Techniques for Optimizing Performance with LDV

1. **Using Mashups**
   One way to deal with diminishing the measure of data in Salesforce is to keep up enormous data collections in an alternate application, and afterward make that application accessible to Salesforce varying. Salesforce alludes to such a plan as a mashup on the grounds that it gives a fast, inexactly coupled combination of the two applications. Mashups use Salesforce introduction to show Salesforce-hosted data and remotely hosted information.
2. **Defer Sharing Calculation**
   In certain conditions, it may be proper to utilize a feature called **defer sharing calculation**, which permits clients to defer the preparation of sharing rules until after new clients, rules, and different content have been loaded. An association's head can utilize a defer sharing calculation permission to suspend and continue sharing estimations and to oversee two procedures: group member calculation and sharing rule calculation.
3. **Using SOQL and SOSL**
   A SOQL query is the equal to a SELECT SQL statement, and a SOSL query is a programmatic way of performing a text-based search.

SOQL	SOSL
You know which objects or fields contain the data.	You don’t know which object or field contains the data, and want to find it using the most efficient way possible.
· Get data from single or different objects that are related to each other · Count the number of records that meet particular criteria · Get data from number, date, or checkbox fields . Sort retrieved data as part of the query	· Get different objects and fields efficiently, and the objects may or may not be related to each other · Get data for a specific division in an organization using the divisions feature

4. **Deleting Data**
   The deletion of Salesforce data can have a profound effect on the performance of large data volumes. Salesforce’s **Recycle Bin** stores the deleted data. It does not actually remove the data, it flags data as deleted and you can access it through the Recycle Bin. This process is known as **soft deletion**. While the data is soft deleted, it still affects database performance because the deleted records still reside in Salesforce org, and these have to be excluded while performing any queries.
   Salesforce stores data in the Recycle Bin for 15 days, or until the Recycle Bin grows to a specific size. The data is deleted from the Recycle Bin after 15 days; or when the size limit is reached; or when the Recycle Bin is emptied using the UI, the API, or Apex.

Best Practices for Performance with LDV

1. **Reporting**
   - Reduce the number of records when querying by using a value (condition) in the data.
   - Reduce the number of joins by minimizing the number of: Objects and Relationships used in the report.
   - Reduce the amount of data by retrieving only required fields in SOQL query, reports and list view.

2. **Loading Data from the API**
   - Improving performance by using the Salesforce Bulk API when you have more than a few hundred thousand records.
   - Reduce data to transfer and process: Update only those fields that have changed.
   - Avoiding loading data into Salesforce by using mashups to create coupled integrations of applications.

3. **Extracting Data from the API**
   - Use the most-efficient operations: When retrieving more than a million records, use the Bulk API query capabilities.
   - Reduce the number of records to return: Be specific, use exact keywords and try to avoid wildcards to filter the data.
   - Improving efficiency by using the setup area for searching to enable language optimizations, and turn on enhanced lookups and auto-complete for better performance on lookup fields.

4. **SOQL and SOSL**
   - Allowing the indexed searches when SOQL queries with multiple WHERE filters cannot use indexes: Decompose the query and join their results.
   - Using the most appropriate out of both, SOQL or SOSL, for a given scenario.
   - Avoid timeouts on long SOQL queries: Optimize the SOQL query by using selective filters and limiting query scope.

5. **Deleting Data**
   - Delete large volumes of data: When deleting large volumes of data, that involves more records approximately more than one million, use the hard delete option of the Bulk API. Because deleting such a large volume of data has more time complexity.
   - Make the data deletion process more efficient: Always delete child records first when parent records have many childrens.

6. **General**
   - Avoid sharing computations and make deployments more efficient: Any user should not own more than 10000 records.
   - Improve performance: Use a data-tier strategy that disperse data over multiple objects, and get data on-demand from another object.