**Database Query Optimization (Ashwini):**

The code structure consists of Hashmap to map table\_name to column\_name\_list

|  |  |
| --- | --- |
| HashMap<String, Column> hmap | |
| table\_name | HashMap<String,Integer> column\_name\_list |

This is for Optimization no. 1, which is to merge multiple select statements into a single select

Another Hashmap to map column\_name to its count

|  |  |
| --- | --- |
| HashMap<String,Integer> column\_name\_list | |
| Column\_name | count |

This is for Optimization no. 2, which is to eliminate the unused columns

All other operations are parsing the file and performing string match

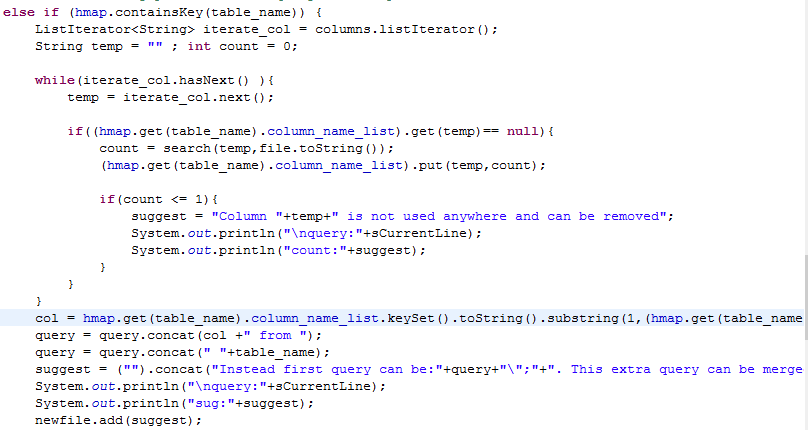
1. Combining multiple select statements into a single Select statement.

Description:

1. Accessing the Database multiple times for the same operation should be avoided
2. Query Processing for each access consumes significant time

Implementation Detail:

1. Create a hashmap for table\_name associated with hashmap of columns associated with their reference count.
2. Combined multiple selects from same table into single select query



Output:



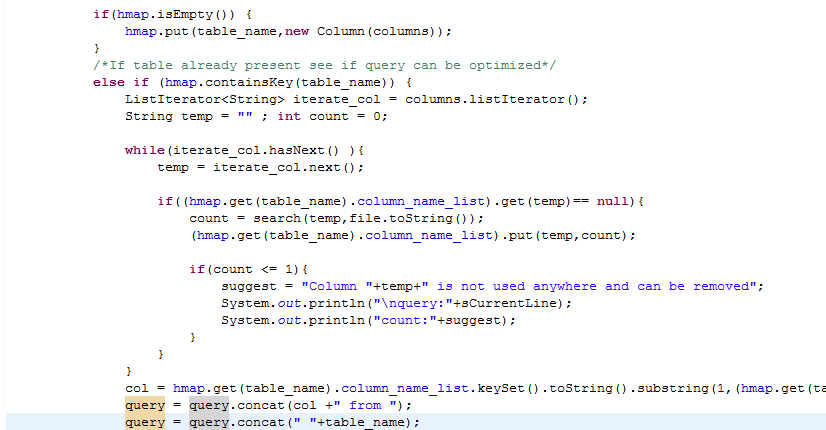
1. Picking only needed columns:

Description:

1. The code might fetch data that was never unused, detecting that and eliminating it.

Implementation Detail:

1. Keep a counter for the mention of column name in the code.
2. If the column data was never used, give the suggestion to eliminate it from the query



Output:



1. Looping Queries: Looping Select queries can be optimized by forming multiple JOIN instead

Description:

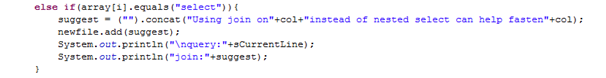
Select queries can be optimized by forming multiple JOIN instead of nested select.

Eg: select name from yourtable where age in (select age from kids)

This query will select all the rows from kids whether or not the age matches the one in yourtable. Instead if we perform join on age column , only the required rows will be fetched.

Implementation Detail:

Suggest that join can be performed instead of nested select loop.

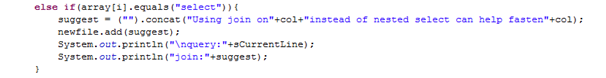


Output:



1. Optimizing queries to retrieve data in indexed manner

Found that it should be on database side. Index needs to be created on the database table. Thus our optimizer suggests the developer to do so.



Output:



Challenges and Time Division:

1. Needed to Revise SQL concepts and learn query optimization – 5 hrs
2. Learned Java Script basics – 5 hrs
3. Code implementation – 10 hrs
4. Debugging and refining – 14 hrs
5. Documentation – 2 hrs

I was a C developer for 4 yrs, hence developing in Java took some time. Learned and applied hashmap, regex operations for optimizations. Most of the code can be used for generic database optimization suggestions. The suggestions are appended into the output file wherever necessary in the form of comments so as to directly point the code that can be optimized.