***USAMA KHALID***

***19I-1236***

***SAOOD KHAN***

***19I-2128***

***Background of the Problem:***

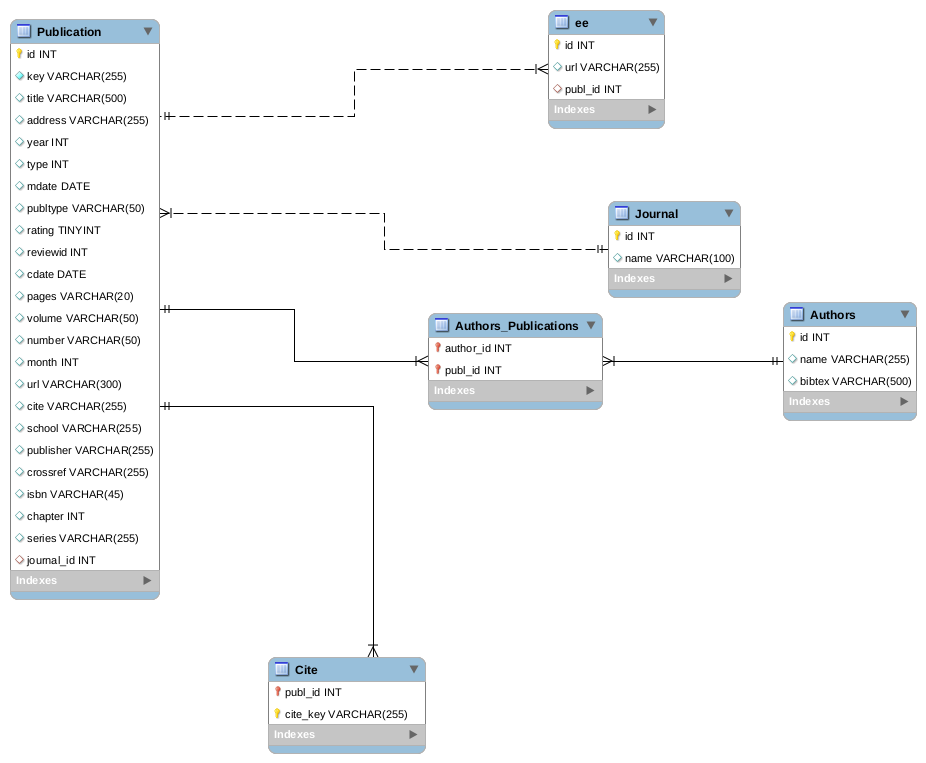
The main background of the problem is the lacking citation information and the varying coverage for different subfields of computer science. DBLP is imperfect for researchers as incomplete and inconsistent information, imperfect software, lack of time, are the limiting constraints for this task. DBLP isn’t a well designed project as it grew from a small scale experimental server to test web technology. In retrospect many ad hoc solutions are poorly designed.

If a person has several names (synonyms) or if there are several persons with the same name (homonyms) then the mapping becomes tricky. The main obstacles are to abbreviate given names beyond recognition and spelling errors. The main algorithmic idea is of the use to identify names we should check more precisely, is to look at person pairs which have the distance two in the co-author graph and have a “similar” name.

***Overview of the dataset and the domain:***

The dblp XML format is modelled after the BibTeX \*.bib file format. The format is defined in the DTD file in the same directory. By design our DTD is not very strict, as it makes no restriction to element order or multiplicity, and even allows nonsensical elements (e.g., ‹school› in <article> elements, <editor> and <author> elements at the same time that won’t be found in the actual dblp data set. Our priority was to keep the definition clean and simple, and not to model every aspect of the publication landscape.

***Proposed ER-Model:***

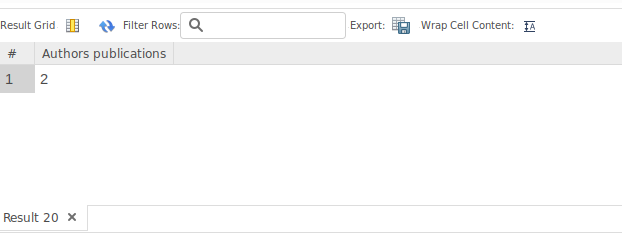


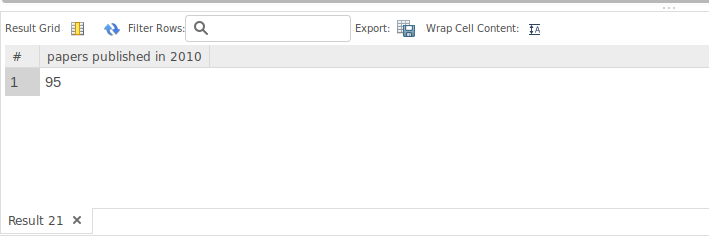
***Issues Faced and Solutions during Pre-Processing:***

