Database Management Systems

February 7^{th} 2011 - Draft solution

1. (6 Points) The following relations are given (primary keys are underlined):

```
SINGER(SCode, SName, City, DateofBirth)
RECORD_LABEL(RLCode, RLName, Address, City)
DISC(DCode, Title, SCode, RLCode, Type, Price)
SALE(DCode, Date, SoldCopyNumber)
```

Assume the following cardinalities:

- card(SINGER)= 10^4 tuples,
  MIN(DateofBirth) = 1-1-1969, MAX(DateofBirth) = 31-12-1998,
- card(RECORD_LABEL)= 10^4 tuples,
  number of City $\simeq 100$,
- card(DISC)= 10^6 tuples,
  MIN(Price) = 10, MAX(Price) = 29,
- card(SALE)= 10^8 tuples for year 2010.

Furthermore, assume the following reduction factor for the group by condition:

- having sum(SoldCopyNumber)$\geq$10.000 $\simeq \frac{1}{100}$.

Consider the following SQL query:

```
select RLName, City
from RECORD_LABEL RL
where City='Rome' and RLCode IN (select D.RLCode
                      from SALE S, DISC D, SINGER SI
                      where S.DCode=D.DCode and SI.SCode=D.SCode
                      D.Price ≤ 11 and SI.DateofBirth ≤ 31/12/1983
                      and S.Date ≤ 31/01/2010
                      group by S.DCode, D.RLCode
                      having sum(SoldCopyNumber)≥10.000)
```

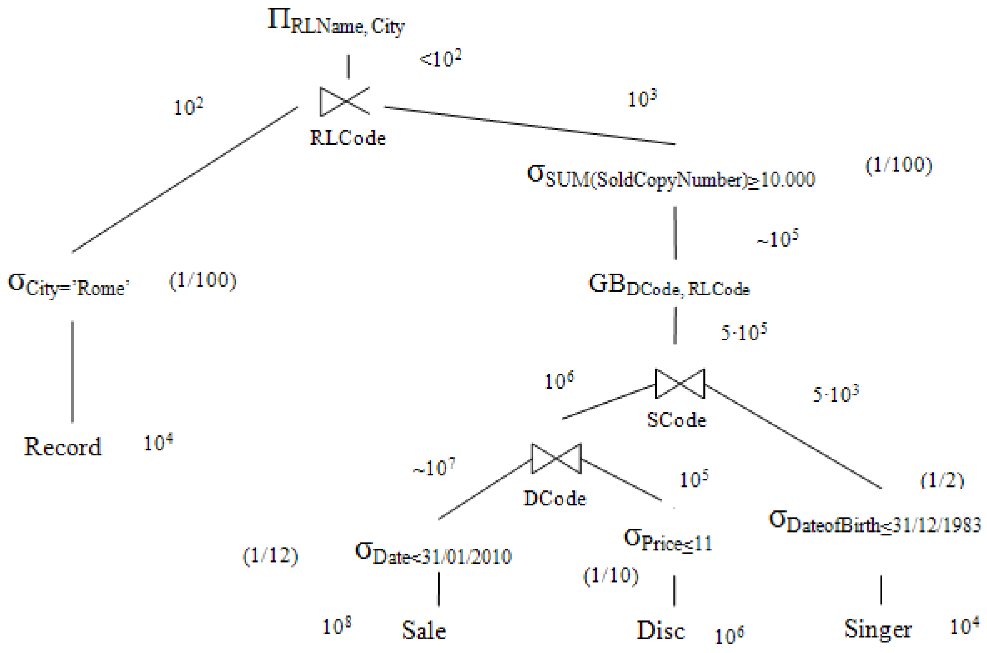
For the SQL query:

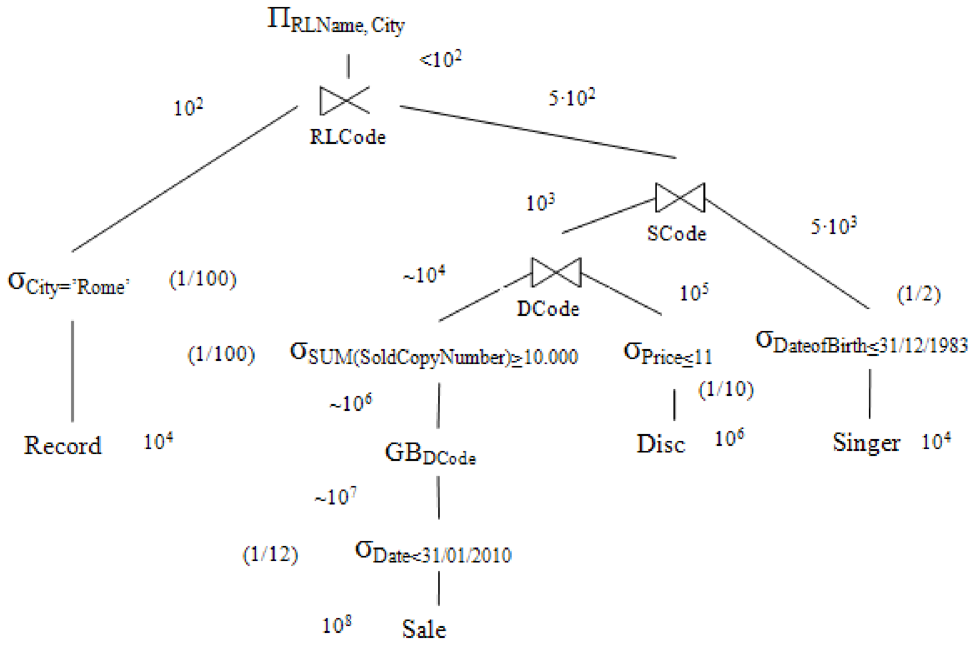
(a) Report the corresponding algebraic expression and specify the cardinality of each node (representing an intermediate result or a leaf). If necessary, assume a data distribution. Also analyze the group by anticipation.

(b) Select one or more secondary physical structures to increase query performance. Justify your choice and report the corresponding execution plan (join orders, access methods, etc.).

Solution

- Without secondary structures
  - Access path: Table access full [+filter]
  - Joins and Group by:
    (a) join Sale-Disc: Hash Join
    (b) join (a)-Singer: Hash Join
    (c) Group by Hash
    (d) join (c)-Record: Nested loop with inner=Record, outer=3
  - Indexes:
    - Secondary hash index on Record(City)





- ∗ Secondary B+Tree on Sale(Date)
- ∗ Secondary B+Tree on Disc(Price)

2. (7 Points) The following relations are given (primary keys are underlined, optional attributes are denoted with *).

   EVENT(<u>ECode</u>, EventName, EventCategory, EventCost, EventDuration)
   CALENDAR_OF_EVENTS(<u>ECode</u>, <u>Date</u>, StartHour, Place)
   CATEGORY_SUMMARY(<u>EventCategory</u>, <u>Date</u>, TotalNumberOfEvents, TotalCostOfEvents)

   The trigger application deals with scheduling the events for the 150th anniversary of Italy's unification (Italia 150) in the city of Torino. Each event belongs to a category (attribute EventCategory), e.g., exhibition, discussion, or film projection, and is characterized by an implementation cost (attribute EventCost). Each event can be repeated more times in different dates. The CALENDAR_OF_EVENTS table reports the scheduling of events in various dates and locations in Torino. Write the triggers managing the following activities.

   (1) *Update the* CATEGORY_SUMMARY *table.* The CATEGORY_SUMMARY table reports, for each event category and for each date, the total number of scheduled events and the total cost to implement them. When a new event is scheduled (a new record is inserted in CALENDAR_OF_EVENTS), the modifications on the CALENDAR_OF_EVENTS table should be propagated to the CATEGORY_SUMMARY table. Write the trigger managing this activity.

```
create trigger update_category_summary
after insert on calendar_of_events
declare
  x char(10);
  y number;
  x number;
begin
  select EventCategory, EventCost into x,y
  from Event
  where Ecode = :NEW.Ecode;

  select count(*) into z
  from Category_Summary
  where EventCategory = x and date=:NEW.date;

  if (z=0) then
    insert into Category_Summary(EventCategory, Date, TotalNumberOfEvents, TotalCostOfEvents)
    values (x, :NEW.date, 1, y);
  else
    update Category_Summary
    set TotalCostOfEvents = TotalCostOfEvents+y,
        TotalNumberOfEvents = TotalNumberOfEvents+1
    where EventCategory = x and and date= :NEW.date;
  end if;
end;
```

(2) *Integrity constraint on the maximum cost of an event.* The cost of an event belonging to the film projection category (attribute `EventCategory`) cannot exceed 1500 euro. If a cost higher than 1500 is entered in the `EVENT` table, the attribute `EventCost` should be set to 1500. Write the trigger enforcing this integrity constraint.

```
create trigger maxEventCost
before insert or update of EventCost, EventCategory on Event
for each row
when ((NEW.EventCategory = ’film projection’) AND (NEW.EventCost > 1500))
begin
     :NEW.EventCost := 1500;
end;
```

