数据库

1.基本概念

数据（data）：描述事物的符号记录称为数据。

数据库（DataBase，DB）：是长期存储在计算机内、有组织的、可共享的大量数据的集合，具有永久存储、有组织、可共享三个基本特点。

数据库管理系统（DataBase Management System，DBMS）：是位于用户与操作系统之间的一层数据管理软件。

数据库系统（DataBase System，DBS）：是有数据库、数据库管理系统（及其应用开发工具）、应用程序和数据库管理员（DataBase Administrator DBA）组成的存储、管理、处理和维护数据的系统。

实体（entity）：客观存在并可相互区别的事物称为实体。

属性（attribute）：实体所具有的某一特性称为属性。

码（key）：唯一标识实体的属性集称为码。

Key: <https://blog.csdn.net/liushuijinger/article/details/12832017>

实体型（entity type）：用实体名及其属性名集合来抽象和刻画同类实体，称为实体型。

实体集（entity set）：同一实体型的集合称为实体集。

实体型：用实体名及属性名集合来抽象和刻划的同类实体。比如，学生(学号，姓名) 3。实体集：性质相同的同类实体的集合，称为实体集。比如，(张三，李四)

联系（relationship）：实体之间的联系通常是指不同实体集之间的联系。

<https://blog.csdn.net/u012234452/article/details/78324727>

模式（schema）：模式也称逻辑模式，是数据库全体数据的逻辑结构和特征的描述，是所有用户的公共数据视图。一个数据库只有一个模式 三级模式

在一个数据库系统中，『模式』与『内模式』都只能有一个，但『外模式』可以有很多个

外模式（external schema）：外模式也称子模式（subschema）或用户模式，它是数据库用户（包括应用程序员和最终用户）能够看见和使用的局部数据的逻辑结构和特征的描述，是数据库用户的数据视图，是与某一应用有关的数据的逻辑表示。

内模式（internal schema）：内模式也称为存储模式（storage schema），一个数据库只有一个内模式。他是数据物理结构和存储方式的描述，是数据库在数据库内部的组织方式。

<https://www.zhihu.com/question/38737183>

2.常用数据模型

层次模型（hierarchical model）

层次数据模型是用树状<层次>结构来组织数据的数据模型

网状模型（network model）

用有向图表示实体和实体之间的联系的数据结构模型称为网状数据模型

关系模型（relational model）

用表格表示实体和实体之间关系的数据模型称之为关系数据模型

<https://www.cnblogs.com/daniumeng/p/8717438.html>

关系（relation）：一个关系对应通常说的一张表

元组（tuple）：表中的一行即为一个元组

属性（attribute）：表中的一列即为一个属性

码（key）：表中可以唯一确定一个元组的某个属性组

域（domain）：一组具有相同数据类型的值的集合

分量：元组中的一个属性值

关系模式：对关系的描述，一般表示为 关系名(属性1, 属性2, ..., 属性n)

面向对象数据模型（object oriented data model）

对象关系数据模型（object relational data model）

半结构化数据模型（semistructure data model）

<https://www.cnblogs.com/x739400043/p/5815099.html>

<https://zhuanlan.zhihu.com/p/29138870>

<https://www.jianshu.com/p/2229ae6abd87>



成绩表中单一一个属性无法唯一标识一条记录，学号和课程编号的组合才可以唯一标识一条记录，所以学号和课程编号的属性组是一个主键。

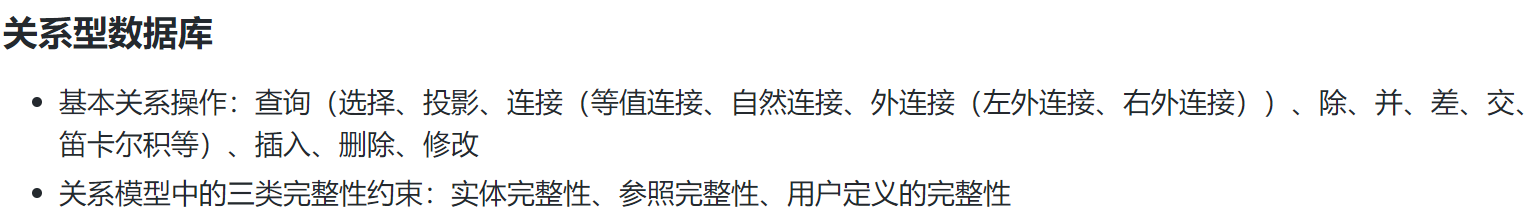
成绩表中的学号不是成绩表中的主键，但它和学生表中的学号相对应，并且学生表中的学号是学生表的主键，则称成绩表中的学号是学生表的外键；同理，成绩表中的课程号是课程表的外键。

