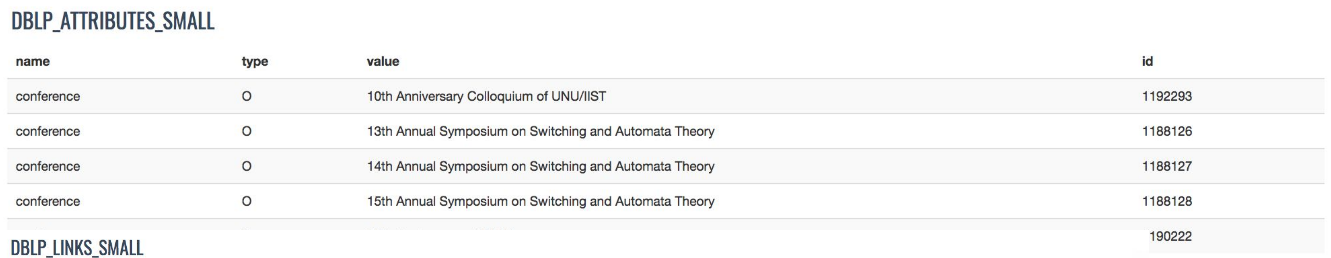
HBase

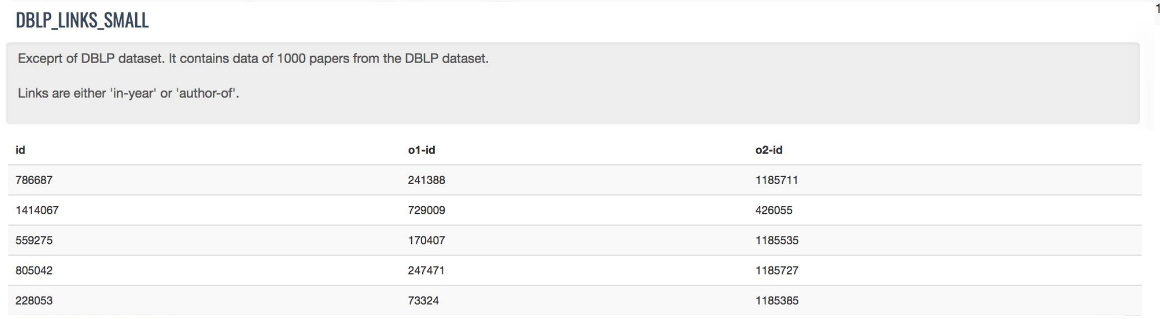
Andrew Gonser
Daniel Margason
Saral Bhagat
Winnie Jao
Xuanzhe Li

Source data



DBLP_ATTRIBUTES_SMALL

name	type	value	id
conference	O	10th Anniversary Colloquium of UNU/IIST	1192293
conference	O	13th Annual Symposium on Switching and Automata Theory	1188126
conference	O	14th Annual Symposium on Switching and Automata Theory	1188127
conference	O	15th Annual Symposium on Switching and Automata Theory	1188128
			190222

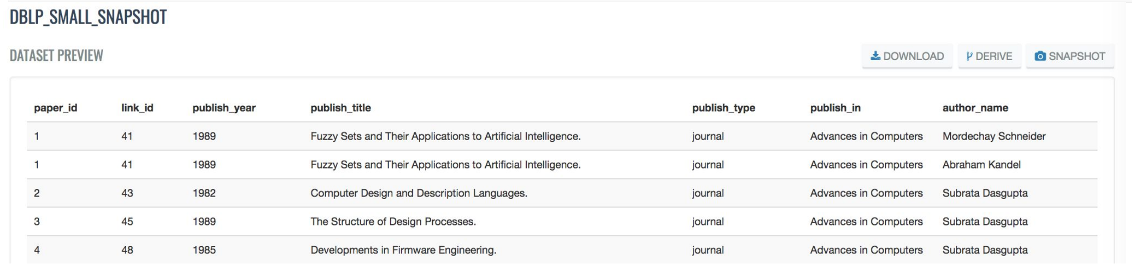


DBLP_LINKS_SMALL

Exceprt of DBLP dataset. It contains data of 1000 papers from the DBLP dataset.

