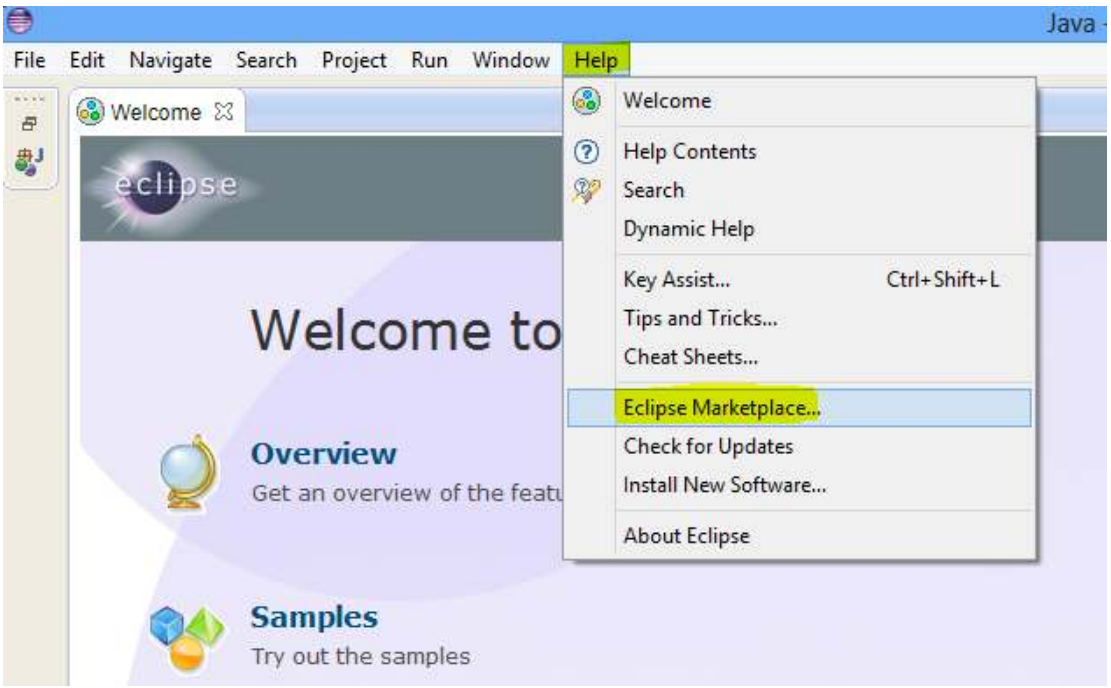
Hibernate İle Veritabanındaki Tablolara Karşılık Gelecek Java Sınıflarının(POJO) Otomatik Oluşturulması

Hibernate **JBoss** Toolbar ile **reverse engineering** yaparak veritabanındaki tabloları karşılık gelecek **Java sınıfları(POJO-Plain** Old Java Object), **Dao** sınıfları, **hbm.xml** dosyaları yaratabiliriz. Bu sayede çok fazla sayıda veritabanı tablolarını **manuel** olarak yaratmaktan **kurtulmuş oluruz.** Bu işlemin yapılmasına reverse engineering(**tersinir mühendislik**) denir. Bu makelede **Eclipse** ide kullanarak bu işlemin nasıl yapılacağı anlatılacaktır.

