Databases Project – Spring 2017

Prof. Anastasia Ailamaki

Team No: 40

Names: Léonard Berney, Pierre Fouché, Chenyu Shi

Contents

Contents 1

Deliverable 1 2

Assumptions 2

Entity Relationship Schema 3

Schema 3

Description 3

Relational Schema 4

ER schema to Relational schema 4

DDL 4

General Comments 11

Deliverable 2 12

Assumptions 12

Data Loading 12

Query Implementation 12

Query a: 12

Description of logic: 12

SQL statement 13

Interface 13

Design logic Description 13

Screenshots 13

General Comments 13

Deliverable 3 14

Assumptions 14

Query Implementation 14

Query a: 14

Description of logic: 14

SQL statement 14

Query Analysis 14

Selected Queries (and why) 14

Query 1 14

Query 2 14

Query 3 14

Interface 15

General Comments 15

# Deliverable 1

## Assumptions

We tried to make the smallest amount of assumptions possible.

We made no assumption about any data being unique or non-null, except for the fields referring to IDs which we assumed to be either usable as primary keys or foreign keys. Countries and Languages were also an exception; because the dataset was small we figured out by looking at it the constraints that could be extracted.

One major assumption that we made (see next section) is that if two characters or people have the same name, we will consider them to be the same person or character.

Another assumption we did sometimes was to consider date columns (usually “year\_started” and “year\_ended”) as integer values, after seeing the data, we might have to reconsider this however.

## Entity Relationship Schema

### Schema

### 

### Description

We tried to stay as close to the provided data as possible because it will later be easier to import it in the database if we don’t need to modify it too much. The provided data appeared to come from a database that was not normalized, so we modified if so that it was at least in the first normal form. The “Story” table contained multi-valued columns, the characters and the people who participated in making the story. We created two new entities, “characters” and “people”. A character can be part of one ore multiple stories, whereas a person can contribute in multiple ways to a story, for example by writing the script, drawing, coloring, etc. Other than that, we made “Issue\_reprint” and “Story\_reprint” weak entities because they cannot exist on their own. The other entities were extracted from the provided csv files in a pretty straightforward manner.

## Relational Schema

### ER schema to Relational schema

We created a table for every entity in the ER diagram. We had to create six additional tables to represent the many to many relationships between people, characters and stories.

You may notice that some relationships should have non-null foreign keys and other constraints. We intentionally did not include them in the DDL because we wanted to make it easier to import the data and avoid conflicts as much as possible. Later on we will add these missing constraints and filter out the invalid data.

### DDL

**DROP** **TABLE** **IF** **EXISTS** `brand\_groups`**;**

**CREATE** **TABLE** `brand\_groups` **(**

`id` **int(**11**)** **NOT** **NULL,**

`name` **varchar(**255**)** **DEFAULT** **NULL,**

`year\_began` **int(**11**)** **DEFAULT** **NULL,**

`year\_ended` **int(**11**)** **DEFAULT** **NULL,**

`notes` text**,**

`url` text**,**

`publisher\_id` **int(**11**)** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**KEY** `\_idx` **(**`publisher\_id`**),**

**CONSTRAINT** `brand\_groups\_publishers\_fk` **FOREIGN** **KEY** **(**`publisher\_id`**)** **REFERENCES** `publishers` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `characters`**;**

**CREATE** **TABLE** `characters` **(**

`name` **varchar(**255**)** **NOT** **NULL,**

`id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**UNIQUE** **KEY** `name\_UNIQUE` **(**`name`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `countries`**;**

**CREATE** **TABLE** `countries` **(**

`id` **int(**11**)** **NOT** **NULL,**

`code` **varchar(**255**)** **NOT** **NULL,**

`name` **varchar(**255**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `name\_UNIQUE` **(**`name`**),**

**UNIQUE** **KEY** `code\_UNIQUE` **(**`code`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `indicia\_publishers`**;**

**CREATE** **TABLE** `indicia\_publishers` **(**

`id` **int(**11**)** **NOT** **NULL,**

`name` **varchar(**255**)** **DEFAULT** **NULL,**

`publisher\_id` **int(**11**)** **DEFAULT** **NULL,**

`country\_id` **int(**11**)** **DEFAULT** **NULL,**

`year\_began` **int(**11**)** **DEFAULT** **NULL,**

`year\_ended` **int(**11**)** **DEFAULT** **NULL,**

`is\_surrogate` **tinyint(**4**)** **DEFAULT** **NULL,**

`notes` text**,**

`url` text**,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**KEY** `indicia\_publishers\_countries\_fk\_idx` **(**`country\_id`**),**

**KEY** `indicia\_publishers\_publishers\_fk\_idx` **(**`publisher\_id`**),**

**CONSTRAINT** `indicia\_publishers\_countries\_fk` **FOREIGN** **KEY** **(**`country\_id`**)** **REFERENCES** `countries` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `indicia\_publishers\_publishers\_fk` **FOREIGN** **KEY** **(**`publisher\_id`**)** **REFERENCES** `publishers` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `issue\_reprints`**;**

