**第一章**

**什么是数据库系统？它有哪些重要组成部分？数据模型和数据语言有什么关系？**

数据库系统面向数据密集型应用，以统-管理和共享数据为特征，具有存储、管理、维护数据的功能。

数据库系统由应用程序、数据库管理系统、数据库、数据库管理员等组成。

数据模型是用来描述现实世界数据的一组概念和定义;数据语言(数据库语言)包括数据定义、操作、查询等基本组成部分(可以以关系模型和SQL为例进-一步展开)。

**设外关系R的物理块数为bR，内关系S的物理块数为bS，nB是可供连接的缓冲块数， 简述应怎样改进连接操作的嵌套循环算法，使其总的访问物理块数为：bR＋⌈bR /(nB-1)⌉ × bs**

DBMS以物理块为基本存取单元，ng-1 块缓冲区给外循环关系，1块缓冲区给内循环关系，目的是尽量减少对内循环关系的扫描读取次数:

具体算法略。只需将书上原算法对外循环关系的每个元组将内循环关系读取扫描一遍 ,改为对外循环关系一次读取nb-1个物理块到内存缓冲区，利用另一块缓冲区将内循环关系读取扫描一遍，在内存中仍利用两重循环进行两个关系元组间的两两比较。

**文件系统和数据库系统**

(1)数据库可以通过DBMS直接进行很多操作，而文件的话只提供了简单的操作。

(2)用文件的话，要针对不同的文件结构编写不同的代码。

(3)由于实际应用中并发同时访问导致数据访问不一致(文件没有解决办法，需要在应用软件层解决，而DBMS中有)

(4)故障处理(文件系统不具备这个能力，但DBMS会自动恢复)

(5)安全和访问控制

**数据库系统建立在文件系统之上。**

**举例说明什么是数据模型？什么是数据模式？两者的关系和区别？//P9**

**数据模型：** 是用来描述数据的一组概念和定义。以文件系统为例，它所用的数据模型包含文件，记录和字段。

**数据模式：** 以一定的数据模型对一个单位数据的类型、结构、相互的关系所进行的描述。例如学生信息记录可以定义为姓名、学号、性别等属性和关系的形式。

**区别：** 数据模型是描述数据的手段，而数据模式是用给定的数据模型对具体的数据的描述

**关系：** 数据模式反映一个单位的各种事物的结构、属性、联系和约束，实质上是用数据模型对一个单位的模拟。

**简述数据库系统中的多级数据模式对数据独立性的影响。**

外模式、概念模式、内模式，有效地组织、管理数据，提高了数据库的逻辑独立性和物理独立性。用户级对应外模式，概念级对应概念模式，物理级对应内模式，使不同级别的用户对数据库形成不同的视图。

**数据独立性**分为逻辑独立性和物理独立性.物理独立性是指内模式改变时,概念模式保持不变,逻辑独立是指概念模式改变时,外模式不变,从而使应用程序保持不变.当内模式改变时,DBMS 只要通过改变概念模式到内模式映射,即可使概念模式保持不变,从而实现了数据的物理独立性.而逻辑独立的实现正好相反.

**数据独立性**表示应用程序与数据库中存储的数据不存在依赖关系，包括逻辑数据独立性和物理数据独立性。 逻辑数据独立性是指局部逻辑数据结构(外视图即用户的逻辑文件)与全局逻辑数据结构(概念视图)之间的独立性。当数据库的全局逻辑数据结构(概念视图)发生变化 (数据定义的修改、数据之间联系的变更或增加新的数据类型等)时，它不影响某些局部的逻辑结构的性质，应用程序不必修改。物理数据独立性是指数据的存储结构与存取方法(内视图)改变时，对数据库的全局逻辑结构(概念视图)和应用程序不必作修改的一种特性，也就是说，数据库数据的存储结构与存取方法独立。

