# 726数据库--博客

## 关系的完整性

**主要包括域完整性、实体完整性和参照完整性三种。  
1．域（列）完整性** 域完整性是对数据表中字段属性的约束，通常指**数据的有效性**,它包括字段的值域、字段的类型及字段的有效规则等约束，它是由确定关系结构时所定义的字段的属性决定的。限制数据类型,缺省值,规则,约束,是否可以为空,域完整性可以确保不会输入无效的值.。  
**2．实体（行）完整性**   实体完整性是对关系中的记录唯一性，也就是**主键的约束**。准确地说，实体完整性是指关系中的主属性值不能为Null且不能有相同值。定义表中的所有行能唯一的标识,一般用主键,唯一索引 unique关键字,及identity属性比如说我们的身份证号码,可以唯一标识一个人.   
**3．参照完整性**也就是对外键的约束。准确地说，参照完整性是指关系中的外键必须是另一个关系的主键有效值，或者是NULL。参考完整性维护表间数据的有效性,完整性,通常通过建立外部键联系另一表的主键实现,还可以用触发器来维护参考完整性

## 数据模型

模型是对现实世界的抽象。在数据库技术中，表示实体类型及实体类型间联系的模型称为“数据模型”。它可分为两种类型：概念数据模型和结构数据模型。

1**概念数据模型：**是独门于计算机系统的模型，完全不涉及信息在系统中的表示，只是用来描述某个特定组织所关心的信息结构。

2**结构数据模型**：是直接面向数据库的逻辑结构，是现实世界的第二层抽象。这类模型涉及到计算机系统和数据库管理系统，所以称为“结构数据模型”。

结构数据模型应包含：数据结构、数据操作、数据完整性约束三部分。它主要有：层次、网状、关系三种模型。

**层次模型：**用树型结构表示实体类型及实体间联系的数据模型。

**网状模型：**用有向图结构表示实体类型及实体间联系的数据模型。

**关系模型：**是目前最流行的数据库模型。其主要特征是用二维表格结构表达实体集，用外鍵表示实体间联系。关系模型是由若干个关系模式组成的集合

## 复习提纲

### 1. 数据抽象

共有三个层次：物理层、逻辑层和视图层。在逻辑层使用的数据模型包括两类：一类是概念数据模型，主要用于**[数据库](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "http://blog.csdn.net/yutianzuijin/article/details/_blank)**设计，它能被一般的用户理解，与人的思维表达方式比较接近。这样的模型有实体-联系模型（ERM）；另一类是逻辑数据模型，按计算机系统的观点对数据建模，使得数据更适合用计算机加以表示。这里模型主要用于DBMS的实现，比如关系模型、面向对象模型、层次模型和网状模型。设计师构建数据库模式的方法通常是首先使用E-R模型在高层对数据建模，然后再将其转变成关系模型。在物理层使用的数据模型称为物理数据模型。

数据模型通常由**[数据结构](http://lib.csdn.net/base/datastructure" \o "算法与数据结构知识库" \t "http://blog.csdn.net/yutianzuijin/article/details/_blank)**、数据操作和完整性约束三部分组成。

### 2.     数据库语言

根据功能不同，可以将数据库系统提供的语言分成三种类型：1）数据定义语言（DDL），用于定义数据库模式；2）数据操纵语言（DML），用于对数据库进行查询和更新；3）数据控制语言（DCL），用于对数据进行权限管理。

### **3.     数据库模式**

根据数据的不同抽象层次，数据库有三级模式：物理模式（内模式）在物理层描述数据库中全体存储结构和存取方法，而逻辑模式（概念模式）则在逻辑层描述数据库中全体数据的逻辑结构和特征。在视图层也可分为若干模式，称为子模式（外模式），它描述了数据库用户能够看见和使用的局部数据的逻辑结构和特征。通常一个数据库只有一个物理模式和一个逻辑模式，但是子模式有若干个。

