1. 数据、数据库、数据库管理系统、数据库系统的概念及其区别

数据：描述事物的符号记录

数据库（DataBase）：长期存储在计算机内，有组织的，可共享的大量数据的集合

数据管理系统（DBMS）：用于科学地组织和存储数据，高效地获取和维护数据的系统软件

数据库系统（DBS）：在计算机系统中引入数据库后的系统，一般由数据库、数据库管理系统（及其开发工具）、应用系统、数据库管理员构成

2. 数据库管理系统的作为软件的地位（系统软件）

3. 数据库系统的四个特点

(1)数据结构化；(2)数据的共享性高，冗余度低，易扩充；(3)数据独立性高；(4)数据由 DBMS 统一管理和控制

4. 模型的概念

5. 数据模型的概念

在数据库中，数据模型被用来对现实世界中的数据和信息进行抽象、表示和处理。【数据模型是用户对现实世界的视图与计算机存储的二进制数据的桥梁，是数据库系统的核心和基础。】

6. 数据模型的三种要求

(1)真实模拟；(2)容易理解；(3)便于在计算机上实现

7. 数据模型的分类

(1)概念模型；(2)逻辑模型；(3)物理模型

8. 逻辑模型的种类

(1)层次模型，(2)网状模型，(3)关系模型，(4)面向对象模型，(5)对象关系模型

9. 两步抽象过程

现实世界->概念模型->DBMS 支持的数据模型

10. 数据模型的三要素

(1)数据结构：【描述数据库的组成对象以及对象之间的联系，是刻画数据模型性质最重要的方面（按数据结构命名数据模型），是对系统静态特性的表述。】

(2)数据操作：【指该数据结构上的各种实例允许操作的集合，包括相关的操作规则，是对系统动态特性的描述】

(3)数据的约束条件：【一组完整性约束的集合】

11. 数据结构是数据模型什么特性的描述？(见 10)

12. 数据操作是数据模型什么特性的描述？(见 10)

13. 概念模型的概念

实体（Entity）：客观存在并可相互区别的事物称为实体，可以是具体的人、事、物或抽象的概念。

属性：实体所具有的某一特性称为属性。一个实体可以由若干个属性来刻画。

码：唯一标识实体的属性集称为码

域：属性的取值范围称为该属性的域。

实体型：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：同型实体的集合

联系：不同实体间的联系，包括一对一，一对多，多对多

14. 概念模型中联系的分类（一对一，一对多，多对多）

15. 根据关联实体个数的联系分类（二元联系，多元联系，单元联系-自联系）

16. ER 图绘制方法

Entity-Relation 图

实体-联系模型(简称 E-R 模型)基于对现实世界这样一种认识：世界是由一组称作实体的基本对象及这些对象间的联系组成。因此，E-R 模型是一种语义模型，其语义主要体现在模型力图去表达数据的意义。

17. 关系模型中的基本概念

关系：一系列域上的笛卡儿积的子集。

元组：关系是笛卡尔积的有限子集，所以是一个二维表，表的每行对应一个元组，每列对应一个域。

属性：域可以相同，为了区分域，必须对每列起一个名字，称为属性（Attribute）。

码：

候选码（Candidate key）：若关系中的某一（最小）属性组的值能唯一地标识一个元组，则该（最小）属性组称为候选码。

主码（Primary key）：若关系中有多个候选码，则选定其中的一个为主码。

主属性（Prime attribute）：候选码的诸属性称为主属性。

非主属性（Nonprimary attribute）：不被任何候选码包含的属性。

全码（All-key）：关系模式的所有属性都是这个关系模式的候选码

域：具有相同数据类型的值的集合

分量：

关系模式：描述关系的名字，每个 field（或 column 或 attribute）和每个 field 的域（domain）. 在关系模式中，域由域名和与之相关的一组值相关。

