Optimization Possibilities for the postgresql database

Here are some possible ways to optimize the queries for the data warehouse.

To optimize our PostgreSQL queries for the data warehouse, consider the following strategies:

Indexing: we can ensure to have appropriate indexes on columns used in JOINs, WHERE clauses, and ORDER BY clauses. Indexes can significantly speed up query performance.

For our queries, we can consider creating indexes on fact_survey(theme_id, question_id, country_id, organization_id, date_id), dim_Theme(theme_id), dim_Question(question_id, question_type_id), dim_Country(country_id), dim_Organization(organization_id), and dim_Date(date_id).

Avoiding SELECT : Instead of selecting all columns with SELECT *, we can specify only the columns we need. This reduces the amount of data transferred and processed.

Partitioning: If your fact_survey table is very large, consider partitioning it by a logical key such as date, country, or organization. This can reduce the amount of data scanned during queries.

Materialized Views: For frequently accessed and complex queries, consider using materialized views. They store the result of a query and can be refreshed periodically.

EXPLAIN and EXPLAIN ANALYZE: Use EXPLAIN and EXPLAIN ANALYZE to understand the query execution plan and identify bottlenecks.

```
EXPLAIN ANALYZE SELECT
   dt.theme_name,
   COUNT(fs.answer_id) AS number_of_answers
FROM
   fact_survey fs
JOIN
   dim_Theme dt ON fs.theme_id = dt.theme_id
JOIN
   dim_Question dq ON fs.question_id = dq.question_id
GROUP BY
   dt.theme_name
ORDER BY
   number_of_answers DESC;
```