数据库技术与应用期末小结

葛乾

December 12, 2022

1 简介

- 数据库 (DB)，数据库系统（DBS）、数据库管理系统（DBMS）的概念
  - 什么是数据？
  - 数据库是指长期存储在计算机内的、有结构的、大量的、可共享的数据集合。
  - 数据库系统是指计算机系统引入数据库后的系统构成，是一个具有管理数据库功能的计算机软硬件综合系统。数据库系统可以实现有组织地、动态地存储大量数据、提供数据处理和资源共享的服务。
  - 数据库管理系统是位于用户与操作系统之间的一层数据管理软件，在数据库建立、运用和维护时对数据库进行统一控制、统一管理，使用户能方便地定义数据和操纵数据，并能够保证数据的安全性、完整性、多用户对数据的并发使用及发生故障后的系统恢复。
- DB，DBS，DBMS 三者之间的关系：DBS 包括 DB 和 DBMS
- 数据库系统的特点
  - 结构化: 结构化的数据是指具有固定格式或有限长度的数据，数据的整体结构化是数据库的主要特征之一；记录的结构和记录间的联系由数据库管理系统描述，无需应用程序定义
  - 共享性：数据面向整个系统，可以被多个用户、多个应用共享使用；好处是（1）减少数据冗余，节约存储空间，（2）避免数据之间的不相容性与不一致性，（3）使系统易于扩充
  - 独立性：物理独立性，用户的应用程序与数据库中数据的物理存储相互独立。数据的物理存储改变时，应用程序不用改变；逻辑独立性，用户的应用程序与数据库的逻辑结构相互独立。数据的逻辑结构改变时，应用程序不用改变。
  - DBMS 统一控制：DBMS 提供的数据控制功能，包括数据的安全性保护，数据的完整性检查，并发控制，数据库恢复
- 数据模型
  - 概念模型，逻辑模型和物理模型（DBMS 系统支持）
    - 概念模型，也称信息模型。它是按用户的观点来对数据和信息建模，用于数据库设计
    - DBMS 系统支持的模型：逻辑模型主要包括网状模型、层次模型、关系模型、面向对象数据模型、对象关系数据模型、半结构化数据模型等。按计算机系统的观点对数据建模，用于 DBMS 实现；物理模型是对数据最底层的抽象，描述数据在系统内部的表示方式和存取方法，在磁盘或磁带上的存储方式和存取方法
  - 组成要素: 数据结构、数据操作、完整性约束
- 常用的数据模型：层次模型；网状模型；关系模型；面向对象数据模型；对象关系数据模型；半结构化数据模型
- 数据库系统的结构

  - 外模式？模式？内模式？
  - 二级映像的概念与各自的作用
- 型与值

2 关系数据库基础

- 数据结构相关概念：域、码、笛卡尔积
- 关系模型: 资料以“关系”的形式表示，也就是以二维表的形式表示，其数据模型就是所谓的关系模型。在关系模型中，无论是从客观事物中抽象出的实体，还是实体之间的联系，都用单一的结构类型——关系来表示
- 基本关系的性质：列是同质的，不同的列可出自同一个域。其中的每一列称为一个属性；不同的属性要给予不同的属性名，列的顺序无所谓，列的次序可以任意交换，任意两个元组的候选码不能相同，行的顺序无所谓，行的次序可以任意交换，分量必须取原子值（不可再分）
- 关系模式与关系辨析：关系模式是对关系的描述，静态的、稳定的，是型；关系是关系模式在某一时刻的状态或内容，动态的、随时间不断变化的，是值。
- 关系操作
  - 有哪些操作？
  - 特点是集合操作方式：操作的对象和结果都是集合，一次一集合的方式
- 完整性约束
  - 实体完整性：实体可区分，主码不为空
  - 参照完整性：外码？参照属性上的值为空值或者被参照属性上的某一值
  - 用户定义的完整性
- 关系代数
- 规范化理论
  - 函数依赖？部分函数依赖，完全函数依赖，传递函数依赖？
  - 第一范式：关系的每个分量都是不可分的数据项
  - 第二范式：关系符合 1NF，且其中的每一非主属性完全函数依赖于候选码（主属性）。消除了非主属性对侯选码的部分依赖
  - 第三范式：从 2NF 转为 3NF，消除了非主属性对侯选码（主属性）的传递依赖