Links are either 'in-year' or 'author-of'.

id	o1-id	o2-id
786687	241388	1185711
1414067	729009	426055
559275	170407	1185535
805042	247471	1185727
228053	73324	1185385



DBLP_SMALL_SNAPSHOT

DATASET PREVIEW

DOWNLOAD DERIVE SNAPSHOT

paper_id	link_id	publish_year	publish_title	publish_type	publish_in	author_name
1	41	1989	Fuzzy Sets and Their Applications to Artificial Intelligence.	journal	Advances in Computers	Mordechay Schneider
1	41	1989	Fuzzy Sets and Their Applications to Artificial Intelligence.	journal	Advances in Computers	Abraham Kandel
2	43	1982	Computer Design and Description Languages.	journal	Advances in Computers	Subrata Dasgupta
3	45	1989	The Structure of Design Processes.	journal	Advances in Computers	Subrata Dasgupta
4	48	1985	Developments in Firmware Engineering.	journal	Advances in Computers	Subrata Dasgupta

Final Design & Data Structure

Paper_info					
paper_id	link_id	publish_year	publish_type	publish_in	author_list

Author_info	
author_name	paper_list

Journal_info		
journal_name	proceeding_id	paper_list

Proceeding_info			
proceeding_id	proceeding_name	year	paper_list

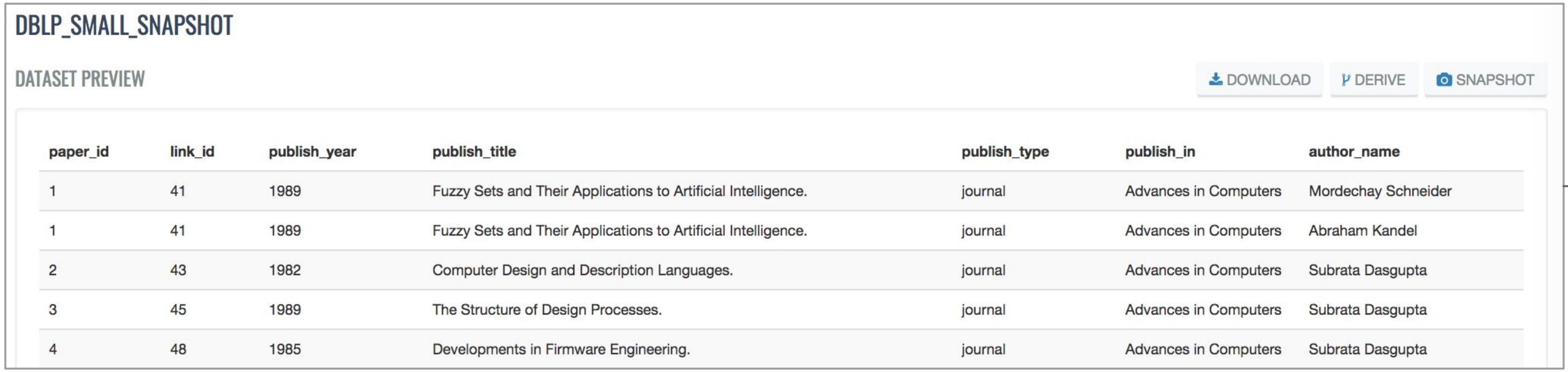
Variant & Design Objective

T1 Version

- Document type dataset
- Need attributed for store relationship in order to connect tables

T3 Version

- Column-family dataset
- Denormalized the relationship to internal structure for quicker scan

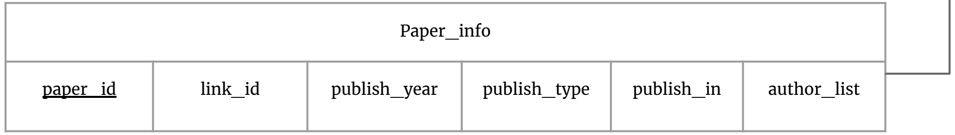

DBLP_SMALL_SNAPSHOT

DATASET PREVIEW

paper_id	link_id	publish_year	publish_title	publish_type	publish_in	author_name
1	41	1989	Fuzzy Sets and Their Applications to Artificial Intelligence.	journal	Advances in Computers	Mordechay Schneider
1	41	1989	Fuzzy Sets and Their Applications to Artificial Intelligence.	journal	Advances in Computers	Abraham Kandel
2	43	1982	Computer Design and Description Languages.	journal	Advances in Computers	Subrata Dasgupta
3	45	1989	The Structure of Design Processes.	journal	Advances in Computers	Subrata Dasgupta
4	48	1985	Developments in Firmware Engineering.	journal	Advances in Computers	Subrata Dasgupta

