**9 Ways to Optimize SQL Queries**

Supercharge your SQL queries

SQL query optimization is important, just like any other component of your database management system. If you don’t optimize the queries that access your data, the database’s performance will suffer. In many cases, this slowdown can prevent users from accessing the necessary information quickly. This article will talk about various SQL query optimization techniques that can be used to improve query performance and reduce the cost of the solution.



Photo by [Hiroshi Kimura](https://unsplash.com/@muukii) on [Unsplash](https://unsplash.com/photos/rtX4wxMEI2M" \t "_blank)

**Tip 1. Instead of \* use column names in a select statement**

If you want to choose only a certain number of columns then column names should be used instead of \* in select statement. Although this is simpler to write, the database will need more time to process the query. By limiting the number of columns you choose, you can decrease the size of the result table, lower network traffic, and improve query performance as a whole.

Example :

**Original Query :**Select \* from sales;**Improved Query :**Select product\_id from sales;

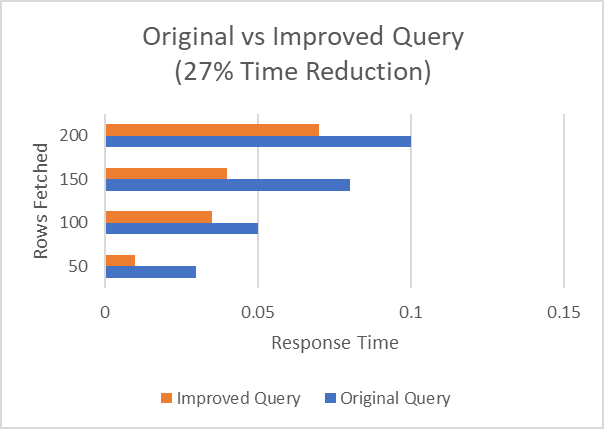


Image by Author

**Tip 2. Instead of HAVING use WHERE to define the filters**

A SQL optimized query will only retrieve the necessary records from the database. HAVING statements are computed after WHERE statements in accordance with the SQL Order of Operations. A WHERE statement is more effective if the goal is to filter a query based on conditions.

Example :

**Original query:**SELECT customer\_id,count(customer\_id)FROM salesGROUP BY customer\_idHAVING customer\_id != '16' AND customer\_id != '2';**Improved query:**SELECT customer\_id,count(customer\_id)FROM salesWHERE customer\_id != '16'AND customer\_id !='2'GROUP BY customer\_id;

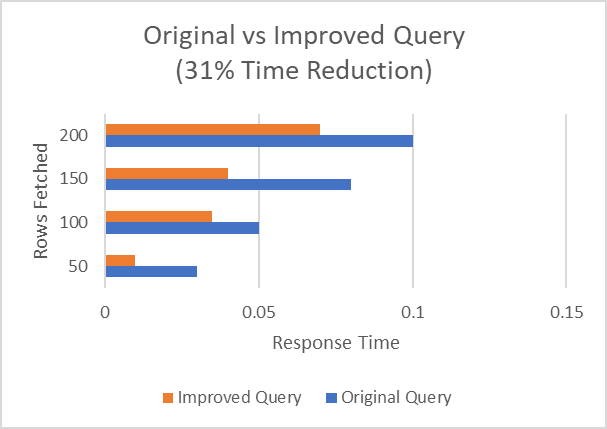


Image by Author

**Tip 3. Avoid Unnecessary distinct conditions**

Using Distinct statement is a handy way to remove duplicates. It works by creating groups in a query. However, a lot of computing power is needed to achieve this goal. Furthermore, data may be inaccurately classified to a certain extent. The solution is to choose more fields to produce distinct results instead of using SELECT DISTINCT.

Example :

**Original Query:**SELECT DISTINCT FirstName, LastName, StateFROM Teachers;**Improved Query**SELECT FirstName, LastName, Address, State,CourseName,TimingsFROM Teachers;

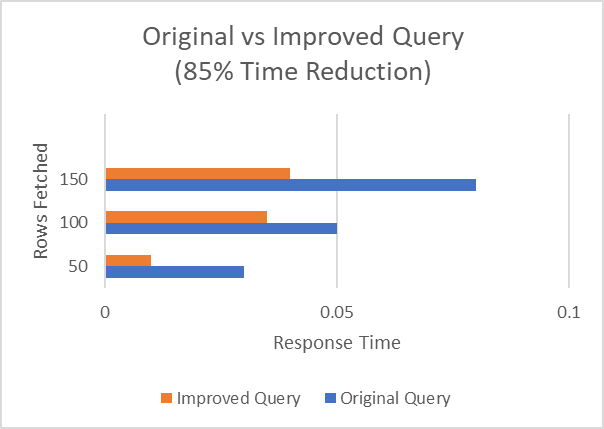


Image by Author

**Tip 4. Use JOIN instead of a SUBQUERY**

The advantage of using join is that it executes faster as compared to subquery. Unlike sub-queries, which will execute all the queries and load all the data to perform the processing, JOINs allow RDBMS to construct an execution plan that is better for your query and can forecast what data should be loaded to be processed and save time.

