基本概念:①**概念**(事或物及相互联系，包括**内涵**[属性]**外延**[实例]),**事物**(所有概念的统称)

②客观对象的抽象:→概念模型(易解 客观世界,eg.E-R模型)→数据模型[**定义**:一种抽象,表示和处理客观世界数据对象结构的描述方式][**要求**:形式化(书面),能够尽可能真实的反映客观世界,容易人所理解,便于在计算机上实现][**三大基本要素**:数据结构,数据操作,约束条件](计:**网状**IDS[图,物理链接法][Find(查找)Get(取数)Store(存数)Modify(修改)Erase(删除)Connect(加入)Reconnect(转接)Disconnect(撤离)][优点:能够更为直接地描述现实世界;具有良好的性能,存取效率较高;**缺点:**其DDL语言极其复杂;数据独立性较差;由于实体间的联系本质上通过存取路径指示的,因此应用程序在访问数据时要指定存取路径]、**层次**IMS[双亲关系pcr,树结构,不适用于多双亲和多多结构,多写几棵树或者虚拟记录解决][邻接法前序遍历存储][GU查找,GN下一记录,GNP下一兄弟][**优点**:数据模型比较简单,操作简单;对于实体间联系是固定的,且预先定义好的应用系统,性能较高;提供良好的完整性支持;**缺点:**不适合于表示非层次性的联系;对插入和删除操作的限制比较多;查询子女结点必须通过双亲结点]、**关系(见下)**、对象)③阶段:手工-程序-文件-数据库 ④数据:描述事物的符号记录,与语义不可分,描述事物的符号记录;数据库:长期储存在计算机内的,有组织的,可共享的大量数据集合（**基本特征**:数据按一定的数据模型组织,描述和储存;可为各种用户即应用程序共享使用;冗余度较小;数据独立性较高(逻辑独立性和物理独立性);易于扩展;库**相比于文件特征**:强制约束;可扩;查便;结构无关;安全(视图);并发;崩溃恢复）数据库管理系统:位于用户(应用程序)与操作系统之间的一层数据库管理软件,用于科学地组织和存储数据和高效地获取和维护数据（DBMS主要功能:数据定义,数据操纵,数据库运行管理建立维护）数据库系统在计算机系统中引入数据库后的系统构成(构成:DB,DBMS,应用系统,数据库管理员DBA)⑤**数据库系统开发运行环节**:需求分析[功能需求+**数据需求**(基本方法:数据流程图＋含数据字典;统一建模语言UML:用例图+时序图+类图等)]→系统设计(系统功能设计,**数据结构设计**(即后面三个设计))→系统实现(**数据库实现**和应用功能实现)→系统运维(运行系统与维护,**数据运行与维护,数据分析**);**数据库设计过程**:数据需求→概念结构设计(E-R图)→逻辑结构设计(eg.关系模式)→物理结构设计(eg.存放关系模式)→数据库实现,数据维护,分析; ⑥数据视图(数据抽象):三层模式两级映射(见后面的模式)⑨数据结构设计(分析特征+建立模型(概念、逻辑))

E-R:①实体集□属性联系集◇;联系集属性◇…□;②复合属性(子列表);多值属性({})一个属性对应一组值(多个电话号码);什么是派生属性(age()) ③一箭头 多就是一根线 全是双线 基数约束l最小...h最大; ④多元联系:每一根联系代表一个角色;一元联系存在自身与自身的联系，线上加属性 ⑤双菱形:弱实体集:无主码,依赖于标识实体集 键由主实体集键和自己的辨识符(eg.不同科室相同的病房号)多值属性 ⑥特化,概化(父实体集total,则全部概化,必属于一子集Ψ-”is a”联系(老师是一个雇员);o-重叠特化(允许同时成为不同的);d-不相交特化[需要标出符号含义]);聚集;联系集及相关实体集看成实体集

