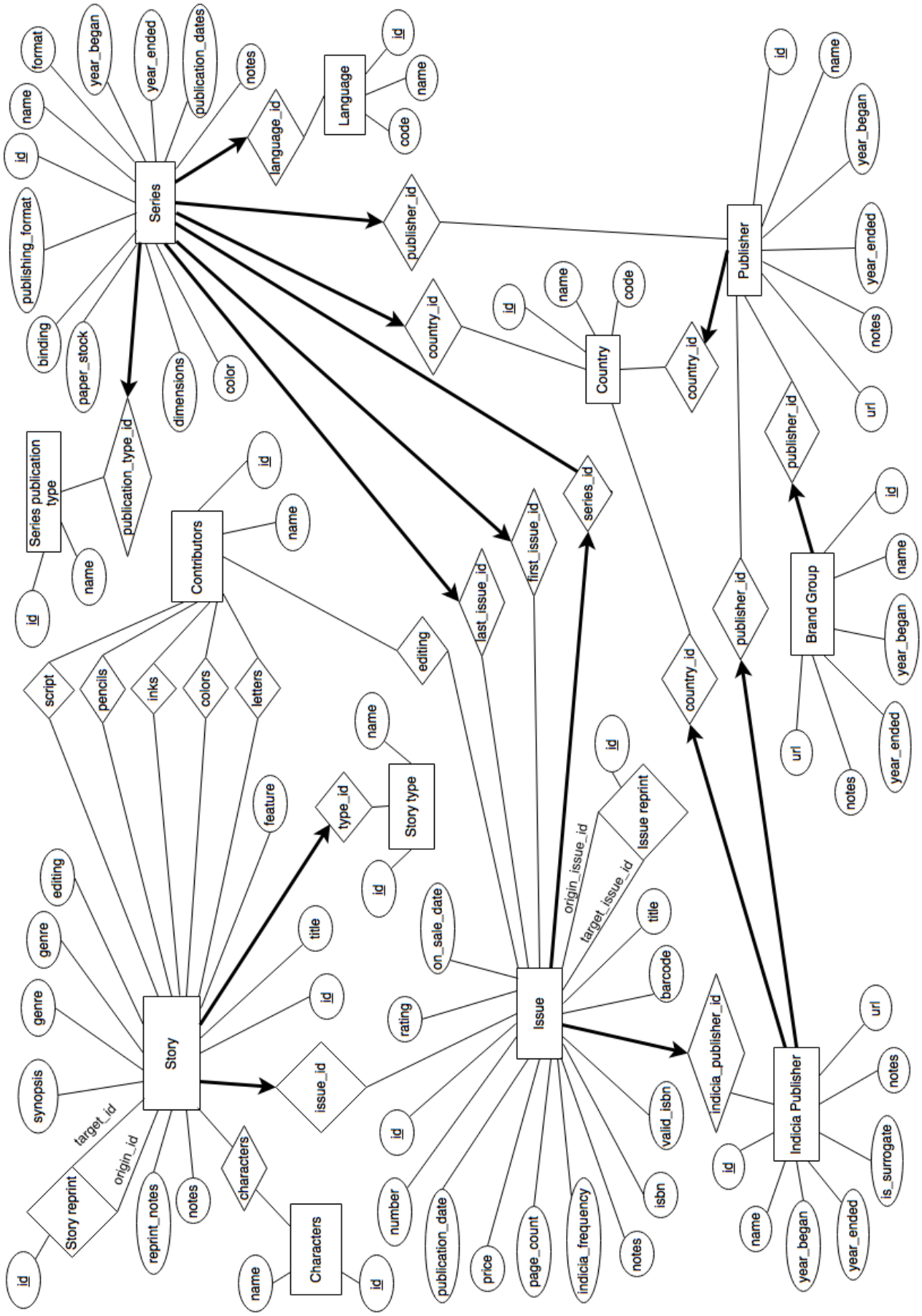
Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

Introduction to Database Systems Project

ER model



Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

Design choices

We chose to move the `Contributors` to their own relation, in order to avoid having to store redundant data. This choice was made because we observed that the script, pencil and ink of various stories were done by the same person. This happens both within the same tuple, as well as between different tuples. Of course, as this new entity set was added the corresponding relationship sets to `Story` as well as `Issue` had to be added.
Additionally we added a relation `Characters` and corresponding relationship set to `Story`.
Other that that we left the data as it was described prior.

