Database Systems (6G5Z1111\_1920\_9Z6)

2CWK30 – Database migration project

C.carr [17105584]

2020

Contents

[Introduction 2](#_Toc36042408)

[Section A – Critiquing the Current Database Design 3](#_Toc36042409)

[Diagram 3](#_Toc36042410)

[Critique 3](#_Toc36042411)

[Section B – New Database Design 5](#_Toc36042412)

[Diagram 5](#_Toc36042413)

[Section C – Database Implementation and Migration 5](#_Toc36042414)

[Section D – SQL Query Creation 5](#_Toc36042415)

[Conclusion 5](#_Toc36042416)

[Bibliography 5](#_Toc36042417)

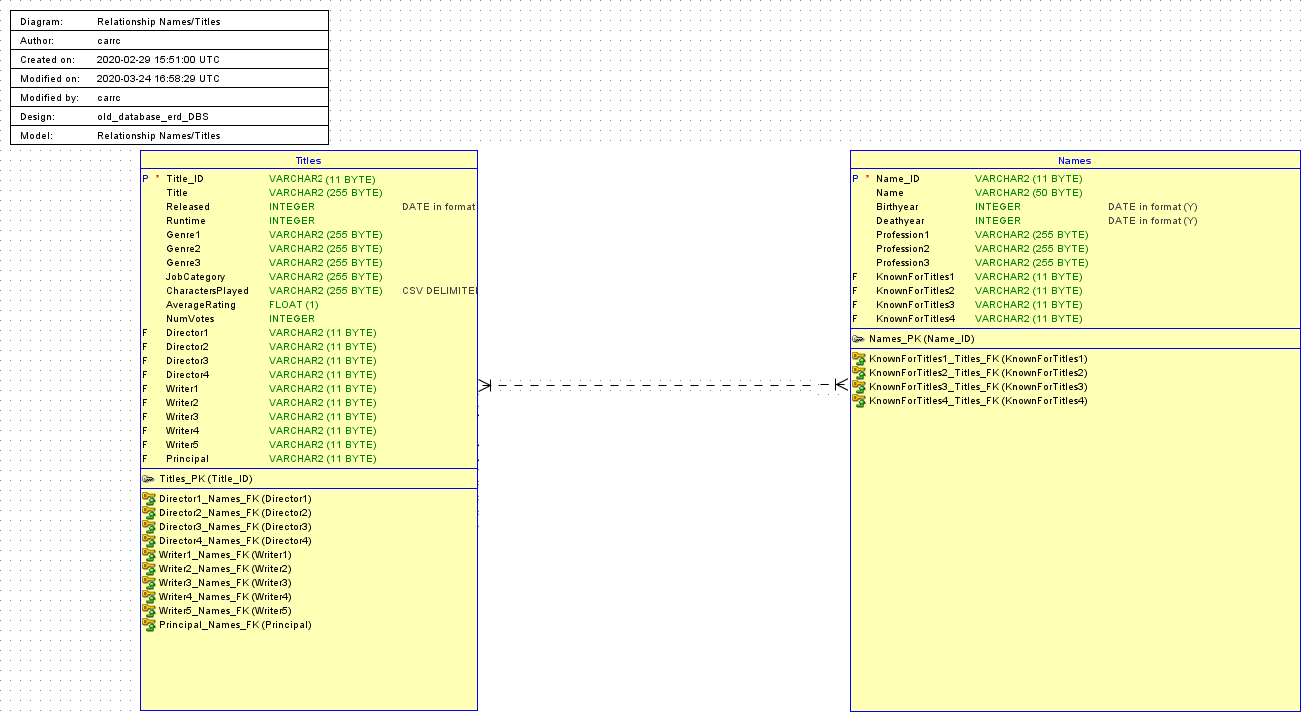
# Introduction

This report focuses on the database system of a media rental company. The current design of the database system will be investigated, analysed, and critiqued. Furthermore, the reasons for investigation are considered upon analysis (mainly, upgradeability). A new database system is presented, described in an Entity Relationship Diagram (ERD) with data normalized up to 5th normal form. Database creation & data migration scripts are included in this report in the format of both plaintext and a SQL file object. Finally, current complex queries which are used on the current system are created to show the data in the new system.

# Section A – Critiquing the Current Database Design

## Diagram

The current database system was analysed and can be shown using general UML notation as shown in Figure 1 below.



