(“Tno”);
        String strTCno = rs.getString(“TeachingClassNo”);
        ps.setString(1,strSno);
        ps.setString(2,strTno);
        ps.setString(3,strTCno);
        ps.setString(4,”老师讲得很出色”);
        ps.setBoolean(5, true);
        ps.executeUpdate( );
}
rs.close( );
ps.close( );
stmnt.close( );
ConnectionManager.release( );
```

3.1使用JDBC操纵数据库的工作流程(cont.)

- **3.教师浏览教学班学生的评价意见，并针对每条评价逐一回复。**

```
Connection conn = ConnectionManager.createConnection( );
Statement stmnt = conn.createStatement( );
ResultSet rs = stmnt.executeQuery("SELECT Sno, Assess, Catype, Feedback FROM ClassAssess WHERE
Tcno="+strTCno+" AND Tno =" +strTno);
String updateSQL = "UPDATE  ClassAssess SET Feedback =? WHERE Sno=? AND Tno=? AND Tcno=?";
PreparedStatement ps = conn.prepareStatement(updateSQL);
while(rs.next( )){
        String strSno = rs.getString("Sno");
        String strAssess = rs.getString("Assess");
        Boolean bCAtype = rs.getString("CAtype");
        String strFeedback = rs.getString("Feedback");
        System.out.printf("[%s, %b,%s] %n",strAccess, bCAtype, strFeedback);
            ps.setString(1,"感谢你的评价" );
            ps.setString(2,strSno);
            ps.setString(3,strTno);
            ps.setString(4,strTCno);
            ps.executeUpdate( );
}
rs.close( );
ps.close( );
stmnt.close( );
ConnectionManager.release( );
```

4.基于MVC框架的数据库应用

- MVC(Model-view-controller,模型-视图-控制器模式)是一种使用业务模型(model)、用户结果呈现(view)及业务模型控制器(controller)来进行Web数据库应用开发的编程框架。
- 采用基于MVC框架进行web数据库应用开发，其核心思想在于功能的解耦及功能的模块化。
- MVC框架的三个部分每个部分各自处理自己的工作，互不干扰。

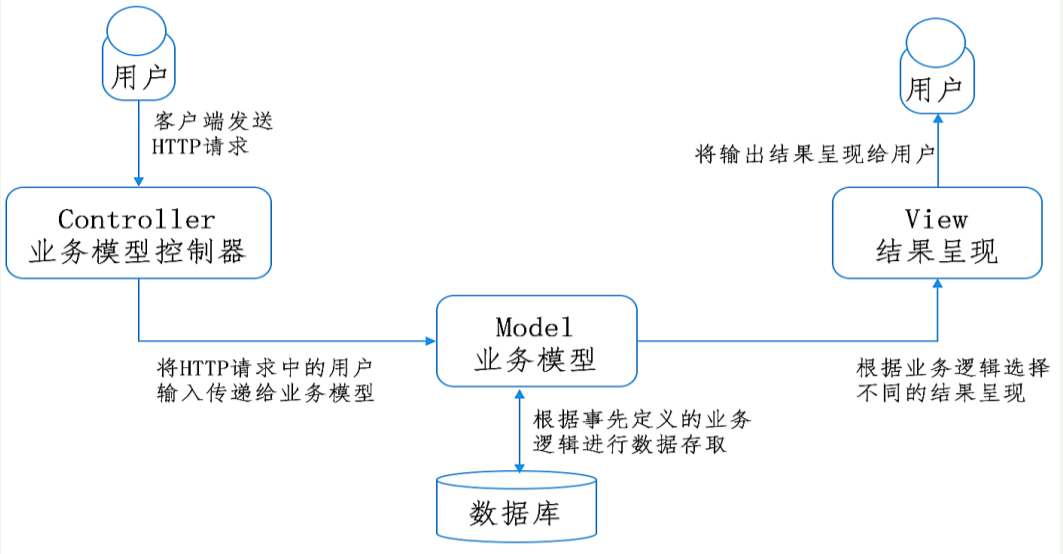
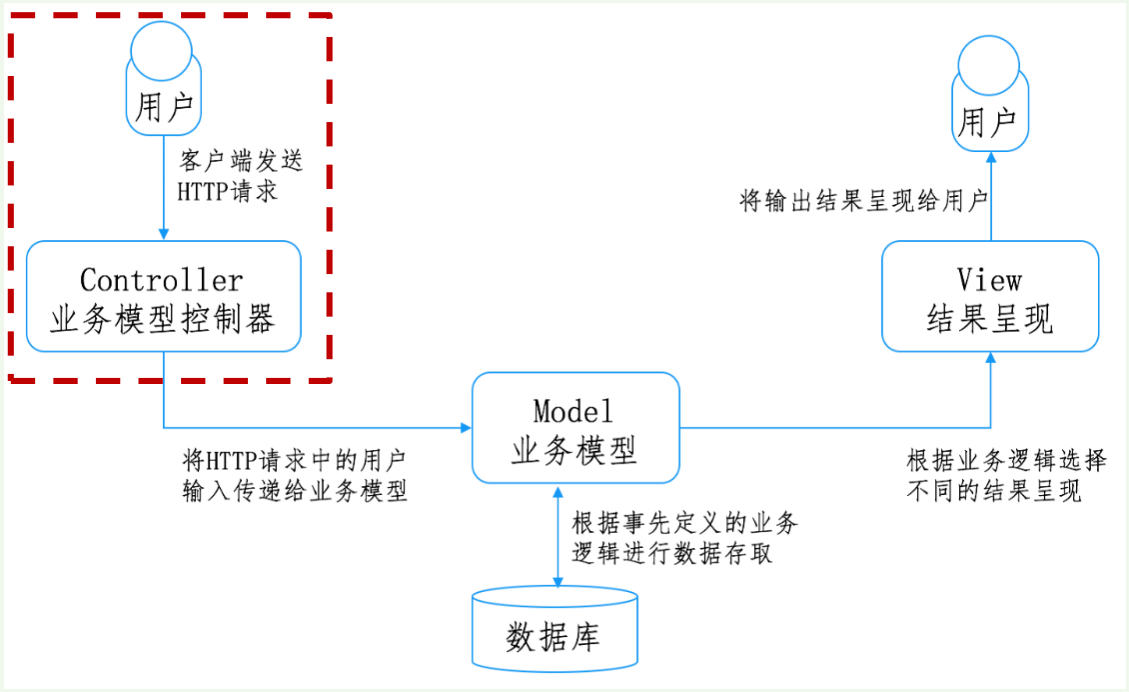


图8.8 MVC框架工作流程图

4.基于MVC框架的数据库应用(cont.)

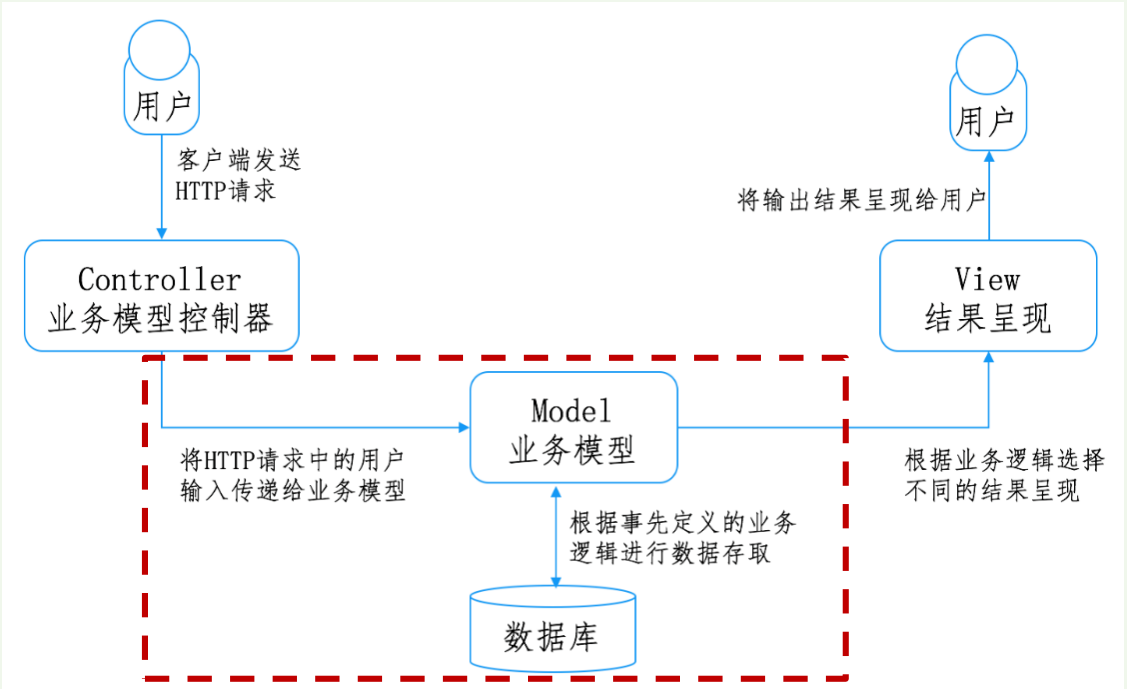
- **业务模型控制器:**
  - 接收用户通过浏览器发送的HTTP服务请求
    - 包括用户需要选择的业务模型，以及其它可选的输入参数（可以是用户在文本框中输入的文本、下拉列表框中的数据项等，也可以是用户的操作类型）



4.基于MVC框架的数据库应用(cont.)

