Методы сбора, хранения, обработки и анализа данных

Лекция 7

Объектные типы данных

Объектные типы данных

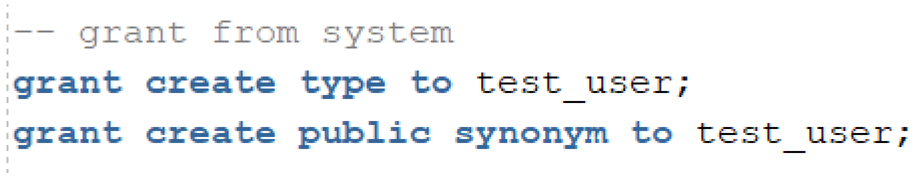
- Встроенные типы данных
  - Простые и привычные
- Объектные типы данных
  - Расширяют реляционную модель
  - Могут объединять данные и операции над ними
  - Могут эффективно использоваться
  - Могут показывать взаимосвязь и наследование данных

Объекты Oracle

- Объектные типы данных
- Экземпляры объектов
- Методы объектных типов
- Хранение объектов в таблицах
- Идентификация объектов
- Ссылки на объект
- Наследование типов
- Объектные представления
- Коллекции объектов

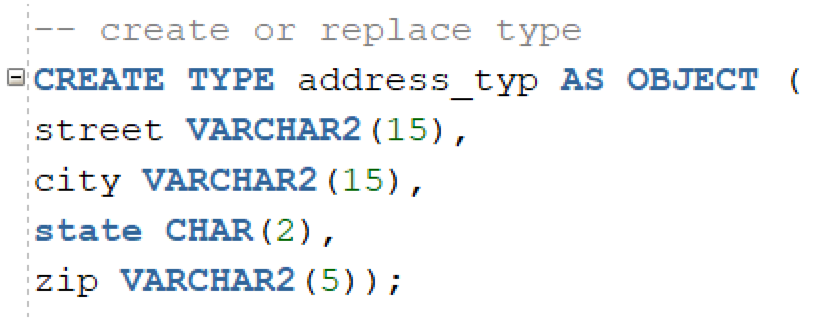
Разрешения

- Привилегии на создание объектов:
  - Таблиц
  - Типов
  - Представлений
  - Синонимов
  - Программного кода

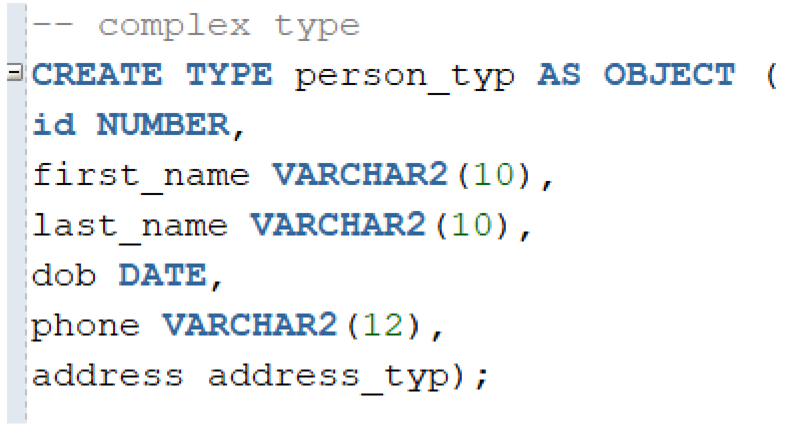


```
-- grant from system
grant create type to test_user;
grant create public synonym to test_user;
```

Объектные типы данных

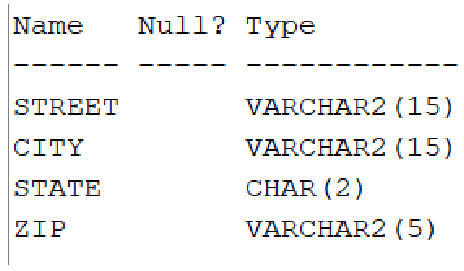


```
-- create or replace type
CREATE TYPE address_typ AS OBJECT (
street VARCHAR2(15),
city VARCHAR2(15),
state CHAR(2),
zip VARCHAR2(5));
```

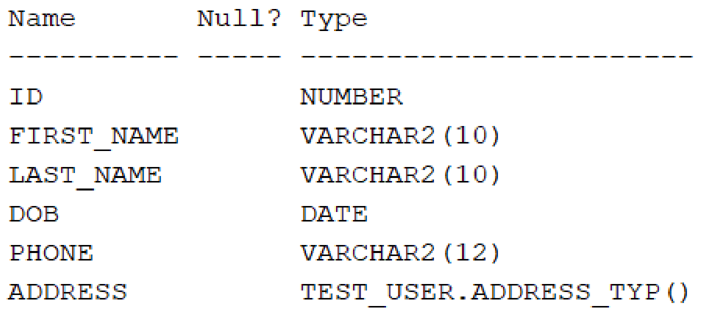


```
-- complex type
CREATE TYPE person_typ AS OBJECT (
id NUMBER,
first_name VARCHAR2(10),
last_name VARCHAR2(10),
dob DATE,
phone VARCHAR2(12),
address address_typ);
```

- Тип может быть создан на основе стандартных типов
- Или на основе созданного типа



```
Name   Null? Type
------ ----- ------------
STREET       VARCHAR2(15)
CITY         VARCHAR2(15)
STATE        CHAR(2)
ZIP          VARCHAR2(5)
```



```
Name       Null? Type
---------- ----- -----------------------
ID               NUMBER
FIRST_NAME       VARCHAR2(10)
LAST_NAME        VARCHAR2(10)
DOB              DATE
PHONE            VARCHAR2(12)
ADDRESS          TEST_USER.ADDRESS_TYP()
```

Объектные типы данных

```
CREATE TYPE product_typ AS OBJECT (
id NUMBER,
name VARCHAR2(15),
description VARCHAR2(22),
price NUMBER(5, 2),
days_valid NUMBER,
MEMBER FUNCTION get_sell_by_date RETURN DATE);

-- body of member function
-- separate compile
CREATE TYPE BODY product_typ AS
    MEMBER FUNCTION get_sell_by_date RETURN DATE IS
    v_sell_by_date DATE;
        BEGIN
            SELECT days_valid + SYSDATE
            INTO v_sell_by_date
            FROM dual;
        RETURN v_sell_by_date;
        END;
END;
```

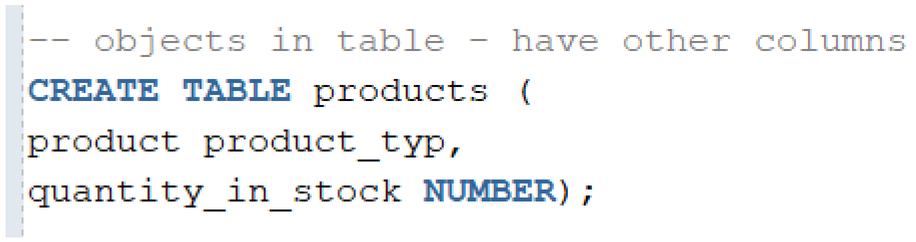
- Тип может содержать методы:
  - Методы member
  - Методы конструкторы
  - Статические методы
  - Методы сравнения
- Заголовок и тело метода компилируются отдельно

```
Name        Null? Type
----------- ----- ------------
ID                NUMBER
NAME              VARCHAR2(15)
DESCRIPTION       VARCHAR2(22)
PRICE             NUMBER(5,2)
DAYS_VALID        NUMBER

METHOD
------
MEMBER FUNCTION GET_SELL_BY_DATE RETURNS DATE
```

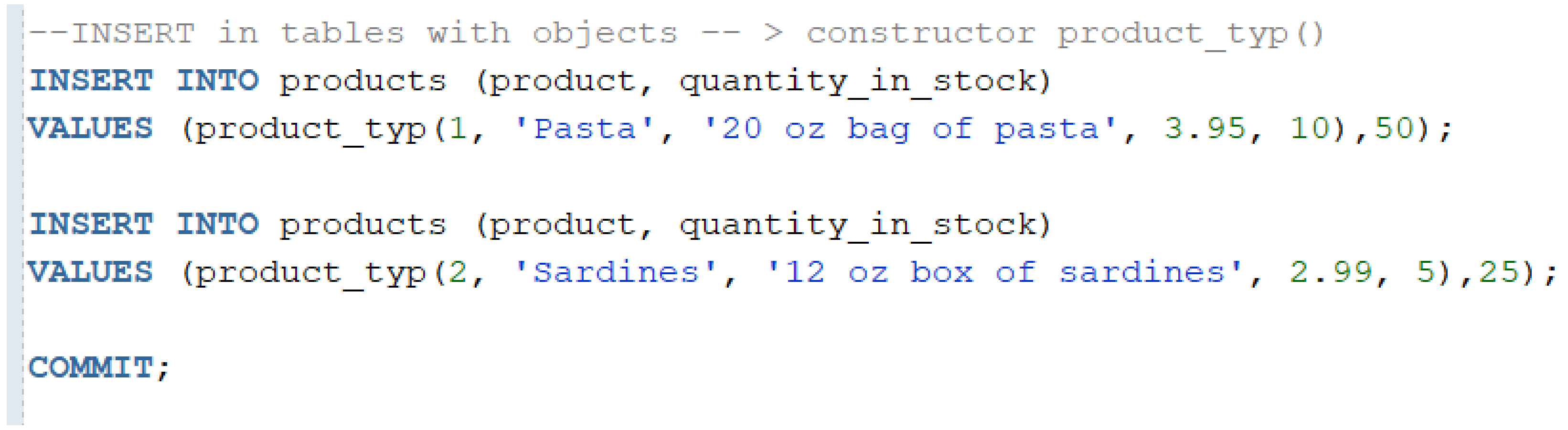
```
-- public synonim for type -- grant
CREATE PUBLIC SYNONYM pub_product_typ FOR product_typ;
```

