# Databases Project – Spring 2017

Prof. Anastasia Ailamaki

Team No: 15

Names: Colin Branca, Jules Courtois, Yoan Martin

Contents

[Deliverable 1 2](#__RefHeading__142_1358239881)

[Assumptions 2](#__RefHeading__144_1358239881)

[Entity Relationship Schema 2](#__RefHeading__146_1358239881)

[Schema 2](#__RefHeading__148_1358239881)

[Description 2](#__RefHeading__150_1358239881)

[Relational Schema 3](#__RefHeading__152_1358239881)

[ER schema to Relational schema 3](#__RefHeading__154_1358239881)

[DDL 4](#__RefHeading__156_1358239881)

[General Comments 6](#__RefHeading__158_1358239881)

# 

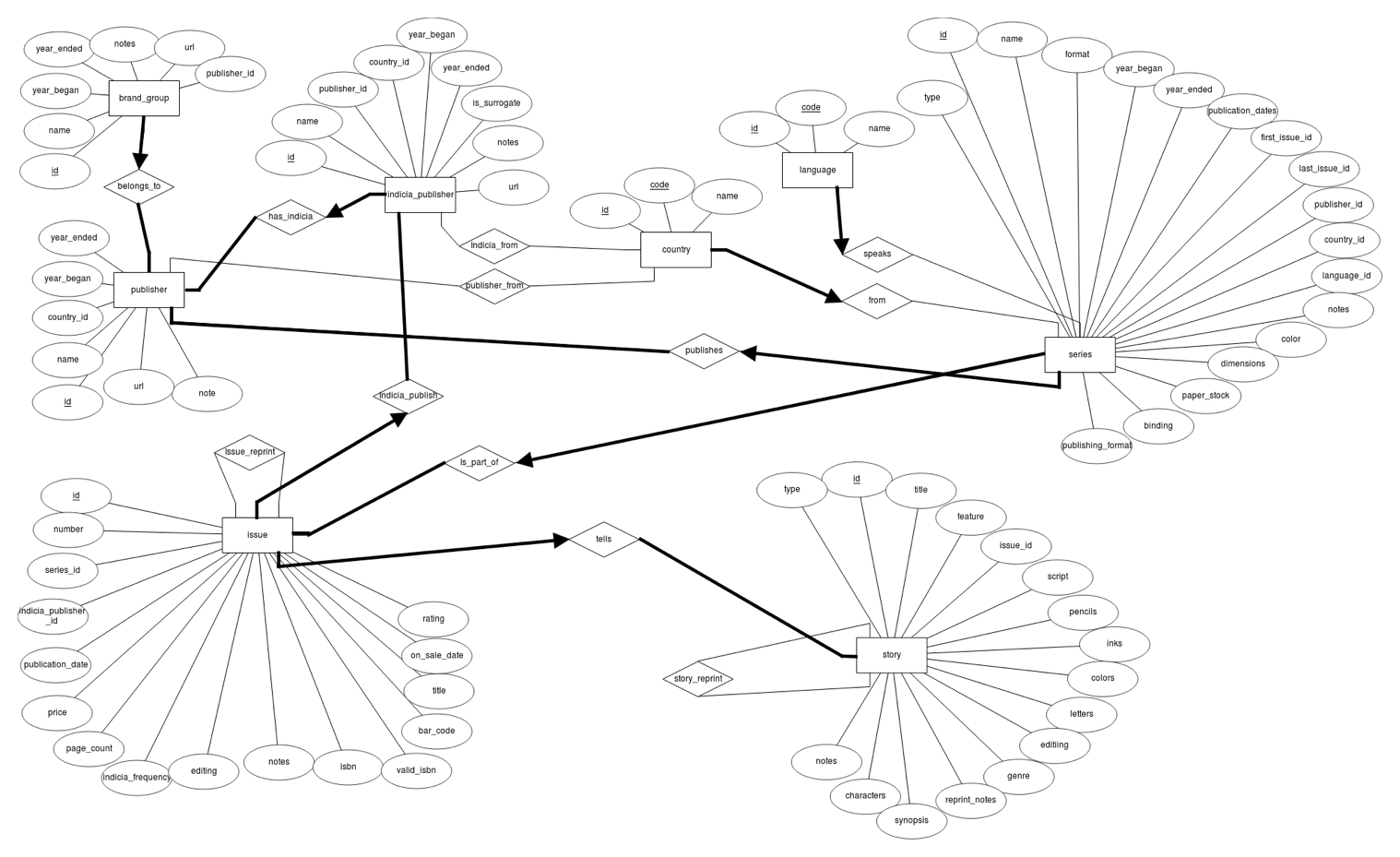
# Deliverable 1

## Assumptions

When we create the tables, we assume that each CHAR has a max size. No longer elements will be added later.