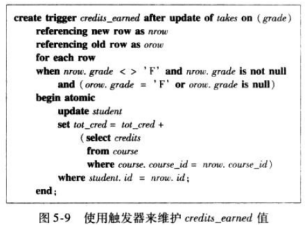
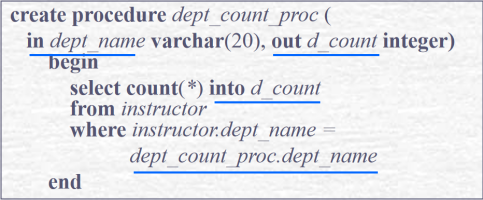
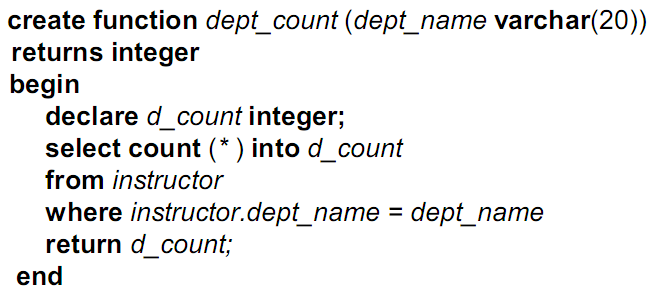
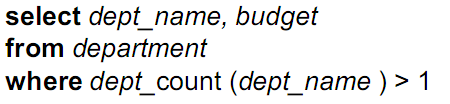
关系模型:1NF(原子性,横向或纵向展开),候选码是最小的超码,主码是唯一指定的候选码;关系模式(属性的一行),关系(元组);实体完整性(主键不能空值重复值)[确保唯一],域完(属性范围)[确保有效],参照完(不引用不存在,外键要和主键的一致)(所以要先删外键再删主键,先插入主键在插入外键)[确保一致];外键的存在:说明元组联系,保证数据有效性;模式图:实体,外键→主键;

关系代数:基本关代,选择σ(有多条则)投影Π并∪差-笛卡尔积×重命名(ρx(E)和ρx(A1,A2,...)(E)),例找max工资 Πsal (ins)-Πins.sal (σins.sal>d.sal(insΧρd(ins))；附加关代:交,自然连接,赋值运算as,外连接运算；(σinstructor.dept\_name = department.dept\_name (instructorΧdepartment)));扩展:广义投影(sal/12)和聚集函数(dept\_name gavg(salary) as avg\_sal (instructor)

)前面是需要聚集的属性名(加了就作为2属性关系,不加的就是一个数)avg min max count(不忽略空值) sum

E-R转关代:实体集开新的(弱实转时+主实主码,复杂实体集逐层展开),联系集只有多对多+新,概化看情况需不需要开父,聚集直接看成新实体集;**一元联系**一对一一对多加一个属性就行,例职工(工号,…,领导工号);多对多要新加,例对于零件(零件号, ...)与自己构成组成的一元联系,+组成(零件号, 子零件号, 数量);**三元联、系**对联系集单开关系模式(拆成三个二元,eg.1:1:1)

SQL:①DDL定义:**create table** ins(name varchar(20) not null **Check**,\n **primary** key (ID),\n **foreign** key(dept\_name) **references** department); **CREATE DATABASE** name\n ON (NAME, FILENAME,SIZE(初),MAXSIZE, FILEGROWTH(增)) ②DQL查询:**Select**属性(聚集的话avg(...))as更名**from**关系(,关系[笛积])**where**条件(between..and..;属性<=;not in (select...)/some&all ()) **group by**(属性) **having** avg(..)>.. ()**order by**属性**desc**降序/**asc**升序**; Union**并;**distinct**;差**Except**;交Intersact;例**select** course\_id **from** section as S **where** semester=’Fall' and year=2009 and **exist** (select \*from section as T where semester='Sring' and year=2010 and S.course\_id=T.course\_id);(空关系测试) A **natural join** B自然连A **join** B **using** 指定属性 都只出现一次 用=[和join on效果相同]则两遍; 外连接**nature left outer join**③DML操纵:delete from关系where;insert into关系values/(select..);update关系set..where/set ..=(case when then else)④视图**create view**视图名**as** select[定义:基于基本表所建立的虚表][作用:可以在任何QL语句中像表一样的被使用;增强查询能力且方便(用户/程序员)使用;还可以提供数据访问安全控制(隐藏数据);作为外模式(1级映射)有利于应用独立性]授权**grant**/revoke select/insert **on**关系(视图)**to**/from **restrict**(防止级联回收权限)授权图:U1授权则U1画下分支,默认级联回收；⑤函数,,**create function**名(输入名类型)**returns** 类型**begin** return ..**end**调用时直接f(1)>1,declare 定义一个新变量default 0(初值为零);过程create procedure(in..out..)begin..end调用时call f;触发器create trigger..after操作on ⑥断言:create assertion <assertion-name> check <predicate>;外键约束 on delete 连带删除