18. 关系模型中对关系的限定和扩充（有限关系；附加属性名去除有序性）

19. 基本关系的六条性质（列同质；异列同域；列无序性；候选码唯一性；行无序性；分量原子性）

20. 关系模式的组成要素

对关系的描述称为关系模式。可以形式化的表示为：R（U，D，dom，F）

其中：R：关系名；U：组成关系的属性名集合；D：属性来自的域；dom：属性到域的映射；F：属性间数据的依赖关系

21. 关系与关系模式的区别

关系包含关系模式（relation schema）和关系实例（relation instance）。关系实例是个表，关系模式描述表的列头信息。

22. 关系操作的特点

集合操作方式。（操作的对象和结果都是集合，非关系数据模型的数据操作是记录方式）

23. 关系模型中完整性约束的分类

实体完整性、参照完整性、用户自定义完整性

24. 关系模型的两个不变性

实体完整性和参照完整性是关系模型必须满足的完整性约束条件，被称作是关系的两个不变性。

25. 实体完整性的含义

规则： 若属性 A 是基本关系 R 的主属性，则属性 A 不能为 NULL。（主属性不为空）

26. 参照完整性对于外码可能取值的规定（空或者对应主码的已有取值）

对外码取值的要求

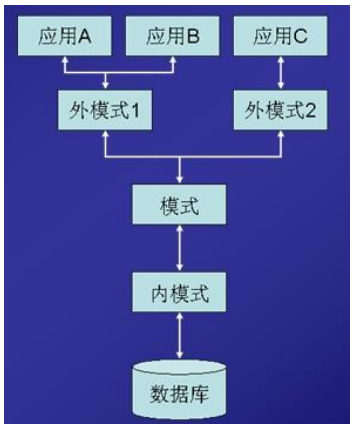
27. 数据库系统的组成四要素

一般由数据库、数据库管理系统（及其开发工具）、应用系统、数据库管理员构成

28. 模式的概念

模式：也称逻辑模式或概念模式，是数据库中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，综合了所有用户的需求
内模式：内模式也称存储模式，是数据物理结构和存储方式的描述，是数据在数据库内部的表示方式
外模式：也称子模式或用户模式，是数据库用户（包括应用程序员和最终用户）使用的局部数据的逻辑结构和特征的描述

29. 三级模式结构的构成



模式是数据库系统模式结构的中间层，与数据的物理存储细节和硬件环境无关，与具体的应用程序、开发工具及高级程序设计语言无关。外模式介于模式与应用之间。

30. 三种模式的数量对应关系

一个数据库只有一个模式、一个内模式和多个外模式。

31. 二级映像的名称

外模式／模式映像和模式／内模式映像

32. 二级映像的作用

(1)外模式／模式映像：保证数据的**逻辑独立性**；当模式改变时，数据库管理员修改有关的外模式／模式映象，使外模式保持不变；应用程序是依据数据的外模式编写的，从而应用程序不必修改，保证了数据与程序的逻辑独立性，简称数据的逻辑独立性。
(2) 模式／内模式映像：保证数据的**物理独立性**；当数据库的存储结构改变了（例如选用了另一种存储结构），数据库管理员修改模式／内模式映象，使模式保持不变；应用程序不受影响。保证了数据与程序的物理独立性，简称数据的物理独立性。

33. 两种数据独立性的作用

34. 等值连接和自然连接的区别

自然连接是一种特殊的等值连接，自然连接会去掉重复的属性

35. 视图

定义：视图是从一个或几个基本表（或视图）导出的表。
特点：虚表，是从一个或几个基本表（或视图）导出的表；只存放视图的定义，不会出现数据冗余；基表中的数据发生变化，从视图中查询出的数据也随之改变

- WITH CHECK OPTION

透过视图进行增删改操作时，不得破坏视图定义中的谓词条件（即子查询中的条件表达式）
作用：
1. 视图能够简化用户的操作。
2. 视图使用户能以多种角度看待同一数据视图机制能使不同用户以不同方式看待同一数

