**大数据系统方案设计**

1. **基础数据收集和准备**

数据收集的基本原则：真实性、即时性、完备性、可靠性。

数据的来源：数据来源皆为互联网。目前比较知名的网站包括中国足彩网（www.zgzcw.com）、澳客足球（www.okooo.com）等。

数据采集的方式：数据收集采用网络爬虫技术，定期爬取网站上的最新数据。之后对数据网页进行提取，获取结构化的信息数据。

数据收集的内容：数据收集的内容包括有竞彩数据、单场数据、足彩数据；比赛双方球队的最近历史战绩、最近一场比赛的双方实力对比（以欧赔、亚盘数据进行分析）；该场比赛的各家欧赔数据、亚盘数据及实时变化、大小球对比数据等。

1. **数据库的设计**
2. **基本功能设计**
3. 同轮比赛关联分析

对同一联赛、同一赛季、同一轮次所有场次的比赛相关数据进行收集与整理，查找同欧赔（主胜、主负欧赔赔率差距不大于一给定域值，如0.03）的比赛场次，进行归类。我们认为，同欧赔的比赛场次属于关联场次。

同欧赔所使用的数据包括初始平均欧赔、Interwetten、Oplipkz等。

1. 战绩与实力差对比分析

对上一步分析出来的关联比赛场次，分别分析比赛对阵双方的实力差距。

球队实力值以本赛季积分榜、近六轮联赛积分榜、近六轮主（客）场比赛积分榜进行加权计算，计算公式如下：

S主=a1×S总+b1×S主+c1×S六

S客=a2×S总+b2×S客+c2×S六

实力差计算公式：

S差=S主－S客

在这里，对实力差值进行分档，初步共计按照十档差距进行计算。

1. 赔率差异常分析

取最近一场对阵双方比赛平均初始欧赔值进行分析比较，与比赛结果值进行对比与分析，判断本场比赛开出的欧赔值是否异常值。

1. 综合分析，推荐可能冷门结果

**数据库设计说明书**

# 引言

在使用任何数据库之前，都必须设计好数据库，包括将要存储的数据类型、数据之间的相互关系及数据的组织形式。数据库设计是指对于一个给定的应用环境，构造最优的数据库模式，建立数据库及其应用系统，使之能够有效存储数据。

为了合理组织和高效率地存取数据，目前最好的方式，就是建立数据库系统，因此在系统的总体设计阶段，数据库的建立与设计是一项十分重要的内容。由于数据库应用系统的复杂性，为了支持相关程序运行，数据库设计就变得异常复杂，因此最佳设计不可能一蹴而就，而只能是一种“反复探寻、逐步求精”的过程，也就是规划和结构化数据库中的数据对象以及这些数据对象之间关系的过程。

## 编写目的

数据库设计是系统设计的关键环节。如果把企业的数据比做生命所必需的血液，那么数据库的设计就是应用中最重要的一部分，是一个系统的根基。用于开发人员进行项目设计，以此为作编码的依据，同时也为后续的数据库维护工作提供良好的使用说明。这个数据模型应当是满足用户需求，既能合理地组织用户需要的所有数据，又能支持用户对数据的所有处理功能。并且要具有较高的范式，数据完整性好，效益高，便于理解和维护，没有数据冲突。

## 编写说明

|  |  |
| --- | --- |
| 名称 | 说明 |
| 数据库名称 | Soccer |
| 数据库系统 | MySQL5.7 |
| 客户端工具 | Navicat10.1.7 |
| 项目任务提出者 | 小东连 |
| 项目开发者 |  |
| 使用用户 | 小东连成员 |

## 定义

E－R图：实体关系图

## 参考资料

# 外部设计

外部设计是研究和考虑所要建立的数据库的信息环境，对数据库应用领域中各种信息要求和操作要求进行详细地分析，了解应用领域中的数据项、数据项之间的关系和所有的数据操作的详细要求，了解哪些因素对响应时间、可用性和可靠性有较大的影响等各方面的因素。

## 2.1. 标识符和状态

* 数据库表前缀：soccer\_
* 用户名：root
* 密码：test
* 权限：全部
* 有效时间：开发阶段
* 说明：系统正式发布后，数据库的用户名/密码等应进行修改，请在统一位置编写数据库连接字符串，在发行前予以改正。

## 2.2. 使用它的程序

本系统主要采用Java作为应用开发工具，使用MySQL作为后台数据库，Linux、Windows均可作为系统平台。

## 2.3. 约定

* 所有命名一定要具有描述性，杜绝一切拼音、或拼音英文混杂的命名方式。
* 字符集采用UTF-8，请注意字符的切换。
* 所有数据表第一个字段都是系统内部使用主键列、自增字段（或UUID字段）、不可空，名称为id，确保不把此字段暴露给最终用户。
* 除特别说明外，所有日期格式都采用int格式，无时间值。
* 除特别说明外，所有字段默认设置不允许为空，需要设置默认值。
* 所有普通索引的命名都是index\_加表名、设置索引的字段名组合，例如用户表User中name字段设置普通索引，则索引名称命名方式为index\_user\_name。

