COMS W4111: Introduction to Databases
Spring 2023, Sections 002, V02

Non-Programming Track, HW2, Part 2

Introduction

Environment

- Test environment.
- Set your MySQL user and password below.

In [1]:
```
mysql_user = "root"
mysql_pw = "dbuserdbuser"
```

In [2]:
```
%load_ext sql
```

In [3]:
```
full_url = f"mysql+pymysql://{mysql_user}:{mysql_pw}@localhost"
full_url
```

Out[3]:
```
'mysql+pymysql://root:dbuserdbuser@localhost'
```

In [4]:
```
%sql $full_url
```

In [5]:
```
%sql select * from db_book.student;
```

```
 * mysql+pymysql://root:***@localhost
13 rows affected.
```

Out[5]:

ID	name	dept_name	tot_cred
00128	Zhang	Comp. Sci.	102
12345	Shankar	Comp. Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp. Sci.	54

55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp. Sci.	58
76653	Aoi	Elec. Eng.	60
98765	Bourikas	Elec. Eng.	98
98988	Tanaka	Biology	120

Submission Instructions

- See Ed for instructions.

Data and Scheme Cleanup

characters and name_basics_all

- The task is to "clean up" `characters` and produce a table `charactersFixed`.
- The task will require adding missing rows to `name_basics_all`. There are two row's in `characters` that have an `actorLink` and `actorName` got which there is no matching row in `name_basics_all`.
- `characters` has two actors with actorNames `Barry John O'Connor` and `Barry O'Connor` who are the same actor.
- My `charactersFixed` has the following columns:
  - `characterId` is a generated primary key. See below for an explanation.
  - `characterName`: The value from `characters`.
  - `characterImdbID`: The `characterLink` from `characters` with `/character/` removed.
  - `characterLink`: The `characterLink` from `characters`.
  - `actorNConst`: `actorLink` from `characters`.
  - `actorLink`: A value of the form `/names/` followed by the `actorNConst`.
  - `characterImageFull`: The value from `characters`.
  - `characterImageThumb`: The value from `characters`.
  - `kingsguard`: The value from `characters`.
  - `royal`: The value from `characters`.
- The algorithm for generating the `characterID` on insert is the following:
  - The prefix for the `character` is either:
    - The substring of `characterName` preceeding the first `' '`.
    - The `characterName` is there is no `' '`.
  - If there are `N` rows in the table, the number after the prefix is `N+1`.
  - Implementing this is tricky. Your first attempt might rely on `auto-increment`, but this does not work. You may also be tempted to count rows, but that does not work. A hint is that you will need to use a trigger and some other table/data that you create.
- The directory with this notebook containers data from my version of `charactersFixed`.
- The cells below load the data to allow you to examine. In your SQL table, `NaN` will be `NULL`.

In [6]:
```
import pandas as pd
```

In [7]:
```
characters_df = pd.read_csv('./charactersFixed.csv')
```

In [8]:
```
characters_df
```

Out[8]:

	characterId	characterName	characterImdbID	characterLink	actorNconst	actorLink	characterImageFull	cha
0	Addam1	Addam Marbrand	ch0305333	/character/ch0305333	nm0389698	/names/nm0389698	NaN	
1	Aegon2	Aegon Targaryen	NaN	NaN	NaN	NaN	NaN	

2	Aeron3	Aeron Greyjoy	ch0540081	/character/ch0540081	nm0269923	/names/nm0269923	https://images-na.ssl-images-amazon.com/images...	ht an
3	Aerys4	Aerys II Targaryen	ch0541362	/character/ch0541362	nm0727778	/names/nm0727778	https://images-na.ssl-images-amazon.com/images...	ht an
4	Akho5	Akho	ch0544520	/character/ch0544520	nm6729880	/names/nm6729880	https://images-na.ssl-images-amazon.com/images...	ht an
...	...	...	...	...	...	...	...	
384	Young385	Young Nan	ch0305018	/character/ch0305018	nm1519719	/names/nm1519719	NaN	
385	Young386	Young Ned	ch0154681	/character/ch0154681	nm7075019	/names/nm7075019	NaN	
386	Young387	Young Ned Stark	ch0154681	/character/ch0154681	nm7509185	/names/nm7509185	NaN	
387	Young388	Young Rodrik Cassel	ch0171391	/character/ch0171391	nm7509186	/names/nm7509186	NaN	
388	Zanrush389	Zanrush	ch0540870	/character/ch0540870	nm0503319	/names/nm0503319	https://images-na.ssl-images-amazon.com/images...	ht an

389 rows × 10 columns

- Your answer below should show all of your SQL statements, including DDL, for creating and loading `charactersFixed` as well as changes to `name_basics_all` .
- You can use the data in the CSV file to test your work. Show at least one test.

