SQL CoP Problem Sheet 1 - Basic Query Statements

George Melrose: george_melrose@yahoo.com, https://github.com/georgemelrose

Introduction

These problems aim to test your basic SQL knowledge, steadily building up in complexity. The questions and solutions are common ones you will come across when querying different datasets. For the purposes of this series of problem sheets, a database of dummy Marathon results data has been generated. More information on the **Marathon** database is presented below.

The concepts tested in this sheet are covered by the LinkedIn learning course **SQL Server Fundamentals: Master Basic Query Techniques** - (https://www.linkedin.com/learning/sql-server-fundamentals-master-basic-query-techniques) .

Useful Prepatory Resources

In addition to this problem sheet, there are two useful resources you can draw upon to better understand these SQL concepts:

- **Two RMarkdown documents** - one to generate some dummy 'Universities' data (https://github.com/georgemelrose/SQL_Practice/blob/main/0_generating_databasestar_dummy_data.Rmd). This was copied from the excellent SQL learning resource databasestar (https://github.com/bbrumm/databasestar/tree/main/sample_databases/sample_db_university/sqlite).

  The other document is an RMD HTML I generated walking you through basic SQL concepts and how they can be applied to this databasestar dummy data (https://github.com/georgemelrose/SQL_Practice/blob/main/01_Basic_Query_Statements.html).

- **A video presentation** - a recording of a meeting in which I presented the **Basic Query Statements** HTML , explaining key SQL concepts - (https://universityofcambridgecloud.sharepoint.com/sites/AD_Progress/SitePages/Learning-SQL-in-a-New-Format.aspx). This video can be found on the aforementioned page under the **SQL and R** title. *Note that this can only be viewed by SQL" Community of Practice Members, email gam55@cam.ac.uk for access.*

Marathon Database

Firstly, the data to be put into the Marathon database was formulated from the following Python script - (https://github.com/georgemelrose/SQL_Practice/blob/main/Dummy_Marathon_Data/marathon_data_generation.ipynb).

The **marathon data generation** python script generates the following tables:

1. Runners - Randomly generate 1000 runners with names common in their locale/country, together with their birth date and sex.

2. Events - The 6 Major World marathons (Berlin, Boston, Chicago,London,New York City, Tokyo), with an event per year from 2012 to 2023.

3. Results - Gives results for runners in hh:mm:ss format, ensuring there aren't duplicate results for each runner per event. Prevents any results breaking either the male marathon world-record (2:00:35 Eliud Kipchoge 2023) or the female marathon world-record (2:11:53 Brigid Kosgei 2019). Also determines, with a True/False column, if a result is elite by the male standard (below 02:15:00) or the female standard (below 02:30:00).

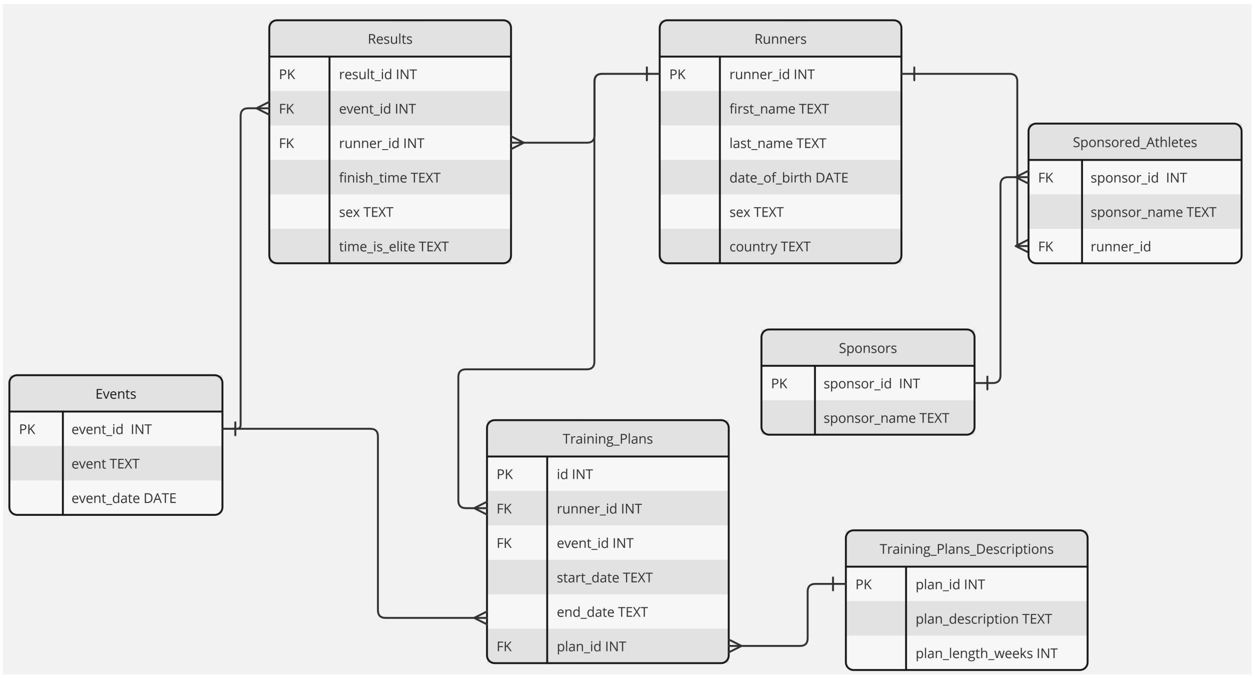
4. Sponsors - Lists the following 10 major companies that typically act as sponsors to runners - "Nike", "Adidas", "Asics", "Saucony", "Hoka","Brooks", "New Balance", "Puma", "Under Armour", "Tracksmith".