据，适应数据库共享的需要。
3.视图对重构数据库提供了一定程度的逻辑独立性。
4. 视图能够对机密数据提供安全保护对不同用户定义不同视图，使每个用户只能看到他有权看到的数据。
5.适当的利用视图可以更清晰的表达查询。

36. 索引

建立索引是加快查询速度的有效手段

37. 数据库完整性

实体完整性：创建主键，包含多属性的主键的创建；主键约束与非空约束和唯一性约束的关系。
参照完整性创建外键，外键所参照属性是否是主键；数据更新操作是否对参照完整性产生影响。
用户自定义完整性创建非空约束；创建唯一性约束；创建检查约束。

38. 函数依赖的名称（平凡、非平凡、完全、部分、传递、直接）看课件的例子

函数依赖：设 R(U)是属性集 U 上的关系模式，X,Y 是 U 的子集。若对于 R(U)的任意一个可能的关系 r, r 中不可能存在两个元组在 X 上的属性值相等，而在 Y 上的属性值不等，则称 X 函数确定 Y 或 Y 函数依赖 X。
平凡的函数依赖： X→Y,但 Y⊆X 则称 X→Y 是平凡的函数依赖。(Sno, Cno) → Cno
非平凡的函数依赖： X→Y,但 Y⊆X 则称 X→Y 是非平凡的函数依赖。(Sno, Cno) → Grade
完全依赖： 在 R(U)中，如果 X→Y,并且对于 X 的任何一个真子集 X',都有 X'→Y,则称 Y 对 X 完全依赖。Sno' → Grade，Cno' → Grade 因此：(Sno, Cno) → Grade
部分函数依赖： 若 *X*→*Y* 但 *Y* 不完全依赖于 *X*,则称 *Y* 对 *X* 部分函数依赖，记作：*X*→*Y*。
传递函数依赖： 在关系模式 R(U)中，如果 X→Y，Y→Z，且 Y⊆X，Y→X，则称 Z 传递函数依赖于 X。Sno → DName，DName → DLeader
直接依赖： 如果 Y→X， 即 X←→Y，则 Z 直接依赖于 X。

39. 使用函数依赖表达的候选码定义

设 K 为 R<U,F>中的属性或属性的组合，若 K → U，则称 K 为 R 的候选码。若候选码多于一个，则选定其中的一个为主码(Primary key)。

40. 1NF，2NF，3NF，BCNF，4NF，5NF 的满足包含关系 看课件例子

范式是符合某一种级别的关系模式的集合。

1NF：如果关系模式 R 的所有属性域都是原子（不可分的基本数据项）的，称 R 属于 1NF,记为：R ∈ 1NF。
2NF：若关系模式 R∈1NF，并且每一个非主属性都完全函数依赖于 R 的码，则 R∈2NF。
3NF：关系模式 *R*<*U*，*F*> 中若不存在这样的码 *X*、属性组 *Y* 及非主属性 *Z*（Z⊆Y），使得 *X*→*Y*，*Y*→*X*，*Y*→*Z* 成立，则称 *R*<*U*，*F*>∈3NF。若 R∈3NF，则 R 的每一个非主属性既不部分函数依赖于候选码也不传递函数依赖于候选码。
BCNF：设关系模式 R<U，F>∈1NF，如果对于 R 的每个函数依赖 X→Y，若 Y 不属于 X，则 X 必含有候选码，那么 R∈BCNF。当且仅当一个关系变量的所有非平凡的、左部不可约的函数依赖的决定因素是候选码，则该关系变量属于 BCNF

41. 明确数据库设计的基本步骤

(1)需求分析：准确了解与分析用户需求（包括数据与处理），是整个设计过程的基础，是最困难、最耗费时间的一步；

(2)概念结构设计：是整个数据库设计的关键，通过对用户需求进行综合、归纳与抽象，形成一个独立于具体 DBMS 的概念模型；
(3)逻辑结构设计：将概念结构转换为某个 DBMS 所支持的数据模型并对其进行优化
(4)物理结构设计：为逻辑数据模型选取一个最适合应用环境的物理结构（包括存储结构和存取方法）；
(5)数据库实施：运用 DBMS 提供的数据语言、工具及宿主语言，根据逻辑设计和物理设计的结果；
(6)数据库运行和维护：数据库应用系统经过试运行后即可投入正式运行。在数据库系统运行过程中必须不断地对其进行评价、调整与修改。

