**数据**是数据库中存储的基本对象，描述事物的符号记录**数据库系统DBS**互相关联的数据的集合和用以访问这些数据的程序组成 **数据库管理系统DBMS**是DBMS:系统软件，对数据库进行统一管理和控制。介于用户与操作系统之间数据管理软件系统。为数据库的建立、运行和维护提供统一管理和控制，提供定义数据和操纵数据接口，保证数据的安全性、完整性、并发使用及发生故障后的系统恢复。文件处理系统：数据冗余不一致/不同程序员创建不同文件不同格式不同程序设计语言不同地方重复储存、数据访问困难/方便而高效的方法获取数据原始系统没有相应程序、数据孤立/分散在不同文件不同格式、完整性问题/满足一致性约束新的约束很难、原子性问题/原子要么发生要么不发生、并发访问异常/多个用户同时更新数据、安全性问题/并非所有用户都可以看到全部数据。**数据库管理员(DBA)**对系统进行集中控制的人，包括数据库设计、规划、协调,最高特权的用户**作用**模式定义/存储结构及存储方法定义/模式及物理组织的修改/数据访问授权/日常维护) **管理信息系统（MIS）**办公系统、决策系统、生产系统和信息系统面向数量的执行系统、面向价值的核算系统、报告监控系统，分析信息系统、规划决策系统**数据管理发展的三个阶段特点：人工管理阶段**没有相应的软件系统，数据不共享，数据不具独立性。计算机主要用于科学计算外存为顺序存取设备没有操作系统，没有数据管理软件 **文件系统阶段**系统提供一定的数据管理功能如索引文件、链接文件、增删文件等弊端（数据冗余不一致、数据检索困难、数据不共享、完整性问题、原子性问题、并发访问一场、数据安全问题） **数据库系统阶段**DBMS统一管理，数据共享，数据的冗余度小，易扩充；数据面向整个系统，反映了客观事物间的本质联系，而不是着眼于面向某个应用，是有结构的数据；数据安全性，一致性，具有较高的数据和程序的独立性把数据库的定义和描述从应用程序中分离出去 数据描述是分级的（全局逻辑、局部逻辑、存储） 数据的存取由系统管理，用户不必考虑存取路径等细节，从而简化了应用程序  **DBMS演变**传统(网状层次关系)新型 **DBMS特点**用户接口，非过程数据库语言/查询处理与优化策略/并发控制，解决冲突/恢复功能，数据一致性状态/完整性约束检查，实体完整性、引用完整性、域完整性、用户定义完整性/访问控制，访问权限，安全性问题 **DBMS:数据定义语言(DDL)**是说明数据库模式和数据的其他特性的语言，提供了定义关系模式、删除关系和修改关系模式的命令。（**完整性约束（**保证授权用户对数据库的修改不会破坏数据的一致性）**1．域约束**属性值应是域中的值，属性的值能否为null，由语义决定。在确立关系模式时规定的，由DBMS负责检查 **2．参照完整性**一个关系中属性集上的取值也在另一关系的某一属性取值中出现**3.断言**数据库时刻满足的条件与约束和参照完整性约束是特殊形式数据库检查断言是否有效以后不破坏断言才被允许更新4、**授权**不同用户不同访问类型读权限插入权限更新权限）**数据操纵语言(DML)**是使得用户可以访问和操纵数据的语言，包括查询语言以及往数据库中插入元组、删除元组和修改数据库中元组的命令。**数据模型**数据库结构的基础是一个描述数据、数据联系、数据语义以及一致性约束的概念工具的集合 关系模型（表的集合表示数据及关系基于记录的模型）、实体-联系模型（现实世界的认识实体的基本对象及对象间的联系）、基于对象的数据模型（面向对象的程序设计）、半结构化数据模型。**数据库运行控制程序(DCL)**初始化安全性控制完整性检查并发控制事务管理运行日志管理等/实用程序， 数据转储恢复程序性能监测数据库重组数据转换通信 **DBMS功能部件**查询处理器（DML编译器DDL解释器查询求值引擎）存储管理器（权限及完整性管理器事物管理器文件管理器缓冲区管理器数据字典数据文件索引）**数据模型**是一个描述数据、数据联系、数据语义以及一致性约束的概念工具的集合**（**关系ER基于对象的半结构化数据模型**）概念数据模型**现实世界的数据抽象，描述一个单位的概念化结构,与DBMS无关，如ER面向对象**逻辑数据模型**DBMS层面上的数据形式。