Pig Latin Examples and Execution Instructions

ADVANCED DATABASE MANAGEMENT
CS 5513
UNIVERSITY OF OKLAHOMA

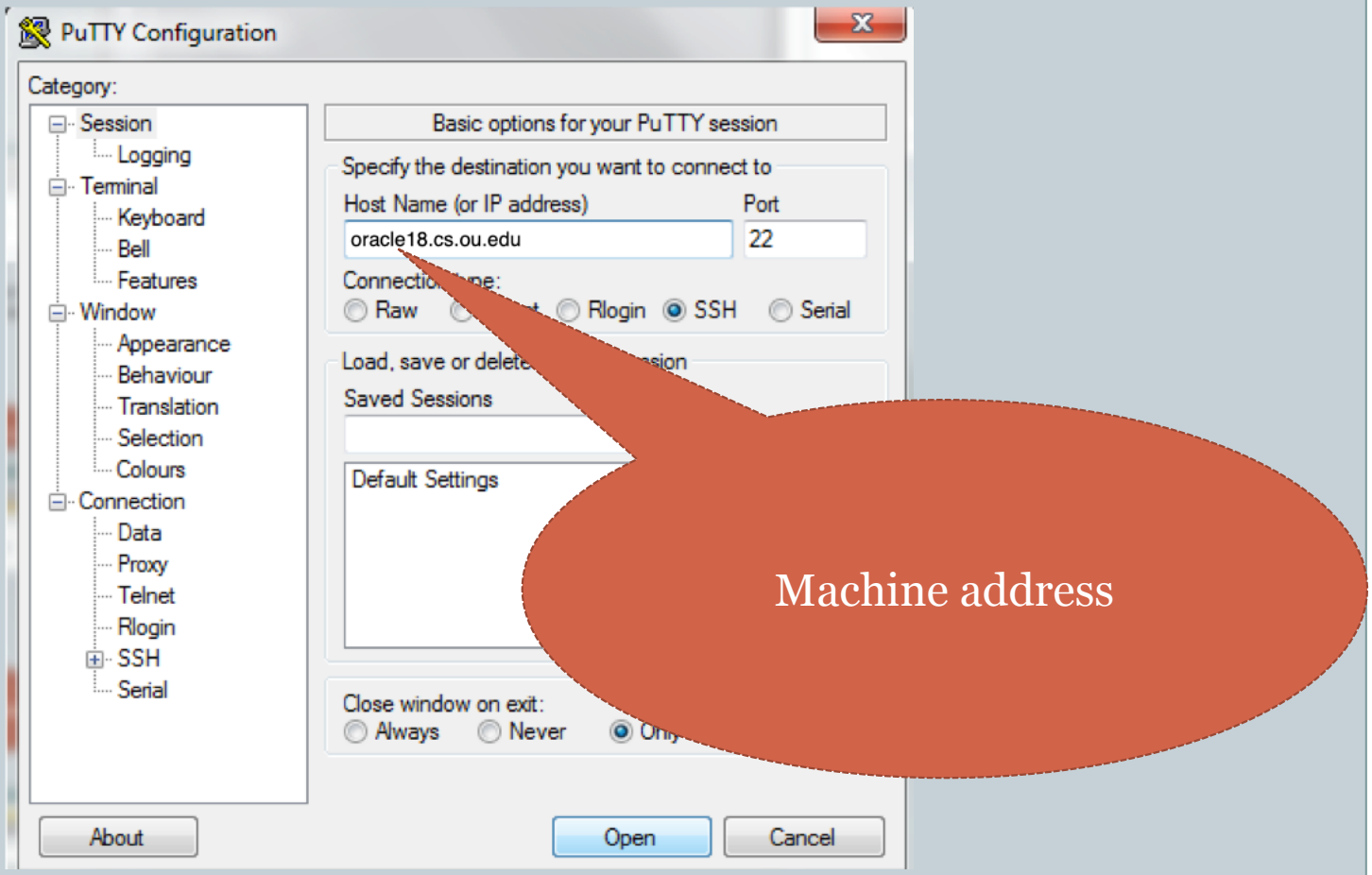
Contents

Connecting to CSN computer

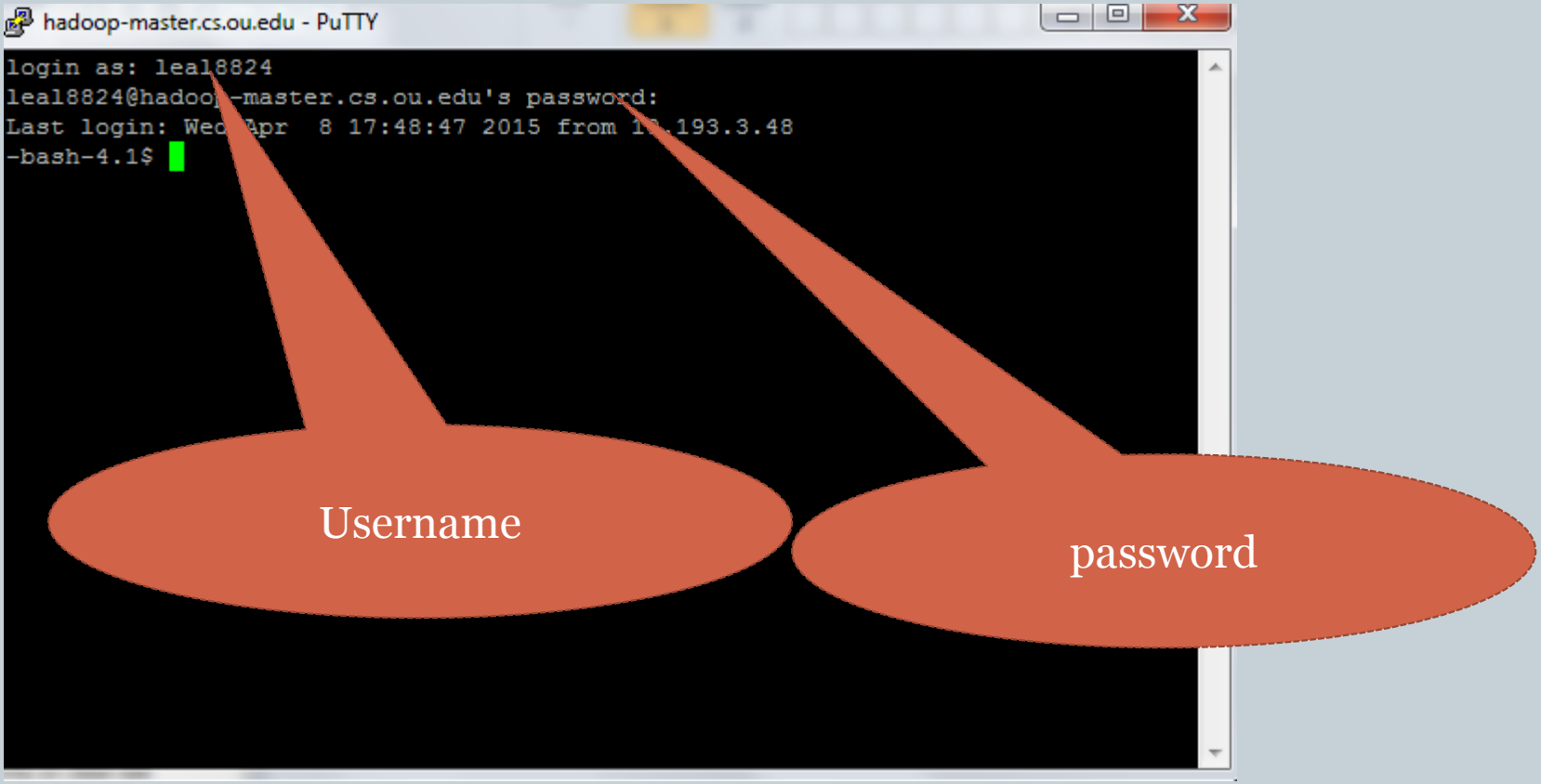
- Pig is running on oracle18.cs.ou.edu
- Connect to oracle18.cs.ou.edu using ssh client
- On Unix, use ssh command in your terminal
- On Windows, we recommend using Putty for ssh client connection
  - First open putty
  - Choose the machine address
  - Provide username and password

Putty can be found here
https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

Putty configuration screen



Putty username and password



Running Pig

- Pig can be run in two ways
  - pig –x local
    - This is for running pig only in a local machine
    - This execution does not require Hadoop
    - **Use this option for Homework 4!**
  - pig –x mapreduce
    - This is for running pig with Hadoop to handle map reduce framework
    - This execution requires Hadoop
  - If Pig is running, you will see a prompt with “grunt”

