Technical Report - Assignment 2

z5414201

2022-04-24

Contents

Task 1 **2**

`createTables()` 2

`dropTables()` 2

Questions 2

Task 2 **3**

`populateTables()`, `readData()`, `populateTable()`, `populateMedals()` 3

Task 3 **3**

1. The number of distinct events that have the sport 'Athletics' 3
2. The year, season, and city for each Olympics, ordered with the earliest entries first 4
3. The total number of each medal colour awarded to athletes from Australia (NOC: AUS) over all Olympics in the database 5
4. The name of all athletes from Ireland (NOC: IRL) who won silver medals, and the year / season in which they won them 5

Task 1

`createTables()`

The SQL statements are hardcoded into the program. When creating the tables, the database is first checked to ensure that a table doesn't currently exist with the same name. This is done by querying the `information_schema` table present in MySQL. If a table does exist, the software will proceed to the next table automatically. If not, the SQL `CREATE TABLE` statement is sent to the server. The software attempts to create the `Medals` table last due to its dependence on the others through `FOREIGN KEY` references. If an `SQLException` occurs, an error message is printed out and an error is thrown by the software.

An index of the columns is created for each table to help improve write performance of the database. This enables efficient performance of subqueries when populating the medals table.

`dropTables()`

`dropTables()` iterates through the four possible tables, first checking if they exist in the database through querying the information schema. If they do, a `DROP TABLE` statement is executed on the database to delete the table. The `Medals` table is attempted to be deleted first due to its dependence on the other tables. If an `SQLException` occurs, an error message is printed out and an error is thrown by the software.

Questions

For the Athletes table, would using name as the primary key be acceptable?

Not necessarily. Whilst the chance that two athletes have the same name is low, it is possible. Hence the name couldn't be used solely as the primary key because that would result in only one athlete with that name being able to be present in the database.

For the Medals table, is there an alternative (possibly composite) primary key that could be used rather than having an explicit row ID? If so, what benefits would using this composite key have?

Alternative option for primary key: `olympicID + eventID + athleteID`. The alternate cannot be `olympicID + eventID + medalColour` because in the event that two or more athletes were to formally draw, each athlete should be awarded a medal of the same colour. Using a composite key would ensure that duplicate medals are not added to the DB by accident. When using the ID PK, multiple rows may reference the same Olympic medal.

Why is 3NF desirable in a database?

In Third Normal Form (3NF), all transitive functional dependencies have been removed. This means that each column is only dependent on columns that are a part of the primary key. This results in redundancy within the tables being reduced. A reduction in redundancy is helpful because it makes it easier to change data, since it only appears in one place within the database. It also helps with search efficiency due to the inherent requirement of smaller tables when data is normalised. Finally, 3NF should also improve overall data quality since there is less of a chance unwanted data will be stored in the tables by requirement.

Task 2

`populateTables(),readData(),populateTable(),populateMedals()`

This is the main method responsible for populating the tables within the database. It makes use of SQL `PreparedStatements` to increase performance over continuous `INSERT` operations. `populateTables()` makes use of two helper methods to ensure efficient database operations. The `readData()` method is responsible for loading data in from the csv files. It parses the given file by-line, splitting the data into columns as a `String[]` and then passing it to a DB interface method to populate the tables. The `populateTable()` and `populateMedals()` methods are responsible for interfacing with the DB directly. They use the `PreparedStatement` provided by `populateTables()` and substitute the data provided by `readData()`. The statement is then added to a query batch that is sent as bulk to the server at the completion of the file read. This significantly increases the performance of the database since each query is not being sent, executed and responded to individually. The connection is also created with client side batch statement rewriting, which optimizes and accelerates batch SQL statements - further increasing performance. The tables are all populated individually, with Medals being populated last due to dependence on the other tables.

The time to populate all tables was approximately 14 seconds.

