第二节

关系数据库

关系模型和关系模式、关系代数

关系数据库系统是支持关系模型的数据库系统。关系模式是一个关系的属性名表。关系模型是所有的关系模式的汇集，是模式描述的对象，它由关系数据结构、关系操作集合和完整性约束三部分组成。

关系模型

1、关系数据结构

关系模型的数据结构简单，无论是实体还是实体之间的关系均由单一的结构类型即关系（表）来表示的。

- **基本表：** 实际存在的表，是实际存储数据的逻辑表示；
- **查询表：** 查询结果所对应的表；
- **视图表：** 是由基本表或其它图表导出的表，是虚表，不对应实际存储的数据。

关系模式

关系模式是对关系的描述。关系实际上是关系模式在某一时刻的状态或内容。也就是说，关系模式是型，关系是它的值。关系模式是静态的、稳定的。关系是动态的、随时间变化的。

关系代数

关系代数是对关系运算的总和，关系运算分为传统的集合运算、专门的关系运算。

1、传统的关系运算

并运算、交运算、差运算、笛卡尔积运算；

2、专用的关系运算

选择（SELECT）、投影（Project）、自然连接（Join）、除法运算（Division）；

2、关系操作

关系操作是采用集合操作的方式，也称为一次一个集合的方式。非关系操作方式则一次一条记录的方式

查询类操作：选择、投影、连接、除、并、交、差；

改动类操作：增、删、改；

3、完整性约束

在关系型数据库中，完整性约束用于确保数据的准确性和一致性

实体完整性：基本关系的所有主属性都不能取空值；

用户定义的完整性：它反映某一具体的应用所涉及的数据必须满足的语义要求；

关系数据库的标准数据语言SQL

SQL是Structured Query Language(结构化查询语言）的缩写，它包括了数据定义、数据查询、数据操纵和数据控制功能四部分，其中数据查询是SQL语言的最重要和最常用的部分。SQL是已经成为关系数据库的的标准数据语言，目前所有的关系型数据库管理系统都支持它。

SQL标准的发展变化

1986年10月美国ANSI公布了最早的SQL标准

1989年4月ISO在此基础上提出了具有完整特征的SQL89

1992年11月ISO在SQL89的基础上进行扩充提出了SQL92标准

SQL语言的特点

SQL是一种一体化的语言，它包括了从数据定义、数据查询、数据操作和数据控制功能，能完成数据库活动中的全部工作

SQL语言是一种高度非过程化的语言，用户只需提出“做什么”，不必指明“怎样做”

SQL语言用一种语法结构提供两种使用方式：

直接以命令方式交互使用，也可以嵌入程序设计语言中以程序方式使用；

SQL语言命令简洁，易学易用。只用几条简单的命令就可完成核心的功能；

SQL语言支持数据库的三层模式结构，使存储文件对用户来说是透明的。

数据类型

各种SQL的实施方案一般都有自己特有的数据类型，在分析和解决实际问题时需要注意

操作符

操作符是保留的字或字符，主要用于在子句中执行比较和数据学运算等操作。在SQL语句中操作符用于确定条件和建立语句中多个条件之间的连接。操作符一般可分为比较操作符、逻辑操作符、用于限制条件的操作符和数学运算操作符

1、比较操作符

“=” “ >” “ <” “<=” “>=” “<>”

Select emp_name from t_employee
where emp_age > 40

Select emp_name from t_employee
where emp_code = ‘123556’

2、逻辑操作符

IS NULL

Select emp_name from t_employee
where emp_dept is null

Select emp_name from t_employee
where emp_code = ‘null’

BETWEEN … AND …

Select emp_name from t_employee
where emp_age between 30 and 40

IN

Select emp_name from t_employee
where emp_code in (‘123’,’234’,’567’)

LIKE

Select emp_name from t_employee
where salary like ’%50%’

Select emp_name from t_employee
where salary like ’50%’

Select emp_name from t_employee
where salary like ’_2%3’

Select emp_name from t_employee
where salary like ’2_ _ _3’

Select emp_name from t_employee
where salary like ’_50_ _1’

注意：在不同的SQL实现中通配符不同。有的系统中使用‘＊’，‘？’作为通配符

3、连接符

AND

Select emp_name from t_employee where emp_old > 30 and emp_salary = 5000

OR

Select emp_name from t_employee where emp_old >30 or emp_salary = 5000

NOT

可与逻辑操作符相结合形成新的操作符

NOT BETWEEN

Select emp_name from t_employee where emp_salary not between 2000 and 5000

NOT IN

Select emp_name from t_employee where emp_salary not in (‘2000’,’5000’,’3000’)