Хранение объектов в таблицах

- Объектные таблицы – таблица состоит из строк объектного типа
- Таблицы, содержащие объекты – есть и другие столбцы

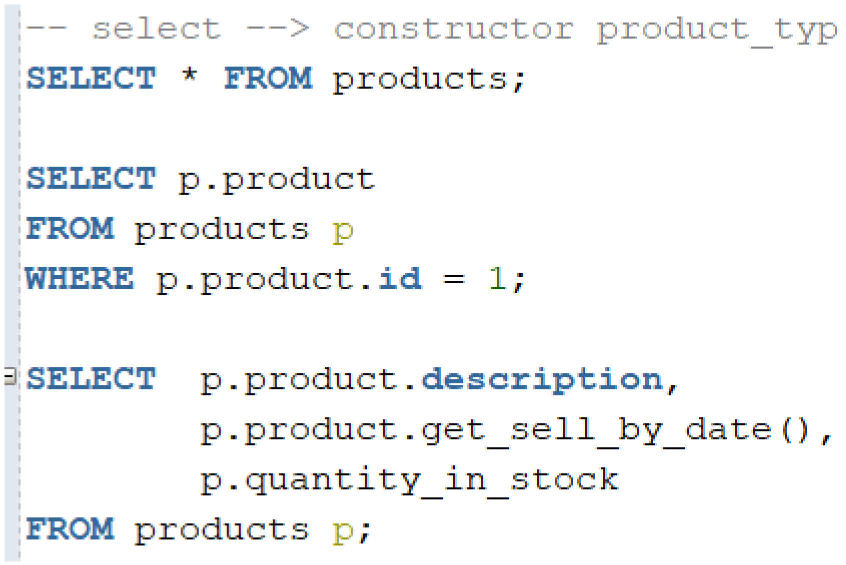


```sql
-- objects in table - have other columns
CREATE TABLE products (
product product_typ,
quantity_in_stock NUMBER);
```

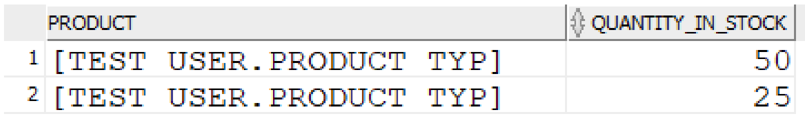


```sql
--INSERT in tables with objects -- > constructor product_typ()
INSERT INTO products (product, quantity_in_stock)
VALUES (product_typ(1, 'Pasta', '20 oz bag of pasta', 3.95, 10),50);

INSERT INTO products (product, quantity_in_stock)
VALUES (product_typ(2, 'Sardines', '12 oz box of sardines', 2.99, 5),25);

COMMIT;
```

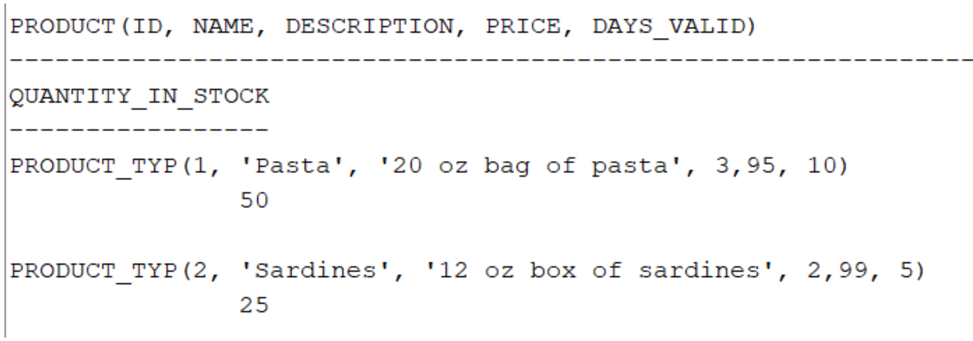
Хранение объектов в таблицах



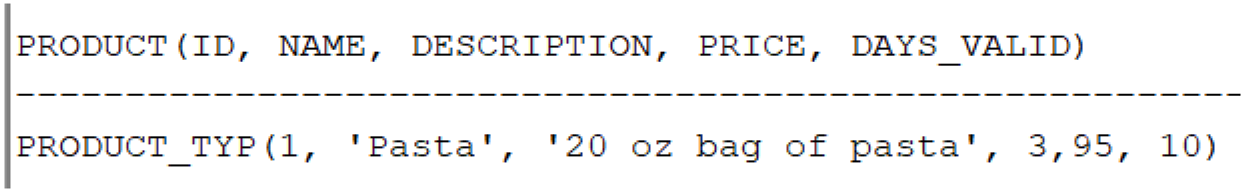
```sql
-- select --> constructor product_typ()
SELECT * FROM products;

SELECT p.product
FROM products p
WHERE p.product.id = 1;

SELECT  p.product.description,
        p.product.get_sell_by_date(),
        p.quantity_in_stock
FROM products p;
```



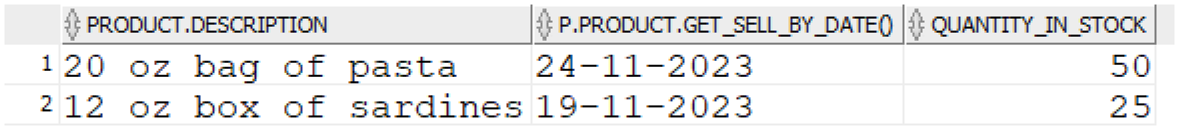
	PRODUCT	QUANTITY_IN_STOCK
1	[TEST USER.PRODUCT TYP]	50
2	[TEST USER.PRODUCT TYP]	25



```
PRODUCT(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
-----------------------------------------------------------------
QUANTITY_IN_STOCK
-----------------
PRODUCT_TYP(1, 'Pasta', '20 oz bag of pasta', 3,95, 10)
                  50

PRODUCT_TYP(2, 'Sardines', '12 oz box of sardines', 2,99, 5)
                  25
```

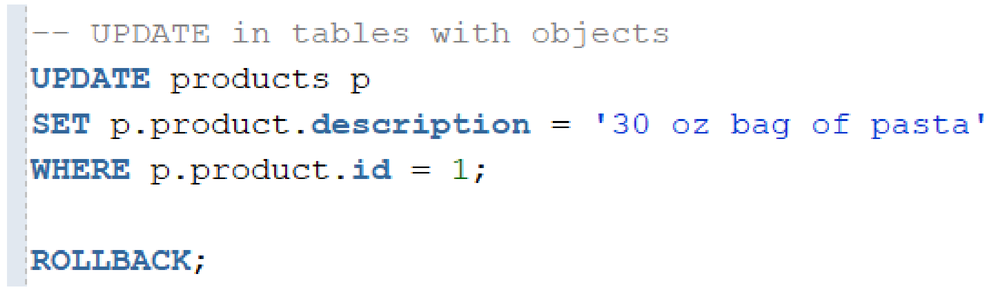


```
PRODUCT(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
--------------------------------------------------------
PRODUCT_TYP(1, 'Pasta', '20 oz bag of pasta', 3,95, 10)
```

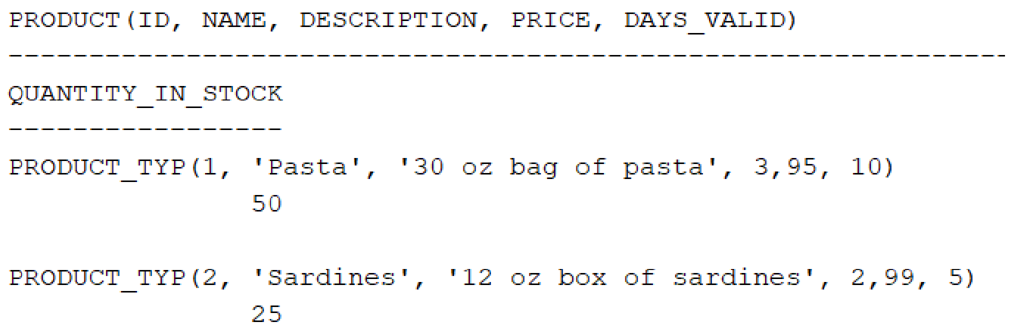


	PRODUCT.DESCRIPTION	P.PRODUCT.GET_SELL_BY_DATE()	QUANTITY_IN_STOCK
1	20 oz bag of pasta	24-11-2023	50
2	12 oz box of sardines	19-11-2023	25

