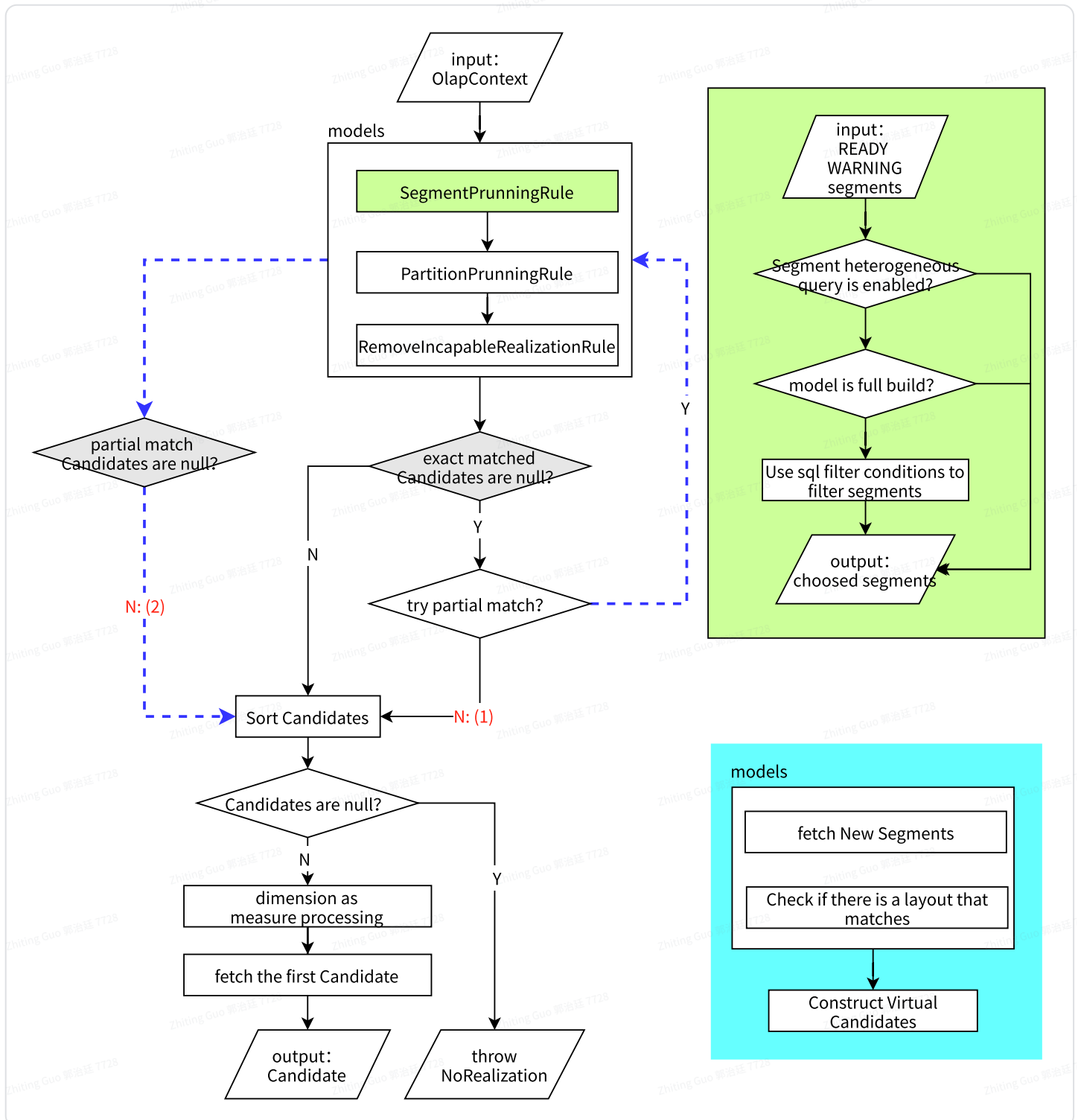


Segment heterogeneous query behavior

Detailed design:



Query matching process transformation

The flowchart on the left basically reflects the query process of the entire query. The green part on the right is the expansion of the green part in the flowchart.

The blue part is the plan newly added logic, which needs to be added in two places marked in red. This place needs to configure a matching policy `kylin.query.index-match-rule`, which is empty or misconfigured, is consistent with the current product behavior; only when its value is correctly configured as `use-vacancy-indexes`, this policy takes effect.

- When the exact matching matched Candidates are empty, if partial matching is not enabled, then try to construct Virtual Candidates.
 - If there is an unbuilt layout in the segments of the New state, then use the model answer and return empty data;
 - If there is no layout in the segments of the New state to answer, then press down;
- When the exact matched Candidates are empty,
 - If partial matching is enabled, and the Candidates obtained by partial matching are not empty, the result of partial matching will be returned;
 - If partial matching is enabled, and the partially matched Candidates are empty, but there is an unbuilt layout in the Virtual Candidates constructed by exact matching, then the exact matching model is returned, and the query result is empty data;
 - All other states are under pressure.

Data integrity priority transformation

Refer to the logic in the product PRD, extract two different sets of candidates sorting logic to ensure that the original function is not affected by this change, and also facilitate subsequent expansion.

Query index matching logic: including two parts of the same model index matching priority and matching priority between different models

- Index selection priority under the same model
 1. Priority for segment integrity
 2. Long segment time range (included build time, calculated in milliseconds)
 3. Prioritize agg (when the cost of aggregation and detail is the same, turn on this switch, [this switch is on by default])
 4. Try not to use derivative queries (if the masked column is turned on, and the snapshot is prioritized, then the snapshot + index answer is prioritized; if the masked column is not turned on, or the masked column is turned on but the quick search query is not prioritized, then the index is prioritized [this switch is turned off by default])

5. Cost (the number of rows of data is small, this is actually an algorithm, the number of rows * penalty factor, the penalty factor is to use the details to answer the select star will become larger, the penalty factor will also be due to some special measures such as topN, etc.)
 6. Filter conditions (shardBy first, other filter conditions)
 7. The fewer number of dimensions of the index is preferred
 8. The smaller number of indexed metrics is preferred
 9. The first row with fewer non-filter conditions
 10. The latest segment is in front.
- Index selection priority under different models
 1. Use detail answer select *, the fewer missing columns, the more priority will be selected (only effective for those with detail answer select * turned on)
 2. SQL hint
 3. Index costs, point 5 above
 4. The uuid natural order of the model, that is, the ascending order of the string



Note: sqlhint will be invalid for non-exact matching sqlhint in the case of exact matching models, because these non-exact matching models will be skipped when querying matching models.

Read data code modification

This place only needs to ensure that Virtual Candidates does not report errors and returns an empty dataset collection.

CH function

No changes were found for the time being (if the test finds a problem and then fixes it, I am not very familiar with it)