Politecnico di Torino
Database Management Systems

February 28^{th} 2012
Draft Solution

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

```
E-SERVICE(E-ServiceID, Name, URL, Company)
TASK(TaskID, E-ServiceID, Type, Priority)
SERVER(ServerID, #Cores, Memory, Hard_Disk, IP_address)
SCHEDULING(Date, Timestamp, TaskID, ServerID, CPU_time, Memory_usage)
```

Assume the following cardinalities:

- card(E-SERVICE)= 10^4 tuples,
  distinct values of Company $\simeq 100$,
- card(TASK)= 10^6 tuples,
  distinct values of Type $\simeq 10$,
  distinct values of Priority $\simeq 5$
- card(SERVER)= 10^3 tuples,
  MIN(#Cores) = 1, MAX(#Cores) = 16,
  MIN(Memory) = 2 GB, MAX(Memory) = 32 GB,
  MIN(Hard_Disk) = 100 GB, MAX(Hard_Disk) = 1 TB
- card(SCHEDULING)= 10^{10} tuples,
  MIN(Date) = 1/1/2011, MAX(Date) = 31/12/2011.

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

- having avg(Memory_usage)<100 MB $\simeq \frac{1}{10}$

Consider the following SQL query:

```
select Name, URL
from TASK T, E-SERVICE E
where E.E-ServiceID=T.E-ServiceID
      Priority <> 'Low' and Company = 'ItalBank'
      and TaskID in (select TaskID from SERVER S, SCHEDULING SC
                     where S.ServerID=SC.ServerID
                           and Date ≥ 1/8/2011 and Date ≤ 30/8/2011
                           and Memory = 32 GB and #Cores ≤ 4
                     group by TaskID
                     having avg(Memory_usage)<100 MB)
```

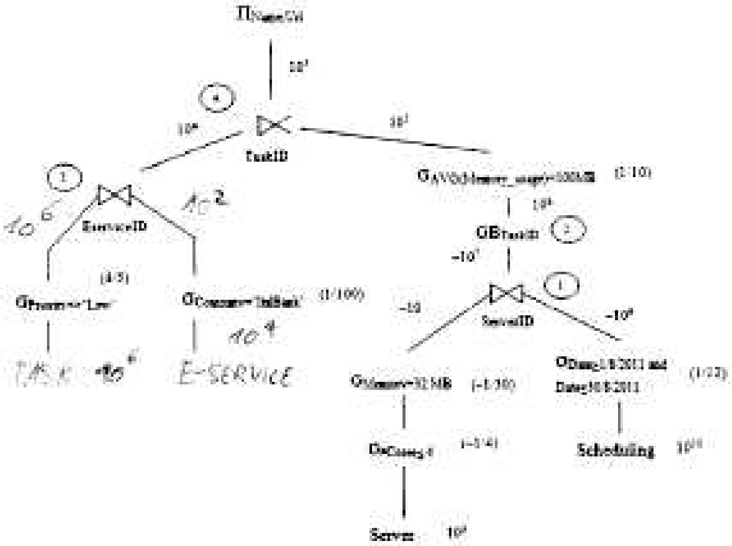
For the SQL query:

(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.).

Join and group by discussion:

(1) Nested Loop: Inner=Server, Outer=Scheduling



(2) GB Hash

(3) Nested Loop: Inner=E-Service, Outer=Task

(4) Has join

Indexes:

- Table E-SERVICE: Hash on Company
- Table SCHEDULING: B^+-Tree on Date

There is no Group By anticipation.

2. Data Warehouse design

Problem specifications

An international company distributes automotive replacement parts (e.g., lamps, suspensions, brakes, and other spare parts). The company would like to evaluate the distribution efficiency of its agencies around the world by analyzing the turnover of each agency and their distribution policies. A spare part can be used for a single car model, while the same car model can use different spare parts. In addition, a spare part can be distributed to several agencies and the same agency can deploy multiple parts.

