Media Database Project

ITCS 3160 - 001

Fall 2020

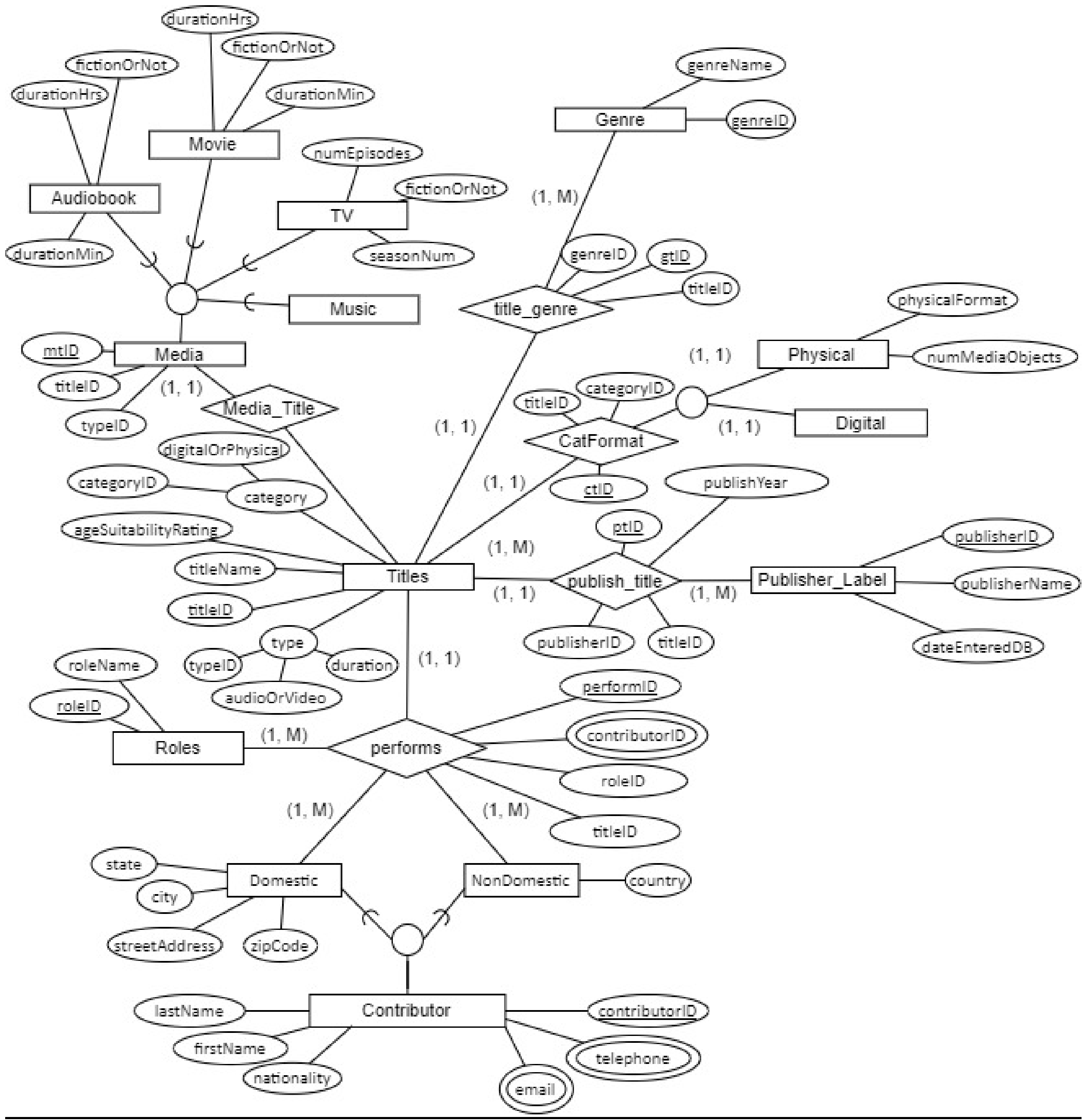
Samuel Garn
Matt Wylie

December 14, 2020

Abstract:

In this report, the group will be reviewing and outlining the SQL database created for a local library to keep track of their media titles. The range of data includes elements such as the contributor information and publisher information, as well as genres, media type, and titles. Our mission was to create a database to contain all this information and allow for ease of access. The database will be used to easily find and locate the selected titles withing the local library. In this report, the group will present diagrams, relational models, and an overview of the database. The diagram and models were created using the client's assumptions and the rules of the business. From this, the group created a database and implemented the data into the database. Additionally, the group ran sample queries in the database to demonstrate the effectiveness and usefulness of the system.

EERD:

