MySQL Sakila movies database exercise

Bridget Bertoni

(Dated: April 27, 2016)

Problem 1

The dataset for this problem was generated using the following query:

```
select ’inventory_id’, ’film_id’, ’store_id’, ’title’, ’rental_date’, ’return_date’
union all
select inventory.inventory_id, inventory.film_id, inventory.store_id, film.title,
rental.rental_date, rental.return_date
into outfile ’/tmp/mysql_movie_dates_3.csv’
fields terminated by ’,’
enclosed by ’“’
lines terminated by ’\n’
from inventory
inner join rental on rental.inventory_id = inventory.inventory_id
inner join film on film.film_id = inventory.film_id
where rental.return_date is not null;
```

It took me about 2 hours to download MySQL, learn about MySQL and the Sakila database, and generate the proper dataset for this problem. The last line of the query is important since I need to calculate rental time intervals.

1 (a)

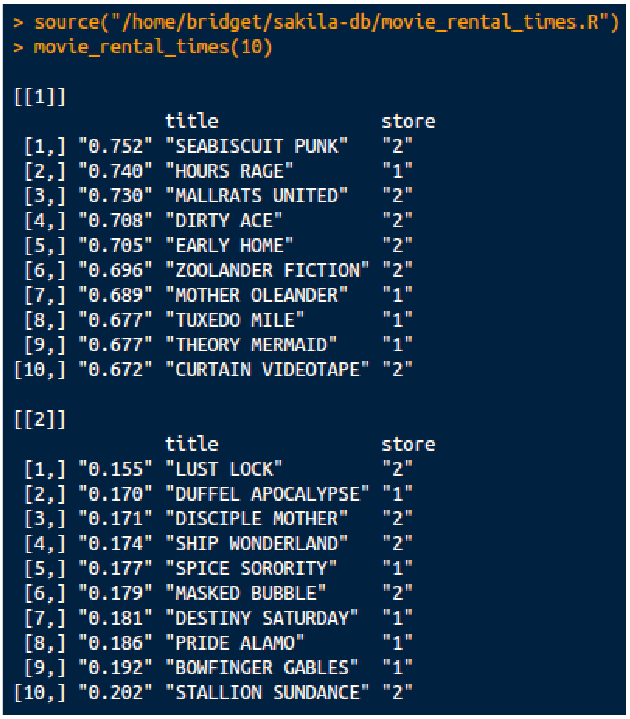
The code I wrote for this problem is contained in the file *movie_rental_times.R*. It is well-commented! Basically, I wrote a function called *movie_rental_times(N)* which takes in an integer N and outputs the movies with the N highest and lowest values of the ratio between the time a given movie at a given store was actually rented and the total time that it could have been rented. This total time is calculated by taking the difference between the latest time the movie was returned and the earliest time the movie was rented.

Sample output for $N = 10$ is shown below in Fig. 1.

It took me about 3 hours to write and debug this code.

1 (b)

The code I wrote for this problem is contained in the file *movie_days.R*. It is also well-commented! This code contains a function *movie_days(N)* which has an integer argument N. The code first calculates the maximum rental time period by taking the difference between the largest return time and the earliest rental time. This time period is then divided into



```
> source("/home/bridget/sakila-db/movie_rental_times.R")
> movie_rental_times(10)

[[1]]
              title               store
 [1,] "0.752" "SEABISCUIT PUNK"    "2"
 [2,] "0.740" "HOURS RAGE"         "1"
 [3,] "0.730" "MALLRATS UNITED"    "2"
 [4,] "0.708" "DIRTY ACE"          "2"
 [5,] "0.705" "EARLY HOME"         "2"
 [6,] "0.696" "ZOOLANDER FICTION"  "2"
 [7,] "0.689" "MOTHER OLEANDER"    "1"
 [8,] "0.677" "TUXEDO MILE"        "1"
 [9,] "0.677" "THEORY MERMAID"     "1"
[10,] "0.672" "CURTAIN VIDEOTAPE"  "2"

[[2]]
              title               store
 [1,] "0.155" "LUST LOCK"          "2"
 [2,] "0.170" "DUFFEL APOCALYPSE"  "1"
 [3,] "0.171" "DISCIPLE MOTHER"    "2"
 [4,] "0.174" "SHIP WONDERLAND"    "2"
 [5,] "0.177" "SPICE SORORITY"     "1"
 [6,] "0.179" "MASKED BUBBLE"      "2"
 [7,] "0.181" "DESTINY SATURDAY"   "1"
 [8,] "0.186" "PRIDE ALAMO"        "1"
 [9,] "0.192" "BOWFINGER GABLES"   "1"
[10,] "0.202" "STALLION SUNDANCE"  "2"
```

FIG. 1: $N = 10$ output of the *movie_rental_times.R* code. The output is a two component list. The first element of the list is a matrix which shows the value of the ratio, the movie title, and the store id for the 10 largest ratio values. The rows are ordered by decreasing ratio. The second element of the list is a matrix which shows the value of the ratio, the movie title, and the store id for the 10 smallest ratio values. The rows are ordered by increasing ratio.