**数据独立性的好处是，**数据的物理存储设备更新了，物理表示及存取方法改变了，但数据的逻辑模式可以不改变。数据的逻辑模式改变了，但用户的模式可以不改变，因此应用程序也可以不变。这将使程序维护容易，另外，对同一数据库的逻辑模式，可以建立不同的用户模式，从而提高数据共享性，使数据库系统有较好的可扩充性，给 DBA 维护、改变数据库的物理存储提供了方便。

**现代数据库怎么管理数据模式的？数据模型怎么影响系统性能？什么是结构化数据，半结构化数据，非结构化数据？**

数据库系统划分为三个层次，称为三级模式，分别为概念模式，外模式，内模式，都存于数据目录中，是数据目录的最基本内容，DBMS 通过数据目录，管理和访问数据模式。

数据模型包含数据结构，数据操作，数据完整性约束，影响性能的因素主要是数据结构的复杂度和数据操作的可优化程度 。

结构化数据：数据整体结构化，通过数据模型描述。

半结构化数据：单条记录内部的数据有结构，数据文件间无联系，整体无结构。

非结构化数据：数据间，数据文件间都没有结构。

**什么是基本表？什么是视图？两者的区别和联系是什么？**

答：基本表是实际存储在数据库中的二维表，它是本身独立存在的表，在SQL中一个关系就对应一个表。视图是关系数据库系统提供给用户以多种角度观察数据库中数据的重要机制。

**区别：**视图是从一个或几个基本表（或视图）中导出的表，是一个虚表，数据库中只存放视图的定义，而不存放视图对应的数据，这些数据仍存放在原来的基本表中。

**试说明数据、数据库、数据库管理系统和数据库系统的概念以及它们之间的关系。**

**(1) 数据(Data)** ：描述事物的符号记录称为数据。数据的种类有数字、文字、图形、图像、声音、正文等。数据与其语义是不可分的。

**(2) 数据库(Database, DB)：**数据库是长期储存在计算机内的、有组织的、可共享的数据集合。数据库中的数据按一定的数据模型组织、描述和储存，具有较小的冗余度、较高的数据独立性和易扩展性， 并可为各种用户共享。

**(3) 数据库系统(Database System)：**数据库系统是指在计算机系统中引入数据库后的系统构成，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。

**(4) 数据库管理系统（ Database Management System，DBMS )：**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，用于科学地组织和存储数据、高效地获取和维护数据。DBMS的主要功能包括数据库的建立和维护功能、数据定义功能、数据组织存储和管理功能、数据操作功能、事务的管理和运行功能。

**它们之间的联系**：数据库系统包括数据库、数据库管理系统、应用系统、数据库管理员，所以数据库系统是个大的概念。数据库是长期存储在计算机内的有组织、可共享的大量的数据集合，数据库管理系统是由管理员操作管理数据库的查询、更新、删除等操作的，数据库应用系统是用来操作数据库的。

**数据库系统由哪几部分组成，每一部分在数据库系统中的作用大致是什么？**

答：数据库系统由三个主要部分组成，即数据库、数据库管理系统和应用程序。数据库是数据的汇集，它以一定的组织形式存于存储介质上；数据库管理系统是管理数据库的系统软件，它可以实现数据库系统的各种功能；应用程序指以数据库数据为核心的应用程序。

**分析其中两级映射关系**

外模式中的试图都是概念模式相关信息的可计算映射，物理模式与逻辑模式之间存在存储映射。

**第二章**

IMG_256

**在网状数据模型和关系数据模型中，如何表达两个记录型之间的 m:n 关系？一般而言，三个实体型之间的多对多联系和这三个实体型两两之间的三个多对多联系等价吗？分别举例说明。**

网状数据模型中，通过Link记录来实现M: N关系，而关系模型中通过关系(表)来表示M: N关系，与网状数据模型相比，实现的是一种“软连接”;