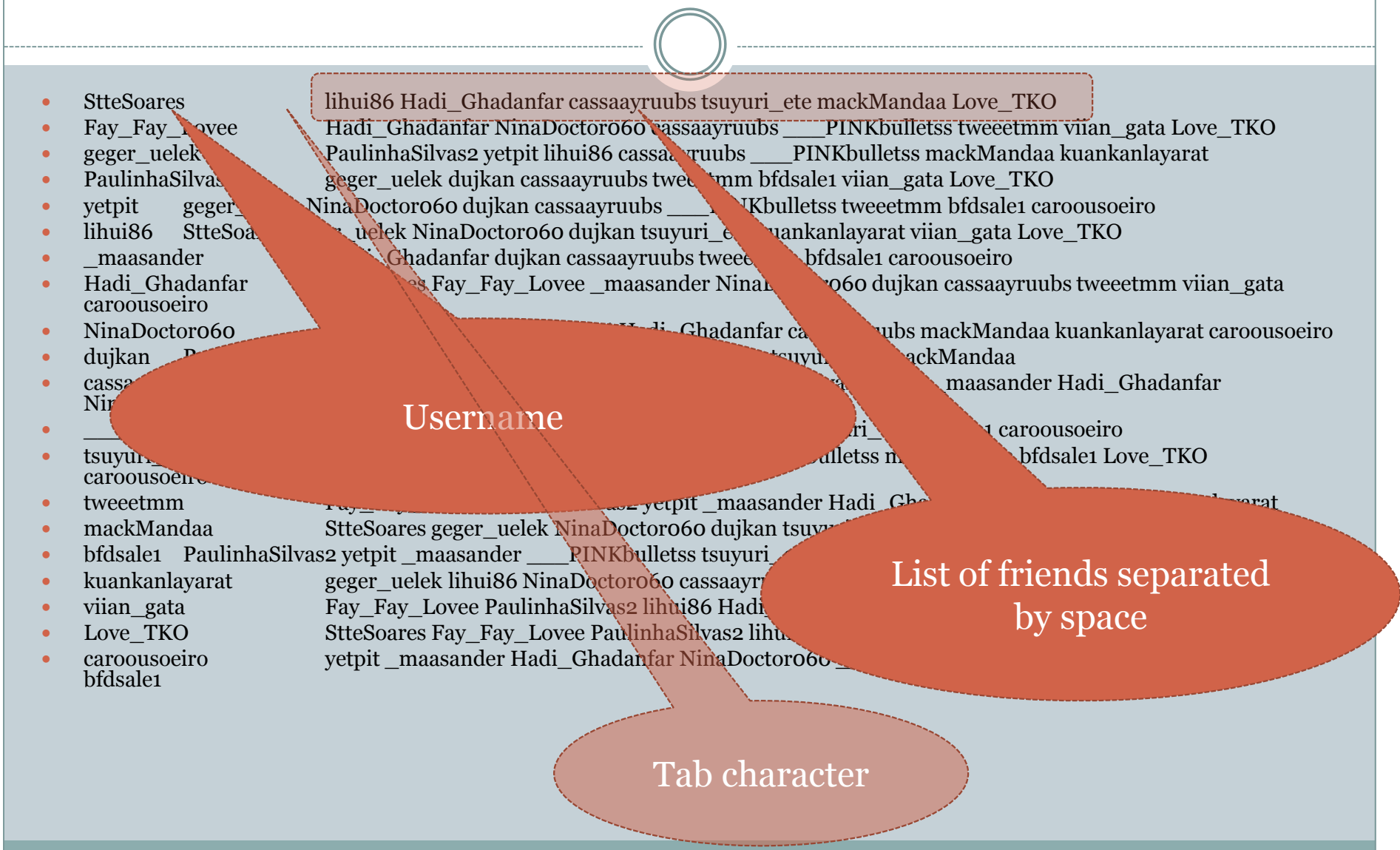
Terminal output

```
[[basi5906@oracle18 ~]$ pig -x local                                                ]
19/04/02 13:46:05 INFO pig.ExecTypeProvider: Trying ExecType : LOCAL
19/04/02 13:46:05 INFO pig.ExecTypeProvider: Picked LOCAL as the ExecType
2019-04-02 13:46:05,621 [main] INFO  org.apache.pig.Main - Apache Pig version 0.17.0 (r1797386) compi
led Jun 02 2017, 15:41:58
2019-04-02 13:46:05,621 [main] INFO  org.apache.pig.Main - Logging error messages to: /home/basi5906/
pig_1554230765617.log
2019-04-02 13:46:05,636 [main] INFO  org.apache.pig.impl.util.Utils - Default bootup file /home/basi5
906/.pigbootup not found
2019-04-02 13:46:05,710 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - mapred.job.tr
acker is deprecated. Instead, use mapreduce.jobtracker.address
2019-04-02 13:46:05,712 [main] INFO  org.apache.pig.backend.hadoop.executionengine.HExecutionEngine -
 Connecting to hadoop file system at: file:///
2019-04-02 13:46:05,905 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.
checksum is deprecated. Instead, use dfs.bytes-per-checksum
2019-04-02 13:46:05,919 [main] INFO  org.apache.pig.PigServer - Pig Script ID for the session: PIG-de
fault-72f33d93-2fe0-4648-9d71-37dad9fea635
2019-04-02 13:46:05,919 [main] WARN  org.apache.pig.PigServer - ATS is disabled since yarn.timeline-s
ervice.enabled set to false
grunt>
```

Pig references

- http://pig.apache.org/docs/latest/
- http://pig.apache.org/docs/latest/basic.html

Sample input file



Example Problem

- We are going to find the list of common friends between two persons

Loading the input file

- DATA = load 'fsample.txt';
- Pig will load the file and save into alias ‘DATA’
- ‘DATA’ is called alias in PigLatin.
- By default,
  - Pig assumes ‘\t’ as separating character
  - Hence, DATA has two items
    - Person name
    - List of friends as chararray (PigLatin’s version of String)
    - Pig has a default indexing for each attribute
      - The first attribute is referred to as $0
      - The second attribute is referred to as $1 and so on…

