Database Management Systems

January 27th, 2016

1. (7 Points) The following relations are given (primary keys are underlined):

   DRUG (DId, Drug_name, Category, Producer_name, Life_saving)
   DRUG_ACTIVE_INGREDIENT (DId, DAId, Max_milligrams_per_day)
   PHARMACY (PId, Pharmacy_name, Country, Address, Phone_number)
   DELIVERY(DId, PId, Date, Amount)

   Assume the following cardinalities:

   - card(DRUGS)= 10^5 tuples,
     number of categories $\simeq 10$,
     values of life saving: {Yes, No},
   - card(DRUG_ACTIVE_INGREDIENT)= 10^6 tuples,
     MIN(Max_milligrams_per_day) = 1, MAX(Max_milligrams_per_day) = 500,
   - card(PHARMACY)= 10^5 tuples,
     number of countries $\simeq 10$.
   - card(DELIVERY)= 10^{10} tuples,
     MIN(Date) = 1-1-1990, MAX(Date) = 31-12-2009,

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

   - having count(*)$\leq$100 $\simeq \frac{1}{10}$.

   Consider the following SQL query:

```
select D.DId, Drug_name
from DRUG DR, DRUG_ACTIVE_INGREDIENT DAI
where DR.DId = DAI.DId and DR.Life_saving = 'No'
and DR.Category = 'Antibiotic' and DAI.Max_milligrams_per_day≥450
and DR.DId IN (select DE.DId
    from DELIVERY DE, PHARMACY P
    where DE.PId = P.PId and P.Country <> 'Italy' and DE.Date ≥ 31-12-1999
    group by DE.DId
    having count(*)≤100)
```

   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.).

2. (8 Points) The following relations are given (primary keys are underlined, optional attributes are denoted as *).

```
CAR(Plate, CurrentState)
CUSTOMER(CustomerCode, CustomerName)
RENTAL(Plate, StartRentalTimeStamp, EndRentalTimeStamp*, CustomerCode)
RENTAL_RATE(TypeOfActivity, CostPerMinute)
RENTAL_ACTIVITY(Plate, StartTimeStamp, TypeOfActivity, EndTimeStamp*)
RENTAL_RECEIPT(ReceiptCode, CustomerCode, Plate, StartRentalTimeStamp,
      EndRentalTimeStamp, TotalCost)
```

You want to automatically manage the process of car rental in car-sharing mode. The `CAR` table stores the information about the current state for the cars owned by the company that manages the rental. The `CurrentState` attribute can take the following two values: `free`, if the car is available for rental, and `occupied`, if the car is currently rented. The `CUSTOMER` table describes the customers registered for car rental. The `RENTAL` table contains general information on the car rental such as the starting and ending rental timestamps. Note that when the rental is in progress, the ending rental timestamp takes the `NULL` value.

Two types of activities can be carried out during the rental: `use` of the car (with the engine running) and `park` of the car (with the engine off). The `RENTAL_RATE` table contains the cost for both types of activities. All rates are expressed as cost per minute, and they are charged starting from the first minute of the rental. During the rental, the customer can alternate the use and park of the car. For each car, the `RENTAL_ACTIVITY` table lists all the activities occurred during the last rental in progress. Specifically, the table records all the time periods during which the car has been used (`use`) and, possibly, parked (`park`). The last activity in progress for the car is characterized by the `NULL` value for the `EndTimeStamp` attribute. The `RENTAL_RECEIPT` table contains the receipts issued at the end of a rental period. For each rental, it reports the starting and ending timestamps and the total cost.

Write the triggers to manage the following activities.

(1) *End of a car rental.* The end of a car rental is defined by the update of the `EndRentalTimeStamp` attribute in the `RENTAL` table. This attribute is updated with the timestamp corresponding to the rental end time. When a car rental ends, the current state of the car should be updated by assigning the `free` value. Moreover, the ending timestamp for the last activity carried out during the rental should be updated (table `RENTAL_ACTIVITY`).

The total cost for the car rental should also be calculated, by taking into account all the activities carried out during the rental period. For both types of activities, the total duration and the corresponding cost should be computed. The total cost is calculated by taking into account all time periods (given by the difference between starting and ending timestamp) in which the car has been used and, possibly, parked. Use the `TO_MINUTES()` function to convert the period durations in minutes. Then, the receipt with the total cost of rental should be inserted in the `RENTAL_RECEIPT` table. Consider that the receipt code (attribute `ReceiptCode`) is a counter to be incremented each time a new receipt is inserted. Finally, all the information on the activities carried out during the last car rental should be removed from the `RENTAL_ACTIVITY` table.

(2) *Integrity constraint on the activities associated with a car.* When a new activity is inserted in the `RENTAL_ACTIVITY` table, the value of the `EndTimeStamp` attribute should be always `NULL`. Every action on the `RENTAL_ACTIVITY` table that causes the violation of this constraint must not be executed.

3. Data Warehouse Design

The European Union would like to analyze the results of publicly funded research activities taking place in the European universities, by collecting researchers' scientific publications into a data warehouse.

Researchers and professors at European universities present their research results by writing scientific papers, generally named publications. Each publication has a specific type (e.g., conference paper, journal article, book chapter, etc.), and it is characterized by a specific date of publication, one or more authors, and a publication venue (e.g., conference, journal, workshop, book, etc.), which determines the specific publication type. The publication venue (conference, journal, etc.) has an editor (e.g., Elsevier) and can have one or more editions, whose year is of interest for the European Union analysis (e.g., International Data Base Conference 2015 edition, International Data Base Conference 2016 edition, etc.).

One of the publication authors is identified as main author, and she belongs to a specific university department. Each department is part of a campus, and each university consists of one or more campuses. Furthermore, universities are divided by size (small, medium, large, depending on the number of their researchers). Each specific department is finally characterized by a scientific sector of interest. For instance, the Department of Computer and Control is characterized by the ING-INF/05 scientific sector, belongs to the Cittadella Politecnica campus, which belongs to the Politecnico di Torino university.

The European Union is interested in analyzing the number of publications according to the following dimensions:

- month, 2-month period, 3-month period, semester, year of publication;
- academic year (from September to August);
- holiday months (i.e., July and August of all years, when no teaching activities are provided);
- month of the year;
- main author's department, campus, and university;
- university size and country, department scientific sector;
- number of authors (from 1 to 10 all integer values, then a single value for 10+), publication venue (conference, journal, etc.), edition (year of the conference or year of the journal publication), editor, publication type (conference paper, journal article, book chapter, etc.).

Design

(a) (6 Points) Design the data warehouse, including both the conceptual model and the fact and dimension tables, to address the given specifications. The data warehouse must also allow efficient execution of the following queries.

(b) (8 Points) Write the following queries using extended SQL language.

(a) For each university, each publication type, and each publication year, select the monthly average number of publications, and the yearly cumulative total. Consider only months when there is at least a publication.

(b) For each year, select the percentage of publications of each department with respect to its university total. Globally rank all departments by number of total publications (the first in rank is the department with the highest number).