

Contents

- [Introduction to Aggregates](#)
- [Only the Root can create internal entities](#)
- [CreateLineItem example](#)

Only the Root entity of an Aggregate should be able to create other members of the Aggregate. The Root should have a Create method for each member. Member classes should be written such that only the root entity can create them. The Order and its LineItems is the canonical example of an Aggregate. This topic introduces the vocabulary of aggregates and explains how to implement the member Create methods.

Introduction to Aggregates

Entities typically cluster into **Aggregates** in which one entity is the designated **Root** and the others are subordinate internal members. The root should control all access to the internal members.

An Order entity and its LineItem entities constitute an "Order Aggregate". The Order is the Root of the aggregate; the LineItems are its internal, subordinate members.

From the domain perspective, the Order and its LineItems are one thing, a conceptual Order, represented by the concrete root Order entity. The LineItems have no meaningful existence apart from their parent Order. You expect to get them from the Order; you wouldn't try to query for them independently of their parent order.

A new, deleted, or changed LineItem is a modification of the Order Aggregate as surely as changing the Order date. You make this insight tangible by observing the following rule: when you would save a LineItem, you save its (modified) parent Order entity at the same time. They travel together in the same transactional package.

Speaking of transaction, the root Order entity is the keeper of the "lock" or "concurrency property" for the Aggregate as a whole. Because the Order is always part of the save transaction and is always modified when any part of the aggregate is modified, concurrency conflicts are detected and resolved with the root Order entity.

Only the Root can create internal entities

When you adopt this way of thinking, you realize that LineItems cannot be created directly. Only the parent Order should be allowed to create them. It follows that:

- 1) The Order class should have a CreateLineItem method.
- 2) You should [block attempts to create a OrderDetail](#) directly by making all OrderDetail constructors *internal*.

It also follows that you must ask the parent Order to delete or remove a member LineItem. You should not be able to delete a LineItem directly; you must go through its parent Order.

CreateLineItem example

++ TBD