如层次网状关系**物理数据模型**数据存储结构，如物理块指针索引方式 **数据独立性**应用程序与DB数据结构之间相互独立**包括物理独立性**数据的物理结构的改变，修改模式/内模式映象，不影响数据库的逻辑结构，从而不致引起应用程序的变化**逻辑独立性**数据库逻辑结构的改变，修改外模式/模式映象，不需要修改相应的应用程序 **数据模型四要素**：数据的结构、数据的联系、数据的操作、数据的约束 **数据模式**数据模型下对数据的逻辑结构的描述 **模式**是指它的逻辑设计 **实例**是指特定时刻存储在数据库中信息的集合 **E—R模型定义**现实世界的事物抽象为实体，数据粒度相当于记录，描述实体用属性，数据粒度相当于字段，实体之间用联系表示。用E—R图表示，局部E—R图，全局E—R图。**面向对象模型定义**现实世界的事物抽象为对象，用类来定义，对象的特征用属性，对象之间用类与子类表示，对象的处理用封装其中的方法，调用方法用消息。**层次模型定义**实体用结点表示，实体之间的联系用树表示**网状模型定义**实体由记录表示，记录间联系由系（set）表示，系是一个指针关系模型**关系模型定义**实体由元组（记录）表示 实体集由关系（表）表示 实体之间的联系也通过关系表示。**DBMS演变**传统(网状层次关系)新型（并行数据库，知识库系统，主动数据库，多媒体数据库、CAD数据库、图形数据库、图像数据库、智能数据库，面向对象数据库，数据仓库，工程数据库，空间数据库。 **数据视图**互相关联的数据和访问修改这些数据的程序的集合数据库主要给用户提供数据的抽象视图**物理层**磁盘存储方式最底层数据描述DBMS设计人员看到**逻辑层**描述储存什么和数据间关系数据库全体数据的逻辑结构和特征的描述DBA数据库管理员看到（物理数据独立性：逻辑层用户不必知道逻辑层复杂的物理结构应用程序不依赖于物理模式）**视图层**数据库部分数据的逻辑结构和特征的描述,用户看到,是模式的一个局部 **数据库系统三级模式结构及其同数据独立性之间的联系**：为了提高数据的物理独立性和逻辑独立性，使数据库的用户观点，即用户看到的数据库，与数据库的物理方面，即实际存储数据库区分开来，数据库系统模式是分级的，美国数据系统语言协会提出模式、外模式、存储模式三级模式的概念。三级模式之间有两级映像：存储结构改变时，修改模式/内模式映像，使模式保持不变，从而应用程序可以保持不变，成为数据的物理独立性；当模式改变时，修改外模式/模式映像，使外模式保持不变，从而应用程序可以保持不变，称为数据的逻辑独立性。**关系**数据库由表的集合组成，关系指代表，关系实例指代特定的行，关系是元组集合而集合中的元素是没有顺序的关系中的元组也就没有先后顺序，域是属性取值集合原子的。**空值**表示不存在或未知，当某个属性为null或者“值未知”表示值存在但目前没有获得该信息，空值参与计算是值为空。关系模式属性及其属性对应的域name(x,m)，**模式**是指它的逻辑设计，**实例**是指它在特定时刻的内容。**关系的性质**每列同类型,同域,不同的列可以同域/列的顺序无关，交换列的次序,仍是同一个表/行的顺序无关，交换行的次序,仍是同一个表/任意两行不能全同 , 码的本质/任意两列的列名不能相同, 属性区分/每一分量是不可分的数据项, 关系数据库本质.关系模式由属性序列及个属性对应域组成。**外码**对于参照关系中每个元组来说，它在外码属性上的取值肯定等于被参照关系中的某个元组在主码上的取值超码一个或多个属性的集合唯一表示一个元组，最小超码是候选码。**完整性约束（作用：**保证授权用户对数据库的修改不会破坏数据的一致性）**域完整性约束**属性值应是域中的值，属性的值能否为null，由语义决定。确立关系模式时规定由DBMS负责检查。