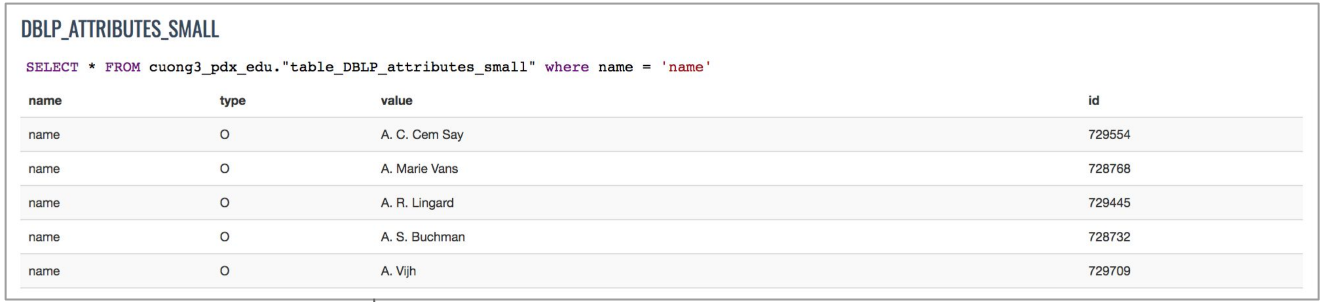
paper_id → paper_id #key
link_id → link_id
publish_year → publish_year
publish_type → publish_type
publish_in → publish_in

Search for all co-author for paper and store the result into author_list



Paper_info					
paper_id	link_id	publish_year	publish_type	publish_in	author_list

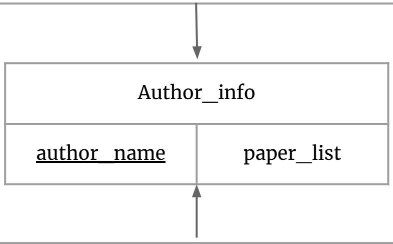
Paper_info



DBLP_ATTRIBUTES_SMALL

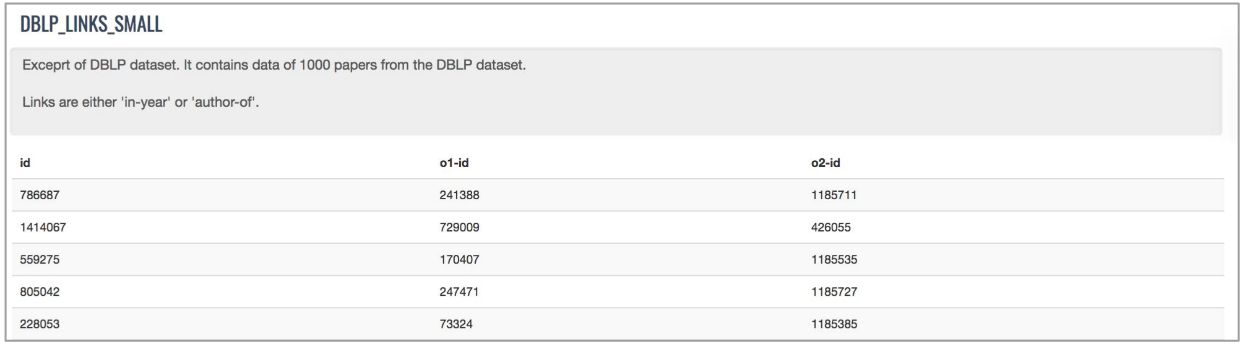
```
SELECT * FROM cuong3_pdx_edu."table_DBLP_attributes_small" where name = 'name'
```

name	type	value	id
name	O	A. C. Cem Say	729554
name	O	A. Marie Vans	728768
name	O	A. R. Lingard	729445
name	O	A. S. Buchman	728732
name	O	A. Vijh	729709



1. Use o1-id and o2-id from DBLP_LINKS_SMALL to find all paper that write by certain author in DBLP_ATTRIBUTES_SMALL
2. Store the result into paper_list

Use value in DBLP_ATTRIBUTE_SMALL Where attribute name is name as author_name as key.



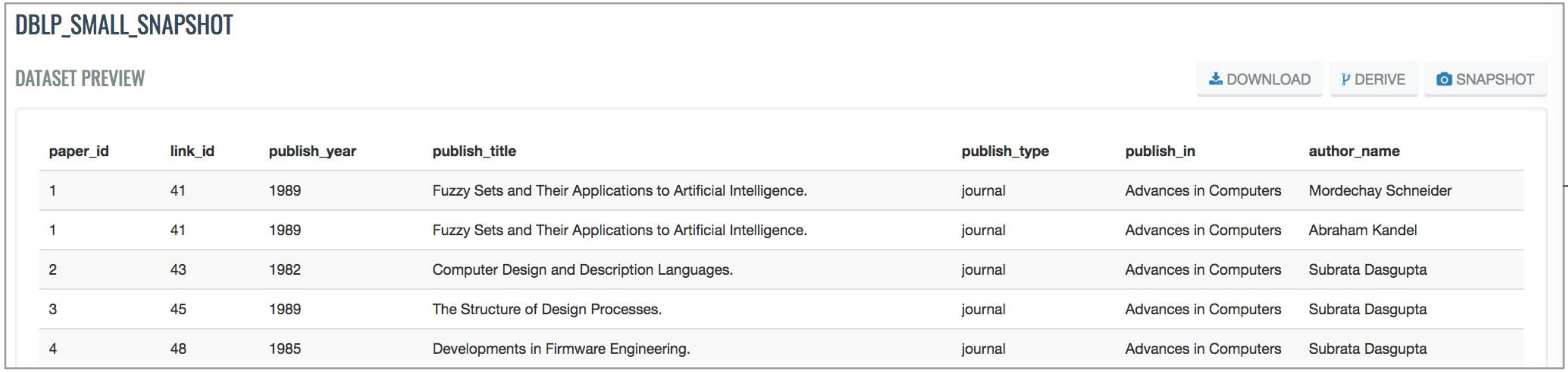
DBLP_LINKS_SMALL

Exceprt of DBLP dataset. It contains data of 1000 papers from the DBLP dataset.