不等价;以供应商、零件、工程三者的关系为例，他们之间的一个三元多对多关系和三个两两之间的二元多对多关系是不同的，因为如果供应商A能够提供零件B、A又是工程C的供应商、而工程C又需要零件B,并不意味着工程C中必须要使用供应商A提供的零件B，也就是说三个两两之间的二元关系并不能决定他们之间的一个三元关系。

**简述 SQL 和关系代数的联系和区别**

关系代数（以关系代数为基础的数据库语言是过程性的）、SQL、关系演算(元组关系演算和域关系演算)，它们的非过程化程度依次递增，主要应用领域也不同.SQL 是关系数据库的标准语言，关系代数和关系演算是它的理论基础 。

**联系：**关系代数是 sql 的理论基础。

**区别：**sql 是结构化查询语言，是数据库具体的技术标准和规范。关系代数是数学理论。

**传统数据模型评价 //P30**

（1）都适用于OLTP【联机事物处理】应用

（2）以记录为基础，基于结构化数据，不能很好的面向用户和应用

（3）不能用很自然的方法表达现实世界的模型，表达能力有限

（4）模型本身缺乏语义信息

（5）支持的数据类型少，难以满足需求应用需求

**第三章**

**第四章**

**相较层次和网状数据库系统，查询优化对关系数据库系统更为重要。你认为这句话对吗？给出理由。**

对。层次和网状数据库系统的语言是过程性的，用户不仅要说明需要什么数据，还要告诉数据库系统获得这些数据的过程。层次和网状数据库的数据模型使用指针表示属性之间的关系，这样的结构也固定了这两种数据库的查询路径，进行优化的空间有限。

而关系数据库系统的语言是非过程性的，用户只用说明需要什么数据，不说明查询的过程，由DBMS确定合理有效的执行策略。对于同一个查询语句，对应的关系代数等价的不同表达式的查询效率有着很大的差异；集合操作不同的执行规则和策略也对查询效率有着很大的影响；同时关系数据在物理存储形式和存取方式和路径上都没有限制。因此对于关系数据库系统来说，查询优化就更为重要，它对系统的性能有着很大的影响。

**为什么查询优化对关系数据库管理系统(RDBMS)来说特别重要?与网状、层次模型的数据库管理系统相比，RDBMS的查询处理有什么本质的不同?**

要点:因为关系模型仍然用表来表示实体间的联系，是一种“软连接”，查询时涉及大量连接操作，因此必须优化以解决效率问题;

本质不同:关系数据库系统采用非过程化的查询语言;

**从查询优化角度分析，为什么 SQL 查询 where 子句应尽量避免使用 OR？**

为了加快查询速度，优化查询效率，主要原则就是应尽量避免全表扫描，尽量在 where 及 order by 涉及的列上建立索引。然而在 where 子句中使用 or 来当连接条件时，会导致引擎放弃使用索引而进行全表扫描，改用 Union 后，性能会大大提高。**//P124**

**简述索引对关系型数据库系统查询优化的意义；应该在什么时候使用索引？是不是正在任何情况下使用索引都能得到益处？举例说明。**

关系型数据库查询优化的途径之一是依赖于存取路径的优化，而在关系型数据库中索引是用得最多的一种存取路径，建立合适的索引是实现查询优化的首要前提。索引提供了对数据的快速访问，根据操作建立合适的索引能够很大程度上优化存取路径，从而提高查询效率。

**意义：**1.大大加快数据的检索速度;

2.创建唯一性索引，保证数据库表中每一行数据的唯一性;

3.加速表和表之间的连接;

4.在使用分组和排序子句进行数据检索时，可以显著减少询中分组和排序的时间。

只有当经常查询索引列中的数据时，才需要在表上创建索引。并非什么时候用索引都有益处，例如应用程序非常频繁地更新数据或磁盘空间有限时，则可能需要限制索引的数量。（索引占用磁盘空间，并且降低添加、删除和更新行的速度。）；例如对于小文件上的顺序查找使用堆文件的形式的开销不大，使用索引反而可能会增加开销。因为索引本身需要占用一定的存储空间，而且维护索引也需要一定的开销。对于增删改操作频繁的属性上建立索引可能会起到反作用。