Output

```
grunt> DUMP DATA;
(StteSoares,lihui86 Hadi_Ghadanfar cassaayruubs tsuyuri_ete mackMandaa Love_TKO)
(Fay_Fa_Lovee,Hadi_Ghadanfar NinaDoctor060 cassaayruubs ___PINKbulletss tweeetmm viian_gata Love_TKO
)
(geger_u_lek,PaulinhaSil_s2 yetpit lihui86 cassaayruubs ___PINKbulletss mackMandaa kuankanlayarat)
(Paulinh_ilvas2,geger_uel_ dujkan cassaayruubs tweeetmm bfdsale1 viian_gata Love_TKO)
(yetpit,_er_uelek NinaDoct_060 dujkan cassaayruubs ___PINKbulletss tweeetmm bfdsale1 caroousoeiro)
(lihui86,_teSoares geger_uel_ NinaDoctor060 dujkan tsuyuri_ete kuankanlayarat viian_gata Love_TKO)
(_maasand_ Hadi_Ghadanfar dujk_ assaayruubs tweeetmm bfdsale1 caroousoeiro)
(Hadi_Gha_ar,StteSoares Fay_Fa_vee _maasander NinaDoctor060 dujkan cassaayruubs tweeetmm viian_
gata caro_iro)
(NinaDocto_ Fay_Fay_Lovee yetpit l_6 Hadi_Ghadanfar cassaayruubs mackMandaa kuankanlayarat car
oousoeiro)
(dujkan,Pa_Silvas2 yetpit lihui86 _m_der Hadi_Ghadanfar tsuyuri_ete mackMandaa)
(cas_Fay_Fay_Lovee geg_Silvas2 yetpit _maasander Hadi_Ghadanfar N
i_tsuyuri_et_
_geger_uri_ete bfdsale1 caroousoeiro)
_6 dujk_mackMandaa bfdsale1 Love_TKO caro
o_
(twee_PaulinhaSilvas2 _Ghadanfar cassaayruubs kuankanlayarat)
(mackMandaa,StteSoares geger_uelek NinaDoctor06_ _tsuyuri_ete bfdsale1 viian_gata caroousoeiro)
(bfdsale1,PaulinhaSilvas2 yetpit _maasander ___PINKbulletss tsuyuri_ete mackMandaa viian_gata Love_TK
O caroousoeiro)
(kuankanlayarat,geger_uelek lihui86 NinaDoctor060 cassaayruubs tweeetmm Love_TKO)
(viian_gata,Fay_Fay_Lovee PaulinhaSilvas2 lihui86 Hadi_Ghadanfar mackMandaa bfdsale1)
(Love_TKO,StteSoares Fay_Fay_Lovee PaulinhaSilvas2 lihui86 tsuyuri_ete bfdsale1 kuankanlayarat)
(caroousoeiro,yetpit _maasander Hadi_Ghadanfar NinaDoctor060 ___PINKbulletss tsuyuri_ete mackMandaa b
fdsale1)
grunt>
```

Person name

Person's friends list

Individual friends

- Remember, the list of friends is chararray
- So, we want to make a list of person ids from chararray
- DATA2 = FOREACH DATA GENERATE $0, TOKENIZE($1);
  - We created DATA2
    - Each entry of DATA2 is a person of DATA and a "bag!"
    - Bag is Pig Latin's version of List
    - Each bag contains a list of friends
    - TOKENIZE, by default, splits the chararray into a bag of friends