Links are either 'in-year' or 'author-of'.

id	o1-id	o2-id
786687	241388	1185711
1414067	729009	426055
559275	170407	1185535
805042	247471	1185727
228053	73324	1185385

Author_info


DBLP_SMALL_SNAPSHOT

DATASET PREVIEW

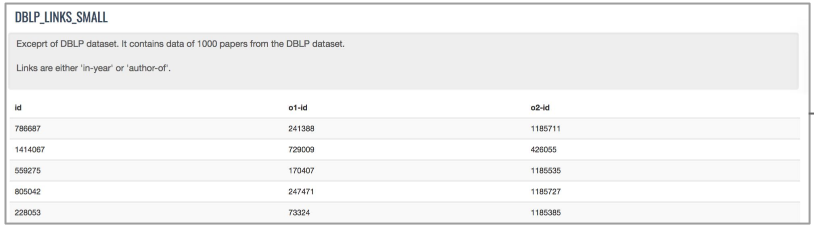
paper_id	link_id	publish_year	publish_title	publish_type	publish_in	author_name
1	41	1989	Fuzzy Sets and Their Applications to Artificial Intelligence.	journal	Advances in Computers	Mordechay Schneider
1	41	1989	Fuzzy Sets and Their Applications to Artificial Intelligence.	journal	Advances in Computers	Abraham Kandel
2	43	1982	Computer Design and Description Languages.	journal	Advances in Computers	Subrata Dasgupta
3	45	1989	The Structure of Design Processes.	journal	Advances in Computers	Subrata Dasgupta
4	48	1985	Developments in Firmware Engineering.	journal	Advances in Computers	Subrata Dasgupta

publish_in → journal_name #key

Search all paper that publish in certain journal and store the result into paper_list

Journal_info		
<u>journal_name</u>	proceeding_id	paper_list

Use link_id in DBLP_SMALL_SNAPSHOT to find the corresponding proceeding_id in DBLP_LINKS_SMALL

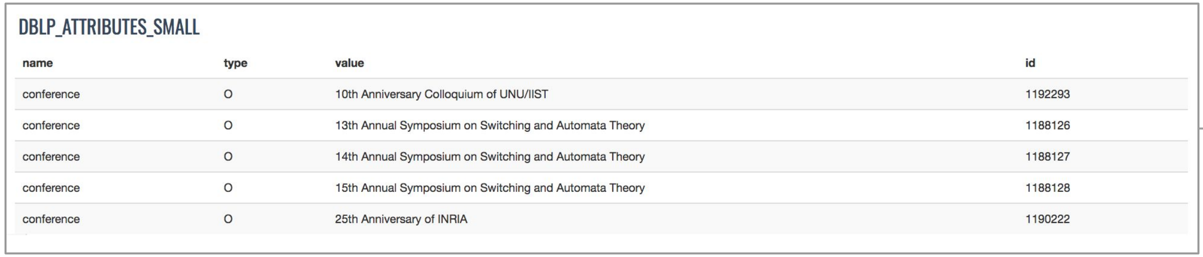

DBLP_LINKS_SMALL

Exceprt of DBLP dataset. It contains data of 1000 papers from the DBLP dataset.

Links are either 'in-year' or 'author-of'.

id	o1-id	o2-id
786687	241388	1185711
1414067	729009	426055
559275	170407	1185535
805042	247471	1185727
228053	73324	1185385