关系模式优化/函数依赖/范式:①函数依赖(如果对关系R中的每个实例r，都满足函数依赖，则该函数依赖在关系R上成立)→冗余→修改插入删除异常(改一动全添一添全删一丢联系);②2NF(非主属性必须依赖于候选码);③函数依赖FDs,X→Y(X-R则超键)[R1+=R2+函数依赖集等价];函数依赖理论Armstrong 公理:自反律(平凡AB-A)增广律(A-B,AC-BC)传递律(A-B B-C,A-C);合并X-Y X-Z,X-YZ分解X-YZ X-Y,X-Z伪传递X-Y WY-Z,WX-Z;④所有函数依赖:F+函数依赖集闭包;属性集依赖:A+(求超键候选键函数依赖F+);⑤模型分解标准:无损连接分解(分解后并起来还是R),保持函数依赖(原来的函数依赖分解后能推出来)⑥BCNF:(符合2NF和3NF)对每一个非平凡的X-A,X是超键;分解:若X-AnotBCNF分解R为R-A和XA(保持无损连接分解,不一定依赖,分解顺序不同结果不同)⑦3NF:(允许存在冗余但保持函数依赖)BCNF+X-A中A是候选键的一部分;F正则覆盖Fc(先合并左边,再去除左右的无关属性),分解:先计算正则覆盖,对于每一个依赖分解,再判断需不需要加候选键

物理设计:①一个数据库被映射到多个不同的文件,文件由操作系统来管理,被永久存储在磁盘上;一个文件在逻辑上被组织成记录的一个序列，记录被映射到磁盘块上;每个文件被分成定长的存储单元-块,块是数据存储和传输的基本单位(默认一般是4-8KB);一个块可以包括很多记录(假设一个记录总比块小；对大数据如图片需单独处理和存储)，且一个记录的数据不能跨块存储; ②**文件组织方式**:定长变长;**文件中记录的组织方式**,堆文件(任意),顺序wj(搜索码,插入删除开销大),散列wj(),多表聚集wj(多个表放一起,比如物理学科,再老师);③**物理设计的任务**:定长变长;确定搜索码;定辅助索引;定是否多表聚;定多少数据库;定数据库物理路径;④[自描述]**数据字典**/系统目录:关系属性视图约束存储方式等等;元数据:关于数据的数据⑤**数据库缓冲区**(buffer)(减少磁盘访问次数,降低I/O开销,提高访问效率),有LRU,MRU最近最常使用,立即丢弃等等;⑥聚集**索引**(主索引),按照搜索码某顺序排序\_\_可稀疏索引,只有某些码有搜索项,相反有非聚集和辅助(指针)\_\_只能密集索引;多级索引提高搜索效率;B+树;⑦**散列**索引:静态(散列函数);动态(根据索引数量动态分配桶数,索引散列函数后k位,没有的不放,满的分裂)

查询处理:①查询代价估算:传送磁盘块数+搜索磁盘次数;②连接操作的实现:嵌套循环方法;排序-合并法;索引连接(index join)方法;Hash Join方法;③表达式计算:**物化**计算(从底向上)-将中间计算结果作为临时关系存放支持上一层计算;**流水线**计算:上一操作的结果直接传到下一个操作;④**查询优化**[总目标:选择有效的策略,求得给定关系表达式的值,使得查询代价最小(实际上是较小)](代数优化):选择下移,投影下移,选择连接顺序(小的先连接);执行计划(在原本树的基础上添加了算法和协调) **优化步骤**:SQL-关代-查询树-一棵最优树-生成执行计划

事务与并发:

事务①**ACID**:Atomicity原子性(事务要么完成要么不完成)Consistency一致性(执行前后数据库一致)Durability持久性(完成之后长久生效)Isolation隔离性(彼此不干扰);②事务基本状态:活动的(初始)部分提交的(last语句执行)失败的(执行不能继续)中止的(回滚后)提交的;③**可串行化调度**(并行调度,串行的正确结果);若S与另一个串行调度冲突等价(通过非冲突指令交换),则S为冲突可串行化;**调度优先图**:j写前i写读或j读前i写,则Ti->Tj:无环,则冲突可串行化;④**可恢复调度**:j读i写,则i先交;**无级联调度**:j读i已提交写;⑤**事务隔离性级别**:可串行化-可重复读(只读已提交,而且两次读之间不得更新)-已提交读-未提交读;⑥按照事务的启动方式:显示事务(手动执行提交回滚),自动提交(..成功提交,失败回滚),隐性,,(自动启动新事务执行)