In terms of what relationship sets we actually store in their own relation, any sets where there may only be one entry were added into the corresponding entity sets (`issue_id` & `type_id` in Story, among many others). Only the relationship sets that connected to `Contributors` or `Characters` remained their own relations, all others were merged as foreign keys into the corresponding entity set.

Data Constraints

We haven't set any attributes to `NOT NULL` as (apart from the primary keys, where this is automatically enforced) any entry may be empty. This was observed by looking over the provided tables.

The same is true for `UNIQUE`, as, apart from the primary keys, entries may be identical.

The primary keys of the entity sets were mostly given as the `id`. The others are:
- the entity set `Contributor`, which we created. Here we added an attribute `id`, as the name may not be unique.
- the relationship sets, where the foreign keys of the referenced entity sets are used to form the primary key

As for the foreign keys, we chose the following actions:
- `ON DELETE` is set to `SET NULL`, in order to keep a valid table, but remove the information.
- In the relationship sets however we set it to `CASCADE`, so that the corresponding tuples disappear along with the tuple within the referenced entity set.

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

Data Import

Before importing, we had to parse the .csv files using Python and some Shell commands, to correct the known issues.

Namely:
- Replace `&` by `\&`
- Replace single quotes `'` by `´`
- Remove `?`, `[nn]`, `[none]`, `none`, …
  - Simplifies finding empty cells, as they will be just empty and not contain a description saying it is empty
- Remove double quotes `"`
  - Simplifies handling the .csv files in other programs
- Remove any characters from `Integer` attributes

Furthermore, we had to properly find the longest fields in any attribute in order to create the tables. As a sidenote to this: For the attribute `notes` of the different entity sets we decided to use `VARCHAR2(*)`, which has a maximal length of 4'000 characters. This meant that some of the texts in Story had to be truncated to be no longer than those 4'000 characters. We decided to truncate at the last complete word.

Once all that was done, we went to SQLDeveloper, created the tables and imported the .csv files one by one. Doing this, we ignored all attributes that we put into `Contributors` or `Characters`.
These was created by parsing the corresponding attributes in `Story` and `Issue`, inserting the names into the respective entity set, and inserting all the links into the corresponding relationship sets.

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

DDL

```
CREATE TABLE Brand_Group
  (
    id INTEGER,
    name char(128),
    year_began INTEGER,
    year_ended INTEGER,
    notes VARCHAR2(4000),
    url CHAR(256),
    publisher_id INTEGER,
    PRIMARY KEY(id),
    FOREIGN KEY(publisher_id) REFERENCES Publisher(id)
      ON DELETE SET NULL
  )

CREATE TABLE Characters
  (
    id INTEGER,
    name CHAR(256),
    PRIMARY KEY(id)
  )

CREATE TABLE Contributors
  (
    id INTEGER,
    name CHAR(256),
    PRIMARY KEY(id)
  )

CREATE TABLE Country
  (
    id INTEGER,
    code CHAR(4),
    name CHAR(64),
    PRIMARY KEY(id)
  )
```

```
CREATE TABLE Indicia_Publisher
  (
    id INTEGER,
    name CHAR(128),
    publisher_id INTEGER,
    country_id INTEGER,
    year_began INTEGER,
    year_ended INTEGER,
    is_surrogate INTEGER, /* 1 or 0 */
    notes VARCHAR2(4000),
    url CHAR(256),
    PRIMARY KEY(id),
    FOREIGN KEY(publisher_id) REFERENCES Publisher(id)
      ON DELETE SET NULL,
    FOREIGN KEY(country_id) REFERENCES Country(id)
      ON DELETE SET NULL
  )

CREATE TABLE Issue
  (
    id INTEGER,
    nb CHAR(32),
    series_id INTEGER,
    indicia_publisher_id INTEGER,
    publication_date CHAR(64),
    price CHAR(128),
    page_count INTEGER,
    indicia_frequency CHAR(128),
    notes VARCHAR2(4000),
    isbn CHAR(32),
    valid_isbn INTEGER,
    barcode INTEGER,
    title CHAR(128),
    on_sale_date CHAR(16),
    rating CHAR(64),
    PRIMARY KEY(id),
    FOREIGN KEY(series_id) REFERENCES Series(id)
      ON DELETE SET NULL,
    FOREIGN KEY(indicia_publisher_id) REFERENCES Indicia_Publisher(id)
      ON DELETE SET NULL
  )

CREATE TABLE Issue_Editing
  (
    issue_id INTEGER,
    editor_id INTEGER,
    PRIMARY KEY(issue_id, editor_id),
    FOREIGN KEY(issue_id) REFERENCES Issue(id)
      ON DELETE CASCADE,
    FOREIGN KEY(editor_id) REFERENCES Contributors(id)
      ON DELETE CASCADE
  )
```