FOREACH means for every item in the relation DATA

GENERATE tells what you want to create

Output

```
grunt> DUMP DATA2;
```

```
(StteSoares,{(lihui86),(Hadi_Ghadanfar),(cassaayruubs),(tsuyuri_ete),(mackMandaa),(Love_TKO)})
(Fay_Fay_Lovee,{(Hadi_Ghadanfar),(NinaDoctor060),(cassaayruubs),(___PINKbulletss),(tweeetmm),(viian_g
ata),(Love_TKO)})
(geger_uelek,{(PaulinhaSilvas2),(yetpit),(lihui86),(cassaayruubs),(___PINKbulletss),(mackMandaa),(kua
nkanlayarat)})
(PaulinhaSilvas2,{(geger_uelek),(dujkan),(cassaayruubs),(tweeetmm),(bfdsale1),(viian_gata),(Love_TKO)
})
(yetpit,{(geger_uelek),(NinaDoctor060),(dujkan),(cassaayruubs),(___PINKbulletss),(tweeetmm),(bfdsale1
),(caroousoeiro)})
(lihui86,{(StteSoares),(geger_uelek),(NinaDoctor060),(dujkan),(tsuyuri_ete),(kuankanlayarat),(viian_g
ata),(Love_TKO)})
(_maasander,{(Hadi_Ghadanfar),(dujkan),(cassaayruubs),(tweeetmm),(bfdsale1),(caroousoeiro)})
(Hadi_Ghadanfar,{(StteSoares),(Fay_Fay_Lovee),(_maasander),(NinaDoctor060),(dujkan),(cassaayruubs),(t
weeetmm),(viian_gata),(caroousoeiro)})
(NinaDoctor060,{(Fay_Fay_Lovee),(yetpit),(lihui86),(Hadi_Ghadanfar),(cassaayruubs),(mackMandaa),(kuan
kanlayarat),(caroousoeiro)})
(dujkan,{(PaulinhaSilvas2),(yetpit),(lihui86),(_maasander),(Hadi_Ghadanfar),(tsuyuri_ete),(mackMandaa
)})
(cassaayruubs,{(StteSoares),(Fay_Fay_Lovee),(geger_uelek),(PaulinhaSilvas2),(yetpit),(_maasander),(Ha
di_Ghadanfar),(NinaDoctor060),(___PINKbulletss),(tsuyuri_ete),(tweeetmm),(kuankanlayarat)})
(___PINKbulletss,{(Fay_Fay_Lovee),(geger_uelek),(yetpit),(cassaayruubs),(tsuyuri_ete),(bfdsale1),(car
oousoeiro)})
(tsuyuri_ete,{(StteSoares),(lihui86),(dujkan),(cassaayruubs),(___PINKbulletss),(mackMandaa),(bfdsale1
),(Love_TKO),(caroousoeiro)})
(tweeetmm,{(Fay_Fay_Lovee),(PaulinhaSilvas2),(yetpit),(_maasander),(Hadi_Ghadanfar),(cassaayruubs),(k
uankanlayarat)})
(mackMandaa,{(StteSoares),(geger_uelek),(NinaDoctor060),(dujkan),(tsuyuri_ete),(bfdsale1),(viian_gata
),(caroousoeiro)})
(bfdsale1,{(PaulinhaSilvas2),(yetpit),(_maasander),(___PINKbulletss),(tsuyuri_ete),(mackMandaa),(viia
n_gata),(Love_TKO),(caroousoeiro)})
(kuankanlayarat,{(geger_uelek),(lihui86),(NinaDoctor060),(cassaayruubs),(tweeetmm),(Love_TKO)})
(viian_gata,{(Fay_Fay_Lovee),(PaulinhaSilvas2),(lihui86),(Hadi_Ghadanfar),(mackMandaa),(bfdsale1)})
(Love_TKO,{(StteSoares),(Fay_Fay_Lovee),(PaulinhaSilvas2),(lihui86),(tsuyuri_ete),(bfdsale1),(kuankan
layarat)})
(caroousoeiro,{(yetpit),(_maasander),(Hadi_Ghadanfar),(NinaDoctor060),(___PINKbulletss),(tsuyuri_ete)
,(mackMandaa),(bfdsale1)})
grunt>
```

FLATTEN operator

- Let's say you have a tuple (a , {b, c, d})
- Now you want to FLATTEN the second item which is {b, c, d}
- The output would be
  - Three tuples
  - (a, b)
  - (a, c)
  - (a, d)

What's next?

- In order to find common friends
  - What we are going to do is to create a new relation with two columns
  - The first column would be the person name
  - The second column would be one friend's name
- DATA3 = FOREACH DATA2 GENERATE $0, FLATTEN($1);