### **4.     E-R图**

E-R图由以下元素构成：1）矩形，代表实体型；2）椭圆，代表属性；3）菱形，代表联系；4）线段，将属性和实体性相连，或将实体型和联系相连。

### 5.     关系模型

关系模型由关系数据结构、关系操作集合和关系完整性约束三部分构成。在关系模型中，现实世界实体以及实体间的联系均用关系来表示。

关系模型中常用的关系操作包括：选择、投影、连接、除、并、交、差等查询操作和增加、删除、修改两大部分。关系操作的特点是集合操作方式，即操作的对象和结果都是集合。关系操作可以使用两种方式定义：基于代数的定义称为关系代数；基于逻辑的定义称为关系演算。由于使用变量的不同，关系演算又分为元组关系演算和域关系演算。

关系模型允许定义三类完整性约束：实体完整性、参照完整性和用户定义完整性。其中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。实体完整性规则是：关系的主码不能取空值。参照完整性规则是：外码必须是另一个表中主码的有效值，或者是“空值”。

### **6.     连接运算**

连接运算是从两个关系的乘运算结果中选取属性间满足一定条件的元组，构成新的关系。连接运算有两种：等值连接和自然连接。自然连接要求两个关系中进行比较的分量必须是相同的属性组，并且在结果中把重复的属性列去掉。

### **7.     SQL**

关系数据库系统支持三级模式结构，其模式（数据库）、外模式和内模式中的基本对象有表、视图和索引。因此SQL的数据定义功能包括模式定义、表定义、视图定义和索引定义。SQL通常不提供修改模式定义、修改视图定义和修改索引定义。用户如果想修改这些对象，只能先将它们删除，然后再重建。

### **8.     基本表的操作**

1)       创建表

create table 基本表名

完整性约束主要有三种子句：主键子句（primary key）、外键子句（foreignkey）和检查子句（check）。

2)       修改表

alter  table  <基本表名>  add  <列名>  <类型>

alter  table  <基本表名>  drop  <列名>  <类型>  [cascade | restrict]（cascade表示所有引用到该列的视图和约束也要一起自动删除；restrict表示在没有视图或约束引用该属性时，才能在本表中删除该列，否则拒绝删除。）

alter  table  <基本表名>  modify  <列名>  <类型>

3)       撤销表

drop  table  <基本表名>  [cascade | restrict]

9.     SQL查询

where子句中可以使用下列运算符：

l  算术运算符

l  逻辑运算符

l  字符串匹配运算符，包括like，not like

l  集合成员资格运算符，包括in，not in

l  谓词，包括exists，all，some，unique

l  聚合函数，包括avg，min，max，sum和count

l  还可以是另一个select语句

select语句完整语法：

        select  目标表的列名或列表达式序列

        from 基本表名和（或）视图序列

        [where 行条件表达式]

        [group by  列名序列]

               [having  组条件表达式]

        [order by 列名[asc | desc]]

整个语句的执行过程如下：

1)       读取from子句中基本表、视图的数据，执行笛卡尔积操作；

2)       选取满足where子句中给出的条件表达式的元组；

3)       按group子句中指定列的值分组，同时提取满足having子句中组条件表达式的那些组；

4)       按select子句中给出的列名或列表达式求值输出；

5)       order子句对输出的目标表进行排序，按附加说明asc升序排列，或按desc降序排列。

10. 视图

创建视图：

        create view <视图名> [<列名> <列名>…]

        as <子查询>

        [with check option]

with checkoption表示对视图进行增删改是要保证操作的行满足视图定义中的谓词条件（即子查询中的条件表达式）。

视图最终是定义在基本表之上的，对视图的一切操作最终也要转换为对基本表的操作。视图的好处：

l  视图能够简化用户的操作

l  视图是用户能以多种角度看待同一数据

l  视图对重构数据库提供了一定程度的逻辑独立性

l  视图能够对机密数据提供安全保护

#### 知识点：笛卡尔积

设A,B为集合，用A中元素为第一元素，B中元素为第二元素构成有序对，所有这样的有序对组成的集合叫做A与B的笛卡尔积

笛卡尔乘积是指在数学中，两个[集合](https://baike.baidu.com/item/%E9%9B%86%E5%90%88" \t "https://baike.baidu.com/item/%E7%AC%9B%E5%8D%A1%E5%B0%94%E4%B9%98%E7%A7%AF/_blank)*X*和*Y*的笛卡尓积（Cartesian product），又称[直积](https://baike.baidu.com/item/%E7%9B%B4%E7%A7%AF" \t "https://baike.baidu.com/item/%E7%AC%9B%E5%8D%A1%E5%B0%94%E4%B9%98%E7%A7%AF/_blank)，表示为*X* × *Y*，第一个对象是*X*的成员而第二个对象是*Y*的所有可能[有序对](https://baike.baidu.com/item/%E6%9C%89%E5%BA%8F%E5%AF%B9" \t "https://baike.baidu.com/item/%E7%AC%9B%E5%8D%A1%E5%B0%94%E4%B9%98%E7%A7%AF/_blank)的其中一个成员。

