**Lightweight Transactions**

In Cassandra, lightweight transactions (also known as LWTs) are a feature that provides limited support for ACID properties within a single partition. LWTs allow you to perform simple compare-and-set operations to ensure data consistency during write operations. They are a compromise between full distributed transactions and the eventual consistency model of Cassandra.

LWTs are based on the use of conditional updates, where a write operation is performed only if certain conditions are met. This allows you to perform atomic operations within a single partition while maintaining some level of data consistency.

Here's an example to illustrate the concept of lightweight transactions in Cassandra:

Let's assume you have a Cassandra table named "inventory" that tracks the quantity of a particular item in stock. The table has the following structure:

| CREATE TABLE inventory (  item\_id UUID PRIMARY KEY,  quantity INT ); |
| --- |

Now, suppose you want to reduce the quantity of an item by a certain amount but ensure that the operation only succeeds if the current quantity is greater than or equal to the requested reduction. This is where lightweight transactions come into play.

Using LWTs, you can perform a conditional update that checks the current quantity before applying the reduction. Here's an example query:

| UPDATE inventory SET quantity = quantity - 5 WHERE item\_id = '123456' IF quantity >= 5; |
| --- |

In the above query, the `IF quantity >= 5` condition ensures that the update is applied only if the current quantity is greater than or equal to 5. If the condition is not satisfied, the write operation will fail, and no changes will be made to the database. This ensures that the inventory quantity is not reduced below a certain threshold.

LWTs in Cassandra rely on a two-phase commit process behind the scenes, which introduces additional latency and coordination overhead compared to regular write operations. Therefore, they should be used judiciously for critical scenarios where data consistency is essential within a single partitions