[Figure 1, Current Database ERD]

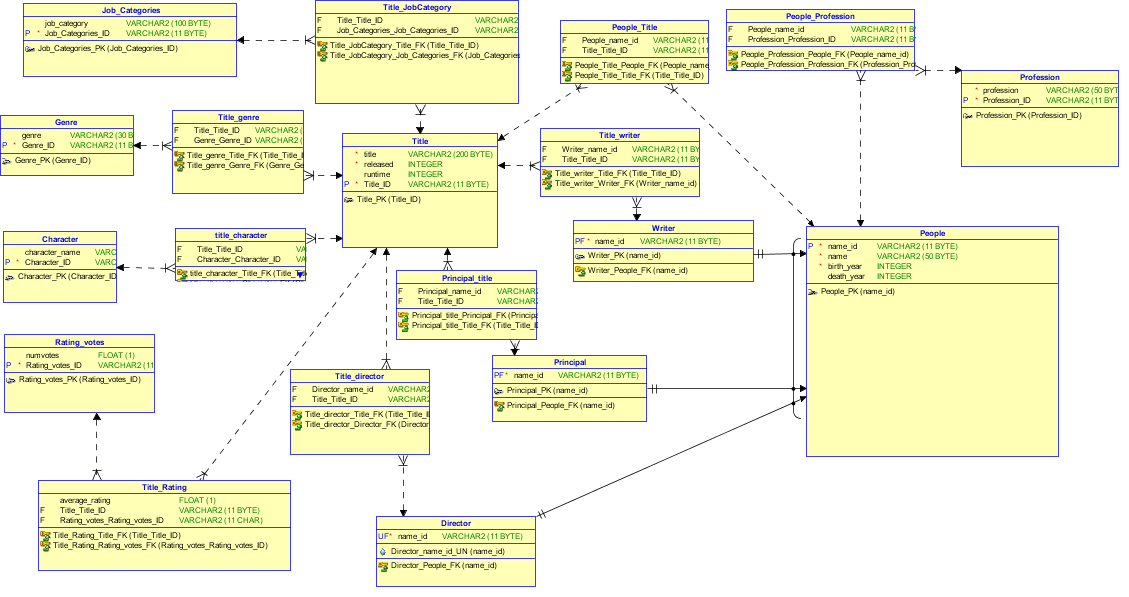
Note that this design includes a many-to-many relationship, which is non-existent in UML. 2 options were available when creating this ERD. Option 1 was to describe the relationship using a non-existent UML notation. Option 2 was to describe each foreign key attributes relationship to the other table, which has the issue of not describing the relationship between entities but describing the relationship between attributes. The first option, a many-to-many relationship was decided upon as the lesser of 2 evils.

## Critique

One issue with the current database is that of null values. In the database, columns allow null values. This results in space being allocated for the data, but nothing being stored there. Hand in hand with this issue is that of the primary keys. In the ERD above, you can see that “Title\_ID” and “Name\_ID” are referenced as the primary keys of the data. This is mostly the case, however, there are duplicate entries for these primary keys, which violates the constraints of a primary key. A primary key should be able to identify a single row in a table, in the current database system there are cases where multiple tuples are returned for a single primary key. This creates a major problem for CRUD (Create, Read, Update, Delete) systems – they’re not sure what to do. Another problem presented with the current database design is upgradability. What if I wanted to make a title with 4 genres? Currently, each title can only be attributed to 3 genres. If I wanted to add more genres to a title, I would need to create a duplicate entry with different genre values. Either that or be stuck with 3 genres. The current database design is not upgradable in multiple attribute sections: Director, Genre, Writer, KnownForTitles, and Profession. Another problem is that the current database does not have atomic data stored in attributes. For example, the “Characters\_played” attribute contains a csv of multiple values. This violates the first rule of normalization and forces whichever system that is interacting with the data to perform logic upon the accessed data. A minor but still existent problem is that of a lack of disparity in the data. The data is not separated much, and this could cause a huge issue with database locking. Database locking is a method by which the DBMS manages concurrent connections. The database “locks” whole columns, rows, tables, or databases. If the DBMS uses a table, column, or database locking system then with the current design there would be many deadlock issues. A problem linked with the lack of disparity in the data is the lack of any indexes. The current tables are massive, containing over 10,000 rows of data. This data is not indexed and therefore is sequentially sequenced by the DBMS whenever it tries to access the data. The more data that is added to the current database, the slower it will get. This would also hold true for an indexed table however the indexed table will still run a lot faster.

# Section B – New Database Design

## Diagram



[Figure 2, Proposed ERD]

## Process, Justification, Evaluation, and Alternatives

The first step in creating the proposed ERD was to normalize the data into 5th normal form.

# Section C – Database Implementation and Migration

# Section D – SQL Query Creation

# Conclusion

# Bibliography

[Figure 1, Current Database ERD] Christopher Carr, Accessed: 24/03/20

[Figure 2, Proposed ERD] Christopher Carr, Accessed: 25/03/20