Entities or DTOs - Which projection should you use?

JPA and Hibernate allow you to use DTOs and entities as projections in your JPQL and Criteria queries. And I get often asked, if it matters which projection you use.

The answer is: YES! Choosing the right projection for your use case can have a huge performance impact.

Projections for write operations

Entity projections are great for all write operations. Hibernate and any other JPA implementation manages the state of your entities and creates the required SQL statements to persist your changes in the database. That makes the implementation of most create, update and remove operations very easy and efficient.

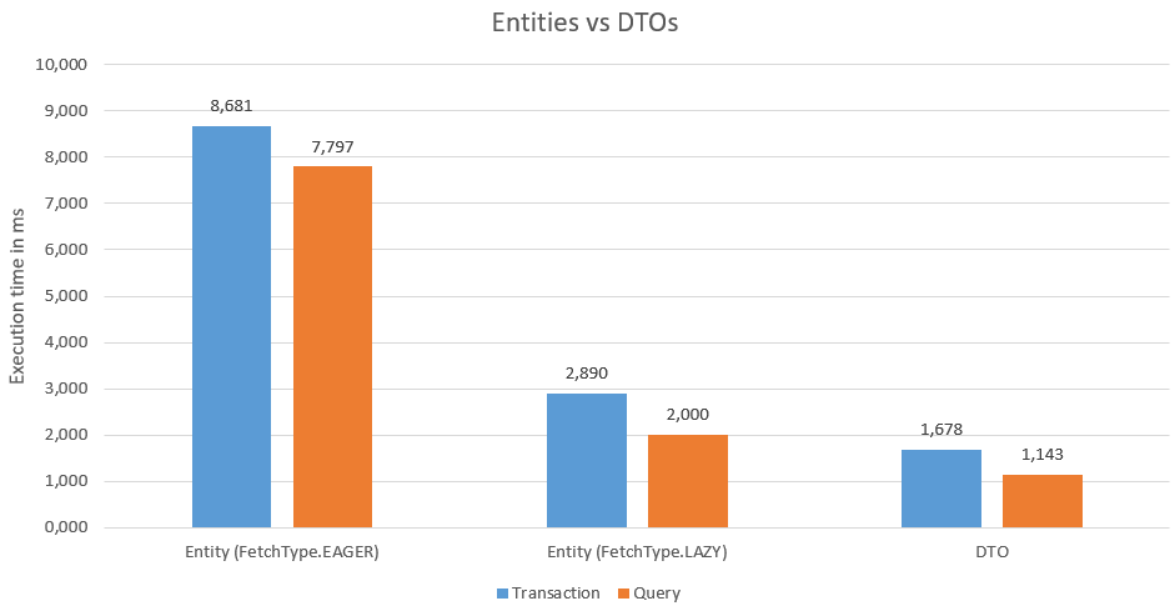
```
EntityManager em = emf.createEntityManager();
em.getTransaction().begin();

Author a = em.find(Author.class, 1L);
a.setFirstName("Thorben");

em.getTransaction().commit();
em.close();
```

Projections for read operations

But read-only operations should be handled differently. Hibernate doesn't need to manage any states or perform dirty checks if you just want to read some data from the database.

So, from a theoretical point of view, DTOs should be the better projection for reading your data. But does it make a real difference?

I did a performance test to answer this question.



And the result was impressive:

1. The query that selected the DTOs performed ~40% better than the one that selected entities.
2. It gets even worse, if you don't use all FetchType for all associations. One to-one association with its default eager fetching more than tripled the execution time.

So, better spend the additional effort to create a DTO for your read-only operations and use it as the projection.