**CREATE** **TABLE** `issue\_reprints` **(**

`id` **int(**11**)** **NOT** **NULL,**

`origin\_issue\_id` **int(**11**)** **DEFAULT** **NULL,**

`target\_issue\_id` **int(**11**)** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**KEY** `issue\_reprints\_target\_issues\_fk\_idx` **(**`target\_issue\_id`**),**

**KEY** `issue\_reprints\_origin\_issues\_fk\_idx` **(**`origin\_issue\_id`**),**

**CONSTRAINT** `issue\_reprints\_origin\_issues\_fk` **FOREIGN** **KEY** **(**`origin\_issue\_id`**)** **REFERENCES** `issues` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `issue\_reprints\_target\_issues\_fk` **FOREIGN** **KEY** **(**`target\_issue\_id`**)** **REFERENCES** `issues` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `issues`**;**

**CREATE** **TABLE** `issues` **(**

`id` **int(**11**)** **NOT** **NULL,**

`**number**` **varchar(**255**)** **DEFAULT** **NULL,**

`series\_id` **int(**11**)** **DEFAULT** **NULL,**

`indicia\_publisher\_id` **int(**11**)** **DEFAULT** **NULL,**

`publication\_date` **varchar(**255**)** **DEFAULT** **NULL,**

`price` **varchar(**255**)** **DEFAULT** **NULL,**

`page\_count` **int(**11**)** **DEFAULT** **NULL,**

`indicia\_frequency` **varchar(**255**)** **DEFAULT** **NULL,**

`editing` text**,**

`notes` text**,**

`isbn` **varchar(**255**)** **DEFAULT** **NULL,**

`valid\_isbn` **varchar(**255**)** **DEFAULT** **NULL,**

`barcode` **varchar(**255**)** **DEFAULT** **NULL,**

`title` **varchar(**255**)** **DEFAULT** **NULL,**

`on\_sale\_date` **varchar(**255**)** **DEFAULT** **NULL,**

`rating` **varchar(**255**)** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**KEY** `issues\_indicia\_publishers\_fk` **(**`indicia\_publisher\_id`**),**

**KEY** `issues\_series\_fk\_idx` **(**`series\_id`**),**

**CONSTRAINT** `issues\_indicia\_publishers\_fk` **FOREIGN** **KEY** **(**`indicia\_publisher\_id`**)** **REFERENCES** `indicia\_publishers` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `issues\_series\_fk` **FOREIGN** **KEY** **(**`series\_id`**)** **REFERENCES** `series` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `languages`**;**

**CREATE** **TABLE** `languages` **(**

`id` **int(**11**)** **NOT** **NULL,**

`code` **varchar(**255**)** **NOT** **NULL,**

`name` **varchar(**255**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `name\_UNIQUE` **(**`name`**),**

**UNIQUE** **KEY** `code\_UNIQUE` **(**`code`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `people`**;**

**CREATE** **TABLE** `people` **(**

`first\_name` **varchar(**255**)** **DEFAULT** **NULL,**

`last\_name` **varchar(**255**)** **DEFAULT** **NULL,**

`id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `publishers`**;**

**CREATE** **TABLE** `publishers` **(**

`id` **int(**11**)** **NOT** **NULL,**

`name` **varchar(**255**)** **DEFAULT** **NULL,**

`country\_id` **int(**11**)** **DEFAULT** **NULL,**

`year\_began` **int(**11**)** **DEFAULT** **NULL,**

`year\_ended` **int(**11**)** **DEFAULT** **NULL,**

`notes` text**,**

`url` text**,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**KEY** `publishers\_countries\_fk\_idx` **(**`country\_id`**),**

**CONSTRAINT** `publishers\_countries\_fk` **FOREIGN** **KEY** **(**`country\_id`**)** **REFERENCES** `countries` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `series`**;**

**CREATE** **TABLE** `series` **(**

`id` **int(**11**)** **NOT** **NULL,**

`name` **varchar(**255**)** **DEFAULT** **NULL,**

`**format**` **varchar(**255**)** **DEFAULT** **NULL,**

`year\_began` **int(**11**)** **DEFAULT** **NULL,**

`year\_ended` **int(**11**)** **DEFAULT** **NULL,**

`publication\_dates` **varchar(**255**)** **DEFAULT** **NULL,**

`first\_issue\_id` **int(**11**)** **DEFAULT** **NULL,**

`last\_issue\_id` **int(**11**)** **DEFAULT** **NULL,**

`publisher\_id` **int(**11**)** **DEFAULT** **NULL,**

`country\_id` **int(**11**)** **DEFAULT** **NULL,**

`language\_id` **int(**11**)** **DEFAULT** **NULL,**

`notes` text**,**

`color` **varchar(**255**)** **DEFAULT** **NULL,**

`dimensions` **varchar(**255**)** **DEFAULT** **NULL,**

`paper\_stock` **varchar(**255**)** **DEFAULT** **NULL,**

`binding` **varchar(**255**)** **DEFAULT** **NULL,**

`publishing\_format` **varchar(**255**)** **DEFAULT** **NULL,**

`publication\_type\_id` **int(**11**)** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**KEY** `series\_first\_issues\_fk\_idx` **(**`first\_issue\_id`**),**