Rename the name of same actor

In [15]:
```
%%sql
use s23_w4111_hw2_jz3543;

SELECT actorName, actorLink FROM characters
WHERE actorName= "Barry John O'Connor"
union all
SELECT actorName, actorLink FROM characters
WHERE actorName= "Barry O'Connor"
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
2 rows affected.
```

Out[15]:

actorName	actorLink
Barry John O'Connor	nm3226454
Barry O'Connor	None

In [19]:
```
%%sql

UPDATE characters c
JOIN (SELECT actorName, actorLink FROM characters
      WHERE actorName="Barry John O'Connor") T
ON c.actorName = "Barry O'Connor"
SET c.actorLink = T.actorLink;
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[19]: []

In [20]:
```
%%sql

SELECT actorName, actorLink FROM characters
WHERE actorName= "Barry O'Connor"
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[20]:

actorName	actorLink
Barry O'Connor	nm3226454

Create a Function to create characterID

In [21]:
```
%%sql

use s23_w4111_hw2_jz3543;

drop function if exists compute_charID;
```

```
set @c:=0;

create function compute_charID(characterName varchar(64)) returns varchar(64)
    reads sql data
BEGIN

    declare name char(64);
    declare num int;
    declare result varchar(64);

    set num = (@c:=@c+1);
    set name = substring_index(characterName,' ',1);
    set result = concat(name,num);

    return result;

end;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[21]: []

In [22]:
```
%%sql
select max(length(characterImageFull)) as max_cif,
max(length(characterImageThumb)) as max_cit from characters
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[22]:

max_cif	max_cit
164	167

In [23]:
```
%%sql
drop table if exists charactersFixed;
create table charactersFixed(characterID varchar(64), characterName varchar(64),
                             characterImdbID varchar(64), characterLink varchar(64),
                             actorNconst varchar(64), actorLink varchar(64),
                             characterImageFull varchar(200), characterImageThumb varchar(200),
                             kingsguard varchar(5), royal varchar(5))
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
```

Out[23]: []

Create a trigger to insert characterID

In [24]:
```
%%sql

drop trigger if exists compute_characterID;

#set @c:=0;

create trigger compute_characterID
    before insert
    on s23_w4111_hw2_jz3543.charactersFixed
    for each row
begin
    set new.characterID = compute_charID(new.characterName);
    #set new.characterID = concat(substring_index(new.characterName,' ',1),(@c:=@c+1));
end;
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
0 rows affected.
```

Out[24]: []

In [25]:
```
%%sql
insert into charactersFixed (characterName, characterImdbID,
                             characterLink, actorNconst, actorLink,
                             characterImageFull, characterImageThumb,
                             kingsguard, royal)
select characterName, substr(characterLink,12,9) as characterImdbID,
       characterLink, actorLink as actorNConst,
       concat('/names/',actorLink) as actorLink,
       characterImageFull, characterImageThumb, kingsguard, royal from characters;
```

```
 * mysql+pymysql://root:***@localhost
389 rows affected.
```

Out[25]: []

In [26]:
```
%%sql

ALTER TABLE charactersFixed add primary key(characterID)
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[26]: []

Not use trigger to create charactersFixed

In [16]:
```
'''
%%sql
use s23_w4111_hw2_jz3543;

alter table characters add column characters_num varchar(64) first;
set @c:=0;
Update characters set characters_num = (@c:=@c+1);
'''
```

Out[16]:
```
'\n%%sql\nuse s23_w4111_hw2_jz3543;\n\nalter table characters add column characters_num varchar
(64) first;\nset @c:=0;\nUpdate characters set characters_num = (@c:=@c+1);\n'
```

In [17]:
```
'''
%%sql

drop table if exists charactersFixed;
CREATE TABLE charactersFixed AS
    SELECT
    concat(substring_index(characterName,' ',1),characters_num) as characterID,
    characterName, substr(characterLink,12,9) as characterImdbID, characterLink,
    actorLink as actorNConst, concat('/names/',actorLink) as actorLink,
    characterImageFull, characterImageThumb, kingsguard, royal
    from characters
'''
```

Out[17]:
```
"\n%%sql\n \ndrop table if exists charactersFixed;\nCREATE TABLE charactersFixed AS\n    SELECT
\n    concat(substring_index(characterName,' ',1),characters_num) as characterID, \n    charact
erName, substr(characterLink,12,9) as characterImdbID, characterLink, \n    actorLink as actorN
Const, concat('/names/',actorLink) as actorLink, \n    characterImageFull, characterImageThumb,
```

