CIS 671, Winter 2002

Homework 8: Data Mining and Active Databases – Solutions – Corrected 3/19

1. Data Mining

a. Transparencies 12 and 15-17 show an analysis for an example finding association rules. Assume the problem changes as follows: 20,000 items, 20 items per basket and 10,000,000 baskets in the database. Repeat the analysis.

Finding large 2-itemsets – Naïve Algorithm

- Supermarket sells 10,000 {**20,000**} different items.
- Average market basket has 20 items.
- Database keeps 1,000,000 {**10,000,000**}baskets as data.
  - Small compared to practice.
- Basket Relation:
  - count(*) **from** Baskets = 20 * 10^6 = 2 * 10^7 {$\mathbf{20*10^7 = 2 * 10^8}$}.
- Size of join (done cleverly):

  = [10^6 baskets] * [$\binom{20}{2}$ pairs per basket]

  = 190 * 10^6 = 190,000,000 tuples {$\mathbf{190 * 10^7}$ **= 1,900,000,000 tuples**}

Example: Finding large 2-itemsets - A-Priori Algorithm

- Assume support s = 1% (.01).
  - I.e., basket count = 10^6 *.01 = 10^4 =10,000 {$\mathbf{10^7 *.01 = 10^5}$ **=100,000**}
- Maximum number of items that appear in at least 10,000 {**100,000**} baskets.
  - Exactly 10,000 baskets each. {**100,000**}
  - 20,000,000 / 10,000 = 2,000 items. {**200,000,000 / 100,000 = 2,000 items**}
- Minimum number of items
  - All customers buy exactly the same 20 items.
  - 20 items. {**no change**}

How large is OKBaskets?

Worst Case: All tuples.

Assume half the 20 items in a basket are in the 2,000, i.e., 10.

10 items/basket * $\mathbf{10^7}$ **baskets**

Find 2-itemsets from 1-itemsets.

= [10^7 baskets] * $\binom{10}{2}$ pairs per basket] = $\mathbf{45 * 10^7}$ = ¼ * 190 * 10^7

i.e., ¼ the work of the naïve algorithm.

2. Active Databases

a. EN 23.15. Assume a new tuple is inserted and/or the number of hours assigned to a project is changed.

23.15 a. Whenever an employee’s project assignments are changed, check if the total hours per week …

```
create trigger INFORM_SUPERVISOR_HOURS
before insert or update of HOURS on WORKS_ON
for each row
when ( select sum(HOURS) from WORKS_ON where ESSN = new.ESSN) < 30
       or
       select sum(HOURS) from WORKS_ON where ESSN = new.ESSN) > 40 )
       INFORM_SUPERVISOR( select   SUPERVISOR_SSN
                          from     EMPLOYEE E
                          where    E.SSN = new.ESSN);
```

23.15 b. Whenever an EMPLOYEE id deleted, delete the PROJECT tuples …

```
create trigger EMPLOYEE_DELETED
after delete on EMPLOYEE
for each row
    begin
    delete     WORKS_ON
               where     WORKS_ON.ESSN = old.SSN;
    delete     DEPENDENT
               where     DEPENDENT.ESSN = old.SSN;
    update     DEPARTMENT
               set       MGRSSN null
               where     MGRSSN = old. SSN;
    update     EMPLOYEE
               set       SUPERVISOR_SSN null
               where     SUPERVISOR_SSN = old. SSN;
    end;
```

b. EN 23.17. Only Oracle part (See PL/SQL pp. 337-339)

```
create trigger SUM_COMMISSIONS_DELETE
after delete on SALES
    update     SALES_PERSON
    set        SUM_COMMISSIONS
                     = SUM_COMMISSIONS – old.COMMISSION
    where      SALES_PERSON.SALESPERSON_ID = old.S_ID;
```

```
create trigger SUM_COMMISSIONS_INSERT
after insert on SALES
    begin
    update   SALES_PERSON
    set      SUM_COMMISSIONS
                   = SUM_COMMISSIONS + new.COMMISSION
    where    SALES_PERSON.SALESPERSON_ID = new.S_ID;

    if  (select  SUM_COMMISSIONS
        from    SALES_PERSON
        where  SALES_PERSON.SALESPERSON_ID = new.S_ID)
               + new.COMMISSION > 100000 then
        notify_manager(new.S_ID);
    end_if;
    end;


create trigger SUM_COMMISSIONS_UPDATE
after update on SALES
    begin
    update   SALES_PERSON
    set      SUM_COMMISSIONS = SUM_COMMISSIONS
             + new.COMMISSION - old.COMMISSION
    where    SALES_PERSON.SALESPERSON_ID = new.S_ID;

    if  (select  SUM_COMMISSIONS
        from    SALES_PERSON
        where  SALES_PERSON.SALESPERSON_ID = new.S_ID)
                    + new.COMMISSION - old.COMMISSION > 100000 then
             notify_manager(new.S_ID);
    end_if;
    end;
```

c. EN 23.18. Give 3 rules.

i. If a faculty member leaves,
  Remove the corresponding entries from COMMITTEE.
  Set advisor of any graduate students advised to null.
  Remove the corresponding BELONGS entries.

ii. If a person leaves,
  And the person is a faculty member,
    Remove the corresponding FACULTY entry.
  And the person is a student,
    Remove the corresponding STUDENT entry.