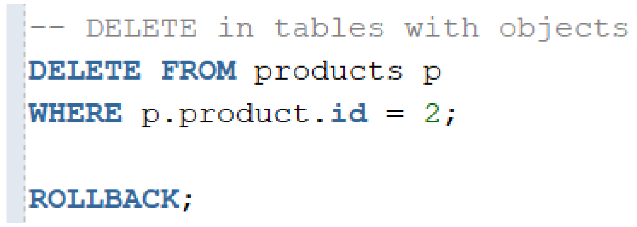
Экземпляры объектов - изменение



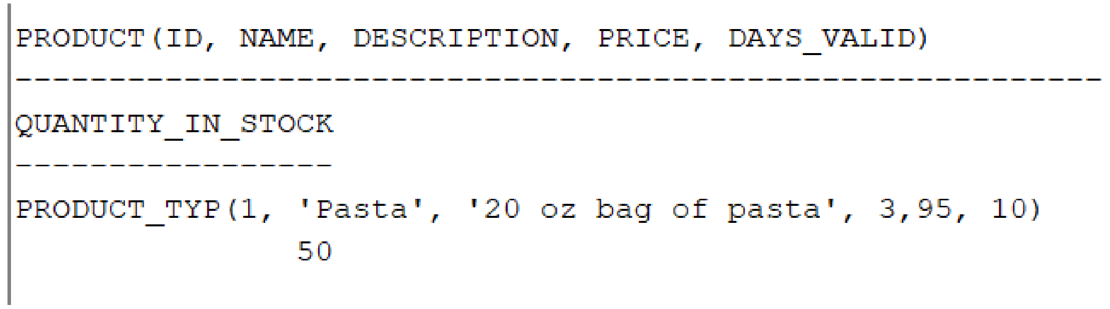
```sql
-- UPDATE in tables with objects
UPDATE products p
SET p.product.description = '30 oz bag of pasta'
WHERE p.product.id = 1;

ROLLBACK;
```



```
PRODUCT(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
--------------------------------------------------------------
QUANTITY_IN_STOCK
-----------------
PRODUCT_TYP(1, 'Pasta', '30 oz bag of pasta', 3,95, 10)
               50

PRODUCT_TYP(2, 'Sardines', '12 oz box of sardines', 2,99, 5)
               25
```



```sql
-- DELETE in tables with objects
DELETE FROM products p
WHERE p.product.id = 2;

ROLLBACK;
```



```
PRODUCT(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
--------------------------------------------------------------
QUANTITY_IN_STOCK
-----------------
PRODUCT_TYP(1, 'Pasta', '20 oz bag of pasta', 3,95, 10)
               50
```

Хранение объектов в таблицах

```
-- object tables - have no other columns
CREATE TABLE object_products OF product_typ;

CREATE TABLE object_customers OF person_typ;
```

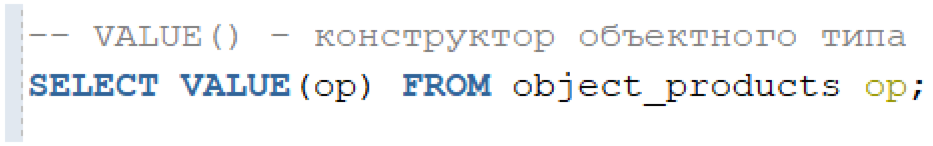
```
Name         Null? Type
------------ ----- ------------
ID                 NUMBER
NAME               VARCHAR2(15)
DESCRIPTION        VARCHAR2(22)
PRICE              NUMBER(5,2)
DAYS_VALID         NUMBER
```

```
-- INSERT in object tables
INSERT INTO object_products VALUES (
product_typ(1, 'Pasta', '20 oz bag of pasta', 3.95, 10));

INSERT INTO object_products (id, name, description, price, days_valid)
VALUES (2, 'Sardines', '12 oz box of sardines', 2.99, 5);
```

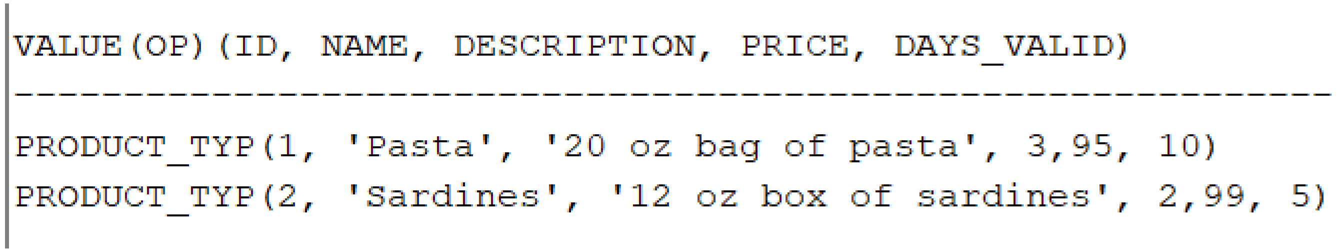
```
      ID NAME            DESCRIPTION                PRICE DAYS_VALID
-------- --------------- ---------------------- ---------- ----------
       1 Pasta           20 oz bag of pasta           3,95         10
       2 Sardines        12 oz box of sardines        2,99          5
```

Значение объекта

- Функция VALUE() – получение значений в объектных таблицах



```sql
-- VALUE() - конструктор объектного типа
SELECT VALUE(op) FROM object_products op;
```



```
VALUE(OP)(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
------------------------------------------------------------
PRODUCT_TYP(1, 'Pasta', '20 oz bag of pasta', 3,95, 10)
PRODUCT_TYP(2, 'Sardines', '12 oz box of sardines', 2,99, 5)
```

Объектные таблицы

- DML операции в объектных таблицах – аналогично стандартным

```
-- UPDATE
UPDATE object_products
SET description = '25 oz bag of pasta'
WHERE id = 1;
```

```
VALUE(OP)(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
------------------------------------------------------------
PRODUCT_TYP(1, 'Pasta', '25 oz bag of pasta', 3,95, 10)
PRODUCT_TYP(2, 'Sardines', '12 oz box of sardines', 2,99, 5)
```

```
DELETE FROM object_products
WHERE id = 2;
```

```
VALUE(OP)(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
------------------------------------------------------------
PRODUCT_TYP(1, 'Pasta', '25 oz bag of pasta', 3,95, 10)
```

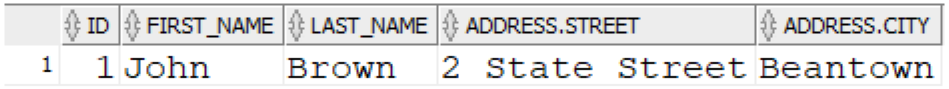
Вложенность объектов

- Вложенность типов – обращение через точку

```
-- INSERT object in object
ALTER SESSION SET nls_date_format = 'DD-MM-YY';
INSERT INTO object_customers
VALUES (person_typ(1, 'John', 'Brown', '01-02-1955', '800-555-1211',
address_typ('2 State Street', 'Beantown', 'MA', '12345')));

INSERT INTO object_customers (id, first_name, last_name, dob, phone,address)
VALUES (2, 'Cynthia', 'Green', '05-02-1968', '800-555-1212',
address_typ('3 Free Street', 'Middle Town', 'CA', '12345'));
COMMIT;
```

```
        ID FIRST_NAME LAST_NAME  DOB      PHONE
---------- ---------- ---------- -------- ------------
ADDRESS(STREET, CITY, STATE, ZIP)
------------------------------------------------------------
         1 John       Brown      01-02-55 800-555-1211
ADDRESS_TYP('2 State Street', 'Beantown', 'MA', '12345')

         2 Cynthia    Green      05-02-68 800-555-1212
ADDRESS_TYP('3 Free Street', 'Middle Town', 'CA', '12345')
```

