Hibernate N+1 Queries Problem

Mansoor Ali
Jan 13 · 7 min read

Hibernate is a very popular ORM framework in Java ecosystem. But being a popular framework doesn't mean that it is free from all predicaments. ORMs are developer friendly and many developers use them to quickly bootstrap their applications and perform database operations without writing database queries — but it can lead to many problems if the developers are not familiar with the underlying functionality of the framework. The n+1 queries problem is one such problem.

N+1 Queries Problem

If you have been using Hibernate(or any other ORM for that matter), chances are you have faced the infamous **n+1 queries problem** one time or another.
The **n+1** queries problem occurs while fetching lazy loaded **One-to-Many** parent-child relationships;
1 query to fetch **n** entities — parent
n queries(1 for each entity) to fetch the lazy loaded child entities

Consider the following example

Environment: Spring Boot, JPA(Hibernate), H2 database.
I have initialized the database with 5 authors and 25 books; and assigned 5 books to each author.

```
@Entity
public class Author {
```

```
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
private String firstName;
private String lastName;
private String address;

@OneToMany(fetch=FetchType.LAZY, mappedBy="author")
private List<Book> books;

}

. . .

@Entity
public class Book {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
private String title;
private String isbn;

@ManyToOne
private Author author;

}

. . .

public interface BookDataService extends JpaRepository<Book, Long>{

}

. . .

public interface AuthorDataService extends JpaRepository<Book, Long>
{

}

. . .

List<Author> authors = authorDataService.findAll();
for (Author author : authors) {
    System.out.printf("Author: %s %s has %d books.%n",
        author.getFirstName(), author.getLastName(),
        author.getBooks().size());
}
```

The above piece of code prints the full name of the author and the number of books associated with each author.

Logs:

```
13-01-2019 20:22:53.182 [restartedMain] INFO
org.hibernate.hql.internal.QueryTranslatorFactoryInitiator.initiateS
ervice - HHH000397: Using ASTQueryTranslatorFactory
Hibernate:
    select
        author0_.id as id1_0_,
        author0_.address as address2_0_,
        author0_.first_name as first_na3_0_,
        author0_.last_name as last_nam4_0_
    from
        author author0_
Hibernate:
    select
        books0_.author_id as author_i4_1_0_,
        books0_.id as id1_1_0_,
        books0_.id as id1_1_1_,
        books0_.author_id as author_i4_1_1_,
        books0_.isbn as isbn2_1_1_,
        books0_.title as title3_1_1_
    from
        book books0_
    where
        books0_.author_id=?
Author: Lucille Ordelt has 5 books.
Hibernate:
    select
        books0_.author_id as author_i4_1_0_,
        books0_.id as id1_1_0_,
        books0_.id as id1_1_1_,
        books0_.author_id as author_i4_1_1_,
        books0_.isbn as isbn2_1_1_,
        books0_.title as title3_1_1_
    from
        book books0_
    where
        books0_.author_id=?
Author: Slade Gerwood has 5 books.
Hibernate:
    select
        books0_.author_id as author_i4_1_0_,
        books0_.id as id1_1_0_,
        books0_.id as id1_1_1_,
        books0_.author_id as author_i4_1_1_,
        books0_.isbn as isbn2_1_1_,
        books0_.title as title3_1_1_
    from
        book books0_
    where
        books0_.author_id=?
Author: Wallis Croall has 5 books.
Hibernate:
    select
        books0_.author_id as author_i4_1_0_,
```

```
            books0_.id as id1_1_0_,
            books0_.id as id1_1_1_,
            books0_.author_id as author_i4_1_1_,
            books0_.isbn as isbn2_1_1_,
            books0_.title as title3_1_1_
        from
            book books0_
        where
            books0_.author_id=?
    Author: Alena Hall has 5 books.
    Hibernate:
        select
            books0_.author_id as author_i4_1_0_,
            books0_.id as id1_1_0_,
            books0_.id as id1_1_1_,
            books0_.author_id as author_i4_1_1_,
            books0_.isbn as isbn2_1_1_,
            books0_.title as title3_1_1_
        from
            book books0_
        where
            books0_.author_id=?
    Author: Marcello Szymanski has 5 books.
    13-01-2019 20:22:53.197 [restartedMain] INFO
    org.hibernate.engine.internal.StatisticalLoggingSessionEventListener
    .end - Session Metrics {
        32411 nanoseconds spent acquiring 1 JDBC connections;
        0 nanoseconds spent releasing 0 JDBC connections;
        811912 nanoseconds spent preparing 6 JDBC statements;
        993417 nanoseconds spent executing 6 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;
        928594 nanoseconds spent executing 1 flushes (flushing a total
    of 30 entities and 5 collections);
        9723 nanoseconds spent executing 1 partial-flushes (flushing a
    total of 0 entities and 0 collections)
    }
```

As you can see 1 query is executed to get all authors and then a separate query is executed to get the books for each author — resulting in 6 queries being executed. Now consider the impact of this problem in a real world application where database doesn't essentially exist on the same server as your application and has thousands of records.

