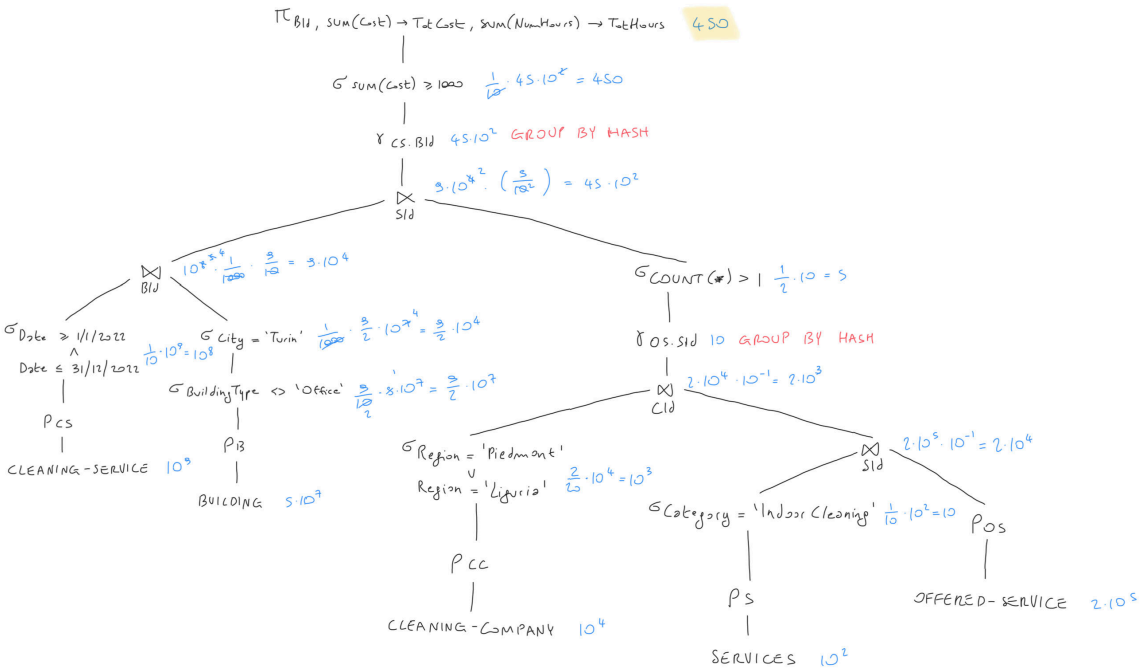
Homework 3

Where not specified we assume a uniform distribution.

SQL Query in Relational Algebra

$$\pi_{\text{BId, SUM(Cost)}\to\text{ TotCost, SUM(NumHours)}\to\text{ TotHours}}\Big(\sigma_{\text{SUM(Cost)}\geq 1000}\Big(\gamma_{\text{CS.BId}}\Big(\Big($$
$$\sigma_{\text{Date}\geq 1/1/2022\ \wedge\ \text{Date}\leq 21/12/2022}(\rho_{\text{CS}}(\text{CLEANING-SERVICES}))\underset{\text{BId}}{\bowtie}$$
$$\sigma_{\text{City}\,=\,\text{'Turin'}}\Big(\sigma_{\text{BuildingType}\neq\text{'Office'}}(\rho_{B}(\text{BUILDING}))\Big)\Big)\underset{\text{SId}}{\bowtie}\Big(\sigma_{\text{COUNT(*)}\geq 1}\Big(\gamma_{\text{OS.SId}}\Big($$
$$\sigma_{\text{Region}\,=\,\text{'Piedmont'}\,\vee\,\text{Region}\,=\,\text{Liguria}}(\rho_{\text{CC}}(\text{CLEANING-COMPANY}))\underset{\text{CId}}{\bowtie}$$
$$\Big(\sigma_{\text{Category}\,=\,\text{'IndoorCleaning'}}(\rho_{S}(\text{SERVICES}))\underset{\text{SId}}{\bowtie}\rho_{\text{OS}}(\text{OFFERED-SERVICE})\Big)\Big)\Big)\Big)\Big)\Big)\Big)$$



Join Orders

We can switch around the order of the two inner `JOIN` operation, however, given that they have the same **Reduction Factor**, this change would not affect overall perfomance. However changing the `JOIN` order from

$\text{CLEANING-COMPANY} \bowtie (\text{SERVICES} \bowtie \text{OFFERED-SERVICE})$

to

$(\text{CLEANING-COMPANY} \bowtie \text{OFFFERED-SERVICE}) \bowtie \text{SERVICES}$

enables us to anticipate the `GROUP BY`.

`GROUP BY` Anticipation

None of the `GROUP BY` operations can be anticipated

Inner QUERY

Cannot be anticipated in the current configuration because the `OS.SId` attribute is not present in the `SERVICE` subtree and `CC.CId` is needed in the other subtree and would be discarded otherwise. However, changing the `JOIN` order enables us to ancipate it in the $\text{CLEANING-COMPANY} \bowtie \text{OFFFERED-SERVICE}$ subtree and reduce the cardinality of the intermediate result.

Outer QUERY

Anticipating this `GROUP BY` would result in the loss of information on attributes invlved in the `JOIN`.

Increasing QUERY Perfomance

To increase QUERY perfomance we can use some indeces, the only attribute with a significant `Reduction Factor` is `BUILDING.City` so a *SECONDARY Hash index* would prove useful here. A *SECONDARY B+Tree index* on `CLEANING-SERVICE.Date` can be constructed for filtering thanks to the high caridinality of the table.