API

smarterator ← aggregate(smarterator r, attrId[] group_by, expressionTree aggr_on, AggrType[] aggr_ops, boolean[] distinct_flags)

Enum AggrType{
    MIN
    MAX
    SUM
    AVG
    CT
}

Each time the aggregate() smarterator’s next() or has_next() function is called, aggregate() iterates through the smarterator r, checking each tuple to determine whether that tuple belongs to the current group. If so, it adds that tuple’s data to whichever aggregate function(s) are in progress; if not, it does any final computations on its current group and returns the resulting tuple.

aggregate() should check each parameter other than smarterator r for null values. If the group_by array is null, the aggregations are performed over one group which includes the entirety of r. If the aggr_ops array is null, no aggregations are performed, and aggregate() returns a smarterator where each tuple is the name of a group. If both group_by and aggr_ops are null, aggregate() returns r unchanged.

Pipelining by Query Processor

The aggregate() function assumes that the smarterator it is passed is already sorted on the attributes specified (in order) in the group_by array. The sort() function, in turn, should be passed a smarterator which is the result of a projection, selecting only the attributes which will be either grouped or aggregated.

```
SELECT B, sum(C) FROM R GROUP BY B
```

This query requires a projection of attributes B and C from relation R, which is passed as a smarterator into a sort on attribute B, which is passed as a smarterator into the aggregation function.

```
aggregate(sort(project(r,[b,c]),[b]),[b],[c],[AggrType.SUM],[false]);
```