NOT LIKE

Select emp_name from t_employee where emp_salary not like ‘200%’

IS NOT NULL、NOT EXISTS

Select emp_name from t_employee where not exists (select emp_id from t_employee where emp_id =‘999999’)

4、算术操作符

+、-、 * 、/

Select salary from t_pay where salary +allowance > 2000

Select salary from t_pay where salary * 10 > 20000

函　数

函数是SQL的关键字，用于操纵数据列的值来达到输出的目的。函数通常是和列名或表达式相联系的命令,包括单行函数、统计函数、格式模型等。

单行函数

单行函数主要分为数值函数、字符函数、日期函数、转换函数等。

统计函数

统计函数主要用于累加、合计和显示数据极限的函数。

格式模型

指定从数据库中检索数据的格式或用于不同数据类型的格式转换。

表达式、条件

表达式由一个或多个值、运算符和函数组合而成，可计算出一个值，其数据类型一般为它的成分的数据类型。

一个条件是由一个或多个表达式及逻辑运算符组合而成的，计算可得TRUE,FALSE或空。

关系数据库的设计理论

前面介绍了关系数据库的基本概念、关系模型的三个部分以及关系数据库的标准语言。下面首先介绍几个与关系理论密切基本概念，如函数依赖、模式分解等。

函数依赖

函数依赖的定义为：R(U)是属性集U上的关系模式。X和Y是U的子集。若对于R(U)的任意一个可能的关系r，若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，则称为“X函数确定Y”或“Y函数依赖于X”。记作：X→Y

属性之间的关系

设A，B为某实体集的两个属性集，

- 1－1关系：若对于A中的任一具体属性值，在B中至多有一个属性值与之对应，而对于B中的任一具体属性值，A中也至多有一个值与之对应，则称A、B两个属性值之间是1－1关系

属性之间的关系

- **1－m关系：若属性集A中的一个属性值至多与另一个属性集B中的一个属性值相关，而B中的一个属性值却可与A中的m个属性值有关，则称该两个属性集之间的关系为从B到A的1－m关系**
- **m－m关系：若属性集A中任意一个值都与另一属性集B中的m个值有关，反之亦然，则称AB之间是m-m关系**

函数依赖与属性的关系

- 如果X,Y是1－1 关系，则存在X→Y或者Y→X
- 如果X,Y是m－1 关系，则存在X→Y但是Y↛X
- 如果X,Y是m－m 关系，则X，Y之间不存在函数依赖关系

完全函数依赖

设X、Y为关系R中的属性集，若Y完全函数依赖于X，是指Y函数依赖于X而并不函数依赖于X中的任一子集，记作：$X \xrightarrow{f} Y$

传递函数依赖

设X、Y、Z为关系R中的属性集，若$X \rightarrow Y$但是$Y \not\rightarrow X$而$Y \rightarrow Z$，则称Z对X为传递函数依赖关系。

