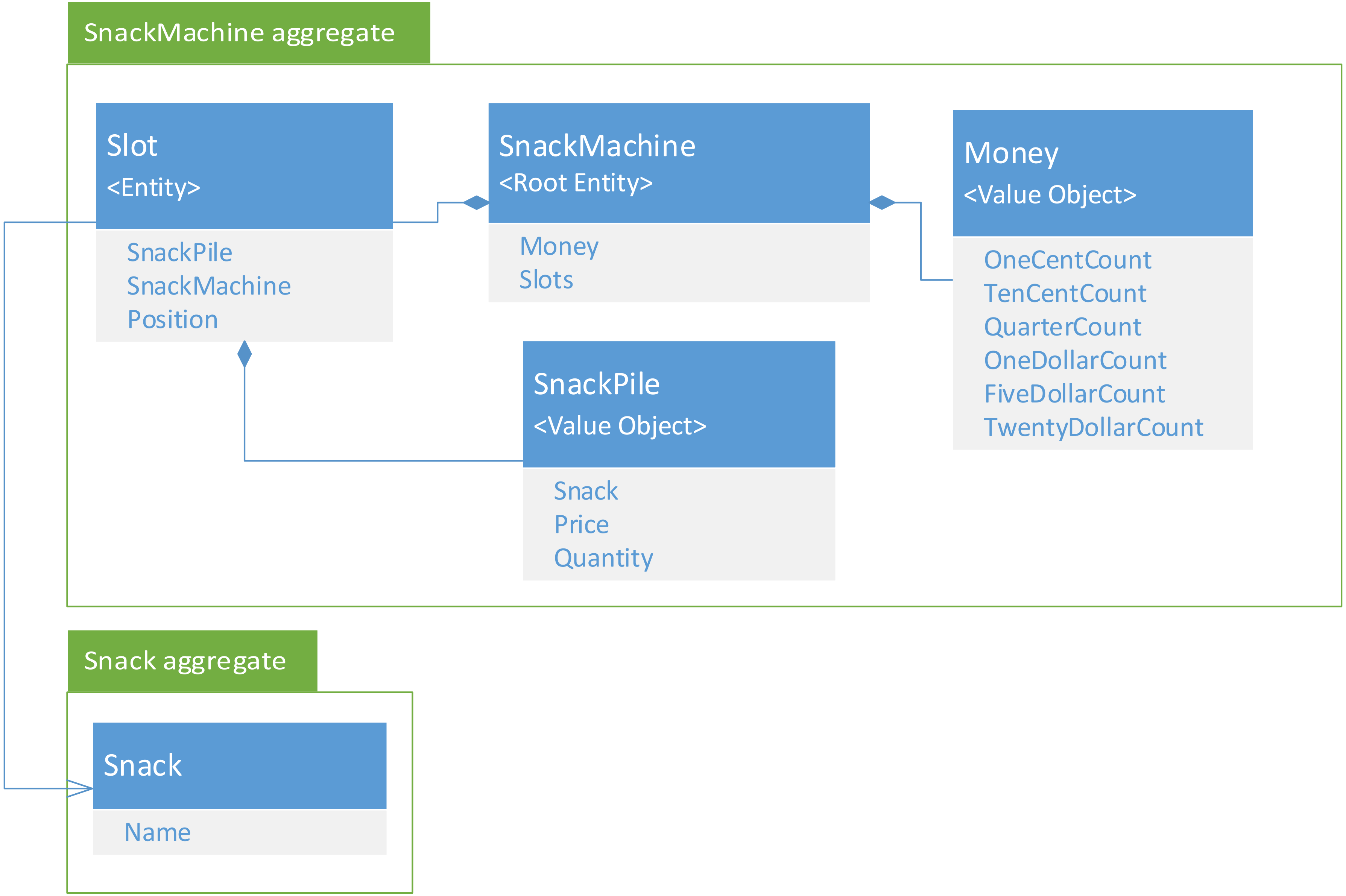
Introducing Repositories

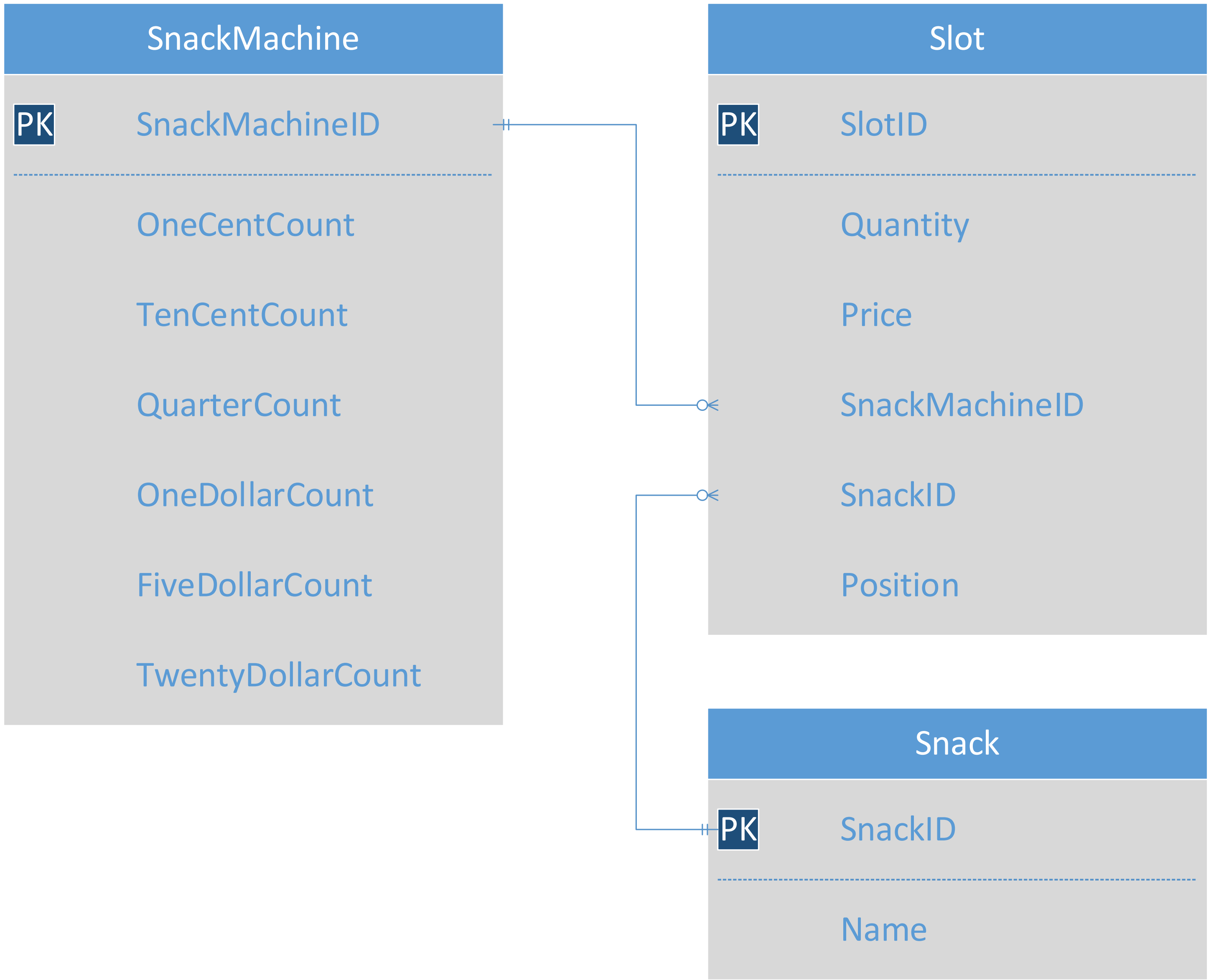
Vladimir Khorikov

@vkhorikov | www.enterprisecraftsmanship.com

Adjusting the Database for the New Entities



Adjusting the Database for the New Entities



SnackMachine

PK SnackMachineID

OneCentCount

