**Bank Application**

**Solution Document**

**By Vivek Malhotra**

# Entity Relationship

Below is the entity relationship diagram for Bank Application. Firstly we describe the entities.

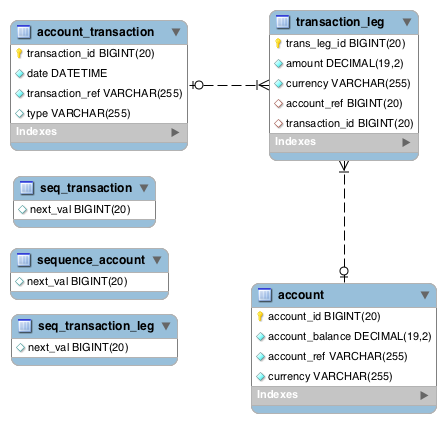
## Entities

Account – This entity defines the details of an account. It has properties like balance and currency.

AccountTransaction – This entity denotes transactions involving atleast 2 or more accounts. It has 2 or more AccountTransactionLeg.

AccountTransactionLeg – This defines a transaction performed on one account. Each AccountTransaction should have atleast two AccountTransctionLeg.

## ER Diagram



# Technology stack

I based my solution on:

* spring boot 1.4 release as container
* jpa as orm
* mysql database for use in production
* h2 database for persistence in test
* maven 3.x for project build management

# Application design

## Important classes

### AccountServiceImpl

Implementation of AccountService.

### TransferServiceImpl

Implementation of TransferService.

### AccountRepository

This is a JpaRepository and contains two functions

* save(Account account) – this saves an Account object and returns the managed entity.
* findByAccountRef(String accountRef) – this is to find an account with an account ref.

### TransactionRepository

This is a JpaRepository and contains three functions

* Save(AccountTransaction) – this saves the account transaction.
* findByAccountRef(String accountRef) – This a @Query based on JPQL to get all the AccountTransaction associated with an account.
* findByTransactionRef(String ref) – This is to find a transaction based on transaction reference.

### AccountAlreadyExistsException

This is a business exception which is thrown when creating an account with reference if it already exists

# Application Setup

## Database in Production

Use the following script to setup the mysql database for production

<PROJ\_DIR>/src/main/resources/bank\_ddl.sql