Guide for product questions:

Some recurring themes throughout the answers:

In case studies, almost never they will explicitly tell you the metric you have to optimize for. Instead, they will just tell you the business goal, such as improve retention, growth, mobile usage, etc. It is your responsibility to mathematically translate the business goal into a metric. If you don't do it, it is an almost automatic fail. If you do it, 90% of the question has been answered

When they ask you to choose a metric for a new feature, product, or even a department, start from the high level company goal. In most cases, that is (or should be) growth. Then narrow it down. Example:

You can grow by increasing user retention or user acquisition
You can increase user retention by increasing user engagement
I pick metric X because it is related to user engagement and, if I move it, I can realistically expect to improve company growth
If you can't eventually link your metric to company growth, that's a huge warning that your metric isn't that useful

The high majority of metrics need a time threshold in order to be evaluated. Average likes per user can't be evaluated. Average likes per user per day can be evaluated. If the question is specifically about building a metric, don't forget this

If asked how to improve a product, the easiest way is: focus on actions that you want to incentivize and users are already performing today, but it takes them several steps on the app/website to finish them. There is no better proxy for feature demand that users already doing something despite a complicated user flow. Simplifying the flow will most likely improve your target metrics

If asked "Should we implement X" or "How to improve Z", the goal is simply explaining step by step how you would find the answer, if you had the data in front of you. They are testing how you approach a problem, not if you are a product visionary

When asked to optimize a long term metric like retention rate or lifetime value, the question means: find a short term metric that can predict the long term one, and then focus on optimizing that one

When asked to pick variables, pick a combination of user characteristics (age, sex, country, etc.) and behavioral ones (related to their browsing behavior)

Guide for take home challenges:
Things to keep in mind while doing a take-home challenge:

Unless otherwise specified, you want to use R or Python. Comment the code as much as possible. They are also evaluating how clear is your code. Also, anyone should be able to understand the conclusions of your take-home even if they are not familiar with the language you used.

Check the data. Never assume data is right. Always check data reliability and, if you find that some data doesn't make sense, clean it. This is also a big part of a data scientist job. There are companies that send take-homes that are only about identifying everything that is wrong with the data!

Take-home challenges are usually fairly open ended. Play to your strengths: this could mean spending more time on visualization, machine learning, product ideas, or business insights depending on your skills.

Don't make the solution over complicated. Focus on a few things and make sure the overall message is clear and consistent.

Along the same lines: when you have to build a machine learning model, don't spend days optimizing its accuracy (this is not Kaggle, it is real world). Pick a model, explain why you picked that model and use parameters that make sense. You can then say what you would do if you had more time to optimize it.

Focus on the business impact that your work could have. How would the company benefit from your analysis? What would you suggest as a next step?

If you find anything interesting in the data, by any means show it even if it is not related to the questions. If you find some info in the data that not even the hiring manager knows is there, you will pass the take-home for sure. After all, that's exactly why they will be hiring you: to discover things they don't know yet.

A take-home challenge is rarely the place where over emphasizing your theoretical knowledge (unless specifically required in a question).
Before extracting insights from a model, make sure your model predicts well. If your model doesn't predict well, its coefficients, splits, variable importance, etc. are totally irrelevant.

**Guide for SQL questions:**

Things to keep in mind regarding SQL shared screen coding challenges:

Make sure you are familiar with window functions.

As always with shared screen interviews, break down the problem as much as possible. This is typically easily doable with SQL. And then solve each small chunk. The goal of shared screen interviews is to check that you can code. Focus on sending that message more than trying to look super smart.

Some companies will ask you to solve the problem with just one query, i.e. you are not allowed to create temporary tables. Even in that case, break down the problem into smaller pieces and solve them independently. Then put these parts together into just one query via subqueries.

The data sets attached are just for you to practice and verify that your query works. Almost never in a shared screen interview you will have to work with actual data. Here there are no wrong data to find, nothing related to the preprocessing steps you usually have to take care of in a take-home challenge, or insights to find. Datasets are just to make sure the query is good.