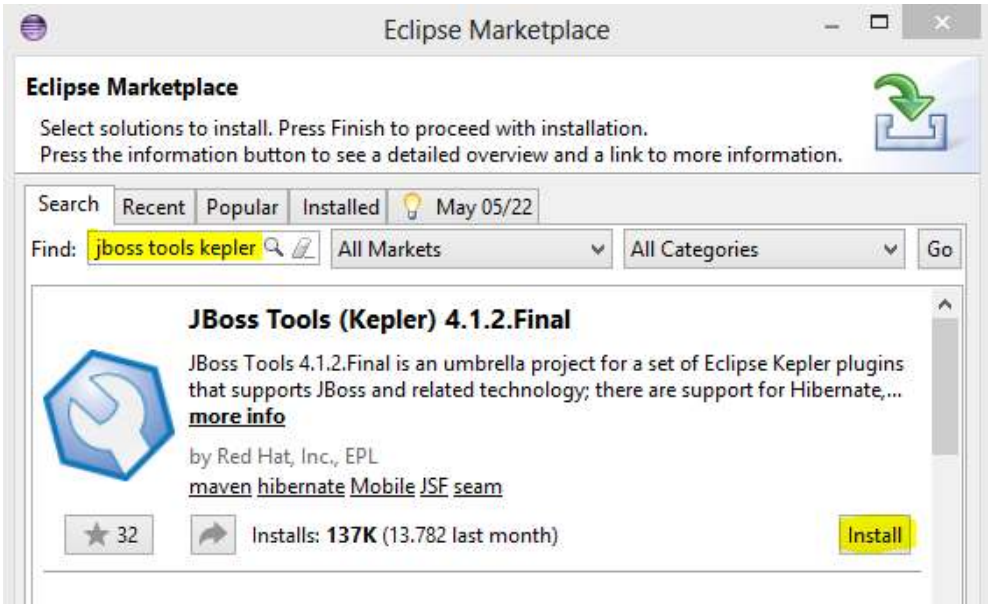
Eclipse İle Reverse Engineering

1. Öncelikle Eclipse Kepler 4.3 uygulaması indirilip çalıştırılır

2. Eclipse programını çalıştırdıktan sonra **Help(Yardım)** kısmından **Eclipse MarketPlace** kısmına girilir

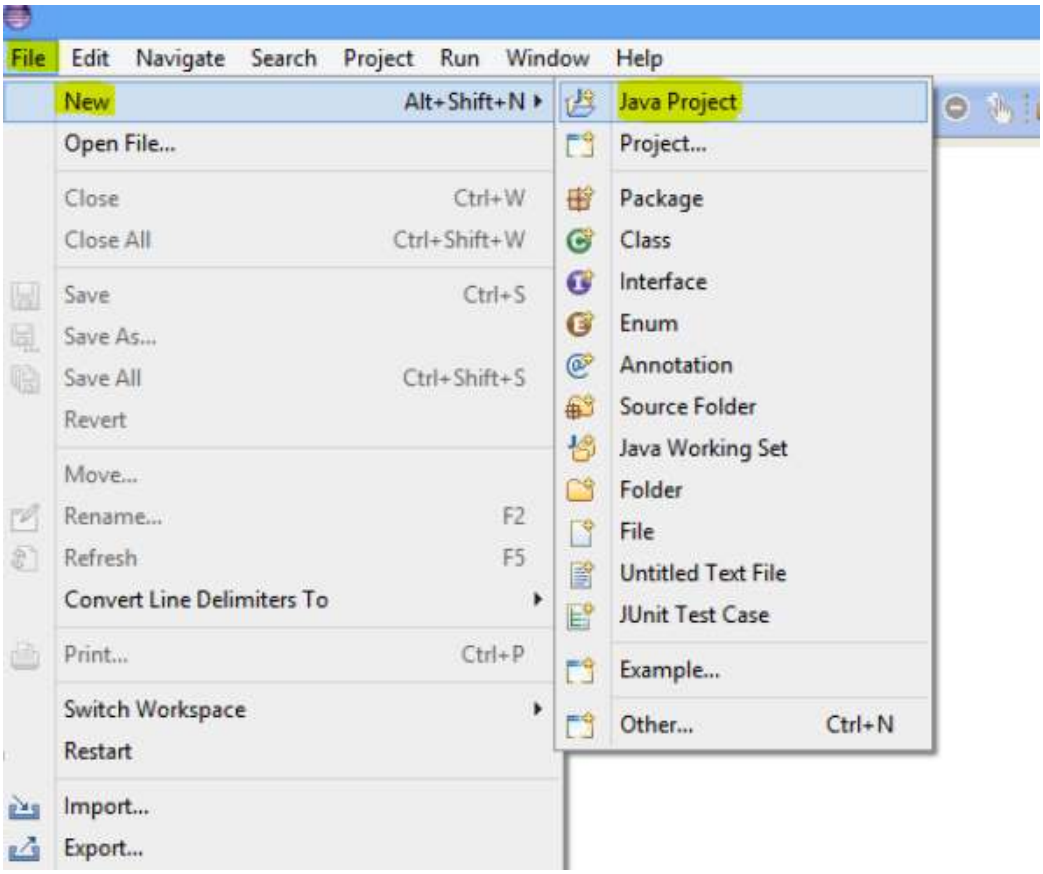


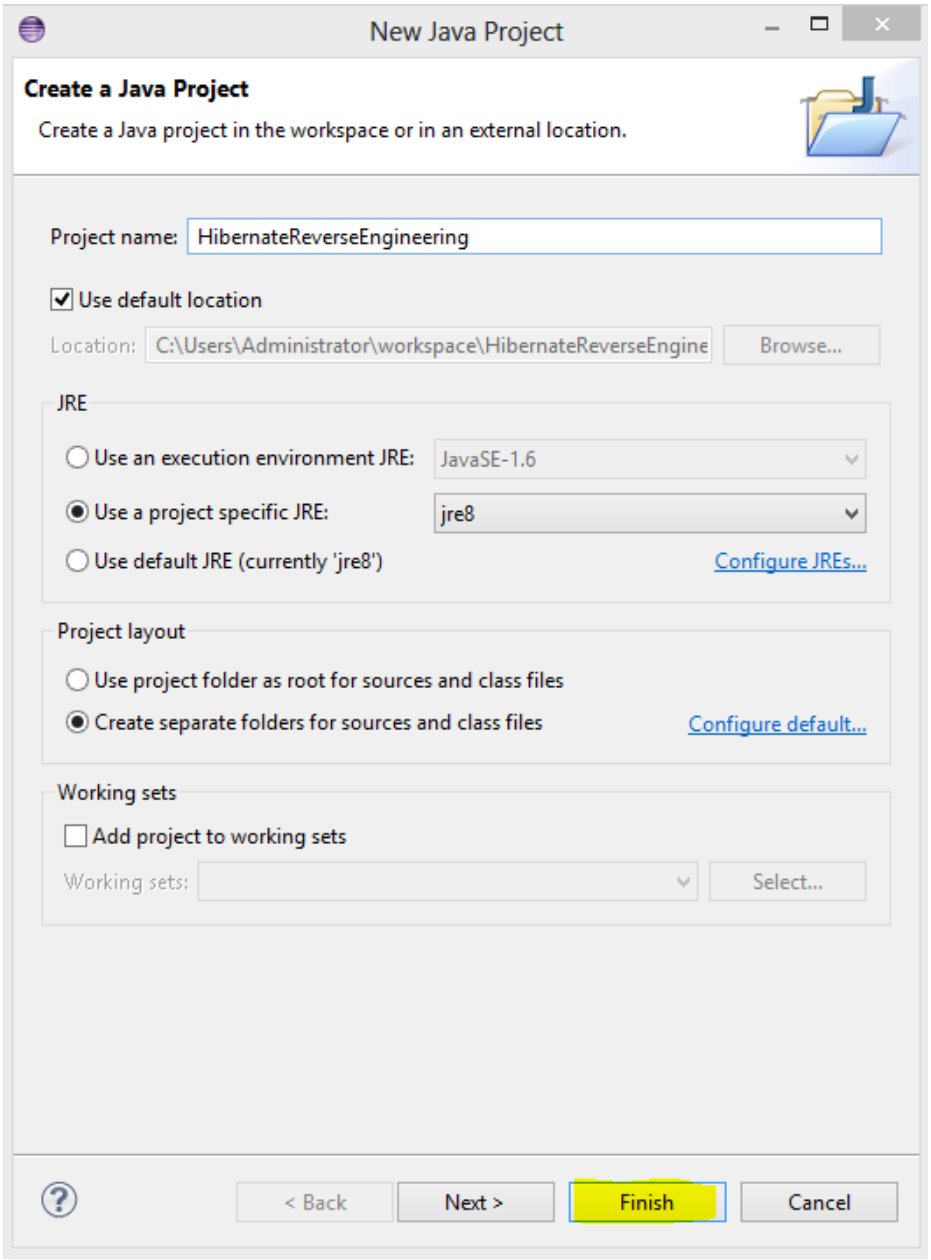
3. Daha sonra **Find(Arama)** kısmına **jboss tools kepler** yazılır sonuçlarda yer alan **JBoss Tools(Kepler)** plugin'i kurulur.