假设集合A={a, b}，集合B={0, 1, 2}，则两个集合的笛卡尔积为{(a, 0), (a, 1), (a, 2), (b, 0), (b, 1), (b, 2)}

**11. 事务**

事务是一系列的数据库操作，是数据库应用程序的基本单元，是反映现实世界需要以完整单位提交的一项工作。事务是用户定义的一个数据库操作序列。

事务的四个特征：原子性、一致性、隔离性和持久性。

事务处理包括数据库恢复和并发控制。数据库恢复有两个目的：保证事务的原子性和使数据库能恢复到正确状态。

数据恢复的原理概括为冗余，建立冗余数据最常用的技术是数据转储和登录日志文件。数据转储是由DBA定期地将整个数据库复制到磁盘或另一个磁盘上保存起来的过程。

**12. 触发器**

一个触发器用来定义一个条件以及在该条件为真时需要执行的动作。通常，触发器的条件以断言的形式定义。动作以过程的形式定义。

**13. 索引**

## 概念模式

13) 概念模式：是数据库中全部数据的整体逻辑结构的描述。它由若干个概念记录类型组成。 概念模式不仅要描述概念记录类型，还要描述记录间的联系、操作、数据的完整性、安全性等要求。

(14) 外模式：是用户与数据库系统的接口，是用户用到的那部分数据的描述。

(15) 内模式：是数据库在物理存储方面的描述，定义所有的内部记录类型、索引和文件的组成方式， 以及数据控制方面的细节。

(16) 模式/内模式映象：这个映象存在于概念级和内部级之间，用于定义概念模式和内模式间的对应性， 即概念记录和内部记录间的对应性。此映象一般在内模式中描述。

(17) 外模式/模式映象：这人映象存在于外部级和概念级之间，用于定义外模式和概念模式间的对应性， 即外部记录和内部记录间的对应性。此映象都是在外模式中描述

# 831

## 1左联接，外联接等

**1. [MySQL](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "http://blog.csdn.net/beauty_1991/article/details/_blank)查询时，只有满足联接条件的记录才包含在查询结果，这种联接是（内联接）。**

**内联接：**典型的联接运算，使用像 = 或 <> 之类的比较运算符。包括相等联接和自然联接。内联接使用比较运算符根据每一表共有的列的值匹配两个表中的行。例如，检索 students 和 courses 表中学生标识号相同的所有行。

**外联接：**外联接可以是左向外联接、右向外联接或完整外部联接。   
在FROM子句中指定外联接时，可以由下列几组关键字中的一组指定：

**LEFT JOIN 或 LEFT OUTER JOIN**   
左向外联接的结果集包括 LEFT OUTER 子句中指定的左表的所有行，而不仅仅是联接列所匹配的行。如果左表的某行在右表中没有匹配行，则在相关联的结果集行中右表的所有选择列表均为空值。

**RIGHT JOIN 或 RIGHT OUTER JOIN**   
右向外联接是左向外联接的反向联接。将返回右表的所有行。如果右表的某行在左表中没有匹配行，则将为左表返回空值。

**FULL JOIN 或 FULL OUTER JOIN**   
完整外部联接返回左表和右表中的所有行。当某行在另一个表中没有匹配行时，则另一个表的选择列表列包含空值。如果表之间有匹配行，则整个结果集行包含基表的数据值。

**交叉联接：**交叉联接返回左表中的所有行，左表中的每一行与右表中的所有行组合。交叉联接也称笛卡尔积

## ****2.** [数据库](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "http://blog.csdn.net/beauty_1991/article/details/_blank)**事务正确执行的四个基本要素****