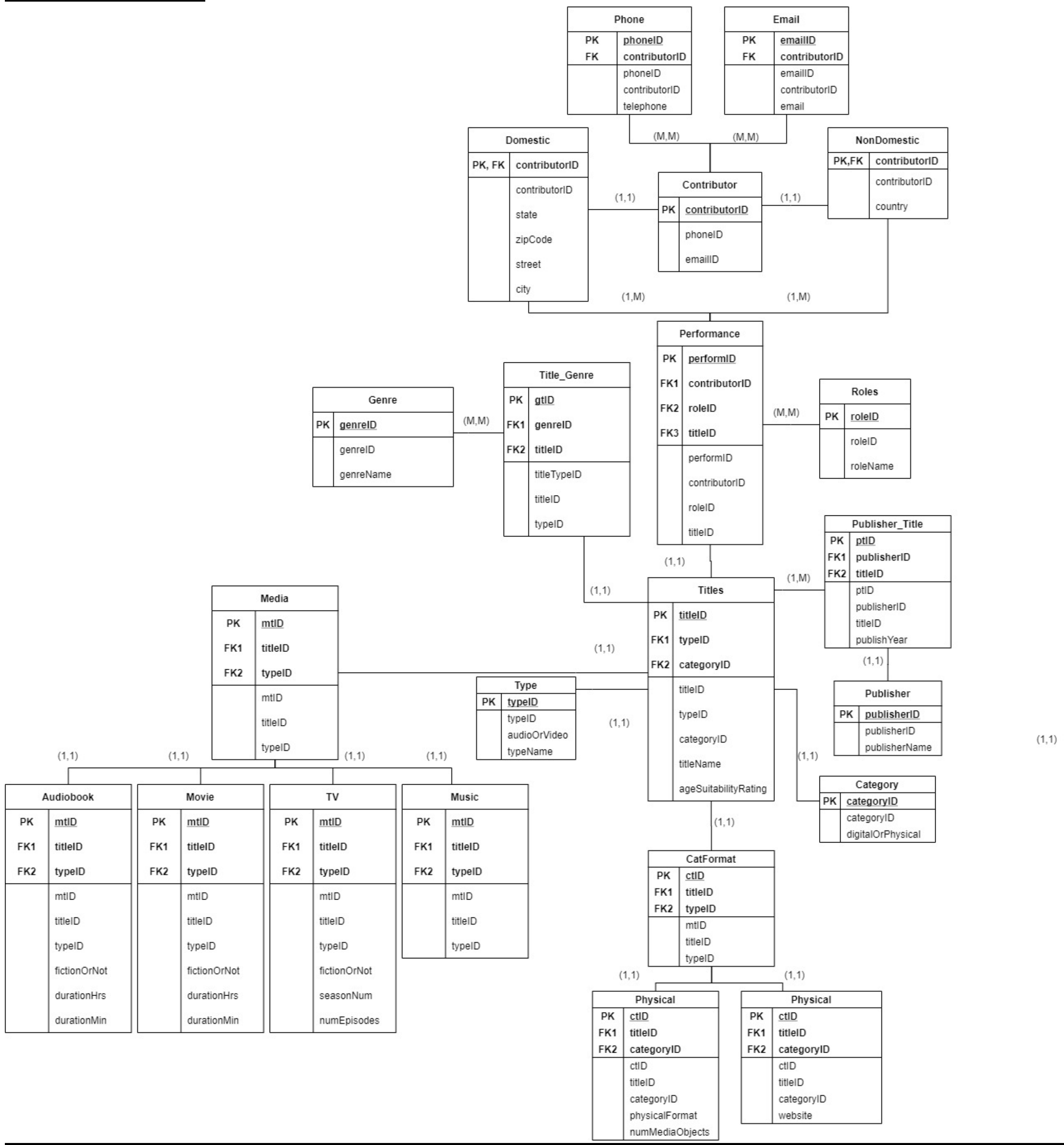

durationHrs
fictionOrNot
fictionOrNot
durationHrs
durationMin
Movie
Audiobook
numEpisodes
fictionOrNot
TV
durationMin
seasonNum
Music
genreName
Genre
genreID
(1, M)
genreID
gtID
titleID
title_genre
mtID
Media
titleID
(1, 1)
Media_Title
typeID
physicalFormat
(1, 1)
Physical
numMediaObjects
categoryID
titleID
Digital
(1, 1)
CatFormat
(1, 1)
digitalOrPhysical
categoryID
category
(1, 1)
ctID
publishYear
ageSuitabilityRating
ptID
(1, M)
publisherID
titleName
Titles
publish_title
Publisher_Label
publisherName
(1, 1)
(1, M)
titleID
publisherID
titleID
dateEnteredDB
type
typeID
duration
roleName
(1, 1)
audioOrVideo
performID
roleID
(1, M)
contributorID
Roles
performs
roleID
(1, M)
(1, M)
titleID
state
Domestic
NonDomestic
country
city
streetAddress
zipCode
lastName
Contributor
contributorID
firstName
telephone
nationality
email

List of Business Rules and Assumptions:

1. The MediaLib database should be normalized to 3NF.
2. Contributor – domestic/non-domestic contributor distinction should be represented a generalization-specialization relationship (superclass/subclass)
3. A title is defined by a name, publication year, publisher/label, and medium.
4. Different editions of a book should be considered different titles.
5. A book and a movie with the same name should be considered different titles.
6. It is possible to have multiple copies of the same title. It is suggested that you use different title names for multiple copies, perhaps with some sort of numerical sequencing, such as The Title (1) and The Title (2).
7. A title must have a category.
8. A title may not have multiple categories.
9. A title must have a publisher/label.
10. A title may not have multiple publisher/labels.
11. All titles must have at least one contributor.
12. A title may have multiple contributors.
13. A contributor must have a role for a title.
14. A contributor may have multiple roles for a title.
15. More than one contributor may have the same name.
16. Identically named contributors may contribute to the same title.
17. Multiple contributors may have the same role for a title.
18. It is expected that library users will want to search the database in combinations of title, year published, date entered, contributor, role, nationality, category, medium, and publisher/label. You should ensure that the formats of these fields are consistent across all titles and contributors.
19. A single genre should be used for a single media type, no single media type should have two genres.
20. Different media types may use the same genre i.e. horror novels and horror movies
21. A publisher may have published multiple titles
22. A publisher label can only have one publisher name/publisher ID.

23. A genre can only have a single name/ID
24. A contributor can only be one subclass, either domestic or non-domestic.
25. Media can only have one suitability rating per title.
26. Media must have a genre.
27. Media can have multiple genres.
28. Media must have a type.
29. Each category must have a format/platform
30. A category can have multiple formats/platforms i.e. movie can have multiple streaming platforms or novels can be hard or soft cover.
31. dateEntered must be greater than or equal to yearPublished
32. ZIP codes must be 5 digits.

Relational Model:



<u>Database, Sample Data, and Test Query</u>

Sample data was created for the entities and inserted into tables for the creation of our database. Below the group will demonstrate the creation of the database, a sample of the sample data used, as well as some test queries to ensure the database was created properly.

<u>Database Creation:</u>

The creation of the database was done by using the below script:

```
DROP DATABASE IF EXISTS medialib;
CREATE DATABASE IF NOT EXISTS medialib;
USE medialib;
```

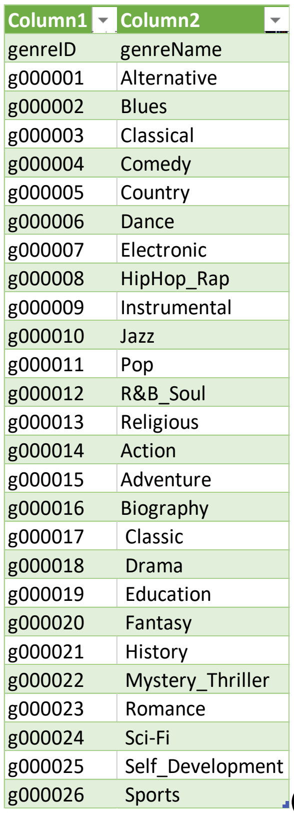
What this script does is firstly drop any preexisting databases with the same name. After this is done it creates the database if it has not done so already. Finally, it selects the created database to use. The creation of the tables looked similarly to these:

```
DROP TABLE IF EXISTS phone;
CREATE TABLE phone (
  phoneID VARCHAR(7) NOT NULL,
  contributorFID VARCHAR(7) NULL,
  telephone VARCHAR(12) NULL,
  PRIMARY KEY(phoneID)
);
DROP TABLE IF EXISTS email;
CREATE TABLE email (
  emailID VARCHAR(5) NOT NULL,
  contributorFID VARCHAR(7) NULL,
  email VARCHAR(80) NULL,
  PRIMARY KEY(emailID)
);
DROP TABLE IF EXISTS contributor;
CREATE TABLE contributor (
  contributorID VARCHAR(7) NOT NULL,
  phoneFID VARCHAR(7) NOT NULL,
  emailFID VARCHAR(5) NOT NULL,
  lastName VARCHAR(20) NOT NULL,
  firstName VARCHAR(20) NULL,
  PRIMARY KEY(contributorID),
  FOREIGN KEY(phoneFID) REFERENCES phone(phoneID),
  FOREIGN KEY(emailFID) REFERENCES email(emailID)
);
```

There are 22 total tables, however, to keep this report concise, these three were chosen to represent the creation of the tables.

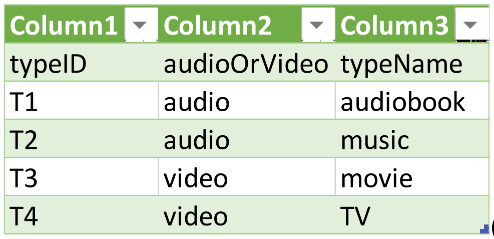
<u>Sample Data:</u>

Below is only a few tables of sample data to give a scope of the elements used in the creation of this database. As there are 22 tables in total, not all of them are included in this report. However, they have all been loaded into the database using the creation of tables and INSERT statements. Here is what that entails:



Column1	Column2
genreID	genreName
g000001	Alternative
g000002	Blues
g000003	Classical
g000004	Comedy
g000005	Country
g000006	Dance
g000007	Electronic
g000008	HipHop_Rap
g000009	Instrumental
g000010	Jazz
g000011	Pop
g000012	R&B_Soul
g000013	Religious
g000014	Action
g000015	Adventure
g000016	Biography
g000017	Classic
g000018	Drama
g000019	Education
g000020	Fantasy
g000021	History
g000022	Mystery_Thriller
g000023	Romance
g000024	Sci-Fi
g000025	Self_Development
g000026	Sports

(Genre table)



Column1	Column2	Column3
typeID	audioOrVideo	typeName
T1	audio	audiobook
T2	audio	music
T3	video	movie
T4	video	TV

(Type of Media table)

