COMP 543, Tools and Models for Data Science

Chengyin Liu, cl93

Assignment #2

#1.1 Connected Components

```
SET SQL_SAFE_UPDATES = 0;

#create a temporary table to store unvisited nodes
drop table if exists unvNod;
create table unvNod(
    id int primary key
);

insert into unvNod(id)
    select n.id
        from nodes n;

#create a temporary table to be the current component
drop table if exists curCom;
create table curCom(
    id int primary key
);

drop procedure if exists getConCom;
delimiter //
create procedure getConCom()
begin
    declare unvNodSize int;
    declare corComSize_old int;
    declare corComSize_new int;
    select count(*) into unvNodSize from unvNod;

    while unvNodSize > 0 do
        insert into curCom(id)
            select u.id
                from unvNod u
                limit 1;

        set corComSize_old = 1;
        set corComSize_new = -1;

        while corComSize_old != corComSize_new do
            select count(*) into corComSize_old from curCom;
            insert into curCom(id)
```

```
                select distinct u.id
                    from unvNod u
                    where (u.id in(
                            select distinct e.refId
                                from edges e, curCom c
                                where e.id = c.id)
                        or u.id in(
                            select distinct e.id
                                from edges e, curCom c
                                where e.refId = c.id))
                        and u.id not in(
                            select c.id
                                from curCom c);
            delete from unvNod
                where id in(
                    select c.id
                        from curCom c);
            select count(*) into corComSize_new from curCom;
        end while;

        if corComSize_new >= 5 and corComSize_new <= 8 then
            select n.id, n.symbol
                from nodes n, curCom c
                where n.id = c.id
                order by n.symbol;
        end if;

        truncate table curCom;
        select count(*) into unvNodSize from unvNod;
    end while;

end //
delimiter ;

call getConCom();
drop table if exists unvNod;
drop table if exists curCom;
drop procedure if exists getConCom;

/*
id, symbol
----------
2252   FGF7
9982   FGFBP1
3339   HSPG2
4504   MT3
7276   TTR
----------
23250  ATP11A
23200  ATP11B
286410 ATP11C
```

```
10396   ATP8A1
5205    ATP8B1
55754   TMEM30A
161291 TMEM30B
----------
8818    DPM2
5277    PIGA
5279    PIGC
5283    PIGH
51227   PIGP
9091    PIGQ
----------
9382    COG1
22796   COG2
83548   COG3
25839   COG4
57511   COG6
84342   COG8
----------
29103   DNAJC15
131118 DNAJC19
51025   PAM16
10440   TIMM17A
10245   TIMM17B
100287932       TIMM23
10469   TIMM44
92609   TIMM50
*/
```

#1.2 PageRank

```
SET SQL_SAFE_UPDATES = 0;

#create a temporary table to store old PageRank
drop table if exists pageRank_old;
create table pageRank_old(
        id int primary key,
        refCount int,
        pr_old float
);

insert into pageRank_old(id, refCount)
select distinct n.id, count(e.refId)
from nodes n
        left join edges e on n.id = e.id
group by n.id;

#create a temporary table to store old PageRank
drop table if exists pageRank_new;
create table pageRank_new(
    id int primary key,
    pr_new float
);

insert into pageRank_new(id)
select distinct n.id
from nodes n;

drop procedure if exists getPageRank;
delimiter //
create procedure getPageRank()
begin
    declare n int;
    declare d float;
    declare sumDif float;
    declare pr_sink float;
    select count(*) into n from pageRank_old;
    set d = 0.85;
    set sumDif = n * 1.0;
    set pr_sink = 0.0;

    update pageRank_new set pr_new = 1.0 / n;

    while sumDif > 0.01 do
        replace into pageRank_old(id, refCount, pr_old)
            select po.id, po.refCount, pn.pr_new
                from pageRank_old po, pageRank_new pn
                where po.id = pn.id;

        select sum(pr_old) into pr_sink
            from pagerank_old
            where refCount = 0;

        replace into pageRank_new(id, pr_new)
            select p1.id, (1.0 - d) / n + d * (ifnull(sum(p2.pr_old / p2.refCount), 0.0)
+ pr_sink / n)
                from pageRank_old p1
                    left join edges e on p1.id = e.refId
                    left join pageRank_old p2 on e.id = p2.id
                group by p1.id;

        select sum(abs(pn.pr_new - po.pr_old)) into sumDif
            from pageRank_old po, pageRank_new pn
            where po.id = pn.id;
    end while;

    select n.id, n.symbol, pn.pr_new as PageRank
        from pageRank_new pn
            join nodes n on pn.id = n.id
        order by pn.pr_new desc
        limit 10;
```

```
end //
delimiter ;

call getPageRank();
drop table if exists pageRank_old;
drop table if exists pageRank_new;
drop procedure if exists getPageRank;

/*
id, symbol, PageRank
----------
7157   TP53   0.00674623
3065   HDAC1         0.00490489
4193   MDM2 0.0044673
2033   EP300  0.00372411
3320   HSP90AA1      0.00342112
1499   CTNNB1        0.00304391
207    AKT1   0.00274938
3066   HDAC2         0.00273742
672    BRCA10.00268912
8454   CUL1   0.0026883
*/
```