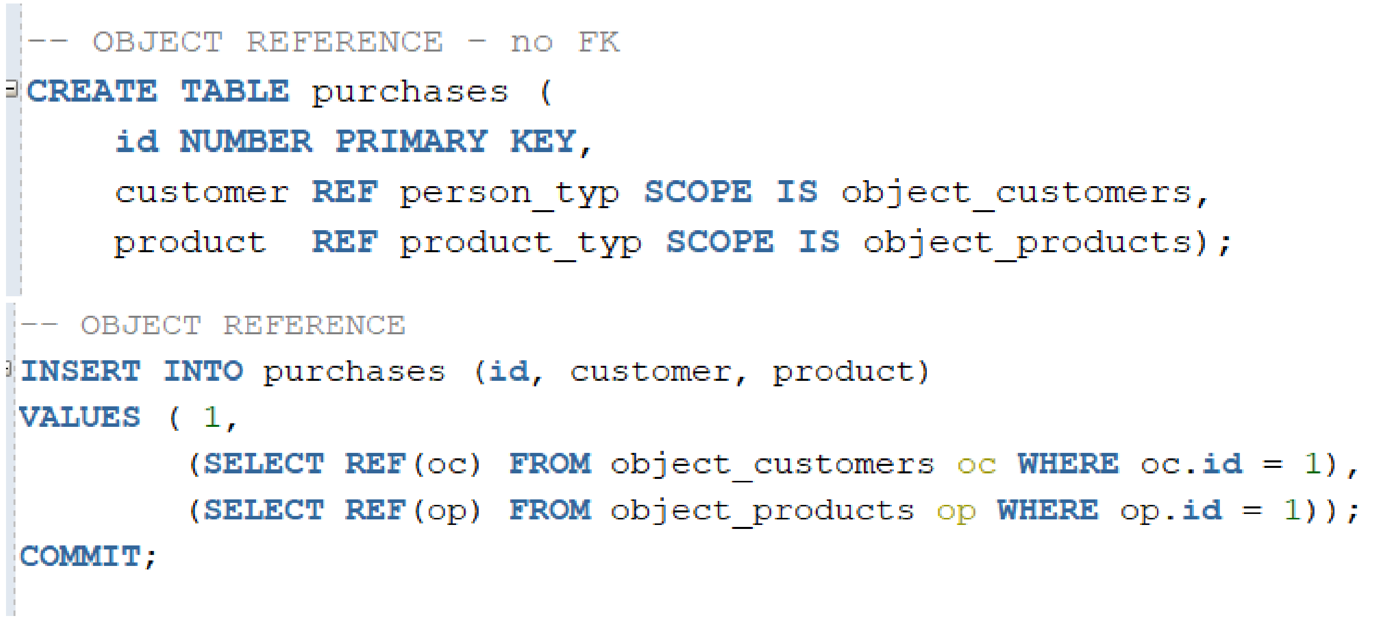
```
SELECT oc.id, oc.first_name, oc.last_name, oc.address.street, oc.address.city
FROM object_customers oc
WHERE oc.id = 1;
```



	ID	FIRST_NAME	LAST_NAME	ADDRESS.STREET	ADDRESS.CITY
1	1	John	Brown	2 State Street	Beantown

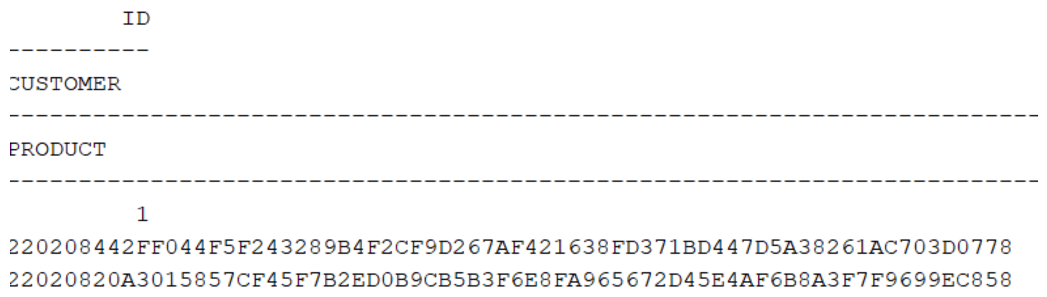
Ссылки на объекты в таблицах

- Вместо внешних ключей используется связь по OID



```
-- OBJECT REFERENCE - no FK
CREATE TABLE purchases (
    id NUMBER PRIMARY KEY,
    customer REF person_typ SCOPE IS object_customers,
    product  REF product_typ SCOPE IS object_products);
```

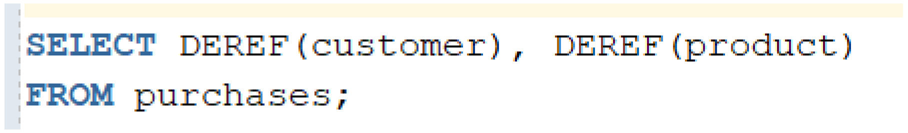
```
-- OBJECT REFERENCE
INSERT INTO purchases (id, customer, product)
VALUES ( 1,
        (SELECT REF(oc) FROM object_customers oc WHERE oc.id = 1),
        (SELECT REF(op) FROM object_products op WHERE op.id = 1));
COMMIT;
```



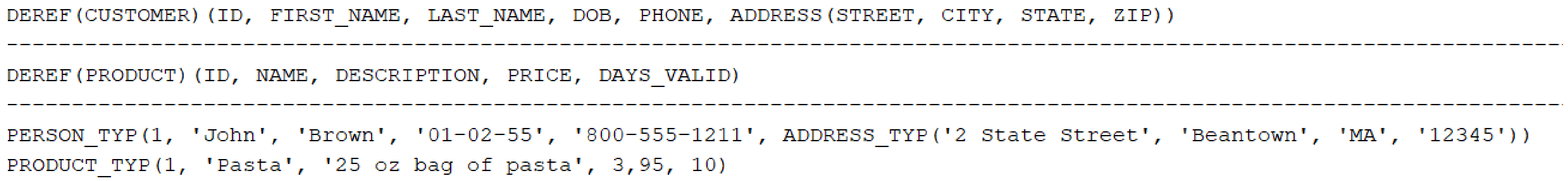
```
        ID
----------
CUSTOMER
------------------------------------------------------------------------
PRODUCT
------------------------------------------------------------------------
         1
220208442FF044F5F243289B4F2CF9D267AF421638FD371BD447D5A38261AC703D0778
22020820A3015857CF45F7B2ED0B9CB5B3F6E8FA965672D45E4AF6B8A3F7F9699EC858
```

Получение значения по ссылке

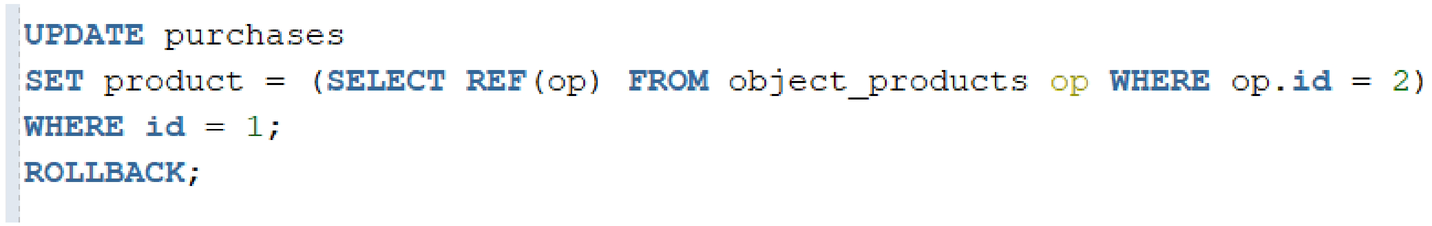
- Функция DEREF() – получение значения по ссылке



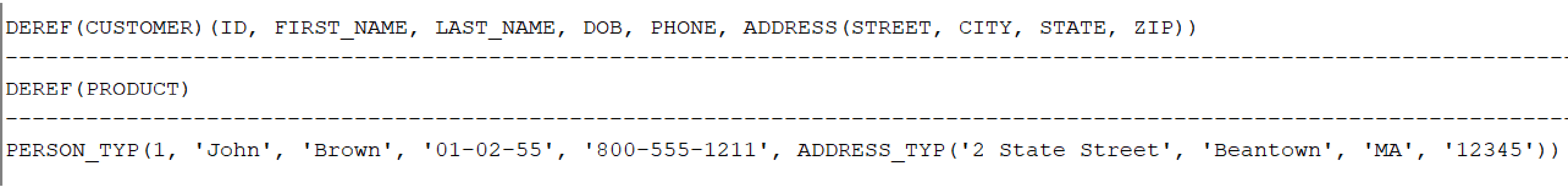
```sql
SELECT DEREF(customer), DEREF(product)
FROM purchases;
```



```
DEREF(CUSTOMER)(ID, FIRST_NAME, LAST_NAME, DOB, PHONE, ADDRESS(STREET, CITY, STATE, ZIP))
------------------------------------------------------------------------------------------------------------------------
DEREF(PRODUCT)(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
------------------------------------------------------------------------------------------------------------------------
PERSON_TYP(1, 'John', 'Brown', '01-02-55', '800-555-1211', ADDRESS_TYP('2 State Street', 'Beantown', 'MA', '12345'))
PRODUCT_TYP(1, 'Pasta', '25 oz bag of pasta', 3,95, 10)
```