**KEY** `series\_last\_issues\_fk\_idx` **(**`last\_issue\_id`**),**

**KEY** `series\_publishers\_fk\_idx` **(**`publisher\_id`**),**

**KEY** `series\_countries\_fk\_idx` **(**`country\_id`**),**

**KEY** `series\_languages\_fk\_idx` **(**`language\_id`**),**

**CONSTRAINT** `series\_countries\_fk` **FOREIGN** **KEY** **(**`country\_id`**)** **REFERENCES** `countries` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `series\_first\_issues\_fk` **FOREIGN** **KEY** **(**`first\_issue\_id`**)** **REFERENCES** `issues` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `series\_languages\_fk` **FOREIGN** **KEY** **(**`language\_id`**)** **REFERENCES** `languages` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `series\_last\_issues\_fk` **FOREIGN** **KEY** **(**`last\_issue\_id`**)** **REFERENCES** `issues` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `series\_publishers\_fk` **FOREIGN** **KEY** **(**`publisher\_id`**)** **REFERENCES** `publishers` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `series\_publication\_types`**;**

**CREATE** **TABLE** `series\_publication\_types` **(**

`id` **int(**11**)** **NOT** **NULL,**

`name` **varchar(**255**)** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `stories`**;**

**CREATE** **TABLE** `stories` **(**

`id` **int(**11**)** **NOT** **NULL,**

`title` **varchar(**255**)** **DEFAULT** **NULL,**

`feature` **varchar(**255**)** **DEFAULT** **NULL,**

`issue\_id` **int(**11**)** **DEFAULT** **NULL,**

`script\_id` **int(**11**)** **DEFAULT** **NULL,**

`pencils\_id` **int(**11**)** **DEFAULT** **NULL,**

`inks\_id` **int(**11**)** **DEFAULT** **NULL,**

`colors\_id` **int(**11**)** **DEFAULT** **NULL,**

`letters\_id` **int(**11**)** **DEFAULT** **NULL,**

`editing` text**,**

`genre` **varchar(**255**)** **DEFAULT** **NULL,**

`characters\_id` **int(**11**)** **DEFAULT** **NULL,**

`synopsis` text**,**

`reprint\_notes` text**,**

`notes` text**,**

`type\_id` **int(**11**)** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**KEY** `stories\_issues\_fk\_idx` **(**`issue\_id`**),**

**KEY** `stories\_script\_fk\_idx` **(**`script\_id`**),**

**KEY** `stories\_pencils\_fk\_idx` **(**`pencils\_id`**),**

**KEY** `stories\_inks\_fk\_idx` **(**`inks\_id`**),**

**KEY** `stories\_colors\_fk\_idx` **(**`colors\_id`**),**

**KEY** `stories\_letters\_fk\_idx` **(**`letters\_id`**),**

**KEY** `stories\_characters\_fk\_idx` **(**`characters\_id`**),**

**KEY** `stories\_story\_types\_fk\_idx` **(**`type\_id`**),**

**CONSTRAINT** `stories\_characters\_fk` **FOREIGN** **KEY** **(**`characters\_id`**)** **REFERENCES** `stories\_characters` **(**`story\_id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `stories\_colors\_fk` **FOREIGN** **KEY** **(**`colors\_id`**)** **REFERENCES** `stories\_colors` **(**`story\_id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `stories\_inks\_fk` **FOREIGN** **KEY** **(**`inks\_id`**)** **REFERENCES** `stories\_inks` **(**`story\_id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `stories\_issues\_fk` **FOREIGN** **KEY** **(**`issue\_id`**)** **REFERENCES** `issues` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `stories\_letters\_fk` **FOREIGN** **KEY** **(**`letters\_id`**)** **REFERENCES** `stories\_letters` **(**`story\_id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `stories\_pencils\_fk` **FOREIGN** **KEY** **(**`pencils\_id`**)** **REFERENCES** `stories\_pencils` **(**`story\_id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `stories\_script\_fk` **FOREIGN** **KEY** **(**`script\_id`**)** **REFERENCES** `stories\_script` **(**`story\_id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `stories\_story\_types\_fk` **FOREIGN** **KEY** **(**`type\_id`**)** **REFERENCES** `story\_types` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `stories\_characters`**;**

**CREATE** **TABLE** `stories\_characters` **(**

`story\_id` **int(**11**)** **NOT** **NULL,**

`character\_id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`story\_id`**,**`character\_id`**),**