## 2.4. 支持软件

* 操作系统：Linux / Windows
* 数据库系统：MySQL
* 查询浏览工具：Navicat
* 命令行工具：mysql
* 注意：MySQL命令行环境下对中文支持不好，可能无法书写带有中文的SQL语句，也不要使用Navicat录入中文。

# 结构设计

数据库的结构设计中有许许多多需要考虑的因素，如数据库的背景、应用环境等方面需要有深入的了解。只有一个对所有这些因素都很了解的数据库设计专家所设计的数据库才能易于使用和维护，并且具有高效和一致的特征。虽然这样只对数据库设计过程有一个概要的了解，但是仍然有助于读者了解和掌握SQL，使读者可以很好的分析数据间的相互关系，在使用SQL进行报表的生成、子查询及视图等操作时，可以更好的进行操作。

## 3.1 概念结构设计

概念数据库的设计是进行具体数据库设计的第一步。概念数据库设计的好坏直接影响逻辑数据库的设计，关系到整个数据库的好坏。

在设计之前，已经对系统的数据流程图和数据字典进行了梳理，现在就是要结合数据规范化的理论，用一种模型将用户的数据要求明确地表示出来。概念数据库的设计应该极易于转换为逻辑数据库模式，又容易被用户所理解。

概念数据库设计中最主要的就是采用实体－关系（E－R关系图）数据模型来确定数据库的结构。

数据是表达信息的一种重要的量化符号，是信息存在的一种重要形式。数据模型则是数据特征的一种抽象，它描述的是数据的共性，页不是描述个别的数据。一般来说，数据模型包含两个方面内容：

* 数据的静态特性：主要包括数据的基本结构、数据间的关系和数据之间的相互约束等特性。
* 数据的动态特性：主要包括对数据操作的方法。

在数据库系统设计中，建立反映客观信息的数据模型，是设计中最为重要的，也是最基本的步骤之一。数据模型是连接客观信息世界和数据库系统数据逻辑组织的桥梁，也是数据库设计人员与用户之间进行交流的共同基础。概念数据库中采用的实体－关系模型，与传统的数据模型有所不同。实体－关系模型是面向现实世界，而不是面向实现方法的，它主要是使用方便，因而在数据库系统应用的设计中，得到了广泛应用。实体－关系模型可以用来说明数据库中实体的等级和属性。

以下是实体－关系模型中的重要标识：

* 在数据库中存在的实体
* 实体的属性
* 实体之间的关系

### 3.1.1 实体和属性的定义

按照定义的数据类型和属性创建实体和实体属性列表。实体形成表，如“用户”就是一个实体，属性则为表中的列，如对应于实体“用户”属性包含“用户名”、“用户ID”等。

**（一）实体**

实体就是实体－关系模型的基本对象，是现实世界中各种事物的抽象。凡是可以相互区别开并可以识别的事、物、概念等对象均可认为是实体。本系统数据库中，基本的实体列表如下：

* 用户
* 球队
* 联赛
* 比赛
* 欧赔
* 亚盘

在绘制实体－关系图（E－R图）时，实体出现在矩形中。如下图所示：



图1 表示实体的E－R图

一般来说，每个实体都相当于数据库中的一个表。上面介绍的实体都是强实体，每个实体都有自己的键。但是在实际应用中，经常存在一些实体，它们没有自己的键，这样的实体称为弱实体。弱实体中不同的记录有可能完全相同，难以区别，这些值依赖于另一个实体（强实体）的意义，必须与强实体联合使用。在创建了实体之后，就可以标识各个实体的属性了。

**（二）属性**

每一个实体都有一组特征或性质，称为实体的属性。实体的属性值是数据库中存储的主要数据，一个属性实际上相当于表中的一个列。如这体文章的属性如下：

* 文章标题
* 文章编号
* 文章简介
* 添加时间
* 文章内容

在绘制E－R关系图中，属性由椭圆包围，在属性和它所属的实体间使用直线进行连接。

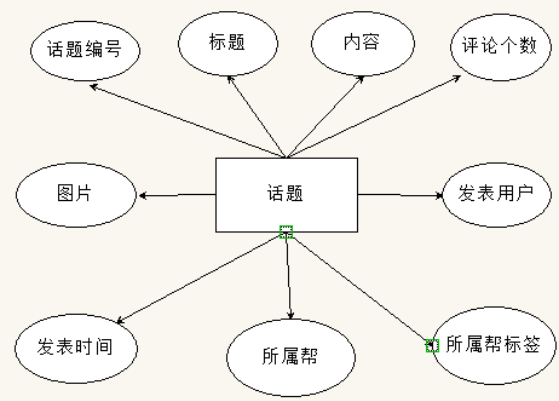


图2 话题实体与属性图

对于每个实体，都有其确定的主属性（实体中的主属性实际上相当于表中的主键），就可以唯一地确定实体的每个记录。最好是创建一个单独的属性作为主属性，在实体文章中可以选择“文章编号”作为主属性。在绘制E－R关系图中，主属性在属性下加下划线来说明。