5. Sponsored Athletes - A table listing the fraction of the elite athletes that have a sponsor.

6. Training Plans Descriptions - The descriptions of 10 different training plans and their respective lengths in weeks.

7. Training Plans - The training plans of athletes. Only 72% of runner-event combinations have an associated training plan.

Marathon Database Entity Relationship Diagram



Basic Query Statment Problems

Single Table 'SELECT' Statements

Q1. - Get all the tables in the database, inspect their layout, and get the first 5 rows of each table?

Q2.i. - Obtain the full names and countries of the runners?

Q2.ii. - Find the top 10 Marathon results by time, male or female?

Filtering on Single Conditions

Q1 - Find the top 20 male marathon results and the top 20 female marathon results?

Q2.i. - Find all the runners from Lithuania, Latvia, and Estonia?

Q2.ii. - Find all the runners *not* from Poland, Czechia, Slovakia or Hungary?

Q3.i. - Find all the runners whose names begin with 'J'?

Q3.ii. - Find all the runners whose names don't begin with 'J'?

Q3.iii. - Find all the runners whose surnames contain 'son'?

Q4.i. - Gather all training plans that have a length between 10 and 12 weeks?

Q4.ii. - Gather all training plans that have a length less than 12 weeks?

Q4.iii. - Gather all training plans that have a length more than 12 weeks?

Q4.iv. - Gather all training plans that have a length more than or equal to 12 weeks?

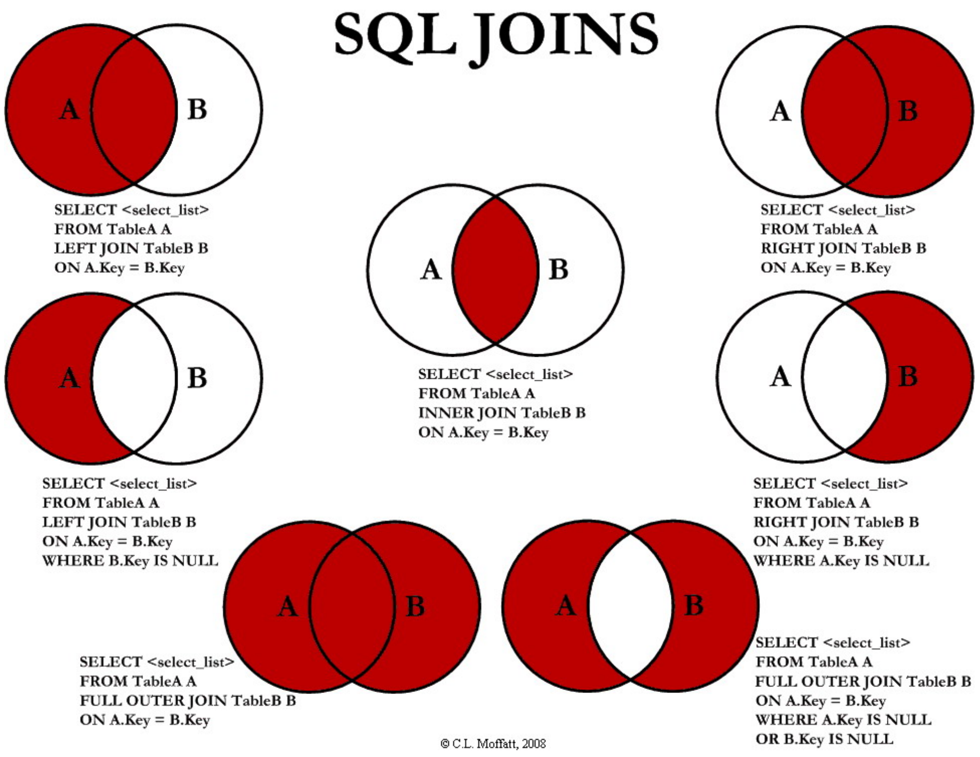
Filtering on Multiple Conditions

Q1.i. - Find all female runners from the United Kingdom born in the 20th century?