**KEY** `stories\_characters\_characters\_fk\_idx` **(**`character\_id`**),**

**CONSTRAINT** `stories\_characters\_characters\_fk` **FOREIGN** **KEY** **(**`character\_id`**)** **REFERENCES** `characters` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `stories\_colors`**;**

**CREATE** **TABLE** `stories\_colors` **(**

`story\_id` **int(**11**)** **NOT** **NULL,**

`person\_id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`story\_id`**,**`person\_id`**),**

**KEY** `stories\_colors\_people\_fk\_idx` **(**`person\_id`**),**

**CONSTRAINT** `stories\_colors\_people\_fk` **FOREIGN** **KEY** **(**`person\_id`**)** **REFERENCES** `people` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `stories\_inks`**;**

**CREATE** **TABLE** `stories\_inks` **(**

`story\_id` **int(**11**)** **NOT** **NULL,**

`person\_id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`story\_id`**,**`person\_id`**),**

**KEY** `stories\_inks\_people\_fk\_idx` **(**`person\_id`**),**

**CONSTRAINT** `stories\_inks\_people\_fk` **FOREIGN** **KEY** **(**`person\_id`**)** **REFERENCES** `people` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `stories\_letters`**;**

**CREATE** **TABLE** `stories\_letters` **(**

`story\_id` **int(**11**)** **NOT** **NULL,**

`person\_id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`story\_id`**,**`person\_id`**),**

**KEY** `stories\_letters\_people\_fk\_idx` **(**`person\_id`**),**

**CONSTRAINT** `stories\_letters\_people\_fk` **FOREIGN** **KEY** **(**`person\_id`**)** **REFERENCES** `people` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `stories\_pencils`**;**

**CREATE** **TABLE** `stories\_pencils` **(**

`story\_id` **int(**11**)** **NOT** **NULL,**

`person\_id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`story\_id`**,**`person\_id`**),**

**KEY** `stories\_pencils\_people\_fk\_idx` **(**`person\_id`**),**

**CONSTRAINT** `stories\_pencils\_people\_fk` **FOREIGN** **KEY** **(**`person\_id`**)** **REFERENCES** `people` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `stories\_script`**;**

**CREATE** **TABLE** `stories\_script` **(**

`story\_id` **int(**11**)** **NOT** **NULL,**

`person\_id` **int(**11**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`story\_id`**,**`person\_id`**),**

**KEY** `stories\_script\_people\_fk\_idx` **(**`person\_id`**),**

**CONSTRAINT** `stories\_script\_people\_fk` **FOREIGN** **KEY** **(**`person\_id`**)** **REFERENCES** `people` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `story\_reprints`**;**

**CREATE** **TABLE** `story\_reprints` **(**

`id` **int(**11**)** **NOT** **NULL,**

`origin\_id` **int(**11**)** **DEFAULT** **NULL,**

`target\_id` **int(**11**)** **DEFAULT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**),**

**KEY** `story\_reprints\_origin\_stories\_fk\_idx` **(**`origin\_id`**),**

**KEY** `story\_reprints\_target\_stories\_fk\_idx` **(**`target\_id`**),**

**CONSTRAINT** `story\_reprints\_origin\_stories\_fk` **FOREIGN** **KEY** **(**`origin\_id`**)** **REFERENCES** `stories` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION**,**

**CONSTRAINT** `story\_reprints\_target\_stories\_fk` **FOREIGN** **KEY** **(**`target\_id`**)** **REFERENCES** `stories` **(**`id`**)** **ON** **DELETE** **NO** ACTION **ON** **UPDATE** **NO** ACTION

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

**DROP** **TABLE** **IF** **EXISTS** `story\_types`**;**

**CREATE** **TABLE** `story\_types` **(**

`id` **int(**11**)** **NOT** **NULL,**

`name` **varchar(**255**)** **NOT** **NULL,**

**PRIMARY** **KEY** **(**`id`**),**

**UNIQUE** **KEY** `name\_UNIQUE` **(**`name`**),**

**UNIQUE** **KEY** `id\_UNIQUE` **(**`id`**)**

**)** ENGINE**=**InnoDB **DEFAULT** CHARSET**=**utf8**;**

## General Comments

We decided to use MySQL as a DBMS because some of us already had experience with it.

We used three different column types in our tables. The INT type was used for the indexes and the integer values such as years (full dates were represented using VARCHAR). The VARCHAR type was used for almost any text type field. The size of 255 was arbitrarily chosen and when it was too small, we instead used the TEXT type.

Since we have not yet started importing the data, we might have to modify some tables to accommodate for larger text fields for example.

We split the work in two parts. Léonard did the DDL, Pierre and Chenyu did the ER diagram.

# Deliverable 2

## Assumptions

The assumptions we did in the first deliverable were pretty good and we did not have to make new ones, or revise existing ones.