Journal_info

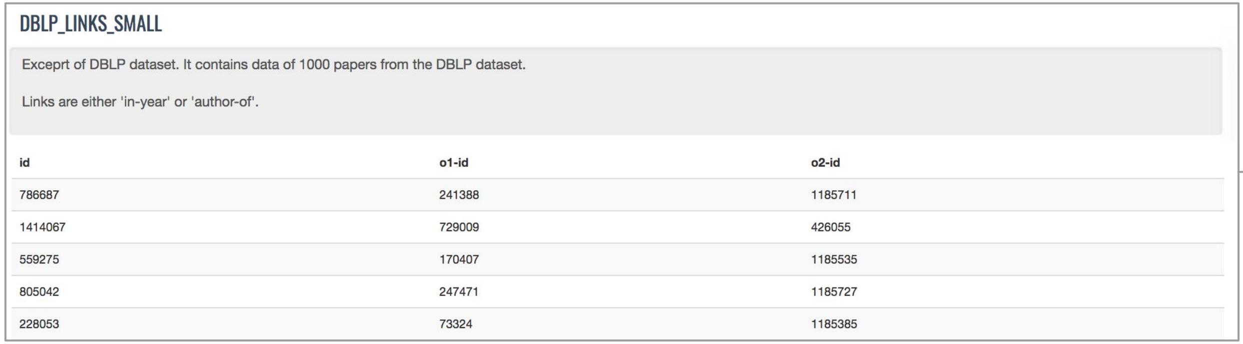


DBLP_ATTRIBUTES_SMALL

name	type	value	id
conference	O	10th Anniversary Colloquium of UNU/IIST	1192293
conference	O	13th Annual Symposium on Switching and Automata Theory	1188126
conference	O	14th Annual Symposium on Switching and Automata Theory	1188127
conference	O	15th Annual Symposium on Switching and Automata Theory	1188128
conference	O	25th Anniversary of INRIA	1190222

Id → proceeding__id # key
value → proceeding__name # key

Proceeding__info			
<u>proceeding__id</u>	<u>proceeding__name</u>	year	paper__list



DBLP_LINKS_SMALL

Exceprt of DBLP dataset. It contains data of 1000 papers from the DBLP dataset.

Links are either 'in-year' or 'author-of'.

id	o1-id	o2-id
786687	241388	1185711
1414067	729009	426055
559275	170407	1185535
805042	247471	1185727
228053	73324	1185385

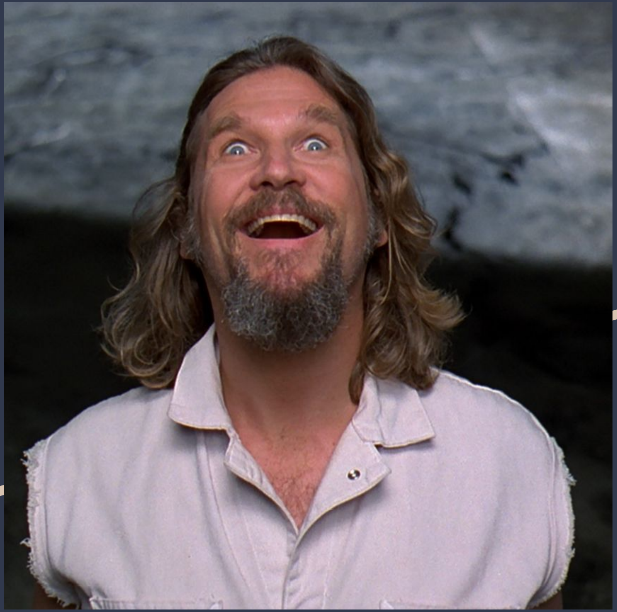
1. Use o1-id and o2-id from DBLP__LINKS__SMALL to find all the info of that paper in certain proceeding in DBLP__ATTRIBUTES__SMALL

2. Use o1-id and o2-id from DBLP__LINKS__SMALL to find Published year for the paper in DBLP__ATTRIBUTES__SMALL

3. Store the result into year and paper__list

Proceed__info Dataset

Process to load into Hbase



Technology

-Python (and associated packages)

-HappyBase && HBase API

Implementation

1) PreProcess data based on schema requirements

2) Extract processed data into TSV format

3) Convert CSV file to TSV

4) Assign columns to existing column family and define schema in bulk call

5) Loader will handle transformation to HFile format

6) Use bulk load tsv load for HBase:

```
org.apache.hadoop.hbase.mapreduce.ImportTsv
```

Query Implementation Strategies

Present:

- Find the paper with the most co-authors.
- FInd out at what level is Moshe Vardi from Joseph M. Hellerstein.
- Is the DBPL graph connected

Not present:

- Find out how many 3rd level co-authors does David DeWitt have.
- Which proceeding in 2005 had the most distinct number of authors?
- Which author participates in the most triangles.

```
connect = happybase.connect() # connect with database
If connect is not built: # check connection
    Return 0
table_paper_count = {(i, count)}        # dictionary to hold index(hashed by paper id) and count of the co-authors of the paper
author_access = table.rows(author:author_name)       # access to author_name column in author family column
for each author, list of paper id in author_access:        # loop through each author_name in order to access the list of paper each author has written/ co-written
    list = list of paper id    # list of paper ids each author has written/ co-written
    for each paper in list:         # loop through each paper id
        index = hash(paper)  # hash paper id to get index for the dictionary
        table_paper_count.insert(index, table_paper_count.get(index)+1)        # increase the count of co-authors for each paper in the dictionary

    Return max(table_paper_count)        # return the maximum count of co-authors
```