Task 3

1. The number of distinct events that have the sport ’Athletics’

```sql
SELECT DISTINCT COUNT(*)
FROM Events
WHERE sport='Athletics';
```

```
83
```

2. The year, season, and city for each Olympics, ordered with the earliest entries first

```sql
SELECT year, season, city
FROM Olympics
ORDER BY year;
```

```
1896 - Summer - Athina
1900 - Summer - Paris
1904 - Summer - St. Louis
1906 - Summer - Athina
1908 - Summer - London
1912 - Summer - Stockholm
1920 - Summer - Antwerpen
1924 - Summer - Paris
1924 - Winter - Chamonix
1928 - Summer - Amsterdam
1928 - Winter - Sankt Moritz
1932 - Summer - Los Angeles
1932 - Winter - Lake Placid
1936 - Summer - Berlin
1936 - Winter - Garmisch-Partenkirchen
1948 - Summer - London
1948 - Winter - Sankt Moritz
1952 - Summer - Helsinki
1952 - Winter - Oslo
1956 - Summer - Melbourne
1956 - Summer - Stockholm
1956 - Winter - Cortina dAmpezzo
1960 - Summer - Roma
1960 - Winter - Squaw Valley
1964 - Summer - Tokyo
1964 - Winter - Innsbruck
1968 - Summer - Mexico City
1968 - Winter - Grenoble
1972 - Summer - Munich
1972 - Winter - Sapporo
1976 - Summer - Montreal
1976 - Winter - Innsbruck
1980 - Summer - Moskva
1980 - Winter - Lake Placid
1984 - Summer - Los Angeles
1984 - Winter - Sarajevo
1988 - Summer - Seoul
1988 - Winter - Calgary
1992 - Summer - Barcelona
1992 - Winter - Albertville
1994 - Winter - Lillehammer
1996 - Summer - Atlanta
1998 - Winter - Nagano
2000 - Summer - Sydney
2002 - Winter - Salt Lake City
2004 - Summer - Athina
2006 - Winter - Torino
2008 - Summer - Beijing
2010 - Winter - Vancouver
2012 - Summer - London
2014 - Winter - Sochi
2016 - Summer - Rio de Janeiro
```

3. The total number of each medal colour awarded to athletes from Australia (NOC: AUS) over all Olympics in the database

```
SELECT COUNT(Medals.ID)
FROM Medals
INNER JOIN Athletes ON Medals.athleteID=Athletes.ID
WHERE Athletes.noc='AUS' AND Medals.medalColour='Gold';

SELECT COUNT(Medals.ID)
FROM Medals
INNER JOIN Athletes ON Medals.athleteID=Athletes.ID
WHERE Athletes.noc='AUS' AND Medals.medalColour='Silver';

SELECT COUNT(Medals.ID)
FROM Medals
INNER JOIN Athletes ON Medals.athleteID=Athletes.ID
WHERE Athletes.noc='AUS' AND Medals.medalColour='Bronze';
```

```
Gold: 348
Silver: 455
Bronze: 517
```

4. The name of all athletes from Ireland (NOC: IRL) who won silver medals, and the year / season in which they won them

```
SELECT Athletes.name, Olympics.year, Olympics.season
FROM Athletes
INNER JOIN Medals ON Athletes.ID = Medals.athleteID
INNER JOIN Olympics ON Medals.olympicID = Olympics.ID WHERE Athletes.noc='IRL' AND Medals.medalColour='Silver';
```

```
Jack Butler Yeats - 1924 - Summer
John McNally - 1952 - Summer
Frederick 'Fred' Tiedt - 1956 - Summer
David Robert Wilkins - 1980 - Summer
James 'Jamie' Wilkinson - 1980 - Summer
John Treacy - 1984 - Summer
Wayne William McCullough - 1992 - Summer
Sonia O'Sullivan - 2000 - Summer
Kenneth 'Kenny' Egan - 2008 - Summer
John Joseph 'Joe' Nevin - 2012 - Summer
Annalise Murphy - 2016 - Summer
Gary O'Donovan - 2016 - Summer
Paul O'Donovan - 2016 - Summer
```