# Hibernate todo list

1. В примере TestPokymorphism при добавлении аннотации @Polymorphism(type = PolymorphismType.EXPLICIT)к одному из подклассов и последующем полиморфном запросе к родительскому классу вытягивается вся иерархия классов. Почему? (Возможно решение здесь - <https://stackoverflow.com/questions/27966134/hibernate-explicit-polymorphism-with-joined-inheritence-strategy>:

The documentation is indeed misleading, but [ANN-314](https://hibernate.atlassian.net/browse/ANN-314) JIRA issue explains what the Hibernate authors had in mind:

Polymorphism explicit only applies on root entities and prevents queries naming a (unmapped) superclass to return mapped sub entities

**If the parent class was not mapped** (e.g. being annotated with @Entity) **and the sub-classes were having explicit polymorphism**, then querying for super-class would not return the sub-class.

Но если root класс сделать не энтити (https://docs.jboss.org/hibernate/orm/5.4/userguide/html\_single/Hibernate\_User\_Guide.html#entity-inheritance-polymorphism), то вылезает ошибка“Root\_classisnotmapped”)

1. В примере ”TestQualifiedPathExpressions” не работает query с **entry()**, хотя аналогичный пример работает с Key() и Value(). Дополнительно разобрать подробнее, как работает мап в этом примере (посмотреть как лучше добавлять timestam для phone, там есть метод addCall()). Разобрать доменную область для HQL/JPQL примеров.

Notes:

17.03.2021

- Multiple roots.

CriteriaQuery query = builder.createQuery();

Root<Men> men = query.from( Men.class );

Root<Women> women = query.from( Women.class );

Predicate menRestriction = builder.and(

builder.equal( men.get( Men\_.gender ), Gender.MALE ),

builder.equal( men.get( Men\_.relationshipStatus ), RelationshipStatus.SINGLE )

);

Predicate womenRestriction = builder.and(

builder.equal( women.get( Women\_.gender ), Gender.FEMALE ),

builder.equal( women.get( Women\_.relationshipStatus ), RelationshipStatus.SINGLE )

);

query.where( builder.and( menRestriction, womenRestriction ) );