FLATTEN works as outer join for bag

Output

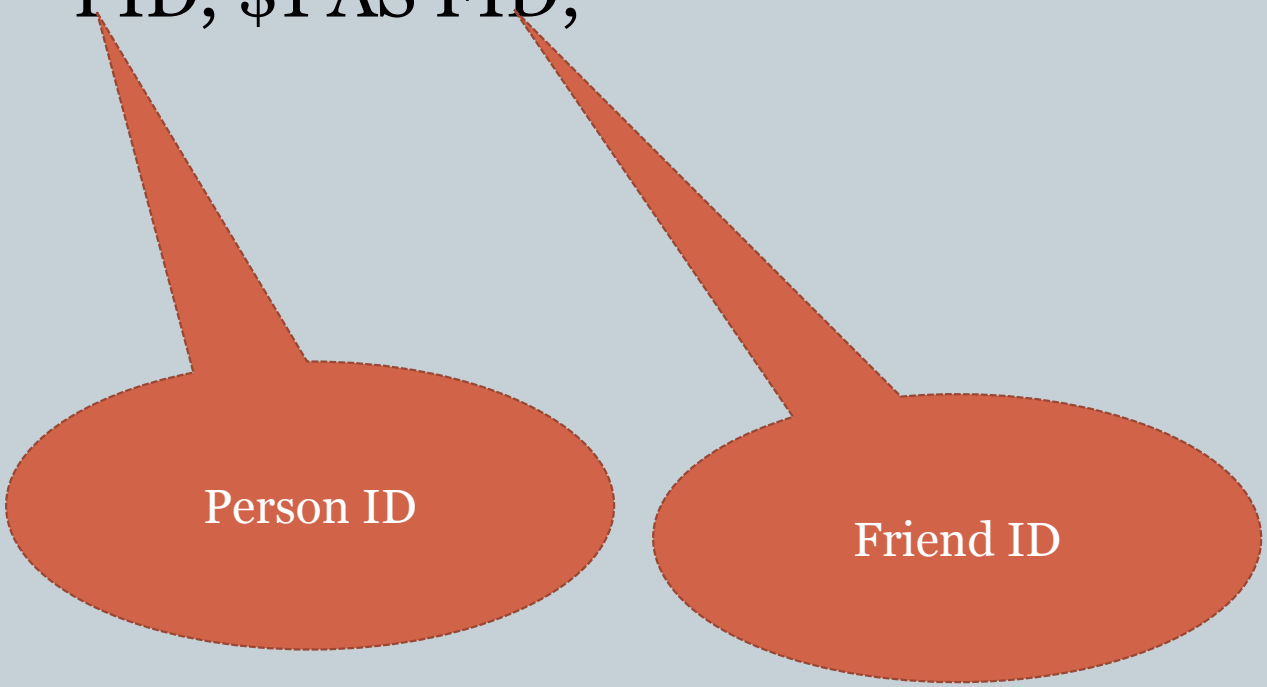
```
grunt> DUMP DATA3;
(kuankanlayarat,geger_uelek)
(kuankanlayarat,lihui86)
(kuankanlayarat,NinaDoctor060)
(kuankanlayarat,cassaayruubs)
(kuankanlayarat,tweeetmm)
(kuankanlayarat,Love_TKO)
(viian_gata,Fay_Fay_Lovee)
(viian_gata,PaulinhaSilvas2)
(viian_gata,lihui86)
(viian_gata,Hadi_Ghadanfar)
(viian_gata,mackMandaa)
(viian_gata,bfdsale1)
(Love_TKO,StteSoares)
(Love_TKO,Fay_Fay_Lovee)
(Love_TKO,PaulinhaSilvas2)
(Love_TKO,lihui86)
(Love_TKO,tsuyuri_ete)
(Love_TKO,bfdsale1)
(Love_TKO,kuankanlayarat)
(caroousoeiro,yetpit)
(caroousoeiro,_maasander)
(caroousoeiro,Hadi_Ghadanfar)
(caroousoeiro,NinaDoctor060)
(caroousoeiro,___PINKbulletss)
(caroousoeiro,tsuyuri_ete)
(caroousoeiro,mackMandaa)
(caroousoeiro,bfdsale1)
grunt>
```

Person

And one of their friends

Let's give them a name

- DATA4 = FOREACH DATA3 GENERATE $0 AS PID, $1 AS FID;



Common friends

- In order to find the common friends we can join DATA4 with itself
- But Pig Latin does not allow that
- So make a copy of DATA4
- DATA5 = FOREACH DATA4 GENERATE $0 AS PID, $1 AS FID;

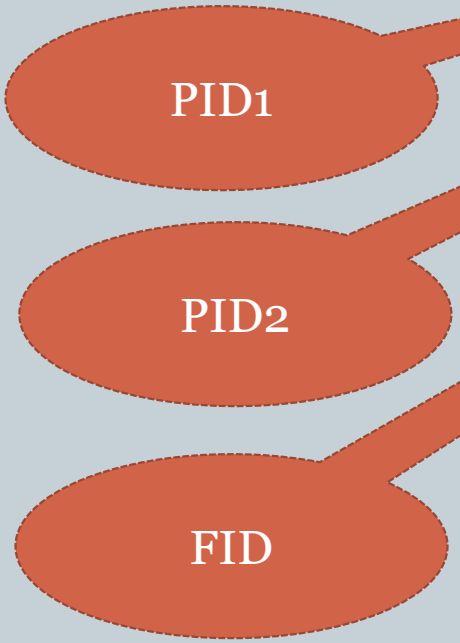
Join DATA4 and DATA5

- DATA6 = JOIN DATA4 BY FID, DATA5 BY FID;
- This will
  - Join DATA4 and DATA5
  - Create a tuple with four items
  - (DATA4::PID, DATA4:: FID, DATA5:: PID, DATA5:: FID) where DATA4:: FID == DATA5:: FID
- Now, we want to create a new relation with just one copy of FID
- DATA7 = FOREACH DATA6 GENERATE $0 AS PID1, $2 AS PID2, $1 AS FID;

What's wrong?