## Data Loading

Data loading was done in two parts. The first part consisted of parsing and cleaning the data, while the second part took care of importing the data in the database.

We cleaned the data using Python and Powershell scripts.

We first applied the same treatment to all CSV files; we inserted “\N” in the empty fields so that MySQL would recognize them as NULL values. We also replaced fields containing only a “?” or “NULL” by “\N”.

For the date fields that were some kind of string, we tried to extract a year from them in order to be able to work with integers. This was done by detecting a pattern of four numbers and assuming it to represent a year.

Extracting the list of characters from the story csv required quite a bit of parsing. According to the GCD official documentation for this field, a character or group of characters can be followed by “[]” in order to indicate an alternate identity to the character or to list the members of a group. We decided that it made more sense to extract every character as well as their alternate identities in separate characters. We had multiple cases where a single character for example, “The Flash” had multiple other identities, and by separating the data the way we did, we were able to keep them. We also removed comments written in parenthesis and other types of comments, usually written in capital letters.

We were able to apply a similar process to extract the data for the different tables from the story csv.

The process of loading the data was pretty straightforward because we previously created a csv for each table so we could just use the “load data infile” command from MySQL.

## Query Implementation

### Query a:

#### Description of logic:

Because there might be multiple brand groups with the highest count, we need to run twice the query. Once to find the max and once to filter the brand groups that have the max number of Belgian publishers.

#### SQL statement

**SELECT**

brand\_groups**.**name

**FROM**

brand\_groups

**INNER** **JOIN**

**(SELECT**

publisher\_id

**FROM**

indicia\_publishers

**WHERE**

country\_id **=** **(SELECT**

id

**FROM**

countries

**WHERE**

name **=** 'belgium'**)**

**GROUP** **BY** publisher\_id

**HAVING** **COUNT(\*)** **=** **(SELECT**

**MAX(**c**)**

**FROM**

**(SELECT**

**COUNT(\*)** **AS** c**,** id

**FROM**

indicia\_publishers

**WHERE**

country\_id **=** **(SELECT**

id

**FROM**

countries

**WHERE**

name **=** 'belgium'**)**

**GROUP** **BY** publisher\_id**)** **AS** b**))** **AS** a **ON** brand\_groups**.**publisher\_id **=** a**.**publisher\_id**;**

### Query b:

#### Description of logic:

Simple select + where and a subquery to find the id corresponding to Denmark.

#### SQL statement

**SELECT** **DISTINCT**

publishers**.**id**,** publishers**.**name

**FROM**

publishers

**INNER** **JOIN**

**(SELECT**

publisher\_id

**FROM**

series

**WHERE**

country\_id **=** **(SELECT**

id

**FROM**

countries

**WHERE**

### name **=** 'denmark'**))** **AS** table\_2 **ON** publishers**.**id **=** table\_2**.**publisher\_id

### Query c:

#### Description of logic:

Find the id corresponding to Switzerland and the id corresponding to magazine. Then select the series corresponding to both.

#### SQL statement

**SELECT**

name

**FROM**

series

**WHERE**

country\_id **=** **(SELECT**

id

**FROM**

countries

**WHERE**

name **=** 'switzerland'**)**

**AND** publication\_type\_id **=** **(SELECT**

id

**FROM**

series\_publication\_types

**WHERE**

name **=** 'magazine'**);**

### Query d:

#### Description of logic:

Select the issues where the publication date is greater or equal to 1990 then order in ascending order

#### SQL statement

**SELECT**

publication\_date**,** **COUNT(\*)** **number**

**FROM**

issues

**WHERE**

publication\_date **>=** 1990

**GROUP** **BY** publication\_date

### **ORDER** **BY** publication\_date **ASC**

### Query e:

#### Description of logic:

We first find the publishers whose name resembles “DC Comics” then we join the result of this subquery with series and count the size of the result.

#### SQL statement

**SELECT**

table1**.**name**,** **COUNT(**id**)** **AS** **count**

**FROM**

series

**INNER** **JOIN**

**(SELECT**

publisher\_id**,** name

**FROM**

indicia\_publishers

**WHERE**

### indicia\_publishers**.**name **LIKE** '%DC comics%'**)** **AS** table1 **ON** series**.**publisher\_id **=** table1**.**publisher\_id

### Query f:

#### Description of logic:

We find the 10 most reprinted stories by finding the different origin\_id in story\_reprint and by grouping by the origin\_id and ordering by count descending. We then apply a limit 10 to have only the top 10 stories.

#### SQL statement

**SELECT**

title

**FROM**

stories

**WHERE**

id **IN** **(SELECT**

origin\_id

**FROM**

**(SELECT** **DISTINCT**

origin\_id**,** **COUNT(**origin\_id**)**

**FROM**

story\_reprints

**GROUP** **BY** origin\_id

**ORDER** **BY** **COUNT(**origin\_id**)** **DESC**

### **LIMIT** 10**)** **as** a**)**

### Query g:

#### Description of logic:

We used multiple joins, the trick was to join with people and in the join condition, assert that the ids from stories\_script, stories\_pencils and stories\_colors were equal.