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

```
CREATE TABLE Issue_Reprint
  (
    id INTEGER,
    origin_issue_id INTEGER,
    target_issue_id INTEGER,
    PRIMARY KEY(id),
    FOREIGN KEY(origin_issue_id) REFERENCES Issue(id)
      ON DELETE CASCADE,
    FOREIGN KEY(target_issue_id) REFERENCES Issue(id)
      ON DELETE CASCADE
  )

CREATE TABLE Language
  (
    id INTEGER,
    code CHAR(4),
    name CHAR(64),
    PRIMARY KEY(id)
  )

CREATE TABLE Publisher
  (
    id INTEGER,
    name CHAR(128),
    country_id INTEGER,
    year_began INTEGER,
    year_ended INTEGER,
    notes VARCHAR2(4000),
    url CHAR(256),
    PRIMARY KEY(id),
    FOREIGN KEY(country_id) REFERENCES Country(id)
      ON DELETE SET NULL
  )
```

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

```
CREATE TABLE Series /* REIMPORT TO CORRECT TYPES ?*/
  (
    id INTEGER,
    name CHAR(256),
    format CHAR(256),
    year_began INTEGER,
    year_ended INTEGER,
    publication_dates CHAR(128),
    first_issue_id INTEGER,
    last_issue_id INTEGER,
    publisher_id INTEGER,
    country_id INTEGER,
    language_id INTEGER,
    notes VARCHAR2(4000),
    color VARCHAR(256),
    dimensions VARCHAR(256),
    paper_stock VARCHAR(256),
    binding VARCHAR(128),
    publishing_format CHAR(128),
    publication_type_id INTEGER,
    PRIMARY KEY(id),
    FOREIGN KEY(first_issue_id) REFERENCES Issue(id)
      ON DELETE SET NULL,
    FOREIGN KEY(last_issue_id) REFERENCES Issue(id)
      ON DELETE SET NULL,
    FOREIGN KEY(publisher_id) REFERENCES Publisher(id)
      ON DELETE SET NULL,
    FOREIGN KEY(country_id) REFERENCES Country(id)
      ON DELETE SET NULL,
    FOREIGN KEY(language_id) REFERENCES Language(id)
      ON DELETE SET NULL,
    FOREIGN KEY(publication_type_id) REFERENCES Publication_Type(id)
      ON DELETE SET NULL
  )

CREATE TABLE Series_Publication_Type
  (
    id INTEGER,
    name CHAR(16),
    PRIMARY KEY(id)
  )
```

```
CREATE TABLE Story
  (
    id INTEGER,
    title CHAR(256),
    feature VARCHAR(256),
    issue_id INTEGER,
    letters VARCHAR2(1024),
    editing VARCHAR(512),
    genre CHAR(128),
    characters VARCHAR(256),
    synopsis VARCHAR2(4000),
    reprint_notes VARCHAR2(4000),
    notes VARCHAR2(4000),
    type_id INTEGER,
    PRIMARY KEY(id),
    FOREIGN KEY(issue_id) REFERENCES Issue(id)
      ON DELETE SET NULL,
    FOREIGN KEY(type_id) REFERENCES Story_Type(id)
      ON DELETE SET NULL
  )

CREATE TABLE Story_Characters
  (
    story_id INTEGER,
    character_id INTEGER,
    PRIMARY KEY(story_id, character_id),
    FOREIGN KEY(story_id) REFERENCES Story(id)
      ON DELETE CASCADE,
    FOREIGN KEY(character_id) REFERENCES Characters(id)
      ON DELETE CASCADE
  )

CREATE TABLE Story_Colors
  (
    story_id INTEGER,
    color_id INTEGER,
    PRIMARY KEY(story_id, color_id),
    FOREIGN KEY(story_id) REFERENCES Story(id)
      ON DELETE CASCADE,
    FOREIGN KEY(color_id) REFERENCES Contributors(id)
      ON DELETE CASCADE
  )

CREATE TABLE Story_Inks
  (
    story_id INTEGER,
    ink_id INTEGER,
    PRIMARY KEY(story_id, ink_id),
    FOREIGN KEY(story_id) REFERENCES Story(id)
      ON DELETE CASCADE,
    FOREIGN KEY(ink_id) REFERENCES Contributors(id)
      ON DELETE CASCADE
  )
```