Find the paper with the most co-authors.

```
connect = Hbase()          # connect with the database
if connect is not built:   # check if connection is built
        return 0

temp = []
author = Table
                                                                                                #checking the base case
moshe=author.row(row_key=’Moshe Vardi’)
hellerstein = author.row(row_key = ‘Joseph M. Hellerstein)

if set(moshe.list_paper_id).intersection(hellerstein.list_paper_id):
        print(‘Level 1 Co-author’)

listOfCoauthors = create_list_of_coauthors(author, moshe.list_paper_id)
result_set = listOfCoauthors
count = 1
While(not found)
        For s in result_set:                                                                    #Creates a combined list of
                temp = author.row(row_key = listOfCoauthor[s])                                  # co-authors from authors in
                tempSet = create_list_of_co_authors(author, temp.list_paper_id)                 # the results_set
```

FInd out at what level is Moshe Vardi from Joseph M. Hellerstein.

(continue...)

```
                result_set = result_set + tempSet
        if len(set(result_set).intersection([‘JosephM. Hellerstein’]):      #Hey Joe! Checks to see if Joe is in the set
                found = true
        count = count + 1

print(‘Co-Author Level: ‘, count)

                                                    #A function definition that creates a list of coauthors for a
                                                                       #given list of paper ids and author table.
Def Create_list_of_co_authors(author,list_of_paper_ids)
        For row in author.scan(column = list_of_paper_id):
                co_author_list = []
                if len(set(list_of_paper_ids).intersection(author.list_of_paper_id) <> 0:
                        co_author_list.append(author.name)
                Return co_author_list
```

FInd out at what level is Moshe Vardi from Joseph M. Hellerstein. (cont.)

```
connect = happybase.connect() # connect with database
If connect is not built:     # check connection
          Return 0
Connected_Flag= False
Not_Connected =[Paper_info, author_info, journal_Info, Proceeding_info]
Connect=[ author_info]
author_access = table.rows(author:author_name)  # access to author_name column in author column family
for author_name, list of paper id in author_access:          # loop through author_name
          list = list of paper id        # list of paper ids each author has written/ co-written

                    for each paper_id in list:           # loop through each paper id
                              paper_access = table.rows(Paper_info :paper_id)
                                        if paper_acess[publish_type]== Journal # Checking if paper in Journal
                                                  journal_access = table.rows(Journal_info:Journal_name)
                                                            if journal_access[proceeding_id]!=null # Checking if Journal is in Proceding
                                                                      Connected_Flag=True
If Connected_Flag==True
          Print (“graph is connected”)
```