```
kingsguard, royal \n    from characters\n"
```

Update name_baiscs_all

In [28]:
```
%%sql

# the actors in charactersFixed but not in name_basics_all
select nconst from name_basics_all
where nconst not in
(select actorLink as nconst from characters where actorLink is not NULL)
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[28]:

nconst

In [27]:
```
%%sql

# the actors in name_basics_all but not in charactersFixed
select actorLink, actorName from characters
where actorLink not in (select nconst from name_basics_all)
and characterLink is not null
```

```
 * mysql+pymysql://root:***@localhost
2 rows affected.
```

Out[27]:

actorLink	actorName
nm3226454	Barry John O'Connor
nm8199963	Michael Patrick

In [29]:
```
%%sql

insert into name_basics_all(nconst, primaryName)
(select actorLink, actorName from characters
where actorLink not in(
 select nconst from name_basics_all)
 and characterLink is not null);
```

```
 * mysql+pymysql://root:***@localhost
2 rows affected.
```

Out[29]:
```
[]
```

In [30]:
```
%%sql

# check
select actorNConst from charactersFixed
where actorNConst not in (select nconst from name_basics_all)
and characterLink is not null
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[30]:

actorNConst

Test SQL Table with Python Dataframe

In [31]:
```
characters_df[characters_df.characterId=='Addam1']
```

Out[31]:

	characterId	characterName	characterImdbID	characterLink	actorNconst	actorLink	characterImageFull	charact
0	Addam1	Addam Marbrand	ch0305333	/character/ch0305333	nm0389698	/names/nm0389698	NaN	

In [32]:
```
%%sql

select * from charactersFixed where characterID = 'Addam1'
```

```
 * mysql+pymysql://root:***@localhost
1 rows affected.
```

Out[32]:

characterID	characterName	characterImdbID	characterLink	actorNconst	actorLink	characterImageFull	characterI
Addam1	Addam Marbrand	ch0305333	/character/ch0305333/	nm0389698	/names/nm0389698	None	

In [33]:
```
%%sql

select * from charactersFixed limit 5;
```

```
 * mysql+pymysql://root:***@localhost
5 rows affected.
```

Out[33]:

characterID	characterName	characterImdbID	characterLink	actorNconst	actorLink
Addam1	Addam Marbrand	ch0305333	/character/ch0305333/	nm0389698	/names/nm0389698

Aegon2	Aegon Targaryen	None	None	None	None	
Aeron3	Aeron Greyjoy	ch0540081	/character/ch0540081/	nm0269923	/names/nm0269923	amazon.cc
Aerys4	Aerys II Targaryen	ch0541362	/character/ch0541362/	nm0727778	/names/nm0727778	amazon.com/images/M/MV5BMW
Akho5	Akho	ch0544520	/character/ch0544520/	nm6729880	/names/nm6729880	amazon.com/i

name_basics_all

- The column `primaryProfessions` is multi-valued and non-atomic. This violates good relational design principle.
- Create a new table `name_basics_all_fixed` which does not have the column `primaryProfessions`.
- You will need to use SQL to create and load other tables with information from `name_basics_all` to enable you to create a view `name_basics_all_fixed_view` that recreates the data in `name_basics_all`. The tables you create should have atomic columns, primary keys and foreign keys, etc.

In [34]:
```
%%sql
describe name_basics_all
```

```
 * mysql+pymysql://root:***@localhost
6 rows affected.
```

Out[34]:

Field	Type	Null	Key	Default	Extra
nconst	varchar(16)	NO		None	
primaryName	text	YES		None	
birthYear	text	YES		None	
deathYear	text	YES		None	
primaryProfession	text	YES		None	
knownForTitles	text	YES		None	

Create name_basics_all_fixed

In [35]:
```
%%sql

use s23_w4111_hw2_jz3543;

drop table if exists name_basics_all_fixed;