#### SQL statement

**SELECT** **DISTINCT**

people**.**name

**FROM**

stories

**INNER** **JOIN**

stories\_script **ON** stories**.**id **=** stories\_script**.**story\_id

**INNER** **JOIN**

stories\_pencils **ON** stories**.**id **=** stories\_pencils**.**story\_id

**INNER** **JOIN**

stories\_colors **ON** stories**.**id **=** stories\_colors**.**story\_id

**INNER** **JOIN**

people **ON** people**.**name **=** stories\_script**.**person\_name

**AND** people**.**name **=** stories\_pencils**.**person\_name

**AND** people**.**name **=** stories\_colors**.**person\_name

### Query h:

#### Description of logic:

The wording of the query was difficult to understand, we might not have done what was asked. We selected all stories where Batman was not in the featured characters but was in the characters of the story. The stories also had to not be reprints, so we added a clause in the where that the id of the story should not be in the origin\_id of the story\_reprints table.

#### SQL statement

**SELECT**

title

**FROM**

stories

**INNER** **JOIN**

stories\_feature **on** stories**.**id **=** stories\_feature**.**story\_id

**WHERE**

stories\_feature**.**character\_id **<>** 'batman'

**AND** character\_id **IN** **(SELECT**

stories\_characters**.**character\_name

**FROM**

stories\_characters

**WHERE**

stories\_characters**.**character\_name **=** 'batman'**)**

**AND** story\_id **NOT** **IN** **(SELECT**

origin\_id

**FROM**

## story\_reprints)

## Interface

### Design logic Description

The application will be web based with a backend in PHP. We will use JTables to display the data in a searchable and sortable table.

### Screenshots

We haven’t yet had the opportunity to work on the front end, so no screenshot is available.

## General Comments

Cleaning the data to make it somewhat useable was a big amount of work and we might need to go back to it later to make some minor adjustments. Because we have a large number of tables, it also took quite some time to write all the SQL scripts to load the data from the CSV and to add the missing foreign key constraints.

The queries were written by Chenyu. Pierre and Léonard did the data cleanup and importation and made the required modifications to the last deliverable.

Here are some comments concerning the modification of the ER diagram:

* We deleted the ternary relationship between Publisher, Indicia\_Publisher and Country (if they come from different countries).
* The reprint relationships are now Many-to-Many relationships according to the data
* A series is published for exactly one country, has only one publication type and is written in exactly one language as it is defined in the data
* We separated 'genre' and 'editing' as two entity sets linked with the story entity through a relationship
* We realize that a story can only be published in one issue and can only have one type that is why we decide to keep 'issue\_id' and 'type\_id' as attributes of the story entity and not create separate ones.
* We represent 'first\_issue' and 'last\_issue' as separate 1-to-1 relationships.
* We represent 'featured character' and 'all other characters' using different relationships. Thus, different tables

# Deliverable 3

# Assumptions

No additional assumptions on the data were made since previous deliverables. When the subject was unclear, we wrote down what we understood about it.

## Query Implementation

### Query a:

#### Description of logic:

We first need to find the most recurrent story type. We then find all stories that do not have this type. We can then run the same query again to find the greatest amount of issues that have these stories. Alter this we can simply join the result with series, keeping only the ones with the highest count of issues as calculated before.

#### SQL statement

**SELECT**

series**.**name

**FROM**

**(SELECT**

series\_id

**FROM**

stories

**INNER** **JOIN** issues **ON** issues**.**id **=** stories**.**issue\_id

**WHERE**

stories**.**type\_id **<>** **(SELECT**

stories**.**type\_id

**FROM**

stories

**GROUP** **BY** stories**.**type\_id

**ORDER** **BY** **COUNT(\*)** **DESC**

**LIMIT** 1**)**

**GROUP** **BY** series\_id

**HAVING** **COUNT(\*)** **=** **(SELECT**

**MAX(**c**)**

**FROM**

**(SELECT**

**COUNT(\*)** **AS** c

**FROM**

**(SELECT**

stories**.**issue\_id

**FROM**

stories

**WHERE**

stories**.**type\_id **<>** **(SELECT**

stories**.**type\_id

**FROM**

stories

**GROUP** **BY** stories**.**type\_id

**ORDER** **BY** **COUNT(\*)** **DESC**

**LIMIT** 1**))** **AS** a

**INNER** **JOIN** issues **ON** issues**.**id **=** a**.**issue\_id

**GROUP** **BY** issues**.**series\_id**)** **AS** b**))** **AS** a

**INNER** **JOIN**

series **ON** series**.**id **=** a**.**series\_id

### Query b:

#### Description of logic:

Find the number of different series publication types, then for each series find the number of publication types. Now select the publishers that have more than one series with all types.

