**Describe your idea for a database along with your thoughts for a partitioned view.**

A database is a structured collection of facts. The details are generally structured to design appropriate factors of truth in a way that facilitates procedures demanding these facts. A selection of details structured in such a way that a software program can quickly choose preferred items of information. One can think of a database as an electronic processing system. Traditional database are structured by fields, records, and files. A field is a single piece of information; a record is one complete set of fields; and a data file is a selection of data.

A partition is a division of a sensible database or its making up components into unique separate parts. Database partitioning is normally done for manageability, performance or accessibility reasons. A partitioned view joins side to side partitioned information from a set of member tables across one or more web servers, creating the information appear as if from one table. It allows data in large table to be split into smaller tables.

**How would you use this partitioned view?**

For using partitioned view, one first has to define individual table for every logical partition and then creating a “check” constraint on any column on which one want to partition so that optimizer gets to know that in which table a specific row must be present, and then defining a view using a “Union All” on every table for creating the partitioned view. Optimizer can then resolve any query against the partitioned view by making use of partition key so as to access only one of the physical tables which is comprising a specific logical partition, i.e. The information varies for each participant table are described in a CHECK constraint specified on the partitioning column. A view that uses UNION ALL to merge chooses of all the participant tables into only one result set is then described. When SELECT claims referencing the view specify a look for situation on the partition column, the question optimizer uses the CHECK constraint explanations to decide which participant table contains the rows

**How would you perform updates on a partitioned view and what is required?**

There are two ways where one can update a partitioned view.

1. First, if one has partitioned his/her tables using primary key, then one can update the view directly.
2. Second, if one is not using the partitioning column, one has to create triggers for creating updating view.

A partitioned view can be considered as updateable when view is a set of “SELECT” whose individual result sets are grouped into single by making use of “UNION ALL” statement.

**What benefits do you see with this partitioned view?**

Benefits of Partitioned view are as follows:

* **Improved availability:** The unavailability of a partition does not include the unavailability of the item. The query optimizer instantly eliminates unreferenced categories from the query plan so concerns are not impacted when the categories are not available.
* **Easier management of schema objects:** A partitioned item has pieces that can be handled either jointly or independently. DDL claims can operate partitions rather than whole platforms or indices. Thus, you can crack up resource-intensive projects such as restoring an index or table. For example, you can shift one table partition at a time. If a problem happens, then only the partition move must be replaced, not the table move. Also, losing a partition prevents performing several DELETE statements.
* **Reduced argument for allocated sources in OLTP systems**: In some OLTP techniques, partitions can reduce contention for a allocated source. For example, DML is allocated over many sections rather than one segment.
* **Enhanced query performance in information warehouses:** In an information factory, dividing can rate handling of ad hoc concerns. For example, a sales table containing a thousand series can be partitioned by quarter.

**References:**

Micrsosoft. (n.d.). Modifying Data in Partitioned Views. Retrieved from http://technet.microsoft.com/en-us/library/ms187067(v=sql.105).aspx

Oracle. (n.d.). Retrieved from http://docs.oracle.com: http://docs.oracle.com/cd/E11882\_01/server.112/e25789/schemaob.htm#CNCPT88858