# Data Science And Database Technology
# Homework 3

The following relations are given (primary keys are underlined):

```
CLEANING-COMPANY(CId, Name, Address, City, Region)
OFFERED-SERVICES(CId, SId)
SERVICES(SId, ServiceName, Category)
BUILDING(BId, BuildingName, BuildingType, Address, City, Region)
CLEANING-SERVICES(CId, BId, Date, SId, Cost, NumberOfHours)
```

Assume the following cardinalities:

- card(CLEANING-COMPANY) = $10^4$ tuples,
  distinct values of Region = 20
- card(OFFERED-SERVICES)= $2 \cdot 10^5$ tuples
- card(SERVICES)= 100 tuples,
  distinct values of Category = 10
- card(BUILDING)= $5 \cdot 10^7$ tuples,
  distinct values of City = 1000
  distinct values of BuildingType = 10
- card(CLEANING-SERVICES)= $10^9$ tuples,
  MIN(Date) = 1/1/2013, MAX(Date) = 31/12/2022

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

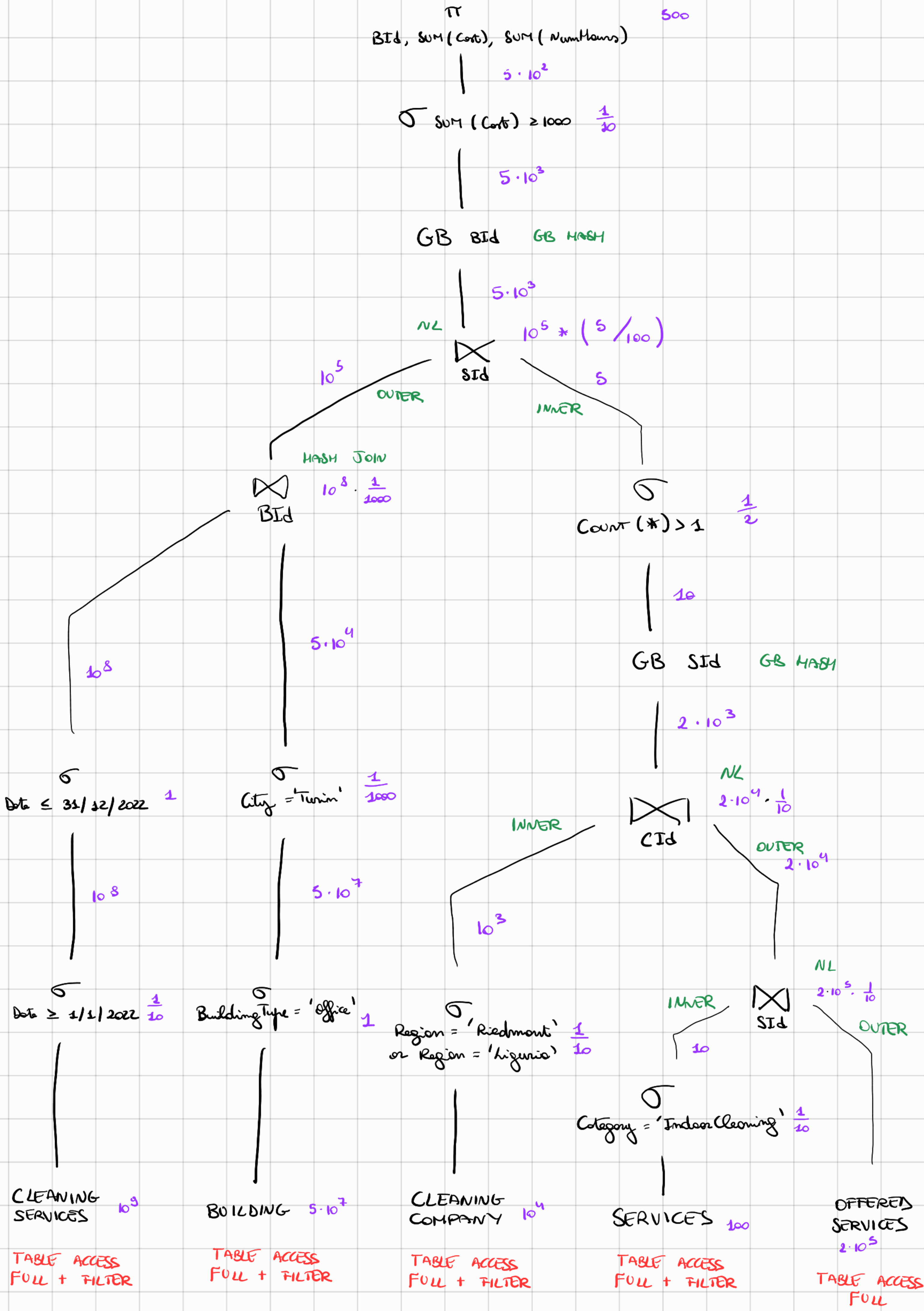
- having COUNT(*)>1 $\simeq \frac{1}{2}$.
- having SUM(Cost)$\geq$1000 $\simeq \frac{1}{10}$.