From these tables the group took the data and using INSERT statements added all the tables into the database. As stated above, to keep the report concise, the group chose these to represent the 22 INSERT statements:

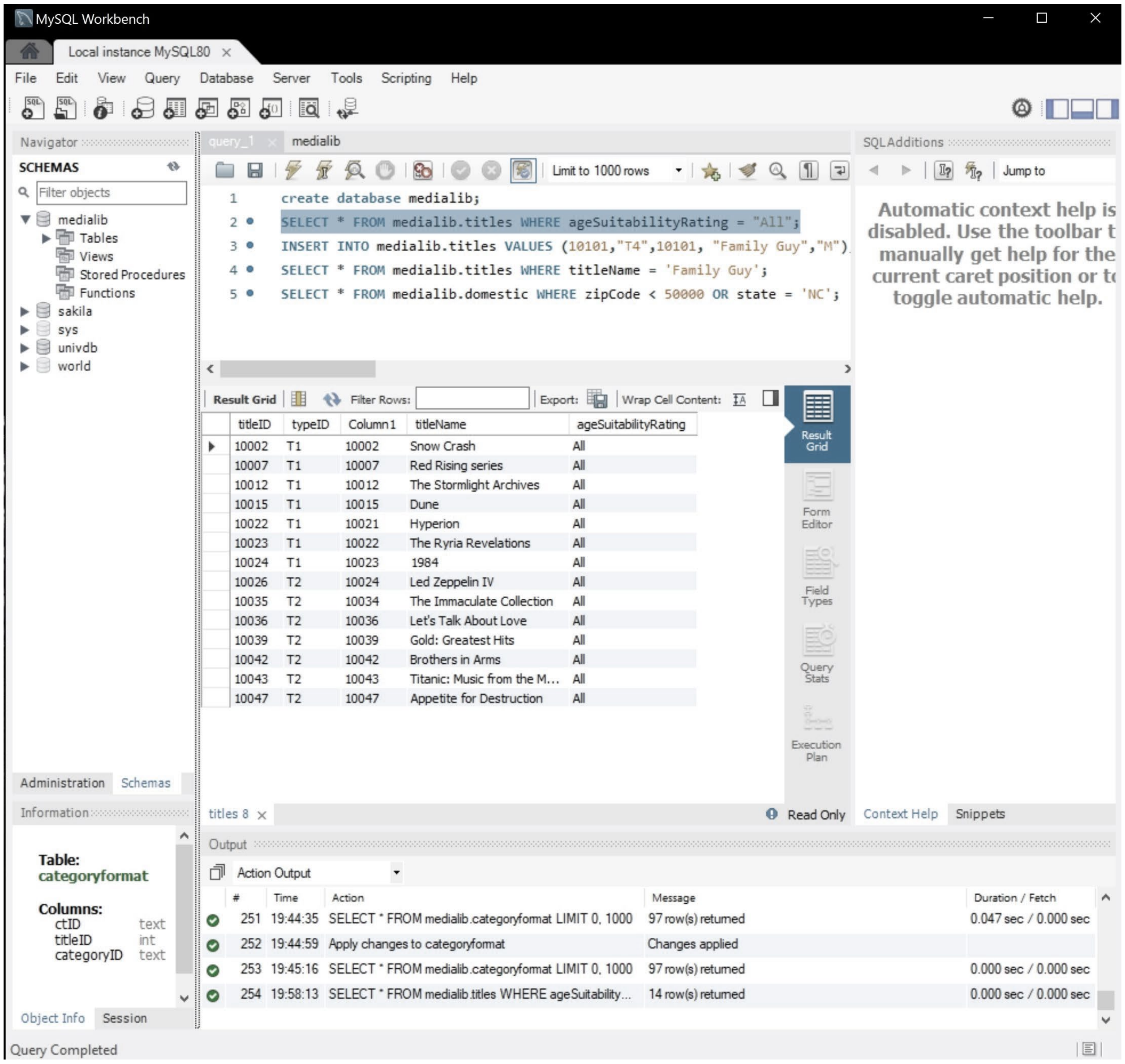
```
INSERT INTO audiobook (mtID,typeFID,titleFID,durationHrs,durationMin,fictionOrNot) VALUES
('mt000001','T1',10000,29,24,'fiction'),('mt000002','T1',10001,20,4,'fiction'),('mt000003','T1',10002,19,20,'fi
ction'),('mt000004','T1',10003,34,42,'fiction'),('mt000005','T1',10004,1,5,'fiction'),('mt000006','T1',10005,10
,8,'fiction'),('mt000007','T1',10006,24,55,'fiction'),('mt000008','T1',10007,23,56,'fiction'),('mt000009','T1',10
008,22,17,'fiction'),('mt000010','T1',10009,1,17,'nonFiction'),('mt000011','T1',10010,23,31,'nonFiction'),('m
t000012','T1',10011,1,40,'fiction'),('mt000013','T1',10012,12,56,'nonFiction'),('mt000014','T1',10013,10,10,
'fiction'),('mt000015','T1',10014,26,39,'fiction'),('mt000016','T1',10015,4,39,'fiction'),('mt000017','T1',10016,
9,43,'fiction'),('mt000018','T1',10017,32,17,'nonFiction'),('mt000019','T1',10018,1,57,'nonFiction'),('mt0000
20','T1',10019,8,57,'nonFiction'),('mt000021','T1',10020,27,11,'fiction'),('mt000022','T1',10022,8,27,'fiction'
),('mt000023','T1',10023,30,36,'fiction'),('mt000024','T1',10024,14,45,'fiction');
INSERT INTO category (categoryID,digitalOrPhysical) VALUES ('CA001','physical'),('CA002','digital');
INSERT INTO genre (genreID,genreName) VALUES
('g000001','Alternative'),('g000002','Blues'),('g000003','Classical'),('g000004','Comedy'),('g000005','Countr
y'),('g000006','Dance'),('g000007','Electronic'),('g000008','HipHop_Rap'),('g000009','Instrumental'),('g0000
10','Jazz'),('g000011','Pop'),('g000012','R&B_Soul'),('g000013','Religious'),('g000014','Action'),('g000015','
Adventure'),('g000016','Biography'),('g000017',' Classic'),('g000018',' Drama'),('g000019','
Education'),('g000020',' Fantasy'),('g000021',' History'),('g000022',' Mystery_Thriller'),('g000023','
Romance'),('g000024',' Sci-Fi'),('g000025',' Self_Development'),('g000026',' Sports');
```

<u>Test Queries:</u>

Finally, the group ran some test queries to make sure the database was working and implemented correctly. The query script is as follows:

SELECT * FROM medialib.titles WHERE ageSuitabilityRating = "All";
INSERT INTO medialib.titles VALUES (10101,"T4",10101, "Family Guy","M");
SELECT * FROM medialib.titles WHERE titleName = 'Family Guy';
SELECT * FROM medialib.domestic WHERE zipCode < 50000 OR state = 'NC';

Running the query produced the results below:



```
create database medialib;
SELECT * FROM medialib.titles WHERE ageSuitabilityRating = "All";
INSERT INTO medialib.titles VALUES (10101,"T4",10101, "Family Guy","M")
SELECT * FROM medialib.titles WHERE titleName = 'Family Guy';
SELECT * FROM medialib.domestic WHERE zipCode < 50000 OR state = 'NC';
```

titleID	typeID	Column1	titleName	ageSuitabilityRating
10002	T1	10002	Snow Crash	All
10007	T1	10007	Red Rising series	All
10012	T1	10012	The Stormlight Archives	All
10015	T1	10015	Dune	All
10022	T1	10021	Hyperion	All
10023	T1	10022	The Ryria Revelations	All
10024	T1	10023	1984	All
10026	T2	10024	Led Zeppelin IV	All
10035	T2	10034	The Immaculate Collection	All
10036	T2	10036	Let's Talk About Love	All
10039	T2	10039	Gold: Greatest Hits	All
10042	T2	10042	Brothers in Arms	All
10043	T2	10043	Titanic: Music from the M...	All
10047	T2	10047	Appetite for Destruction	All