## Entity Relationship Schema

### Schema



### Description

There are two main parts in the schema: Books with Story, Issue and Series and Company with Publisher, Indicia Publisher and Brand Group. Let’s first describe each part before explaining the connections between them.

Books: This part is describing a book in the more general sense. The physical book is an issue. It tells a story and it is part of a series. For example, Harry Potter is a famous series. Harry Potter and the Philosopher’s Stone is a story from this series. Finally, the book with ISBN X is a physical book telling this story. Since these three components are highly connected. There are a lot of constraints between them.

1. An issue tells at least a story.
2. A story must be told by exactly one issue.
3. Each series has at least one issue.
4. An issue is part of exactly one series.

Company: This part is describing the company which publish some books. The main company is the publisher. It has smallest companies, the indicia publishers and it holds some brand group. For example, Marvel is the general company. It holds a brand group called Thor and the physical books are published by an indicia publisher, Thor Entertaining Group, which is part of Marvel. As for the book part, the company part has also a lot of constraints.

1. Each publisher has at least one brand group.
2. Each brand group belongs to exactly one publisher.
3. Each publisher has at least one indicia publisher.
4. Each indicia publisher is part of exactly one publisher.

Then, there are some connections between these two parts. The publisher publishes a series. It means that the publisher creates a series but the issues are printed by the indicia publisher. Here are the constraints related to these connections.

1. A publisher publishes at least one series.
2. A series is published by exactly one publisher.
3. An indicia publisher publishes at least one issue.
4. An issue is published by exactly one indicia publisher.

Finally, there are some smallest relationships. Firstly, the publisher, the indicia publisher and the series come from a country and the series have a language. Secondly, there is a reprint relationship between two stories or two issues. We consider only two constraints.

1. A publisher, indicia publisher or series comes from exactly one country.
2. A series has exactly one language.

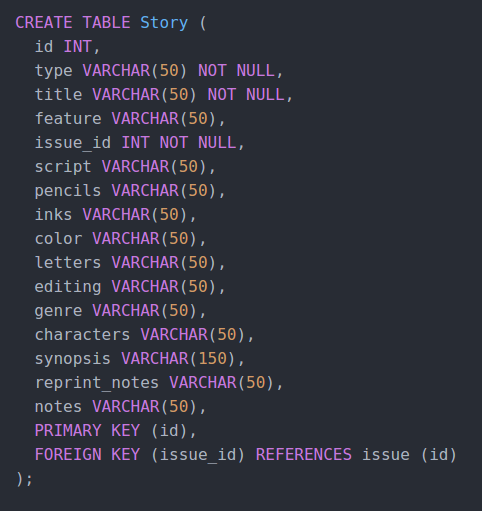
N.B. We do not create a story\_type and series\_publication\_type entity since these data contain only one attribute if we exclude the id. To simplify the schema, we add an attribute type containing the name of the type in the Story/Series entity.

## Relational Schema

### ER schema to Relational schema

To create the table, we simply took each entity or each relationship and we translate them into a table. We took care to respect the constraints using NOT NULL or PRIMARY KEY to state the “at least” constraint and we permit each “at most” relation to appear in only one column in the connected table.

### DDL



For this deliverable, we design the schema together. Then we split the work individually. Colin Branca and Jules Courtois wrote the SQL commands to create the tables and Yoan Martin wrote this pdf document.

## Deliverable 2

### Schema

### Description and Changes :

First of all, we had made a few mistakes in the direction and nature of the constraints which we have corrected, for example with the “Country” entity. Each Indica\_Publisher or Series originates from exactly one country, whereas a country does not require to produce even one of either.

The other big change is around the “Story” entity : we have created numerous new entities and relations which represent some of the attributes in the original database like the people who worked on each story based on their role, the genre of the story, or the characters featured in it. At first, we divided the people into artists, authors, editors and typesetters, for which we created the corresponding N-to-N relations as seen on the diagram. But after some reflexion and looking more into the data, we've realized that some authors are also artists or editors, and some artists are also typesetters which creates a duplication of the data. We are looking into changing this for the next version of the project, maybe by creating a “People” table and using ISA relationships.

To parse the original data and create the new tables, we had to write extra scripts in Python which automatically turned the story table into multiple smaller tables, and removed duplicates.