**实体完整性约束**每个关系应有一个主码，主码的值不能为null**引用完整性约束**不同关系之间或同一关系的不同元组间的约束。若关系R中有一个外码（相对于关系S），则R中每个元组外码的值必须满足：(1) 或者取空值 (2) 或者等于S中某个元组的主码值。R与S可以是同一个关系**用户定义的完整性约束**用户定义的完整性约束是针对某一具体数据库的约束条件，由具体应用要求决定。**关系查询语言**定义了一组运算集，这些运算可作用于表上并输出表作为结果；可以组合成表达式表达所需的查询。**关系代数（过程化：**用户指导系统对数据库执行一系列操作以计算出结果**）**一种抽象的查询语言，基于集合论，DML的传统表达方式定义了一套在表上运算且输出结果也是表的代数运算。关系代数定义了关系查询语言中使用的**基本运算（**选择投影并集合差笛卡尔积**）元组关系演算**是**非过程化语言（**用户只需描述信息而不用给出获取该信息的具体过程**）**，代表了关系查询语言所需的基本能力。 **关系查询语言**（基于谓词逻辑表示表的操作**）SQL：**商用语言，基于集合运算，非过程化与过程化 结合。**域定义**：create domain person-name char（20）**表定义：** primary key (.) ,foreign key(.),check (. > 0) **修改表**alter table 表名[add]增加新列[drop]删除列[modify ]修改列定义**用法：**distinct去除重复，all不去除，逻辑连词and,or,not；运算符<,>,<=,<>;natural join；as（select或from子句中）大写转换upper(s),小写转换lower（s）集合运算(自动去除重复)union,intersect(交),except集合成员资格in,not in(where语句中)至少比某一个大（>some）,比所有的都大（>all）关系A包含关系B not exists(B except A)**顺序显示**order by(desc降序，asc升序) order by salary desc,name asc**聚集函数**（出现在select语句中但是没有被聚集的属性只能是出现在group by中属性，出现在having字句中但是没有被聚集的属性必须在group by中）**索引：**在关系的属性上所创建的索引是一种数据结构，它允许数据库系统高效的找到关系中那些在索引属性上取给定值的元组而不用扫描关系中的所有元组。可以动态地定义索引，不允许用户在数据操作中引用索引。索引如何使用完全由系统决定，应该在使用频率高的、经常用于连接的列上建索引。一个表上可建多个索引。例create[unique/distinct][cluster]index 索引名on 表名(列名[asc/desc] [ , 列名asc/desc])…）eg：create clust er index s-index on S（SN）删除：drop index 索引名**转义字符**：eg:列出姓名中含有4个字符以上，且倒数第3个字符是d，倒数第2个字符是\_的教师的所有信息select \*from PROF where PNAME like’% ＿d \＿＿’escape’\’**空值注意事项**除is [not] null之外，空值不满足任何查找条件如果null参与算术运算，则该算术表达式的值为null如果null参与比较运算，则结果可视为unknown如果null参与聚集运算，则除count(\*)之外其他聚集函数都忽略null对于聚集函数，若输入集合为空，count返回0，其他返回null **视图定义**create view .. as ( select..)