JPA Query Structure (JPQL / Criteria)

The syntax of the Java Persistence Query Language (JPQL) is very similar to the syntax of SQL. Having an SQL-like syntax in JPA queries is an important advantage because SQL is a very powerful query language and many developers are already familiar with it.

The main difference between SQL and JPQL is that SQL works with relational database tables, records and fields, whereas JPQL works with Java classes and objects. For example, a JPQL query can retrieve and return entity objects rather than just field values from database tables, as with SQL. That makes JPQL more object oriented friendly and easier to use in Java.

JPQL Query Structure

As with SQL, a JPQL SELECT query also consists of up to 6 clauses in the following format:

```
SELECT ... FROM ...
[WHERE ...]
[GROUP BY ... [HAVING ...]]
[ORDER BY ...]
```

The first two clauses, SELECT and FROM are required in every retrieval query (update and delete queries have a slightly different form). The other JPQL clauses, WHERE, GROUP BY, HAVING and ORDER BY are optional.

The structure of JPQL DELETE and UPDATE queries is simpler:

```
DELETE FROM ... [WHERE ...]

UPDATE ... SET ... [WHERE ...]
```

Besides a few exceptions, JPQL is case insensitive. JPQL keywords, for example, can appear in queries either in upper case (e.g. SELECT) or in lower case (e.g. select). The few exceptions in which JPQL is case sensitive include mainly Java source elements such as names of entity classes and persistent fields, which are case sensitive. In addition, string literals are also case sensitive (e.g. "ORM" and "orm" are different values).

A Minimal JPQL Query

The following query retrieves all the `Country` objects in the database:

```
SELECT c FROM Country AS c
```

Because SELECT and FROM are mandatory, this demonstrates a minimal JPQL query.

The FROM clause declares one or more query variables (also known as identification variables). Query variables are similar to loop variables in programming languages. Each query variable represents iteration over objects in the database. A query variable that is bound to an entity class is referred to as a range variable. Range variables define iteration over all the database objects of a binding entity class and its descendant classes. In the query above, `c` is a range variable that is bound to the `Country` entity class and defines iteration over all the `Country` objects in the database.

The SELECT clause defines the query results. The query above simply returns all the `Country` objects from the iteration of the c range variable, which in this case is actually all the `Country` objects in the database.

Organization of this Section

This section contains the following pages:

Detailed explanations on how to set criteria query clauses are provided as follows:

- Criteria SELECT (`select`, `distinct`, `multiselect`, `array`, `tuple`, `construct`).
- Criteria FROM (`from`, `join`, `fetch`).
- Criteria WHERE (`where`).
- Criteria GROUP BY / HAVING (`groupBy`, `having`, `count`, `sum`, `avg`, `min`, `max`, ...).
- Criteria ORDER BY (`orderBy`, `Order`, `asc`, `desc`).

< Setting & Tuning ^ JPA Queries JPQL SELECT >