---

## 3.1 SQL debugging

If a type creation is unsuccessful, Oracle usually only provides a warning "created with compilation errors". To obtain more information about the reasons for the compilation errors, type

```
show error
```

The error message contains information about the line number and column in which the error occurred and about the type of error.

To recreate a type after fixing the compilation errors, use

```
CREATE OR REPLACE ....
```

instead of DROP TYPE ... and CREATE. But this will only work if the type does not yet have dependent tables. A type with dependent tables can only be ALTERed, not REPLACEd.

---

## 3.2 Member Methods

An object type can have several methods. These can be member methods (instance methods) or static methods (class methods). An object type with methods must have a separate type body which contains the code for the methods. Methods can either be functions (which return a value) or procedures (which do not have a return statement). Methods can be written in PL/SQL or almost any other programming language. To use another language apart from PL/SQL the language name must be stated in the type body (e.g., MEMBER FUNCTION ... RETURN ... AS LANGUAGE JAVA ...). The examples in this tutorial all use PL/SQL.

The following example shows a type job2, which contains the same columns as the type job (from last week's exercises), but also declares a method "evaluate_qualification".

```
CREATE OR REPLACE TYPE job2 AS OBJECT (
    jobtitle varchar(20),
    job_id int,
    salary_amount int,
    years_of_experience int,
MEMBER FUNCTION evaluate_qualification RETURN STRING
);
```

The details of "evaluate_qualification" are defined in the type body. It prints "too bad" for all jobs that have less than 2 years of experience, "OK" for 2 years of experience and "great" for more than two years of experience.

```
CREATE OR REPLACE TYPE BODY job2 AS
MEMBER FUNCTION evaluate_qualification RETURN STRING IS
BEGIN
    IF self.years_of_experience < 2 THEN
    RETURN 'too bad';
    ELSIF self.years_of_experience = 2 THEN
    RETURN 'OK';
    ELSE
    RETURN 'great!';
    END IF;
```

```
END evaluate_qualification;
END;
```

PL/SQL is fairly easy to use. If a method needs any variables, they must be declared between IS and BEGIN. Datatypes are STRING, REAL and NUMBER (not INT or VARCHAR). The word "self" refers to the current object, with which the method is called.

If j is a table alias for a table of type job2, then this method can be invoked with

```
j.evaluate_qualification()
```

The brackets are always required even if the method has no arguments. SQL select statements can only invoke methods that do not change the data in the tables. Therefore, only functions, which return values, but not procedures, which change data, can be invoked in queries.

**Exercises**:

- 20. Add a second method to the type declaration of job2. This method is called salary_fraction. It takes a real number N as an argument and returns the salary divided by N. For example, if N is 12, it returns the monthly salary. The argument "N REAL" must be included in brackets between the method name and RETURN (both in the type declaration and in the type body).
- 21. Create an object table for job2 and insert three rows. Write an SQL statement that selects evaluate_qualification and monthly salaries for all rows of the table. Write an SQL statement that selects all columns for rows which have an 'OK' value for evaluate_qualification and a half-yearly salary of more than 10000.

---

## 3.3 Adding methods to existing types

It is quite common in object-relational databases that object types need to be modified because the data model may have changed.

The job type from last week's exercises can be altered:

```
ALTER TYPE job
ADD MEMBER FUNCTION evaluate_qualification RETURN STRING
CASCADE;
```

"CASCADE" makes certain that the alterations apply to existing dependent object tables. It is necessary to logout of SQLPLUS and login again after altering types with dependent tables.

The type body for "job" can then be defined in the same manner as for "job2".

**Exercises:**

- 22. Alter the person type from last week's exercises so that it has a print_name method. This method prints the lastname, a comma, a space, the firstname, a space and then the middle initial. The PL/SQL string concatenation operator is ||.
- 23. Alter the person type from last week's exercises so that it has a method that counts how many categories of phone numbers a person has. The categories are "home phone", "business phone", "mobile phone". Thus the result must be between 0 and 3. (Note: NULL can be used within PL/SQL in the same way as in SQL.)

---

## 3.4 Map Methods

Normal datatypes have means for comparisons. For example, numbers are compared according to size and strings according to alphabetical ordering. For object types, map methods define an ordering. In addition to map methods, there are also order methods, but they are not discussed in this tutorial.

The following map method orders the objects of type job (or job2) by years_of_experience multiplied with salary_amount.

```
ALTER TYPE job
    ADD MAP MEMBER FUNCTION sorting RETURN NUMBER
    CASCADE;

CREATE OR REPLACE TYPE BODY job AS
    MAP MEMBER FUNCTION sorting RETURN NUMBER IS
    BEGIN
    RETURN self.years_of_experience * self.salary_amount;
    END sorting;
    ....
END;
```

It is now possible to compare objects from job_table (for example, j1.sorting() < j2.sorting()) or to use sorting() in ORDER BY, GROUP BY and other statements.

```
select * from job_table j order by j.sorting() desc;
```

**Exercise**

- 24. Create a map method for "person" so that objects of type person are ordered by lastname, firstname, middle initial.