**ACID — 数据库事务正确执行的四个基本要素**

**--原子性（Atomicity）**、**一致性（Consistency）**、**隔离性（Isolation）**、**持久性（Durability）**

**<http://blog.csdn.net/beauty_1991/article/details/51209107>**

## **从表TABLE\_NAME中提取10条记录**

-- oracle:select \* from TABLE\_NAME where rownum <= 10;

-- sql server:select top 10 \* from TABLE\_NAME;

-- mysql: select \* from TABLE\_NAME limit 10;

-- db2:select \* from TABLE\_NAME fetch first 10 rows only;

--查询表的TOP 10 条记录：

set ROWCOUNT 10

select \* from tablename;

## ****SQL 约束有哪几种？****

NOT NULL: 用于控制字段的内容一定不能为空（NULL）。

UNIQUE: 控件字段内容不能重复，一个表允许有多个 Unique 约束。

PRIMARY KEY: 也是用于控件字段内容不能重复，但它在一个表只允许出现一个。

FOREIGN KEY: 用于预防破坏表之间连接的动作，也能防止非法数据插入外键列，因为它必须是它指向的那个表中的值之一。

CHECK: 用于控制字段的值范围。

DEFAULT: 用于设置新记录的默认值

## 5三大范式

**第一范式（1NF）**

所谓第一范式（1NF）是指**数据库表的每一列都是不可分割的基本数据项，同一列中不能有多个值，**即实体中的某个属性不能有多个值或者不能有重复的属性。如果出现重复的属性，就可能需要定义一个新的实体，新的实体由重复的属性构成，新实体与原实体之间为一对多关系。在第一范式（1NF）中表的每一行只包含一个实例的信息。

**第二范式（2NF）**

第二范式（2NF）是在第一范式（1NF）的基础上建立起来的，即满足第二范式（2NF）必须先满足第一范式（1NF）。**第二范式（2NF）要求数据库表中的每个实例或行必须可以被惟一地区分。**为实现区分通常需要为表加上一个列，以存储各个实例的惟一标识。简而言之，第二范式就是**非主属性非部分依赖于主关键字**。

**第三范式（3NF）**

满足第三范式（3NF）必须先满足第二范式（2NF）。简而言之**，第三范式（3NF）要求一个数据库表中不包含已在其它表中已包含的非主关键字信息。**

**（2）**

第一范式（1NF）：数据库表中的字段都是单一属性的，不可再分。这个单一属性由基本类型构成，包括整型、实数、字符型、逻辑型、日期型等。

第二范式（2NF）：数据库表中不存在非关键字段对任一候选关键字段的部分函数依赖（部分函数依赖指的是存在组合关键字中的某些字段决定非关键字段的情况），也即所有非关键字段都完全依赖于任意一组候选关键字。

第三范式（3NF）：在第二范式的基础上，数据表中如果**不存在非关键字段对任一候选关键字段的传递函数依赖**则符合第三范式。所谓传递函数依赖，指的是如 果存在"A → B → C"的决定关系，则C传递函数依赖于A。因此，满足第三范式的数据库表应该不存在如下依赖关系： 关键字段 → 非关键字段 x → 非关键字段y

## 6什么是事务？什么是锁？

答：事务就是被绑定在一起作为一个逻辑工作单元的SQL语句分组，如果任何一个语句操作失败那么整个操作就被失败，以后操作就会回滚到操作前状态，或者是上有个节点。为了确保要么执行，要么不执行，就可以使用事务。要将有组语句作为事务考虑，就需要通过ACID测试，即原子性，一致性，隔离性和持久性。

锁：在DBMS中，锁是实现事务的关键，锁可以保证事务的完整性和并发性。与现实生活中锁一样，它可以使某些数据的拥有者，在某段时间内不能使用某些数据或数据结构。当然锁还分级别的。

# [常见面试题整理--数据库篇](http://www.cnblogs.com/remember-forget/p/6140112.html)

## （一）什么是存储过程 ？有哪些优缺点？

存储过程是一些预编译的SQL语句。

更加直白的理解：存储过程可以说是一个记录集，它是由一些T-SQL语句组成的代码块，这些T-SQL语句代码**像一个方法一样实现一些功**能（对单表或多表的增删改查），然后再给这个代码块取一个名字，在用到这个功能的时候调用他就行了。

