**Steps in Partitioning the Data**

1. **Divide the Database:** Split the transaction database into smaller, manageable partitions.
2. **Find Local Frequent Itemsets:** For each partition, run the Apriori algorithm to find frequent itemsets within that partition.
3. **Combine Local Frequent Itemsets:** Merge the frequent itemsets from all partitions to create a candidate set of global frequent itemsets.
4. **Validate Global Frequent Itemsets:** Scan the entire database to count the support of the candidate global frequent itemsets to determine which are truly frequent.

**Example**

Assume we have a transaction database with the following transactions in the DB:

|  |  |  |
| --- | --- | --- |
| **Transaction ID** | | **Itemset** |
| T1 | {A, B, C} | |
| T2 | {A, B} | |
| T3 | {A, C} | |
| T4 | {B, C} | |
| T5 | {A, B, C, D} | |
| T6 | {B, C} | |
| T7 | {A, C} | |
| T8 | {A, B} | |
| T9 | {B, D} | |
| T10 | {A, B, C} | |

Let's partition this database into two partitions:

* **Partition 1:** T1, T2, T3, T4, T5
* **Partition 2:** T6, T7, T8, T9, T10

Assume the minimum support threshold is 3.

**Step 1: Find Local Frequent Itemsets**

**Partition 1:**

* **Pass 1 (1-itemsets):**
  + {A}: 4
  + {B}: 4
  + {C}: 4
  + {D}: 1
  + Frequent 1-itemsets: {A}, {B}, {C}
* **Pass 2 (2-itemsets):**
  + {A, B}: 3
  + {A, C}: 3
  + {B, C}: 3
  + Frequent 2-itemsets: {A, B}, {A, C}, {B, C}
* **Pass 3 (3-itemsets):**
  + {A, B, C}: 2 (not frequent)

Local frequent itemsets for Partition 1: {A}, {B}, {C}, {A, B}, {A, C}, {B, C}

**Partition 2:**

* **Pass 1 (1-itemsets):**
  + {A}: 4
  + {B}: 4
  + {C}: 3
  + {D}: 2
  + Frequent 1-itemsets: {A}, {B}, {C}
* **Pass 2 (2-itemsets):**
  + {A, B}: 3
  + {A, C}: 2
  + {B, C}: 3
  + {B, D}: 1
  + Frequent 2-itemsets: {A, B}, {B, C}
* **Pass 3 (3-itemsets):**
  + {A, B, C}: 2 (not frequent)

Local frequent itemsets for Partition 2: {A}, {B}, {C}, {A, B}, {B, C}

**Step 2: Combine Local Frequent Itemsets**

Combine the frequent itemsets from both partitions:

Combined local frequent itemsets: {A}, {B}, {C}, {A, B}, {A, C}, {B, C}

**Step 3: Validate Global Frequent Itemsets**

Now, scan the entire database to count the support of these combined itemsets:

* **Pass 1 (1-itemsets):**
  + {A}: 7
  + {B}: 8
  + {C}: 7
  + Frequent 1-itemsets: {A}, {B}, {C}
* **Pass 2 (2-itemsets):**
  + {A, B}: 5
  + {A, C}: 4
  + {B, C}: 6
  + Frequent 2-itemsets: {A, B}, {A, C}, {B, C}

So, the global frequent itemsets are:

* Frequent 1-itemsets: {A}, {B}, {C}
* Frequent 2-itemsets: {A, B}, {A, C}, {B, C}

By partitioning the database, we can independently find local frequent itemsets within each partition. These local frequent itemsets are then combined and validated against the entire database to determine the global frequent itemsets.

**Advantage:**

This approach can significantly reduce the computational complexity and memory usage compared to processing the entire database as a whole.