《数据库概论》实验一：用SQL进行数据操作 实验报告

张涵之 191220154 191220154@smail.nju.edu.cn

实验环境

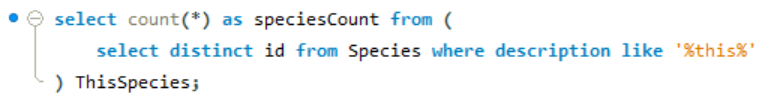
操作系统：Windows 10，64位操作系统, 基于x64的处理器

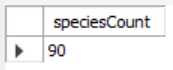
软件版本：MySQL Community 8.0.26.0

实验过程

1. 有多少species的description中含有单词“this”？返回：(speciesCount)

直接使用通配符与species中的description进行匹配，用count()统计



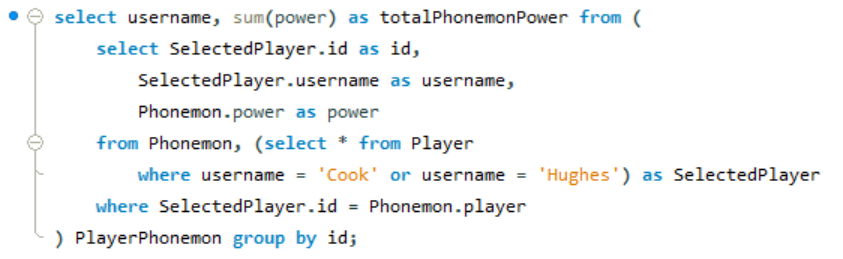


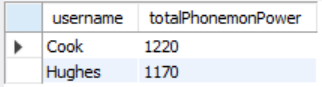
1. 对于player‘Cook’与player‘Hughes’，显示他们的username和各自拥有的Phonemon的总能量。返回：(username, totalPhonemonPower)

先选出username为‘Cook’和‘Hughes’的player为SelectedPlayer

再从SelectedPlayer和Phonemon中选出两个被两个player捕捉的phonemon

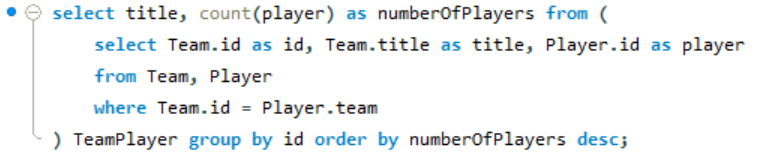
将这些phonemon按player id分组，用sum()计算power的和

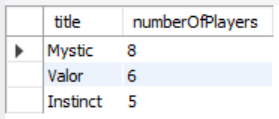




1. 每个team有多少名player？按玩家数量降序返回：(title, numberOfPlayers)

先按team id和player team连接，对结果进行分组count(), 降序输出

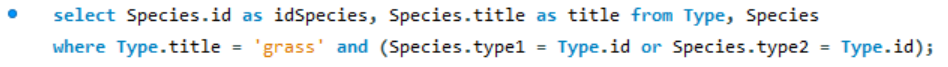


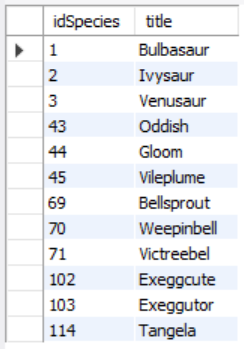


1. 哪些species具有type‘grass’？返回：(idSpecies, title)

从Species和Type中找到species type1对应type id的type title为‘grass’

或type2对应的type id的type title为‘grass’的并输出

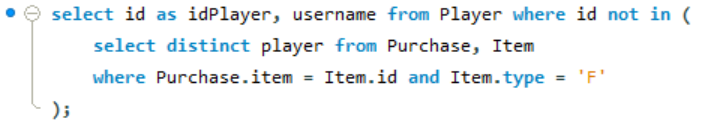


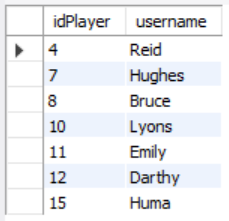


1. 列出从未购买过food的player。返回：(idPlayer, username)

先找出所有在Purchase中有对应item的type为‘F’的player的id

即所有购买过food的player，再找出不在这个集合中的player





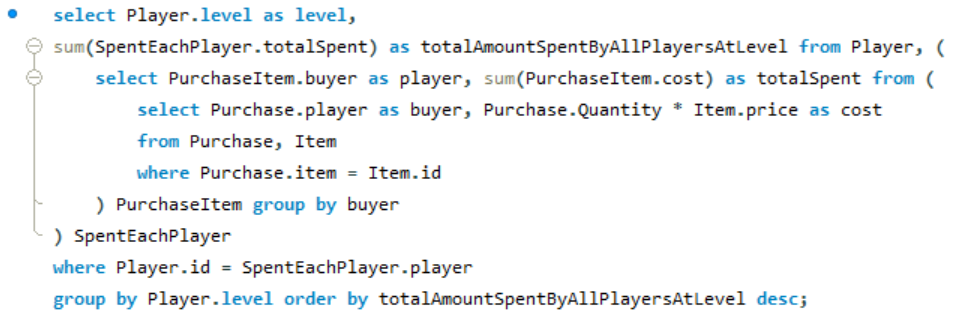
1. 以金额降序列出每一特定level以及该等级的所有player在购买上花费的总金额。返回：(level, totalAmountSpentByAllPlayersAtLevel)

