1. Region

A region is required to store and retrieve data from the cache. Region is an interface that extends **java.util.Map**to perform basic data access using familiar key/value semantics. The Region interface is wired into classes that require it, so the actual region type is decoupled from the programming model. Typically, each region is associated with one domain object, similar to a table in a relational database.

**It’s very important for you to know that GemFire doesn’t manage the associations or enforce relational integrity among regions.**

The data for a region resides in memory in the JVM for the server. When a region is persistent, the data for that region is also preserved in a disk store. The disk store uses a directory in a file system to save the operations on regions. Unlike non-persistent regions, the data for a persistent region is available even after a period where no servers for that region are running.