The Database Explosion Problem

Toon Calders

The *data explosion problem* refers to the observation that fully materializing all possible aggregated values in a data cube requires much more storage space than storing the base table alone. Even for modest relational tables the size increase can be enormous. As we will see further on, the effect increases with dimensionality and sparsity of the data cube.

In the remainder of the document, we will assume a given relational table $R(D_1, \ldots, D_k, M)$, where D_i is a dimensional attribute, with domain $dom(D_i)$ for all $i = 1 \ldots k$, and M is a numerical attribute that will be aggregated; that is, the measure.

example 1 *Consider the following base relation:*

D_1	D_2	D_3	M
a	b	c	5
a	d	e	7

The fully materialized cube for this base relation is the following:

D_1	D_2	D_3	M
a	b	c	5
a	b	$*$	5
a	$*$	c	5
a	$*$	$*$	13
$*$	b	c	5
$*$	b	$*$	5
$*$	$*$	c	5
$*$	$*$	$*$	13
a	d	e	7
a	d	$*$	7
a	$*$	e	7
$*$	d	e	7
$*$	d	$*$	7
$*$	$*$	e	7

In order to store the fully materialized cube, we need to store 14 aggregates, whereas the original data table contains 2 tuples.

The *sparsity* of the base relation $R(D_1, \ldots, D_k)$ denotes 1 minus the ratio between the actual size of R, and the size of R if all possible combinations of the values of the different attributes would actually occur. That is:

$$sparsity(R) = 1 - \frac{|R|}{|adom(D_1)| \times \ldots \times |adom(D_k)|} \quad .$$

Notice that $adom(D_i)$ denotes the *active domain* of attribute D_i, that is, the set of all values that occur for attribute D_i inb the relation R. Hence, sparsity will be minimal; i.e., equal to 0, if every combination occurs.

example 2 *In the example relation given above, the sparsity is equal to:*

$$1 - \frac{2}{1 \times 2 \times 2} \;=\; 0.5 \;\; .$$

Of the 4 possible combinations (a, b, c), (a, b, e), (a, d, c), (a, d, e), *2 occur in* R.

Dense Base Relation

When looking at the reasons for the database explosion problem, we first inspect the relation between the size of the datacube, expressed in number of tuples in the fully aggregated table, and the dimensionality of the relation. To ease the computation, we will assume that all data attributes D_i, $i = 1 \ldots k$ have the same number of values d in their active domain.

Suppose now that relation R is fully dense; i.e., has sparsity 0. Then the number of tuples in R is d^k, whereas the size of the fully materialized cube is $d + 1^k$. This can easily be seen as follows: we can represent all possible aggregates of the base relation R by adding one special dedicated symbol "*", expressing that this attribute is not a grouping attribute. Then, any tuple $t \in adom(D_1) \cup \{*\} \times \ldots \times adom(D_k) \cup \{*\}$ denotes a distinct aggregate that needs to be stored if we choose to fully materialize the datacube. There are $(d + 1)^k$ such tuples, hence the size of the cube.

In conclusion, for a fully dense data cube, the ratio between the size of the fully materialized data cube and original database equals:

$$ratio\ dense \;=\; \frac{(d+1)^k}{d^k} \;=\; \left(\frac{d+1}{d}\right)^k \;\; .$$

Although this ratio is exponential in the number of dimensions k, the base of the exponential, $\frac{d+1}{d}$ is usually fairly small. As an illustration, suppose that the active domains of the dimensions contain 10 values each, and there are 20 dimensions. In that case, the size of the data cube will be $(11/10)^20 = 6.7$ times the size of the original table. The larger the active domains, the smaller the base of the exponential becomes.

When the data is dense, only for very small domains and very high dimensionality the database explosion becomes a problem. Therefore, the dimensionality of the base relation alone cannot justify the observed database explosion problem, because we have shown that in the case of dense data the increase is relatively modest.

Sparse Base Relation

For dense databases the databaes explosion problem is not such a big problem as we saw in the previous section. In this section we will show that the problem is much different when the sparsity of the data is high. Consider, e.g., the following example.

example 3 *Consider the following relation:*

D_1	D_2	D_3	D_4	M
a	b	c	d	5
e	f	g	h	7
i	j	k	l	4
m	n	o	p	3

Our base relation contains 4 *tuples. The fully materialized cube will contain* 61 *tuples; the ratio is hence* 15.25. *This in comparison with the maximal ratio of* $(5/4)^4 \approx 2.44$ *when the base relation is dense (assuming 4 dimensions with 4 values each).*

The main reason why the effect is so much higher for sparse relations is as follows: in the last example, every tuple contributes to 16 aggregates; for example, $(a, b, c, d, 5)$ contributes to $(a, b, c, *, 5)$, $(a, b, *, d, 5)$, ..., $(*, *, *, *, 19)$. Of all these 16 aggregates, only the last one, $(*, *, *, *, 19)$, is shared with other tuples. The sparser a relation is, the less aggregates will be shared among tuples. For dense relations the situation is exactly opposite; if the size of the domains is d, d tuples will contribute to the aggregate $(*, b, c, d, x)$, d^2 to the aggregate $(*, *, c, d, y)$, and so on. This "sharing" of aggregates among the tuples in the base relation makes that the increase is far less than in the sparse case.