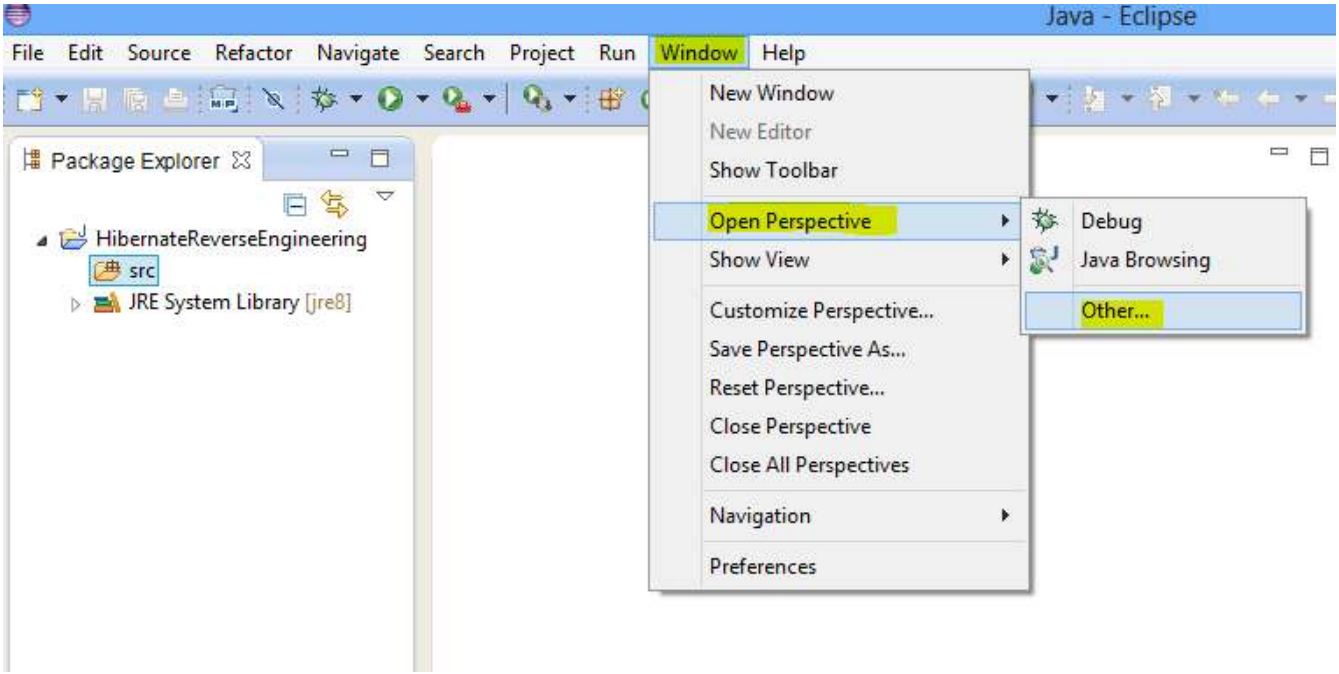
Bu aşamalar tamamlandıktan sonra aşağıdaki adımlar yapılır:

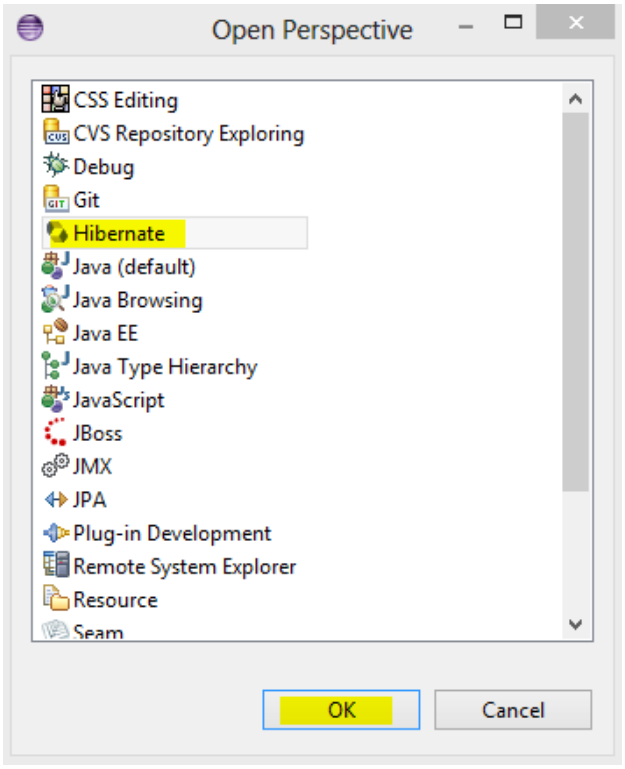
1. **Yeni** bir Java **projesi** oluşturulur