This is the default behavior of Hibernate and in my opinion it is fine as long as you have to fetch the books of only a single author.

Fetch Mode

Hibernate provides an annotation `@Fetch(...)` which can be used to specify how the associated collection is fetched.

FetchMode.SUBSELECT

"use a subselect query to load the additional collections" — SUBSELECT

Using `@Fetch(FetchMode.SUBSELECT)` will issue just 1 additional query to fetch the books.

Modify the class Author.java as follows:

```
. . .

@Entity
public class Author {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
private String firstName;
private String lastName;
private String address;

@OneToMany(fetch=FetchType.LAZY, mappedBy="author")
@Fetch(FetchMode.SUBSELECT)
private List<Book> books;

}

. . .
```

And take a look at the logs:

```
13-01-2019 21:45:55.231 [restartedMain] INFO
org.hibernate.hql.internal.QueryTranslatorFactoryInitiator.initiateS
ervice - HHH000397: Using ASTQueryTranslatorFactory
Hibernate:
    select
        author0_.id as id1_0_,
        author0_.address as address2_0_,
        author0_.first_name as first_na3_0_,
        author0_.last_name as last_nam4_0_
    from
        author author0_
Hibernate:
    select
        books0_.author_id as author_i4_1_1_,
        books0_.id as id1_1_1_,
```

```
            books0_.id as id1_1_0_,
            books0_.author_id as author_i4_1_0_,
            books0_.isbn as isbn2_1_0_,
            books0_.title as title3_1_0_
        from
            book books0_
        where
            books0_.author_id in (
                select
                    author0_.id
                from
                    author author0_
            )
Author: Lucille Ordelt has 5 books.
Author: Slade Gerwood has 5 books.
Author: Wallis Croall has 5 books.
Author: Alena Hall has 5 books.
Author: Marcello Szymanski has 5 books.
13-01-2019 21:45:55.262 [restartedMain] INFO
org.hibernate.engine.internal.StatisticalLoggingSessionEventListener
.end - Session Metrics {
    43756 nanoseconds spent acquiring 1 JDBC connections;
    0 nanoseconds spent releasing 0 JDBC connections;
    2011683 nanoseconds spent preparing 2 JDBC statements;
    1827476 nanoseconds spent executing 2 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;
    771397 nanoseconds spent executing 1 flushes (flushing a total
of 30 entities and 5 collections);
    9724 nanoseconds spent executing 1 partial-flushes (flushing a
total of 0 entities and 0 collections)
}
```

We can see that only 2 queries have been executed; one to fetch the authors, and the other to fetch the books associated with those authors.

FetchMode.JOIN

According to Hibernate Docs:

“use an outer join to load the related entities, collections or joins” — JOIN

Using `@Fetch(FetchMode.JOIN)` should essentially fetch the associated collection using an outer join, but in my experience it doesn’t work on collections.

JOIN FETCH

What if we want to fetch all authors and their books in a single query?

If I were to write an SQL query for this purpose I would take a join of these tables.

Let’s do that in a developer friendly ORM way by using JPA’s `@Query` annotation.

Modify `AuthorDataService.java` as follows:

```
public interface AuthorDataService extends JpaRepository<Author,
Long>{

@Query("select a from Author a join fetch a.books")
List<Author> findAll();

}
```

Let's take a look at the logs now:

```
Hibernate:
    select
        author0_.id as id1_0_0_,
        books1_.id as id1_1_1_,
        author0_.address as address2_0_0_,
        author0_.first_name as first_na3_0_0_,
        author0_.last_name as last_nam4_0_0_,
        books1_.author_id as author_i4_1_1_,
        books1_.isbn as isbn2_1_1_,
        books1_.title as title3_1_1_,
        books1_.author_id as author_i4_1_0__,
        books1_.id as id1_1_0__
    from
        author author0_
    inner join
        book books1_
            on author0_.id=books1_.author_id

Author: Lucille Ordelt has 5 books.
Author: Lucille Ordelt has 5 books.
Author: Lucille Ordelt has 5 books.
Author: Lucille Ordelt has 5 books.
Author: Lucille Ordelt has 5 books.
Author: Slade Gerwood has 5 books.
Author: Slade Gerwood has 5 books.
Author: Slade Gerwood has 5 books.
Author: Slade Gerwood has 5 books.
Author: Slade Gerwood has 5 books.
Author: Wallis Croall has 5 books.
Author: Wallis Croall has 5 books.
Author: Wallis Croall has 5 books.
Author: Wallis Croall has 5 books.
Author: Wallis Croall has 5 books.
Author: Alena Hall has 5 books.
Author: Alena Hall has 5 books.
Author: Alena Hall has 5 books.
Author: Alena Hall has 5 books.
Author: Alena Hall has 5 books.
Author: Marcello Szymanski has 5 books.
```

```
Author: Marcello Szymanski has 5 books.
Author: Marcello Szymanski has 5 books.
Author: Marcello Szymanski has 5 books.
Author: Marcello Szymanski has 5 books.

13-01-2019 22:19:06.954 [restartedMain] INFO
org.hibernate.engine.internal.StatisticalLoggingSessionEventListener
.end - Session Metrics {
    37273 nanoseconds spent acquiring 1 JDBC connections;
    0 nanoseconds spent releasing 0 JDBC connections;
    426213 nanoseconds spent preparing 1 JDBC statements;
    324117 nanoseconds spent executing 1 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;
    616901 nanoseconds spent executing 1 flushes (flushing a total
of 30 entities and 5 collections);
    9723 nanoseconds spent executing 1 partial-flushes (flushing a
total of 0 entities and 0 collections)
}
```

