Java MySQL系列课程01

------ MySQL基础知识

本文档包含以下内容：

1. MySQL介绍
2. SQL基础
3. MySQL数据类型
4. MySQL运算符
5. 数据库范式

#### 前言：

MySQL是Java研发方向在校招中必考的内容，本节属于MySQL的基础知识，在工作中一定会用到，其中数据库范式还是校招考察的重点问题，怎样设计出遵循数据库范式的表？希望大家好好的学习理解范式设计。

#### 一 MySQL介绍：

详细的MySQL介绍，相信大家在网上搜索的内容会更详细，在这里给大家列出一些重点内容。

**了解MySQL：**

1. MySQL目前属于Oracle甲骨文公司，大家熟悉的关系型数据库有微软的SQL Server，甲骨文的Oracle和MySQL
2. MySQL分为企业版和社区版，其中社区版是完全免费并且开源的
3. MySQL和其它关系型数据库有一个非常大的区别，就是支持可更换的插件式的存储引擎，其中InnoDB非常强大
4. 目前goole、淘宝、百度、腾讯、新浪、facebook等大公司都在使用MySQL作为数据存储层方案

**从技术上再了解：**

1. MySQL设计成C/S客户端服务器模型，应用作为MySQL Client向MySQL Server发送请求，获取响应，因此MySQL非常适用于集群环境，方便做主从复制，读写分离操作
2. 为了提高效率，MySQL Client和MySQL Server如果处在不同主机上，当然是通过Socket进行网络通信的；如果它们在同一台机器上，那么Client和Server之间是通过共享内存进行通信的，效率比Socket通信更高
3. MySQL的服务器模块采用的是I/O复用+可伸缩的线程池，是实现网络高并发服务器的经典模型

#### 二 SQL基础：

SQL是结构化查询语言的缩写（Structure Query Language），它是关系型数据库的通用语言，非常强大，可以非常高效的进行数据路的增删改查操作，SQL+索引更是可以实现带各种附加条件的查询，效率很高。

**SQL 语句主要可以划分为以下 3 个类别：**

**DDL（Data Definition Languages）语句：**数据定义语言，这些语句定义了不同的数据段、数据库、表、列、索引等数据库对象的定义。常用的语句关键字主要包括 **create、drop、alter**等。  
**DML（Data Manipulation Language）语句：**数据操纵语句，用于添加、删除、更新和查询数据库记录，并检查数据完整性，常用的语句关键字主要包括 **insert、delete、update 和select** 等。  
**DCL（Data Control Language）语句：**数据控制语句，用于控制不同数据段直接的许可和访问级别的语句。这些语句定义了数据库、表、字段、用户的访问权限和安全级别。主要的语句关键字包括 **grant、revoke** 等。

**学习关系型数据库，SQL是必须熟练应用的，这需要不断的实践操作！**

#### 三 MySQL数据类型：

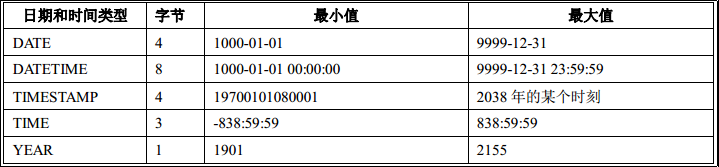
MySQL作为关系型数据库，在二维表中存数据，肯定要规定数据的类型，类型规定了数据的大小，因此使用的时候选择合适的类型，不仅会降低表占用的磁盘空间，间接减少了磁盘I/O的次数，提高了表的访问效率，而且索引的效率也和数据的类型息息相关。

1. **数值类型**



上图主要列出了整数类型和浮点数类型，在实际使用中，**根据具体的场景选择合适的数据类型很重要！**

1. **日期和时间类型**



日期类型也是做项目过程中，经常使用的类型信息，尤其是TIMESTAMP和DATETIME两个类型，但是**注意TIMESTAMP会自动更新时间，非常适合那些需要记录最新更新时间的场景，而DATETIME需要手动更新**。

1. **字符串类型**



字符串类型应该是使用最多的了，首先**注意char和varchar的区别**，然后注意**TEXT和BLOB的区别**，一般的字符串类型，我们选择varchar类型就可以了，如像博客之类，数据量比较大的类型就选择TEXT或者BLOB，但是TEXT只能存文本，而BLOB还可以存储二进制文件，比如图片等。