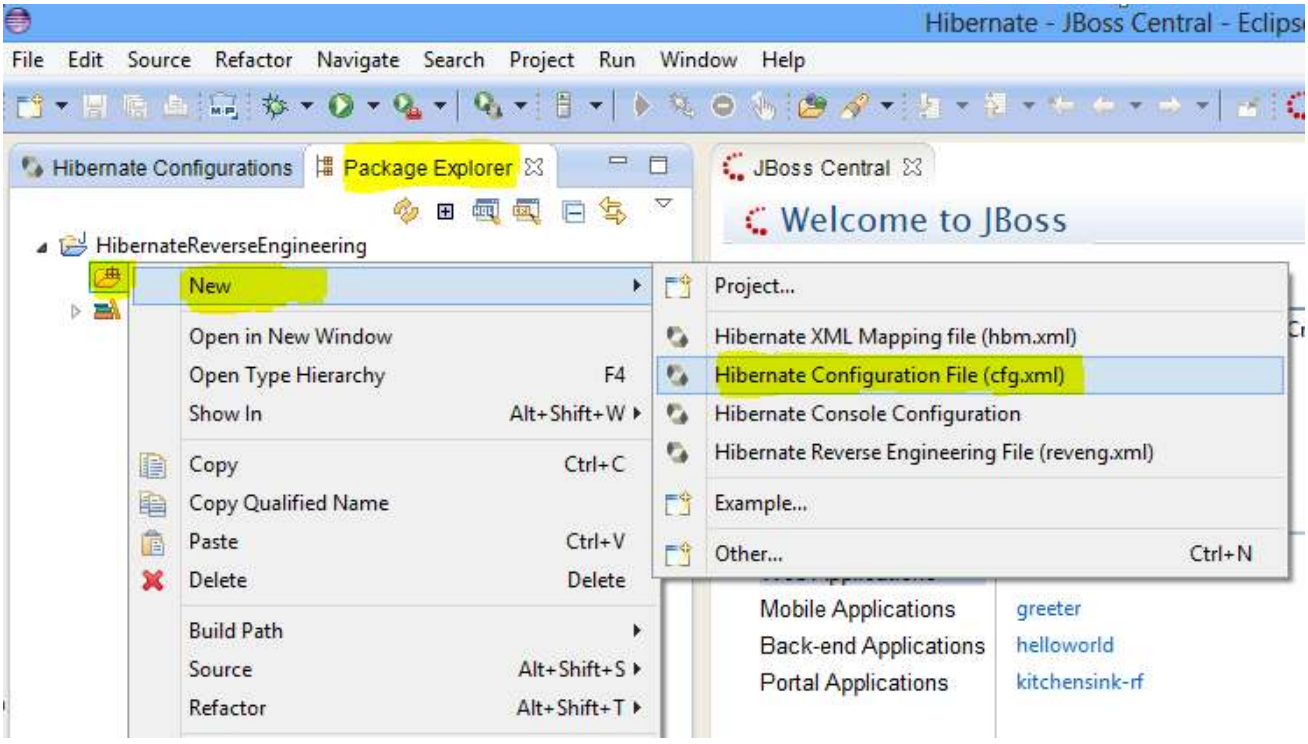


2. Window -> Open Perspective kısmından **Hibernate** seçilir

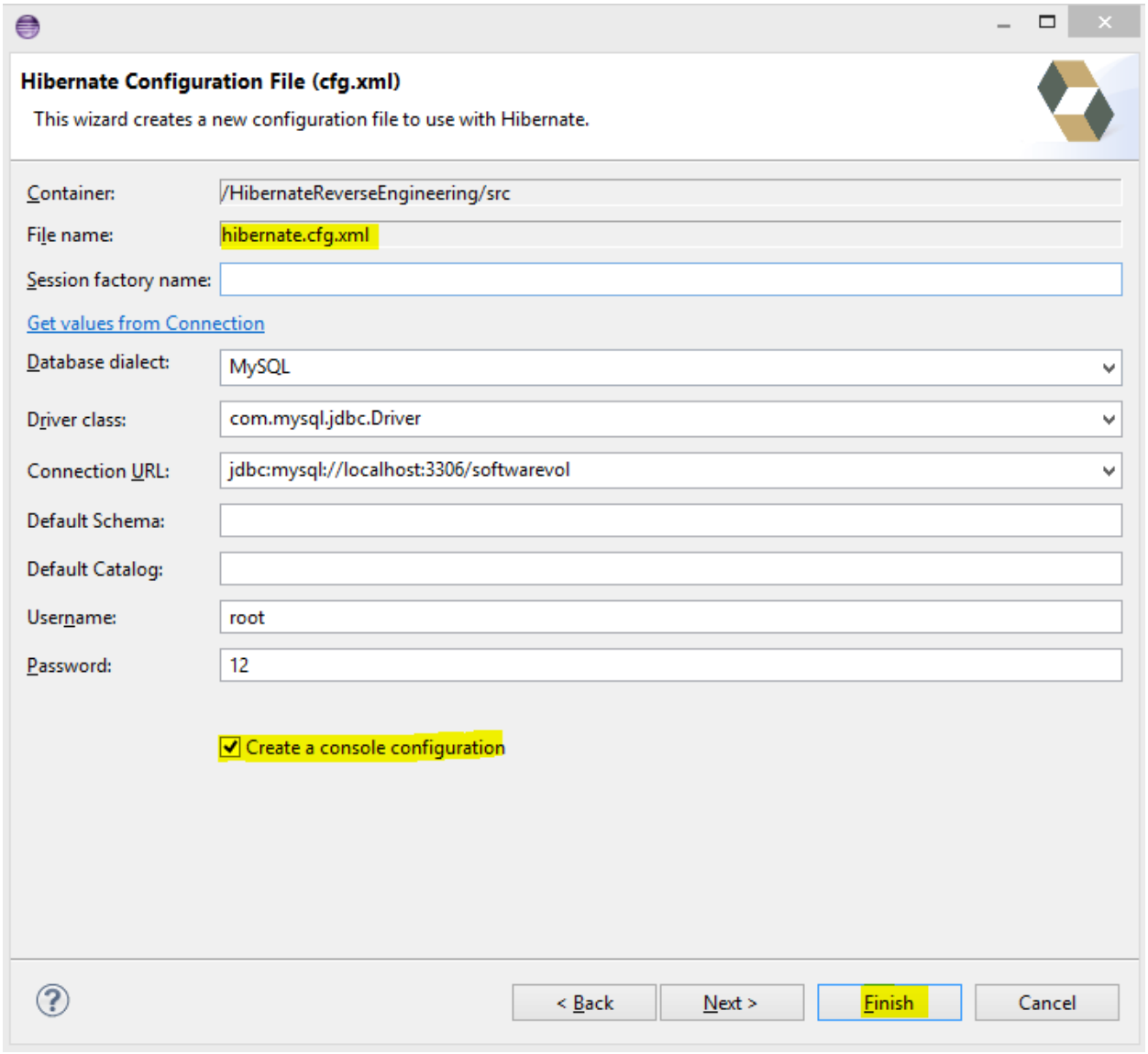




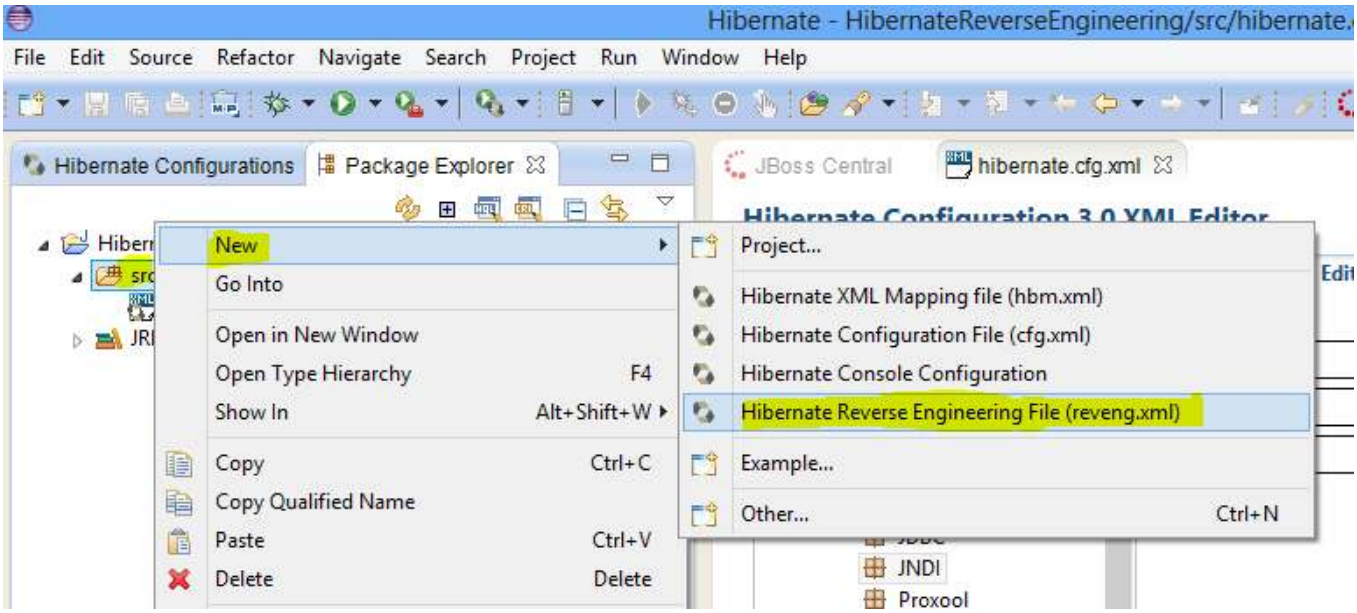
3. **Package Explorer** kısmına gelinir. Burada **src** klasörünün üzerine **sağ** tıklanıp New -> Hibernate Configuration File (.cfg.xml) seçilir