## Data Loading

## Query Implementation

/\*a) Print the brand group names with the highest number of Belgian indicia publishers \*/

Select BG.name

From BrandGroup BG, (

Select COUNT(distinct \*)

From Publisher P, (

Select \*

From IndiciaPublisher IP, Country C

Where IP.country\_id = C.id and C.name = 'Belgium')

Where P.id = IP.publisher\_id)

Where BG.publisher\_id = P.id and P.ROWNUM = 1

/\*b) Print the ids and names of publishers of Danish book series\*/

SELECT P.id, P.name

FROM Publisher P, Series S, Country C

WHERE S.country\_id = C.id and C.name = 'Denmark' and S.publisher\_id = P.id

/\*c) Print the names of all Swiss series that have been published in magazines \*/

SELECT S.name

FROM Series S, Country C, Serie\_type T, Has\_Serie\_Type H

WHERE H.serie\_id = H.type\_id and Serie\_type.name = 'magazine' and S.country\_id = C.id and C.name = 'Switzerland'

/\*d) Starting from 1990, print the number of issues published each year\*/

Select \*

From Issue I

Group By I.publication\_date

Where I.publication\_date >= 1990

/\*e) Print the number of series of each indicia publisher whose names resembles 'DC comics'\*/

Select Count(distinct \*)

From Issue I, Series S

Where (I.id = S.first\_issue\_id or I.id = S.last\_issue\_id) and I.Indiciapublisher\_id IN(

Select IP.id

From IndiciaPublisher IP

Where IP.name LIKE '%DC comics%')

/\*f) Print the titles of the 10 most reprinted stories \*/

Select title

From (

Select S.title AS title, COUNT(DISTINCT SR.origin\_id) AS count\_reprint

From Story S, StoryReprint SR

Where S.id = SR.origin\_id

Group By S.id, S.title

Order By count\_reprint DESC)

Where ROWNUM <= 10

/\*g) Print the artists that have scripted, drawn, and colored at least one of the stories they where involved in \*/

Select A

From Artists A, Story S, Inks I, Colors C, Pencils P

Where A.id = I.artist\_id and S.id = I.story\_id and A.id = C.artist\_id and S.id = C.story\_id and A.id = C.artist\_id and S.id = C.story\_id

/\*h) Print all non-reprinted stories involving Batman as a non-featured character \*/

Select S

From Characters C, Has\_charaters HC (

Select S

From Story S, StoryReprint SR

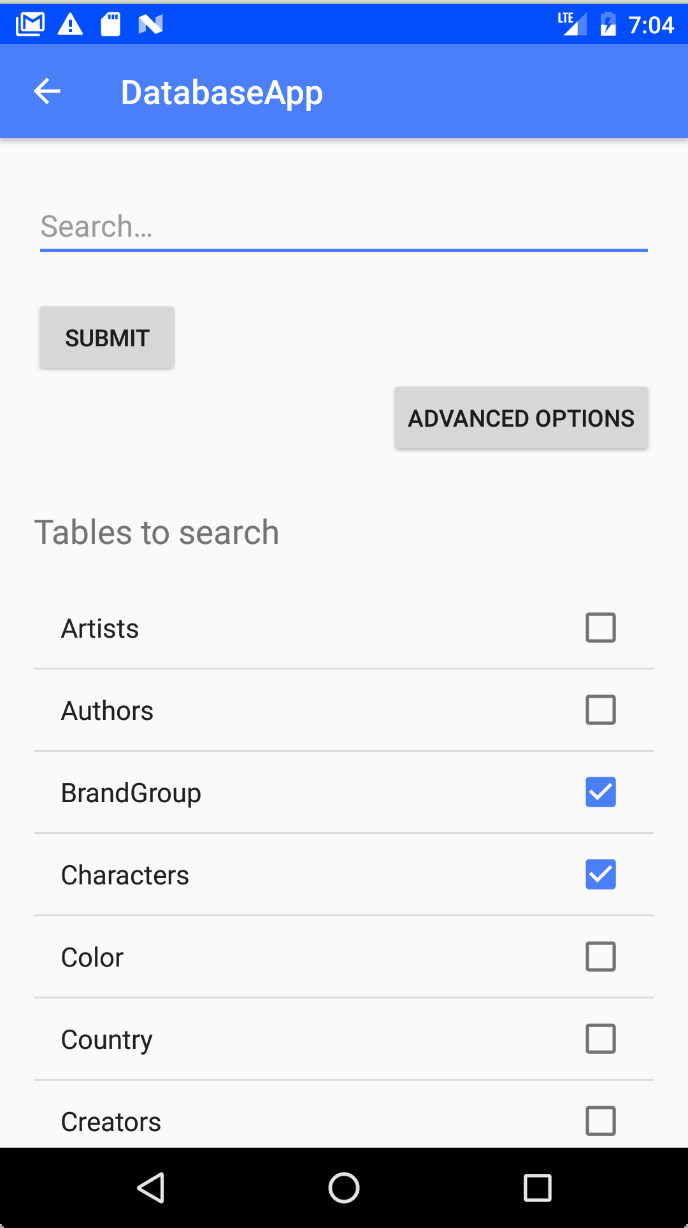
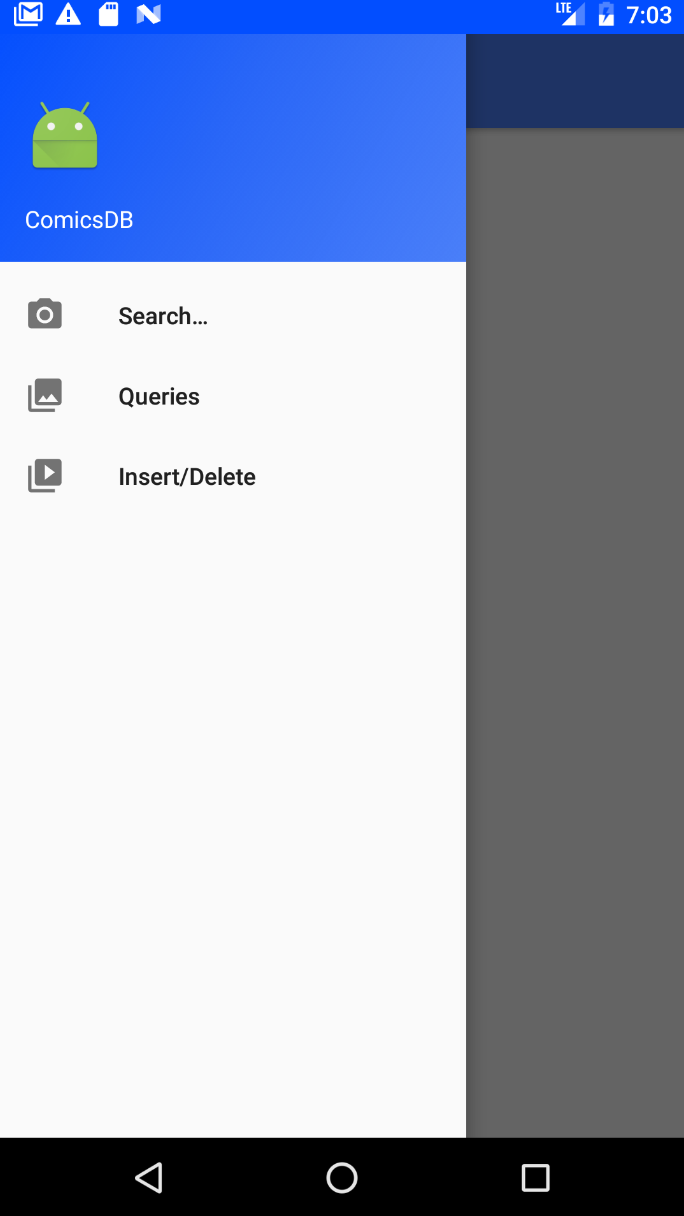
Where not exists (select \* from StoryReprint SR where SR.origin\_id = S.id)