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

```
CREATE TABLE Story_Letters
  (
    story_id INTEGER,
    letter_id INTEGER,
    PRIMARY KEY(story_id, letter_id),
    FOREIGN KEY(story_id) REFERENCES Story(id)
      ON DELETE CASCADE,
    FOREIGN KEY(letter_id) REFERENCES Contributors(id)
      ON DELETE CASCADE
  )

CREATE TABLE Story_Pencils
  (
    story_id INTEGER,
    pencil_id INTEGER,
    PRIMARY KEY(story_id, pencil_id),
    FOREIGN KEY(story_id) REFERENCES Story(id)
      ON DELETE CASCADE,
    FOREIGN KEY(pencil_id) REFERENCES Contributors(id)
      ON DELETE CASCADE
  )

CREATE TABLE Story_Reprint
  (
    id INTEGER,
    origin_id INTEGER,
    target_id INTEGER,
    PRIMARY KEY(id)
    FOREIGN KEY(origin_id) REFERENCES Story(id)
      ON UPDATE CASCADE,
    FOREIGN KEY(target_id) REFERENCES Story(id)
      ON UPDATE CASCADE
  )

CREATE TABLE Story_Script
  (
    story_id INTEGER,
    script_id INTEGER,
    PRIMARY KEY(story_id, script_id),
    FOREIGN KEY(story_id) REFERENCES Story(id)
      ON DELETE CASCADE,
    FOREIGN KEY(script_id) REFERENCES Contributors(id)
      ON DELETE CASCADE
  )

CREATE TABLE Story_Type
  (
    id INTEGER,
    name char(64),
    PRIMARY KEY(id)
  )
```

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

The tables on the right show up to the first 10 rows of the output.

SQL Queries 1

a) Print the brand group names with the highest number of Belgian indicia publishers.

```
select bg.NAME
from INDICIA_PUBLISHER ip inner join BRAND_GROUP bg
    on bg.PUBLISHER_ID=ip.PUBLISHER_ID
  inner join COUNTRY c on ip.COUNTRY_ID=c.ID
where c.NAME='Belgium'
group by bg.ID, bg.NAME
order by count(*) desc
```

	NAME
1	Spotlight
2	Mezzanine
3	Ukje
4	Puceron
5	Collectie Vrolijke Vlucht
6	RepéRages Dupuis
7	Dupuis
8	Graton
9	Uitgeverij Dupuis
10	Aire Libre

b) Print the ids and names of publishers of Danish book series.

```
select p.ID, p.NAME
from PUBLISHER p inner join COUNTRY c
    on p.COUNTRY_ID=c.ID
  inner join SERIES s on s.PUBLISHER_ID=p.ID
  inner join SERIES_PUBLICATION_TYPE spt
    on s.PUBLICATION_TYPE_ID=spt.ID
where c.NAME='Denmark' and spt.NAME='book'
```

	ID	NAME
1	849	Bogfabrikken
2	1318	Faraos Cigarer
3	3843	DC=Data
4	4520	Borgens Forlag
5	4551	Interpresse
6	4858	Donovan Comics
7	5180	Lego Publishing
8	7425	Bibelselskabets Forlag
9	7426	Cobolt
10	7428	Forlaget Zoom