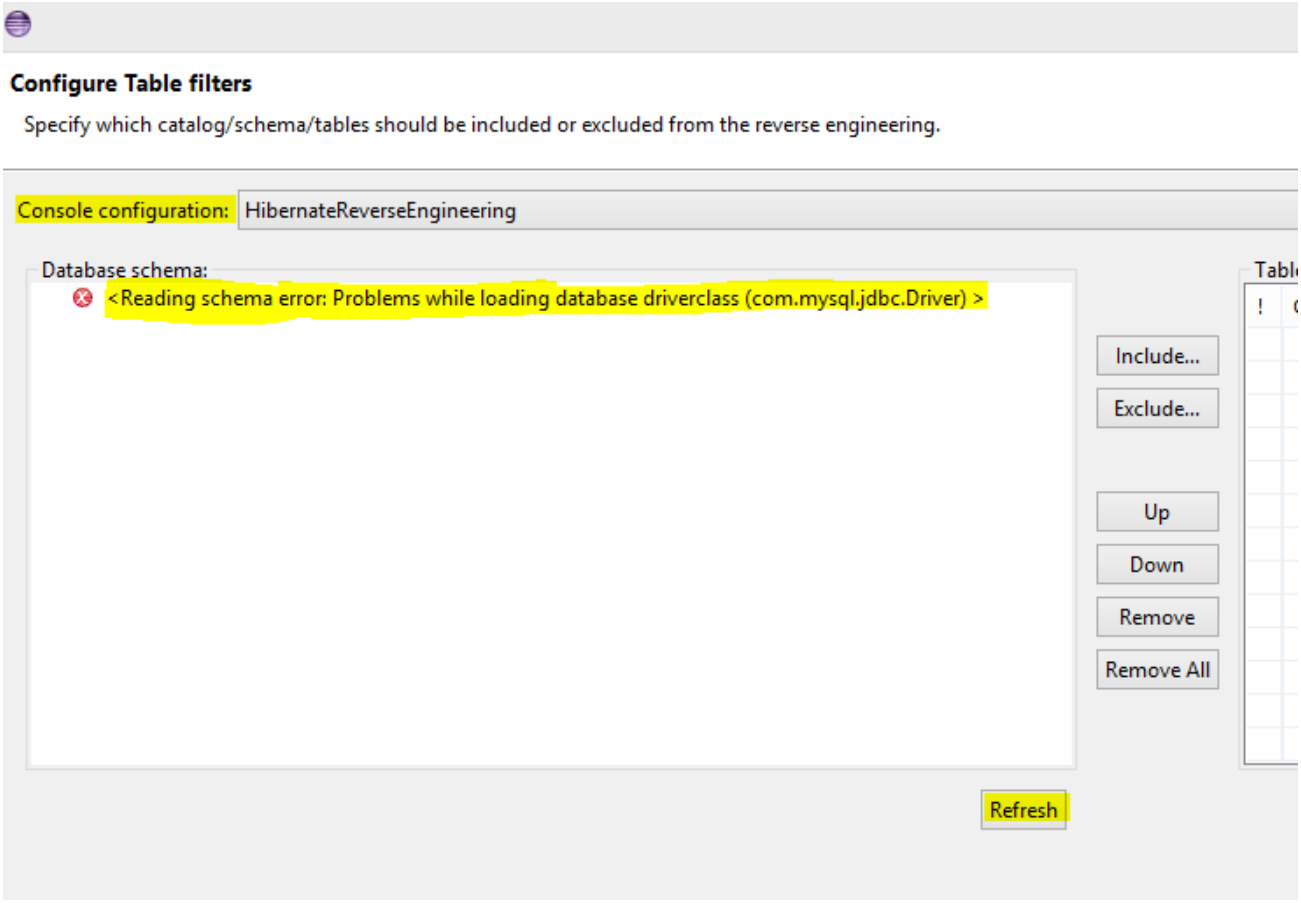
4. Gelen pencerede **next** denilir ve daha sonra açılan pencereden veritabanı ayarları yapılır. Bu örneğimizde veritabanı bağlantısı açıyoruz. Ayrıca bu ekranda **Create Console Configuration** seçeneğinin seçilmesine dikkat ediniz