### 3.1.2 绘制E－R图

实体－关系图是表现实体－关系模型的图形工具，简称E－R图。这节以数据库为例，给出一个完整的数据库的E－R图设计示例。下图给出了在E－R关系图中使用的各种元素的图形符号。

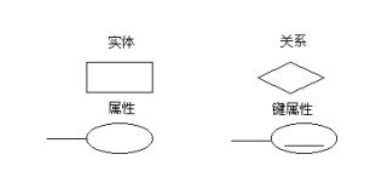


图3 实体－关系图中所使用的图例

在E－R图中，实体之间的关系以菱形表示，关系中各方面的表通过直线与菱形中的关系名称相连接。还要为每个关系命名一个“关系名称”，实体与关系相连的直线旁都根据关系的属性标注有“1”或“N”。E－R图为读者的数据库提供了一个不错的蓝图。

可以分成三步进行：首先设计局部E－R图；然后合并各局部E－R图，并解决可能存在的冲突，得到初步E－R图；最后修改和重构初步E－R图，消除其中的冗余部分，得到全局的最终E－R图，即概念模式。

设计全局E－R模式的目的不在于把若干局部E－R模式形式上合并为一个E－R模式，而在消除冲突使之成为能够被全系统中所有用户共同理解和接受的统一的概念模型。使设计人员仅从用户角度看待数据及处理要求和约束，产生一个反映用户观点的概念模式。

### 3.1.3 设计局部E－R模式

先设计局部E－R图，也称为用户视图。在设计初步E－R图时，要尽量充分地把组织中各部门对信息的要求集中起来，而不需要考虑数据的冗余问题。局部概念模型设计是从用户的观点出发，设计符合用户需求的概念结构。局部概念模型设计的就是组织、分类收集到的数据项，确定哪些数据项作为实体，哪些数据项作为属性，哪些数据项是同一实体的属性等。确定实体与属性原则：

* 能作为属性的尽量作为属性而不要划为实体；
* 作为属性的数据元素与所描述的实体之间的联系只能是1：n的联系；
* 作为属性的数据项不能再用其他属性加以描述，也不能与其它实体或属性发生联系。

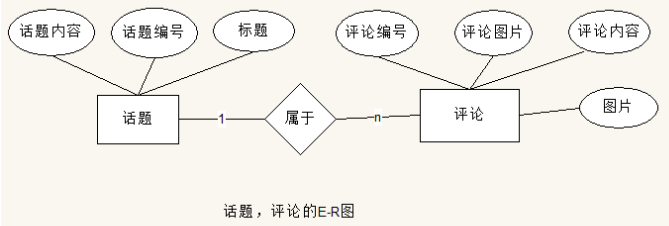
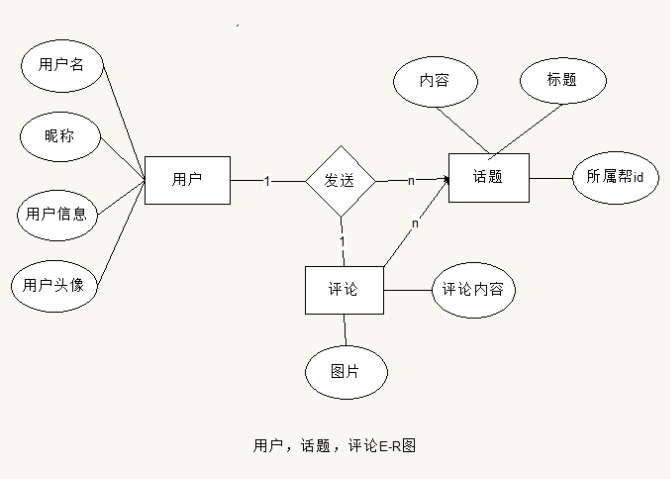


图4 实体－关系图中所使用的图例



### 3.1.4 设计全局E－R模式

综合各局部E－R图，形成总的E－R图，即用户视图的集成。所有局部E－R模式都设计好了之后，接下来就是把它们综合成单一的全局概念结构。全局概念结构不仅要支持所有局部E－R模式，而且必须合理地表示一个完整、一致的数据库概念结构。

