项目#4：数据库编程

ECE 650 – 2022 年春季

截止日期：2023 年 3 月 23 日，星期四，晚上 11:59

一般说明

1. 您将单独完成此任务。
2. 应使用 Linux Virtual 开发和测试此作业的代码

您可以在以下位置创建的机器：

https://vcm.duke.edu/reservations/new/vm

选择下图：Ubuntu 20.04

不幸的是，由于复杂性，将不支持其他环境。您必须仔细遵守此作业规范并上交所有

3. 内容

问（并以正确的格式，如所述）。由于班级规模较大，因此需要这样做才能提高评分效率。

概述

在此作业中，您将构建一个 PostgreSQL 数据库。将提供数据库的关系模式（表）以及数据库值。您将实施：
1. 一个 C++ 程序，用于读取包含每个表的条目（行）的文本文件，并通过创建每个表和添加条目来构建数据库。
2. C++ 库（函数集），它将为程序提供与数据库交互的接口（例如，向表中添加行和查询某些信息）。

该数据库与 ACC 篮球队相关，并允许查询发现有关球员统计、球队属性等信息。该数据库将有 4 个表，指定如下（带下划线的属性表示表的主键）：

播放器（播放器 _ID , TEAM_ID, UNIFORM_NUM, FIRST_NAME, LAST_NAME, MPG, PPG, RPG, APG, SPG, BPG）
团队 （团队 _ID , NAME, STATE_ID, COLOR_ID, WINS, LOSSES) STATE
(STATE_ID , 姓名) 颜色 (颜色_ID ， 姓名）

* 请注意以下缩写：MPG = 每场上场时间，PPG = 每场得分，RPG = 每场篮板，APG = 每场助攻，SPG = 每场抢断，BGP = 每场盖帽

使用虚拟机的技巧

当您第一次创建虚拟机并登录时，您会注意到可能安装了很少的程序（例如没有 gcc、emacs、vim 等）。您可以使用以下命令轻松下载您选择的软件：sudo apt-get install <package name>。例如：

sudo apt install build-essential emacs

为了创建 PostgreSQL 数据库并通过 C++ API 与数据库交互，您需要安装软件包，可以通过 git 获取，如下所示：

sudo apt 安装 libpqxx-dev

你将实施什么

提供了几个框架文件和数据库表信息文件来帮助您入门。下面对每个文件进行说明：

● **数据库源文本文件：player.txt、team.txt、state.txt、color.txt**. 这些文件包含每个条目和应该插入到相应数据库表中的每个属性的类似表格的信息。

● **主.cpp**: 主要功能。在这里，您应该实现在每次执行程序时设置数据库的代码。具体来说，它应该删除（如果需要）每个表并将其添加到数据库（名为 ACC_BBALL），然后从源文本文件中读取信息并根据需要向每个表添加行。

● **query_funcs.h 和 query_funcs.cpp**：在这里您将实现与数据库交互的功能（添加新条目并打印下面指定的 5 个不同查询的结果）。

如果需要，您可以向这些文件添加新函数（您不必这样做），但不应更改现有函数的定义。

● **锻炼者.h 和锻炼者.cpp**: 在这里你可以在 exercise() 函数中添加代码来测试你的查询功能。exercise() 函数在数据库初始化后从 main() 调用。

● **生成文件**: 将所有源文件编译成名为“test”的可执行程序

您的任务是执行以下操作：
● 创建一个名为**ACC_BBALL 球**(全部大写）
● 为名为“postgres”的 ACC_BBALL 数据库创建一个用户，密码为“passw0rd”（如何设置postgres用户密码见ppt图）。
● 在 main.cpp 中实现支持以连接（作为用户“postgres”）到 ACC_BBALL 数据库并初始化其表和数据。当然，通常数据库会持久存在，但是为了评估提交的目的，程序应该每次都重新创建数据库。在程序中创建初始化表之前，您应该删除可能已经存在的表（例如，来自之前的测试运行）。

● 实现以下查询功能：
○ **查询1()**：显示每个球员的所有属性，平均统计数据介于每个已启用统计信息的最小值和最大值（含）
○ **查询2()**：用指定的统一颜色显示每个团队的名称
○ **查询3()**：显示为指定球队效力的每位球员的名字和姓氏，从最高到最低的 ppg（每场比赛得分）排序
○ **查询4()**：显示在指定状态下比赛并穿着指定球衣颜色的每位球员的球衣号码、名字和姓氏