TenCentCount

QuarterCount

OneDollarCount

FiveDollarCount

TwentyDollarCount

Slot

PK SlotID

Quantity

Price

SnackMachineID

SnackID

Position

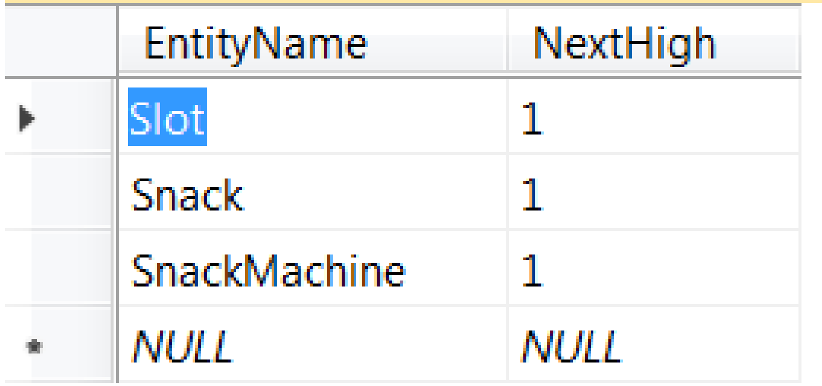
Snack

PK SnackID

Name

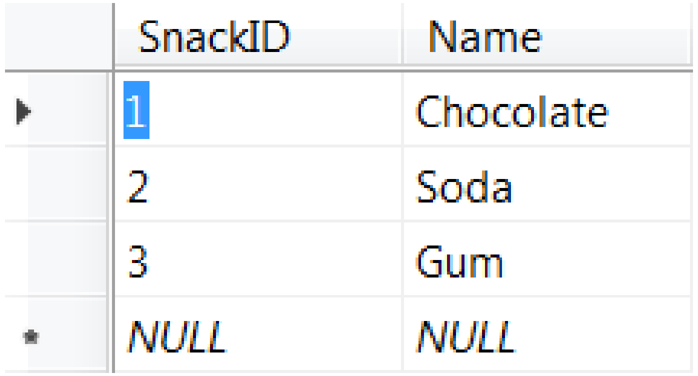
Adjusting the Database for the New Entities