Q1.ii. - Find all male runners from Brazil born in 2000?

Q2. - Find the full names of all runners whose names begin with the letters G or J?

Single Inner Joins



Q1.i. - Obtain the runner ids of runners with an elite time in the London Marathon(any year)?

Q1.ii. - Obtain the number of runner ids of runners with an elite time in the London Marathon(any year)?.

Q1.iii. - Obtain the number of unique runner ids of runners with an elite time in the London Marathon(any year)?

Q2. - Find the the following information for male runners of any of the Boston Marathon events in the database: runner_id; finish_time; time_is_elite?

Multiple Inner Joins

Q1.i. - Find the full names of runners from the UK who had an elite time in the London Marathon (any year)?

Q1.ii. - Find the number of male and female runners respectively, from the UK, who had an elite time in the London Marathon (any year)?

Q2. - Find the full names, finish time and countries of sponsored runners from the UK who had an elite time in the Berlin Marathon (any year)?

Left Outer Joins

Q1. - Find the full names,sex, and country of runners sponsored by Nike?

Q2.i. - Find dates and times of results that had a finish time under 02:10:00 in Chicago?

Q2.ii. - Count the number of distinct results that had a finish time under 02:10:00 in Chicago?

Subqueries

Q1. - Using a subquery, find the full names, birth dates, and countries of runners who achieved a time below 02:20:00?

Q2. - Using a subquery, find all the runners by full name and sex who are sponsored by Nike or Tracksmith?

Q3. - Using a subquery, find all the male runners ids and their times, when their times were elite in the 2023 Boston Marathon?

Case When Statements

Q1. - Using a subquery and a case when statement, categorise of the 2023 London Marathon in the following way: >3 hours; 02:30:00 to 3hours; below 02:30:00 ? Provide the runner's full name, date_of_bith, sex, and country in the query result? Order from fatest to slowest?

Q2. - Utilising a case when statement, categorise all runners by which country their Sponsor is from (Adidas & Puma - Germany) (Asics - Japan) (Nike, Saucony, Hoka, Brooks, New Balance,Under Armour, Tracksmith), providing their full name, date of birth, sex, and country in the query result?

Q3. - Using a case when statement, generate the following categories for the different generations present amongst the runners: Silent Generation; Boomers; Gen X, Millennials; Gen Z; Gen Alpha? Include their full name, country, sex, and date of birth?

Aggregate Functions

Q1. - Count how many rows there are in each table, using an aggregate function?

Q2. - Count how many runners are present per country in the Runners table?

Q3. - Count how many runners obtained an elite time in any Berlin Marathon event?

Query Processing Order

A SQL query is processed in the following order (not necessarily written in this order) -

1. **FROM Clause -** The query starts by identifying the tables from which data will be retrieved. If there are any JOIN operations, they are also processed at this stage to create the base dataset.
2. **WHERE Clause -** After forming the initial dataset from the FROM clause, the WHERE clause is applied to filter rows based on specified conditions.
3. **GROUP BY Clause -** The data is then grouped according to the columns specified in the GROUP BY clause.
4. **HAVING Clause -** The HAVING clause is applied next, which allows filtering groups created by the GROUP BY clause based on aggregate conditions.
5. **SELECT Clause -** The SELECT clause is then processed to determine which columns or expressions should be included in the final result.
6. **ORDER BY Clause -** The ORDER BY clause sorts the result set based on the specified columns or expressions.
7. **LIMIT / OFFSET Clause -** If present, the LIMIT and OFFSET clauses are applied to restrict the number of rows returned by the query, and to skip a specified number of rows, respectively.

Q1. - Bearing the above in mind, generate a query that the top 10 fastest times of athletes at the 2016 London Marathon, with their full names, dates of birth, sexes, countries, sponsors, and training plans included in the query result?

Q2. - For the top 10 fastest times recorded in all of the database, obtain the runners' full names, respective times and training plans, sponsors if they have them?

Q3. - Retrieve the top 10 male runners who have participated in the highest number of events. For each runner, provide their full name, total number of events they participated in, and their sponsors if they have any?