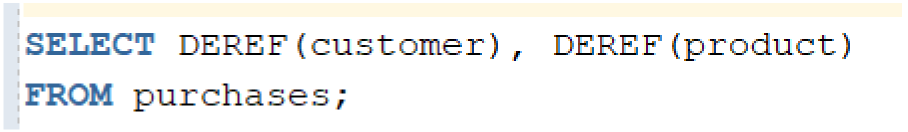
```sql
UPDATE purchases
SET product = (SELECT REF(op) FROM object_products op WHERE op.id = 2)
WHERE id = 1;
ROLLBACK;
```



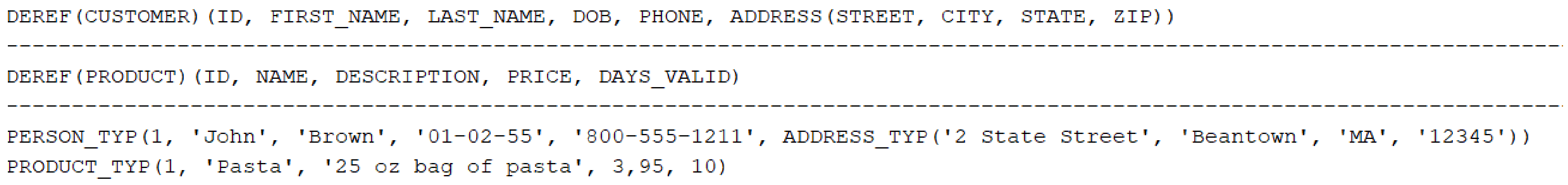
```
DEREF(CUSTOMER)(ID, FIRST_NAME, LAST_NAME, DOB, PHONE, ADDRESS(STREET, CITY, STATE, ZIP))
------------------------------------------------------------------------------------------------------------------------
DEREF(PRODUCT)
------------------------------------------------------------------------------------------------------------------------
PERSON_TYP(1, 'John', 'Brown', '01-02-55', '800-555-1211', ADDRESS_TYP('2 State Street', 'Beantown', 'MA', '12345'))
```

Ссылки на объекты

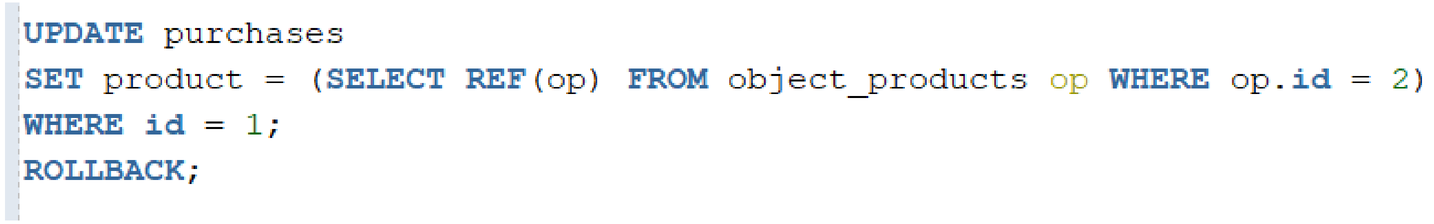
- Функция DEREF() – получение значения по ссылке
- Функция REF() – получение ссылки по значению



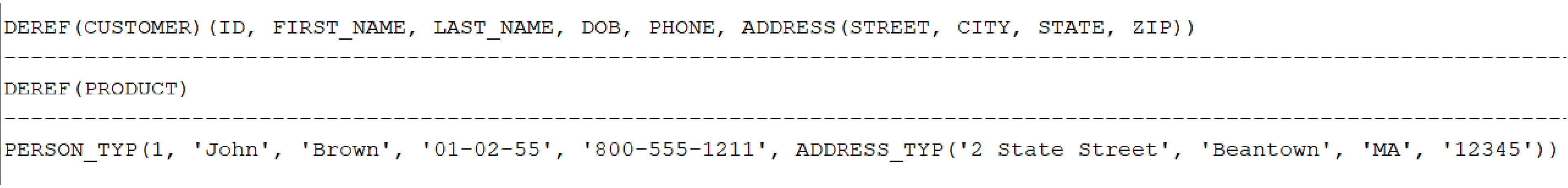
```
SELECT DEREF(customer), DEREF(product)
FROM purchases;
```



```
DEREF(CUSTOMER)(ID, FIRST_NAME, LAST_NAME, DOB, PHONE, ADDRESS(STREET, CITY, STATE, ZIP))
----------------------------------------------------------------------------------------------
DEREF(PRODUCT)(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
----------------------------------------------------------------------------------------------
PERSON_TYP(1, 'John', 'Brown', '01-02-55', '800-555-1211', ADDRESS_TYP('2 State Street', 'Beantown', 'MA', '12345'))
PRODUCT_TYP(1, 'Pasta', '25 oz bag of pasta', 3,95, 10)
```



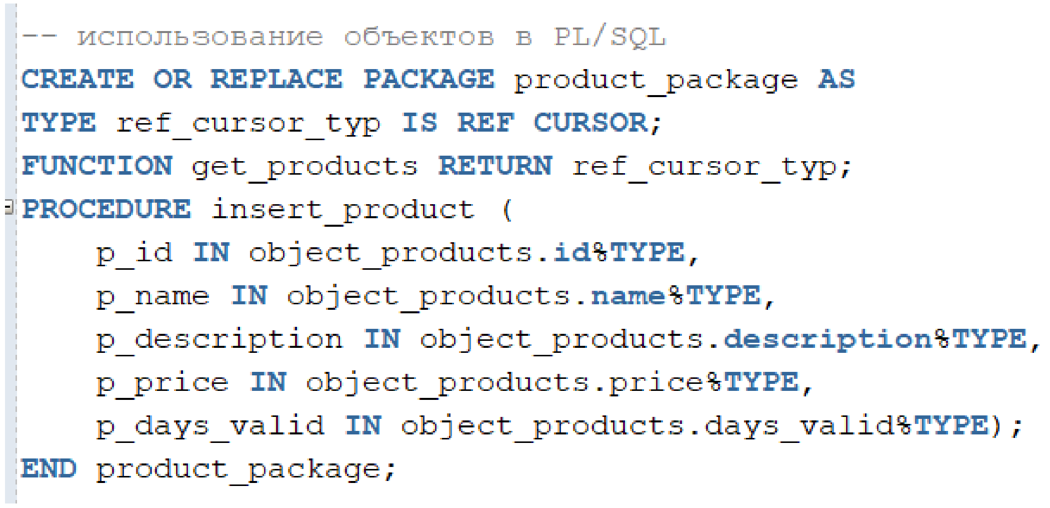
```
UPDATE purchases
SET product = (SELECT REF(op) FROM object_products op WHERE op.id = 2)
WHERE id = 1;
ROLLBACK;
```



```
DEREF(CUSTOMER)(ID, FIRST_NAME, LAST_NAME, DOB, PHONE, ADDRESS(STREET, CITY, STATE, ZIP))
----------------------------------------------------------------------------------------------
DEREF(PRODUCT)
----------------------------------------------------------------------------------------------
PERSON_TYP(1, 'John', 'Brown', '01-02-55', '800-555-1211', ADDRESS_TYP('2 State Street', 'Beantown', 'MA', '12345'))
```