Worst Case

The wost case is when any two tuples only share the grand total aggregate. In that case, with k dimensions and r tuples, the size of the data cube becomes $r \times d^k - (r-1)$ (the factor $r-1$ refers to the grand total being counted r times, which is an excess of $r-1$). As such, in worst case, the ratio between data cube size and base relation becomes as big as d^k. Obviously this is a pathological case; it is highly unlikely anyone wants to browse the cube of a base relation where nothing is common between any two tuples. In the next subsection we will study the average case.

Average Case

Let us assume that our base relation contains r tuples. We will now compute the expected ratio between the size of the data cube versus the size of the base relation *under the assumption that every base relation of size r is equally likely.* This will give us some point of reference in case we do not know anything about the base relation.

The following mathematical analysis has been added for illustrative purposes only, and is not part of the examination material.

Consider an arbitrary aggregate that is part of the result of a group by ℓ attributes. For example, $(a, b, c, *, m)$ is an aggregate that is part of the group by D_1, D_2, D_3; ℓ equals 3 in this particular case. There are $d^{k-\ell}$ tuples that contribute to the count of this aggregate (not all are neessarily in our base relation, though). For our example aggregate $(a, b, c, *, m)$, all $d^{k-\ell} = d$ tuples (a, b, c, x, m) with $x \in dom(D_4)$ contribute to this aggregate.

In general, there are $C^r_{d^k - d^{k-\ell}}$ databases out of $C^r_{d^k}$ in which this aggregate is *not* part of the fully materialized cube. Hence, the probability that a random

aggregate of a grouping by ℓ attributes is not part of the materialized cube equals

$$\begin{aligned}\frac{C^r_{d^k-d^{k-\ell}}}{C^r_{d^k}} &= \frac{(d^k-d^{k-\ell})(d^k-d^{k-\ell}-1)\ldots(d^k-d^{k-\ell}-r+1)}{d^k(d^k-1)\ldots(d^k-r+1)} \\ &\leq \left(\frac{d^k-d^{k-\ell}}{d^k}\right)^r = \left(1-1/d^\ell\right)^r = \left[\left(1-1/d^\ell\right)^{d^\ell}\right]^{r/d^\ell} \\ &\leq e^{-r/d^\ell}\end{aligned}$$

In other words, the probability that a cell of the data cube is empty, decreases exponentially fast with r. The approximation is reasonably accurate under the condition that r is much smaller than d^k, which is the case since R is sparse, and d^ℓ has moderate size; say at least 50.

As an illustration, assume a relation with 500 000 tuples, the dimensionality k is 10, and the domain size of every dimension is 10. Hence, the relation is extremely sparse. The probability that an aggregate resulting from a group by with 4 attributes is empty, is less than $e^{-50} \approx 2.2510^{-22}$. There are $10^4 C^4_{10} = 2\,100\,000$ such aggregate cells, and most likely all of them wil be non-zero. The following table lists for this particular case, for every number ℓ of attributes in the group by, what is the probability that a random cell holding a tuple of such a group by is empty, and how many of such cells there are:

ℓ	$P(empty)$	# cells	Exp. number non-empty
1	≈ 0	100	100
2	≈ 0	4 500	4 500
3	≈ 0	120 000	1210^4
4	≈ 0	2 100 000	2110^5
5	$\approx 0.8\%$	25 200 000	$\approx 2510^6$
6	$\approx 60\%$	210 000 000	$\approx 8410^6$
7	$\approx 94\%$	1 200 000 000	$\approx 7210^6$
8	≈ 1	4 500 000 000	$> 500\,000$
9	≈ 1	10 000 000 000	$> 500\,000$
10	≈ 1	10 000 000 000	500 000
total			$> 17510^6$

For this example, the expected ratio between the fully materialized cube and the base relation is at least 350. This is what is meant by the database explosion problem when fully materializing all aggregations. The effect strengthens with increasing domain size d (more possible values hence less overlap between aggregates), and dimensionality k.

Conclusion

The database explosion problem refers to the observation that a fully materialized data cube can be many times bigger than the original base table. This observation makes that in most practical cases it is impossible to fully materialize the data cube. Therefore, alternative methods, such as only partially materializing a well-chosen subset of the aggregates have been developped, trading in space for storing pre-computed aggregates for higher performance at query time.