Hibernate Loglama ve İstatistik Bilgileri

harezmi bilişim çözümleri

Loglama ve İstatistik Bilgileri

- Hibernate'i çalışma sırasında **monitor** etmek ve **istatistik bilgileri toplamak** mümkündür
- Monitoring sırasında entity insert, delete, update işlemleri'nin sayısı, çalıştırılan sorgular, bu sorguların kaç defa çalıştırıldığı, çalışma süreleri, TX sayısı vb elde edilebilmektedir
- Default olarak bu **monitor özelliği kapalı**dır
  - **hibernate.generate_statistics** property tanımının true olarak set edilmesi gerekir
  - Yada SessionFactory.getStatistics() üzerinden enable edilebilir

Loglama ve İstatistik Bilgileri

- İstatistik bilgileri devreye alındığı vakit **Session sonunda** açılan JDBC bağlantı, oluşturulan ve çalıştırılan PreparedStatement, 2nd level cache etkileşim, flush sayılarını vb listeleyen bir log ifadesi yazdırılmaktadır

```
391316 nanoseconds spent acquiring 1 JDBC connections;
0 nanoseconds spent releasing 0 JDBC connections;
3870319 nanoseconds spent preparing 3 JDBC statements;
6025561 nanoseconds spent executing 3 JDBC statements;
0 nanoseconds spent executing 0 JDBC batches;
0 nanoseconds spent performing 0 L2C puts;
0 nanoseconds spent performing 0 L2C hits;
0 nanoseconds spent performing 0 L2C misses;
13584261 nanoseconds spent executing 1 flushes (flushing a total of 1 entities and 0 collections);
224563 nanoseconds spent executing 1 partial-flushes (flushing a total of 1 entities and 1 collections)
```

Log ve İstatistik Kabiliyetinin Programatik Aktivasyonu

JAVA Eğitimleri

- SessionFactory üzerinden Statistics nesnesine erişerek log ve istatistik kabiliyetini **dinamik olarak** da kontrol edebiliriz

```
Statistics stats = sf.getStatistics();

stats.setStatisticsEnabled(false);
```

- Bu işlem **Session metrikleri**nin oluşturulmasına etki etmemektedir
- Session metrikleri için **hibernate.generate_statistics** property tanımı **şart**tır

Detaylı İstatistik Bilgilerine Erişim

- Statistics üzerinden entity, collection, query, cache ile ilgili spesifik **detay bilgile**re de erişilebilir

```
String[] entityNames = stats.getEntityNames();

String[] roleNames = stats.getCollectionRoleNames();

String[] queries = stats.getQueries();

String[] regionNames = stats.getSecondLevelCacheRegionNames();
```

Detaylı İstatistik Bilgilerine Erişim

```
EntityStatistics entityStatistics =
stats.getEntityStatistics("com.javaegitimleri.petclinic.Owner");
```

Entity ile ilgili load, insert, delete, update, fetch, optimistic failure count gibi bilgiler elde edilebilir

```
CollectionStatistics collectionStatistics = stats
.getCollectionStatistics("com.javaegitimleri.petclinic.Owner.pets");
```

İlişki ile ilgili load, recreate, update, remove, fetch count gibi bilgiler elde edilebilir

Detaylı İstatistik Bilgilerine Erişim

```
QueryStatistics queryStatistics = stats
.getQueryStatistics("from Owner o where o.firstName like ?");
```

Query ile ilgili execution, max, min, avg, total zaman bilgilerine, cache hit, miss, put, execution count gibi bilgilerine erişilebilir

```
SecondLevelCacheStatistics cacheStatistics = stats
.getSecondLevelCacheStatistics("com.javaegitimleri.petclinic.Owner");
```

Second level cache region ile ilgili hit, miss, put count bilgilerine, elemanların disk ve memory deki sayılarına, region'ın içindeki entry'lere, region'ın büyüklük bilgisine erişilebilir

SQL İfadelerinin Çalışma Sürelerinin Loglanması

JAVA Eğitimleri

- **org.hibernate.stat** logger'ının düzeyini **DEBUG** yaptığımız takdirde çalıştırılan sorguların süresi de log'a yazılacaktır

```
<logger name="org.hibernate.stat">
    <level value="DEBUG"/>
</logger>
```

```
Hibernate:
    /*
from
    Owner */ select
        owner0_.id as id1_1_,
        owner0_.first_name as first_na2_1_,
        owner0_.last_name as last_nam3_1_
    from
        owner owner0_
DEBUG 10747 --- [           main] o.h.s.internal.ConcurrentStatisticsImpl  : HHH000117: HQL: from Owner, time:
4ms, rows: 1
```

SQL Input Parametrelerinin Loglanması

- **org.hibernate.type** logger düzeyi **TRACE** yapılarak çalıştırılan SQL ifadelerine geçilen parametreler de loglanabilir

```
<logger name="org.hibernate.type">
    <level value="TRACE"/>
</logger>
```

```
Hibernate:
    /* from
    Owner o where
    o.firstName like ? */ select
        owner0_.id as id1_1_,
        owner0_.first_name as first_na2_1_,
        owner0_.last_name as last_nam3_1_
    from
        owner owner0_
    where
        owner0_.first_name like ?
2018-12-21 10:24:33.802 TRACE 11174 --- [           main] o.h.type.descriptor.sql.BasicBinder      : binding parameter [1]
as [VARCHAR] - [K%]
```

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com