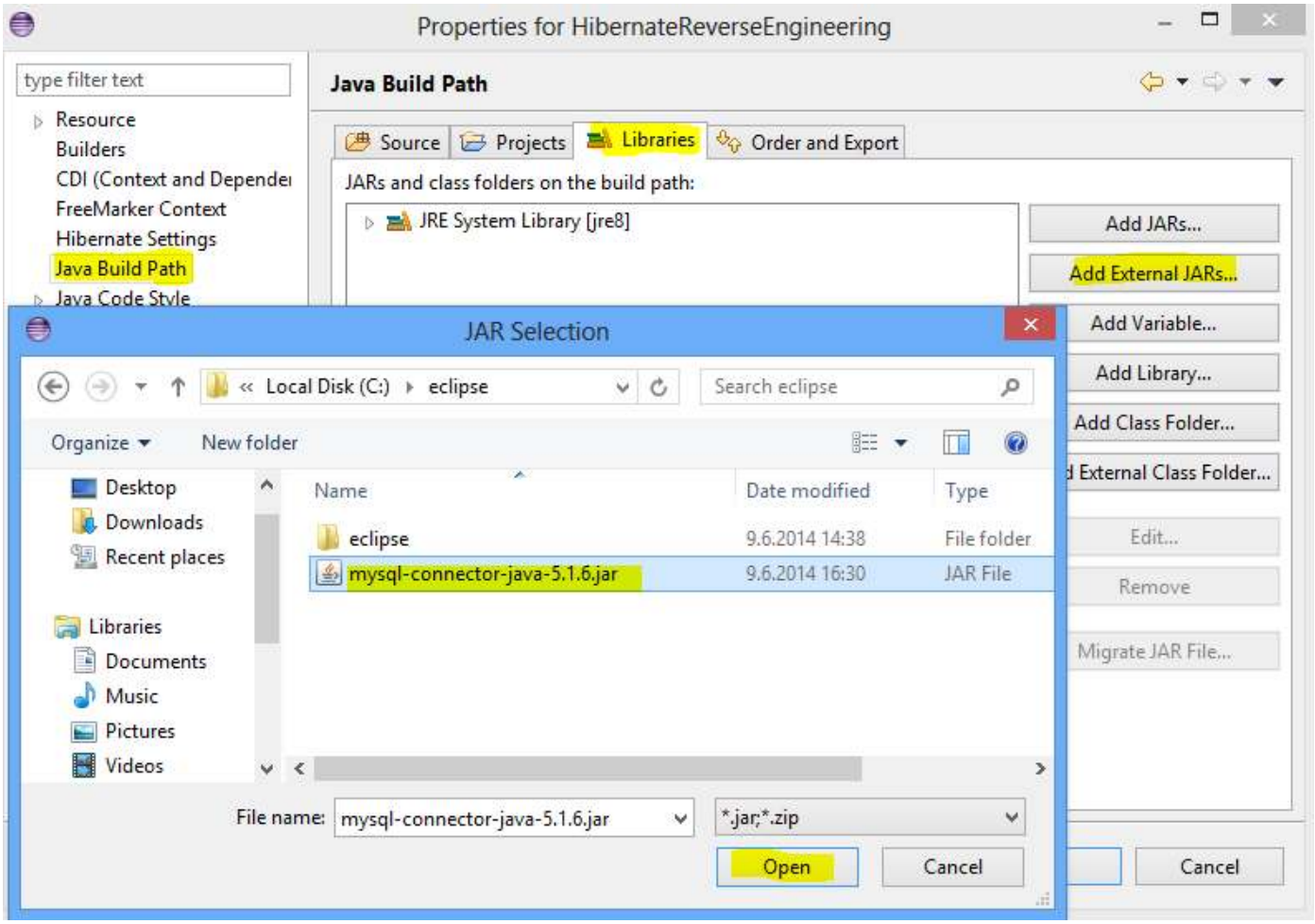
5. Tekrar src klasörünün üzerine gelip sağ tıkladıktan sonra New -> Hibernate Reverse Engineering File(reveng.xml) seçilir



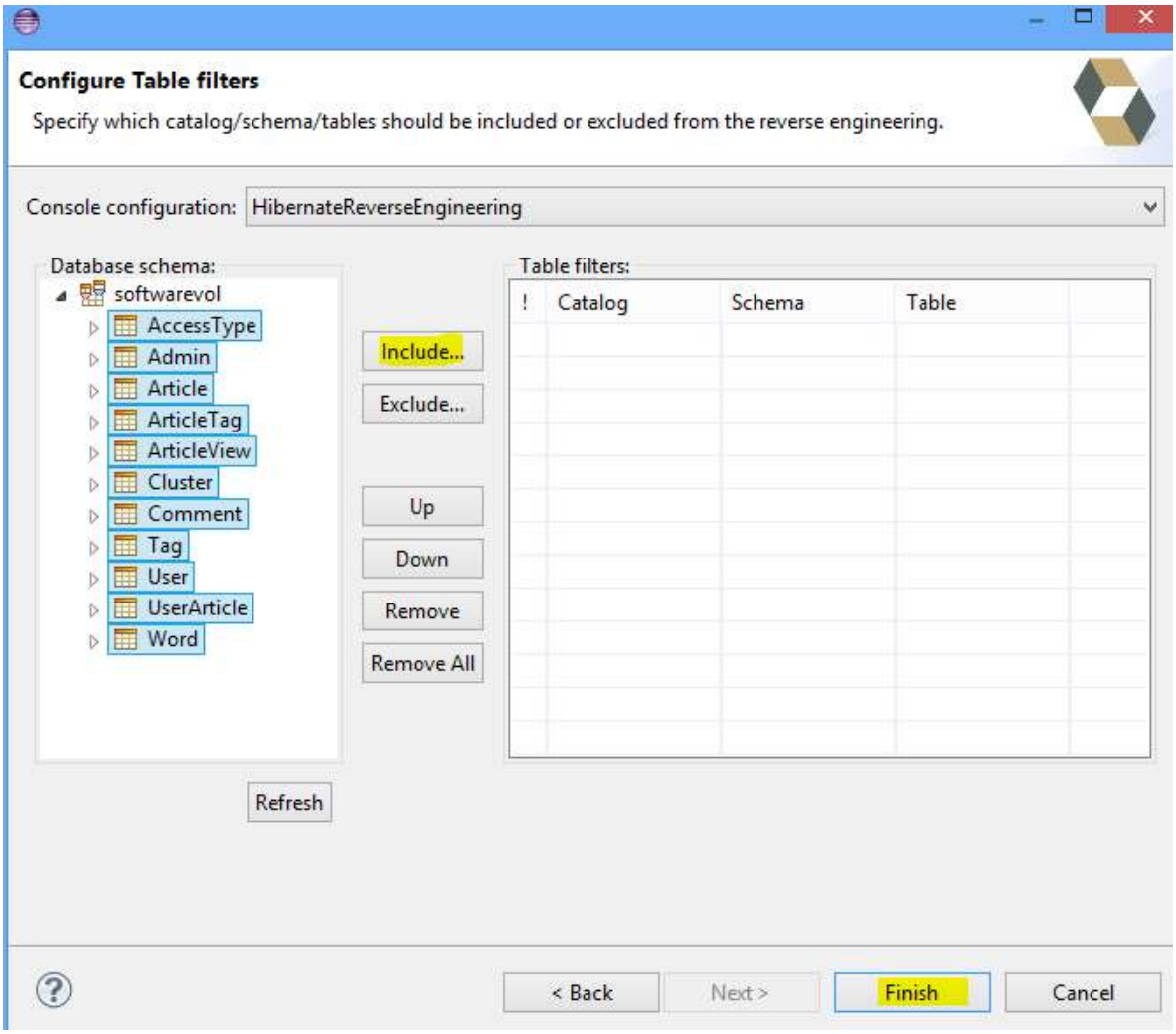
6. Gelen pencerede **next** denilir ve daha sonra açılan pencerede **Console** configuration kısmından 4. adımda yarattığımız konfigurasyon seçilir. Zaten tek bir tane seçenek gelecektir. Daha sonra **Refresh** butonuna tıklandığında **loading database driver class hatası** meydana gelir. Bu hatayı gidermek için **mysql.jar** dosyasını **external** library olarak yüklememiz gereklidir.