**稠密索引是否一定能够提高针对索引属性查询的效率？**

不一定。

① 如果是查询小文件中的全部或相当多的记录时，使用索引并不能提高查询效率，反而会因为索引查询增加开销；

② 如果稠密索引为次索引，且不是簇集索引，则再最低索引中，每个键值对应的不是一个地址而是一个地址集。很可能一个键值对应的多条记录分散在不同的物理块中；当一个键值对应的记录较多时，取这些记录时访问物理块的 I/O 开销反而会降低查询的效率。

**B树和B+树**  
在 B+ 树中，键是存储在内部节点中的索引，而记录存储在叶节点。在 B 树中，key 不能重复存储，这意味着没有重复的键或记录。在 B+ 树中，叶子节点相互链接以提供顺序访问。 在 B 树中，叶子节点之间没有链接。 B+树是平衡二叉搜索树。 B+ 树确保所有叶节点保持在 一样的高度。在 B+ 树中，叶节点使用链表链接。因此，B+树可以支持随机访问和顺序访问。

**B-树为代表的树形索引成为当前数据库系统主流索引具有必然性**

索引文件是一种适应面比较广的文件结构，因此在数据库系统中得到了广泛的应用。对于经常变动的文件，静态索引的性能会随时间变化而变坏，所以目前在数据库系统中应用更多的是动态索引。而 B-数就是为了磁盘或其它存储设备而设计的一种平衡多分树，能很好地进行动态索引。B-树提供了三种存取路径：

1.通过索引集进行树形搜索；

2.通过顺序集进行顺序搜索；

3.先通过索引找到入口，再沿顺序集顺序搜索。

B-树不仅提供了灵活的存取路径，而且能够自动保持平衡，不须定期重组，因此 B-树为代表的树索引系列在数据库系统中应用甚广，成为数据库主流具有必然性。

**建立簇集索引的条件-P240；建立索引和不建索引的条件-P242**

**为什么都给了外循环关系，而内循环关系始终只给一个缓冲区？**

**如事务不遵守 ACID 准则，则对数据库产生何种后果？为什么一般不涉及数据库的程序中不提 ACID 准则？**

若事务不遵守 ACID 准则，数据库中会产生脏数据，数据不一致、难以恢复等情况。不涉及数据库的程序不提 ACID 是因为它们很少需要满足数据库这样的高并发性和一致性需求。一个事务是由应用程序中对数据库的一组操作序列组成的。如果事务不遵守 ACID 准则，则数据库中数据的完整性和一致性等就可能会因为事务的执行而遭到破坏。而一般不涉及数据库的程序不存在多用户之间数据的共享问题，所以在一般不涉及数据库的程序中不提 ACID 准则

**假设运行记录与数据库的存储磁盘具有独立失效模式，介质失效恢复时，对运行记录中上一检查点以前的已提交事务应该 redo 否？为什么？**

介质失效恢复时，对运行记录中上一检查点以前的已提交的事务应该 redo。介质失效是指磁盘发生故障，数据受损。因为介质失效会丢失数据库中所有的数据，恢复时需要再加载最近的后备副本后，根据运行记录中的后像，重做最近后备副本以后提交的所有更新事务。因此最近一次检查点以前提交的事务也要做 redo 操作。 但是对于后备副本以前的已提交的事务不应该redo。