视图名指代该视图生成的虚关系，视图名可以出现在关系名任何地方**视图可更新有限制**from子句中只有一个数据库关系 select子句中只包含关系的属性名，不包含任何表达式，聚集或distinct声明 任何没有出现在select子句重的属性可以取空值，没有notnull约束也不是主码 查询中无groupby 或having子句 not null unique check **参照完整性约束：**保证在一个关系中给定属性集上的取值也在另一关系的特定属性集的取值中出现 **外码约束：**空值或另一个表值 **删除和修改基本关系元组**RESTRICT方式只有当依赖关系中没有一个外码值与要删除的基本关系的主码值相对应时才可以**级联删除**将依赖关系中所有外码值与基本关系中要删除的主码值所对应的元组一起删 **断言**CREATE ASSERTION <断言名> CHECK <条件>断言是谓词，表达数据库总应该满足的一个条件 只有不破坏断言的修改才允许 撤消断言：drop assertion 断言名 **游标**在查询结果的记录集合中移动的指针 用于把集合操作转换成单记录处理方式。若一个SQL语句返回单个元组，则不用游标 若一个SQL语句返回多个元组，则使用游标 **权限的转授和回收**允许用户把已获得的权限转授给其他用户，也可以把已授给其他用户的权限再回收上来 **授权命令** grant 权限 on 表/视图名 to 用户 **回收权限** revoke insert on S from Liming角色：数据库中建立一个角色集可以给角色授予权限 create role x grant select on takes to x **JDBC**标准定义了java程序连接数据库服务器的应用程序接口，结果提取到resultset（类似游标）对象变量rest中并每次取出一个next是否还有 **ODBC**标准定义了了一个API，应用程序用它来打开一个数据库连接，发送查询和更新开放的数据库连接**动态SQL**：通过函数或方法连接数据库程序运行时动态指定 **嵌入式SQL**：嵌入在高级语言的程序中固定写死在程序中不随程序变化。嵌入以EXEC SQL开始，以分号(;) 或END\_EXEC结束EXEC SQL delete from..；**连接数据库：**EXEC SQL connect to server user user-name using password**宿主变量**C变量，既可以用在C语句中，也可用在SQL语句中，用来在两者之间传递数据 **声明**声明为通常的C变量，并将其放在下列标识语句之间EXEC SQL BEGIN/END DECLARE SECTION 宿主变量出现于SQL语句中时，前面加（:）以区别列名 **SQL与主语言之间操作方式的协调1.**执行方式的差别SQL：一次一集合C语言：一次一记录 **游标**在查询结果的记录集合中移动的指针.EXEC SQL DECLARE C CURSOR FOR select open c fetch c into: **不需要游标的数据操作**结果是一个元组的select语句；insert；delete；update EXEC SQL select PNAME , SAL into : prof \_name, :salary from PROF where PNO = :prof\_no ;**需要游标的数据操作**当select语句的结果中包含多个元组时，使用游标可以逐个存取这些元组活动。**触发器**是一条语句，数据库做修改时自动被系统执行,实现某些完整性约束 **定义**指明什么条件下触发器被执行,指明触发器执行的动作是什么 **作用**示警.满足特定条件时自动执行某项任务事件动作Insert、delete、update. for each row显示的在每一个被插入的行上进行迭代。