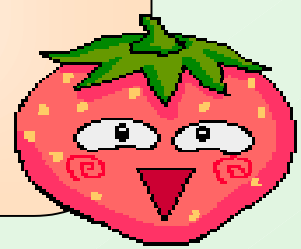
函数依赖对关系模式的影响

数据冗余问题

数据更新异常

数据插入异常

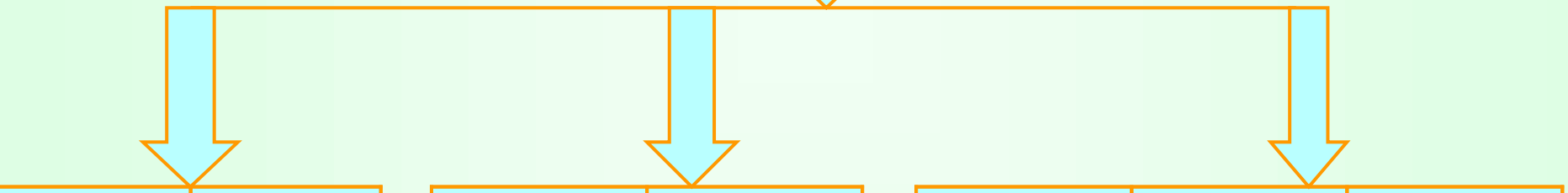
数据删除异常



模式分解

要解决上述问题的方法就是进行模式分解，即把一个关系模式分解成两个或多个关系模式，在分解的过程中消除那些“不良”的函数依赖，从而获得好的关系模式。

仓库号	地点	设备号	设备名	库存数
1	北京	D1	打印机	10
2	广州	D2	计算机	5
1	北京	D2	计算机	5
2	广州	D1	打印机	3



仓库号	地点
1	北京
2	广州

设备号	设备名
D1	打印机
D2	计算机

仓库号	设备号	库存数
1	D1	10
2	D2	5
1	D2	5
2	D1	3

规范化

规范化理论是的1971年首先提出的，目的是要设计“好的”关系数据模式。规范化是的关系型数据库中减少数据冗余的过程。除了数据以外，在数据库中，名称、对象名称和形式都需要规范化。

数据库设计时要考虑的因素

- 用户怎样访问数据库；
- 何种数据将存储在数据库中；
- 用户要求何种特权；
- 哪种数据被访问的次数最多；
- 数据在数据库中怎样被分组；
- 数据在数据库中怎样互相联系；
- 怎样保证数据的安全性。

范式

范式是衡量数据库规范的层次或深度，数据库规范化层次由范式来决定，根据关系模式满足的不同性质和规范化的程度，把关系模式分为第一范式、第二范式、第三范式、BC范式和第四范式等，范式越高、规范化的程度也越高，关系模式则越好。