Ids table:



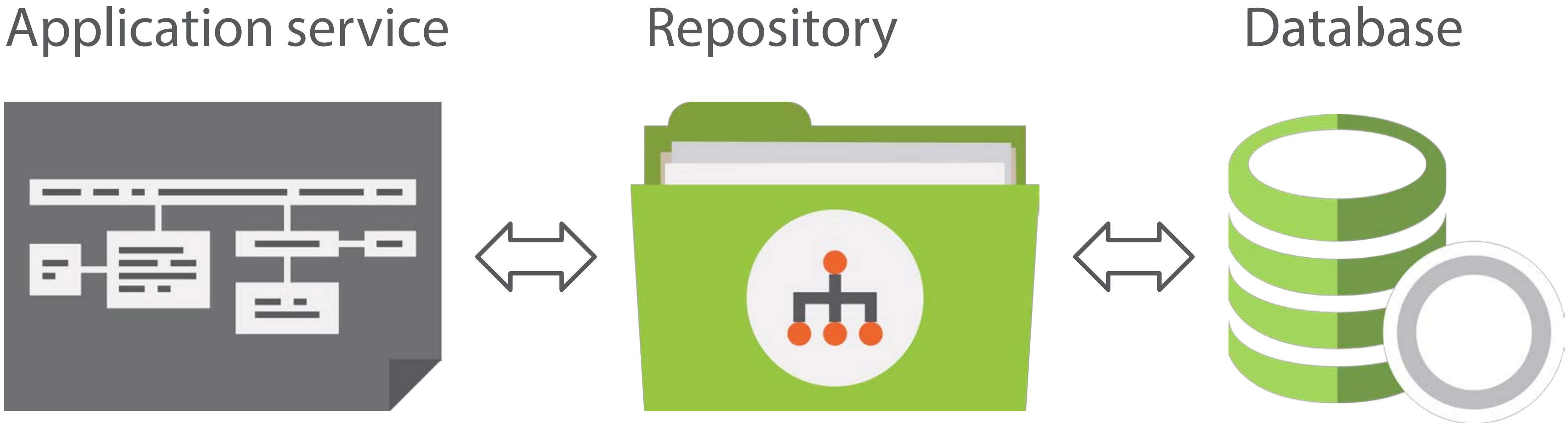
	EntityName	NextHigh
▶	Slot	1
	Snack	1
	SnackMachine	1
*	*NULL*	*NULL*

Snack table:



	SnackID	Name
▶	1	Chocolate
	2	Soda
	3	Gum
*	*NULL*	*NULL*

Repositories



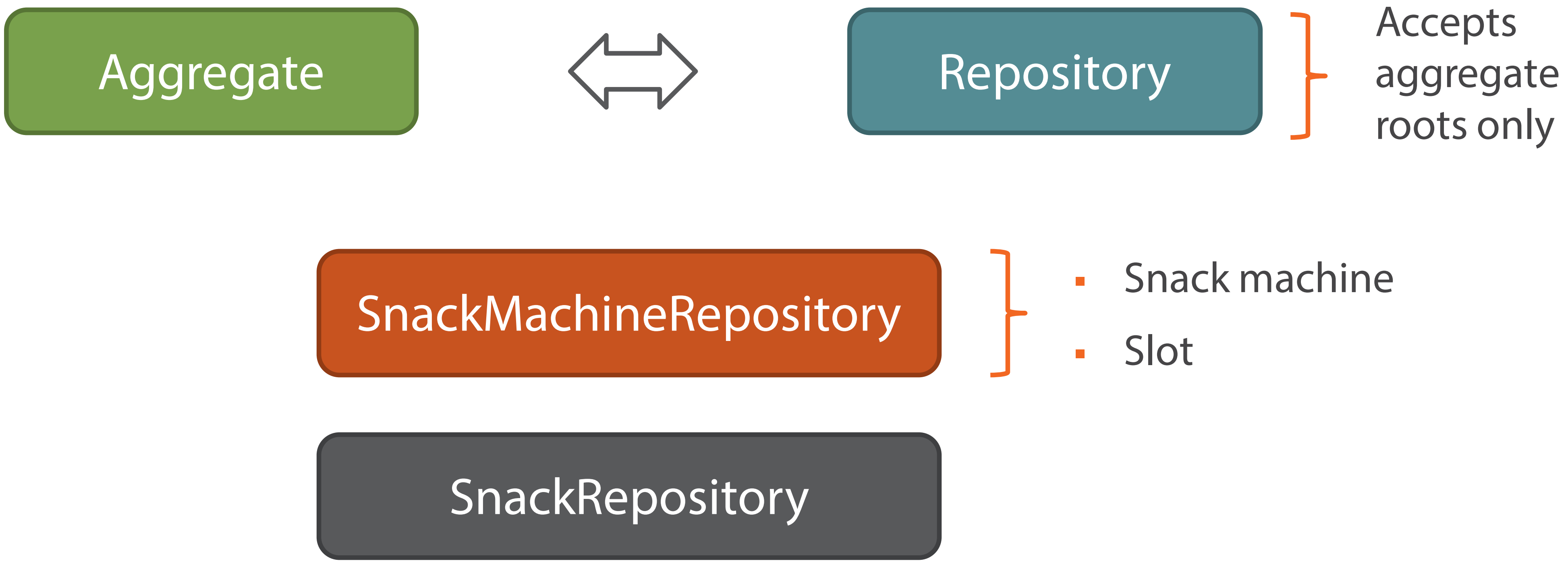
Application service

Repository

Database

Repositories

```
var repository = new SnackMachineRepository();
SnackMachine snackMachine = repository.GetById(1);
```



Aggregate

Repository

Accepts aggregate roots only

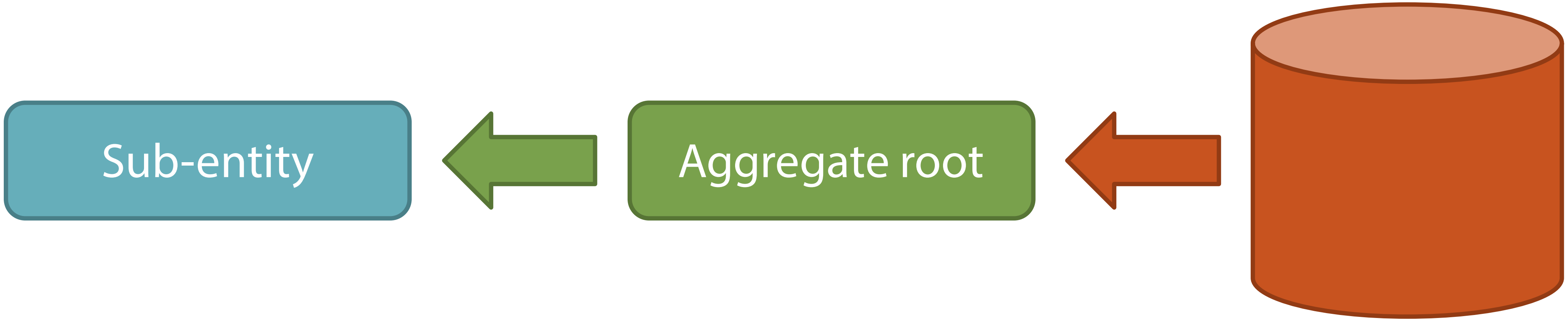
SnackMachineRepository

- Snack machine
- Slot

SnackRepository

Repositories

How to get a sub-entity?



```
var repository = new SnackMachineRepository();
SnackMachine snackMachine = repository.GetBySlotId(slotId);
```

Repositories

Repository

□ Uses an ORM

□ Works with the database directly

Repository Base Class

```
public abstract class Repository<T>
    where T : AggregateRoot
{
    public T GetById(long id)
    {
        using (ISession session = SessionFactory.OpenSession())
        {
            return session.Get<T>(id);
        }
    }

    public void Save(T aggregateRoot)
    {
        using (ISession session = SessionFactory.OpenSession())
        using (ITransaction transaction = session.BeginTransaction())
        {
            session.SaveOrUpdate(aggregateRoot);
            transaction.Commit();
        }
    }
}
```

Repository Base Class

```
public class SnackMachineRepository : Repository<SnackMachine>
{
    public IReadOnlyList<SnackMachine> GetAllWithSnack(Snack snack)
    {
        /* ... */
    }

    public IReadOnlyList<SnackMachine> GetAllWithMoneyInside(Money money)
    {
        /* ... */
    }
}
```

```
public class SnackMachineSqlRepository
{
}
```

Summary

- Repositories encapsulate all communication with the external storage
  - Single repository per each aggregate
  - Public API works with aggregate roots only
  - Perform persistence of sub-entities behind the scenes
- Define reference data in your domain model explicitly

In the Next Module

Bounded contexts