3 SQL

- SQL 的特点：综合统一，高度非过程化，面向集合的操作方式，以同一种语法结构提供多种使用方式（既可单独使用，也可以嵌入高级语言），语言简洁，易学易用
- 数据定义：模式/表/索引分别通过哪些语句建立/删除/修改
- SQL 的功能
  - 数据定义：可用于定义 SQL 模式、基本表、视图和索引
  - 数据操纵：可分成数据查询和数据更新两类，其中数据更新又分为插入、删除和修改三种操作
  - 数据控制：包括对基本表和视图的授权，完整性规则的描述

- MySQL 中常用数据类型：int, smallint, float, double, decimal, char, varchar, date, datetime 等的使用场景
- 数据查询
  - 单表查询
    - 语句：select [属性] from [表][子查询] where [条件]
    - select, from, where, group by, having, order by 等关键词的作用分别是什么？
    - 确定范围 ([not] between...and...) 与确定集合 ([not] in)
    - 字符匹配（% 用于匹配任意长度字符串，_ 用于匹配单个字符）
    - 空值查询 (is [not] null，注意此处 is 不能用 = 代替)
    - 多重查询 (and 与 or，and 优先级高于 or)
    - 排序（order by）
    - 聚集函数: where 语句中不能使用聚集函数
    - 分组（having...group by），having 仅与 group by 连接。having 与 where 区别（作用对象不同，having 用于分组，where 用于表或者视图，不能相互替代）
  - 连接查询: 实现方法（嵌套循环、排序合并、索引连接）
  - 嵌套查询: 允许多层嵌套；子查询不能使用 order by 语句；不相关子查询，相关子查询分别是如何实现的？any/all, (not) exists 等关键词作用分别是什么？
  - 派生表：select [属性] from [子查询] where [条件]
  - 集合查询：union, intersect, except
- 数据更新
  - 添加行：insert into [表名 (属性名)] values [常量]
  - 删除行：delete from [表名] where [条件]
  - 修改行：update [表名] set [列名 = 表达式] where [条件]
- 空值
  - 有 not null 约束条件的不能取空值；加了 unique 限制的属性不能取空值；码属性不能取空值
  - 空值与另一个值 (包括另一个空值) 的算术运算的结果为空值，比较运算的结果为 unknown
- 视图
  - 创建语句：create view [视图名] as [子查询]
  - 作用：简化用户的操作；使用户能以多种角度看待同一数据；对重构数据库提供了一定程度的逻辑独立性；对机密数据提供安全保护；适当利用可以更清晰的表达查询
  - 视图的查询——视图消解法：进行有效性检查，转换成等价的对基本表的查询，执行修正后的查询
  - DB2 中关于视图更新有何限制？

4 安全性与完整性

- 安全性：保护数据库，防止恶意破坏和非法存取
- 存取控制方法：自主存取控制，强制存取控制
- 自主存取控制：用户对不同的数据对象有不同的存取权限，不同的用户对同一对象也有不同的权限，用户还可将其拥有的存取权限转授给其他用户。

  - 授予权限 (grant[权限]on[对象] to[用户])
  - 收回权限 (revoke[权限]on[对象]from[用户])
  - 操作权限：查询权，插入权，删除权，修改权以及它们的一些组合
  - 操作对象：关系、元组、属性、视图
- 强制存取方法：每一个数据对象被标以一定的密级，每一个用户也被授予某一个级别的许可证；对于任意一个对象，只有具有合法许可证的用户才可以存取
- 审计：audit [操作] on [对象]。启用一个专用的审计日志 (audit Log)，将用户对数据库的所有操作记录在上面，审计员利用审计日志监控数据库中的各种行为，找出非法存取数据的人、时间和内容
- 完整性：对数据库中数据的正确性和一致性的维护
  - DBMS 在完整性保护的职责：提供定义完整性约束条件的机制；提供完整性检查的方法；违约处理
  - 完整性保护的措施：完整性约束（包括实体完整性，引用完整性和用户自定义完整性约束），触发器，并发控制和故障恢复（包括转储与日志）