Consider the following SQL query:

```
select BId, SUM(Cost) as TotCost, SUM(NumHors) as TotHours
from CLEANING-SERVICES CS, BUILDING B
where CS.Date>=1/1/2022 and CS.Date<=31/12/2022
  and B.BuildingType <> ‘Office’
  and B.City=’Turin’
  and CS.BId=B.BId
  and CS.SId IN ( select OS.SId
                  from CLEANING-COMPANY CC, SERVICES S, OFFERED-SERVICE OS
                  where OS.SId=S.SId and OS.CId=CC.CId
                    and (Region=‘Piedmont’ or Region=‘Liguria’)
                    and Category=‘IndoorCleaning’
                  group by OS.SId
                  having COUNT(*)>1)
group by CS.BId
having SUM(Cost)>=1000
```

**Homework tasks**
For the SQL query:

1. 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.
2. 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.).


π
BId, SUM(Cost), SUM(NumHours)
500
5 · 10^2
σ SUM(Cost) ≥ 1000
1/10
5 · 10^3
GB BId
GB HASH
5·10^3
NL
10^5 * (5/100)
⋈ SId
10^5
OUTER
5
INNER
HASH JOIN
10^8 · 1/1000
⋈ BId
σ Count(*) > 1
1/2
10
GB SId
GB HASH
2 · 10^3
NL
2·10^4 · 1/10
10^8
5·10^4
σ Date ≤ 31/12/2022
1
σ City = 'Turin'
1/1000
INNER
⋈ CId
OUTER
2·10^4
10^8
5·10^7
10^3
NL
2·10^5 · 1/10
σ Date ≥ 1/1/2022
1/10
σ BuildingType = 'Office'
1
σ Region = 'Piedmont' or Region = 'Liguria'
1/10
INNER
⋈ SId
OUTER
10
σ Category = 'IndoorCleaning'
1/10
CLEANING SERVICES
10^9
BUILDING
5·10^7
CLEANING COMPANY
10^4
SERVICES
100
OFFERED SERVICES
2·10^5
TABLE ACCESS FULL + FILTER
TABLE ACCESS FULL + FILTER
TABLE ACCESS FULL + FILTER
TABLE ACCESS FULL + FILTER
TABLE ACCESS FULL

A group by can be anticipated only if there are joins that use the same attribute as the GB itself. In this case both of them cannot be pushed down

CLEANING SERVICES: Secondary B+ Tree on Date
BUILDING : Secondary HASH on City
CLEANING COMPANY : Secondary HASH on Region
SERVICES : No index
OFFERED SERVICES: No index

B+ tree is used for ranges, Hash is used for big tables with no range and no index is used for little tables or for tables that has no selection. The attribute on which the index is built is the one useful for the selection

The joins order is determined by the size of the tables: it is better to join firstly the smaller ones in order to simplify the complexity of calculations
Talking about the nested query is better to compute the join between SERVICES and OFFERED SERVICES as first and after the join with cleaning company