#### SQL statement

**SELECT**

publishers**.**name

**FROM**

publishers

**INNER** **JOIN**

series **ON** publishers**.**id **=** series**.**publisher\_id

**WHERE**

series**.**name **IN** **(SELECT**

name

**FROM**

**(SELECT** **DISTINCT**

series**.**name**,** series**.**publication\_type\_id

**FROM**

series

**WHERE**

publication\_type\_id **IS** **NOT** **NULL**

**GROUP** **BY** series**.**name **,** series**.**publication\_type\_id**)** **AS** a

**GROUP** **BY** name

**HAVING** **COUNT(\*)** **=** **(SELECT**

**COUNT(\*)**

**FROM**

series\_publication\_types**))**

**GROUP** **BY** publishers**.**id

### **HAVING** **COUNT(\*)** **>** 1

### Query c:

#### Description of logic:

Simply join all the required tables, keeping only the stories by Alan Moore, then group by the characters names and keep only the top 10.

#### SQL statement

**SELECT**

stories\_characters**.**character\_name

**FROM**

stories

**INNER** **JOIN**

stories\_script **ON** stories**.**id **=** stories\_script**.**story\_id

**INNER** **JOIN**

stories\_characters **ON** stories**.**id **=** stories\_characters**.**story\_id

**INNER** **JOIN**

story\_reprints **ON** story\_reprints**.**origin\_id **=** stories**.**id

**WHERE**

stories\_script**.**person\_name **=** 'Alan Moore'

**GROUP** **BY** stories\_characters**.**character\_name

**ORDER** **BY** **COUNT(\*)** **DESC**

### **LIMIT** 10

### Query d:

#### Description of logic:

The trick is to join stories\_pencils and stories\_genres together in order to keep only stories that have both the same person who did the pencilwork and the script.

#### SQL statement

**SELECT** **DISTINCT**

stories\_script**.**person\_name

**FROM**

stories\_script

**INNER** **JOIN**

stories **ON** stories**.**id **=** stories\_script**.**story\_id

**INNER** **JOIN**

stories\_genres **ON** stories**.**id **=** stories\_genres**.**story\_id

**INNER** **JOIN**

stories\_pencils **ON** stories\_script**.**story\_id **=** stories\_pencils**.**story\_id

**WHERE**

### stories\_genres**.**genre\_name **=** 'nature'

### Query e:

#### Description of logic:

We first find the top 10 publishers. We can then join the result of this query with languages and publishers. The only thing left is then to group by publisher and concatenate the languages keeping only the first 3.

#### SQL statement

**SELECT**

publishers**.**name**,**

**SUBSTRING\_INDEX(**GROUP\_CONCAT**(DISTINCT** languages**.**name

SEPARATOR ', '**),**

','**,**

3**)** **AS** languages

**FROM**

publishers

**INNER** **JOIN**

**(SELECT**

publishers**.**id **AS** pid

**FROM**

publishers

**INNER** **JOIN** series **ON** series**.**publisher\_id **=** publishers**.**id

**GROUP** **BY** publisher\_id

**ORDER** **BY** **COUNT(\*)** **DESC**

**LIMIT** 10**)** **AS** a **ON** a**.**pid **=** publishers**.**id

**INNER** **JOIN**

series **ON** series**.**publisher\_id **=** publishers**.**id

**INNER** **JOIN**

languages **ON** languages**.**id **=** series**.**language\_id

### **GROUP** **BY** publisher\_id

### Query f:

#### Description of logic:

By “original stories” we understood “stories excluding their reprints”. We joined the required tables and filtered out the stories that were reprints then grouped by languages and kept only the rows with at least 10000 values.

#### SQL statement

**SELECT**

languages**.**name**,** **COUNT(\*)** **AS** **number**

**FROM**

stories

**INNER** **JOIN**

issues **ON** issues**.**id **=** stories**.**issue\_id

**INNER** **JOIN**

series **ON** issues**.**series\_id **=** series**.**id

**INNER** **JOIN**

languages **ON** series**.**language\_id **=** languages**.**id

**INNER** **JOIN**

series\_publication\_types **ON** series**.**publication\_type\_id **=** series\_publication\_types**.**id

**WHERE**

series\_publication\_types**.**name **=** 'magazine'

**AND** stories**.**id **NOT** **IN** **(SELECT**

story\_reprints**.**target\_id

**FROM**

story\_reprints**)**

**GROUP** **BY** languages**.**id

### **HAVING** **COUNT(\*)** **>** 10000

### Query g:

#### Description of logic:

First find all story types that were published in an Italian magazine, then find the story types that are not in this subquery.