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

c) Print the names of all Swiss series that have been published in magazines.

```
select s.NAME
from SERIES s inner join SERIES_PUBLICATION_TYPE spt
    on s.PUBLICATION_TYPE_ID=spt.ID
  inner join COUNTRY c on s.COUNTRY_ID=c.ID
where c.NAME='Switzerland'
    and spt.NAME='magazine'
```

	NAME
1	Micky Maus Zeitung
2	Tip Top
3	Junior
4	Melanie
5	Comixene

d) Starting from 1990, print the number of issues published each year.

```
select year, count(*) as nb
from
  (select i.id,
    to_number(regexp_substr(i.PUBLICATION_DATE,'\d{4}'))
      as year
  from ISSUE i)
where year>=1990 and year<=2017
group by year
order by year
```

	YEAR	NB
1	1990	6406
2	1991	6473
3	1992	6618
4	1993	7211
5	1994	7245
6	1995	7107
7	1996	6522
8	1997	6210
9	1998	6000
10	1999	5618

e) Print the number of series for each indicia publisher whose name resembles 'DC comics'.

```
select ip.NAME, count(*) as nb
from SERIES s
  inner join INDICIA_PUBLISHER ip
    on s.PUBLISHER_ID=ip.PUBLISHER_ID
where ip.NAME like '%DC Comics%'
group by ip.ID, ip.NAME
order by count(*) desc
```

	NAME	NB
1	Marvel Comics Group and DC Comics Inc.	7628
2	Marvel Entertainment Group Inc. and DC Comics Inc.	7628
3	DC Comics	6993
4	DC Comics Inc.	6993
5	DC Comics; Top Cow Productions Inc.	6993
6	DC Comics	29
7	Marvel Comics; DC Comics	29
8	DC Comics	23
9	DC Comics and Dark Horse Comics Inc.	23
10	Dark Horse Comics Inc. and DC Comics	23

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

f) Print the titles of the 10 most reprinted stories

```
select s.TITLE
from STORY_REPRINT sr inner join STORY s
    on sr.ORIGIN_ID=s.ID
where s.TITLE is not null
group by s.ID, s.TITLE
order by count(*) DESC
fetch first 10 rows only
```

	NAME
1	Spaghetti Brothers - historien om søsknene Centobucchi
2	Spider-Man!
3	Verso l`ignoto
4	Spider Man
5	The Fantastic Four!
6	Le nain de Corneloup
7	Le sortilège du haricot
8	Flash of Two Worlds!
9	Spider-Man
10	Is He Man or Monster or... Is He Both!

g) Print the artists that have scripted, drawn, and colored at least one of the stories they were involved in.

```
select distinct c.NAME
from CONTRIBUTORS c inner join STORY_SCRIPT ss
    on c.ID=ss.SCRIPT_ID
  inner join STORY_COLORS sc
    on c.ID=sc.COLOR_ID and ss.STORY_ID=sc.STORY_ID
  inner join STORY_PENCILS sp
    on c.ID=sp.PENCIL_ID and ss.STORY_ID=sp.STORY_ID
```

	NAME
1	J.R. Hager
2	M. E. Brady
3	Jack Burnley
4	Bob Davis
5	John Chaffin
6	Georges Rémi
7	Matt Fox
8	Mel Lazarus
9	Clemens Gretter
10	Bob Fujitani

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

h) Print all non-reprinted stories involving Batman as a non-featured character.

```
select s.TITLE
from STORY s inner join STORY_CHARACTERS sc on s.ID=sc.STORY_ID
  inner join CHARACTERS c on c.ID=sc.CHARACTER_ID
  left join STORY_REPRINT sr on s.ID=sr.ORIGIN_ID
where c.NAME='Batman'
  and not s.FEATURE like '%Batman%'
  and s.TITLE is not null
  and sr.ORIGIN_ID is null
```

	NAME
1	You Don`t Need X-Ray Vision
2	October 30 2013
3	Hero`s Journey
4	Cover - Justice League of America #80
5	The Doom Patrol #104
6	Joker: Wer zuletzt lacht Teil 4: Ein Ausflug ins Nirgendwo
7	Re: Action
8	Panic from a Blackmail Box!
9	Die Abschussliste Epilog: Wieder an der Front
10	Where Valor Fails... Will Magic Triumph

