Question 2dii: Explain your answer to the previous part (Question 2di) based on your knowledge from lectures and details from the query plans. Your explanation should also include why you didn't choose certain options. Please answer in maximum 5 sentences.

The main difference between virtual views and materialized views is when they are executed. Virtual views, in a way, are on standby. The view doesn't load until we run a query with it, while the materialized view runs and stays updated. Due to the fact that the materialized view is often updated, regenerating the view takes less time so the overall execution time and cost is much less for that. Calling a virtual view is almost the same as not using a view at all in runtime and cost because both are running at the time the query is called, nothing is preloaded like the materialized view.

0.1 Question 3c:

Given your findings from inspecting the query plans of queries from Questions 3a and 3b, fill in the blank and **justify your answer**. Explain your answer based on your knowledge from lectures, and details from the query plans (your explanation should include why you didn't choose other options). Your response should be no longer than 3 sentences.

Note: Your answer should be formatted as follows: `A because ...`

Adding a filter _____ the cost. A. increased B. decreased C. did not change

B because as we can see above, the cost with the filter (209.86) is significantly lower than that of the query without the filter (529.58). Since the view is sorted (by the way that we constructed it), the program knows that once it reaches a year after 2010, it doesn't need to search through the rest of the scan.

0.2 Question 3d:

Given your findings from inspecting the query plans of queries from Questions 3a and 3b, fill in the blank and **justify your answer**. Explain your answer based on your knowledge from lectures, and details from the query plans (your explanation should include why you didn't choose other options). Your response should be no longer than 3 sentences.

Note: Your answer should be formatted as follows: `A because ...`

Adding a filter _____ the execution time. A. increased B. decreased C. did not change

B because as we can see above, the execution time with the filter is significantly lower (0.796) than that of the query without the filter (4.462). Since the view is sorted (by the way that we constructed it), the program knows that once it reaches a year after 2010, it doesn't need to search through the rest of the scan.

0.3 Question 4d

Given your findings above, why did the query optimizer ultimately choose the specific join approach you found in each of the above three scenarios in Questions 4a, 4b, and 4c? Feel free to discuss the pros and cons of each join approach as well.

If you feel stuck, here are some things to consider: Does a non-equijoin constrain us to certain join approaches? What's an added benefit in regards to the output of merge join?

Note: Your answer should be formatted as follows: `Q4a: A because ... Q4b: A because ...` You should write no more than 5 sentences.

Q4a: C because hash join is used here because we have very large tables, and hash join is best on large tables. Nested Loop would take too much time and sort merge is requires another condition.
Q4b: B because merge join is best here because, again we have large tables, but in addition to that, the hash partitions aren't uniformly sized.
Q4c: A because nested loop join is useful here because we have small tables, so there is still enough space for nested loop join to run properly and efficiently.

0.3.1 Question 5di Justification

Explain your answer to `Question 5d` above based on your knowledge from lectures, and details from inspecting the query plans (your explanation should include why you didn't choose certain options). Your answer should be no longer than 3 sentences.

We can see based on the numerical results of cost and execution time that the g_batting index affected both while salary index did not. The reason for these differences are due to the query plans. Without the indexes, the query plan invovled sequential scans, which can be lengthy since it must run through all rows. The reason the g_batting index is faster than the salary index is because the g_batting index can be filtered, which speeds up the process, while the salary index does not help with finding the average as we still need to see all salaries.

0.3.2 Question 6ei Justification

Explain your answer to `Question 6e` above based on your knowledge from lectures, and details from inspecting the query plans (your explanation should include why you didn't choose certain options). Your answer should be no longer than 3 sentences.

We can see that adding an index on a column used in an AND predicate and adding a multicolumn index on columns in an OR predicate will reduce both query cost and execution time. This is because AND adds a level of filtering that requires both conditions to be true, thus limiting how much information we are processing. Furthermore, the multicolumn index with OR also requires multiple columns to be unique at a time, which also limits how much information is processed.

0.4 Question 7c

Given your findings from `Question 7`, which of the following statements is true? A. An index on the column being aggregated in a query will always provide a performance enhancement. B. A query finding the MIN(salary) will always benefit from an index on salary, but a query finding MAX(salary) will not. C. A query finding the COUNT(salary) will always benefit from an index on salary, but a query finding AVG(salary) will not. D. Queries finding the MIN(salary) or MAX(salary) will always benefit from an index on salary, but queries finding AVG(salary) or COUNT(salary) will not.

Justify your answer. Explain your answer based on your knowledge from lectures, and details of the query plans (your explanation should include why you didn't choose certain options). Your response should be no longer than 3 sentences.

Note: Your answer should be formatted as follows: "A because …"

D because index on salary allows us to order the salary in such a way that we can find the minimum and maximum quickly. However, in order to calculate average or count, we still need to process all of the salaries, regardless of their index.

0.5 Question 9c:

What difference did you notice when you added an index into the salaries table and re-timed the update? Why do you think it happened? Your answer should be no longer than 3 sentences.

Adding the index took longer to run through the query than without the index. This is clear because we must first access each row and assign an index value to that row.

1 Question 10: Project Takeaways

In this project, we explored how the database system optimizes query execution and how users can futher tune the performance of their queries.

Familiarizing yourself with these optimization and tuning methods will make you a better data engineer. In this question, we'll ask you to recall and summarize these concepts. Who knows? Maybe one day it will help you during an interview or on a project.

In the following answer cell, 1. Name 3 methods you learned in this project. The method can be either the optimization done by the database system, or the fine tuning done by the user. 2. For each method, summarize how and why it can optimize query performance. Feel free to discuss any drawbacks, if applicable.

Your answer should be no longer than ten sentences. Each method identification/discussion is 2 points.

Three of the methods I learned were creating an index, using boolean operators, and clustering by a given index. Creating an index can optimize query performance especially for situations where the order matters. For example, calculating the minimum and maximum with an index might be easier to do than without an index. Boolean operators allow for adding filtering conditions to minimize how much of the relation we are processing. Clustering by a given index is essentially the same as creating an index but with an additional feature. The CLUSTER command reorders the rows such that they are grouped based on index, which has the same benefits as the first method, but we may not need to sort in this situation.