数据库系统原理课程

数据库系统考试

查漏补缺-选填

烂石

2025 年 3 月 19 日

1 数据库概念

1.1 模型概念体系

概念题选择题出的多, 如图 1所示

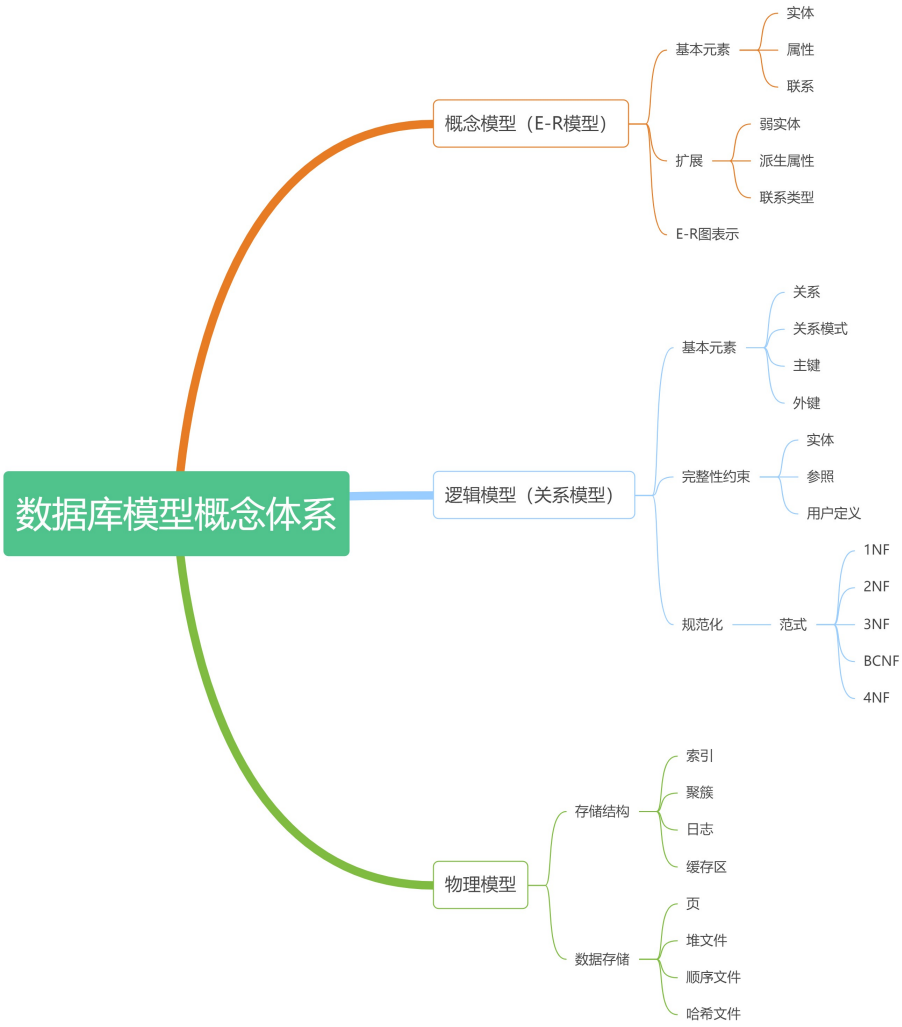

图 1: 数据库模型概念体系图

1.2 数据库设计阶段体系

了解框架, 如图 2所示可以看到, 需求分析阶段和 DFD 挂钩, 概念设计阶段和概念模型 E-R 图相关, 逻辑设计和逻辑模型相关规范化, 物理设计阶段和物理模型相关.

图 2: 数据库设计阶段图

1.3 数据库三级模式

主要注意哪些独立性和哪些模式相关, 把握好对应关系, 如图 3所示.

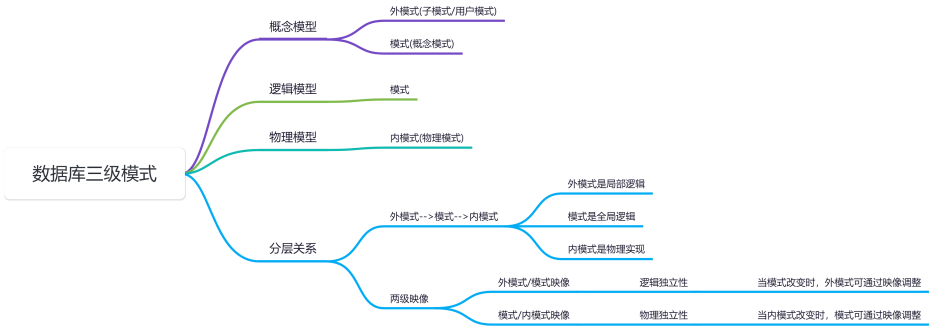

图 3: 数据库三级模式

1.4 小结

选择填空常考的基础知识点, 如图 4所示.

