EOF-Using EOF-Faulting

Faulting
When an enterprise object is about to retrieve data for one or more of its properties, it is required to notify other objects that it's about to take action. It does this by invoking willRead. An EOGenericRecord class using takeStoredValueForKey to access data automatically invokes willRead.

This method is a part of Enterprise Object's faulting mechanism. Faulting is the mechanism by which Enterprise Objects delays the full initialization of an enterprise object instance until that object's data is actually required. You can think of faulting as creating a shell of an enterprise object that includes just some (or perhaps none) of its data. See Figure 6-3 for an illustration.

Faulting reduces memory consumption and provides a performance improvement to applications by delaying fetches to the database until data is actually needed. Database fetches are expensive, especially during the resolution of relationships. Often, an enterprise object needs a reference to a particular relationship but doesn't necessarily need the data in that relationship. By providing that enterprise object with a reference to the relationship using a fault, you save the expense of performing a fetch if it's not necessary.

The default implementation of willRead checks to see if its receiver has already been fully initialized (that is, if its receiver is a full-formed enterprise object). If it hasn't been fully initialized, it fills the object with values fetched from the database. Before an application attempts to invoke a method on a particular enterprise object, you must ensure that object has already fetched its data. To ensure that an enterprise object is in the correct state before its data is accessed, you need to invoke willRead, typically in "get" methods. (Enterprise objects don't have to invoke willRead in "set" methods because the default implementation of willChange invokes willRead internally.)

Again, if you use EOGenericRecord subclasses that don't access their properties with fields, willRead is handled for you by storedValueForKey.

(Copied from the Apple Documentation)