先回顾一下概念数据库的设计，其中主要是实体－关系模型的建立。简要总结一下实体－关系模型建立的步骤：

* 对需求进行分析，从而确定系统中所包含的实体；
* 分析得出每个实体所具有的属性；
* 保证每一个实体有一个主属性，该主属性可以是实体的一个属性或多个属性的组合。主属性必须能唯一地描述每个记录。
* 确定实体之间的关系。

经过这些步骤后，读者就可以绘制出E－R图。之后再看看数据库的需要，判断是否获取了所需的信息，是否有遗漏信息等。然后再对E－R图进行修改，添加或删除实体与属性。

### 3.1.5 全局E－R模式的优化

在得到全局E－R模式后，为了提高数据库系统的效率，还应进一步依据处理需求对E－R模式进行优化。一个好的全局E－R模式，除能准确、全面地反映用户功能需求外，还应满足下列条件：

* 实体类型的个数尽可能少
* 实体类型所含属性个数尽可能少
* 实体类型间联系无冗余

## 3.2 逻辑结构设计

逻辑结构设计的任务是把概念设计阶段建立的基本E－R图，按照选定的内容管理系统软件支持的数据模型，转化成相应的逻辑设计模型。也就是将实体、实体间的关系等模型结构转变为关系模式，即生成数据库中的表，并确定表的列。下面是实体－关系模型生成表的方法。

**（一）任务**

将基本E－R图转换为与选用的产品所支持的数据模型相符合的逻辑结构。

**（二）过程**

* 将概念结构转换为现有数据库支持的关系模型。
* 从功能和性能要求上对转换的模型进行评价，看它是否满足用户要求。
* 对数据模型进行优化。

### 3.2.1 将E－R图转化为关系模型

在上文实体之间的关系的基础上，将实体、实体属性和实体之间的联系转换为关系模式。这种转换的原则是：

（1）一个实体转换为一个关系，实体的属性就是关系的属性，实体的码就是关系的码。

（2）一个联系转换为一个关系，联系的属性及联系所连接的实体的码都转化为关系的属性，但是关联的码会根据关系的类型变化。如果是

* 1：1联系，两端实体的码都成为关系的候选。
* 1：n联系，n端实体的码成为关系的码。
* m：n联系，两端的实体的码组成关系的码。

### 3.2.2 消除冗余

所有冗余的数据是指可由基本数据导出的数据，冗余的联系是指可由其他联系导出的联系。冗余数据和冗余联系容易破坏数据库的完整性，给数据库的维护增加困难，应当予以消除。本系统的冗余数据和冗余关系在概念结构设计中进行处理。

## 3.3 物理结构设计

数据库设计的最后阶段是确定数据库的物理设备上的存储结构和存储方法，也就是设计数据库的物理数据模型，主要是设计表结构。一般地，实体对应于表，实体的属性对应于表的列，实体之间的关系成为表的约束。逻辑设计中的实体大部分可以转换为物理设计中的表，但是它们并不是一一对应的。

### 3.3.1 设计数据表结构

在利用MySQL创建一个新的数据表以前，应当根据逻辑模型和数据字典先分析和设计数据表，描述出数据库中基本表的设计。需要确定数据表名称，所包含字段名称、数据类型、宽度及建立的主键、外键等描述表的属性的内容。

表1 比赛数据表结构设计

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | soccer\_match 用于保存比赛数据的表，表引擎为MyISAM类型、字符集为UTF-8 | | | |
| 列名 | 数据类型 | 属性 | 约束条件 | 说明 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| 补充说明 |  | | | |

### 3.3.2 创建数据表

通过数据库表结构的详细设计，再结合MySQL的创建数据表的语法，本系统的数据表的完整创建SQL语句如下表示。

### 3.3.3 数据表记录的输入

在创建数据表的同时可以根据系统提示输入记录，但是也可以暂时不输入记录。

# 数据库设计的其它说明

### 4.1 完整性

数据库应用对数据一般都具有一定的限制，这种限制称为完整性。数据库的完整性是保证数据库正确的关键。广义上说，数据库完整性包括数据库中数据的准确性和一致性。理想情况下，数据库软件提供了检查数据完整性的各种方法。

关系数据库系统应该保证输入的值符合其规定的数据类型，并保证值在系统支持的范围内。不同的关系数据库系统可能提供了不同的数据类型，但所有关系数据库系统都检查输入的值，并且拒绝不符合定义的数据类型的值或者不在系统支持范围内的值。一般来说，关系数据库支持3种完整性：

* 域约束
* 实体完整性约束
* 关联完整性约束

### 4.2 其它问题

索引：索引是数据库实现的重要组成部分，对于数据的检索、存储等有很大的影响。

安全性：在数据库中安全性是非常重要的问题。必须保证数据库数据的安全。

数据字典：数据字典是用来存储数据库中存储数据的描述信息和数据库管理系统需要的控制信息，数据字典与数据库管理软件密切相关。