Currently, each agency has an independent database to manage his business, but the company is interested in designing and implementing a data warehouse allowing to analyze the average daily turnover and average revenue for spare part of its agencies The analysis has to be performed according to:

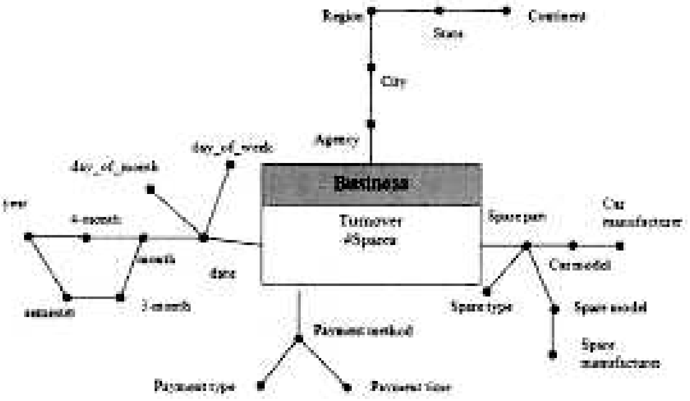
- date, month, three-month period, four-month period, semester, year of the turnover,
- holiday, day of the week, day of the month,
- agency
- payment method, which means both the payment type (e.g., credit card, bank transfer, etc.) and the payment time (within 30 days, 60 days, etc.),
- spare part, and its type,
- car model (e.g., Punto, Golf, Focus, etc.) where the spare part can be used,
- car manufacturer (e.g., FIAT, Volkswagen, Ford, etc.) where the spare part can be used,
- spare model and manufacturer,
- city, region, state and continent where the agency is located.

```
Business (AId, PId, DateId, SID, Turnover, #Spares)
Agency (AId, Agency, City, Region, State, Continent)
Payment (PId, Payment_method, Payment_type, Payment_time)
Spare (SId, Spare_part, Spare_type, Spare_model,
Spare_manufacturer, Car_model, Car_manufacturer)
Time (DateId, Date, Day_of_Week, Day_of_Month, Month, 3-Month, 4-Month,
Semester, Year)
```

The following are some of the frequent analyses the company is interested in:

(a) Considering only Italian agencies, for each agency and each car model, select the monthly turnover, the average daily turnover for each month, and the cumulative monthly turnover since the beginning of the year, separately for each payment type.

```
SELECT Agency, Car_Model, Month, Payment_type,
       SUM (Turnover)
       SUM (Turnover)/COUNT(Distinct Date),
       SUM( SUM (Turnover)) OVER (PARTITION BY Agency, Car_Model,
                                     Payment_type, Year
                                     ORDER BY Month
                                     ROWS UNBOUNDED PRECEEDING)
```



```
FROM B, A, P, S, T
WHERE B.TID=T.TID AND State='Italy'  AND  B.AID=A.AID  AND  B.PID=P.PID AND B.SID=
                                                                          S.SID
GROUP BY Agency, Car_Model, Month, Payment_type, Year;
```

(b) For each spare part, select the total yearly turnover for each Cinese agency.

(c) For each spare part of cars manufactured by FIAT and considering only the turnover of the European agencies in 2011, for each city where an agency is located, select the monthly turnover, the monthly number of distributed spare parts, and the total monthly turnover for the region where the agency is located.

```
SELECT Spare_part, City, Month,
       SUM (Turnover),
       SUM (#Spares),
       SUM( SUM (Turnover)) OVER (PARTITION BY Region)
FROM B, A, P, S, T
WHERE B.AID=A.AID AND Car_model='FIAT'  AND  B.PID = P.PID  AND  B.SID=S.SID  AND
                                                          B.TID = T.TID
AND Year=2011 AND Continent='Europe'
GROUP BY Spare_part, City, Month, Region;
```

(d) For each three-month period in 2006 and 2007, select the total turnover and the average turnover per spare part, separately for each payment type.

The data warehouse contains data related to the years 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011.
Design

(a) (6 Points) Design the data warehouse to address the described issues. In particular, the designed data warehouse must allow efficient execution of all the queries described in the specifications.

(b) (8 Points) Write frequent queries (a) and (c) of the "problem specifications" using the extended SQL language.