○ **查询5()**：显示每个球员的名字和姓氏，以及每支赢得超过指定比赛次数的球队的球队名称和获胜次数
○ **重要的提示：**每个查询函数都应打印其输出。此输出的格式应如下所示。输出的第一行应包含每个字段名称（由一个空格字符分隔。下一行输出应包含从查询返回的值，每个值也由一个空格分隔。字段名称和每一行输出应出现一行接一行。

● 您可以通过在 exercise() 函数中添加对查询函数的调用来测试您的数据库和查询函数。出于对作业评分的目的，我们将用一个新文件替换 exercise.cpp 文件，该文件将测试各种查询调用。

示例输出和检查结果

以下文本显示示例输出。此输出对应于**查询1**调用它来显示所有玩家每场比赛的最小分钟数 (MPG) 为 35，每场比赛的最大分钟数 (MPG) 为 40。

PLAYER_ID TEAM_ID UNIFORM_NUM FIRST_NAME LAST_NAME MPG PPG RPG APG SPG BPG 153 12
3 Andrew WhiteIII 37 19 5 1 1.6 0.4
154 12 20 泰勒・莱登 36 13 9 2 1.0 1.4 30 3 5 卢克・肯纳
德 36 20 5 3 0.8 0.4 59 5 44 本・拉莫斯 35 14 9 2 1.2 3.4 87
7 5 达文・里德 35 15 5 2 J 1.3 0.5 r 4.5 丹尼斯 98 98 35 18
5 6 1.9 0.4 130 10 32 史蒂夫・瓦斯图里亚 35 13 4 3 1.2 0.1

Unix 命令“diff”可用于比较结果。例如，如果将上面的输出复制到名为 validation.txt 的文件中，则可以将其与程序中相同查询的输出进行比较。如果您的程序位于名为 output.txt 的文件中，则 diff 命令可能如下所示：

$ diff -w 输出.txt 验证.txt

请注意，-w 表示“diff”不应报告输出中空白（空格、制表符）的差异。如果要逐个字符地比较两个文件，可以省略 -w 标志。

SPG、BPG 的小数点后的精度应为1，例如如果值为0 或1，则输出为0.0 或1.0。

确保您的 SQL 输出标头应位于**相同的顺序和相同的名称**作为项目 4 文档的表属性定义，（例如 PLAYER_ID TEAM_ID UNIFORM_NUM FIRST_NAME LAST_NAME MPG PPG RPG APG SPG BPG）。由于我们这次使用的是autograder，如果没有打印出正确的header，即使得到正确的结果也会很麻烦。

详细提交说明

您的提交将包括以下文件： 1.

源代码文件：main.cpp、query_funcs.h、query_funcs.cpp、练习器.h、练习器.cpp

2. **生成文件–即使您没有对所提供的进行更改**
3. **数据库源文本文件**: player.txt, team.txt, state.txt, color.txt

你应该把上面的文件放在一个名为“**hw4_数据库**”并提交此目录的 zip 文件，名为“**hw4_database.zip**“到**等级范围**名为“Project 4 Database Programming”。

额外学分 (+10%)

通过在 C++ 中实现 SQL 事务，您可能会更深入地了解 SQL 代码并将其视为使用数据库的常用方式。然而，现代软件架构中数据库的典型用途是使用更高级别的框架（例如对象关系映射或 ORM），这样就无需担心事务细节。具体来说，对象关系映射 (ORM) 是一种允许您使用面向对象范例编写事务的技术。许多图书馆已经实施了这种技术。

例如，你们中的一些人可能使用过 Django 来开发 Web 应用程序。Django 包含一个默认的 ORM 层来与来自关系数据库管理系统（如 PostgreSQL）的数据进行交互。而且除了Django ORM之外，还有其他的ORM库，比如Hibernate、SQLAlchemy、Doctrine等，选择一个ORM库**不包括 Django**(因为您已经在 ECE 568 中完成了此操作）并实现了前面语句中显示的五个查询。将您的代码作为名为“ **proj4_extra_netid.zip**“ 到**名为“Project 4 Extra Credit”的 Gradescope**. 要对其评分，您需要向其中一位助教展示演示。有关此事的进一步公告将在截止日期后发布。