As a Data Analyst, As you should be proficient in advance concepts of SQL . Here are a few advanced SQL methods:  
  
1. Common Table Expressions (CTEs) - CTEs allow you to create temporary result sets that can be referenced within a SELECT, INSERT, UPDATE, or DELETE statement. They are especially useful for breaking down complex queries into simpler, more readable components.  
  
2. Window Functions - These functions perform calculations across a set of table rows that are related to the current row. Unlike aggregate functions, window functions do not cause rows to become grouped into a single output row, allowing you to retain individual row information while performing calculations.  
  
3. Subqueries and Correlated Subqueries - Subqueries can be used to provide data that will be used in the main query as a condition to further restrict the data to be retrieved. Correlated subqueries, which reference columns from the outer query, can dynamically adapt their results based on the current row being processed.  
  
4. Pivot and Unpivot Operations - These operations enable you to transform rows into columns and vice versa, which is particularly useful for generating reports or reformatting data for analysis.  
  
5. Recursive Queries - Using recursive CTEs, you can perform hierarchical or recursive data analysis, such as traversing a tree structure or generating a series of numbers.  
  
6. Analytic Functions - These include ranking functions (e.g., RANK, DENSE\_RANK), NTILE, and LEAD/LAG, which provide sophisticated ways to analyze data within partitions of the data set.  
  
  
7. Advanced Joins - Mastering various types of joins (e.g., INNER JOIN, LEFT JOIN, CROSS JOIN) and understanding their implications on query performance and result set structure is crucial for combining data from multiple tables effectively.  
  
8. Indexing Strategies - Understanding how to create and use indexes efficiently can drastically improve the performance of your SQL queries, especially in large databases.  
  
9. Query Optimization - Techniques such as analyzing query execution plans, avoiding unnecessary columns in SELECT statements, and leveraging database-specific optimization features can help in writing more efficient and faster SQL queries.  
  
10. Dynamic SQL - This involves building SQL queries dynamically at runtime, which can be useful for applications that need to adapt queries based on user input or other dynamic factors.

SQL questions asked in many product based companies:  
  
Question 1:  
Imagine a table named "Books" with columns: BookID, Title, AuthorID. There's another table "Authors" with columns: AuthorID, AuthorName. Write a SQL query to fetch the authors who don't have any books associated with them.  
  
Question 2:  
You are given a table named "WorkoutSessions" with columns: UserID, SessionDate, IsCompleted (a boolean where 1 indicates completion and 0 indicates non-completion). Write a SQL query to identify users who have missed more than 5 consecutive workout sessions.  
  
Question 3:  
Consider a table named "Sales" with columns: SalespersonID, CustomerID, SaleDate. Write a SQL query to calculate the salesperson who made the highest number of sales each quarter.  
  
Question 4:  
You have a table named "WebsiteTraffic" with columns: PageID, VisitDate, Visits. Write a SQL query to find the top 5 pages that have shown the most significant increase in visits month-over-month.  
  
Question 5:  
You are provided with a table named "Rentals" with columns: RentalID, CustomerID, RentalDate, DueDate, ReturnDate. Write a SQL query to find out which rentals are currently overdue without being returned.  
  
Question 6:  
Consider a table named "OnlineExams" with columns: ExamID, StudentID, EnrollmentDate, CompletionDate. Write a SQL query to determine the exams with the highest failure rate (i.e., students enrolling but not passing).  
  
Question 7:  
You have a table named "CustomerReviews" with columns: CustomerID, ReviewDate, Rating (from 1 to 5). Write a SQL query to identify customers whose rating has been declining for the past 3 consecutive reviews.  
  
Question 8:  
There are two tables: "Articles" and "Likes". The "Articles" table has columns: ArticleID, Title, PublishDate, AuthorID. The "Likes" table has columns: LikeID, ArticleID, LikeDate. Write a SQL query to fetch the articles that have not received any likes within two weeks of their publishing date.  
  
Question 9:  
You are given a table named "Membership" with columns: MemberID, JoinDate, ExpirationDate. Write a SQL query to count the number of active memberships on the last day of each month in the past year.  
  
Question 10:  
Consider a table named "MuseumVisits" with columns: ExhibitID, ExhibitName, VisitorID, VisitDate. Write a SQL query to find the least visited exhibits in the last winter.  
  
Question 11:  
There are two tables: "Products" and "Manufacturers". The "Products" table has columns: ProductID, ProductName, ManufacturerID, UnitsSold. The "Manufacturers" table has columns: ManufacturerID, ManufacturerName. Write a SQL query to find manufacturers whose products, on average, have sold more than 50,000 units, but have fewer than 5 products listed.  
  
Question 12:  
You have a table named "HotelBookings" with columns: BookingID, CheckInDate, GuestID, HotelLocation. Write a SQL query to determine which hotel location has seen a consistent month-on-month increase in bookings over the last year.