42. 概念模型到逻辑模型的转换规则

1. 一个实体型转换为一个关系模式
2. 一个 m:n 联系转换为一个关系模式。
3. 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。
4. 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。
5. 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. 同一实体集的实体间的联系，即自联系，也可按上述 1:1、1:n 和 m:n 三种情况分别处理。
7. 具有相同码的关系模式可合并。

43. 查询处理的步骤

44. 代数优化和物理优化的区别

45. 事务的基本概念

事务(Transaction)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。
事务是恢复和并发控制的基本单位。
明白 COMMITE 和 ROLLBACK 的含义

46. 事务应具有的四个特性（ACDI）

原子性（Atomicity）：事务中包括的诸操作要么都做，要么都不做
一致性（Consistency）：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（Isolation）：一个事务的执行不能被其他事务干扰
持续性（Durability）：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。

47. 故障的种类 具体的含义看课件

事务故障、系统故障、介质故障

48. 数据转储的两种分类方式（静态/动态；海量/增量）

静态：在系统中无运行事务时进行转储，转储开始时数据库处于一致性状态，转储期间不允许对数据库的任何存取、修改活动。
优点：实现简单
缺点：降低了数据库的可用性；转储必须等用户事务结束；新的事务必须等转储结束
动态：转储操作与用户事务并发进行，转储期间允许对数据库进行存取或修改
优点：不用等待正在运行的用户事务结束；不会影响新事务的运行
缺点：不能保证副本中的数据正确有效
海量转储：每次转储全部数据库
增量转储：只转储上次转储后更新过的数据
海量转储与增量转储比较
从恢复角度看，使用海量转储得到的后备副本进行恢复往往更方便。但如果数据库很大，事

务处理又十分频繁，则增量转储方式更实用更有效。

49. 对三种故障的恢复策略及步骤 见课件

事务故障的恢复：UNDO

系统故障的恢复：UNDO + REDO

介质故障的恢复：重装备份并恢复到一致性状态 + REDO

50. 建立检查点的步骤

1. 将当前日志缓冲区中的所有日志记录写入磁盘的日志文件上。
2. 在日志文件中写入一个检查点记录。
3. 将当前数据缓冲区的所有数据记录写入磁盘的数据库中。
4. 把检查点记录在日志文件中的地址写入一个重新开始文件。

检查点可分为两种：

定期：按照预定的一个时间间隔

不定期：按照某种规则，如日志文件已写满一半建立一个检查点

51. 利用检查点恢复的步骤

1.从重新开始文件中找到最后一个检查点记录在日志文件中的地址

2.由该地址在日志文件中找到最后一个检查点记录

3.由该检查点记录得到检查点建立时所有正在执行的事务清单 ACTIVE-LIST

4.从检查点开始正向扫描日志文件，直到日志文件结束

5.对 UNDO-LIST 中的每个事务执行 UNDO 操作，对 REDO-LIST 中的每个事务执行 REDO 操作

52. 利用检查点恢复时对不同事务的处理策略差异判断

53. 并发执行可能导致的数据不一致性问题 课件例子

（1）丢失修改（lost update）：丢失修改是指事务 1 与事务 2 从数据库中读入同一数据并修改，事务 2 的提交结果破坏了事务 1 提交的结果，导致事务 1 的修改被丢失。

（2）不可重复读（non-repeatable read）：不可重复读是指事务 1 读取数据后，事务 2 执行更新操作，使事务 1 无法再现前一次读取结果。

（3）读“脏”数据（dirty read）：事务 1 修改某一数据，并将其写回磁盘。事务 2 读取同一数据后，事务 1 由于某种原因被撤消，这时事务 1 已修改过的数据恢复原值，事务 2 读到的数据就与数据库中的数据不一致，是不正确的数据，又称为“脏”数据。

