



Hibernate – crud

Lister les données stockées

ListAll

QueryVoitureDemo.java

```
...
try {
    // debut de transaction
    session.beginTransaction();

    // Lister les voitures
    List<Voiture> voitures=
        session
        .createQuery("from Voiture ")
        .getResultList();

    //Afficher les voitures
    System.out.println("1- All voitures : ");
    voitures.stream().forEach(System.out::println);
    System.out.println("\n");

    // commit de la transaction
    session.getTransaction().commit();

    System.out.println("Terminé !");
} finally {
    factory.close();
}
...
```



Markers Servers Properties Data Source Explorer Snippets Console

```
<terminated> getAlldemo [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/
juil. 29, 2020 11:45:59 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: select voiture0_.id as id1_0_, voiture0_.immatriculation as im
1- All voitures :
Voiture [id=1, modele=picasso, immat=CDF-548-13]
Voiture [id=2, modele=countach, immat=ZE-022-13]
Voiture [id=3, modele=kangoo, immat=Zg-047-84]
Voiture [id=4, modele=megane, immat=abc-123-75]
Voiture [id=5, modele=megane, immat=abc-123-75]

Terminé !
juil. 29, 2020 11:46:00 PM org.hibernate.engine.jdbc.connections.internal
INFO: HHH000100: Closing up connections pool [id=...]
...
```



Essayez les requêtes filtrées que nous avons vu ensemble

- Cf `QueryDemo.java`



Log monitorer les paramètres SQL

- Dans la sortie console on peut voir les requêtes exécutées mais il manque les détails des paramètres des requêtes.
- On va solutionner ce besoin via log4j

Pas à pas

- Télécharger log4j 1.2.17
- <https://downloads.apache.org/logging/log4j/1.2.17/log4j-1.2.17.zip>
- Ajoutez le au classpath de votre application
- Creez log4j.properties dans le répertoire "src"
- Copiez ce contenu à l'intérieur

```
# Root logger option
log4j.rootLogger=DEBUG, stdout

# Redirect log messages to console
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.out
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{yyyy-MM-dd HH:mm:ss} %-5p %c{1}:%L - %m%n

log4j.logger.org.hibernate=TRACE
```

important

Relancez votre application => vous allez voir beaucoup de logs

- Dans la console faites un clic droit et sélectionner **Find/Replace...**

Ou recherchez "binding parameter"

Ou encore : "extracted value"

cette recherche dépend de la version d'hibernate qu'on utilise

Find/Replace

Find:

Replace with:

Direction: ☒ Forward ☐ Backward

Scope: ☒ All ☐ Selected lines

Options:

- ☐ Case sensitive ☒ Wrap search
- ☐ Whole word ☐ Incremental
- ☐ Regular expressions

Find Replace/Find Replace All Close

Console

```
<terminated> QueryDemo [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/Contents/Home/bin/java (30 juil. 2020 à 00:20:01 - 00:20:04)
2020-07-30 00:20:03 TRACE JdbcCoordinatorImpl:325 - Registering last query statement [com.mysql.cj.jdbc.ClientPreparedStatement]
2020-07-30 00:20:03 TRACE Loader:2166 - Bound [1] parameters total
2020-07-30 00:20:04 TRACE ResourceRegistryStandardImpl:211 - Registering result set [com.mysql.cj.jdbc.result.ResultSetImpl]
2020-07-30 00:20:04 TRACE Loader:1031 - Processing result set
2020-07-30 00:20:04 DEBUG Loader:384 - Result set row: 0
2020-07-30 00:20:04 TRACE BasicExtractor:60 - extracted value ([id1_0_] : [BIGINT]) - [1]
2020-07-30 00:20:04 DEBUG Loader:1594 - Result row: EntityKey[com.spring.hibernate.entity.Voiture#1]
2020-07-30 00:20:04 TRACE Loader:1842 - Initializing object from ResultSet: [com.spring.hibernate.entity.Voiture#1]
2020-07-30 00:20:04 TRACE AbstractEntityPersister:3018 - Hydrating entity: [com.spring.hibernate.entity.Voiture#1]
2020-07-30 00:20:04 TRACE BasicExtractor:60 - extracted value ([immatric2_0_] : [VARCHAR]) - [CDF-548-13]
2020-07-30 00:20:04 TRACE BasicExtractor:60 - extracted value ([modele_n3_0_] : [VARCHAR]) - [picasso]
2020-07-30 00:20:04 DEBUG Loader:384 - Result set row: 1
2020-07-30 00:20:04 TRACE BasicExtractor:60 - extracted value ([id1_0_] : [BIGINT]) - [2]
2020-07-30 00:20:04 DEBUG Loader:1594 - Result row: EntityKey[com.spring.hibernate.entity.Voiture#2]
2020-07-30 00:20:04 TRACE Loader:1842 - Initializing object from ResultSet: [com.spring.hibernate.entity.Voiture#2]
2020-07-30 00:20:04 TRACE AbstractEntityPersister:3018 - Hydrating entity: [com.spring.hibernate.entity.Voiture#2]
```