* The main issues and problems that we faced is that the dblp.xml file was too large to read and search for data on windows.
  + **Solution** : We downloaded a third party software i.e. Xml Validatorbuddy to view and search this data.
* We assumed that the number tag in publications would always contain an integer but that wasn’t the case and we faced many issues while inserting data.
  + **Sloution** : We changed the data type of number to varchar.
* We assumed that the key attribute of publications would be unique so we made it the primary key but when we were inserting data there were many primary key violations because of this.
  + **Solution :** The solution to this issue is that we removed the key attribute of publications and we use another auto increment value as a primary key.
* Many tags defined in dblp.DTD are missing from dblp.xml.
  + **Solution :** To solve this issue we had to place many checks in our python code.
* We were reading the dblp data line by line the citation keys contained in a publication were most of which were below in the dblp data and hadn’t been read yet. So we had many foreign keys violations because of this.
  + **Solution :** So we accumulated all the citation in one list and inserted them in the end when all the dblp data has been inserted.
* One author can have many publications and one publication can have many authors. we didn’t realize this in the beginning.
* **Solution :** To solve this we made m to m relationship in db.

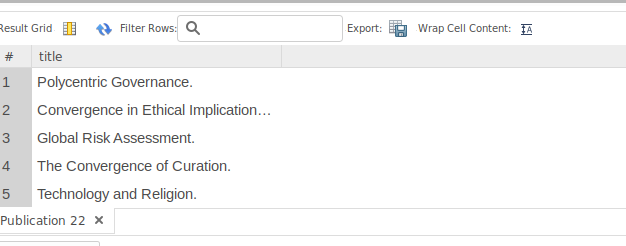
***Results of 5 different queries showing different statistics:***

SELECT count(\*) as Authors publications FROM Authors\_Publications where author\_id=10;

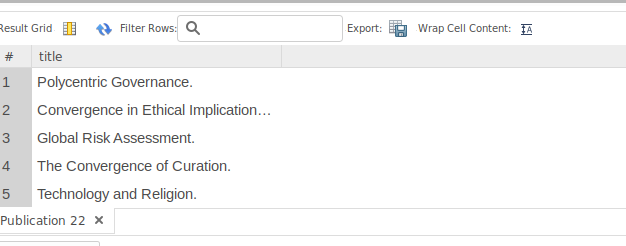
  
  
SELECT count(\*) as papers published in 2013 FROM Publication where year=2013;



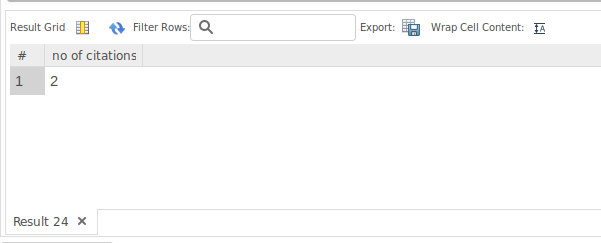
SELECT title FROM Publication where year=2016;



SELECT count(\*) as papers FROM Publication where journal\_id=22;



SELECT count(\*) as no of citations FROM Cite where publ\_id=1024;



***Programming Code:***

from lxml import etree  
import mysql.connector  
from datetime import datetime  
  
ELEMENTS = ['article', 'inproceedings', 'proceedings', 'book', 'incollection', 'phdthesis', 'mastersthesis', 'www']  
SUB\_ELEMENTS = ['author', 'editor', 'title', 'booktitle', 'pages', 'year', 'address', 'journal', 'volume', 'number',  
 'month', 'url', 'ee', 'cdrom', 'cite', 'publisher', 'note', 'crossref', 'isbn', 'series', 'school',  
 'chapter', 'publnr']  
ATTRIBUTES = ['key', 'mdate', 'publtype', 'reviewid', 'rating', 'cdate']  
  
month = ['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November',  
 'December']  
  
mydb = mysql.connector.connect(  
 host="localhost",  
 user="root",  
 passwd='12345678',  
 database="dblp"  
)  
  
mycursor = mydb.cursor(buffered=True)  
  