Is the DBPL graph connected

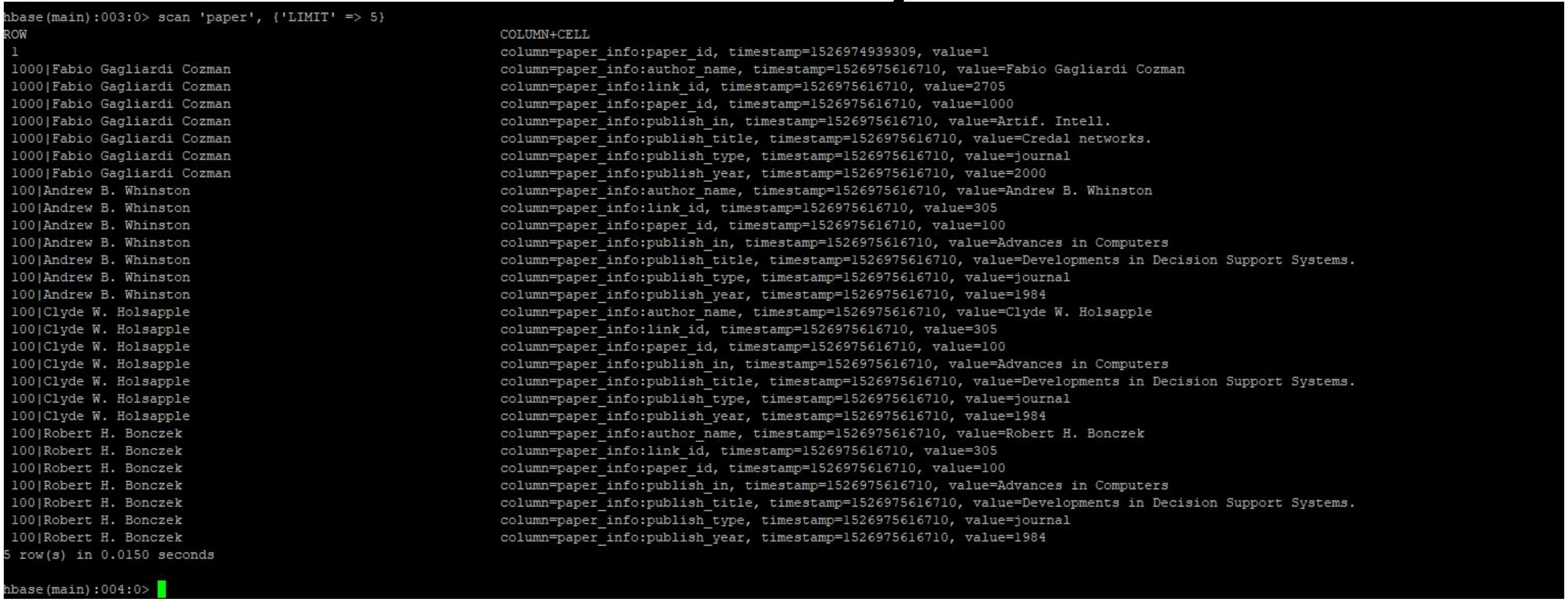
```
Set={{None,None,None}} # I used set here, because set eliminates duplicates by itself. Set has no duplicates.
author_access = table.rows(author:author_name)      # access to author name column in author column family
for each author_name, list of paper id in author_access:                # loop through
          listofpapers = list of paper id  # list of paper ids each author has written/ co-written

          for I,J pair of paper_id in listofpapers:                    # loop through each pair of paper ids
                    I.paper_access = table.rows(Paper_info :I.paper_id)
                    J.paper_access = table.rows(Paper_info :J.paper_id)

                    I.co_author_list=I.paper_access[ author_list]
                    J.co_author_list=I.paper_access[ author_list]
                    For every object in I.co_author_list
                              I.author_access = table.rows(author:author_name==I.co_author_list[Object])
                              I.Object.ListOfPapers= I.author_access.list of paper id
                    For every object in J.co_author_list
                              J.author_access = table.rows(author:author_name==J.co_author_list[Object])
                              J.Object.ListOfPapers= I.author_access.list of paper id
                    For every Paper_id in I.Object.ListOfPapers
                              For every Paper_id in J.Object.ListOfPapers
                                        If J.Object.ListOfPapers.paper_id==I.Object.ListOfPapers.paper_id
                                        Set.sort(Set.add(author_name,I.author_name,J_author-name)) # Addition of Author names in triangle into set
          List=[ ]
          For every object in Set      # Loop to add all entries of Set in a List
                    List.add(Set[i].[1])
                    List.add(Set[i].[2])
                    List.add(Set[i].[3])
max(list,key=list.count) # This function returns the Name with Maximum count in a list.
```

Which author participates in the most triangles.

Thank you!



```
hbase(main):003:0> scan 'paper', {'LIMIT' => 5}
ROW                                                         COLUMN+CELL
 1                                                          column=paper_info:paper_id, timestamp=1526974939309, value=1
 1000|Fabio Gagliardi Cozman                                column=paper_info:author_name, timestamp=1526975616710, value=Fabio Gagliardi Cozman
 1000|Fabio Gagliardi Cozman                                column=paper_info:link_id, timestamp=1526975616710, value=2705
 1000|Fabio Gagliardi Cozman                                column=paper_info:paper_id, timestamp=1526975616710, value=1000
 1000|Fabio Gagliardi Cozman                                column=paper_info:publish_in, timestamp=1526975616710, value=Artif. Intell.
 1000|Fabio Gagliardi Cozman                                column=paper_info:publish_title, timestamp=1526975616710, value=Credal networks.
 1000|Fabio Gagliardi Cozman                                column=paper_info:publish_type, timestamp=1526975616710, value=journal
 1000|Fabio Gagliardi Cozman                                column=paper_info:publish_year, timestamp=1526975616710, value=2000
 100|Andrew B. Whinston                                     column=paper_info:author_name, timestamp=1526975616710, value=Andrew B. Whinston
 100|Andrew B. Whinston                                     column=paper_info:link_id, timestamp=1526975616710, value=305
 100|Andrew B. Whinston                                     column=paper_info:paper_id, timestamp=1526975616710, value=100
 100|Andrew B. Whinston                                     column=paper_info:publish_in, timestamp=1526975616710, value=Advances in Computers
 100|Andrew B. Whinston                                     column=paper_info:publish_title, timestamp=1526975616710, value=Developments in Decision Support Systems.
 100|Andrew B. Whinston                                     column=paper_info:publish_type, timestamp=1526975616710, value=journal
 100|Andrew B. Whinston                                     column=paper_info:publish_year, timestamp=1526975616710, value=1984
 100|Clyde W. Holsapple                                     column=paper_info:author_name, timestamp=1526975616710, value=Clyde W. Holsapple
 100|Clyde W. Holsapple                                     column=paper_info:link_id, timestamp=1526975616710, value=305
 100|Clyde W. Holsapple                                     column=paper_info:paper_id, timestamp=1526975616710, value=100
 100|Clyde W. Holsapple                                     column=paper_info:publish_in, timestamp=1526975616710, value=Advances in Computers
 100|Clyde W. Holsapple                                     column=paper_info:publish_title, timestamp=1526975616710, value=Developments in Decision Support Systems.
 100|Clyde W. Holsapple                                     column=paper_info:publish_type, timestamp=1526975616710, value=journal
 100|Clyde W. Holsapple                                     column=paper_info:publish_year, timestamp=1526975616710, value=1984
 100|Robert H. Bonczek                                      column=paper_info:author_name, timestamp=1526975616710, value=Robert H. Bonczek
 100|Robert H. Bonczek                                      column=paper_info:link_id, timestamp=1526975616710, value=305
 100|Robert H. Bonczek                                      column=paper_info:paper_id, timestamp=1526975616710, value=100
 100|Robert H. Bonczek                                      column=paper_info:publish_in, timestamp=1526975616710, value=Advances in Computers
 100|Robert H. Bonczek                                      column=paper_info:publish_title, timestamp=1526975616710, value=Developments in Decision Support Systems.
 100|Robert H. Bonczek                                      column=paper_info:publish_type, timestamp=1526975616710, value=journal
 100|Robert H. Bonczek                                      column=paper_info:publish_year, timestamp=1526975616710, value=1984
5 row(s) in 0.0150 seconds

hbase(main):004:0>
```