Editer , mettre à jour les objets avec hibernate

```
Long id= 1L;

//lire ou récupérer une voiture avec son id:
//clé primaire
Voiture uneVoiture =
session.get(Voiture.class, id);

//modifier le modèle de la voiture
uneVoiture.setModele("Zoe");

// commit de la transaction
session.getTransaction().commit();
```

Simple edition

```
// nouvelle session et ouverture d'une transaction
Session session = factory.getCurrentSession();
session.beginTransaction();
```

```
Long id= 1L;
```

```
//lire ou récupérer une voiture avec son id: clé primaire
System.out.println("\nRecuperation d'une voiture avec l'id :
"+id);
```

```
Voiture uneVoiture = session.get(Voiture.class, id);
```

```
//modifier le modèle de la voiture
```

```
System.out.println("mise à jour de la voiture...");
uneVoiture.setModel("Zoe");
```

```
// commit de la transaction
```

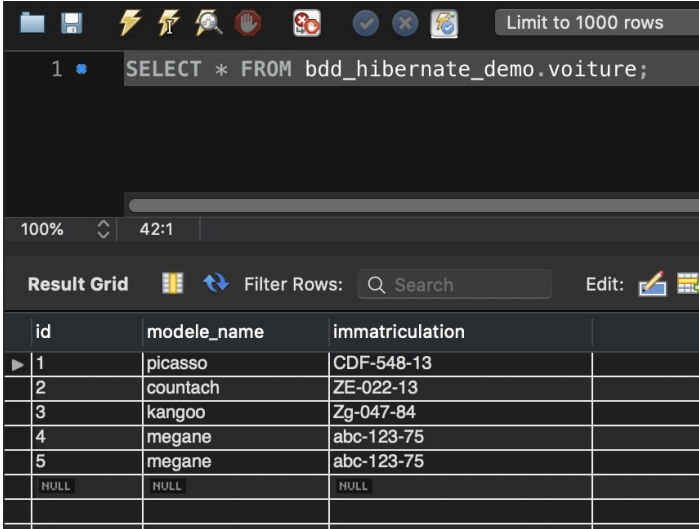
```
session.getTransaction().commit();
```

L'objet est mis à jour
uniquement en mémoire

C'est le commit qui va
Prendre en compte la
modification en bdd

Inutile d'expliquer avec un `session.save()` ou un `session.update()`

Avant l'exécution



Limit to 1000 rows

1 • `SELECT * FROM bdd_hibernate_demo.voiture;`

100% 42:1

Result Grid Filter Rows: Search Edit:

id	modele_name	immatriculation
1	picasso	CDF-548-13
2	countach	ZE-022-13
3	kangoo	Zg-047-84
4	megane	abc-123-75
5	megane	abc-123-75
NULL	NULL	NULL

Pendant l'exécution

Markers Servers Properties Data Source Explorer Snippets Console

<terminated> UpdateVoitureDemo [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/Contents/Home

INFO: HHH10001001: Connection properties: {password=****, user=padawan}

juil. 31, 2020 12:30:54 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerCon

INFO: HHH10001003: Autocommit mode: false

juil. 31, 2020 12:30:54 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerCon

INFO: HHH000115: Hibernate connection pool size: 1 (min=1)

juil. 31, 2020 12:30:54 AM org.hibernate.dialect.Dialect <init>

INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect

Recuperation d'une voiture avec l'id : 1

Hibernate: select voiture0_.id as id1_0_0_, voiture0_.immatriculation as immatric2_0_0_, v