在取后备副本后，以前运行的数据就失去价值，可以清空CTL表。

**判断并发事务运行正确性标准是什么？封锁法的基本思想是什么？它是怎么样保证并发事务的正确执行的？采用封锁法以后必须解决的问题是什么？ //P148**

（对于串行调度，各事务的操作没有交叉，也就没有互相干扰，当然不会产生并发引起的问题。可串行化调度和某一个串行化调度等价，所以它也不会产生并发所引起的问题。）

**正确性准则：**可串行化

**封锁法的基本思想：**并发事务对同一数据对象操作前，向系统发出请求对操作对象进行加锁。从而强迫有冲突的事务按照抢到锁的次序执行 。

**如何保证：**事务对某个数据对象的加锁请求获准后，该事务便对该对象有了一定的控制，在这个事务释放它的锁之前，其他事务对该数据对象的锁的请求需要根据相容矩阵进行锁的申请，如果没有申请到锁（证明有冲突），则无法对其进行操作，从而避免了访问冲突，保证并发事务正确执行。

**需要解决的问题：**活锁（先申请，先服务），死锁（等待-死亡，击伤-等待 **P158**），其中必须解决的问题是由于事物之间的循环等待导致的死锁问题。

**若事务并发执行的调度可串行化，即认为该并发结果是正确的，为什么？**

事务并发执行的调度是可串行化的，也就是说对于该事务并发执行的调度与该事务的串行调度等价；对于串行调度，各事务的操作没有交叉，没有互相干扰，因此不会产生并发执行时的冲突问题，因此与之等价的事务并发调度也不会产生冲突，即并发结果是正确的。

**描述更新事务的提交规则和日志提前规则的必要性//P142**

提交规则是为了确保在事务提交前将AI写入非易失性存储器中，这样即使事务进入commit阶段后发生故障，记录的AI仍然可以用于重做和更新，以保证事务符合ACID原则。

日志提前规则是指如果在事务提交之前直接将AI写入数据库，则必须在事务提交之前将相应的BI写入日志，以便在发生故障时进行undo在交易进入之前commit阶段，事务的执行符合ACID原则。

**已有的（S,X）、（S,U,X）锁能解决事务并发中的死锁问题么？为什么？**

不能解决并发事务中的死锁问题。当一个事物 A 占用数据对象 a 的 X 锁，事务 B 占用数据对象 b 的 X 锁，事务 A 和事务 B 又分别申请数据对象 b 和数据对象 a 的锁，在（S,X）和 （S,U,X）锁中，均无法获准，需要等待对方事务释放锁，而进入等待状态则无法释放自己所占用的锁，从而陷入循环等待，即死锁。

**（S,U,X）锁的相容矩阵为什么已经加了 U 锁，不允许其它事务申请加 U 锁？如果允许会出现什么情况？**

U 锁表示事务对数据对象进行更新的操作，在最后写入阶段事务再将其升级为 X 锁，导致最终写操作时若在 U 锁阶段允许其他事务申请 U 锁，则在事务 A 想将 U 锁升级为 X 锁进行数据写操作时，由于存在其他事物对数据对象的 U 锁，而无法升级为 X 锁，从而导致死锁。

**第五章**

**试分析空值产生的原因。为了处理空值，DBMS 要做哪些主要工作？**

原因:①某些数据不确定:②某些数据当前仍不知道:③某些数据不存在:

DBMS需要在查询处理中支持包括True、False、 Null 的三值逻辑;需要支持针对空值的实体完整性约束、引用完整性约束等。

**第六章-P197、223**

**数据依赖：**关系数据库内数据之间存在一定的数据依赖关系。

**函数依赖：**一个属性的值可以唯一的决定其它属性的值

**多值依赖：**一个属性的值决定其它属性的一组值，实际生活存在但很少。

**连接依赖：**关系属性之间能够无损连接（连接后的元组数一个不多，一个不少），实际生活存在但很少。

**函数依赖是行，多值依赖是列。**

**1NF:**关系中的每一个属性必须是原子的（不可再分），不支持表中套表。

**2NF:**1NF+不存在属性间的部分函数依赖。

·例子：若属性由（学号、姓名、班级、课程号、成绩）五个属性组成，其中（学号，课程号）共同组成主键。此时不满足二范式。其中的姓名和班级，只需要依赖主键中的学号就可以得到。