第一范式（1NF）

- 如果一个关系模式R的每个属性值都是不可再分的数据单位，则称R满足第一范式

E-NO	ENAME	JOB	SALARY	
			S1	S2

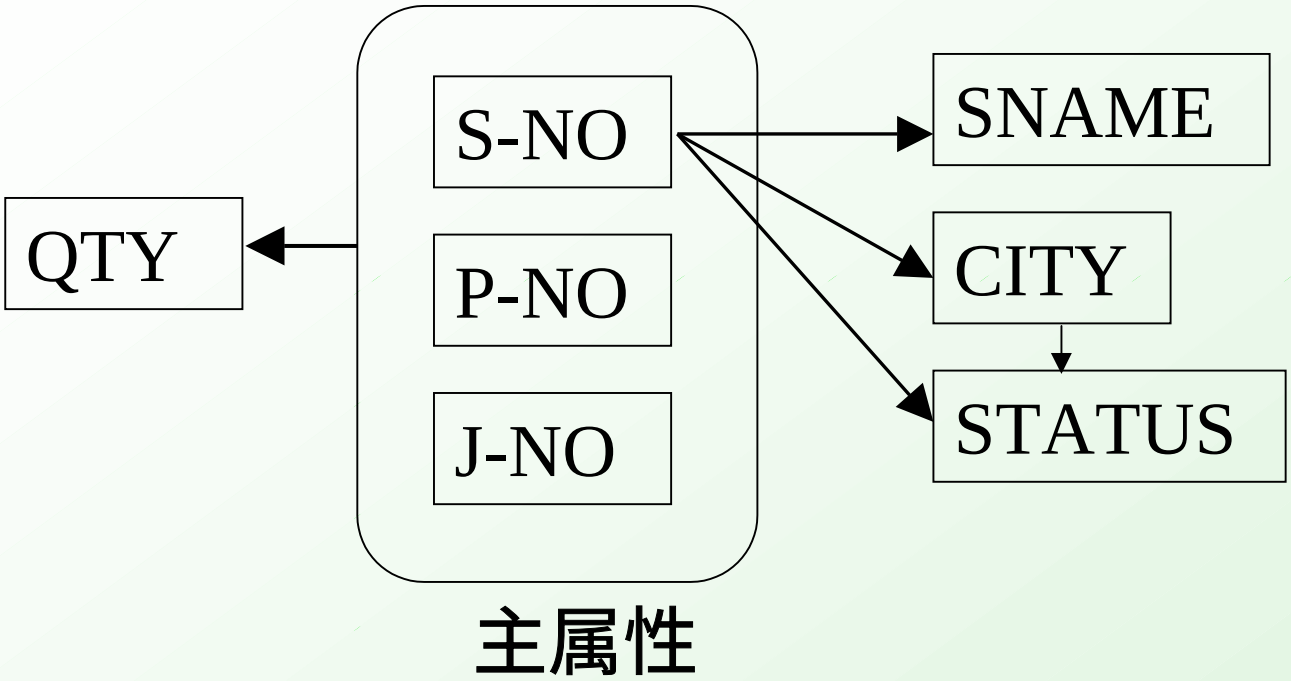
E-NO	ENAME	JOB	S1	S2

第二范式（2NF）

- 如果一个关系模式R满足第一范式，且非主属性完全函数依赖于主属性（主关键字），则称R满足第二范式

S-NO	SNAME	STATUS	CITY	P-NO	QTY	J-NO

上述关系的有向图表示

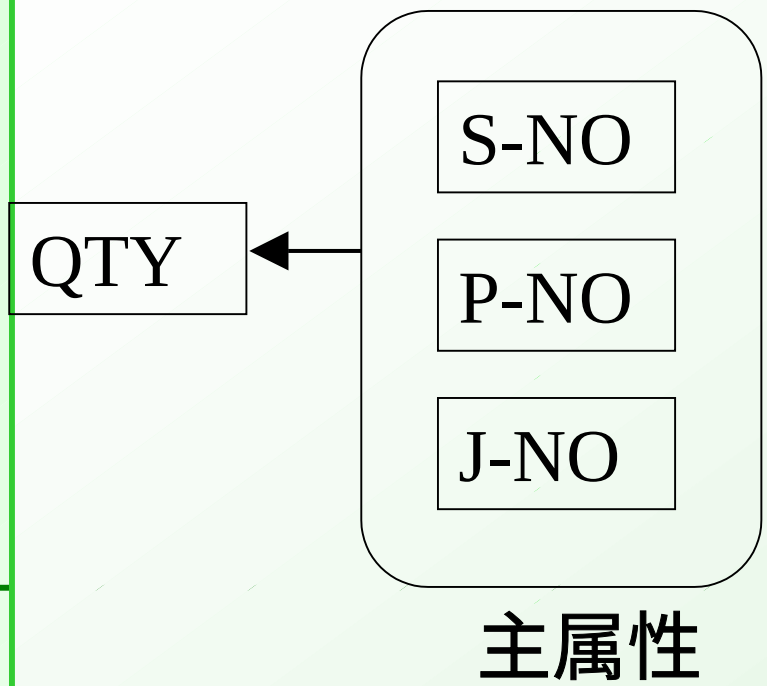


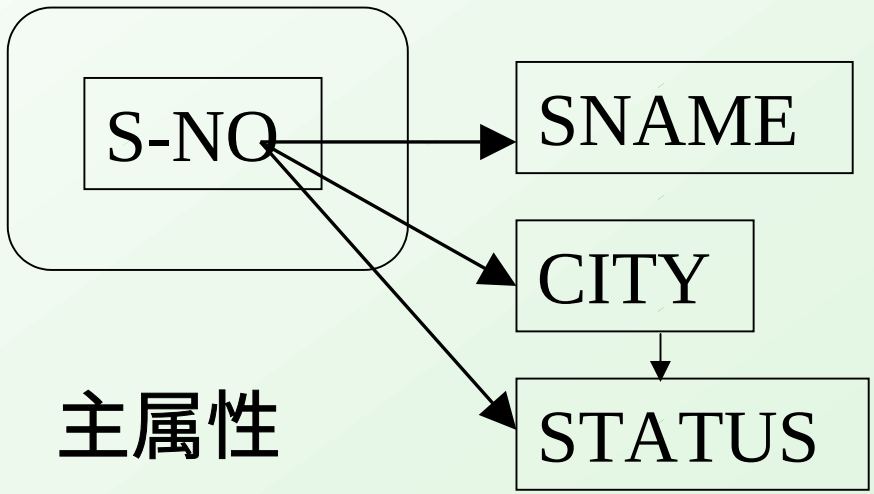
- 属性QTY完全依赖于主属性（S-NO,P-NO,J-NO）,属性SNAME、CITY、STATUS只是部分函数依赖于主属性（只是依赖于其中的S-NO），不满足2NF

存在的问题

- 插入异常：供应商没有零件时，该供应商的有关信息不能插入关系中（因主关键字不全）
- 删除异常：当删除供应商提供的最后一批零件时，有关该供应商的信息将丢失
- 更新异常：当某供应商信息需要更改时，需要更新所有与该供应商有关的记录，否则将出现数据的不一致，这是由于数据冗余引起的

改进





规范化的优缺点

- 提供了大型的总体数据库组织；
- 减少了数据冗余；
- 保证了数据库中数据的一致性；
- 提供了更好的数据库安全性处理；
- 降低了数据库的可操作性；