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

SQL Queries 2

a) Print the series names that have the highest number of issues which contain a story whose type (e.g., cartoon) is not the one occurring most frequently in the database (e.g, illustration).

```
select s.NAME, count(distinct i.ID) as nb
from SERIES s inner join ISSUE i
    on s.ID=i.SERIES_ID
  inner join STORY st on i.ID=st.ISSUE_ID
where st.TYPE_ID not in
  (select type_id
  from story
  group by type_id
  order by count(*) desc
  fetch first 1 rows only)
group by s.ID, s.NAME
order by count(distinct i.ID) desc
```

	NAME
1	Commando
2	Donald Duck \& Co
3	Love Story Picture Library
4	Gespenster Geschichten
5	Kamp-serien
6	Four Color
7	Hjerterevyen
8	Action Comics
9	Detective Comics
10	Jukan

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

```
select p.NAME
from PUBLISHER p inner join SERIES s
  on p.ID=s.PUBLISHER_ID
group by p.ID, p.NAME
having count(distinct s.PUBLICATION_TYPE_ID)=3
order by p.ID
```

	NAME
1	Casterman
2	DC
3	Marvel
4	Sergio Bonelli Editore
5	Atlas Publishing
6	Western
7	Classics/Williams
8	Åhlén \& Åkerlunds
9	Hemmets Journal
10	Williams Förlags AB

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

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

```
select ch.NAME
from STORY_CHARACTERS sc
  inner join CHARACTERS ch
    on ch.ID=sc.CHARACTER_ID
  inner join STORY_SCRIPT ss
    on sc.STORY_ID=ss.STORY_ID
  inner join CONTRIBUTORS c
    on ss.SCRIPT_ID=c.ID
where c.name='Alan Moore'
group by ch.id, ch.NAME
order by count(*) desc
fetch first 10 rows only
```

	NAME
1	Miracle Man
2	Liz Moran
3	Swamp Thing
4	Captain Britain
5	Evelyn Cream
6	Kid MiracleMan
7	Abigail Cable
8	Thomas Lennox
9	Tom Strong
10	GUESTS: Opal Luna Saturnyne

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

```
select c.NAME
from STORY s inner join STORY_SCRIPT sc
    on s.ID=sc.STORY_ID
  full join STORY_PENCILS sp
    on s.ID=sp.STORY_ID
      and sc.SCRIPT_ID=sp.PENCIL_ID
  inner join CONTRIBUTORS c
    on sc.SCRIPT_ID=c.ID
where s.GENRE like '%nature%'
group by c.ID, c.NAME
having count(s.ID)=count(sp.PENCIL_ID)
  and count(s.ID)=count(sc.STORY_ID)
```

	NAME
1	Joe Kubert
2	Harry Pettit
3	Derek Sayer
4	Dexter Taylor
5	Herman Browner
6	Kate Craig
7	Mario DeMarco
8	Martin Flink
9	Dierdre Latimer Sayer
10	Morris Waldinger

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

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

Not entirely correct, gets all languages for the top 10 publishers.

```
select p.NAME as pub, l.NAME as lang
from SERIES s inner join LANGUAGE l on
s.LANGUAGE_ID=l.ID
  inner join PUBLISHER p on s.PUBLISHER_ID=p.ID
where p.ID in
  (select p.ID
  from series s inner join publisher p on
s.PUBLISHER_ID=p.ID
  group by p.ID, p.NAME
  order by count(*) desc
  fetch first 10 rows only)
group by p.ID, p.NAME, l.ID, l.NAME
order by p.ID
```

	PUB	LANG
1	DC	English
2	DC	French
3	DC	Bosnian
4	Marvel	English
5	Marvel	Spanish
6	Marvel	French
7	Fantagraphics	English
8	Dark Horse	English
9	Egmont Ehapa	German
10	Egmont Ehapa	English

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

```
select l.NAME, count(*) as nb
from SERIES s inner join SERIES_PUBLICATION_TYPE spt
    on s.PUBLICATION_TYPE_ID=spt.ID
  inner join ISSUE i on i.SERIES_ID=s.ID
  inner join STORY st on st.ISSUE_ID=i.ID
  inner join LANGUAGE l on l.ID=s.LANGUAGE_ID
where spt.NAME='magazine'
group by l.ID, l.NAME
having count(*)>=10000
order by count(*) desc
```