* 存储过程是一个预编译的代码块，**执行效率比较高**
* 一个存储过程替代大量T\_SQL语句 ，可以降低网络通信量，提高通信速率
* 可以一定程度上确保数据安全

## （二）索引是什么？有什么作用以及优缺点？

索引是对数据库表中一或多个列的值进行排序的结构，是帮助MySQL高效获取数据的数据结构

你也可以这样理解：索引就是加快检索表中数据的方法。数据库的索引类似于书籍的索引。在书籍中，索引允许用户不必翻阅完整个书就能迅速地找到所需要的信息。在数据库中，索引也允许数据库程序迅速地找到表中的数据，而不必扫描整个数据库。

**MySQL**数据库几个基本的索引类型：普通索引、唯一索引、主键索引、全文索引

* 索引加快数据库的检索速度
* 索引降低了插入、删除、修改等维护任务的速度
* 唯一索引可以确保每一行数据的唯一性
* 通过使用索引，可以在查询的过程中使用优化隐藏器，提高系统的性能
* 索引需要占物理和数据空间

## （三）什么是事务？

事务（Transaction）是并发控制的基本单位。所谓的事务，它是一个操作序列，这些操作要么都执行，要么都不执行，它是一个不可分割的工作单位。事务是数据库维护数据一致性的单位，在每个事务结束时，都能保持数据一致性。

## （四）数据库的乐观锁和悲观锁是什么？

数据库管理系统（DBMS）中的并发控制的任务是确保在多个事务同时存取数据库中同一数据时不破坏事务的隔离性和统一性以及数据库的统一性。

乐观并发控制(乐观锁)和悲观并发控制（悲观锁）是并发控制主要采用的技术手段。

* 悲观锁：假定会发生并发冲突，屏蔽一切可能违反数据完整性的操作
* 乐观锁：假设不会发生并发冲突，只在提交操作时检查是否违反数据完整性。

## （五） 使用索引查询一定能提高查询的性能吗？为什么

通常,通过索引查询数据比全表扫描要快.但是我们也必须注意到它的代价.

索引需要空间来存储,也需要定期维护；索引范围查询(INDEX RANGE SCAN)适用于两种情况:

* 基于一个范围的检索,一般查询返回结果集小于表中记录数的30%
* 基于非唯一性索引的检索

## （六）简单说一说****drop、delete****与****truncate****的区别

SQL中的**drop、delete、truncate**都表示删除，但是三者有一些差别

* **delete和truncate**只删除表的数据**不删除表的结构**
* 速度,一般来说: **drop> truncate >delete**
* **delete**语句是dml,这个操作会放到**rollback segement**中,事务提交之后才生效;  
  如果有相应的**trigger**,执行的时候将被触发. **truncate,drop**是ddl, 操作立即生效,原数据不放到r**ollback segment**中,不能回滚. 操作不触发**trigger**.

## （七）****drop、delete****与****truncate****分别在什么场景之下使用？

* 不再需要一张表的时候，用**drop**
* 想删除部分数据行时候，用**delete**，并且带上where子句
* 保留表而删除所有数据的时候用**truncate**

（八） 超键、候选键、主键、外键分别是什么？

超键：在关系中能**唯一标识元组的属性集**称为关系模式的超键。一个属性可以为作为一个超键，多个属性组合在一起也可以作为一个超键。**超键包含候选键和主键。**

候选键：是最小超键，即没有冗余元素的超键。

主键：数据库表中对储存数据对象予以唯一和完整标识的数据列或属性的组合。一个数据列只能有一个主键，且主键的取值不能缺失，即不能为空值（Null）。

外键：在一个表中存在的另一个表的主键称此表的外键。

## （九）什么是视图？以及视图的使用场景有哪些？

视图是一种虚拟的表，具有和物理表相同的功能。可以对视图进行增，改，查，操作，试图通常是有一个表或者多个表的行或列的子集。对视图的修改不影响基本表。它使得我们获取数据更容易，相比多表查询。

* 只暴露部分字段给访问者，所以就建一个虚表，就是视图。
* 查询的数据来源于不同的表，而查询者希望以统一的方式查询，这样也可以建立一个视图，把多个表查询结果联合起来，查询者只需要直接从视图中获取数据，不必考虑数据来源于不同表所带来的差异