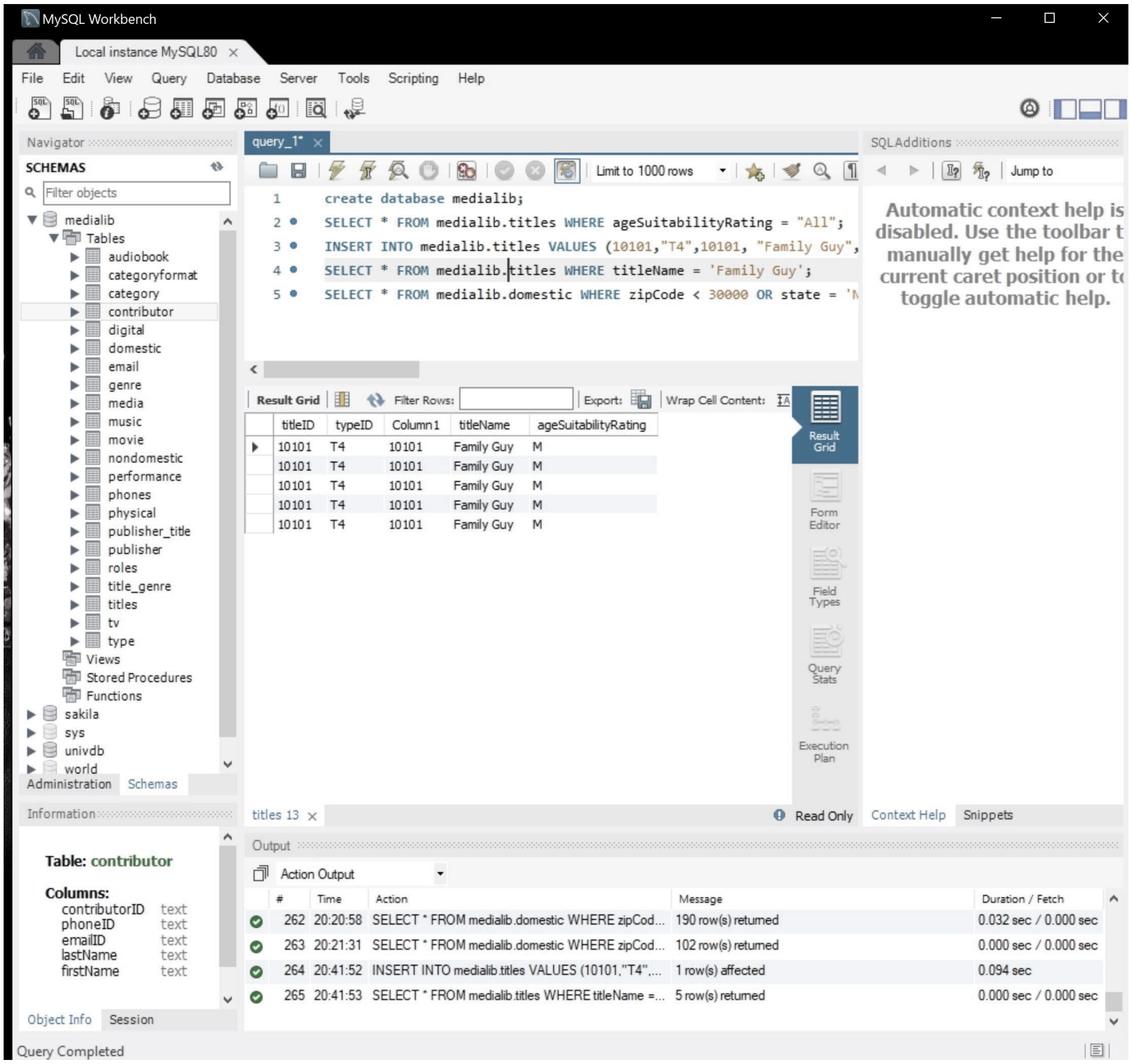
Table: categoryformat

Columns:
ctID text
titleID int
categoryID text

```
#    Time      Action                                                Message              Duration / Fetch
251  19:44:35  SELECT * FROM medialib.categoryformat LIMIT 0, 1000  97 row(s) returned   0.047 sec / 0.000 sec
252  19:44:59  Apply changes to categoryformat                      Changes applied
253  19:45:16  SELECT * FROM medialib.categoryformat LIMIT 0, 1000  97 row(s) returned   0.000 sec / 0.000 sec
254  19:58:13  SELECT * FROM medialib.titles WHERE ageSuitability...  14 row(s) returned   0.000 sec / 0.000 sec
```

Query Completed

(Query 1.1)



```sql
create database medialib;
SELECT * FROM medialib.titles WHERE ageSuitabilityRating = "All";
INSERT INTO medialib.titles VALUES (10101,"T4",10101, "Family Guy",
SELECT * FROM medialib.titles WHERE titleName = 'Family Guy';
SELECT * FROM medialib.domestic WHERE zipCode < 30000 OR state = 'N
```

titleID	typeID	Column1	titleName	ageSuitabilityRating
10101	T4	10101	Family Guy	M
10101	T4	10101	Family Guy	M
10101	T4	10101	Family Guy	M
10101	T4	10101	Family Guy	M
10101	T4	10101	Family Guy	M

Table: contributor

Columns:

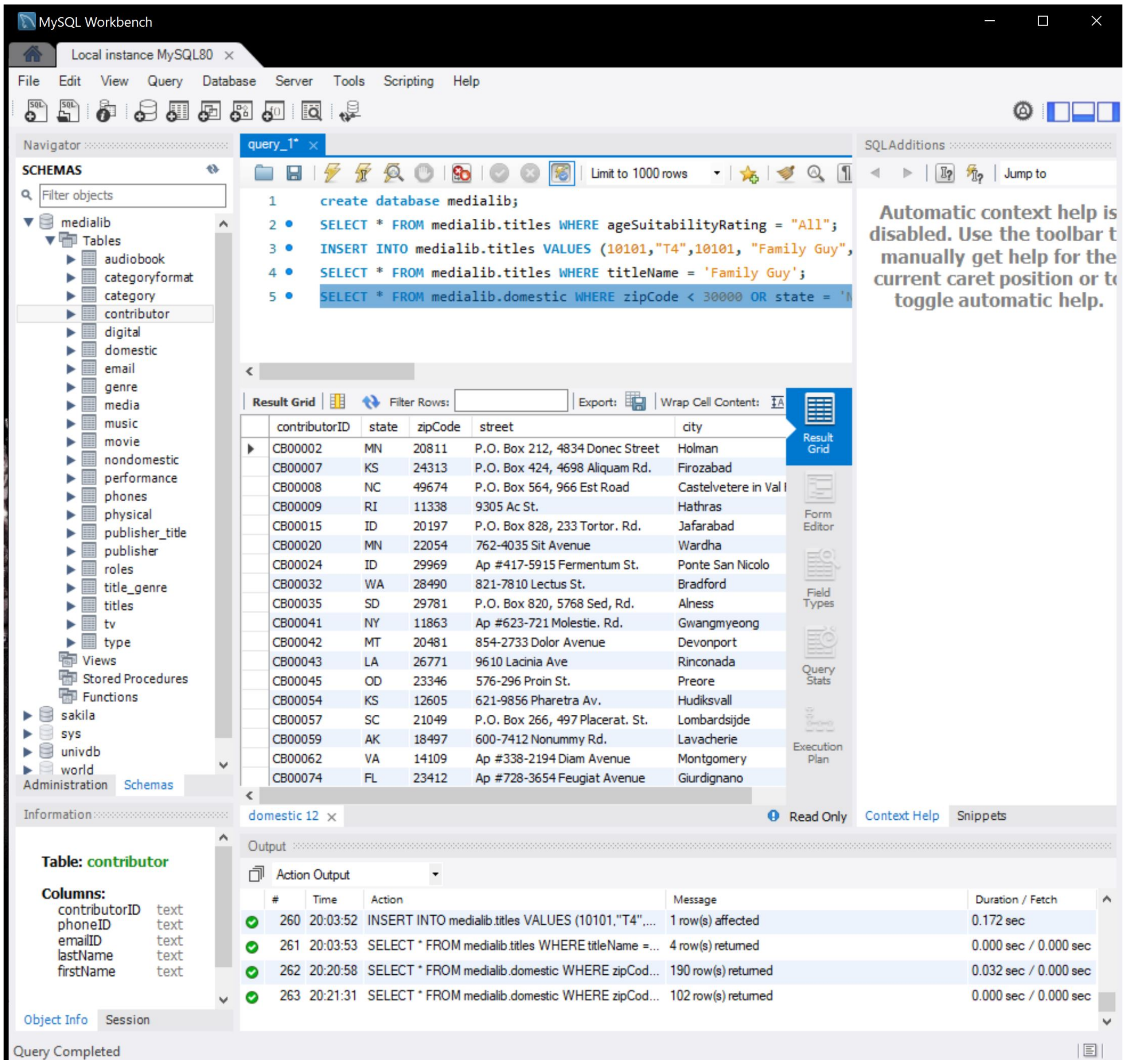
contributorID	text
phoneID	text
emailID	text
lastName	text
firstName	text

```
#    Time      Action                                            Message             Duration / Fetch
262  20:20:58  SELECT * FROM medialib.domestic WHERE zipCod...   190 row(s) returned  0.032 sec / 0.000 sec
263  20:21:31  SELECT * FROM medialib.domestic WHERE zipCod...   102 row(s) returned  0.000 sec / 0.000 sec
264  20:41:52  INSERT INTO medialib.titles VALUES (10101,"T4",... 1 row(s) affected   0.094 sec
265  20:41:53  SELECT * FROM medialib.titles WHERE titleName =... 5 row(s) returned   0.000 sec / 0.000 sec
```

Query Completed

(Query 1.2)

(Please note, the reason for the multiple entries of “Family Guy” resulted from running the query multiple times. These were later deleted to avoid repetition and confusion.)



```
create database medialib;
SELECT * FROM medialib.titles WHERE ageSuitabilityRating = "All";
INSERT INTO medialib.titles VALUES (10101,"T4",10101, "Family Guy",
SELECT * FROM medialib.titles WHERE titleName = 'Family Guy';
SELECT * FROM medialib.domestic WHERE zipCode < 30000 OR state = 'N
```

Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or t toggle automatic help.

contributorID	state	zipCode	street	city
CB00002	MN	20811	P.O. Box 212, 4834 Donec Street	Holman
CB00007	KS	24313	P.O. Box 424, 4698 Aliquam Rd.	Firozabad
CB00008	NC	49674	P.O. Box 564, 966 Est Road	Castelvetere in Val
CB00009	RI	11338	9305 Ac St.	Hathras
CB00015	ID	20197	P.O. Box 828, 233 Tortor. Rd.	Jafarabad
CB00020	MN	22054	762-4035 Sit Avenue	Wardha
CB00024	ID	29969	Ap #417-5915 Fermentum St.	Ponte San Nicolo
CB00032	WA	28490	821-7810 Lectus St.	Bradford
CB00035	SD	29781	P.O. Box 820, 5768 Sed, Rd.	Alness
CB00041	NY	11863	Ap #623-721 Molestie. Rd.	Gwangmyeong
CB00042	MT	20481	854-2733 Dolor Avenue	Devonport
CB00043	LA	26771	9610 Lacinia Ave	Rinconada
CB00045	OD	23346	576-296 Proin St.	Preore
CB00054	KS	12605	621-9856 Pharetra Av.	Hudiksvall
CB00057	SC	21049	P.O. Box 266, 497 Placerat. St.	Lombardsijde
CB00059	AK	18497	600-7412 Nonummy Rd.	Lavacherie
CB00062	VA	14109	Ap #338-2194 Diam Avenue	Montgomery
CB00074	FL	23412	Ap #728-3654 Feugiat Avenue	Giurdignano

Table: contributor

Columns:

contributorID	text
phoneID	text
emailID	text
lastName	text
firstName	text

#	Time	Action	Message	Duration / Fetch
260	20:03:52	INSERT INTO medialib.titles VALUES (10101,"T4",...	1 row(s) affected	0.172 sec
261	20:03:53	SELECT * FROM medialib.titles WHERE titleName =...	4 row(s) returned	0.000 sec / 0.000 sec
262	20:20:58	SELECT * FROM medialib.domestic WHERE zipCod...	190 row(s) returned	0.032 sec / 0.000 sec
263	20:21:31	SELECT * FROM medialib.domestic WHERE zipCod...	102 row(s) returned	0.000 sec / 0.000 sec

(Query 1.3)

Data Dictionary:

Below is the groups database data dictionary created from our database. The data dictionary is used to hold defining information about the database and is updated every time an event or change occurs.

