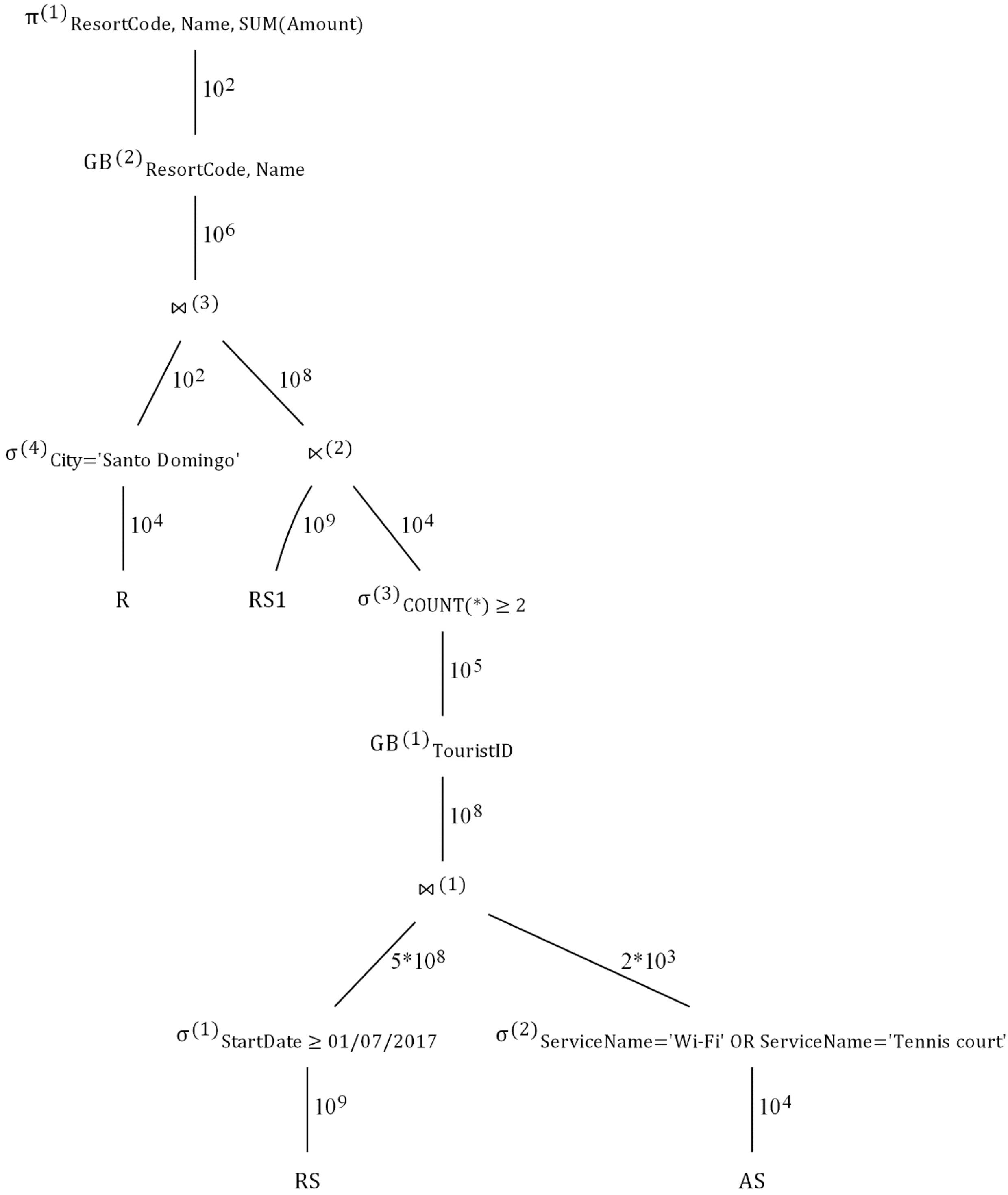
Homework #4

Matteo Corain S256654
Data Science and Database Technology – A.Y. 2018-19

1 Execution schema without indices

1.1 Execution tree



1.2 Intermediate results' cardinality

Node	Cardinality
RS	The cardinality is 10^9 (tuples in the RESERVATION_STAY table).
$\sigma^{(1)}$	Assuming a uniform distribution of data, the selection predicate has a selectivity of about $\frac{1}{2}$, since the minimum value of the field is 01/01/2017 and the maximum value is 31/12/2017. Therefore, the cardinality is about $5 * 10^8$ tuples.
AS	The cardinality is 10^4 (tuples in the AVAILABLE_SERVICES table).
$\sigma^{(2)}$	Assuming a uniform distribution of data, the selection predicate has a selectivity of about $\frac{1}{5}$, since two possible values out the ten possible ones are selected. Therefore, the cardinality is about $2 * 10^3$ tuples.
$\bowtie^{(1)}$	The cardinality after the join operation can be evaluated by multiplying the cardinality of the subtree containing the foreign key ($5 * 10^8$) by the reduction factor on the other subtree ($\frac{1}{5}$); as a result, the cardinality is about 10^8 tuples.
$GB^{(1)}$	Since the cardinality of the `TouristID` attribute (which is the grouping key) in table RS after selection $\sigma^{(1)}$ is not known, it is possible to pessimistically estimate in 10^5 tuples the cardinality after the group by operation (it is the primary key of the TOURIST table).
$\sigma^{(3)}$	It is explicitly stated that the selectivity of this predicate is $\frac{1}{10}$; thus, the cardinality of the result is 10^4 tuples.
$RS1$	The cardinality is 10^9 (tuples in the RESERVATION_STAY table).
$\ltimes^{(2)}$	The cardinality after the semi-join operation can be evaluated by multiplying the cardinality of the outer table (10^9) by the reduction factor of the result of the inner query ($\frac{1}{10}$). Consequently, the cardinality is about 10^8 tuples.
R	The cardinality is 10^4 (tuples in the RESORT table).
$\sigma^{(4)}$	Assuming a uniform distribution of data, the selection predicate has a selectivity of about $\frac{1}{100}$, since a single possible value out of 100 possible ones is selected. Therefore, the cardinality is about 10^2 tuples.
$\bowtie^{(3)}$	The cardinality after the join operation can be evaluated by multiplying the cardinality of the subtree containing the foreign key (10^8) by the reduction factor on the other subtree ($\frac{1}{100}$); as a result, the cardinality is about 10^6 tuples.
$GB^{(2)}$	Due to selection $\sigma^{(4)}$, the grouping key (`ResortCode`) has at this point 10^2 distinct values (it is the primary key of the RESORT table, on which $\sigma^{(4)}$ has been applied); therefore, it is possible to estimate in 10^2 tuples the cardinality after the group by operation.
$\pi^{(1)}$	The projection operation does not affect the number of tuples in the result set, since it is performed on a set of attributes including at least a unique field; thus, the cardinality remains about 10^2 tuples.

1.3 Access methods

Table	Access method	Motivation
RS	Table access full	No index is available, a full table scan has to be performed.