54. 锁的分类及兼容性矩阵

(1)排他锁（eXclusive lock，简记为 X 锁）排它锁又称为写锁，若事务 T 对数据对象 A 加上 X 锁，则只允许 T 读取和修改 A，其它任何事务都不能再对 A 加任何类型的锁，直到 T 释放 A 上的锁。

(2)共享锁（Share lock，简记为 S 锁）共享锁又称为读锁，若事务 T 对数据对象 A 加上 S 锁，则其它事务只能再对 A 加 S 锁，而不能加 X 锁，直到 T 释放 A 上的 S 锁。

	S	**X**
S	**true**	**false**
X	**false**	**false**

55. 活锁的避免方法 看课件的死锁及避免方法

活锁：一个事务总是不能在某数据项上加上锁，因此该事务也就永远不能取得进展。（锁的

分配不公平）
采用先来先服务的策略避免：当多个事务请求封锁同一数据对象时，按请求封锁的先后次序对这些事务排队。该数据对象上的锁一旦释放，首先批准申请队列中第一个事务获得锁。

56. 可串行化调度的概念以及正确的并发调度的概念 课件例子

几个事务的并行执行是正确的，当且仅当其结果与按某一次序串行地执行它们时的结果相同。这种并行调度策略称为可串行化（Serializable）的调度。

可串行性是并行事务正确性的唯一准则。

将所有事务串行起来的调度策略一定是正确的调度策略。

57. 判断是不是事务的所有串行执行方案都会得到相同的结果

课件例子

58. 两段锁协议与可串行化调度的关系

事务遵守两段锁协议是可串行化调度的充分条件，而不是必要条件。

可串行化的调度中，不一定所有事务都必须符合两段锁协议。

59．两段锁协议的含义

“两段”锁的含义：事务分为两个阶段

第一阶段是获得封锁，也称为扩展阶段：在对任何数据进行读、写操作之前，事务首先要获得对该数据的封锁；

第二阶段是释放封锁，也称为收缩阶段：在释放一个封锁之后，事务不再获得任何其他封锁。

60. 两段锁协议和三级封锁协议的区别 （三级协议看课件）

两类不同目的的协议：

两段锁协议：保证并发调度的正确性

三级封锁协议：在不同程度上保证数据一致性

遵守第三级封锁协议必然遵守两段锁协议。

61. 多粒度封锁中显式封锁和隐式封锁的含义

封锁对象的大小称为封锁的粒度(Granularity)。

多粒度封锁(multiple granularity locking)：在一个系统中同时支持多种封锁粒度供不同的事务选择。

显式封锁: 直接加到数据对象上的封锁

隐式封锁: 由于其上级结点加锁而使该数据对象加上了锁

显式封锁和隐式封锁的效果是一样的。

62．意向锁的含义

对任一结点加基本锁，必须先对它的上层结点加意向锁，如果对一个结点加意向锁，则说明该结点的下层结点正在被加锁。

意向共享锁(Intent Share Lock，简称 IS 锁)：如果对一个数据对象加 IS 锁，表示它的后裔结点拟（意向）加 S 锁。

意向排它锁(Intent Exclusive Lock，简称 IX 锁)：如果对一个数据对象加 IX 锁，表示它的后裔结点拟（意向）加 X 锁。

共享意向排它锁(Share Intent Exclusive Lock，简称 SIX 锁)：如果对一个数据对象加 SIX 锁，表示对它加 S 锁，再加 IX 锁，即 SIX = S + IX。

	S	X	IS	IX	SIX
S	Y	N	Y	N	N
X	N	N	N	N	N
IS	Y	N	Y	Y	Y
IX	N	N	Y	Y	N
SIX	N	N	Y	N	N