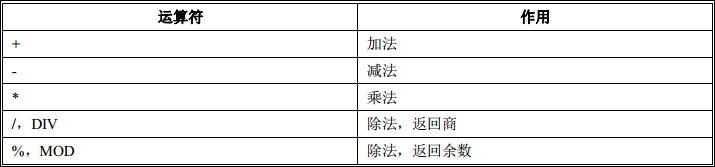
1. **enum枚举类型和set集合类型**

注意，这两个类型，都是限制该字段只能取固定的值，但是枚举字段只能取一个唯一的值，而集合字段可以取任意个值。

#### 四 MySQL运算符：

MySQL的运算符和Java语言很多运算符的含义是一样的，但也有区别，运算符多用在SQL语句当中，对SQL查询做各种条件过滤的。

1. **算数运算符**



1. **逻辑运算符**

NOT逻辑非 AND逻辑与 OR逻辑或

1. **比较运算符**



**举几个SQL示例，使用以上种类的运算符，如下：**

//使用算数运算符，把tcount\_tbl表中所有记录的runoob\_count字段更新，在原来的值上加1

**update tcount\_tbl set runoob\_count=runoob\_count+1;**

//使用逻辑运算符and和关系运算符>=，把成绩及格的学生信息获取出来

**select a.id a.name a.age a.sex from runoob\_user a inner join runoob\_score b on a.id = b.id and b.average>60.0;**

#### 五 数据库范式：

**应用数据库范式可以带来许多好处，但是最重要的好处归结为三点：**

1）减少数据冗余（这是最主要的好处，其他好处都是由此而附带的）

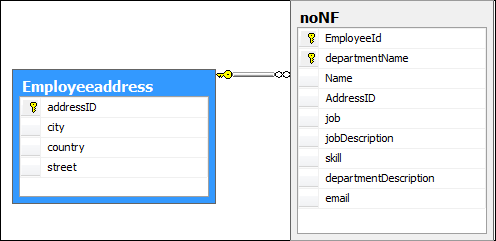
2）消除异常（插入异常，更新异常，删除异常）

3）让数据组织的更加和谐…

但是数据库范式绝对不是越高越好，范式越高，意味着表越多，多表联合查询的几率就越大，SQL的效率就变低。

1. **第一范式（1NF） 每一列保持原子特性**

列都是基本数据项，不能够再进行分割，否则设计成一对多的实体关系。例如表中的地址字段，可以再细分为省，市，区等不可再分割（即原子特性）的字段，如下：

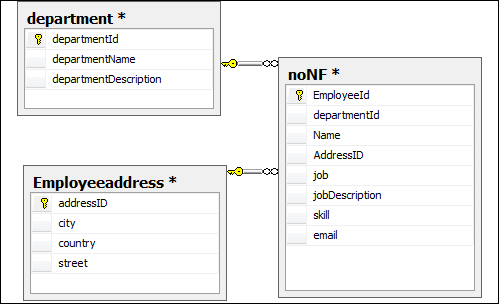
**（图1）**

上图的表就是把地址字段分成更详细的city,country,street三个字段，注意，不符合第一范式不能称作关系型数据库。

1. **第二范式（2NF） 属性完全依赖于主键(主要针对联合主键)**

非主属性完全依赖于主关键字，如果不是完全依赖主键，应该拆分成新的实体，设计成一对多的实体关系。

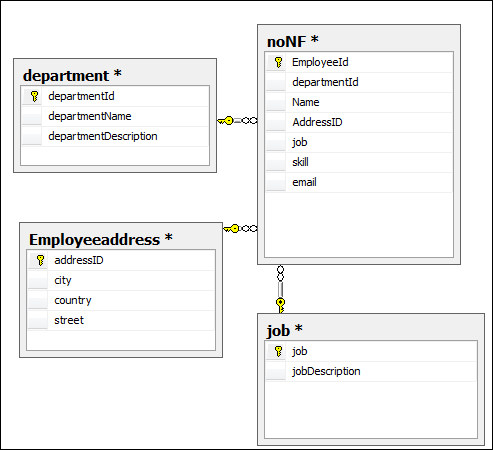
**示例：**选课关系表为SelectCourse(学号, 姓名, 年龄, 课程名称, 成绩, 学分)，（学号，课程名称）是联合主键，但是学分字段只和课程名称有关，和学号无关，相当于只依赖联合主键的其中一个字段，不符合第二范式。