AS	Table access full	No index is available, a full table scan has to be performed.

$RS1$	Table access full	No index is available, a full table scan has to be performed.
R	Table access full	No index is available, a full table scan has to be performed.

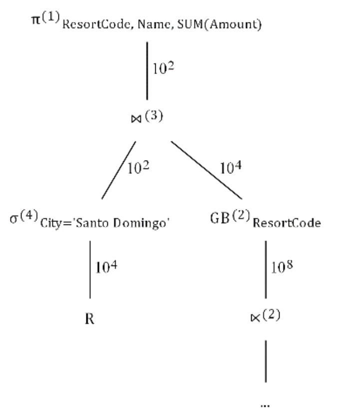
1.4 Algorithms for join operations

Join operation	Algorithm	Motivation
$\bowtie^{(1)}$	Hash join	Both of the tables have a high cardinality, hash join should be used.
$\ltimes^{(2)}$	Hash join	Both of the tables have a high cardinality, hash join should be used.
$\bowtie^{(3)}$	Nested loops	One of the two tables is small enough to make the usage of nested loops feasible.

1.5 Algorithms for group by operations

Group by operation	Algorithm	Motivation
$GB^{(1)}$	Hash group by	The input set is large, hashing has not been performed yet.
$GB^{(2)}$	Hash group by	The input set is large, hashing has not been performed yet.

1.6 Group by anticipation

Group by operation	Possibility of anticipation	Resulting tree
$GB^{(1)}$	The group by operation cannot be anticipated on the left subtree of the join operation since the join attribute is `ResortCode`, which does not appear in the grouping attributes.	-
$GB^{(2)}$	The group by operation could be anticipated on the right subtree of the join operation, given that it is executed using only `ResortCode` as the grouping attribute (the Name attribute is obtained via joining this intermediate result with the `Resort` table, in which `ResortCode` is the primary key). The modified execution tree is shown in the figure aside.	

2 Execution schema with indices

2.1 Selection of indices

Index type	Attribute	Table	Useful	Motivation
Secondary tree	`StartDate`	RS	No	The selectivity of the corresponding predicate is low ($\frac{1}{2}$).
Secondary hash	`ServiceName`	AS	No	The selectivity of the corresponding predicate is low ($\frac{1}{5}$).

Secondary hash	`City`	R	Yes	The selectivity of the corresponding predicate is high ($\frac{1}{100}$).

2.2 Access methods

Table	Access method	Motivation
RS	Table access full	No index is available, a full table scan has to be performed.
AS	Table access full	No index is available, a full table scan has to be performed.
$RS1$	Table access full	No index is available, a full table scan has to be performed.
R	Index full scan Table access by rowID	The introduced index supports the retrieval operation of data based on the selection predicate; access by rowID is necessary since the index is not covering.

2.3 Algorithms for join operations

Join operation	Algorithm	Motivation
$\bowtie^{(1)}$	Hash join	Both of the tables have a high cardinality, hash join should be used.
$\ltimes^{(2)}$	Hash join	Both of the tables have a high cardinality, hash join should be used.
$\bowtie^{(3)}$	Nested loops	One of the two tables is small enough to make the usage of nested loops feasible.

2.4 Algorithms for group by operations

Group by operation	Algorithm	Motivation
$GB^{(1)}$	Hash group by	The input set is large, hashing has not been performed yet.
$GB^{(2)}$	Hash group by	The input set is large, hashing has not been performed yet.