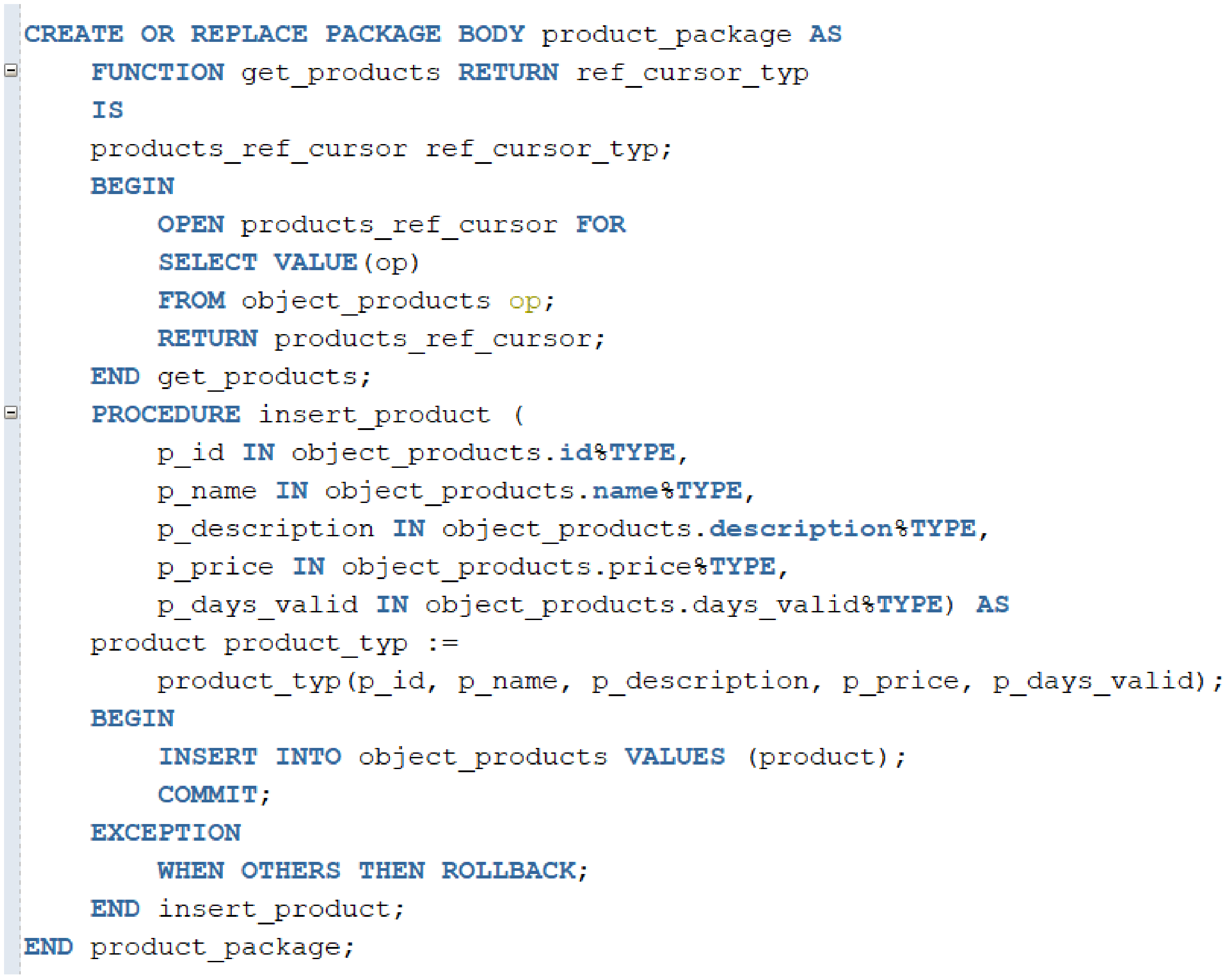
Объекты в PL/SQL

- Спецификация и реализация пакета – так же, как и для скалярных переменных
- Объектом может быть параметр, возвращаемое значение, переменная



```
-- использование объектов в PL/SQL
CREATE OR REPLACE PACKAGE product_package AS
TYPE ref_cursor_typ IS REF CURSOR;
FUNCTION get_products RETURN ref_cursor_typ;
PROCEDURE insert_product (
    p_id IN object_products.id%TYPE,
    p_name IN object_products.name%TYPE,
    p_description IN object_products.description%TYPE,
    p_price IN object_products.price%TYPE,
    p_days_valid IN object_products.days_valid%TYPE);
END product_package;
```

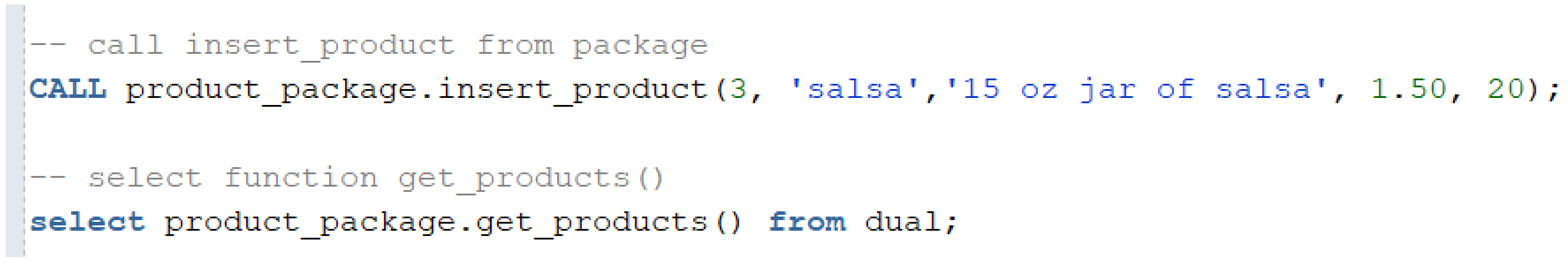
Объекты в PL/SQL



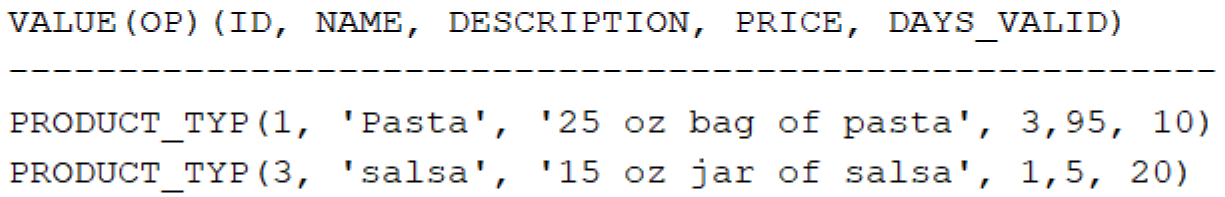
```
CREATE OR REPLACE PACKAGE BODY product_package AS
    FUNCTION get_products RETURN ref_cursor_typ
    IS
    products_ref_cursor ref_cursor_typ;
    BEGIN
        OPEN products_ref_cursor FOR
        SELECT VALUE(op)
        FROM object_products op;
        RETURN products_ref_cursor;
    END get_products;
    PROCEDURE insert_product (
        p_id IN object_products.id%TYPE,
        p_name IN object_products.name%TYPE,
        p_description IN object_products.description%TYPE,
        p_price IN object_products.price%TYPE,
        p_days_valid IN object_products.days_valid%TYPE) AS
    product product_typ :=
        product_typ(p_id, p_name, p_description, p_price, p_days_valid);
    BEGIN
        INSERT INTO object_products VALUES (product);
        COMMIT;
    EXCEPTION
        WHEN OTHERS THEN ROLLBACK;
    END insert_product;
END product_package;
```

Объекты в PL/SQL

- Вызов процедур и обращение к функции – аналогично скалярным переменным



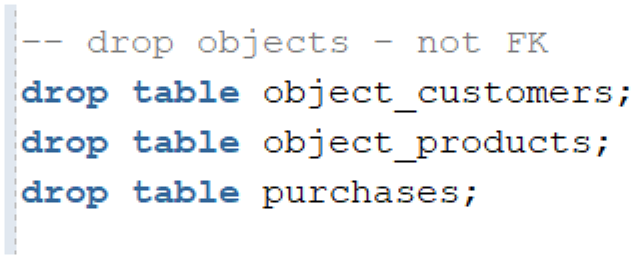
```
-- call insert_product from package
CALL product_package.insert_product(3, 'salsa','15 oz jar of salsa', 1.50, 20);

-- select function get_products()
select product_package.get_products() from dual;
```



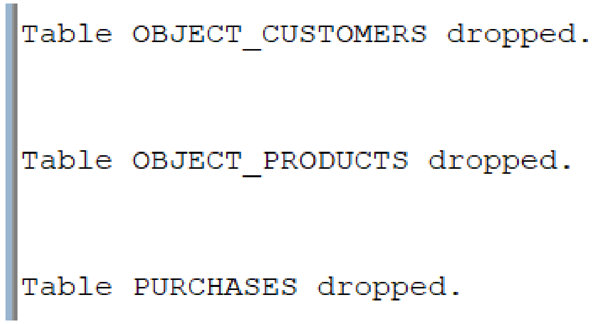
```
VALUE(OP)(ID, NAME, DESCRIPTION, PRICE, DAYS_VALID)
-------------------------------------------------------
PRODUCT_TYP(1, 'Pasta', '25 oz bag of pasta', 3,95, 10)
PRODUCT_TYP(3, 'salsa', '15 oz jar of salsa', 1,5, 20)
```

Удаление объектных таблиц

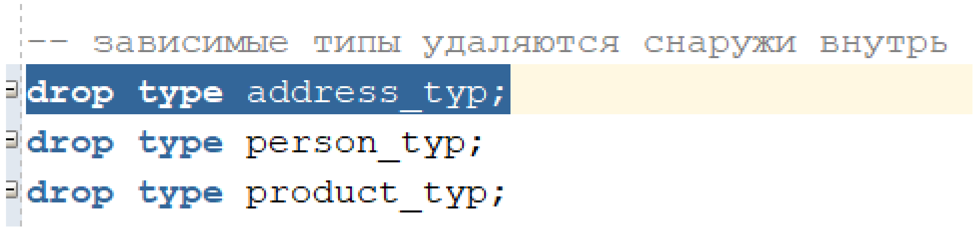
- Внешних ключей нет – порядок не важен
- Если тип используется в другом типе или таблице – его удалить нельзя



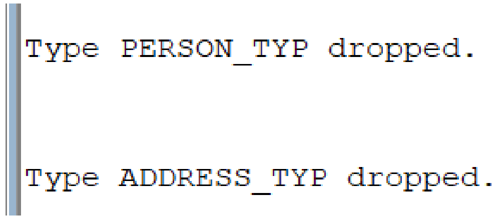
```
-- drop objects - not FK
drop table object_customers;
drop table object_products;
drop table purchases;
```



```
Table OBJECT_CUSTOMERS dropped.


Table OBJECT_PRODUCTS dropped.


Table PURCHASES dropped.
```