referencing new row as nrow用来在插入完成后存储所插入行的值eg:职工工资增幅不得超过10% create trigger RAISE\_LIMIT after update of SAL on EMP referencing new row as nrow old row as orow for each row when (nrow.SAL >1.1\*orow.SAL)begin atomic signal SQLSTATE (750 0’,’Salary increase 10%’) **联机分析处理系统（OLAP）**是一个交互式系统，允许分析人员看多维数据的不同种类的汇总数据数据仓库的主要任务OLTP面向客户、办事员、操作员的事务和查询处理OLAP面向经理、主管、分析人员的数据分析OLTP处理的数据是基本表中的元组数据OLAP处理的数据是不同粒度的汇总数据OLTP采用E-R模型和面向应用的数据库设计 OLAP采用星型或雪花模型和面向主题的多维数据立方体设计 OLTP主要功能是DBMSOLAP主要功能是检索查询工具，多维数据分析工具、统计分析及数据挖掘工具 **上卷操作**沿维的概念分层向上攀升，然后在数据立方体上进行的聚集。**下钻操作**沿维的概念分层向下延伸然后在大的数据立方体上进行小立方体上值的细化 **切片操作**在数据立方体的一个维上选择。**切块操作**在数据立方体的两个或多个维上选择 **转轴操作**转动数据的视角一种目视操作 **度量属性**：度量了某个值，而且可以在其上进行聚集操作 **维属性**：定义了度量属性以及度量属性的汇总可以在其上观察的各个维度 **多维数据：**能够模式化为维属性和度量属性的数据 **实体-联系模型ER**现实世界是由实体和实体之间的联系构成，用于数据库设计，方便的图形化表示方法以查看数据、联系和约束。实体客观存在并可相互区分叫**实体**（唯一标识）**实体标识符**能惟一标识实体的属性或属性集，有时也称为**关键码**，或简称为**键**。实体名和属性名组成**实体型**。**实体集的属性**是将实体集映射到域的函数。**联系**是多个实体间的关联。参与联系的实体组成的集合为**联系集**  **属性（**实体特征，通过其值区分不同的实体）**分类**简单属性：不可再分的属性；复合属性：可以划分为更小的属性；单值属性：每一个特定的实体在该属性上的取值唯一；多值属性（**双椭圆**）：属性上有多于一个的取值；派生属性(虚椭圆)：从其他相关的属性或实体派生出来的属性值 **实体集的属性**是将实体集映射到域的函数 **映射的基数**表示通过联系集可以和另一实体相关联的实体的个数 度：参与联系集的实体集的数目**联系的参与度**每个实体有一个参与联系的次数，取最小、最大的参与次数（部分参与，全参与）和（最大参与次数） **角色**实体在联系中的作用称为实体的角色（同一实体集不止一次参与一个联系集） 如果实体x的存在依赖于实体y的存在，则称x**存在依赖**于y。y称作支配实体，x称作从属实体 如果y被删除，则x也要被删除 设A R B，若A存在依赖于B，则A**全部参与**联系R ER**图中的表示：**弱实体集以双边框的矩形表示 ，标识性联系以双边框的菱形表示，从联系集用双线（全部参与）连接弱实体集，用箭头（一对多联系）指向强实体集，弱实体集的分辨符用下划虚线标明 **弱实体集**不具有足够属性构成主码的实体集。有主码的实体集称为强实体集。弱必然存在依赖于强，弱实体集与强实体集之间是一对多的联系弱实体集与其拥有者之间的联系称作**标识性联系。分辨码是**弱实体集中用于区别依赖于某个特定强实体集的属性集合，也称作**部分码。**分辨码和强实体集主码构成主码 **特化概化**定义了一个高层实体集合一个或多个底层实体集之间的包含关系。**特化**是取出高层实体集的一个子集来形成一个低层实体集。**概化**是用两个或多个不相交的（低层）实体集的并集形成的一个高层实体集。两者互逆，都用标记为ISA的三角形来表示。高层实体集的属性被低层实体集自动继承（层次结构/格结构）**继承**：高层实体集属性被底层实体集继承 **聚集**是一种抽象，其中联系集（和跟他们相关的实体集一起）被看作高层实体集并且可以参与联系。