CREATE TABLE name_basics_all_fixed AS
```

```
    SELECT
    nconst, primaryName, birthYear, deathYear
    from name_basics_all
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
350 rows affected.
```

Out[35]: []

In [53]:
```
%%sql
ALTER TABLE name_basics_all_fixed
MODIFY COLUMN birthYear int DEFAULT NULL;
ALTER TABLE name_basics_all_fixed
MODIFY COLUMN deathYear int DEFAULT NULL;
```

```
 * mysql+pymysql://root:***@localhost
350 rows affected.
350 rows affected.
```

Out[53]: []

Methods

For this question, since primaryProfession and knownForTitles are non-atomic columns, I need to seperate their values into atomic columns, stpes:

- build a table reflect the relationship between nconst and profession/title
- build a table to record all uniques values of profession/titles
- set PK and FK to those new created tables
- join tables according to their relationshp to create the final view

Atomic primaryProfession

In [36]:
```
%%sql

drop table if exists name_profession;

CREATE TABLE name_profession AS
```

```
    select nconst,
        substring_index(substring_index(primaryProfession,',',1),',',-1) as Profession
    from name_basics_all
    where substring_index(substring_index(primaryProfession,',',1),',',-1) is not null
    Union
    select nconst,
        substring_index(substring_index(primaryProfession,',',2),',',-1) as Profession
    from name_basics_all
    where substring_index(substring_index(primaryProfession,',',2),',',-1) is not null
    Union
    select nconst,
        substring_index(substring_index(primaryProfession,',',3),',',-1) as Profession
    from name_basics_all
    where substring_index(substring_index(primaryProfession,',',3),',',-1) is not null
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
610 rows affected.
```

Out[36]: []

In [44]:
```
%%sql

select * from name_profession order by nconst limit 5;
```

```
 * mysql+pymysql://root:***@localhost
5 rows affected.
```

Out[44]:

nconst	Profession
nm0000293	actor
nm0000293	producer
nm0000293	animation_department
nm0000596	soundtrack
nm0000596	actor

In [39]:
```
%%sql

drop table if exists profession_types;

CREATE TABLE profession_types AS
```

```
    select distinct(Profession) as profession_name from name_profession
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
28 rows affected.
```

Out[39]: []

In [40]:
```
%%sql

select * from profession_types limit 5;
```

```
 * mysql+pymysql://root:***@localhost
5 rows affected.
```

Out[40]:

profession_name
actor
actress
stunts
writer
assistant_director

In [41]:
```
%%sql

ALTER TABLE profession_types modify profession_name varchar(64);
ALTER TABLE name_profession modify Profession varchar(64);
ALTER TABLE name_profession modify nconst varchar(16);
```

```
 * mysql+pymysql://root:***@localhost
28 rows affected.
610 rows affected.
0 rows affected.
```

Out[41]: []

In [42]:
```
%%sql

ALTER TABLE name_basics_all_fixed add primary key(nconst);
ALTER TABLE profession_types add primary key(profession_name);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
```

Out[42]: []

In [43]:
```
%%sql
ALTER TABLE name_profession
ADD FOREIGN KEY (nconst) REFERENCES name_basics_all_fixed(nconst);

ALTER TABLE name_profession
ADD FOREIGN KEY (Profession) REFERENCES profession_types(profession_name);
```

```
 * mysql+pymysql://root:***@localhost
610 rows affected.
610 rows affected.
```

Out[43]: []

Atomic knownForTitles

In [45]:
```
%%sql

drop table if exists name_title;

CREATE TABLE name_title AS
    select nconst,
        substring_index(substring_index(knownForTitles,',',1),',',-1) as title
    from name_basics_all
    where substring_index(substring_index(knownForTitles,',',1),',',-1) is not null
    Union
    select nconst,
        substring_index(substring_index(knownForTitles,',',2),',',-1) as title
    from name_basics_all
    where substring_index(substring_index(knownForTitles,',',2),',',-1) is not null
    Union
    select nconst,
```