- DATA7 contains some item where PID1 == PID2
- We don't want them
- Now we are going to filter them out
- DATA8 = FILTER DATA7 BY PID1 != PID2;
  - This will keep only the items where PID1 != PID2;

Output



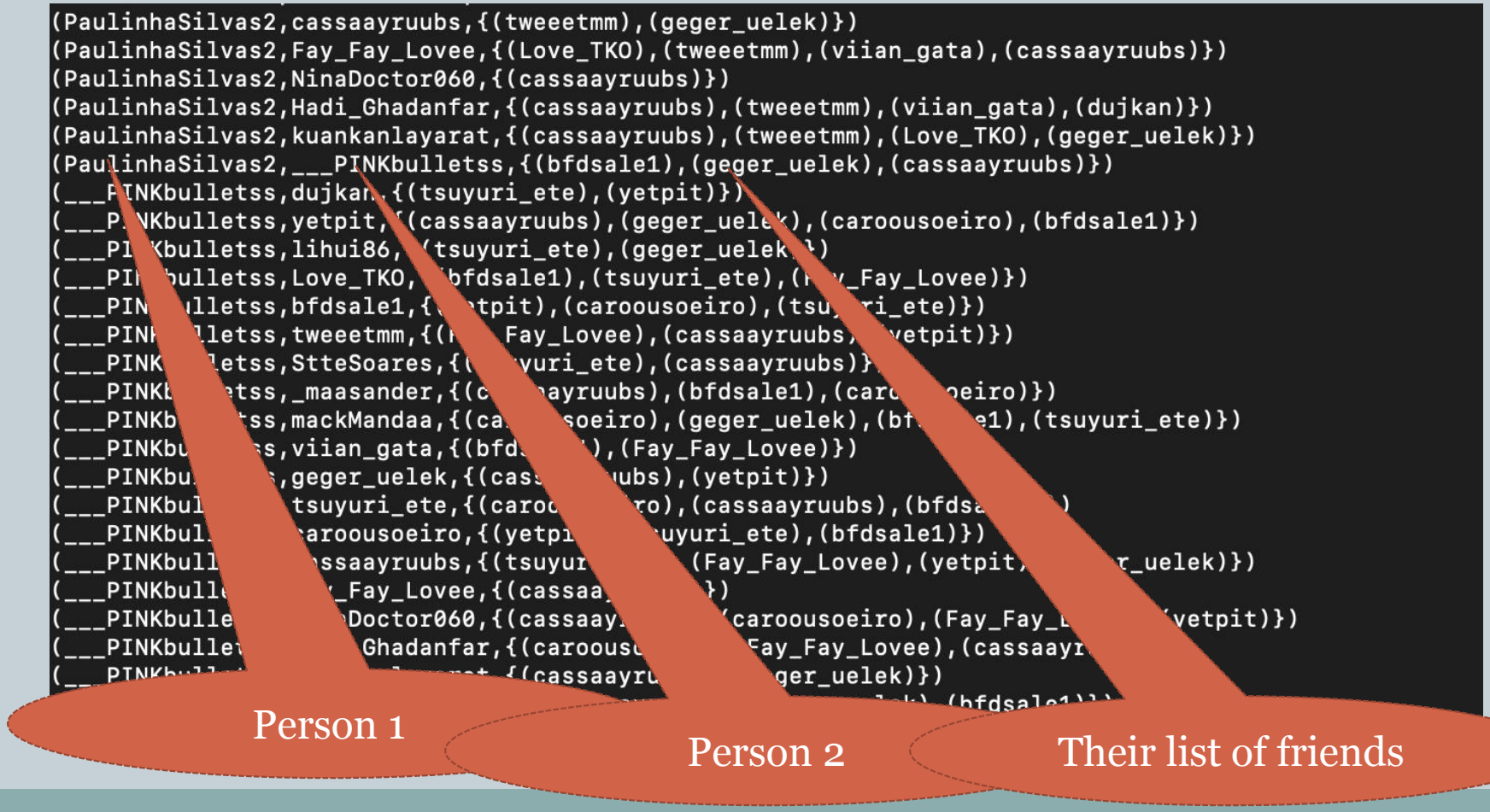

```
grunt> DUMP DATA8;
(cassaayruubs,tsuyuri_ete,___PINKbulletss)
(cassaayruubs,caroousoeiro,___PINKbulletss)
(cassaayruubs,yetpit,___PINKbulletss)
(cassaayruubs,Fay_Fay_Lovee,___PINKbulletss)
(cassaayruubs,bfdsale1,___PINKbulletss)
(cassaayruubs,geger_uelek,___PINKbulletss)
(Fay_Fay_Lovee,tsuyuri_ete,___PINKbulletss)
(Fay_Fay_Lovee,caroousoeiro,___PINKbulletss)
(Fay_Fay_Lovee,yetpit,___PINKbulletss)
(Fay_Fay_Lovee,cassaayruubs,___PINKbulletss)
(Fay_Fay_Lovee,bfdsale1,___PINKbulletss)
(Fay_Fay_Lovee,geger_uelek,___PINKbulletss)
(bfdsale1,tsuyuri_ete,___PINKbulletss)
(bfdsale1,caroousoeiro,___PINKbulletss)
(bfdsale1,yetpit,___PINKbulletss)
(bfdsale1,cassaayruubs,___PINKbulletss)
(bfdsale1,Fay_Fay_Lovee,___PINKbulletss)
(bfdsale1,geger_uelek,___PINKbulletss)
(geger_uelek,tsuyuri_ete,___PINKbulletss)
(geger_uelek,caroousoeiro,___PINKbulletss)
(geger_uelek,yetpit,___PINKbulletss)
(geger_uelek,cassaayruubs,___PINKbulletss)
(geger_uelek,Fay_Fay_Lovee,___PINKbulletss)
(geger_uelek,bfdsale1,___PINKbulletss)
grunt>
```

List of Friends?