5 数据库设计

- 基本步骤：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
- 数据字典记录的信息：（1）关系模式定义；（2）视图定义；（3）索引定义；（4）完整性约束定义；（5）各类用户对数据库的操作权限；（6）统计信息
- 概念结构设计
  - E-R 图画法: 实体型用矩形表示，矩形框内写明实体名；属性用椭圆形表示，并用无向边将其与相应的实体型连接起来；联系用菱形表示，菱形框内写明联系名，并用无向边分别与有关实体型连接起来
  - 如何把 E-R 图转换为关系模型？
    * 一个实体型转换为一个关系模式。关系的属性：实体的属性；关系的码：实体的码
    * 一个 1:1 联系可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并; 一个 1:n 联系可以转换为一个独立的关系模式，也可以与 n 端对应的关系模式合并; 一个 m:n 联系转换为一个关系模式; 三个或三个以上实体间的一个多元联系转换为一个关系模式

6 数据库实现技术：查询优化、恢复、并发

- 查询优化：代数优化，物理优化
- 事务
  - 用户定义的一个数据库操作序列，这些操作要么全做，要么全不做，是一个不可分割的工作单位
  - 事务的定义 (begin transaction...commit/rollback)
  - 事务的特性——ACID：原子性（atomicity）、一致性（consistency）、隔离性（isolation）、持续性（durability）。
    * 原子性：事务是数据库的逻辑工作单位，事务中包括的诸操作要么都做，要么都不做
    * 一致性：事务执行的结果必须是使数据库从一个一致性状态变到另一个一致性状态
    * 隔离性：一个事务的执行不能被其他事务干扰；一个事务内部的操作及使用的数据对其他并发事务是隔离的并发执行的各个事务之间不能互相干扰

    - 持续性：一个事务一旦提交，它对数据库中数据的改变就应该是永久性的，接下来的其他操作或故障不应该对其执行结果有任何影响
  - 事务是恢复的基本单位，也是并发控制的基本单位
- 数据恢复
  - 数据库管理系统必须具有把数据库从错误状态恢复到某一已知的正确状态 (亦称为一致状态或完整状态) 的功能，这就是数据库的恢复管理系统对故障的对策
    - 撤销（undo）故障发生时未完成的事务
    - 重做（redo）已完成的事务
  - 故障类型：事务内部的故障、系统故障、介质故障
  - 数据恢复技术：数据库转储、登记日志文件
    - 事务故障恢复和系统故障恢复必须使用日志文件
  - 日志文件 (log file) 是用来记录事务对数据库的更新操作的文件
    - 以记录为单位的日志文件：各个事务的开始标记 (begin transaction)、各个事务的结束标记 (commit 或 rollback)、各个事务的所有更新操作。每条日志记录的内容：事务标识（标明是哪个事务），操作类型（插入、删除或修改）操作对象（记录 ID、Block NO.），更新前数据的旧值（对插入操作而言，此项为空值），更新后数据的新值（对删除操作而言, 此项为空值）
    - 以数据块为单位的日志文件，记录内容：事务标识，被更新的数据块
  - 数据转储：静态/动态转储，海量/增量转储
- 并发操作
  - 数据库为什么要有并发控制机制？
    - 当用户存取数据时，可能是串行执行（每个时刻只有一个用户程序运行），也可能是多个用户并行存取数据
    - 要实现数据资源的共享，串行执行意味着一个用户在运行程序时，其他用户程序必须等到这个程序结束，才能对数据库进行存取，这样数据库系统的利用率会极低。因此，数据库并发执行成为主流
    - 数据库的并发控制机制能解决这类问题，以保持数据库中数据的在多用户并发操作时的一致性、正确性
  - 并发操作带来的数据不一致性: 丢失修改、不可重复读、读“脏”数据

