New *getResultStream()* method

JPA 2.2 introduced several new features, and one of them is the new *getResultStream()* method. This method is now part of the *Query* interface. It returns the result of your query as a *Stream*. The goal of this method is to provide an efficient way to move through a result set. In the best case, it allows you to scroll through the result set instead of fetching all records at once.

```
Stream<Author> authors = em.createQuery(
        "SELECT a FROM Author a",
        Author.class).getResultStream();
```

One method with different implementations

The implementation of the *getResultStream()* method depends on the persistence provider, and you should check the documentation and code before you use it.

The default implementation, provided by the *Query* interface, just calls the *getResultList()* method to retrieve the result set as a *List* and calls the *stream()* method to transform it into a *Stream*. This approach doesn't provide any benefits compared to the *getResultList()* method available in JPA 2.1.

But it's to be expected that most JPA implementations provide their own implementation of the *getResultStream()* method. Hibernate, for example, introduced its *stream()* method in version 5.2 and I would be surprised if they don't reuse it for JPA's *getResultStream()* method.

Don't do this after retrieving the result set

Filter the elements in the Stream

The *filter* method allows you to select the elements in the *Stream* that fulfill certain criteria. But databases are optimized for this use case and can do that a lot faster. So please, don't implement any

additional filter when you process your *Stream*. You should better use JPQL or the Criteria API to specify the filter criteria in your query. And if that's not possible, you can still use a native SQL query.

Limit the number of elements in the Stream

You should only select the database records that you want to process in your application. So, if you already know that you just need to a certain number of records, you should always limit the size of the result set within the query.

You can do that with the *setFirstResult* and the *setMaxResult* methods on the *Query* and *TypedQuery* interface.

```
Stream<Author> authors = em.createQuery(
        "SELECT a FROM Author a ORDER BY a.id ASC",
        Author.class)
                .setMaxResults(5)
                .setFirstResult(10)
                .getResultStream();
```

Sort the elements in the Stream

You could use the sorted method provided by the *Stream* interface to sort the elements of your *Stream*. But that's another operation which the database can do much faster than your Java code. You just need to add an *ORDER BY* clause to your query and the database returns the result set in your preferred order.

In JPQL, you can do that with a similar syntax as you probably know from SQL.

```
Stream<Author> authors = em.createQuery(
        "SELECT a FROM Author a ORDER BY a.id ASC",
        Author.class).getResultStream();
```