- Now we need to create a list of friends
  - How are we going to do that?
  - Group By!
- DATA9 = GROUP DATA8 BY (PID1, PID2);
  - At this point we are done
- We have found a relation with PID1, PID2 and a bag containing a list of friends
- But in order to make it more presentable we are going to give them names
- CFRNDS = FOREACH DATA9 GENERATE $0.PID1 AS PID1, $0.PID2 AS PID2, $1.$2 AS FIDS;

Output

```
grunt> DUMP CFRNDS;
```



Input Parameters in Pig Latin

- Input parameters can be entered using the Pig command line:
  - exec -param <parameter name1>=<input value for parameter1> -param <parameter name2>=<input value for parameter2>….<name of the pig program you want to run>
  - Example: enter input data value of 0.4 for parameter support S input data value of 0.2 for confidence C for the pig program named “query 4.pig”
  - exec -param S=0.4 -param C=0.2 query4.pig

Input Parameters in Pig Latin

```
[basi5906@oracle18 ~]$ tail -n 20 query.pig
DATA = load 'fsample.txt';
DATA2 = FOREACH DATA GENERATE $0, TOKENIZE($1);
DATA3 = FOREACH DATA2 GENERATE $0, FLATTEN($1);
DATA4 = FOREACH DATA3 GENERATE $0 AS PID, $1 AS FID;
DATA5 = FOREACH DATA4 GENERATE $0 AS PID, $1 AS FID;
DATA6 = JOIN DATA4 BY FID, DATA5 BY FID;
DATA7 = FOREACH DATA6 GENERATE $0 AS PID1, $2 AS PID2, $1 AS FID;
DATA8 = FILTER DATA7 BY PID1 != PID2;
DATA9 = GROUP DATA8 BY (PID1, PID2);
CFRNDS = FOREACH DATA9 GENERATE $0.PID1 AS PID1, $0.PID2 AS PID2, $1.$2 AS FIDS;
RESULT = FILTER CFRNDS BY (PID1 == '$fid1') AND (PID2 == '$fid2');
DUMP RESULT;

[basi5906@oracle18 ~]$
```

```
grunt> exec -param fid1=mackMandaa -param fid2=yetpit query.pig
```

```
(mackMandaa,yetpit,{(bfdsale1),(dujkan),(geger_uelek),(caroousoeiro),(NinaDoctor060)})
grunt>
```

Storing Relation in a File

To store relation A in a CSV file “output.csv”:

1. STORE A INTO 'tmp_out_dir' USING PigStorage(',');

```
grunt> STORE A INTO 'tmp_out_dir' USING PigStorage(',');
```

2. fs -getmerge tmp_out_dir output.csv;
3. rm tmp_out_dir;

```
grunt> fs -getmerge tmp_out_dir output.csv;
grunt> rm tmp_out_dir;
2019-04-15 14:51:31,405 [main] INFO  org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.
checksum is deprecated. Instead, use dfs.bytes-per-checksum
2019-04-15 14:51:31,405 [main] INFO  org.apache.pig.tools.grunt.GruntParser - Waited 0ms to delete fi
le
grunt> ls
file:/home/basi5906/pig/hw4/HW4-follows_account.csv<r 1>        357277
file:/home/basi5906/pig/hw4/HW4-old_twitter_account_rank.csv<r 1>       4884
file:/home/basi5906/pig/hw4/HW4-stack_overflow_account.csv<r 1> 5590
file:/home/basi5906/pig/hw4/HW4-twitter_account.csv<r 1>        11798
file:/home/basi5906/pig/hw4/twitter_account_rank_iteration.pig<r 1>     1627
file:/home/basi5906/pig/hw4/find_ninety_nine_percentile_accounts.pig<r 1>       1141
file:/home/basi5906/pig/hw4/pig_1555357511969.log<r 1>  8220
file:/home/basi5906/pig/hw4/output.csv<r 1>     11726
grunt>
```

Pig Latin Diagnostic Operators

- Diagnostic Operators may come in handy,
  - DUMP relation // print a relation into the screen
  - DESCRIBE relation // print the schema of a relation
  - QUIT // exit Pig

Pig with Hadoop (NOT for Homework 4)

- Make sure you run with
  - pig -x mapreduce
- Everything remains the same except file operation!
- You have to copy your file into Hadoop dfs first in order to load into Pig
- How to do that?
  - hadoop dfs –copyFromLocal <srcFile> <dstFile>
  - e.g., hadoop dfs -copyFromLocal fsample.txt hdfs://localhost:8020/user/<username>/ fsample.txt
- We are good to go!
- If you have 100 machines,
  - Pig will automatically generate underlying maps and reduces
  - Hadoop will automatically parallelize them for you!

Thanks…