图 4: 数据库基本知识

2 关系数据库

2.1 易混淆的概念

1. 关系代数是以集合运算 为基础的运算.
2. 字段是列信息, 记录是行信息, 选择操作是对**行/记录**操作, 投影操作是对**列/字段**操作.
3. $R - S$ 表示在 R 但不在 S 的集合

2.2 常考填空

如图 5 所示, 常见的分类和概念.

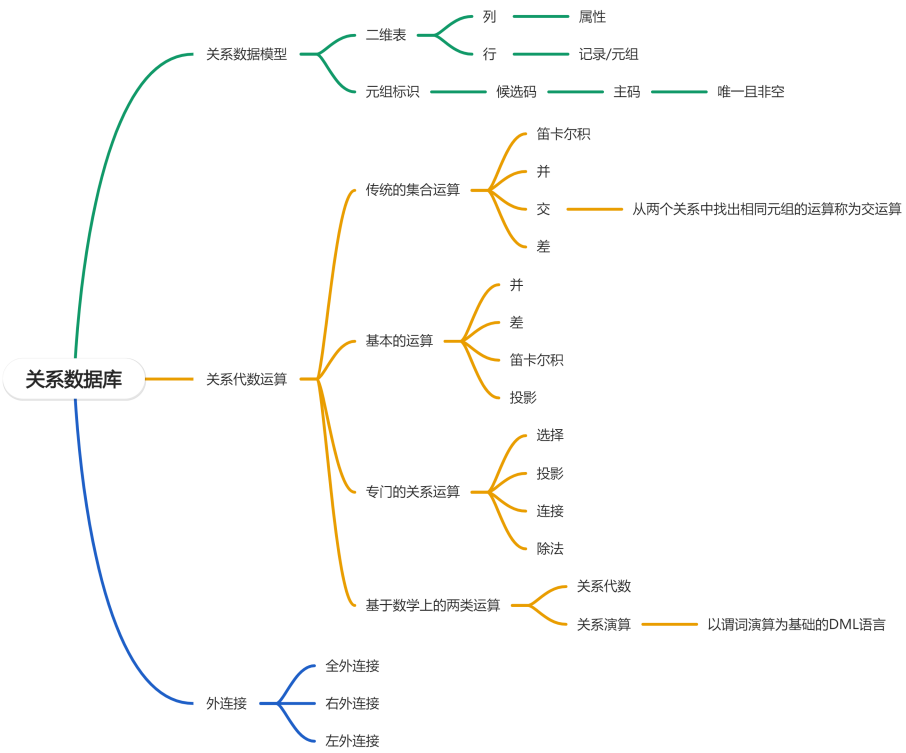

图 5: 关系数据库思维导图

2.3 关系代数表达式例题

eg. 在“学生一选课一课程”数据库中的 3 个关系如下:

表 1: 数据库中的 3 个关系

S (S#, SNAME, SEX, AGE)	SC (S#, C#, GRADE)	C (C#, CNAME, TEACHER)

查找选修“数据库技术”这门课程的学生的学生姓名和成绩，若用关系代数表达式来表示为:

$$\pi_{SNAME,GRADE}\left((S \bowtie SC) \bowtie (\sigma_{CNAME='\text{数据库技术}'}(C))\right)$$

3 关系数据库标准语言 SQL

3.1 错题知识点

1. 过程化注重流程, 如何做
2. 非过程化注重结果, 做出什么

删除行/记录是用:

```
DELETE row FROM TABLES
```

删除列/字段用:

```
ALTER TABLES DROP colomn
```

3.2 常考填空

1. SQL(Structured Query Language) 的中文全称是结构化查询语言

2. SQL 功能:

- 数据查询
- 数据定义
- 数据操纵
- 数据控制

3. **基本表**有对应的物理存储,**视图**没有对应的物理存储.
视图是从基本表或视图 中导出的表, 数据库中实际存放的是视图的定义.