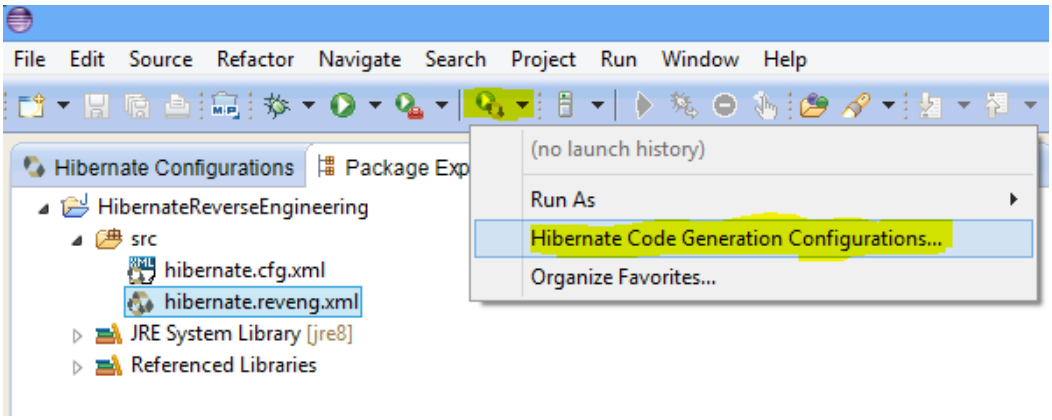
7. Gerekli jar dosyasını indirmek için **tıklayınız** (http://www.softwarevol.com/sources/docs/setup/hibernate/reverseEngineering/mysql-connector-java-5.1.6.jar) . İndirdiğimiz bu jar dosyasını File->Properties kısmına geldikten sonra aşağıdaki gibi **External Jars** library'e eklememiz gerekir



8. Daha sonra adım **6** daki işlemler tekrarlanır. **Refresh** butonuna tıkladığımızda veritabanındaki tabloları görebiliriz. Bu tablolardan reverse engineering yapılacak tabloyu seçtikten sonra **Include** butonuna tıklarız. Daha sonra **Finish** butonuna tıklanır



9. **Finish** dedikten sonra aşağıdaki şekilde olduğu gibi **Ok** işaretine tıklarız. Daha sonra **Hibernate Code Generation Configuration** kısmına geliriz.



10. **Hibernate Code Generation Configuration** ekran aşağıdaki gibidir. Bu kısımda yer alan **Main** tab'ındaki **Console configuration** kısmında **4.** adımda yarattığımız konfigürasyon seçilir. **Output** directory kısmından **src** klasörünün yolu seçilir. **Reverse engineer from JDBC Connection** seçeneği seçilmek **zorundadır,** eğer seçilmezse java sınıfları oluşturulmuz. **Package** kısmında **src** klasörüne **eklenen** packet yazılabilir. Biz bu örnekte boş bırakıyoruz. Paket yazıldığı zaman oluşturulacak Java sınıfları bu paket içerisine eklenir. **reveng.xml** dosyasının yolu **Setup** butonuna basılarak belirtilir. Burada **reveng.xml** dosyası **5.** adımda oluşan dosyadır. Daha sonra **Exporters** tab'ına gelinir. Bu kısımda ise üretilecek Java sınıflarının **annotasyon** özellikli veya **Java 5** sentakslı seçeneklerinden **biri** seçilir. Java 5 sentaksı demek Java sınıflarında tablo adı vs gibi belirtilecek ayarların hbm.xml dosyalarında yapılması anlamına gelmektedir. Exporters kısmında yer alan önemli seçenekler şunlardır:

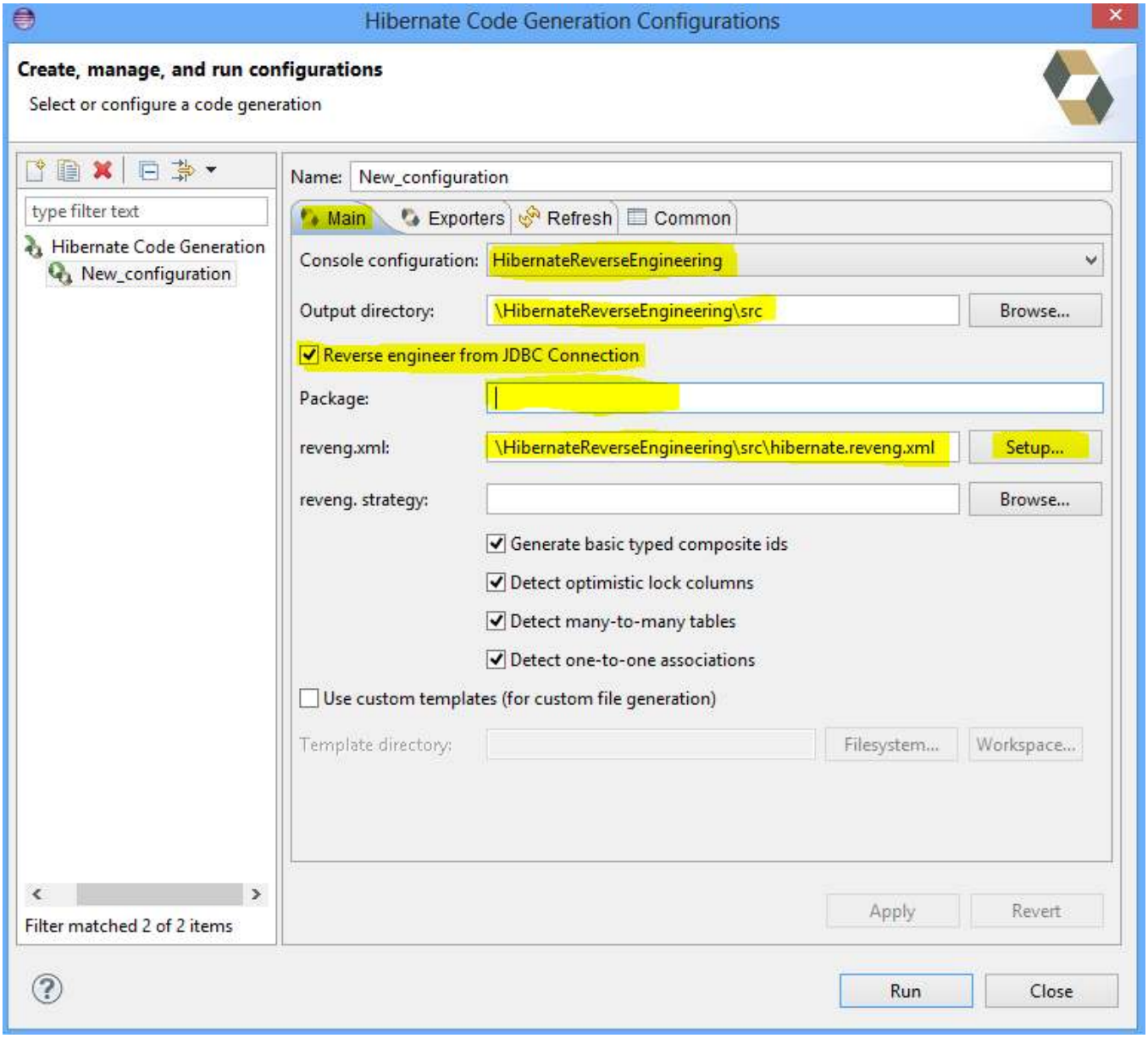
Domain code(.java): Veritabanındaki tabloların sınıf karşılıklarını temsil eder.