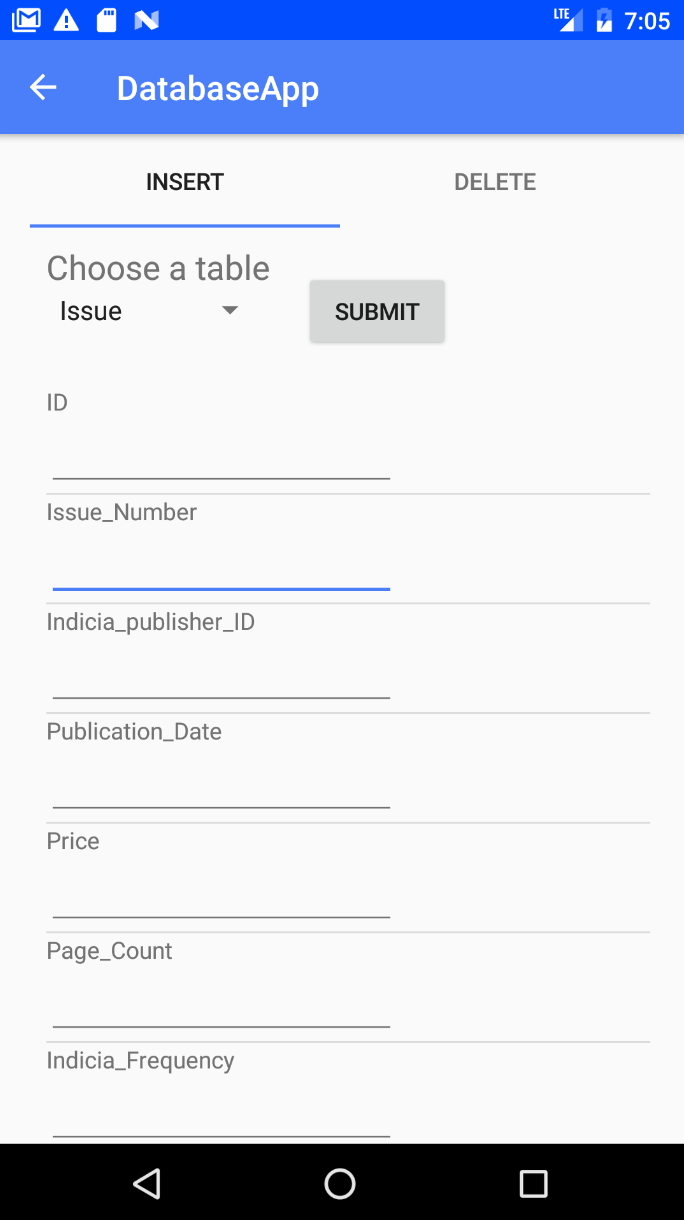
)

