Question 1 [24 points]: Data Loading, Cleaning, Indexing and Setting Constraints in PostgreSQL

In this problem set, we consider a movie database containing information of over 44 thousand movies and over 230 thousand actors and directors (inspired by IMDB).

Schema. This database consists of four tables: (1) a people table, containing actors and directors, (2) the movies table, containing information about the movies, (3) the acted_in table, describing which actors acted in which movies, (4) the directed table, which indicates which directors directed which movies. The schema is the following, where the underscore indicates the primary key of a relation:

```sql
people (pid: INTEGER, fname: VARCHAR(64) NOT NULL, lname: VARCHAR(64), gender: ENUM('male','female'))
movies (mid: INTEGER, name: VARCHAR(255) NOT NULL, year: INTEGER NOT NULL, mpaa_rating: ENUM('G','PG','PG-13','R','NC-17'), ave_user_ratings: FLOAT NOT NULL, count_user_ratings: INTEGER NOT NULL)
acted_in (pid: INTEGER, mid: INTEGER)
directed (pid: INTEGER, mid: INTEGER)
```

We provide four raw text files to be loaded to the database named by your NetID. Please proceed with the following steps.

(1) Write the commands, “CREATE TABLE” for the four datasets.

(2) See the dataset under “/courses/cs600/cs645/cs645/edmovie_txt” (Fields are delimited by tab ‘\t’)

```
> ls /courses/cs600/cs645/cs645/edmovie_txt
people.txt movies.txt acted_in.txt directed.txt
```

Hint: the dataset directory is READ-ONLY to you, please use the absolute path in your data loading commands

(3) Load the datasets into the tables.

(4) Data cleaning

However, these datasets are not clean due to the following reasons:

- The “movies” dataset contains duplicates of “name” and “year”. Therefore, the combination of “name” and “year” are not unique, but that is our goal to make (name, year) UNIQUE in the data cleaning process.
- The “movies” dataset also contains NULLs in the “name” of some movies. We would like to make “name” NOT NULL, by removing those movies with name = NULL in the data cleaning process.
- The “people” dataset contains duplicates. Therefore, the combination of “fname” and “lname” cannot be declared as unique, but that is our goal to achieve (fname, lname) UNIQUE in the data cleaning process.
- The “people” dataset also contains NULLs in the “fname” of some tuples. We would like to make “fname” NOT NULL by removing those people with fname = NULL.
Note:
- You can perform data cleaning any way you want. For instance, you can write a Python script, use SQL queries over the dirty data, etc.

(5) UNIQUE constraints
After cleaning, we want to add the unique constraints to the combination of “name” and “year” of the movies relation, and the combination of “fname” and “lname” of people.

Hint:
- You can add a unique constraint in PostgreSQL after table creation as follows:
  `ALTER TABLE <table-name> ADD CONSTRAINT <constraint-name> UNIQUE(<attribute-list>).`
- If you do data cleaning before loading the data into PostgreSQL, you can directly declare UNIQUE in your schema.

(6) Foreign key constraints (and more data cleaning)
In addition, we would like to have the following constraints that
a) the pid of the acted_in table is a foreign key referring to the people table.
   b) the mid of the acted_in table is a foreign key referring to the movies table.
   c) the pid of the directed table is a foreign key referring to the people table.
   d) the mid of the directed table is a foreign key referring to the movies table.

(7) Indexing
Your database is allowed and only allowed to have the following indexes:
- All primary key indexes, which are automatically created on relations with primary keys declared.
- A B+ tree on the (name, year) attributes of the movies table.
- A B+ tree on the (fname, lname) attributes of the people table, and

Other useful commands in postgresql:
```
postgresql: \d
postgresql: \di
postgresql: \d table
postgresql: \d+ table
postgresql: ANALYZE table
```

Linux commands:
Sometimes your commands may lead to long running queries (processes). To check or kill a background process, you can use the following Linux commands:
   a) you can use command "ps" to see your process.
   b) use command "kill" to kill your processes after you locate the process you want to kill.