4. DML(Data Manipulation Language) 特点:

- 操作对象与结果均为关系
- 操作的非过程性强
- 语言一体化
- 以数学理论为基础

5. 插入语句

```
INSERT INTO TABLES(field1,field2,field3,,,) VALUES(x,y,z,,,)
```

6. 修改语句

```
UPDATE TABLES SET field=vlaue
WHERE conditions
```

7. 删除语句

```
DELETE FROM TABLES
WHERE conditions
```

8. 例题

1. 设关系 R (A, B, C) 和 S (A, D, E, F), 有 R.A=S.A。若将关系代数表达式：$\pi_{R.A,R.B,S.D,S.F}(R \bowtie S)$ 用 SQL 语言的查询语句表示，则为：

```
SELECT R.A,R.B,S.D,S.F
FROM R,S
WHERE R.A=S.A
```

2. 在“学生—选课—课程”数据库中的 3 个关系如下：

表 2: 数据库中的 3 个关系

S (S#, SNAME, SEX, AGE)	SC (S#, C#, GRADE)	C (C#, CNAME, TEACHER)

查找选修“数据库技术”这门课程的学生的学生名和成绩。若使用连接查询的 SQL 语句是：

```
SELECT SNAME,GRADE
FROM S
JOIN SC ON S.S\#=SC.S\#
JOIN C ON SC.C\#=C.C\#
WHERE CNAME='数据库技术'
```

3. 设有两个关系 R (A, B, C) 和 S (C, D, E)，用 SQL 查询语句表达下列关系代数表达式 $\pi_{A,E}(\sigma_{B=D}(R \bowtie S))$ 的语句是

答案:

```
SELECT R.A,S.E FROM R
JOIN S ON R.C=S.C
WHERE R.B=S.D
```

9. 游标（Cursor）是数据库系统中用于逐行处理查询结果集的一种机制。它允许应用程序对结果集中的数据执行精细控制，支持遍历、读取、更新或删除特定行，适用于需要逐行操作的场景。(类似于指针)

表 3: 对比：游标 vs. 集合操作

特性	游标	集合操作
处理方式	逐行	批量
性能	低效	高效
适用场景	复杂逐行逻辑	简单查询、聚合、连接
资源占用	高	低

一个 SQL 语句原则上可产生或处理一组记录，而主语句一次只能处理一个记录，为此必须协调两种处理方式，这是通过使用游标或 Cursor 机制 来解决的。

10. DBMS 中的语言系统分为主语言 和SQL 语言

11. 删除/修改/插入操作 可以引发触发器.

4 数据库安全性

4.1 选择

4.1.1 安全性控制

授权的数据对象的范围越小, 授权子系统就越灵活;
授权的数据对象的约束越细致, 授权子系统就越安全, 但灵活性降低;

4.2 填空

1. 安全性问题包含技术安全类, 管理安全类和政策法律类
2. 鉴别用户的常用方法:用户名和口令
3. 安全子系统由用户权限定义 和合法权检查机制 组成.
4. DBMS 支持 DAC(自主存取控制), 有些还支持 MAC(强制存取控制)

表 4: DAC 和 MAC 对比总结

对比项	**DAC**（自主访问控制）	**MAC**（强制访问控制）
主体与客体权限	用户自主决定	系统根据策略自动执行
灵活性	高（用户自主管理）	低（系统策略决定）
安全性	较低（可能出现权限传播问题）	较高（严格控制权限）
权限分配者	资源所有者	系统或管理员
常见应用场景	日常软件与常规数据库系统	军事、政府、机密系统

5. 用户权限由数据对象 和操作类型 组成
6. 授权–GRANT; 收回–REVOKE
7. 用户查看保护–视图机制
8. 审计分为用户级 和系统级

5 数据库完整性

5.1 选择

出现和**主键**有关的完整性考点就选**实体完整性**

出现和**外键**有关的完整性考点就选**参照完整性**

出现和**自定义规则**有关的完整性考点就选**用户定义的完整性**

5.2 填空

1. 数据库的完整性指的是数据的正确性 和相容性.

2. 关系模型的完整性包括实体完整性, 参照完整性和用户定义完整性.

3. 数据库完整性的定义一般由 SQL 的DDL(Data Definition Language) 语句来实现, 它们作为数据库模式的一部分存入数据字典 中.

4. 实体完整性在CREATE TABLE 中用PRIMARY KEY 定义.

5. 参照完整性在CREATE TABLE 中用FOREIGN KEY 定义外码, 用REFERENCE 指明这些外码参照哪些表的主码.