**使用弱实体集的原因：**避免数据冗余（强实体集码重复）以及因此带来的数据不一致性。反映了一个实体对其他实体依赖的逻辑结构，可以随他们强实体的集的删除而自动删除。 **四种异常：**插入删除冗余更新 **导致四中异常现象的原因**是构造关系模式过程中，忽视了属性之间存在着相互关联，相互依赖，相互制约的联系，盲目地把依赖不紧密的属性硬凑在一起。**范式**是关系模式的集合。某个域的元素被认为是不可再分的单元那么这个域就是原子的 函数依赖规定某些元组不能出现在关系中,也称为相等产生依赖;多值依赖要求某种形 式的其它元组必须在关系中,称为元组产生依赖。 X→Y 的有效性仅决定于 X、Y 属性集上的值;X→→Y 的有效性与属性集范围有关 **事务**访问并可能更新各种数据项的一个程序执行单元。BEGIN TRANSACTION 开始以COMMIT 语句或ROLLBACK语句结束**ACID特性**：原子性**（保证**事务所有影响在数据库中要么全反映要么根本不反映**）**一致性**（保证**事务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**（保证**并发执行的事务相互隔离.一个事务内部的操作及使用的数据对并发的其它事务是隔离的**）**持久性**（保证**是指一个事务一旦提交，它对数据库中数据的改变就应该是永久性的**）视图可串性化的扩展优先图**头事务Tb只有写操作Wb（Q）,尾事务Tf只有写操作Rf（Q）得s’找出所有写读对Wi(Q)Rj(Q), 加0标记有向边Ti-0→Tj 对调度中间的每个Wk(Q) 对应头写读对Wb(Q)Ri(Q), 0标记Ti→Tk 对应尾写读对Wj(Q) Rf(Q),加0标记有向边Tk-0→Tj 对应中间的写读对Wi(Q)Rj(Q), 加p标记有向边Tk-P→Ti和Tj -P→Tk如果无环，是视图可串性化。如果去掉P标记边对的一条，无环，也是视图可串性化。等价的串性调度是，按有向边出口依次截取结点。**“活锁”问题**：系统可能使某个事务永远处于等待状态，得不到封锁的机会。解决方法：采用“先来先服务”的策略，也就是简单的排队方式。如果运行时，事务有优先级，那么很可能使优先级低的事务，即使排队也很难轮的上封锁的机会。此时可以采取“升级”方法来解决，也就是当一个事务等待若干时间还轮不上封锁时，可以提高其优先级别，这样总轮的上封锁。**死锁问题**：系统中有两个或两个以上的事务都处于等待状态，并且每个事务都在等待其中另一个事务解除封锁造成任何一个事务都无法继续执行当死锁发生时，系统必须回滚两个事务中的一个。一旦某个事物回滚，该事务锁住的数据项就被解除了，其他事物就可以访问这些数据项，继续自己执行。在 DBS 运行时,死锁状态是不希望发生的,死锁本身是一件坏事。但是坏事可以转换为好事。如果不让死锁发生,事务任意并发做下去,那么有可能破坏DB中数据,或用户读了错误的数据。从这意义上讲,死锁的发生是件好事,能防止错误的发生。在发生死锁后,系统的死锁处理机制和恢复程序就能起作用,抽取某个事务撤消做 ROLLBACK 操作,继续运行下去。 **封锁协议**事务运用X锁和S锁对数据对象加锁时的约定**一级封锁协议**修改X，只读时不加.可防止丢失修改。 **二级封锁协议**修改X读S读完释放。可防止丢失修改，防止读“脏”数据。 **三级封锁协议**修改X读S事务完释放。可防止丢失修改，防止读“脏”数据，保证可重复读。 **两阶段封锁协议**对任何数据读写之前，先要获得对该数据的封锁，释放一个封锁之后，事务不再加任何锁。增长阶段加锁；缩减阶段放锁。一旦释放锁，进入缩减阶段。所有事务都遵守两阶段协议，并发调度是可串性化的两阶段封锁协议可能发生死锁 **时间戳**：每一个事务Ti，唯一的固定的时间戳和它联系起来记为TS(Ti)事务启动时间记为 TS（Ti）**W-TS（Q）**所有执行了写Q操作的事务中最年轻事务的时间戳（值最大）。**R-TS（Q）**所有执行了读Q操作的事务中最年轻事务的时间戳（值最大）。