Now please answer the following questions:

(a) [12 Points] Table sizes of people, movies, and acted_in

Before data cleaning, how many tuples are there in the people table? _______________________

After data cleaning, how many tuples are there in the people table? _______________________

Before data cleaning, how many tuples are there in the movies table? _______________________

After data cleaning, how many tuples are there in the movies table? _______________________

Before data cleaning, how many tuples are there in the acted_in table? _______________________

After data cleaning, how many tuples are there in the acted_in table? _______________________

(b) [8 Points] Indexes and constraints for movies and acted_in

Show the content after typing the PostgreSQL “\d movies” command.

Show the content after typing the PostgreSQL “\d acted_in” command.

(c) [4 Points] Indexes created for the database

Show the content after typing the PostgreSQL “\di” command.

Question 2 [16 points]: SQL Queries on the Movie Dataset

(a) [4 points] Find the actors who have acted in the largest numbers of movies. Please list the top-5 such actors, and return the pid of each actor, and the number of movies they acted in.

(b) [4 points] Find the movie with the largest cast. For such a movie, return the movie id and the size of the cast.

(c) [4 points] Find the first 5 actors, in order of first name followed by last name, who acted only in films before 1985. Make sure to exclude movies whose year is unknown (indicated by year=0). Show only the pid, the first name and the last name of these actors.

(d) [4 points] Among the movies with more female actors than male actors, find the top-5 movies with the largest numbers of female actors. Return the movie id and the number of female actors for each of these movies.
Question 3 [18 points]: Functional Dependencies and Normalization

(1) [6 Points] Suppose that we have the following three tuples in a legal instance of a relation schema S with three attributes ABC:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Which of the following dependencies does not hold over schema S?
(a) A \(\rightarrow\) B
(b) BC \(\rightarrow\) A
(c) B \(\rightarrow\) C

(2) [4 Points] Assume that you are given a relation R with attributes ABC. Assume that no record can have NULL values. Write an SQL query that checks whether the functional dependency A \(\rightarrow\) B holds, and if not, returns the number of pairs of tuples that violate the functional dependency.

(3) [8 Points] Consider a relation R(A,B,C,D) and the functional dependencies that hold over R:

- AB \(\rightarrow\) C
- AB \(\rightarrow\) D
- C \(\rightarrow\) A
- D \(\rightarrow\) B

Propose a lossless-join decomposition of R into a set of smaller relations in BCNF. Please also state whether this decomposition is dependency-preserving or not.

Question 4 [22 points] Disk Storage

(1) [8 points] Consider a disk with a sector size of 512 bytes, 63 sectors per track, 16,383 tracks per surface, 10 double-sided platters (i.e., 20 surfaces). The disk platters rotate at 7,200 rpm (revolutions per minute). The average seek time is 9 msec, whereas the track-to-track seek time is 1 msec (use these numbers in appropriate places in your calculation).

Suppose that a page size of 4096 bytes is chosen, and a page can span sectors on difference tracks. Suppose that a file containing 1,000,000 records of 256 bytes each is to be stored on such a disk. No record is allowed to span two pages.

(a) [2 Points] What is the capacity of a track (in number of bytes)?
(b) [2 Points] What is the capacity of the disk (in number of bytes)?
(c) [1 Points] How many records fit in a page?
(d) [3 Points] How many records fit in a cylinder?

To answer each question, please write a clean formula and your final answer.
(2) [14 points] The figure below shows a page containing variable length records. The page size is 1KB (1024 bytes). It contains 3 records, some free space, and a slot directory in that order. Each record has its record id, in the form of Rid=(page id, slot number), as well as its start and end addresses in the page, as shown in the figure.

(a) [7 Points] Now a new record of size 200 bytes needs to be inserted into this page. Apply the record insertion algorithm (with page compaction, if necessary) that we learned in class to this page. Show the content of the slot directory after the new record is inserted.

(b) [7 Points] The next question proceeds after the operation in Part (1). Now, the record with Rid=(15,3) needs to be deleted. Afterwards, another record of size 300 bytes needs to be inserted. Show the content of the slot directory after the deletion and new insertion.

Question 5 [20 points]: B+ Trees

(1) [10 points] Show the results of entering the keys 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 (in that order) to an initially empty B+ tree. Assume that every non-leaf node can hold up to 3 index entries and every leaf node can hold up to 3 data entries. In case of overflow, split the node (do not re-distribute keys to neighbors).

(2) [10 points] Now demonstrate a different insertion order that leads to a tree of different depth than the one in Part (1).