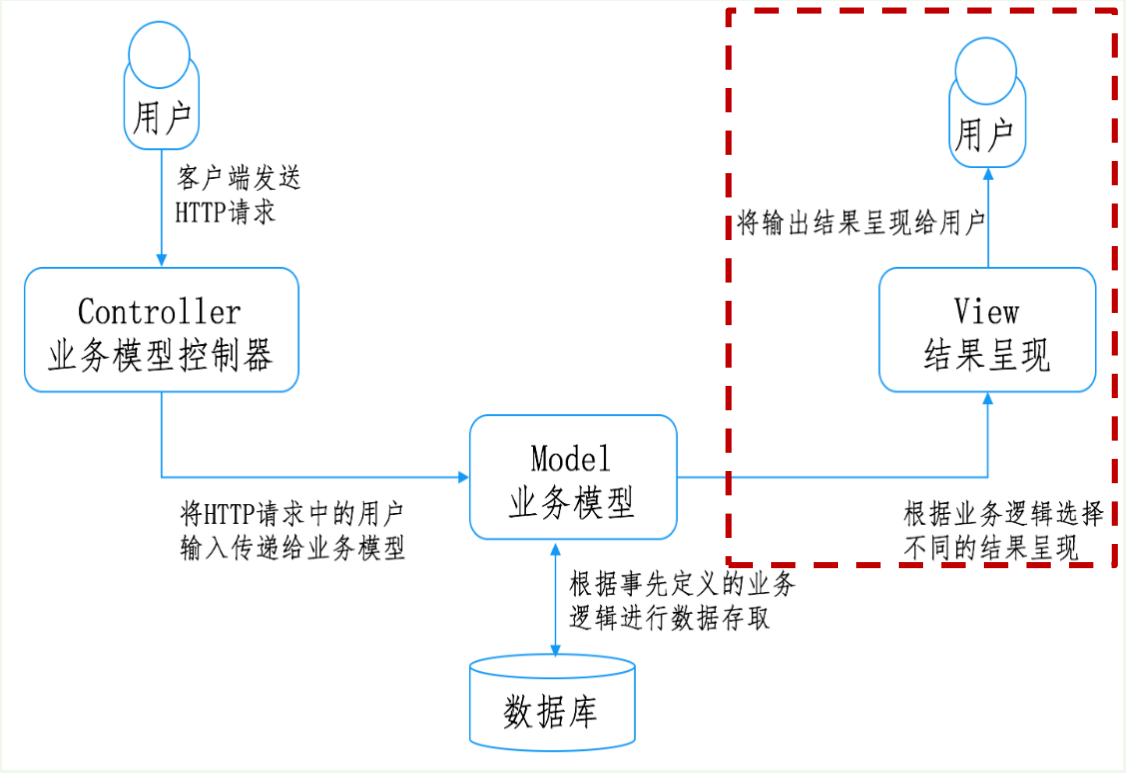
厦門大學 XIAMEN UNIVERSITY

- **业务模型:**
  - 建立用户的业务逻辑。
    - 解析出来的用户数据，执行事先定义的业务逻辑，并操纵数据库存取数据。
    - 输出独立于具体的数据格式，可为多个用户结果呈现提供展示所需要的数据，减少了代码的重复性。



4.基于MVC框架的数据库应用(cont.)

- **用户结果呈现:**
  - 用户看到并与之交互获得结果展示的界面。根据业务模型输出的结果反馈给客户端进行呈现。



4.基于MVC框架的数据库应用(cont.)

- **[例8.14]** 使用MVC框架实现任务4.