**时间戳协议**Ti 读数据Q时（1）若TS（Ti）< W-TS（Q），撤消Ti并重新启动（2）若TS（Ti）> W-TS（Q），执行读操作，R-TS（Q）= MAX（R-TS（Q）,TS（Ti ））Ti 写数据Q时（1）若TS（Ti）＜ R-TS（Q），撤消Ti并重新启动（thomas）若TS（Ti）＜ W-TS（Q），撤消Ti并重新启动。若TS（Ti）＜ W-TS（Q），撤消Ti并重新启动（3）其它情况，执行写操作，W-TS（Q）=TS（Ti） **数据仓库**：多个数据源中收集来的信息以同一模式存储在单个仓库。**决策树算法的基本策略：**决策树是一种用于产生分类规则的树结构树中的内节点表示在一个属性上的测试每个树叶代表类或类分布树中的每个分枝代表一个测试输出，即一条规则。样本的属性为结点，信息量最大的为根结点 属性的值为分支 **递规划分停止的条件**给定结点的所有样本属于同一类；没有剩余属性可以用来进一步划分样本，在此情况下，使用多数表决。决策树：计算每个属性的商，选择最大的当做根，然后根据此属性分组，假如有一组的类别单一则不用分这一组，继续按照全部属性接着分**决策树的优点**：可以生成可理解的规则。计算量相对来说不是很大，速度较快。可以处理连续和种类属性。决策树可以清晰的显示那些属性比较重要。**决策树的缺点：**对连续性的属性比较难预测。对有时间顺序的数据，需要预处理 **属性信息增益（互信息）计算**：I（U，V）=H（U）- H（U/V）=类别熵–条件熵。U为类别 V为属性取值 **类别熵：**H（U）= ∑P( ui ) log2(1/ P( ui ))=-∑P( ui ) log2( P( ui )) P( ui )=属于ui 类的样本数/全体样本数  **条件熵：**H（U/V）= ∑P( vj ) ∑ P( ui︱vj )log2 (1/ P( ui ︱vj )) P( vj )=属性取值为vj 的样本数/全体样本数 **粗糙集方法**上、下近似集来处理不确定性问题 基本思想：把对象（元组）的属性分为条件属性和决策属性(类)，按各属性相同划分成等价类条件属性和决策属性之间有三种关系：下近似：决策属性包含条件属性上近似：决策属性和条件属性之交非空无关 **贝叶斯定理:**设X是未知类的样本,H是X属于C的一个假定,X的分类问题变为求P(H︱X),即给定X,H成立的概率 C为类别，X为满足条件 给定条件X求P(Xi|Ci),然后乘，得P(X|Ci)，最后求P(Ci)，哪个大选Ci类别 **Apriori方法找出n项集**:每一个Li和数量列表，挑选>=最小阈值的项，两两结合在初始表中寻找数目并>=最小阈值，并且每一项的子集必须在前一表中出现，依次递归。**Apriori的性质：**频繁项集的所有非空子集都必须也是频繁的 **求关联规则：**求全部子集，A=>B支持度为B/A,根据最小支持度选择大于的 **IF-TDF**：权= tfj(Di)\* Log(N/nj)词条频率和文档频率倒数对数，在某一页出现的多不是全部文档，减少对N **文本挖掘**从大量文本中提取未知的知识 将一个嵌套关系转换为1NF的过程称为 **解除嵌套** select title, A as author, date.day, date.month, date.year, C as keyword from doc as B, B.author-list as A, B.keyword-list as C 将一个1NF关系转换为嵌套关系。通过SQL分组的扩展来实现select title, set(author) as author-list, (day, month, year) as date, set(keyword) as keyword-list from flat-doc groupby title, date 复杂类型：数组同类型有序，集合同类无序一次出现，结构不同类型有序，多集同类无序多次出现