#### SQL statement

**SELECT**

story\_types**.**name

**FROM**

story\_types

**WHERE**

id **NOT** **IN** **(SELECT**

story\_types**.**id

**FROM**

stories

**INNER** **JOIN**

issues **ON** issues**.**id **=** stories**.**issue\_id

**INNER** **JOIN**

series **ON** issues**.**series\_id **=** series**.**id

**INNER** **JOIN**

languages **ON** series**.**language\_id **=** languages**.**id

**INNER** **JOIN**

series\_publication\_types **ON** series**.**publication\_type\_id **=** series\_publication\_types**.**id

**INNER** **JOIN**

story\_types **ON** story\_types**.**id **=** stories**.**type\_id

**WHERE**

languages**.**name **=** 'italian'

### **AND** series\_publication\_types**.**name **=** 'magazine'**)**

### Query h:

#### Description of logic:

We first find the unique tuples (stories\_script.person\_name , indicia\_publishers.name). We now have tuples containing for each person the indicia publishers that person worked for. Because we are not interested in the name of the indicia publishers but only the count, we can group by person name and keep only those with a count greater than one. The rest is just a matter of joining the correct tables together.

#### SQL statement

**SELECT** **DISTINCT**

stories\_script**.**person\_name

**FROM**

stories\_script

**INNER** **JOIN**

stories **ON** stories**.**id **=** stories\_script**.**story\_id

**INNER** **JOIN**

story\_types **ON** story\_types**.**id **=** stories**.**type\_id

**WHERE**

story\_types**.**name **=** 'cartoon'

**AND** stories\_script**.**person\_name **IN** **(SELECT**

**\***

**FROM**

**(SELECT**

stories\_script**.**person\_name

**FROM**

indicia\_publishers

**INNER** **JOIN** publishers **ON** indicia\_publishers**.**publisher\_id **=** publishers**.**id

**INNER** **JOIN** series **ON** series**.**publisher\_id **=** publishers**.**id

**INNER** **JOIN** issues **ON** issues**.**series\_id **=** series**.**id

**INNER** **JOIN** stories **ON** stories**.**issue\_id **=** issues**.**id

**INNER** **JOIN** stories\_script **ON** stories**.**id **=** stories\_script**.**story\_id

**GROUP** **BY** stories\_script**.**person\_name **,** indicia\_publishers**.**name**)** **AS** a

**GROUP** **BY** person\_name

### **HAVING** **COUNT(\*)** **>** 1**)**

### Query i:

#### Description of logic:

This query is just a matter of doing a couple of joins and grouping by brand groups names and taking the 10 with the highest count.

#### SQL statement

**SELECT**

brand\_groups**.**name

**FROM**

brand\_groups

**INNER** **JOIN**

publishers **ON** publishers**.**id **=** brand\_groups**.**publisher\_id

**INNER** **JOIN**

indicia\_publishers **ON** publishers**.**id **=** indicia\_publishers**.**publisher\_id

**GROUP** **BY** brand\_groups**.**name

**ORDER** **BY** **COUNT(\*)** **DESC**

**LIMIT** 10

### Query j:

#### Description of logic:

This query is a matter of doing a join and grouping by indicia publishers ids and use the average function.

#### SQL statement

select avg(series\_length), indicia\_publishers

from(

select (series.year\_ended-series.year\_began) as series\_length, indicia\_publishers.id as indicia\_publishers

from series

inner join indicia\_publishers on indicia\_publishers.publisher\_id = series.publisher\_id

)

group by indicia\_publishers

### Query k:

#### Description of logic:

This query is a matter of doing a join and grouping, using having clause to set the conditions with aggregate functions.

#### SQL statement

select indicia\_publisher\_id

from

(

select count(id) as issue\_number, indicia\_publisher\_id, series\_id

from (

select issues.series\_id, issues.indicia\_publisher\_id, issues.id

from issues

inner join series on issues.series\_id = series.id

)

group by series\_id

)

group by indicia\_publisher\_id

having count(issue\_number) =1

order by count(series\_id) desc

limit 10

### Query l:

#### Description of logic:

This query is a matter of doing a join and grouping, and using order by clause.

#### SQL statement

select table1.writer\_number, issues.indicia\_publisher\_id

from

(

select count(stories\_script.person\_name) as writer\_number, stories\_script.story\_id, stories.issue\_id

from stories\_script as table1

inner join stories on stories\_script.story\_id = stories.id

group by story\_id

)

inner join issues on issues.id = table1.issue\_id

order by writer\_number desc

limit 10

### Query n:

#### Description of logic:

This query is a matter of doing a join and grouping, using order by clause and limit.

#### SQL statement

select count(issues.id) as issue\_number, series.id as series

from issues

inner join series on issues.series\_id = series.id

group by series

order by issue\_number

limit 5

# Interface

### Design logic Description

<Describe the general logic of your design as well as the technology you decided to use>

### Screenshots

<Provide some initial screen shots of your interface>

# General Comments

Léonard did the queries A through I of deliverable 3 and fixed the queries from deliverable 2. Chenyu did the queries J to O of deliverable 3.