**图（2） 《= 图（1）**

1. **第三范式（3NF） 属性不依赖于其它非主属性**

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。

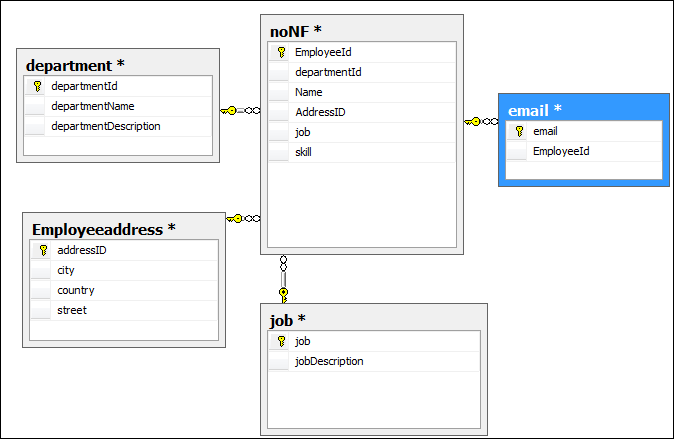
**示例：**学生关系表为Student(学号, 姓名, 年龄, 所在学院, 学院地点, 学院电话)，学号是主键，但是学院电话只依赖于所在学院，并不依赖于主键学号，因此该设计不符合第三范式，应该把学院专门设计成一张表，学生表和学院表，两个是一对多的关系。

**图（3） 《= 图（2）**

**注意：一般关系型数据库满足第三范式就可以了！**

1. **BC范式（BCNF） 每个表中只有一个候选键**

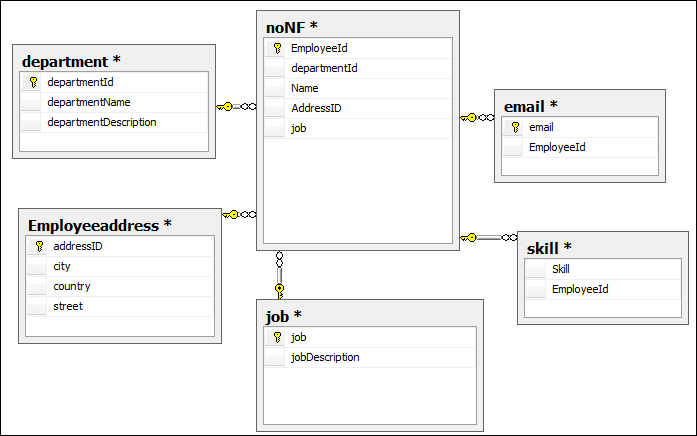
简单的说，BC范式是在第三范式的基础上的一种特殊情况，即每个表中只有一个**候选键（在一个数据库中每行的值都不相同，则可称为候选键）**，在上面**第三范式的noNF表（上面图3）**中可以看出，每一个员工的email都是唯一的（不可能两个人用同一个email），则此表不符合BC范式，对其进行BC范式化后的关系图为:



**图（4） 《= 图（3）**

1. **第四范式（4NF） 消除表中的多值依赖**

简单来说，第四范式就是要消除表中的多值依赖，也就是说可以减少维护数据一致性的工作。比如图4中的noNF表中的skill技能这个字段，有的人是“java，mysql”，有的人描述的是“Java，MySQL”，这样数据就不一致了，解决办法就是将多值属性放入一个新表，所以满足第四范式的关系图如下：



**图（5） 《= 图4**

从上面对于数据库范式进行分解的过程中不难看出，应用的范式越高，表越多。表多会带来很多问题：

**1 查询时需要连接多个表，增加了SQL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**

因此，并不是应用的范式越高越好，要看实际情况而定。**第三范式已经很大程度上减少了数据冗余，并且基本预防了数据插入异常，更新异常，和删除异常了**。