延伸扩展, 如图 6所示,SQL 语言分类

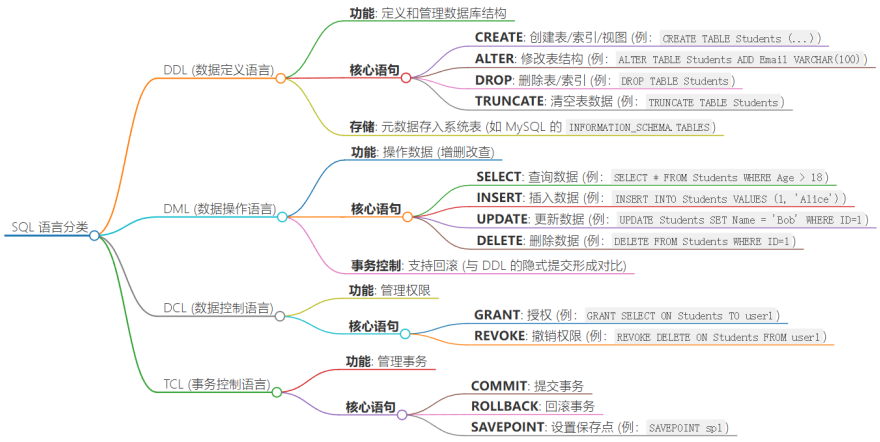

图 6: SQL 语言分类图

6 关系数据理论

6.1 选择

规范化, 如表 5所示

表 5: 数据库范式核心条件

范式	定义与核心条件
1NF (原子范式)	所有属性值为不可再分的原子值，但存在非主属性对候选键的部分函数依赖。
2NF (候选键范式)	满足 1NF，且非主属性完全函数依赖于候选键，但存在非主属性对候选键的传递函数依赖。
3NF (第三范式)	满足 2NF，且不存在传递函数依赖，但允许非主属性之间存在依赖关系。
BCNF (巴斯-科德范式)	满足 3NF，且所有函数依赖的决定因素均为超键（候选键或主键）。
4NF (第四范式)	满足 BCNF，且不存在非平凡的多值依赖（即属性间无非平凡的多值关联）。
5NF (第五范式/投影-连接范式)	满足 4NF，且不存在非平凡的连接依赖，确保关系无法通过投影-连接分解为更小的关系集合。

6.2 填空

1. 合并规则:$X \to Y$ 和 $Y \to Z$ 时，可以推导出 $X \to YZ$.

2. 平凡函数依赖可以通过自反律推出

- 平凡函数依赖：$Y \subseteq X \Rightarrow X \to Y$ 总成立，信息量小。
- 自反律：$X \to X$ 总成立，是平凡依赖的特例（当 $Y = X$ 时）。

3. 规范化设计中, 对模式等价分解时, 要具有无损连接性 和保持函数依赖.

4. 多种数据依赖, 其中最重要的是函数依赖 和多值依赖.

5. 设关系模式 R（A，B，C），F 是 R 上成立的 FD 集，F = {B→A，B→C}，则分解 ρ = {AB，AC} 丢失的 FD 是?。

易知, 候选键为 B, 其中 B->A,B->C, 但在 AB 中不包含 C,AC 中不包含 B, 即 B->C 不成立, 所以丢失 $B \to C$.

6. 关系范式不属于 2NF 时, 会产生插入异常, 删除异常和修改复杂, 只满足 1NF 的关系可能存在**数据冗余大, 修改异常, 插入异常和删除异常**.

7. 两个函数依赖集等价的充要条件为F 包含在 G 的超集,G 包含在 F 的超集中.

7 数据库设计

1. 数据库建设的基本规律:” 三分技术, 七分管理, 十二分基础数据”.

2. 规范设计法的基本思想是过程迭代 和逐步求精.

3. 数据库的生命周期可分为两个阶段, 一是数据库需求分析和设计阶段, 二是数据库实现和运行阶

4. 数据库设计阶段可分为:需求分析, 概念结构设计, 逻辑结构设计, 物理设计阶段, 数据库实施阶段

5. 数据库实施阶段包括组织数据入库 和应用程序的编码和测试.

6. 数据库的物理设计分为两步, 一是确定数据库的物理结构, 二是对其评价, 指标为空间 和时间效率.