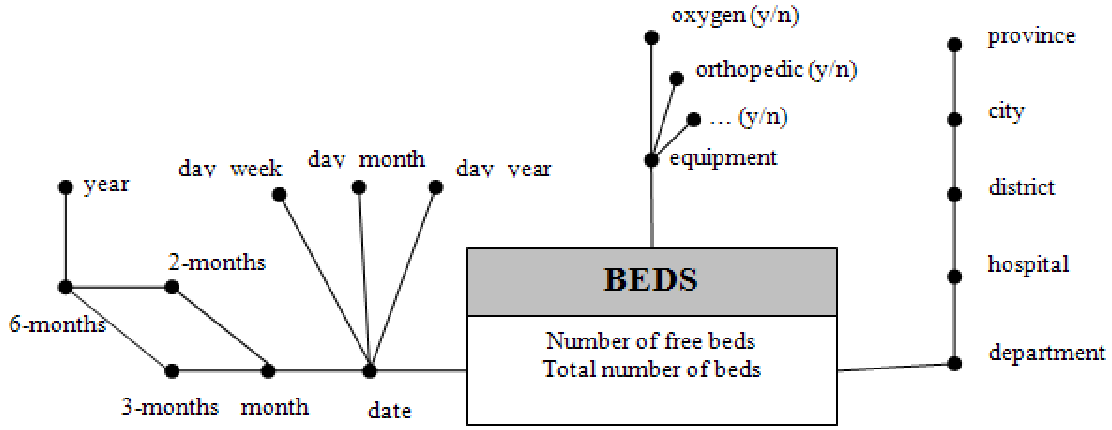
3. Data Warehouse design

The Piedmont Region wants to analyze the admissions and the usage of the hospitals to identify the structures requiring financial investments or cost cuts. To perform these analyses, a suitable data warehouse must be designed.

Each hospital is divided into several departments (e.g., department of internal medicine, emergency department, cardiology department, etc.). For each department, the number of available beds for the admissions is known for each room. Each bed can be equipped with different features (e.g., oxygen, automatic reclining, orthopedic mattress, etc.).

The Piedmont Region is interested in analyzing the percentage of free beds according to:

- the date, the month, the 2-month period, the trimester, the 6-month period and the year;
- the day of the year (from 1 to 366), the day of the week (monday-sunday), the day of the month (1-31);
- the department, the hospital, the city district, the city, and the province;
- the configuration of bed equipment (oxygen, automatic reclining, orthopedic mattress, etc.).



LOGICAL SCHEMA

```
EQUIPMENT(EquiID, bed_equipment, oxygen, orthopedic, ...)
TIME(TimeID, date, day_year, day_month, day_week, month, 2-month, 3-month, 6-month, year)
DEPARTMENT(DepID, department, hospital, district, city, province)
BEDS(EquiID, TimeID, DepID, Number_free_beds, Total_Number_beds)
```

The following are some of the frequent analysis the Region is interested in:

(a) Considering only year 2010, for each hospital and each month, select the percentage of free beds. Associate a rank to the results (rank 1 is the highest percentage).

```
SELECT  hospital, month,
        SUM(Number\_free\_beds)/SUM(Total\_Number\_beds),
        RANK() OVER (ORDER BY SUM(Number\_free\_beds)/SUM(Total\_Number\_beds) DESC)
    FROM BEDS B, DEPARTMENT D, TIME T
    WHERE B.DepID=D.DepID and B.TimeID=T.TimeID and year = 2010
    GROUP BY hospital, month
```

(b) For each configuration of bed equipment, select the total number of free beds during the different days of the week and the percentage of free beds in each day of the week respect to all the days of the week. Associate a rank to total number of free beds in descending order.

```
SELECT bed_equipment, day_of_week
       SUM(Number\_free\_beds),
       SUM (Number\_free\_beds)/SUM(SUM(Number\_free\_beds)) OVER
       (PARTITION BY bed_equipment)
       RANK () OVER (ORDER BY SUM(Number\_free\_beds) DESC)
    FROM BEDS B, EQUIPMENT E, TIME T
    WHERE B.EquiID=D.EquiID and B.TimeID=T.TimeID
    GROUP BY bed_equipment, day_of_week
```

(c) Considering only the beds with oxygen and orthopedic mattress, select the total number of free beds for each hospital and each month.

```
    SELECT hospital, month,
        SUM(Number\_free\_beds)
    FROM BEDS B, DEPARTMENT D, TIME T
    WHERE B.DepID=D.DepID and B.TimeID=T.TimeID and oxygen=’y’ and orthopedic=’y’
    GROUP BY hospital, month
```