Where C.name != 'Batman' and C.id = HC.character\_id and S.id = HC.story\_id

## Interface

For the interface, we use an android application. The navigation in the app is entirely made using the top-left menu button. The user is able to select what he wants to do with the database and he is redirected to the corresponding activity. The frontend part of the application is working well. Unfortunately, we encounter problems with the database connection. So the biggest part of the work for the last deliverable is to work on the backend.





## General Comments

We have met some difficulties due to the data format, for example in dates and how to compare them, or inconsistencies in the names of the authors or even the actual initial database. This will require some work before having a fully functional import.

Concerning the work attribution, Yoan Martin did the Interface, Jules Courtois coded some scripts to clean the dataset and Colin Branca corrected the work done for the last milestone and wrote the SQL queries.

Deliverable 3

Interface

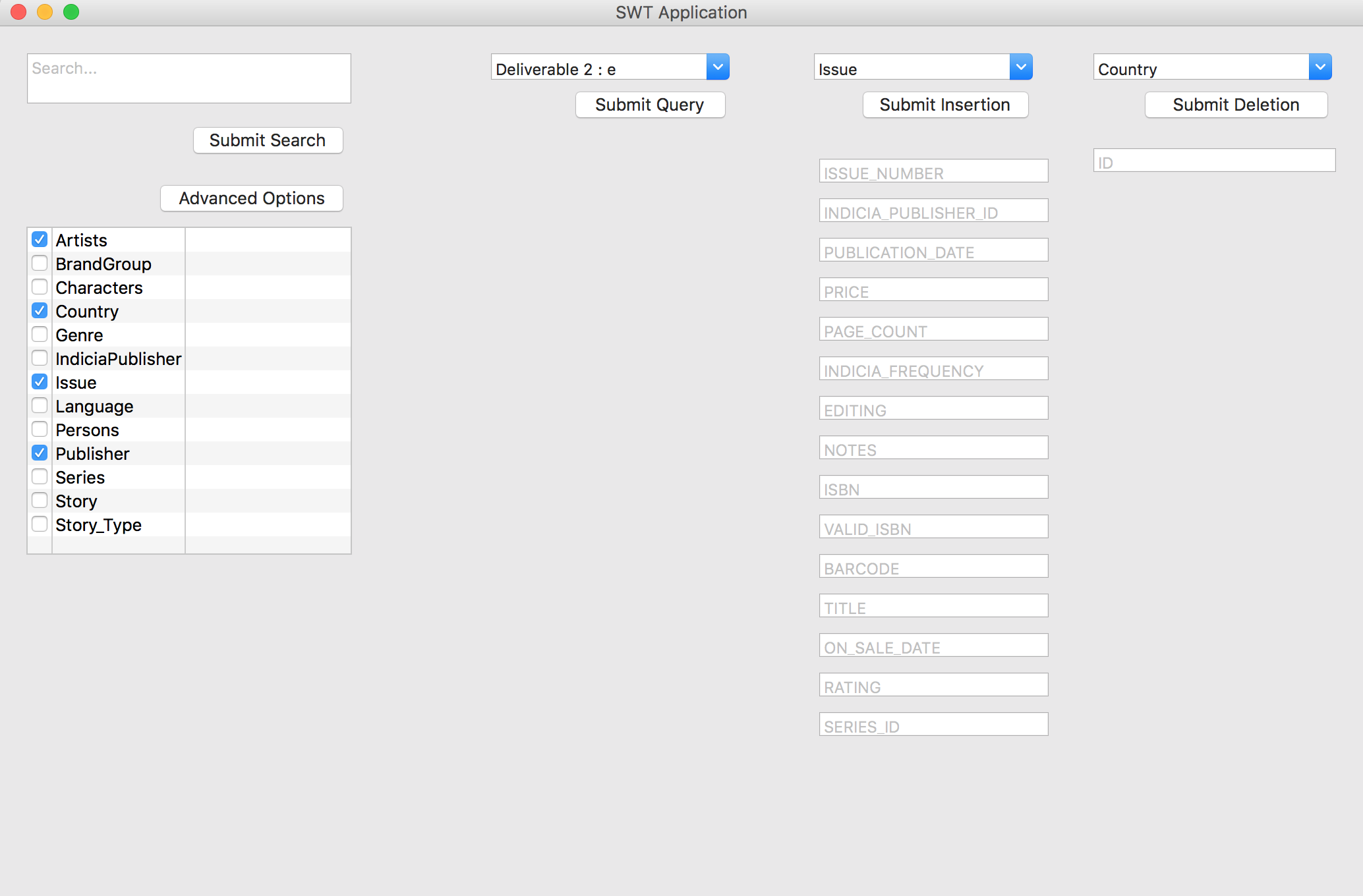
Unfortunately, the user interface presented in the last deliverable was not our final choice. The android application permits to create a simple and user-friendly interface. But we dealt with problem regarding the connection to the database. The connection with the server is different with an android application and a simple Java Applet. When we encountered this difficulty, we had the choice to use an intermediate web-server between our app and the database server or to recycle our Java code and build a new interface in a Java Applet. We selected this last choice. So, our new interface is just a simple window containing four columns (Search, Preselected Queries, Insertion, Deletion).

