Account Transaction

Description

Objective:

To work with One to Many Relationship between entities and implement as Bidirectional and use of Entity Relationship Dependency.

Concept Explanation:

1. When one instance of an entity is associated with multiple instances of another entity, then we call that relationship **one-to-many.**
2. That relationship will be **bi-directional** if both the entities are aware of each other and can navigate from one entity to another.

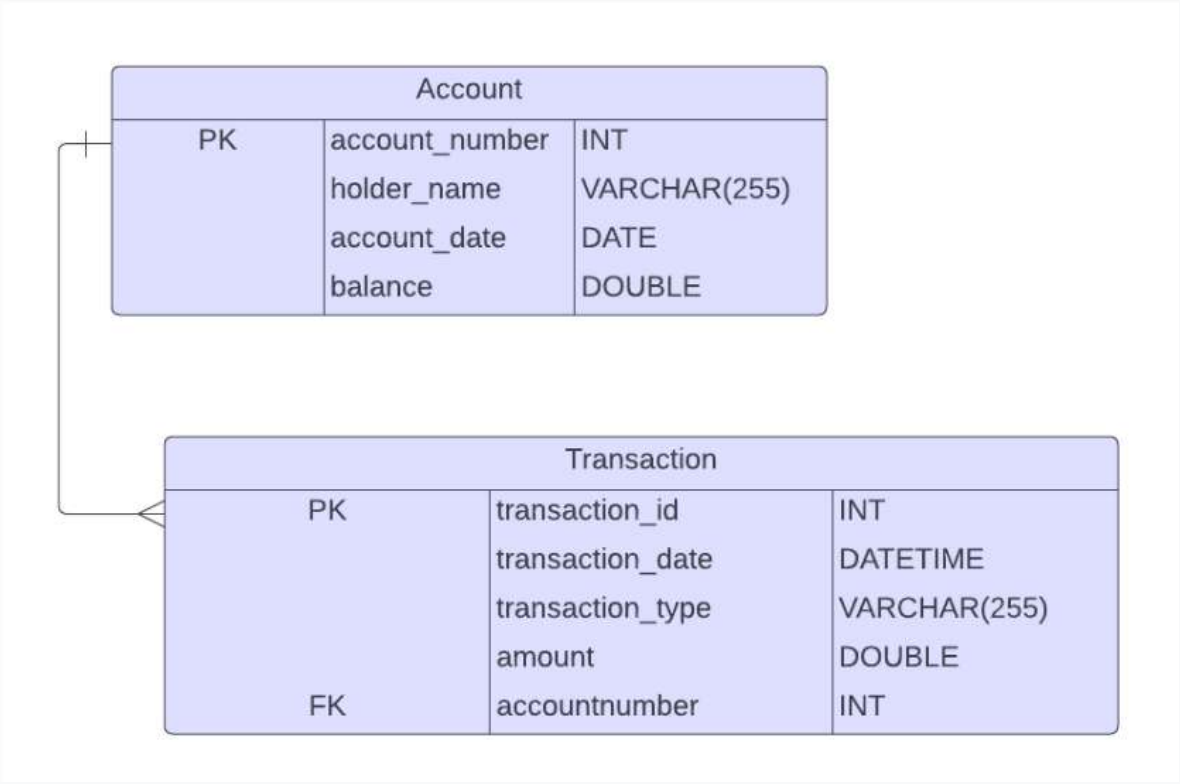
Concept Implementation:

1. We establish a one-to-many relationship between the Account and Transactions entities by providing appropriate annotation with the proper attribute pointing to the Account above the transactionList attribute in the Account entity.
2. The Transactions entity holds a reference to the Account entity, establishing a many-to-one relationship, where each transaction is associated with one Account. Provide proper annotation that specifies the foreign key column in the Transaction entity, enabling bidirectional retrieval of Account and Transactions details.

Account Transactions

Alia Bank wants to store the details of account and transactions performed on those accounts. Develop a Spring Data JPA application to perform the task.

Database Design:



Note:

- Account and Transactions has one to many relationships.
- Establish the relationship between Account and Transactions as Bi-directional.

Functionalities:

1. Add Account

Alia Bank stores the account information by providing the details like accountNumber, holderName, accountDate and balance. This information is stored in Account table.

Utility class:

Component Name	Method Name	Method Description	Arguments	Output
AccountDAO	addAccount	This method stores the account details into the "Account" table.	Account account	void

Model class

Component Name	Attributes	Methods
Account	accountNumber - int holderName -String accountDate - LocalDate balance - double The transaction details - the attribute name should be "transactionList"	getter and setter methods
Transactions	transactionId - int transactionDate - LocalDateTime transactionType - String amount - double The account information- the attribute name should be "account".	getter and setter methods

2. Perform Transaction on an Account

When a transaction is performed on an account, the Alia Bank wants to add that transaction to the account by providing the details like transaction id, transaction date, type of transaction (can be either debit or credit) and the amount . The transaction will be added to the already existing account.

Utility class:

Component Name	Method Name	Method Description	Arguments	Output
AccountDAO	performTransactionOnAccount	To the already existing account add the transaction. This method should add the transaction object into "Transaction" table.	Int accountNumber, Transaction transaction	void

3. Retrieve transaction for a particular account number for a specific period

This functionality should list out the trasactions performed on a particular account for a specific period.

Utility class:

Component Name	Method Name	Method Description	Arguments	Output
AccountDAO	retrieveTransactionDetails	This method should return the transaction for a specific period for a particular account.	int accountNumber, LocalDate startDate, LocalDate endDate	List<Transaction>

Design Constraints

1. All the classes and methods should have public access specifier.

2. All the attributes should have private access specifier.

3. Use Annotation for all the mapping.

4. accountNumber and transactionId should be marked as primary key.

5. Use Spring data JPA API for persisting the object into database.

6. The below tables should get created by hibernate automatically with the proper parent and child relationship.

7. The table should be created with correct table name and column name as specified in the database design diagram.