7 C♯ 基础语法

- 数据类型：简单类型（如 bool、int、double 等）、结构类型、枚举类型等值类型，类（如 object 和 string）、接口、数组、委托等引用类型
  - 值类型与引用类型区别：前者直接包含它们的数据；后者存储对数据的引用
- 变量定义与类型转换
- 控制语句：顺序，循环（for、foreach、while、do...while...），选择（if、switch）。注意与 C/C++ 不同，C♯ 中算术表达式不能用作条件表达式。
  - for 与 foreach 的异同
  - break 与 continue 的区别
- 类：访问修饰符（public，private，protected 的区别），对象，类的成员（字段、方法、属性），静态成员，构造函数与析构函数，this 关键词，方法与运算符重载，继承与多态，数组

  - 属性与字段相同点：都是类成员，都有类型，可以被赋值和读取；不同点：字段是数据成员，而属性是方法成员。

- 数组：一维数组，二维数组，交错数组

8 C♯ 窗体设计

- 控件共有属性：Name 属性、Text 属性、尺寸大小（Size）和位置（Location）属性、、字体属性（Font）、颜色属性（BackColor 和 ForeColor）、Cursor 属性、可见（Visible）和有效（Enabled）属性
- 控件的事件注册（订阅）：建立控件的事件（如点击 Click）与处理方法的关联
- 常用控件：窗体（Form），按钮（Button），复选框（CheckBox），单选按钮（RadioButton），分组框（GroupBox），标签（Label），文本框（TextBox），列表框（ListBox），下拉菜单（Combobox）等
  - 窗体启动时，事件的处理顺序：Load，Activated，其他 Form 级事件，窗体上控件的相应事件
- 多窗体调用，数据传递

9 ADO.NET 基础

- ADO.NET 访问数据库的一般流程（见图1）

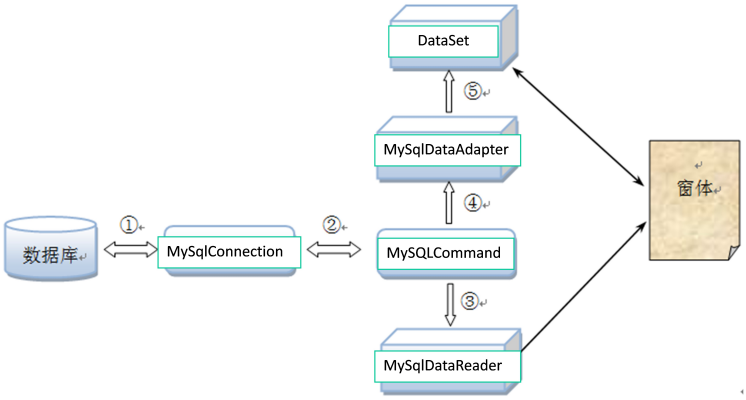

Figure 1: ADO.NET 访问数据库的一般流程

- 数据访问对象（MySqlConnection，MySqlCommand，MySqlDataReader，MySqlDataAdapter）分别怎么使用？连接数据库如下例

```
mystr = "server=localhost;user=root;database=smk;port=3306;password=123456";
myconn.ConnectionString = mystr;
myconn.Open();
mysql = "SELECT * FROM Product";
MySqlDataAdapter myda = new MySqlDataAdapter(mysql, myconn);
DataSet mydataset = new DataSet();
myda.Fill(mydataset, "Product");
myconn.Close();
```

  - MySqlDataReader 如何读数据，有何特点？如何分别通过 ListView 与 Combobox 将数据库数据显示在窗体上？

  - MySqlCommand 对象的 ExecuteReader(), ExecuteNonQuery(), Executescalar() 方法分别实现什么功能？

- DataSet 对象: 把数据从数据库放入 DataSet 对象中，需要使用 DataAdapter 对象的 Fill() 方法；反过来，要把修改过的数据保存到数据库，需要使用 DataAdapter 对象的 Update() 方法

10 数据控件

- 数据绑定
  - 如何为文本框添加数据绑定？
- DataView
- DataGridView
- 掌握实验三的 1，2，3，6，10，14，15