```
        substring_index(substring_index(knownForTitles,',',3),',',-1) as title
    from name_basics_all
    where substring_index(substring_index(knownForTitles,',',3),',',-1) is not null
    Union
    select nconst,
        substring_index(substring_index(knownForTitles,',',4),',',-1) as title
    from name_basics_all
    where substring_index(substring_index(knownForTitles,',',4),',',-1) is not null
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
1317 rows affected.
```

Out[45]: []

In [48]:
```
%%sql

select * from name_title order by nconst limit 5;
```

```
 * mysql+pymysql://root:***@localhost
5 rows affected.
```

Out[48]:

nconst	title
nm0000293	tt1181791
nm0000293	tt0120737
nm0000293	tt0167261
nm0000293	tt0944947
nm0000596	tt0104348

In [46]:
```
%%sql

drop table if exists title_types;

CREATE TABLE title_types AS
    select distinct(title) as title_name from name_title
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
851 rows affected.
```

Out[46]: []

In [47]:
```
%%sql

select * from title_types limit 5;
```

```
 * mysql+pymysql://root:***@localhost
5 rows affected.
```

Out[47]:

title_name
tt0970411
tt0472160
tt1655420
tt4154664
tt0102797

In [50]:
```
%%sql

ALTER TABLE title_types modify title_name varchar(16);
ALTER TABLE name_title modify title varchar(16);
ALTER TABLE name_title modify nconst varchar(16);
```

```
 * mysql+pymysql://root:***@localhost
851 rows affected.
1317 rows affected.
0 rows affected.
```

Out[50]: []

In [51]:
```
%%sql

ALTER TABLE title_types add primary key(title_name);
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
```

Out[51]: []

In [52]:
```
%%sql
ALTER TABLE name_title
ADD FOREIGN KEY (nconst) REFERENCES name_basics_all_fixed(nconst);

ALTER TABLE name_title
ADD FOREIGN KEY (title) REFERENCES title_types(title_name);
```

```
 * mysql+pymysql://root:***@localhost
1317 rows affected.
1317 rows affected.
```

Out[52]: []

Final View

In [54]:
```
%%sql

drop view if exists name_basics_all_fixed_view;

CREATE VIEW name_basics_all_fixed_view as
    select
        name_basics_all_fixed.nconst,
        primaryName, birthYear, deathYear, T1.primaryProfession, T2.knownForTitles
    from
        name_basics_all_fixed
    left join(
        select nconst, group_concat(Profession) as primaryProfession
        from name_profession group by nconst
    ) T1 on name_basics_all_fixed.nconst = T1.nconst
    left join(
        select nconst, group_concat(title) as knownForTitles
        from name_title group by nconst
    ) T2 on name_basics_all_fixed.nconst = T2.nconst
```

```
 * mysql+pymysql://root:***@localhost
0 rows affected.
0 rows affected.
```

Out[54]: []

In [55]:
```
%%sql

select * from name_basics_all_fixed_view LIMIT 10;
```

```
 * mysql+pymysql://root:***@localhost
10 rows affected.
```

Out[55]:

nconst	primaryName	birthYear	deathYear	primaryProfession	knownForTitles
nm0000293	Sean Bean	1959	None	actor,producer,animation_department	tt0120737,tt0167261,tt0944947,tt1181791
nm0000596	Jonathan Pryce	1947	None	actor,soundtrack,producer	tt0104348,tt8404614,tt0120347,tt3750872
nm0000980	Jim Broadbent	1949	None	actor,writer,soundtrack	tt0203009,tt1431181,tt1007029,tt0217505
nm0001097	Charles Dance	1946	None	actor,director,writer	tt0944947,tt0107362,tt2084970,tt0280707
nm0001290	Richard E. Grant	1957	None	actor,soundtrack,director	tt4595882,tt0280707,tt0102070,tt0094336
nm0001354	Ciarán Hinds	1953	None	actor,soundtrack	tt1340800,tt1596365,tt1201607,tt12789558
nm0001671	Diana Rigg	1938	2020	actress,soundtrack,costume_department	tt0054518,tt9639470,tt0064757,tt0944947
nm0002103	Julian Glover	1935	None	actor,soundtrack	tt0082398,tt0332452,tt0080684,tt0097576
nm0004355	Roger Ashton-Griffiths	1957	None	actor,director,writer	tt0088846,tt0944947,tt4575576,tt0217505
nm0004692	Mark Addy	1964	None	actor,soundtrack	tt0944947,tt0955308,tt0119164,tt0183790