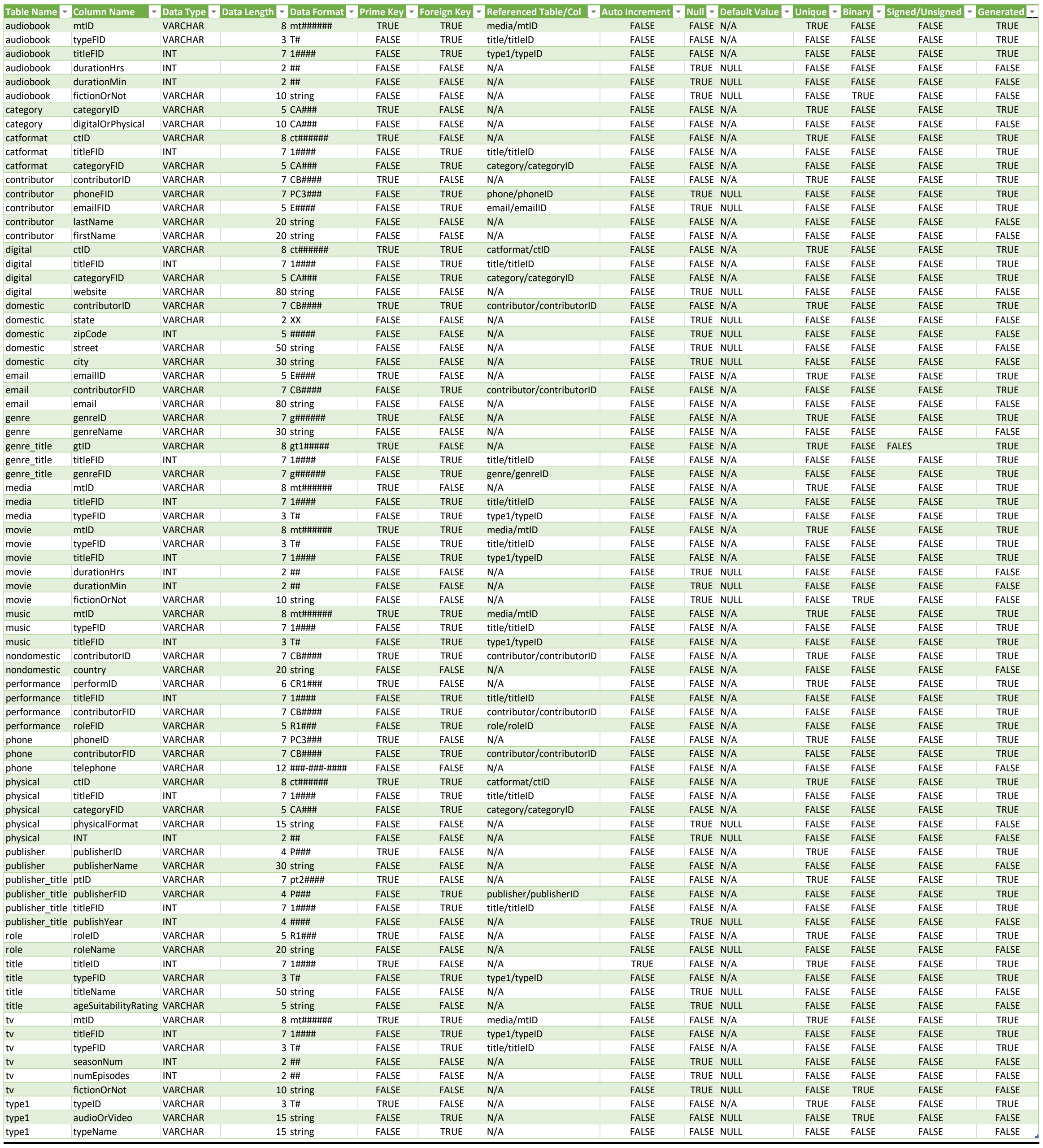


Table Name	Column Name	Data Type	Data Length	Data Format	Prime Key	Foreign Key	Referenced Table/Col	Auto Increment	Null	Default Value	Unique	Binary	Signed/Unsigned	Generated
audiobook	mtID	VARCHAR	8	mt######	TRUE	TRUE	media/mtID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
audiobook	typeFID	VARCHAR	3	T#	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
audiobook	titleFID	INT	7	1####	FALSE	TRUE	type1/typeID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
audiobook	durationHrs	INT	2	##	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
audiobook	durationMin	INT	2	##	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
audiobook	fictionOrNot	VARCHAR	10	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	TRUE	FALSE	FALSE
category	categoryID	VARCHAR	5	CA###	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
category	digitalOrPhysical	VARCHAR	10	CA###	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
catformat	ctID	VARCHAR	8	ct######	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
catformat	titleFID	INT	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
catformat	categoryFID	VARCHAR	5	CA###	FALSE	TRUE	category/categoryID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
contributor	contributorID	VARCHAR	7	CB####	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
contributor	phoneFID	VARCHAR	7	PC3###	FALSE	TRUE	phone/phoneID	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	TRUE
contributor	emailFID	VARCHAR	5	E####	FALSE	TRUE	email/emailID	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	TRUE
contributor	lastName	VARCHAR	20	string	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
contributor	firstName	VARCHAR	20	string	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
digital	ctID	VARCHAR	8	ct######	TRUE	TRUE	catformat/ctID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
digital	titleFID	INT	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
digital	categoryFID	VARCHAR	5	CA###	FALSE	TRUE	category/categoryID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
digital	website	VARCHAR	80	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
domestic	contributorID	VARCHAR	7	CB####	TRUE	TRUE	contributor/contributorID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
domestic	state	VARCHAR	2	XX	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
domestic	zipCode	INT	5	#####	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
domestic	street	VARCHAR	50	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
domestic	city	VARCHAR	30	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
email	emailID	VARCHAR	5	E####	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
email	contributorFID	VARCHAR	7	CB####	FALSE	TRUE	contributor/contributorID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
email	email	VARCHAR	80	string	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
genre	genreID	VARCHAR	7	g######	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
genre	genreName	VARCHAR	30	string	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
genre_title	gtID	VARCHAR	8	gt1#####	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALES	TRUE
genre_title	titleFID	INT	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
genre_title	genreFID	VARCHAR	7	g######	FALSE	TRUE	genre/genreID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
media	mtID	VARCHAR	8	mt######	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
media	titleFID	INT	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
media	typeFID	VARCHAR	3	T#	FALSE	TRUE	type1/typeID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
movie	mtID	VARCHAR	8	mt######	TRUE	TRUE	media/mtID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
movie	typeFID	VARCHAR	3	T#	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
movie	titleFID	INT	7	1####	FALSE	TRUE	type1/typeID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
movie	durationHrs	INT	2	##	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
movie	durationMin	INT	2	##	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
movie	fictionOrNot	VARCHAR	10	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	TRUE	FALSE	FALSE
music	mtID	VARCHAR	8	mt######	TRUE	TRUE	media/mtID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
music	typeFID	VARCHAR	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
music	titleFID	INT	3	T#	FALSE	TRUE	type1/typeID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
nondomestic	contributorID	VARCHAR	7	CB####	TRUE	TRUE	contributor/contributorID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
nondomestic	country	VARCHAR	20	string	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
performance	performID	VARCHAR	6	CR1###	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
performance	titleFID	INT	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
performance	contributorFID	VARCHAR	7	CB####	FALSE	TRUE	contributor/contributorID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
performance	roleFID	VARCHAR	5	R1###	FALSE	TRUE	role/roleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
phone	phoneID	VARCHAR	7	PC3###	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
phone	contributorFID	VARCHAR	7	CB####	FALSE	TRUE	contributor/contributorID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
phone	telephone	VARCHAR	12	###-###-####	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
physical	ctID	VARCHAR	8	ct######	TRUE	TRUE	catformat/ctID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
physical	titleFID	INT	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
physical	categoryFID	VARCHAR	5	CA###	FALSE	TRUE	category/categoryID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
physical	physicalFormat	VARCHAR	15	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
physical	INT	INT	2	##	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
publisher	publisherID	VARCHAR	4	P###	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
publisher	publisherName	VARCHAR	30	string	FALSE	FALSE	N/A	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	FALSE
publisher_title	ptID	VARCHAR	7	pt2####	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
publisher_title	publisherFID	VARCHAR	4	P###	FALSE	TRUE	publisher/publisherID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
publisher_title	titleFID	INT	7	1####	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
publisher_title	publishYear	INT	4	####	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
role	roleID	VARCHAR	5	R1###	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
role	roleName	VARCHAR	20	string	FALSE	FALSE	N/A	FALSE	FALSE	NULL	FALSE	FALSE	FALSE	FALSE
title	titleID	INT	7	1####	TRUE	FALSE	N/A	TRUE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
title	typeFID	VARCHAR	3	T#	FALSE	TRUE	type1/typeID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
title	titleName	VARCHAR	50	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
title	ageSuitabilityRating	VARCHAR	5	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
tv	mtID	VARCHAR	8	mt######	TRUE	TRUE	media/mtID	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
tv	titleFID	INT	7	1####	FALSE	TRUE	type1/typeID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
tv	typeFID	VARCHAR	3	T#	FALSE	TRUE	title/titleID	FALSE	FALSE	N/A	FALSE	FALSE	FALSE	TRUE
tv	seasonNum	INT	2	##	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
tv	numEpisodes	INT	2	##	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	FALSE	FALSE	FALSE
tv	fictionOrNot	VARCHAR	10	string	FALSE	FALSE	N/A	FALSE	TRUE	NULL	FALSE	TRUE	FALSE	FALSE
type1	typeID	VARCHAR	3	T#	TRUE	FALSE	N/A	FALSE	FALSE	N/A	TRUE	FALSE	FALSE	TRUE
type1	audioOrVideo	VARCHAR	15	string	FALSE	TRUE	N/A	FALSE	FALSE	NULL	FALSE	TRUE	FALSE	FALSE
type1	typeName	VARCHAR	15	string	FALSE	TRUE	N/A	FALSE	FALSE	NULL	FALSE	FALSE	FALSE	FALSE

<u>Queries for Reporting:</u>

The group will demonstrate how the database functions by presenting four reports in the following categories: Title Report, Contributor Report, Media Report, Category Report. The group will provide the queries required to produce these reports at the end of this document for the sake of readablitity

<u>Title Report</u>

This query will produce three outputs. The first being a list including titleName, publication year, entry date, categoryName, mediaName, publisherName, and firstName, lastName, roleName, and nationalityName for all contributors. They will also be listed in alphabetical order based on the titleName. The second output is a list that includes titlename, categoryName, mediaName for all movies available in DVD or BluRay format. Finally, the third list includes the titles of all drama movies with counts for DVD, BluRay, and streaming titles.