addpublicationsql = "INSERT INTO Publication (`key`, `title`, `address`, `year`,`type`,`mdate`,`publtype`,`rating`,`reviewid`,`cdate`, `pages`, `volume`, `number`, `month`, `url`, `cite`, `school`, `publisher`, `crossref`, `isbn`, `chapter`, `series`,`journal\_id`) " \  
 "VALUES (%(key)s, %(title)s, %(address)s, %(year)s,%(type)s,%(mdate)s,%(publtype)s,%(rating)s,%(reviewid)s,%(cdate)s, %(pages)s, %(volume)s, %(number)s, %(month)s, %(url)s, %(cite)s, %(school)s, %(publisher)s, %(crossref)s, %(isbn)s, %(chapter)s, %(series)s,%(journal\_id)s)"  
  
addeesql = "INSERT INTO ee (`url`,`publ\_id`) VALUES (%s,%s)"  
  
addcitessql = "INSERT INTO Cite (`cite\_key`,`publ\_id`) VALUES (%s,%s)"  
  
addjournalsql = "INSERT INTO Journal (`name`) VALUES (%s)"  
  
addauthorssql = "INSERT INTO Authors (`name`,`bibtex`) VALUES (%s,%s)"  
  
addauthorpublicationsql = "INSERT INTO Authors\_Publications (`author\_id`,`publ\_id`) VALUES (%s,%s)"  
  
searchauthorsql = "SELECT `id` FROM Authors WHERE name = %s"  
  
searchjournalsql = "SELECT `id` FROM Journal WHERE name = %s"  
  
count = 0  
  
for event, elem in etree.iterparse('dblp.xml', load\_dtd=True):  
 if elem.tag in ELEMENTS:  
  
 # Insert Publication information  
 data\_publication = {  
 'key': elem.get('key'),  
 'type': ELEMENTS.index(elem.tag),  
 'publtype': None,  
 'title': None,  
 'address': None,  
 'year': None,  
 'mdate': None,  
 'rating': None,  
 'reviewid': None,  
 'cdate': None,  
 'pages': None,  
 'volume': None,  
 'number': None,  
 'month': None,  
 'url': None,  
 'cite': None,  
 'school': None,  
 'publisher': None,  
 'crossref': None,  
 'isbn': None,  
 'chapter': None,  
 'series': None,  
 'journal\_id': None  
 }  
  
 data\_ee = []  
  
 data\_journal = []  
  
 data\_author = []  
  
 data\_author\_publication = []  
  
 data\_cite = []  
  
 for attribute in elem.keys():  
 value = elem.get(attribute)  
 if attribute == 'mdate' or attribute == 'cdate':  
 value = datetime.strptime(value, '%Y-%m-%d').date()  
  
 elif attribute == 'reviewid':  
 value = int(value)  
  
 data\_publication[attribute] = value  
  
 for e in SUB\_ELEMENTS:  
 data = elem.findall(e)  
  
 if not data:  
 continue  
  
 if e == 'ee':  
 for d in data:  
 data\_ee.append(d.text)  
  
 elif e == 'journal':  
 for d in data:  
 mycursor.execute(searchjournalsql, (d.text,))  
 if mycursor.rowcount == 0:  
 mycursor.execute(addjournalsql, (d.text,))  
 data\_publication[e] = mycursor.lastrowid  
 else:  
 data\_publication[e] = mycursor.fetchone()[0]  
  
 elif e == 'author':  
 for d in data:  
 mycursor.execute(searchauthorsql, (d.text,))  
 if mycursor.rowcount == 0:  
 mycursor.execute(addauthorssql, (d.text, d.get('bibtext')))  
 data\_author\_publication.append(mycursor.lastrowid)  
 else:  
 data\_author\_publication.append(mycursor.fetchone()[0])  
  
 elif e == 'cite':  
 for d in data:  
 if d.text != '...':  
 data\_cite.append(d.text)  
  
 elif e == 'month':  
 data\_publication[e] = month.index(data[0].text)  
  
 elif e in ['year', 'number', 'chapter']:  
 data\_publication[e] = int(data[0].text)  
  
 else:  
 data\_publication[e] = data[0].text  
  
 mycursor.execute(addpublicationsql, data\_publication)  
 latest\_publication\_id = mycursor.lastrowid  
 ee = []  
 for data in data\_ee:  
 ee.append((data,latest\_publication\_id))  
 mycursor.executemany(addeesql, ee)  
  
 author\_publication = []  
 for data in data\_author\_publication:  
 author\_publication.append((data,latest\_publication\_id))  
 mycursor.executemany(addauthorpublicationsql, author\_publication)  
  
 cite = []  
 for data in data\_cite:  
 cite.append((data,latest\_publication\_id))  
 mycursor.executemany(addcitessql, cite)  
 mydb.commit()  
  
 elem.clear()  
 count += 1  
 print("Inserted Publication: " + str(count) + " ")  
 print(data\_publication)  
 print(cite)  
  
mydb.close()