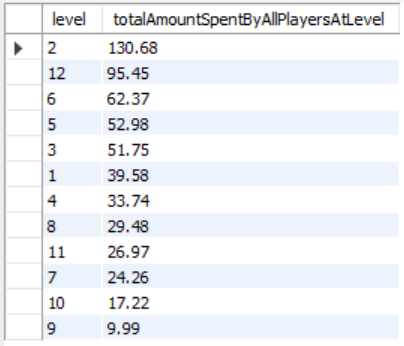
先将Purchase与Item对应，按quantity和price算出每个purchase的金额

再按purchase的player id进行分组，算出每个player消费总金额

然后对SpentEachPlayer按player的level分组，算出每个level消费总金额

最后按每个level对应的totalAmountSpentByAllPlayersAtLevel降序输出

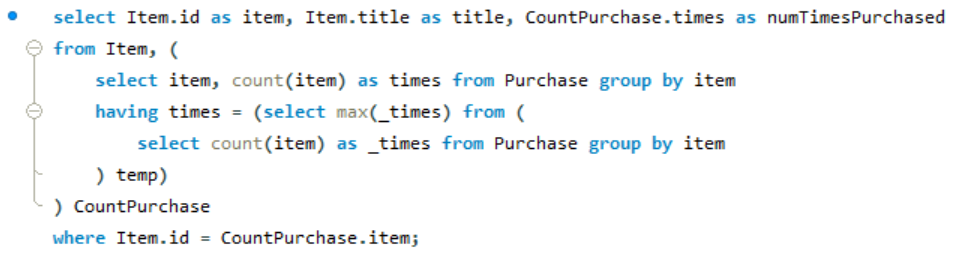


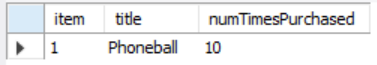


1. 什么item购买次数最多（含并列）？返回：(item, title, numTimesPurchased)

先按purchase的item id分组找出购买次数最多的item被购买的次数

再找出所有购买次数等于这个值的item，输出id，title和购买次数





1. 找到可获取的食物的数量，和购买所有种类食物至少各一次的玩家。返回：(playerID, username, numberDistinctFoodItemsPurchased)

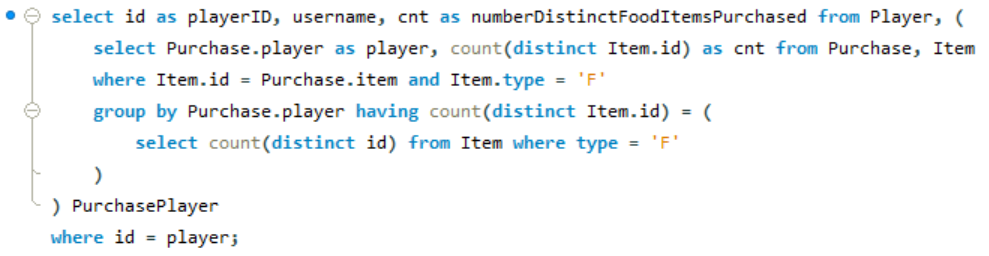
首先用count()统计Item中type为‘F’的数量，即“可获取的食物数量”

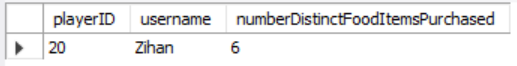
（理论上，Item中type为‘F’的id应该与Food中的id一一对应，否则如果某个Food的id在item中找不到对应，则它是一个不可（通过购买）获取的食物，因为Purchase中的id是标记item的。故这里不直接统计Food）