	NAME	NB
1	English	768673
2	Norwegian	142573
3	Italian	30263
4	Dutch	25054
5	German	17661
6	Swedish	15807
7	French	11593

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

```
select st.NAME
from STORY_TYPE st
where st.ID not in
  (select distinct s.TYPE_ID
  from STORY s inner join ISSUE i
      on i.ID=s.ISSUE_ID
    inner join SERIES se on i.SERIES_ID=se.ID
    inner join COUNTRY c on se.COUNTRY_ID=c.ID
    inner join SERIES_PUBLICATION_TYPE spt
      on spt.ID=se.PUBLICATION_TYPE_ID
  where c.NAME='Italy'
    and spt.NAME='magazine')
```

	NAME
1	(backcovers) *do not use* / *please fix*
2	biography (nonfictional)
3	recap
4	statement of ownership
5	(unknown)

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

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

```
select c.NAME
from STORY_TYPE st inner join STORY s
    on st.ID=s.TYPE_ID
  inner join STORY_SCRIPT sc on s.ID=sc.STORY_ID
  inner join CONTRIBUTORS c on c.ID=sc.SCRIPT_ID
  inner join ISSUE i on s.ISSUE_ID=i.ID
where st.NAME='cartoon'
group by c.ID, c.NAME
having count(i.INDICIA_PUBLISHER_ID)>1
```

	NAME
1	Gene Ahern
2	Russell Cole
3	Frank Beck
4	Rube Goldberg
5	Chic Young
6	Ray Houlihan
7	SAM
8	Frank Doyle
9	Fritz Wilkinson
10	Robert Ripley

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

```
select bg.NAME
from BRAND_GROUP bg inner join INDICIA_PUBLISHER ip
  on bg.PUBLISHER_ID=ip.PUBLISHER_ID
group by bg.ID, bg.NAME
order by count(*) desc
fetch first 10 rows only
```

	NAME
1	2099
2	Disney Comics
3	Spider-Man Group
4	Curtis Magazines
5	Atlas
6	Razorline
7	Nelson
8	Pumping Iron
9	Crossgen
10	Thumbtack

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

```
select ROUND(AVG(ip.YEAR_ENDED-ip.YEAR_BEGAN), 3) AS average
from INDICIA_PUBLISHER ip
where ip.YEAR_BEGAN is not null
  and ip.YEAR_ENDED is not null
```

	AVERAGE
1	5.125

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

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

```
select ip.NAME
from SERIES s
  inner join INDICIA_PUBLISHER ip
    on s.PUBLISHER_ID=ip.PUBLISHER_ID
where s.FIRST_ISSUE_ID=s.LAST_ISSUE_ID
group by ip.ID, ip.NAME
order by count(distinct s.ID) desc
fetch first 10 rows only
```

	NAME
1	National Periodical Publications Inc.
2	National Periodical Publications
3	Tilsam Publications Inc.
4	Superman Inc.
5	Signal Publishing Company
6	Gainlee Publishing Co.
7	Jolaine Publications Inc.
8	J. R. Publishing Co.
9	William H. Wise \& Company
10	World`s Best Comics Company

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

```
select NAME
from
  (select ip.ID, ip.NAME, count(*) as nb
  from INDICIA_PUBLISHER ip
    inner join ISSUE i
      on ip.ID=i.INDICIA_PUBLISHER_ID
    inner join STORY s on i.ID=s.ISSUE_ID
    inner join STORY_SCRIPT sc
      on s.ID=sc.STORY_ID
  group by ip.ID, ip.NAME, s.ID
  order by count(*) desc)
group by ID, NAME
order by MAX(NB) desc
```

	NAME
1	Dell Publishing Co. Inc.
2	Raw Books \& Graphics
3	Random House
4	Andrews McMeel Publishing LLC
5	Houghton Mifflin Company
6	Harper \& Row Publishers
7	Kitchen Sink Press Inc.
8	Jabberwocky Graphix
9	Black Dog \& Leventhal Publishers
10	Rob Geertsen en Hanne Dewachter