Loaded Data (Screenshot)

```
connect = Hbase()       # connect with the database
if connect is not built:    # check if connection is built
        return 0
level_1 = []          # list of level 1 co-author
level_2 = []          # list of level 2 co-authors
level _3 = 0# count of level_3 co-authors
author_access = table.rows(author:author_name)     # access to author_name column in author column family
for name, list of paper id in author_access:             # loop through author_name for level 1 co-author
        if name is "David DeWitt":      # check if the name is "David DeWitt"
                list = list of paper id              # get the list of paper ids from "David DeWitt"
                for id, list of author name in list:      # loop through the ids from "David DeWitt"
                        for name_1, list of paper id_1 in list of author name:          # loop through list of author name to build list of level 1 author of "David DeWitt"
                                if name_1 is not "David DeWitt":     # if the author writes the same paper with "David DeWitt" and he/she is not "David DeWitt"
                                        level_1.append(name_1)      # add the co-author's into level_1 list
                                        for id_1, list of author name_2 in list of paper id_1:    # loop through each paper that level 1 co-authors have written
```

Find out how many 3rd level co-authors does David DeWitt have.

(Continue...)

```
for name_2, list of paper id_2 in list of author name_2:      # loop through list of second level co-authors
    if name_2 is not in level_1 && name_2 is not “David DeWitt”:      # if the second level co-author is not David DeWitt and he/she is not in the list of level one co-author
        level_2.append(name_2)      # add the found level 2 co-author to level_2 list
        for id_2, list of author name_3 in list of paper id_3:    # loop through ids from level 2 co-authors
            for name_3 in list of author name_3:      # loop through list of author name from level 2 co-authors’ paper list
                if name_3 is not “David DeWitt” && name_3 is not in level_1 and level_2:    # if the third level co-author is not David DeWitt and he/she is not in the list of level one and level two co-author
                    ++level_3   # increment the count for level 3 co-author count
return level_3      # return the count of level 3 co-authors
```

```
connect = Hbase()         # connect with the database
if connect is not built:  # check if connection is built
        return 0

proceedings = Table
author = Table
proPaperList = []
                                        #Puts the list of paper IDs and row ids of a proceeding from 2005, into a list.

for 2005, data in proceedings.scan(column = year):
        tempList = []
        tempList.append(row_id)
        tempList.append(proceedings.list_of_paper_id)
        proPaperList.append(tempList)

(Continue...)
```

Which proceeding in 2005 had the most distinct number of authors?

(Continue...)

```
author_count, max_author_count, i = 0
top_proceeding_row_id = 0
                                        #Scans the rows of authors and determining if an author wrote
                                                              #at least one paper in the proceeding.
For element in proPaperList:
    For row in author.scan(column = list_of_paper_id):
        if len(set(proPaperList[i][1]).intersection(row.list_of_paper_id) <> 0
            author_count=author_count+1
                                                              #Checks to see if there is a new max
                                                                   #and then store the id if so
    if author_count > max_author_count:
        max_author_count = author_count
        top_proceeding_row_id = tempList[i][0]

print(proceedings.row(top_proceeding_row_id))
```

Which proceeding in 2005 had the most distinct number of authors? (cont.)