对Purchase取item type为‘F’的，按player进行分组，

用count distinct根据purchase中的item id统计每个player买过的食物种数

取购买食物种数等于“可获取食物数量”的player，输出id和username



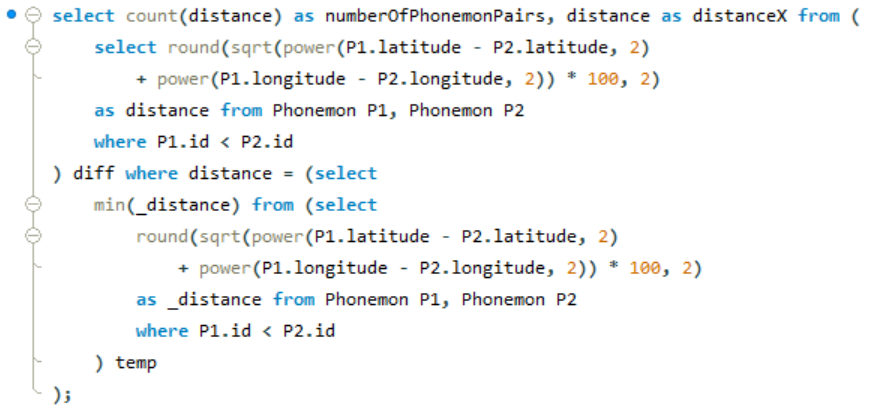


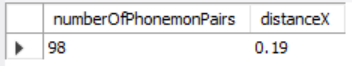
1. 将距离最近的两个Phonemon之间的欧氏距离称为X。计算相互之间距离为X的Phonemon对的数量。返回：(numberOfPhonemonPairs,distanceX)

对Phonemon取副本P1、P2，通过P1.id < P2.id防止pair的重复计算

对每个pair用公式算出最近欧氏距离X，用round()保留两位小数

再计算一次每对之间距离，取等于X的对并进行计数





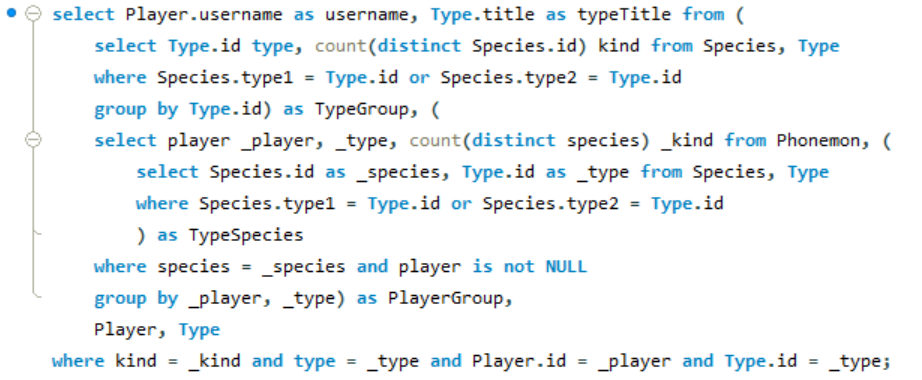
1. 列出捕捉了任一特定type中每一species至少各一个Phonemon的玩家的名称以及该类型的名称。返回：(username, typeTitle)

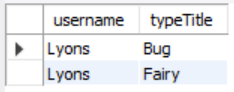
先根据Type，Species找出每种type拥有的species种数

从Player，Type，Phonemon找出每个player拥有每个type的species种数

从TypeGroup，PlayerGroup选出type和对应species种数相同的组

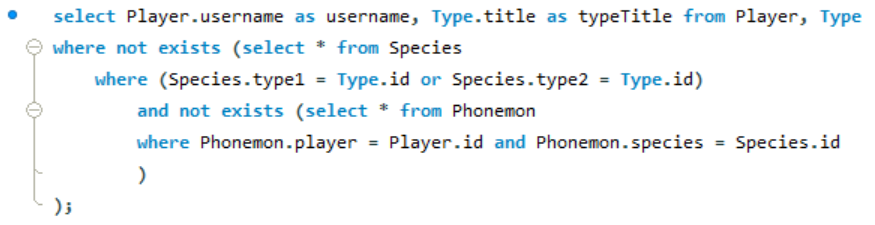
结合Type和Player输出这些组的player username和type title





实验中遇到的困难及解决办法

1. 有些表定义的域在手册中找不到解释，如Food的manna不知道是什么有什么用，令人有些困惑，虽然对实验本身没有什么影响；
2. 实验任务部分比较难读，需要结合上下文实际代码进行理解，如“可获取的食物的数量”究竟该统计Food还是Item中的‘F’取决于具体含义；
3. 数据库规模较大，输出结果是否正确很难手动验算，可以考虑根据不同的逻辑实现同一个任务，并将输出结果进行比较，如10中找出集齐了特定type中所有species的player，上面已经用计算每个type下species数量和每个玩家捕捉过每个type的species数量进行比对的方法实现。除此之外，还可以单独对每个player，type组合判断是否存在某一个被这个type包含而player没有捕捉过的species（即没有对应player的phonemon属于这个species），如果不存在这样的species，那么这个player就集齐了这个type，否则没有集齐。这种方法的输出结果与上面的相同，其他一些较复杂的任务也可以用这种方法进行验证，从而最大程度上保证代码运行结果的正确性。



参考文献及致谢

课程课件（主要是database\_03\_2.pdf）

部分函数使用（如round()的参数和具体用法）参考了网络，如runnoob.com的SQL教程部分：https://www.runoob.com/sql/sql-tutorial.html