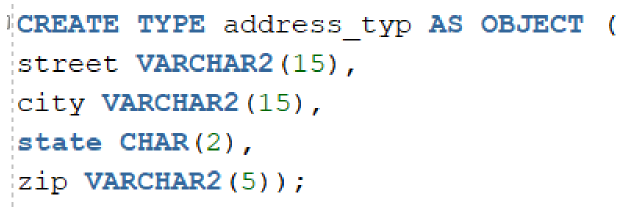
```
-- зависимые типы удаляются снаружи внутрь
drop type address_typ;
drop type person_typ;
drop type product_typ;
```



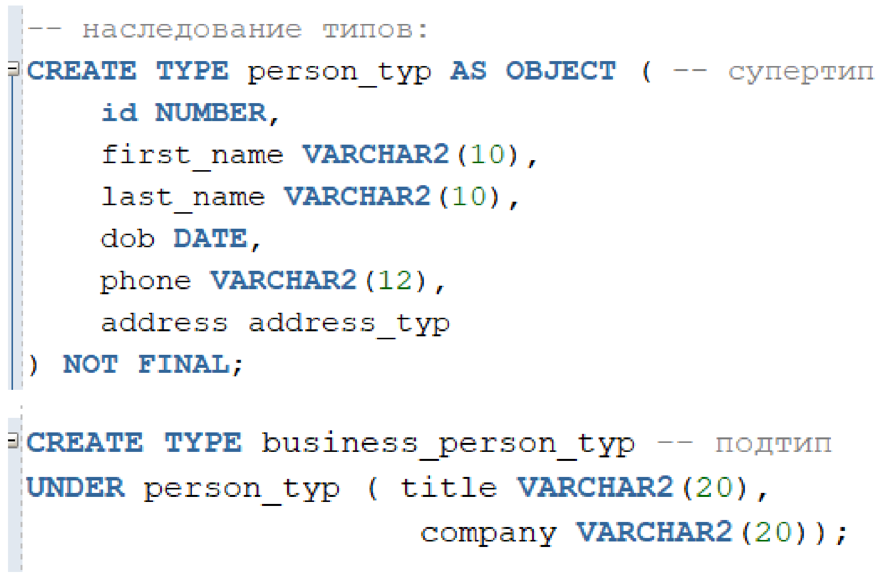
```
Type PERSON_TYP dropped.


Type ADDRESS_TYP dropped.
```

Наследование типов

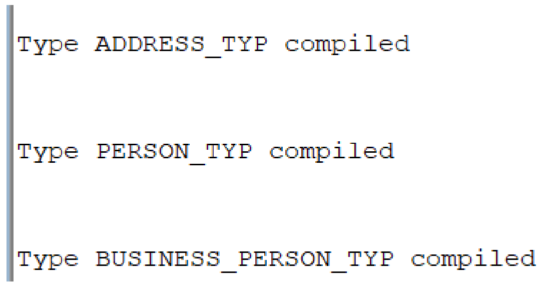
- Типы могут наследоваться
- NOT FINAL – тип будет наследоваться, NOT INSTANTIABLE – создать экземпляр типа нельзя



```
CREATE TYPE address_typ AS OBJECT (
street VARCHAR2(15),
city VARCHAR2(15),
state CHAR(2),
zip VARCHAR2(5));
```

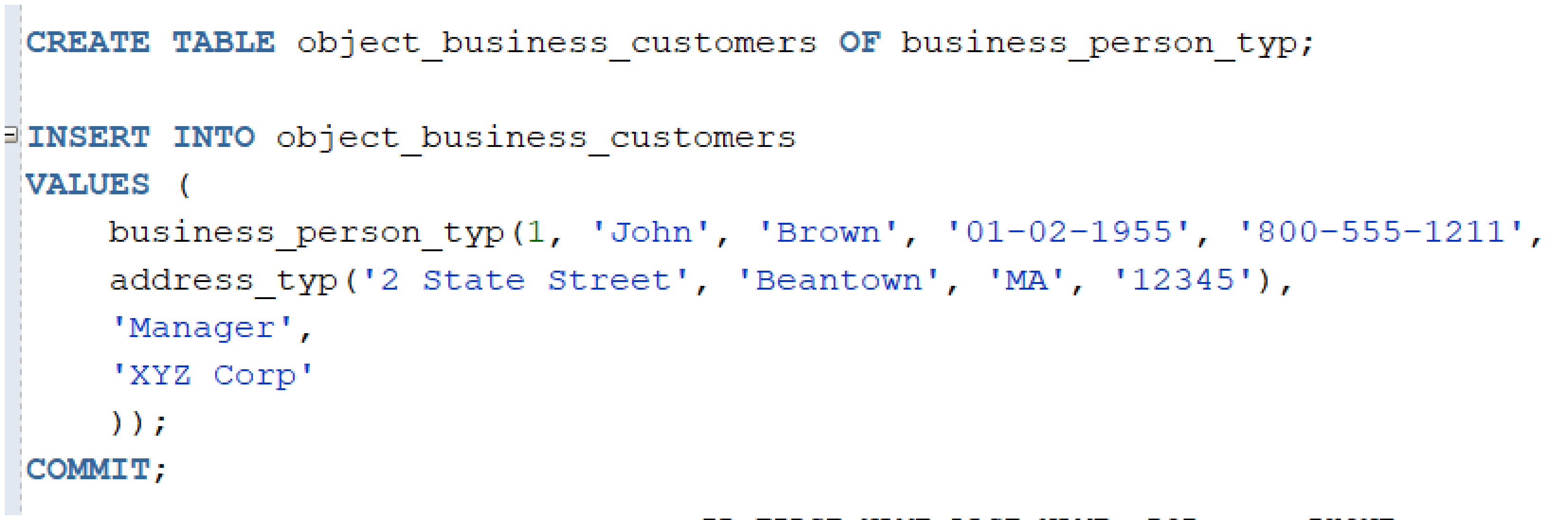


```
-- наследование типов:
CREATE TYPE person_typ AS OBJECT ( -- супертип
    id NUMBER,
    first_name VARCHAR2(10),
    last_name VARCHAR2(10),
    dob DATE,
    phone VARCHAR2(12),
    address address_typ
) NOT FINAL;

CREATE TYPE business_person_typ -- подтип
UNDER person_typ ( title VARCHAR2(20),
                   company VARCHAR2(20));
```

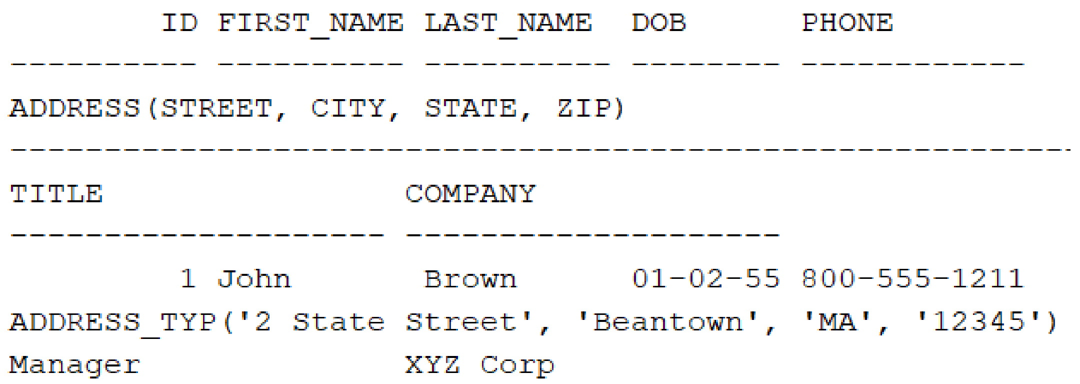


```
Type ADDRESS_TYP compiled


Type PERSON_TYP compiled


Type BUSINESS_PERSON_TYP compiled
```

Наследование типов



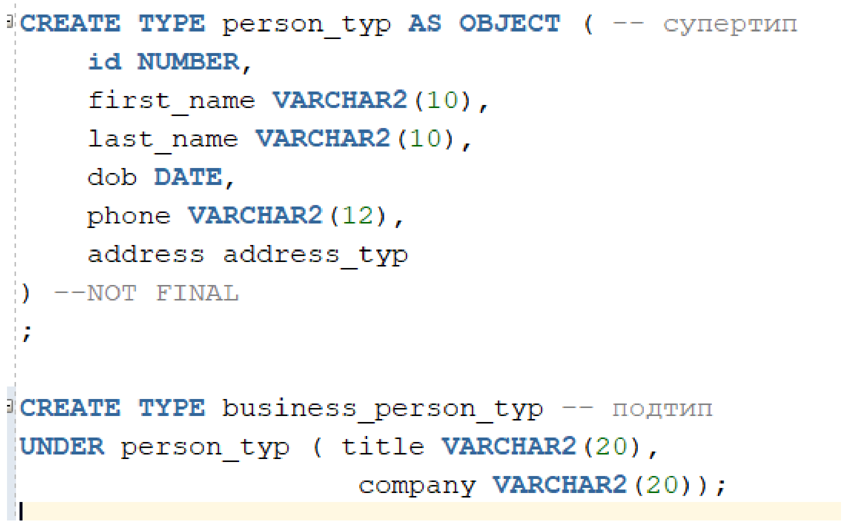
```
CREATE TABLE object_business_customers OF business_person_typ;

INSERT INTO object_business_customers
VALUES (
    business_person_typ(1, 'John', 'Brown', '01-02-1955', '800-555-1211',
    address_typ('2 State Street', 'Beantown', 'MA', '12345'),
    'Manager',
    'XYZ Corp'
    ));
COMMIT;
```