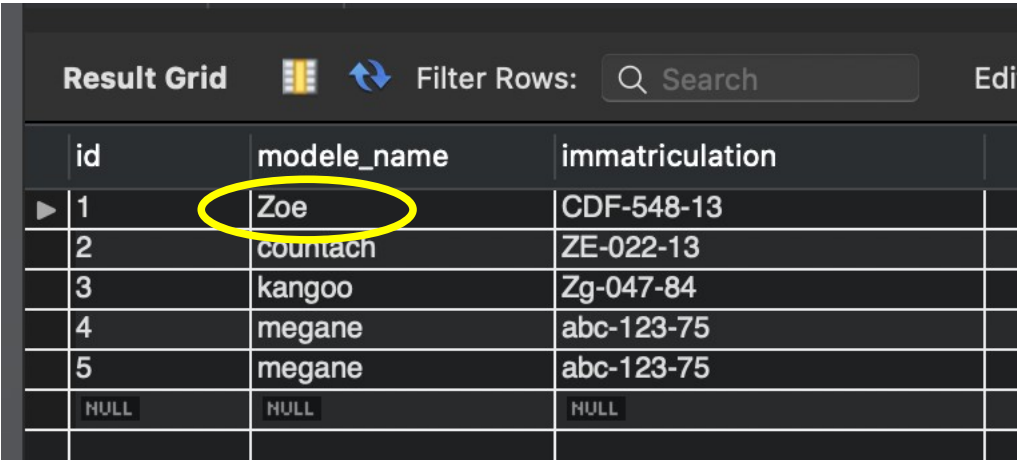
mise à jour de la voiture...

Terminé !

juil. 31, 2020 12:30:55 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerCon

INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/bdd_hibernate_

Après l'exécution



Result Grid Filter Rows: Search Edit:

id	modele_name	immatriculation
1	Zoe	CDF-548-13
2	countach	ZE-022-13
3	kangoo	Zg-047-84
4	megane	abc-123-75
5	megane	abc-123-75
NULL	NULL	NULL

Editer plusieurs objets

- On peut exécuter une requête de modification en sélectionnant plusieurs objets via une requête HQL (ici elle simple mais il est possible de rajouter par exemple un filtre `where` pour sélectionner certaines voitures).

```
//modifier toutes les voitures  
session  
    .createQuery("update Voiture set immatriculation='WWW-WWW'")  
    .executeUpdate();
```

On insère cette instruction dans que sorte de coquille

```
session
.createQuery("update Voiture set immatriculation='WWW-WWW'")
.executeUpdate();
```

```
// create session
Session session = factory.getCurrentSession();
// debut de transaction
session.beginTransaction();
```

```
//on écrit ou on lit dans la base
...
```

```
// commit de la transaction
session.getTransaction().commit();
```

On exécute cette classe

```
Markers Servers Properties Data Source Explorer Snippets Console
<terminated> EditAllVoitureDemo [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/Content
juil. 31, 2020 12:45:21 AM org.hibernate.engine.jdbc.connections.internal.DriverMan
INFO: HHH10001005: using driver [com.mysql.cj.jdbc.Driver] at URL [jdbc:mysql://loc
juil. 31, 2020 12:45:21 AM org.hibernate.engine.jdbc.connections.internal.DriverMan
INFO: HHH10001001: Connection properties: {password=****, user=padawan}
juil. 31, 2020 12:45:21 AM org.hibernate.engine.jdbc.connections.internal.DriverMan
INFO: HHH10001003: Autocommit mode: false
juil. 31, 2020 12:45:21 AM org.hibernate.engine.jdbc.connections.internal.DriverMan
INFO: HHH000115: Hibernate connection pool size: 1 (min=1)
juil. 31, 2020 12:45:21 AM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: update voiture set immatriculation='WWW-WWW'
Terminé !
juil. 31, 2020 12:45:22 AM org.hibernate.engine.jdbc.connections.internal.DriverMan
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/bdd_hib
```

Coté bdd cela donera

Good job !

Result Grid			
Filter Rows: Search			
	id	modele_name	immatriculation
▶	1	Zoe	WWW-WWW
	2	countach	WWW-WWW
	3	kangoo	WWW-WWW
	4	megane	WWW-WWW
	5	megane	WWW-WWW
	NULL	NULL	NULL