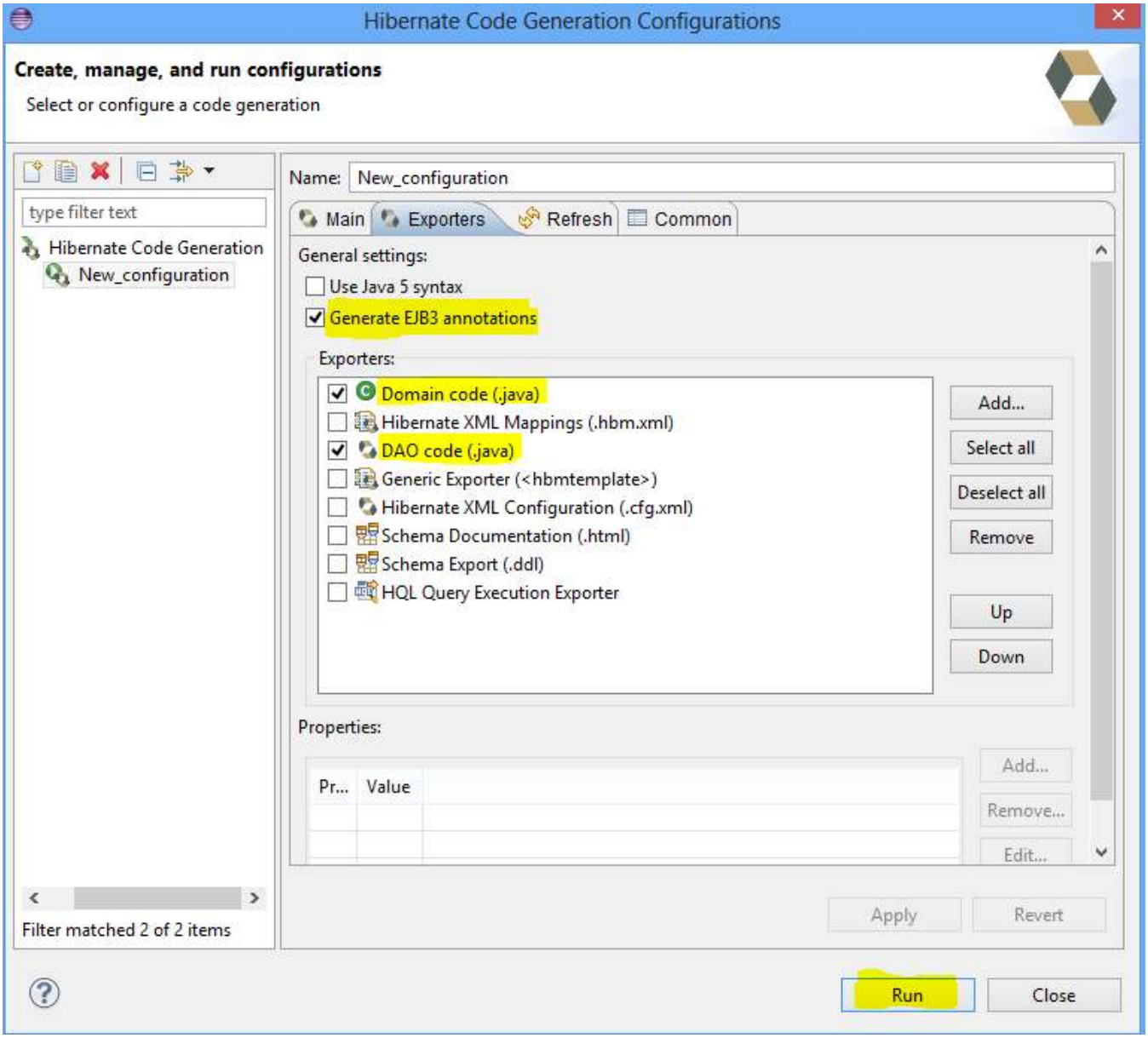
Hibernate XML Mapping(.hbm.xml): Annotasyon kullanılmadığı zaman Java sınıfları ile veritabanı sınıfları arasındaki ilişkiyi tanımlamak için gerekli ayarların yapıldığı dosyadır. Örneğin, Java sınıfındaki bir değişkenin veritabanında karşılığı olduğu sütun adına denk geldiğini ayarlamak gibi..

DAO code(.java): Bu sınıf içerisinde veritabanı işlemlerinin(insert, update, delete, find) yapılmasını sağlamak için kodlar bulunur.

Son olarak **Run** butonuna tıklanarak işlem tamamlanmış olur.

Not: Hibernate otomatik olarak bu sınıfları üretir.




Hibernate Code Generation Configurations
Create, manage, and run configurations
Select or configure a code generation
type filter text
Hibernate Code Generation
New_configuration
Filter matched 2 of 2 items
Name: New_configuration
Main
Exporters
Refresh
Common
General settings:
Use Java 5 syntax
Generate EJB3 annotations
Exporters:
Domain code (.java)
Hibernate XML Mappings (.hbm.xml)
DAO code (.java)
Generic Exporter (<hbmtemplate>)
Hibernate XML Configuration (.cfg.xml)
Schema Documentation (.html)
Schema Export (.ddl)
HQL Query Execution Exporter
Add...
Select all
Deselect all
Remove
Up
Down
Properties:
Pr...
Value
Add...
Remove...
Edit...
Apply
Revert
Run
Close