具体设计阶段如图 7所示

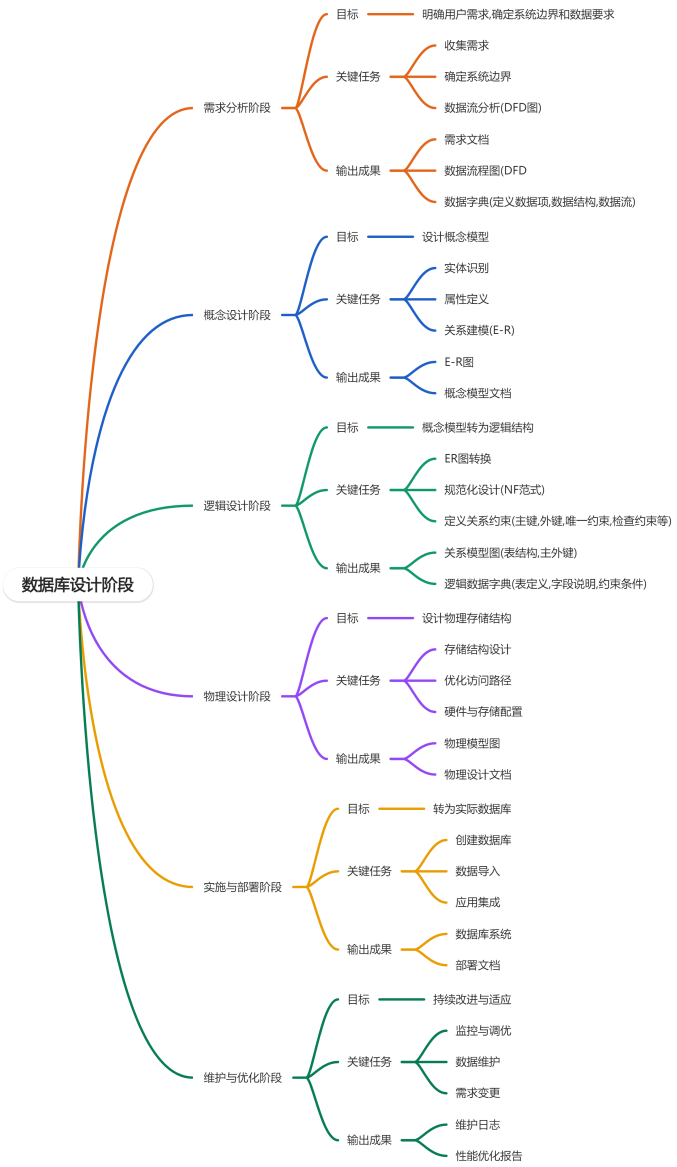

图 7: 数据库设计阶段图

7. 按应用目的分类, 模型可分为概念模型 和数据模型.

8. 类的继承提高了软件的可重用性/共享性, 继承的模型称为子模型.

9. 概念结构是对现实世界中的一种抽象, 抽象有分类, 聚集, 概括.

10. E-R 图设计冲突主要来自属性, 命名和结构.

8 数据库恢复技术

8.1 易错概念

1. 事务的持续性是永久的, 一旦提交, 对数据库的改变是永久的.

2. 事务日志用于保存关于” 数据” 和” 更新数据” 相关的操作 (必须是数据, 且数据要发生改变).

8.2 填空

3. 数据库应用程序的基本逻辑单元是事务.

4. 事务处理技术主要包括数据库恢复技术 和并发控制技术.

5. 事务的特性:原子性 (Atomicity), 一致性 (Consistency), 隔离性 (Isolation), 持续性 (Durability), 也称为 **ACID**

6. 数据库系统故障分为:事务故障, 系统故障, 介质故障, 计算机病毒.

7. 建立冗余数据最常用的技术是数据转储 和登录日志文件

8. 转储可分为静态转储 和动态转储, 转储方式可分为海量转储 和增量转储.

9. **日志文件**是用来记录事务对数据的更新操作的文件. 主要分为两种格式: 以记录为单位和以数据块 为单位的日志文件.

9 并发控制

1. 死锁是**资源分配策略不当**导致的, 两个或多个事务同时处于互相等待状态, 称为死装.

2. 并发操作带来的问题包括:读取" 脏" 数据, 丢失修改, 不可重复读

3. 多个事务的并发执行是正确的，当且仅当其结果与按某一次序串行地执行它们的结果相同，我们称这种调度策略为可串行化 的调度。

4. 封锁对象的大小被称为封锁的颗粒度.

5. 封锁类型分为排他锁 (Exclusive Locks)–X 锁 和共享锁 (Share Locks)–S 锁.(巧记: 首字母大小,E 不发音)