Example :

**Original query:**SELECT \*FROM products pWHERE p.product\_id =(SELECT s.product\_idFROM sales sWHERE s.customer\_id = 2468AND s.quantity\_sold = 12 );**Improved query:**SELECT p.\*FROM products p, sales sWHERE p.product\_id = s.product\_idAND s.customer\_id = 2468AND s.quantity\_sold = 12;

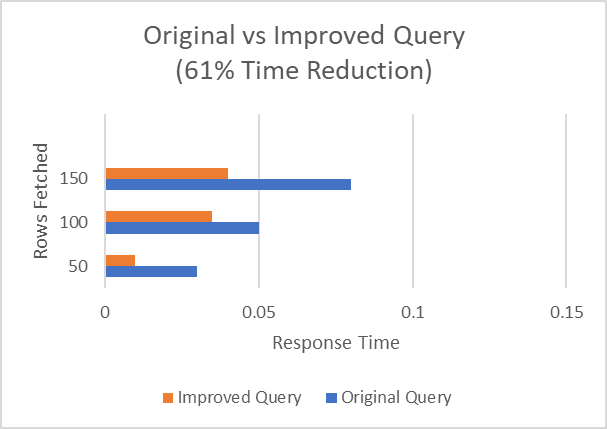


Image by Author

**Tip 5. Use In predicate when querying an index column**

For indexed retrieval, the IN-list predicate can be used, and the optimizer can sort the IN-list to match the index’s sort order for more effective retrieval. Keep in mind that the IN-list can only contain constants — that is, things that remain the same during a single execution of the query block — like outer references.

Example :

**Original query:**SELECT \*FROM salesWHERE product\_id = 4OR product\_id = 7;**Improved query:**SELECT \*FROM salesWHERE product\_id IN (4, 7);

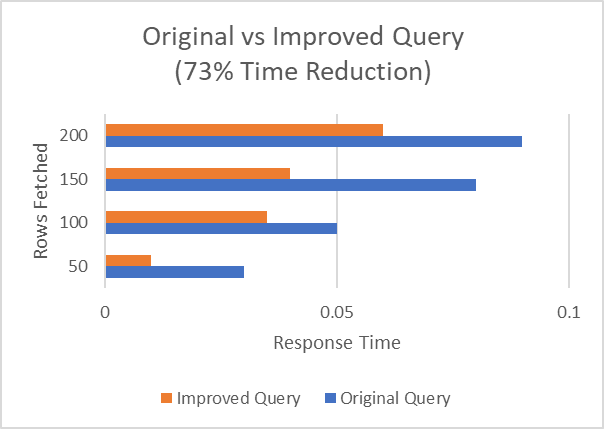


Image by Author

**Tip 6. When using table joins that involve tables with one-to-many relationships, use EXISTS rather than DISTINCT.**

DISTINCT works by creating groups in a query which takes a lot of computation power. You can use subquery with EXISTS keyword so that you can avoid returning an entire table.

Example :

**Original query:**SELECT DISTINCT c.country\_id, c.country\_nameFROM countries c, customers eWHERE e.country\_id = c.country\_id;**Improved query:**SELECT c.country\_id, c.country\_nameFROM countries cWHERE EXISTS (SELECT \* FROM customers eWHERE e.country\_id = c.country\_id);

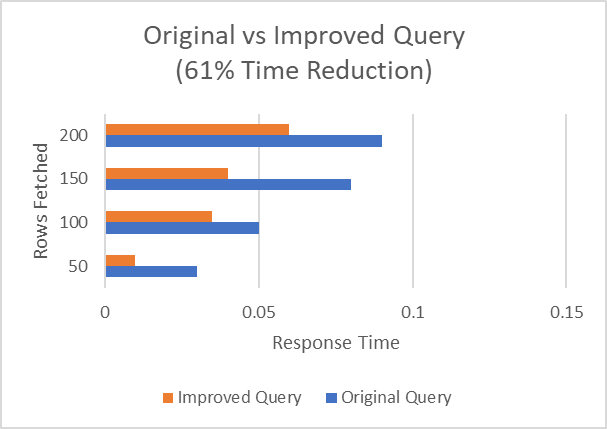


Image by Author

**Tip 7. Use Union ALL instead of Union wherever possible**

Union ALL executes faster than Union because, in UNION, duplicates are removed whether they exist or not. Union ALL displays the data with duplicates.