## （十）说一说三个范式。

第一范式（1NF）：数据库表中的字段都是单一属性的，不可再分。这个单一属性由基本类型构成，包括整型、实数、字符型、逻辑型、日期型等。

第二范式（2NF）：数据库表中不存在非关键字段对任一候选关键字段的部分函数依赖（部分函数依赖指的是存在组合关键字中的某些字段决定非关键字段的情况），也即所有非关键字段都完全依赖于任意一组候选关键字。

第三范式（3NF）：在第二范式的基础上，数据表中如果不存在非关键字段对任一候选关键字段的传递函数依赖则符合第三范式。所谓传递函数依赖，指的是如 果存在"A → B → C"的决定关系，则C传递函数依赖于A。因此，满足第三范式的数据库表应该不存在如下依赖关系： 关键字段 → 非关键字段 x → 非关键字段y

# 基础

2. SQL语言包括数据定义、数据操纵(Data Manipulation),数据控制(Data Control)

数据定义：Create Table,Alter Table,Drop Table, Craete/Drop Index等

数据操纵：Select ,insert,update,delete,

数据控制：grant,revoke

**完整性约束：**实体完整性、参照完整性、用户定义完整性

.       列举几种表连接方式

Answer：等连接（内连接）、非等连接、自连接、外连接（左、右、全）

# 6.3-2018

4种基本：创建读取更新删除

## 《数据库》

DDL命令

ALTER、CREATE、DROP、UPDATE、DELETE、INSERT、SELECT

GRANK,REVOKE

Create table if not exists A(

Idstudent int **not nul**l auto\_increment,

Firstname varchar(45) NOT NULL

Primary key(idstudent)

)字段类型

Blob，boolean，char，date，datetime。Decimal,int ,number,time,timesatmp,varchar

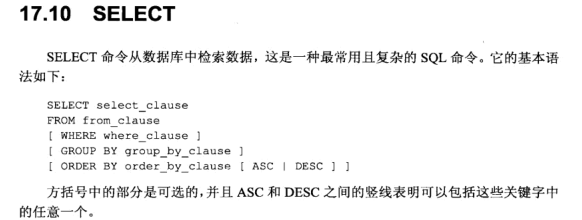
Smallint,bigint,tinyint

# 外键

Drop x on person（删除person表的x索引）

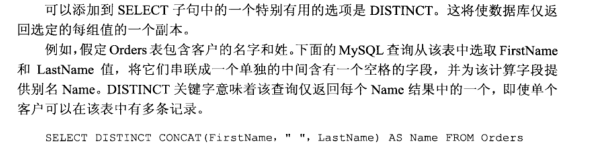
Drop table if exists A







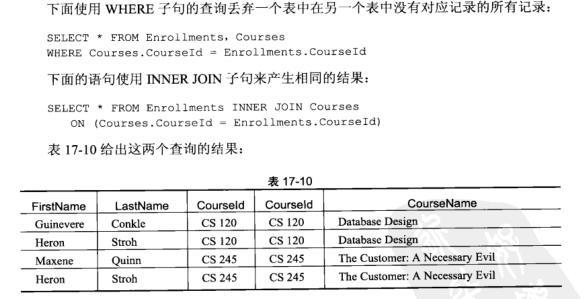
**Distinct**



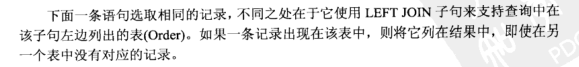
From子句



Where和inter join



Left、right join

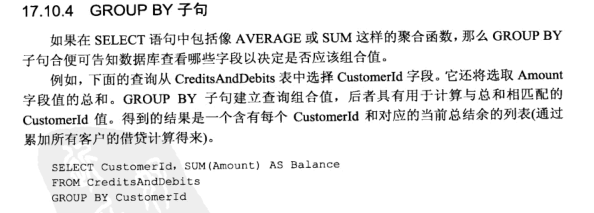




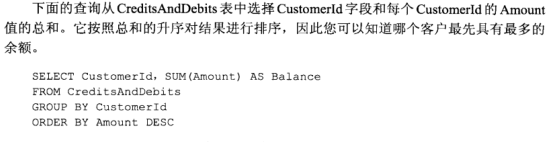
Union



Group by



Order by



Concat



Update set