定义主键和外键主要是为了维护关系数据库的完整性，总结一下：

1.主键是能确定一条记录的唯一标识。比如，一条记录包括身份证号码，姓名，年龄。身份证号码是唯一确认你这个人的，其他的都可能有重复，所以身份证号码是主键。

外键用于与另一张表相关联。是能确认另一张表记录的字段，用于保持数据的一致性。比如，A表中的一个字段，是B表的主键，那它就可以是A表的外键。





1. 数据完整性分为以下四类：

1) 实体完整性：规定表的每一行在表中是惟一的实体。

2) 域完整性：是指表中的列必须满足某种特定的数据类型约束，其中约束又包括取值范围、精度等规定。

3) 参照完整性：是指两个表的主关键字和外关键字的数据应一致，保证了表之间的数据的一致性，防止了数据丢失或无意义的数据在数据库中扩散。

4) 用户定义的完整性：不同的关系数据库系统根据其应用环境的不同，往往还需要一些特殊的约束条件。用户定义的完整性即是针对某个特定关系数据库的约束条件，它反映某一具体应用必须满足的语义要求。



1. 索引

数据库索引：顺序索引、B+ 树索引、hash 索引

<http://blog.codinglabs.org/articles/theory-of-mysql-index.html>

1. 数据库完整性

数据库的完整性是指数据的正确性和相容性。

完整性：为了防止数据库中存在不符合语义（不正确）的数据。

安全性：为了保护数据库防止恶意破坏和非法存取。

触发器：是用户定义在关系表中的一类由事件驱动的特殊过程。

6.关系数据理论

数据依赖是一个关系内部属性与属性之间的一种约束关系，是通过属性间值的相等与否体现出来的数据间相关联系。

最重要的数据依赖：函数依赖、多值依赖

7.范式

第一范式（1NF）：属性（字段）是最小单位不可再分。

第二范式（2NF）：满足 1NF，每个非主属性完全依赖于主键（消除 1NF 非主属性对码的部分函数依赖）。

第三范式（3NF）：满足 2NF，任何非主属性不依赖于其他非主属性（消除 2NF 主属性对码的传递函数依赖）。

鲍依斯-科得范式（BCNF）：满足 3NF，任何非主属性不能对主键子集依赖（消除 3NF 主属性对码的部分和传递函数依赖）。

第四范式（4NF）：满足 3NF，属性之间不能有非平凡且非函数依赖的多值依赖（消除 3NF 非平凡且非函数依赖的多值依赖）。

<https://segmentfault.com/a/1190000013695030>

<https://blog.csdn.net/Dove_Knowledge/article/details/71434960>

1. 数据库恢复

事务：是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。

事物的 ACID 特性：原子性、一致性、隔离性、持续性。

恢复的实现技术：建立冗余数据 -> 利用冗余数据实施数据库恢复。

建立冗余数据常用技术：数据转储（动态海量转储、动态增量转储、静态海量转储、静态增量转储）、登记日志文件。

<https://blog.csdn.net/logogcn/article/details/6828531>

9.并发控制

事务是并发控制的基本单位。

并发操作带来的数据不一致性包括：丢失修改、不可重复读、读 “脏” 数据。

并发控制主要技术：封锁、时间戳、乐观控制法、多版本并发控制等。



基本封锁类型：排他锁（X 锁 / 写锁）、共享锁（S 锁 / 读锁）。

<https://blog.csdn.net/sunsfan/article/details/52344860>

活锁死锁：

活锁：事务永远处于等待状态，可通过先来先服务的策略避免。

活锁的情况：如果事务T1封锁了数据R，事务T2又请求封锁R，于是T2等待；T3也请求封锁R，当T1释放了R上的封锁之后系统首先批准了T3的请求，T2仍然等待；然后T4又请求封锁R，当T3释放了R上的封锁之后系统又批准了T4的请求…T2有可能永远等待

死锁：事物永远不能结束

一个用户A 访问表A（锁住了表A），然后又访问表B；另一个用户B 访问表B（锁住了表B），然后企图访问表A；这时用户A由于用户B已经锁住表B，它必须等待用户B释放表B才能继续，同样用户B要等用户A释放表A才能继续，这就死锁就产生了

预防：一次封锁法、顺序封锁法；

诊断：超时法、等待图法；

<http://www.ishenping.com/ArtInfo/3560949.html>

解除：撤销处理死锁代价最小的事务，并释放此事务的所有的锁，使其他事务得以继续运行下去。

并发调度的可串行性

DBMS对并发事务不同的调度(schedule)可能会产生不同的结果

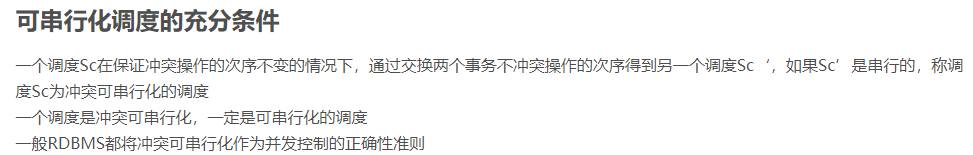
什么样的调度是正确的？

串行化(Serial)调度是正确的

对于串行调度，各个事务的操作没有交叉，也就没有相互干扰，当然也不会产生并发所引起的。如前所述，事务对数据库的作用是将数据库从一个一致的状态转变为另一个一致的状态。多个事务串行执行后，数据库仍旧保持一致的状态。

可串行化(Serializable)调度

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这些事务时的结果相同。可串行化调度当然也保持数据库的一致状态



<https://blog.csdn.net/xiangminjing/article/details/5922325>

<https://www.jianshu.com/p/478c6dca1b74>