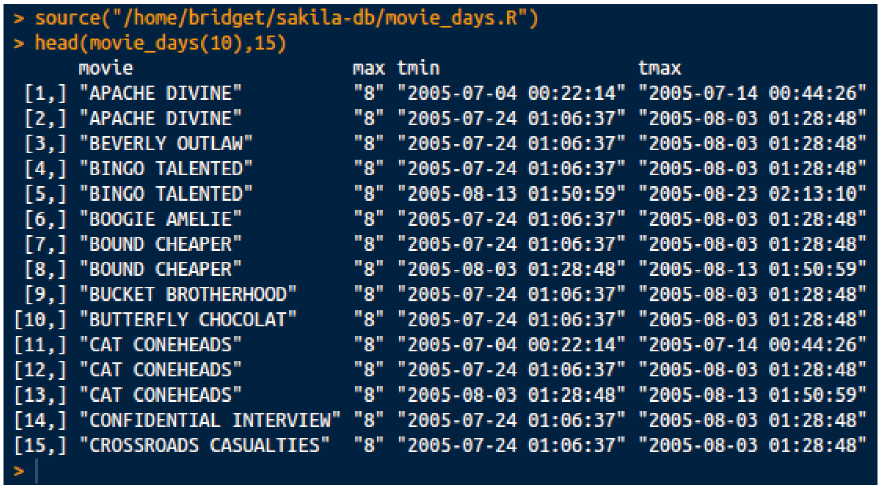
N intervals and the code calculates how many copies of a given movie were checked out during each time interval from any movie rental store. The code outputs a matrix of data concerning only the movies that were rented for the maximum number of times during the time intervals. It outputs a matrix with columns: movie title, the maximum number of rental times in a given time interval, the beginning time of the time interval ($tmin$), and the ending time of the time interval ($tmax$).

Note that the matrix *counts* contains the number of times a give movie was rented within each time interval, for all movies, so in principle, more data could be extracted from *movie_days(N)* besides the maximum count values which I chose to output.

Some of the output from this code for the case $N = 10$ is shown in Fig. 2.

I did not finish this part—I did not answer the question "Of the movie-store-days in which this is the case, how often is it that the same movie is available at a different store on the same day?" and I also would have liked to spend more time debugging it.

It took me about 2 hours to write and debug this code.



```
> source("/home/bridget/sakila-db/movie_days.R")
> head(movie_days(10),15)
      movie                    max tmin                  tmax
 [1,] "APACHE DIVINE"          "8" "2005-07-04 00:22:14" "2005-07-14 00:44:26"
 [2,] "APACHE DIVINE"          "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
 [3,] "BEVERLY OUTLAW"         "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
 [4,] "BINGO TALENTED"         "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
 [5,] "BINGO TALENTED"         "8" "2005-08-13 01:50:59" "2005-08-23 02:13:10"
 [6,] "BOOGIE AMELIE"          "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
 [7,] "BOUND CHEAPER"          "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
 [8,] "BOUND CHEAPER"          "8" "2005-08-03 01:28:48" "2005-08-13 01:50:59"
 [9,] "BUCKET BROTHERHOOD"     "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
[10,] "BUTTERFLY CHOCOLAT"     "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
[11,] "CAT CONEHEADS"          "8" "2005-07-04 00:22:14" "2005-07-14 00:44:26"
[12,] "CAT CONEHEADS"          "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
[13,] "CAT CONEHEADS"          "8" "2005-08-03 01:28:48" "2005-08-13 01:50:59"
[14,] "CONFIDENTIAL INTERVIEW" "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
[15,] "CROSSROADS CASUALTIES"  "8" "2005-07-24 01:06:37" "2005-08-03 01:28:48"
> 
```

FIG. 2: Top 15 entries of the $N = 10$ output of *movie_days.R*. The total rental time in this case was divided into 10 intervals, making each interval about 10 days long. In these 10 day time periods, the maximum number of times any movie was rented was 8 times. The output lists each movie that was rented for the maximum of 8 times in any time period, and also lists the time period during which the maximum was achieved.

Problem 2

2 (a)

2 (b)

2 (c)