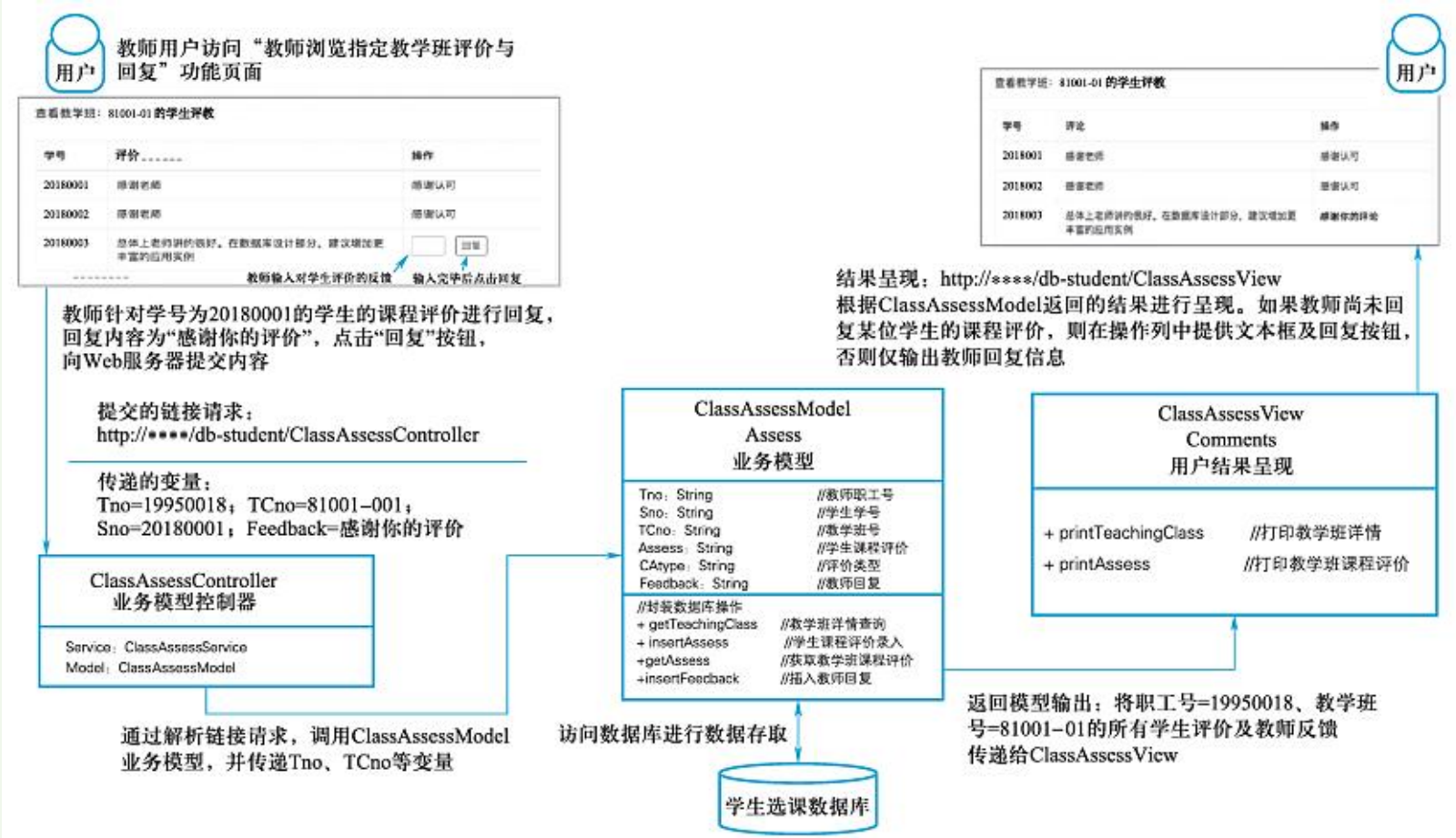


图 8.9　使用 MVC 框架实现任务 4 的系统流程图

5.本章小结

厦門大學 XIAMEN UNIVERSITY

- 扩展SQL语言的功能
  - 引入新的SQL子句、新的内置函数、PL/SQL以及存储过程和存储函数等技术
  - SQL与高级语言具有不同的数据处理方式。
    - SQL是面向集合的，而高级语言是面向记录的
    - 游标就是用来协调这两种不同的处理方式的机制
- 通过高级语言实现复杂应用
  - JDBC的工作原理和工作流程
  - 基于MVC框架的开发方式

本章作业

- 自行选做教材第1题，但无需提交。