·不满足2NF问题：

①插入异常：不能插入一个还未选课的学生的信息

②删除异常 ：如果一个学生申请休学，把选过的课退了，那么他的信息也会被删除。

③更新异常：更新中难以保持数据的一致性，上例的设计有大量的数据冗余

·（学号、姓名、班级）（学号、课程号、成绩）

**3NF:**2NF+不存在属性对主键的传递依赖。

·例子：若属性由（职工编号、工资级别、工资）三个属性组成，其中（职工编号）为主键。工资取决于工资级别，工资级别取决于职工编号。

·不满足3NF问题：

①插入异常：当一个人的工资级别还没定的时候，他对应的工资也没有。

②删除异常：若只是删除一个员工的工资信息时，会把对应的工资级别信息也删除了。

③更新异常：有大量的数据冗余。

·（职工编号、工资级别）、（工资级别、工资）

**2NF、3NF解决方法：**一视一地：一张表只管一件事情。

**4NF:**3NF+消除多值依赖。

**5NF:**4NF+消除连接依赖。

**BCNF与3NF基本等价，而BCNF的决定子都是主键。**

**数据模式遵循的范式越高越好吗？**

数据库的范式主要目的是防止数据冗余、更新异常、插入异常和删除异常，因此，如果达到了该目的也就可以了，但范式越高可能带来处理速度缓慢和处理逻辑复杂的问题，因此需要权衡考虑。并不是应用的范式越高越好，要看实际情况而定。应用的范式等级越高，则表越多。表多会带来很多问题：1、查询时要连接多个表，增加了查询的复杂度 2、查询时需要连接多个表，降低了数据库查询性能 。

**第七章**

**试分析分布式数据库系统出现的技术背景和应用背景。它与后来出现的联邦式数据库系统的类似之处和本质区别是什么？**

**背景：**分布式处理技术的发展，当时网络宽还不足，因此出现了由网络连接的多台计算机共同协作解决大量数据的存储、管理、查询的需求，通过将数据就近存放提高访问效率。

**本质区别：**前者在物理上分布的，但逻辑上却是集中的。这种分布式数据库只适宜用途比较单一的、不大的单位或部门。而联邦式分布数据库系统在物理上和逻辑上都是分布的。由于组成联邦的各个子数据库系统是相对“自治”的，这种系统可以容纳多种不同用途的、差异较大的数据库，比较适宜于大范围内数据库的集成。

**相似之处：**分布式数据库系统的不同类别。是在集中式数据库系统的基础上发展来的。是数据库技术与网络技术结合的产物。包含分布式数据库管理系统(DDBMS)和分布式数据库(DDB)。

**分布式数据库的全局死锁。**

由于系统提供的资源数比多个进程所需的资源数少，并且系统的资源分配策略和进程并发执行的速度不当。死锁是占有资源并申请资源的事务之间循环等待造成的。（举一个例子，T1 握有 T2 需要的资源的同时等待 T2 的握有的资源），客户端A向服务器提交事务Ta，Ta需要R1,R2,R3,R4,R5五个资源，并已申请到R1和R2；而客户端B几乎同时提交事务Tb，需要R5,R4,R3,R2,R1五个资源，并已申请到R5和R3.至此Ta和Tb的循环等待开始，造成全局死锁。

**为什么虽然主流的数据库产品基本上都支持分布式数据库功能，但实际应用中分布式数据库的成功案例并不多?**

应用并不理想的原因:①系统实现和使用都比较复杂，从数据库设计角度看，物理上分布、逻辑上集中的数据库设计对开发人员要求较高:②难于管理，单个DBA很难维护大量物理上分布的数据库，多个DBA又很难协调，系统安全较难控制;③网络带宽和服务器性能的飞速提升，使当初的一些瓶颈不再成为障碍，基于集中式数据库服务器的三层(多层)信息系统架构成为流行。