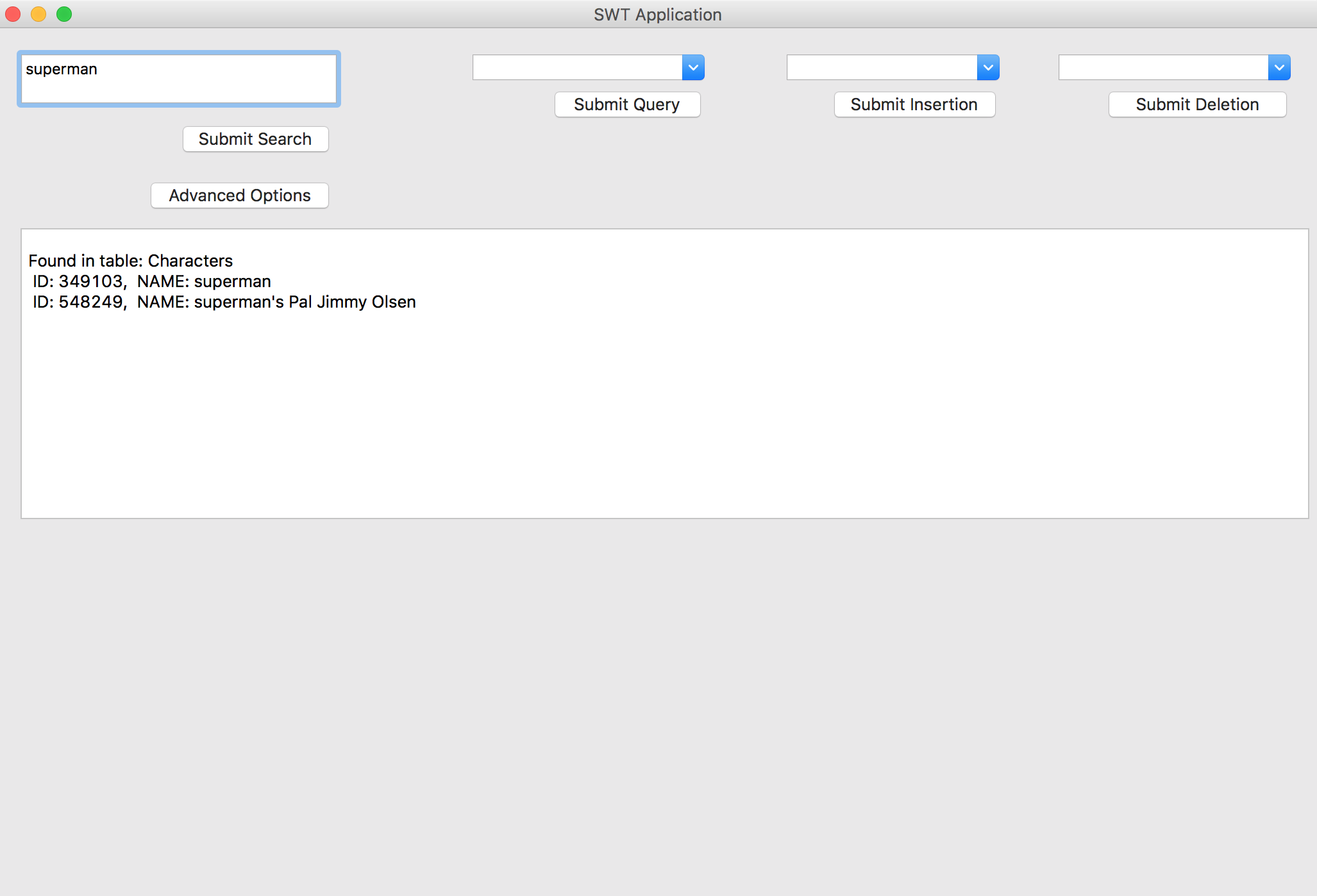
For searching, the user is able to provide any element and select any desirable table. We simple get the search from the user and use it to create an SQL query. If the user does not select any table, we perform the search over every tables. Note that we are not looking for every attribute in a table. We selected only the user-understandable attribute. For example, in the table Language, we perform the search query in the name attribute but not in the id. We consider that the user is not interested to look for database-related attributes, so we focus only on user-friendly attributes.