Example :

**Original query:**SELECT customer\_idFROM salesUNIONSELECT customer\_idFROM customers;**Improved query:**SELECT customer\_idFROM salesUNION ALLSELECT customer\_idFROM customers;

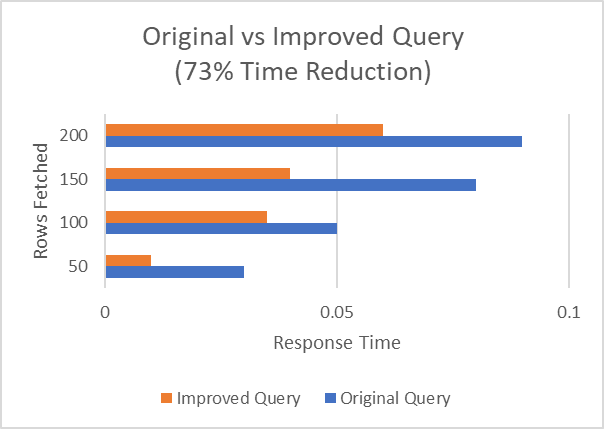


Image by Author

**Tip 8. Avoid using OR in join queries**

The query slows down by a factor of 2 if OR is used while joining queries.

Example :

**Original query:**SELECT \*FROM costs cINNER JOIN products p ON c.unit\_price =p.product\_min\_price OR c.unit\_price = p.product\_list\_price;**Improved query:**SELECT \*FROM costs cINNER JOIN products p ON c.unit\_price =p.product\_min\_priceUNION ALLSELECT \*FROM costs cINNER JOIN products p ON c.unit\_price =p.product\_list\_price;

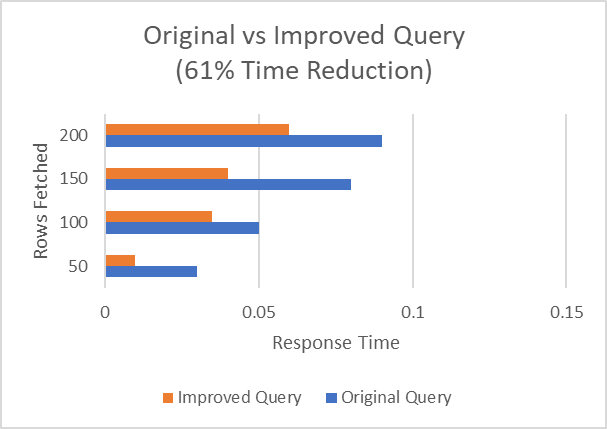


Image by Author

**Tip 9. Avoid using aggregate functions on the right side of the operator**

Avoiding using aggregate functions on the right side of the operator will optimize SQL query drastically.

Example :

**Original query:**SELECT \*FROM salesWHERE EXTRACT (YEAR FROM TO\_DATE (time\_id, ‘DD-MON-YYYY’)) = 2021 AND EXTRACT (MONTH FROMTO\_DATE (time\_id, ‘DD-MON-YYYY’)) = 2002;**Improved query:**SELECT \* FROM salesWHERE TRUNC (time\_id) BETWEENTRUNC(TO\_DATE(‘12/01/2001’, ’mm/dd/yyyy’)) ANDTRUNC (TO\_DATE (‘12/30/2001’,’mm/dd/yyyy’));

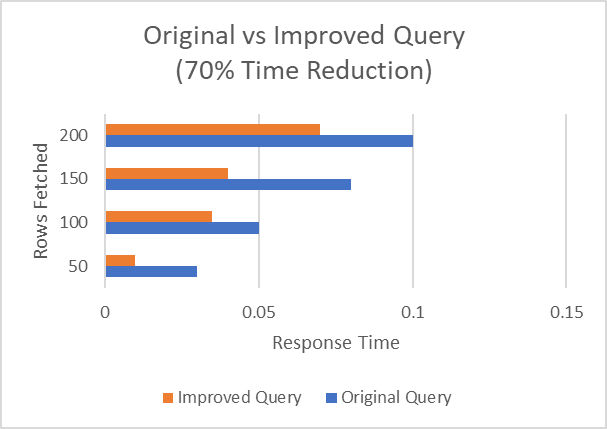


Image by Author

**Conclusion**

Query optimization is a regular operation carried out by database administrators, data analyst and application designers to fine tune the overall performance of the database system.So following these simple tips will help to optimize the sql query.