锁①为保证一致.隔离.并发引入锁,分为**共享锁S**(可读不可写,其他事务可以加S读)和**排他锁X**(可读可写不可加锁)②**死锁**ij互相等待无法执行;**活锁**一方长时间(或**永久**)等待另一方事务;等待图:i等j解锁,则i指向j(与调度优先图方向是反的)③**两阶段**[一组加锁规则]**封锁协议**(保证冲突可串行化,冲突可串行化未必两阶段):事务分只加锁和只解锁阶段;**封锁点**:增长阶段结束点;④级联回滚仍可能发生(未提交的读,撤销导致全部回滚):**严格两阶段封锁协议**:原基础上X锁在提交后释放;**强两阶段封锁协议**:提交前不得释放任何锁;⑤**锁转换**:封锁机制的实现:链表;⑥**图形协议/树形协议**:数据项的访问顺序,确保了无法产生死锁,但是可能读了提交的需要回滚,不能保证可恢复性(解决方式:X锁在提交后释放**≠严格**);⑦**死锁的检测**:等待图(i等待j释放)出现环;**死锁的解除**:选择牺牲者回滚,部分回滚,防止饿死(代价包括回滚次数);**死锁的预防**:全加锁;加访问顺序(树形);时间戳决定等待回滚(**wait-die**对于被占有的数据,老(时间戳小)事务等待,新事务回滚;**would-die**对于被占有的数据,新事务等待,老事务把占有者回滚);⑧**多粒度锁**(对集体加锁):意向锁IS(对后裔加S锁) IX(对后裔加X锁) SIX(对本人加S对后裔加X);[加锁自上而下,解锁自下而上]⑨相容性矩阵(加锁顺序):对于IS,IX,S,SIX,X,123467.11.13.16true

时间戳排序协议:①对每个事务给一个时间戳TS(Ti)[赋值方法:系统时钟或者逻辑计数器];同时对各项数据有写事务最大时间戳**W-timestamp**和读..**R-timestamp**②执行机制:TireadQ:TS(Ti)<W-ts(Q),则读完还写,无效读,回滚Ti;否则执行read,更新R-ts(更新为时间戳和这个TS的最大值);TiwriteQ:a.TS(Ti)＜R-ts(Q)或W-ts(Q),则读错值或无效写,回滚;否则执行write并更新;[回滚事务后将赋予新的时间戳]③优点:冲突可串行化;不会产生死锁;缺点:容易产生不可恢复调度;④快照隔离:事务开始执行时进行快照,事务在快照上操作;需要处理更新事务写的冲突:先提交者获胜(即将提交时);先更新者获胜(类似请求写锁)⑤基于有效性的协议:读阶段-有效性检查-写阶段;三个时间戳start validation finish;检查:对于老事务要么start前finish了要么写的数据项不相交而且检查前finish了;可串行化;预防级联回滚;

备份与恢复:①软硬件故障:事务故障(逻辑/系统错误);系统崩溃;磁盘故障;②易失性存储器(内存)/非易失性存储器(磁盘)/稳定存储器(多个非易失独立储存)③远程备份系统(目的:高可用性)[主站点-冷/热备份-备份/辅助站点]:故障检测(几条故障线路)-控制权移交(备份→主,执行redo日志)-恢复时间(周期性处理/热备份)-提交时间:一方保险(提交日志写入主则提交),两方强保险(主+备份才提交),两方保险(均活跃则等都写入,只主活跃则等主)

基于日志的恢复机制:a.日志<T0 **start/commit**><T0,数据A,修改前值,修改后值>b.[未提交/回滚]**undo**(T0)(恢复为事务执行前)增加日志<T0, B, 2000>, <T0, A, 1000>, <T0, abort> ;c.[已提交/回滚]redo(T0)数据更新为新的值;d.检查点:<**checkpoint** L>崩溃发生后只需要对L和最后检查点之后执行的事务redo或undo;e.redo正向扫描日志以重做;undo反向扫描日志以回滚;

DBMS体系结构:①集中式体系结构:通过公共总线共享内存(单用户系统/多用户系统)(粗粒度并行:多任务并行/细粒度并行:更多处理器,单任务并行);②客户服务器体系结构:前端(应用事务逻辑)/后端(数据库事务逻辑);[扩展]数据模型的三个层次:外模式(数据视图)-逻辑独立性-(概念)模式(数据中全体数据的逻辑结构和特征描述,公共数据视图)-物理独立性-内模式(数据物理结构和存储方式的描述);③服务器系统体系架构:事务服务器(查询服务器)发送请求[进程:服务器进程,数据库写..,锁管理器..,日志写..,检查点..,进程监控..];数据服务器 直接与用户交互;④并行数据库系统[并行能力的衡量要素:吞吐量(完成任务数)和响应时间(单个事务提交到完成的时间)];⑤分布式数据库(站点):数据共享,自治性和可用性【异构分布式数据库:连接多个已存在的数据库】⑥并行与分布式的区别:目标:并行优势vs局部自治,全局共享;高速网络连接vs局域网;各结点完全非独立vs场地自治性