For selecting a preselected query, the user has a simple menu in which he can chose the desirable query. Note that for the last query, we display a new text field. Then the user is able to provide the id of an issue and finally we can perform our query like all the others.

For the insertion, the user has a menu to select the table in which he would like to delete an element. Then, we dynamically create some text fields for each attribute. Finally, the user enters the different value in each field and we perform the insertion.

For the deletion, this is our less user-friendly part. Since the user does not have every required attribute value to delete an element. We only provide a deletion using an ID. The idea is that the user search for the desirable element. He looks into the result to find the correct ID. And then, he performs the deletion using a menu to select the correct table and then entering the value of the ID.





Queries for Deliverable 3

a) Print the series names that have the highest number of issues which contain a story whose type is not the one occurring most frequently in the database

SELECT SE.name

FROM SERIES SE,

(SELECT DISTINCT I.name, I.series\_id

FROM ISSUE I, STORY S,

(SELECT mycount.typeid as id

FROM (SELECT ST.id AS typeid

FROM STORY\_TYPE ST, STORY S

WHERE S.type\_id = ST.id

GROUP BY S.type\_id

ORDER BY COUNT(S.id) DESC) mycount

WHERE ROWNUM = 1) maxtype

WHERE S.issue\_id = I.id AND S.type\_id <> maxtype.id) myissues

WHERE myissues.series\_id = SE.id AND ROWNUM = 1

GROUP BY myissues.series\_id

ORDER BY COUNT(myissues.series\_id) DESC

b) Print the names of publishers who have series with all series types

Select P.name

From Publisher P, (Select distinct P.id as id, Count(distinct S.publication\_type) as num

From Publisher P, Series S

Where S.publisher\_id = P.id

Group by P.id) MyP

Where P.id = MyP. id and MyP.num = (Select Count(distinct S.publication\_Type) From Series S);

c) Print the 10 most-reprinted characters from Alan Moore's stories.

Select MyTable.MyName

From (

Select C.name as MyName, Count(C.name) as MyCount

From Main M, Feature F, Characters C, (Select S.id as Myid

From Story S, Persons P, Authors A, StoryReprint SR

Where P.id = A.persons\_id and Lower(P.name) like '%alan moore%' and A.story\_id = S.id and S.id = SR.target\_id) Stid

Where ((C.id = M.character\_id and M.story\_id in Stid.Myid) OR (C.id = F.character\_id and F.story\_id in Stid.Myid))

Group By C.name

Order By Count(C.name) desc) MyTable

Where ROWNUM <= 10;

d) Print the writers of nature-related stories that have also done the pencilwork in all their nature-related stories.

Select distinct P.name

From Persons P, Authors A, Pencils Pen, (Select S.id as id

From Story S, Genre G, Has\_Genre HG

Where S.id = HG.story\_id and HG.genre\_id = G.id and LOWER(G.name) LIKE '%nature%') MyTable

Where P.id = A.persons\_id and P.id = Pen.artist\_id and MyTable.id = A.story\_id and MyTable.id = Pen.story\_id;

e) For each of the top-10 publishers in terms of published series, print the 3 most popular languages of their series.

Select L.id, Count(S.id)

From Series S, Language L, (Select MyPublisher.id as id

From (Select P.id as id, Count(S.id)

From Publisher P, IndiciaPublisher IP, Series S

Where IP.publisher\_id = P.id and S.publisher\_id = P.id

Group by P.id

Order by Count(S.id) desc) MyPublisher

Where ROWNUM <= 10) MyTopPublisher

Where S.publisher\_id = MyTopPublisher.id and S.language\_id = L.id

Group by L.id

Order by Count(S.id) desc

f) Print the languages that have more than 10000 original stories published in magazines, along with the number of those stories.