Problem solved! … almost — Using **JOIN FETCH** solves the problem of multiple queries by using a join but it returns duplicate records for Author.
But that is what a join is supposed to do.

```
select * from author author0_
inner join book books1_
on author0_.id=books1_.author_id
```

Executing the above query gives the following result in which each author's data is duplicated with their books.

ID	ADDRESS	FIRST_NAME	LAST_NAME	ID	ISBN	TITLE	AUTHOR_ID
1	73 Schurz Center	Lucille	Ordelt	1	960988384-2	My Boyfriends' Dogs	1
1	73 Schurz Center	Lucille	Ordelt	2	599825640-9	Lodger: A Story of the London Fog, The	1
1	73 Schurz Center	Lucille	Ordelt	3	763517577-7	The Youngest Profession	1
1	73 Schurz Center	Lucille	Ordelt	4	849507811-2	Happiness	1
1	73 Schurz Center	Lucille	Ordelt	5	782690657-1	King of the Children (Hai zi wang)	1
2	57 Nobel Avenue	Slade	Gerwood	6	869400285-X	Phenomenon	2
2	57 Nobel Avenue	Slade	Gerwood	7	831317975-9	Plough and the Stars, The	2
2	57 Nobel Avenue	Slade	Gerwood	8	352249130-0	For Love or Country: The Arturo Sandoval Story	2
2	57 Nobel Avenue	Slade	Gerwood	9	192917399-7	Man Who Loved Cat Dancing, The	2
2	57 Nobel Avenue	Slade	Gerwood	10	680785263-0	Kelly's Heroes	2
3	11 Transport Center	Wallis	Croall	11	599116877-6	Nice Guys Sleep Alone	3
3	11 Transport Center	Wallis	Croall	12	126338738-1	Nickelodeon	3
3	11 Transport Center	Wallis	Croall	13	682351107-9	Coming Down the Mountain	3
3	11 Transport Center	Wallis	Croall	14	384695155-2	Air Bud: Golden Receiver	3
3	11 Transport Center	Wallis	Croall	15	823113697-5	The Prince	3
4	1185 Graceland Pass	Alena	Hall	16	921384844-7	Tony Rome	4

4	1185 Graceland Pass	Alena	Hall	17	795954377-1	Cocoon	4
4	1185 Graceland Pass	Alena	Hall	18	887233386-5	Fail-Safe	4
4	1185 Graceland Pass	Alena	Hall	19	473712639-X	White Water Summer	4

The solution to duplicate records problem is adding `DISTINCT` to the query.

```
public interface AuthorDataService extends JpaRepository<Author,
Long>{

@Query("select distinct a from Author a join fetch a.books")
List<Author> findAll();

}
```

Logs:

```
Hibernate:
    select
        distinct author0_.id as id1_0_0_,
        books1_.id as id1_1_1_,
        author0_.address as address2_0_0_,
        author0_.first_name as first_na3_0_0_,
        author0_.last_name as last_nam4_0_0_,
        books1_.author_id as author_i4_1_1_,
        books1_.isbn as isbn2_1_1_,
        books1_.title as title3_1_1_,
        books1_.author_id as author_i4_1_0__,
        books1_.id as id1_1_0__
    from
        author author0_
    inner join
        book books1_
            on author0_.id=books1_.author_id
Author: Slade Gerwood has 5 books.
Author: Lucille Ordelt has 5 books.
Author: Alena Hall has 5 books.
Author: Wallis Croall has 5 books.
Author: Marcello Szymanski has 5 books.
13-01-2019 22:20:00.730 [restartedMain] INFO
org.hibernate.engine.internal.StatisticalLoggingSessionEventListener
.end - Session Metrics {
    38353 nanoseconds spent acquiring 1 JDBC connections;
    0 nanoseconds spent releasing 0 JDBC connections;
    494818 nanoseconds spent preparing 1 JDBC statements;
    979372 nanoseconds spent executing 1 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;
    681185 nanoseconds spent executing 1 flushes (flushing a total
of 30 entities and 5 collections);
```

```
    10264 nanoseconds spent executing 1 partial-flushes (flushing a
total of 0 entities and 0 collections)
}
```

Note: All of these operations can also be done using criteria query.

Java Hibernate Jpa