<u>Contributor Report</u>

This query will produce two outputs. The first being a favorite musical artist selected by the group and all of their contributions. The second of the two list is that of all British artists along with their contributions.

<u>Media Report</u>

This query will count the number of Jazz titles in all forms of musical media. This includes all CD’s, MP3’s, etc.

<u>Category Report</u>

Finally, this query will count the number of audiobooks in each format they appear in. This could be a Kindle or an audio CD.

To see the full queries, please refer to the documentation at the end of this report. They will be listed in the order presented here.

List and Descriptions of Advanced Features:

1. Generalization-specialization relationships
    a. Generalization is defined as taking shared elements of data and combining them into a single “master” class. On the contrary, specialization is defined by creating smaller classes or tables from a larger or existing class/table.
2. Stored Procedure
    a. A stored procedure in SQL is simple a routine that is stored in the database itself and can be called and used on the database. Such procedures are stored in the database itself as to allow for anyone with access to use them. Below is a sample procedure:

```
DELIMITER //
CREATE PROCEDURE getTitles()
BEGIN
        SELECT *  FROM title
                ORDER BY title.titleID
                        LIMIT 10
END //
DELIMITER ;
CALL getTitles();
```

3. Trigger
    a. A trigger is similar to the above-mentioned stored procedures. A trigger is a stored procedure that will automatically run anytime a specified condition is met or a certain activity occurs. Below is a sample of a trigger:

```
DROP TRIGGER IF EXISTS before_delete_contributor;
DELIMITER //
CREATE TRIGGER before_delete_contributor
 BEFORE DELETE
 ON contributor FOR EACH ROW
BEGIN
 INSERT INTO altered_contributors
(contributorID,phoneFID,emailFID,lastName,firstName) values (old.contributorID,
old.phoneFID, old.emailFID,old.lastName,old.firstName);
END//
DELIMITER ;
SHOW TRIGGERS;
```

4. DELETE/UPDATE examples
    a. DELETE is a key statement used to remove a single column/row of data as well as removing multiple columns/rows. On the other hand, UPDATE is a statement used to change the value of the data in a single row or column as well as being able to change multiple rows or columns. Here is an example of each.
    b. DELETE

```
SELECT *
```

```
        FROM contributor
                ORDER BY contributorID
                        LIMIT 5;
DELETE FROM medialib.contributor WHERE contributor.contributorID = 'CB00001' AND
contributor.lastName = 'Fitzgerald';
SELECT *
        FROM contributor
                ORDER BY contributorID
                        LIMIT 5;
SET SQL_SAFE_UPDATES = 0;
```

c. UPDATE

```
UPDATE contributor
        SET lastName = 'Wylie'
                WHERE contributor.contributorID = 'CB00002';
SELECT *
        FROM contributor
                WHERE lastName = 'Wylie';
```

5. Indexes for reports
    a. Indexes are used in SQL to quickly retrieve data by providing a table of codes that will lead the user to the information they are looking for. This allows for increased search times and better organization.

Queries for Required Reports:

Title Reports:

```
DROP TABLE IF EXISTS newTable;
CREATE TABLE newTable
        AS SELECT
                title.titleName, media.mediaName, title.titleID
                FROM title
                        LEFT JOIN media ON media.titleFID = title.titleID;
DROP TABLE IF EXISTS newTable1;
CREATE TABLE newTable1
        AS SELECT
                newTable.titleName, newTable.mediaName, newTable.titleID, physical.physicalFormat,
physical.categoryFID
                FROM newTable
                        LEFT JOIN physical ON physical.titleFID = newTable.titleID;
DROP TABLE IF EXISTS newTable2;
CREATE TABLE newTable2
        AS SELECT
                newTable1.titleName, newTable1.mediaName, newTable1.titleID,
newTable1.physicalFormat, newTable1.categoryFID, category.digitalOrPhysical
                FROM newTable1
                        LEFT JOIN category ON category.categoryID = newTable1.categoryFID;
SELECT newTable2.titleID, newTable2.titleName AS 'Title', newTable2.digitalOrPhysical AS 'Category
Name', newTable2.mediaName AS 'Media', newTable2.physicalFormat AS'FORMAT'
                FROM newTable2
                        WHERE newTable2.physicalFormat =  'CD' OR newTable2.physicalFormat =
'BluRay';
-- Title Reports 1a. List by titleName, publication year, entry date, categoryName, mediaName,
publisherName, and firstName, lastName, roleName, and nationalityName for all contributors.  List in
titleName order.
DROP TABLE IF EXISTS newTable3;
CREATE TABLE newTable3
        AS SELECT
                title.titleID, title.typeFID, title.titleName, performance.performID,
performance.contributorFID, performance.roleFID
                FROM title
                        LEFT JOIN performance ON performance.titleFID = title.titleID;
DROP TABLE IF EXISTS newTable4;
CREATE TABLE newTable4
        AS SELECT
                newtable3.titleID, newtable3.typeFID, newtable3.titleName, newtable3.performID,
newtable3.contributorFID, newtable3.roleFID, contributor.contributorID, contributor.lastName,
contributor.firstName, contributor.phoneFID, contributor.emailFID, contributor.nationality
                FROM newtable3
                        LEFT JOIN contributor ON contributor.contributorID = newtable3.contributorFID;
SELECT * FROM medialib.newtable4 LIMIT 10;
DROP TABLE IF EXISTS newTable5;
CREATE TABLE newTable5
        AS SELECT
                newTable4.titleID, newTable4.typeFID, newTable4.titleName, newTable4.performID,
newTable4.contributorFID, newTable4.nationality,
     newTable4.roleFID, newTable4.contributorID, newTable4.lastName, newTable4.firstName,
newTable4.phoneFID, newTable4.emailFID,
     publisher_title.ptID, publisher_title.publisherFID, publisher_title.titleFID, publisher_title.publishYear
                FROM newTable4
```

```
                        LEFT JOIN publisher_title ON publisher_title.titleFID = newTable4.titleID;
SELECT * FROM medialib.newTable5 LIMIT 10;
DROP TABLE IF EXISTS newTable6;
CREATE TABLE newTable6
        AS SELECT
                newTable5.titleID, newTable5.typeFID, newTable5.titleName, newTable5.performID,
newTable5.contributorFID, newTable5.nationality,
     newTable5.roleFID, newTable5.contributorID, newTable5.lastName, newTable5.firstName,
newTable5.phoneFID, newTable5.emailFID,
     newTable5.ptID, newTable5.publisherFID, newTable5.titleFID, newTable5.publishYear,
catformat.ctID, catformat.categoryFID
                FROM newTable5
                        LEFT JOIN catformat ON catformat.titleFID = newTable5.titleID;
DROP TABLE IF EXISTS newTable7;
CREATE TABLE newTable7
        AS SELECT
                newTable6.titleID, newTable6.typeFID, newTable6.titleName, newTable6.performID,
newTable6.contributorFID, newTable6.nationality,
     newTable6.roleFID, newTable6.contributorID, newTable6.lastName, newTable6.firstName,
newTable6.phoneFID, newTable6.emailFID,
     newTable6.ptID, newTable6.publisherFID, newTable6.titleFID, newTable6.publishYear,
newTable6.ctID, newTable6.categoryFID,
     category.digitalOrPhysical
                FROM newTable6
                        LEFT JOIN category ON category.categoryID = newTable6.categoryFID;
DROP TABLE IF EXISTS newTable8;
CREATE TABLE newTable8
        AS SELECT
                newTable7.titleID, newTable7.typeFID, newTable7.titleName, newTable7.performID,
newTable7.contributorFID, newTable7.nationality,
     newTable7.roleFID, newTable7.contributorID, newTable7.lastName, newTable7.firstName,
newTable7.phoneFID, newTable7.emailFID,
     newTable7.ptID, newTable7.publisherFID, newTable7.titleFID, newTable7.publishYear,
newTable7.ctID, newTable7.categoryFID,
     newTable7.digitalOrPhysical, media.mtID, media.mediaName
                FROM newTable7
                        LEFT JOIN media ON media.titleFID = newTable7.titleID;
DROP TABLE IF EXISTS newTable9;
CREATE TABLE newTable9
        AS SELECT
                newTable8.titleID, newTable8.typeFID, newTable8.titleName, newTable8.performID,
newTable8.contributorFID, newTable8.nationality,
     newTable8.roleFID, newTable8.contributorID, newTable8.lastName, newTable8.firstName,
newTable8.phoneFID, newTable8.emailFID,
     newTable8.ptID, newTable8.publisherFID, newTable8.titleFID, newTable8.publishYear,
newTable8.ctID, newTable8.categoryFID,
     newTable8.digitalOrPhysical, newTable8.mtID, newTable8.mediaName, publisher.publisherName
                FROM newTable8
                        LEFT JOIN publisher ON publisher.publisherID = newTable8.publisherFID;
DROP TABLE IF EXISTS newTable10;
CREATE TABLE newTable10
        AS SELECT
                newTable9.titleID, newTable9.typeFID, newTable9.titleName, newTable9.performID,
newTable9.contributorFID, newTable9.nationality,
     newTable9.roleFID, newTable9.contributorID, newTable9.lastName, newTable9.firstName,
newTable9.phoneFID, newTable9.emailFID,
      newTable9.ptID, newTable9.publisherFID, newTable9.titleFID, newTable9.publishYear,
newTable9.ctID, newTable9.categoryFID,
      newTable9.digitalOrPhysical, newTable9.mtID, newTable9.mediaName, newTable9.publisherName,
role.roleName
                  FROM newTable9
                        LEFT JOIN role ON role.roleID = newTable9.roleFID;

-- Title Reports c.       List the titles of all drama movies with counts for DVD, BluRay, and streaming
titles.

DROP TABLE IF EXISTS newTable11;
CREATE TABLE newTable11
      AS SELECT
                  newTable10.titleID, newTable10.typeFID, newTable10.titleName, newTable10.performID,
newTable10.contributorFID, newTable10.nationality,
      newTable10.roleFID, newTable10.contributorID, newTable10.lastName, newTable10.firstName,
newTable10.phoneFID, newTable10.emailFID,
      newTable10.ptID, newTable10.publisherFID, newTable10.titleFID, newTable10.publishYear,
newTable10.ctID, newTable10.categoryFID,
      newTable10.digitalOrPhysical, newTable10.mtID, newTable10.mediaName,
newTable10.publisherName, newTable10.roleName, genre_title.gtID,
      genre_title.genreFID
                  FROM newTable10
                        LEFT JOIN genre_title ON genre_title.titleFID = newTable10.titleID;
DROP TABLE IF EXISTS newTable12;
CREATE TABLE newTable12
      AS SELECT
                  newTable11.titleID, newTable11.typeFID, newTable11.titleName, newTable11.performID,
newTable11.contributorFID, newTable11.nationality,
      newTable11.roleFID, newTable11.contributorID, newTable11.lastName, newTable11.firstName,
newTable11.phoneFID, newTable11.emailFID,
      newTable11.ptID, newTable11.publisherFID, newTable11.titleFID, newTable11.publishYear,
newTable11.ctID, newTable11.categoryFID,
      newTable11.digitalOrPhysical, newTable11.mtID, newTable11.mediaName,
newTable11.publisherName, newTable11.roleName, newTable11.gtID,
      newTable11.genreFID, genre.genreName
                  FROM newTable11
                        LEFT JOIN genre ON genre.genreID = newTable11.genreFID;
DROP TABLE IF EXISTS newTable13;
CREATE TABLE newTable13
      AS SELECT
                  newTable12.titleID, newTable12.typeFID, newTable12.titleName, newTable12.performID,
newTable12.contributorFID, newTable12.nationality,
      newTable12.roleFID, newTable12.contributorID, newTable12.lastName, newTable12.firstName,
newTable12.phoneFID, newTable12.emailFID,
      newTable12.ptID, newTable12.publisherFID, newTable12.titleFID, newTable12.publishYear,
newTable12.ctID, newTable12.categoryFID,
      newTable12.digitalOrPhysical, newTable12.mtID, newTable12.mediaName,
newTable12.publisherName, newTable12.roleName, newTable12.gtID,
      newTable12.genreFID, newTable12.genreName, physical.physicalFormat,
physical.numMediaObjects
                  FROM newTable12
                        LEFT JOIN physical ON physical.titleFID = newTable12.titleID;
DROP TABLE IF EXISTS newTable14;
CREATE TABLE newTable14
      AS SELECT
```

```
                    newTable13.titleID, newTable13.typeFID, newTable13.titleName, newTable13.performID,
newTable13.contributorFID, newTable13.nationality,
      newTable13.roleFID, newTable13.contributorID, newTable13.lastName, newTable13.firstName,
newTable13.phoneFID, newTable13.emailFID,
      newTable13.ptID, newTable13.publisherFID, newTable13.titleFID, newTable13.publishYear,
newTable13.ctID, newTable13.categoryFID,
      newTable13.digitalOrPhysical, newTable13.mtID, newTable13.mediaName,
newTable13.publisherName, newTable13.roleName, newTable13.gtID,
      newTable13.genreFID, newTable13.genreName, newTable13.physicalFormat,
newTable13.numMediaObjects, digital.website, digital.digitalFormat
                    FROM newTable13
                              LEFT JOIN digital ON digital.titleFID = newTable13.titleID;
```

Contributor Reports:

```
DROP TABLE IF EXISTS newTable;
CREATE TABLE newTable
   AS SELECT
      title.titleID, performance.contributorID, title.titleName
      FROM title
         LEFT JOIN performance ON performance.titleFID = title.titleID;

SELECT * FROM newTable WHERE contributorID = 'CB00097';
SELECT
          DISTINCT newTable14.titleName, newTable14.genreName, newTable14.mediaName,
COUNT(DISTINCT(newTable14.physicalFormat)) AS 'Physical Format',
COUNT(newTable14.digitalFormat) AS 'Streaming'
   FROM newTable14
                    WHERE (newTable14.physicalFormat = 'DVD' OR newTable14.physicalFormat =
'BluRay' OR newTable14.digitalFormat = 'Streaming') AND newTable14.mediaName = 'movies' AND
newTable14.genreName = ' Drama';

SELECT
          newTable14.lastName, newTable14.firstName, newTable14.titleName, newTable14.roleName
          FROM newTable14, nondomestic
                    WHERE newTable14.contributorID = nondomestic.contributorID AND
nondomestic.country='England';
```

Media Report:

```
SELECT
          genre.genreName, count(title.titleName) AS 'Counter'
                    FROM title, genre, genre_title
                    WHERE title.titleID = genre_title.titleFID AND genre_title.genreFID = genre.genreID AND
genreName = 'Jazz';
```

Category Report:

```
SELECT
          COUNT(typeID) FROM medialib.audiobook
          WHERE typeID = 'T1';
```