Select L.name, Res.num

From (Select SE.language\_id as id, Count(SE.language\_id) as num

From Story ST, Series SE, Issue I

Where ST.issue\_id = I.id and I.series\_id = SE.id and SE.publication\_type = 'magazine'

Group By SE.language\_id

Having Count(\*) > 10000) Res, Language L

Where L.id = Res.id;

g) Print all story types that have not been published as a part of Italian magazine series

Select Type.name

From Story\_Type Type

Where Type.id not in (

Select distinct ST.type\_id

From Story ST, Series SE, Issue I, Language L

Where ST.issue\_id = I.id and I.series\_id = SE.id and SE.language\_id = L.id and L.name = 'Italian' and SE.publication\_type = 'magazine');

h) Print the writers of cartoon stories who have worked as writers for more than one indicia publisher

Select Pe.name

From (Select distinct P.name as name, P.id

From Persons P, Authors A

Where P.id = A.persons\_id and A.story\_id in(

Select S.id

From Story S, Story\_type ST

Where S.type\_id = ST.id and ST.name = 'cartoon')) Pe, Authors A, Story S, Issue I, IndiciaPublisher IP

Where Pe.id = A.persons\_id and A.story\_id = S.id and S.issue\_id = I.id and I.indicia\_publisher\_id = IP.id

Group by Pe.name

Having Count(\*) > 2;

i) Print the 10 brand groups with the highest number of indicia publishers

Select Br.name

From(Select B.name

From Brandgroup B, Publisher P, IndiciaPublisher IP

Where B.id = P.id and P.id = IP.publisher\_id

Group By B.name

Order By Count(B.name) DESC) Br

Where ROWNUM <= 10;

j) Print the average series length (in terms of years) per indicia publisher.

Select IP.name, ROUND(AVG(CAST(S.year\_ended+1 - S.year\_began as FLOAT)), 2)

From Series S, Publisher P, IndiciaPublisher IP

Where S.publisher\_id = P.id and IP.publisher\_id = P.id

Group By IP.name;

k) Print the top 10 indicia publishers that have published the most single-issue series

Select MyIndiciaPublisher.name

From (Select IP.name as name, Count(distinct MyUniqueIssueSeries.id)

From Publisher P, IndiciaPublisher IP, Series S, (Select MySeries.id as id

From (Select S.id as id, Count(I.id) as num

From Issue I, Series S

Where I.series\_id = S.id

Group by S.id) MySeries

Where MySeries.num = 1) MyUniqueIssueSeries

Where P.id = IP.publisher\_id and MyUniqueIssueSeries.id = S.id and S.publisher\_id = P.id

Group by IP.name

Order by Count(MyUniqueIssueSeries.id) desc) MyIndiciaPublisher

Where ROWNUM <= 10;

l) Print the 10 indicia publishers with the highest number of script writers in a single story.

Select IP1.name

From (

Select IP.name as name, Count(distinct myStory.id)

From IndiciaPublisher IP, Issue I, Story S,

(

Select A.Story\_id as id, Count(A.Persons\_id) as num

From Authors A

Group By A.Story\_id

Order By Count(A.Persons\_id) DESC

) myStory

Where IP.id = I.INDICIA\_PUBLISHER\_ID AND I.id = S.ISSUE\_ID AND myStory.id = S.id

Group by IP.name

order by Count(distinct myStory.id) DESC

) IP1

Where ROWNUM <= 10 ;

m) Print all Marvel heroes that appear in Marvel-DC story crossovers.

Select distinct C1.name

From

(Select C.id as id, C.name as name

From Indiciapublisher IP, Characters C, Main M, Feature F, Issue I, Story S

Where Lower(IP.name) like '%marvel%' and Lower(IP.name) like '%dc%' and I.indicia\_publisher\_id = IP.id and S.issue\_id = I.id and ((F.story\_id = S.id and F.character\_id = C.id)

OR (M.story\_id = S.id and M.character\_id = C.id))) C1,

(Select C.id as id, C.name as name

From Characters C, Main M, Feature F, Issue I, IndiciaPublisher IP, Story S

Where Lower(IP.name) like '%marvel%' and I.indicia\_publisher\_id = IP.id and S.issue\_id = I.id and ((F.story\_id = S.id and F.character\_id = C.id)

OR (M.story\_id = S.id and M.character\_id = C.id))) C2

Where C1.id = C2.id;

n) Print the top 5 series with most issues

Select S.name

From Series S,

(Select I.series\_id as id, Count(I.id)

From Issue I

Group by I.series\_id

Order by Count(I.id) desc) MyIssue

Where S.id = MyIssue.id and ROWNUM <= 10

o) Given an issue, print its most reprinted story.

Select S.title

From Issue I, Story S, (Select SR.origin\_id as id, Count(SR.target\_id)

From StoryReprint SR

Group by SR.origin\_id

Order by Count(SR.target\_id) desc) MyMostReprinted

Where I.id = elem and I.id = S.issue\_id and S.id = MyMostReprinted.id and ROWNUM = 1;

## General Comments

Concerning the work attribution, Yoan Martin finished the Interface, Jules Courtois and Colin Branca finished the cleaning of the dataset and finished to import all the data into the database. Finally, we work together to write the queries.