

Computer Science COMP3420 (2010) – Tutorial Three

Question 1. Review the syntax in the definitions of *data cube* and *dimension* tables, can you give a concrete example?

Answer: **define cube** cube_name [*dimension_1, dimension_2, ..., dimension_n*]:*measure_1, measure_2, ..., measure_k*

define dimension dimension_name **as** (*key_1, key_2, ..., key_l*).

For example:

define cube sales [time, item branch, location]:*dollars_sold = sum(sales_in_dollars), units_sold = count(*)*

define dimension time **as** (*time_key, day, month, quarter, year*)

Question 2. What is the lattice of data cube? what is the cuboid? what is the cuboid cells? and what is the base cell or aggregate cell?

Answer: Each table derived from a different dimensional combination of the base table is a cuboid in the data cube. All cuboids in a data cube form a lattice of data cube. Each cell in the base table is a base cell. Otherwise, the cell in a non-base table (an aggregated cuboid) is the aggregate cell.

Question 3. What is the *iceberg* data cube? what is the *Apriori* property? what is the *iceberg condition*?, and what is the *anti-monotonic* property?

Answer: In the computation of data cube, the cells that cannot pass the threshold are likely to be too trivial to warrant further analysis. Such partially materialized cubes are known as *iceberg cubes*. The minimum support threshold in the **having** statement is called the *iceberg condition*.

In the context of data cubes, *Apriori* property states as follows. If a given cell does not satisfy the minimum support, then no descendant (i.e., more specialized or detailed version) of the cell will satisfy minimum support either.

If a condition is violated for some cell *c*, then every descendant of *c* will also violate the condition. Measures that obey this property are known as *anti-monotonic* property.

Question 4. Given a data cube consisting four dimensions D_1, D_2, D_3 and D_4 , assume that the numbers of levels associated these dimensions are 10, 4, 3, and 15. How many cuboids does the data cube contain?

Answer: The number of cuboids contained in the data cube is $\pi_{i=1}^4 (L_i + 1) = 11 * 5 * 4 * 16 = 3,520$.

Question 5. Can you point out the differences between the multi-way array aggregation algorithm and the UBC algorithm in terms of the data cube computation? in which case,

which approach is preferable?

Answer: In terms of storage of data structure, Multi-way makes use of the multi-dimensional array while the BUC approach uses the Relational table.

Multi-way's computation starts from the base cuboid and progresses upward toward more generalized, ancestor cuboids. It cannot take advantage of Apriori pruning, which requires a parent node to be computed before its child nodes.

BUC's major contribution is the idea of sharing partitioning costs. However, unlike multi-way, it does not share the computation of aggregates between parent and children group-bys.

Question 6. When we apply the high OLAP minimal cubing approach for OLAP queries, which important aspects should be considered to trade-off the precomputed sub-data cubes and the on-the-fly datacube computation. Can you list them?

Answer:

- The partition of all dimensions into shell segments, how to proceed the segment partition (clustering, similarities)
- The size of each segment
- The tradeoff between the time used for computing the subcubes and the time used for entire datacube computation
- The storage requirement
- The query pattern and identifying the dimensions involved, etc.

Question 7. Suppose that a base cuboid has three dimensions, A, B, C with the following number of cells: $|A| = 1,000,000$, $|B| = 100$, $|C| = 1,000$. Suppose that each dimension is evenly partitioned into 10 portions for chunking.

- Assuming each dimension has only one level, draw the complete lattice of the cube.
- If each cube cell stores one measure with 4 bytes, what is the total sizes of the computed cube if the cube is dense?
- State the order for computing the chunks in the cube that requires the least amount of space, and compute the total amount of main memory space required for computing the 2-D planes.

Answer:

- Assuming each dimension has only one level, draw the complete lattice of the cube.
- If each cube cell stores one measure with 4 bytes, what is the total sizes of the computed cube if the cube is dense?

The total sizes of the datacube is $(|A| * |B| * |C| + |A| * |B| + |B| * |C| + |A| * |C| + |A| + |B| + |C| + |All|) * 4 = (10^{11} + 10^8 + 10^5 + 10^9 + 10^6 + 10^2 + 10^3 + 1) * 4$.

- State the order for computing the chunks in the cube that requires the least amount of space, and compute the total amount of main memory space required for computing the 2-D planes.

The minimum cost of computation is (in the order of BC, AB, and AC) $100 \cdot 1000$ (for the whole BC plane) + $100 \cdot 100,000$ (for one row of the AB plane) + $100,000 \cdot 100$ (for one chunk of the AC plane) = $10^5 + 10^7 + 10^7 = 20,100,000$ memory units.

Question 8. Suppose that we would like to compute an iceberg cube for the dimensions, A, B, C , and D , where we wish to materialize all cells that satisfy a minimum support count of at least θ , and where $|A| < |B| < |C| < |D|$. Show the BUC processing tree (which shows the order in which the BUC algorithm explores the lattice of a data cube, starting from *all*) for the construction of the above iceberg cube.

Answer: *Hint:* The performance of BUC is sensitive to the order of the dimensions and to skew in the data. Ideally, the most discriminating dimensions should be processed first. Dimensions should be processed in order of decreasing cardinality. The higher the cardinality is, the smaller the partition size is, and thus, the more partitions there will be, thereby providing BUC with greater opportunity for pruning. Similarly, the more uniform a dimension is, the better it is for pruning.