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

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

```
select distinct c.NAME
from STORY s inner join ISSUE i on s.ISSUE_ID=i.ID
  inner join INDICIA_PUBLISHER ip
    on i.INDICIA_PUBLISHER_ID=ip.ID
  inner join STORY_CHARACTERS sc on s.ID=sc.STORY_ID
  inner join CHARACTERS c on sc.CHARACTER_ID=c.ID
where REGEXP_LIKE(ip.NAME, 'Marvel.*DC|DC.*Marvel')
  and c.ID in
  (select distinct sc.CHARACTER_ID
  from STORY s inner join ISSUE i on s.ISSUE_ID=i.ID
    inner join INDICIA_PUBLISHER ip
      on i.INDICIA_PUBLISHER_ID=ip.ID
    inner join STORY_CHARACTERS sc on s.ID=sc.STORY_ID
  where REGEXP_LIKE(ip.NAME, 'Marvel')
    and not REGEXP_LIKE(ip.NAME, 'DC'))
```

	NAME
1	Cyclops
2	Wolverine
3	Doctor Octopus
4	Glory Grant
5	The Leader
6	Batman
7	Alfred Pennyworth
8	Starfire
9	Shaper of Worlds
10	Toto

n) Print the top 5 series with most issues

```
select s.NAME
from ISSUE i inner join SERIES s on i.SERIES_ID=s.ID
group by s.ID, s.NAME
order by count(*) desc
fetch first 5 rows only
```

	NAME
1	Commando
2	Fliegende Blätter
3	Spirou
4	Robbedoes
5	The Beano

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

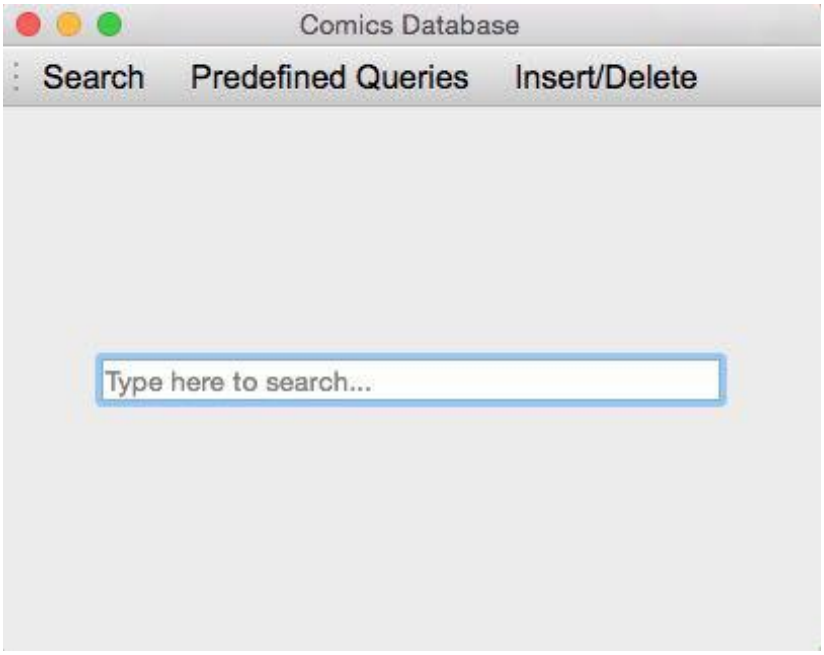
```
SELECT s.ID, s.TITLE
FROM STORY s inner join STORY_REPRINT sr on s.ID=sr.ORIGIN_ID
WHERE s.ISSUE_ID = 1234 -- INSERT GIVEN ISSUE ID
group by s.ID, s.TITLE
order by count(*) desc
fetch first 1 rows only
```

	ID	NAME
1	17915	Orchids of Doom

Mike Bardet 208714
Robin Lang 249582
Diana Petrescu 250971

Interface

We started the graphical interface using C++ with the Qt library. The database is not yet connected, we merely started working on the window with some buttons.



We chose the Qt library because it is a simple cross-platform graphical library that allows us to quickly create windows with any functionality through buttons, text fields, drop down menus and similar.
We decided to use C++ as the Qt library is designed mainly for this language.