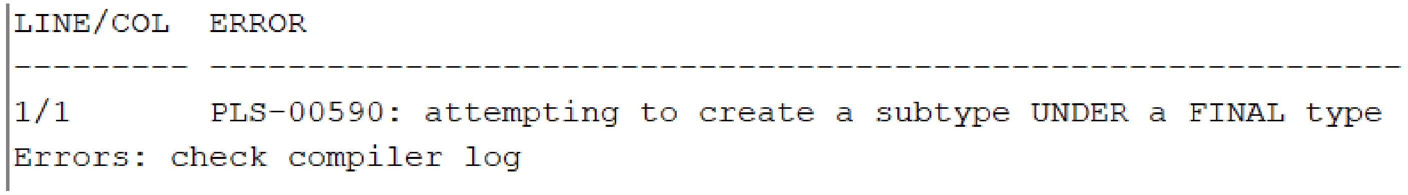
```
        ID FIRST_NAME LAST_NAME  DOB       PHONE
---------- ---------- ---------- --------- ------------
ADDRESS(STREET, CITY, STATE, ZIP)
---------------------------------------------------------
TITLE                COMPANY
-------------------- --------------------
         1 John       Brown      01-02-55  800-555-1211
ADDRESS_TYP('2 State Street', 'Beantown', 'MA', '12345')
Manager              XYZ Corp
```

Наследование типов

```
Table OBJECT_BUSINESS_CUSTOMERS dropped.


Type BUSINESS_PERSON_TYP dropped.


Type PERSON_TYP dropped.


Type PERSON_TYP compiled


Type BUSINESS_PERSON_TYP compiled
```

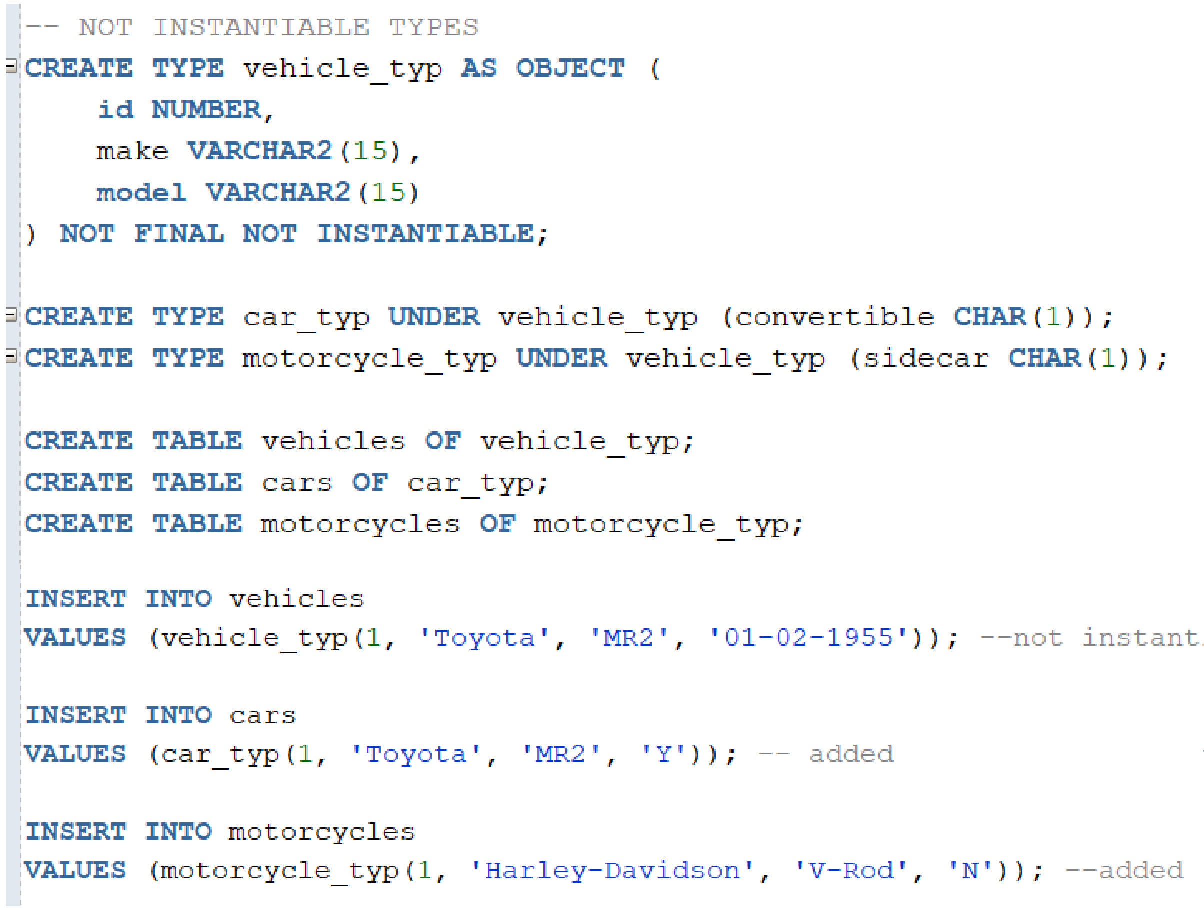


```sql
CREATE TYPE person_typ AS OBJECT ( -- супертип
    id NUMBER,
    first_name VARCHAR2(10),
    last_name VARCHAR2(10),
    dob DATE,
    phone VARCHAR2(12),
    address address_typ
) --NOT FINAL
;

CREATE TYPE business_person_typ -- подтип
UNDER person_typ ( title VARCHAR2(20),
                    company VARCHAR2(20));
```

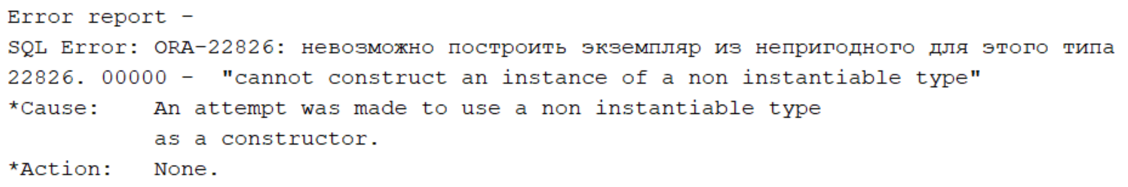


```
LINE/COL  ERROR
--------- -------------------------------------------------------------
1/1       PLS-00590: attempting to create a subtype UNDER a FINAL type
Errors: check compiler log
```

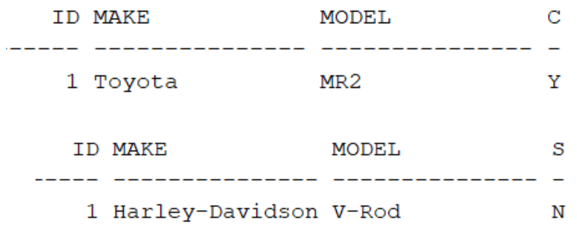
Наследование типов



```
-- NOT INSTANTIABLE TYPES
CREATE TYPE vehicle_typ AS OBJECT (
    id NUMBER,
    make VARCHAR2(15),
    model VARCHAR2(15)
) NOT FINAL NOT INSTANTIABLE;

CREATE TYPE car_typ UNDER vehicle_typ (convertible CHAR(1));
CREATE TYPE motorcycle_typ UNDER vehicle_typ (sidecar CHAR(1));

CREATE TABLE vehicles OF vehicle_typ;
CREATE TABLE cars OF car_typ;
CREATE TABLE motorcycles OF motorcycle_typ;

INSERT INTO vehicles
VALUES (vehicle_typ(1, 'Toyota', 'MR2', '01-02-1955')); --not instantiable

INSERT INTO cars
VALUES (car_typ(1, 'Toyota', 'MR2', 'Y')); -- added

INSERT INTO motorcycles
VALUES (motorcycle_typ(1, 'Harley-Davidson', 'V-Rod', 'N')); --added
```



```
Error report -
SQL Error: ORA-22826: невозможно построить экземпляр из непригодного для этого типа
22826. 00000 -  "cannot construct an instance of a non instantiable type"
*Cause:    An attempt was made to use a non instantiable type
           as a constructor.
*Action:   None.
```

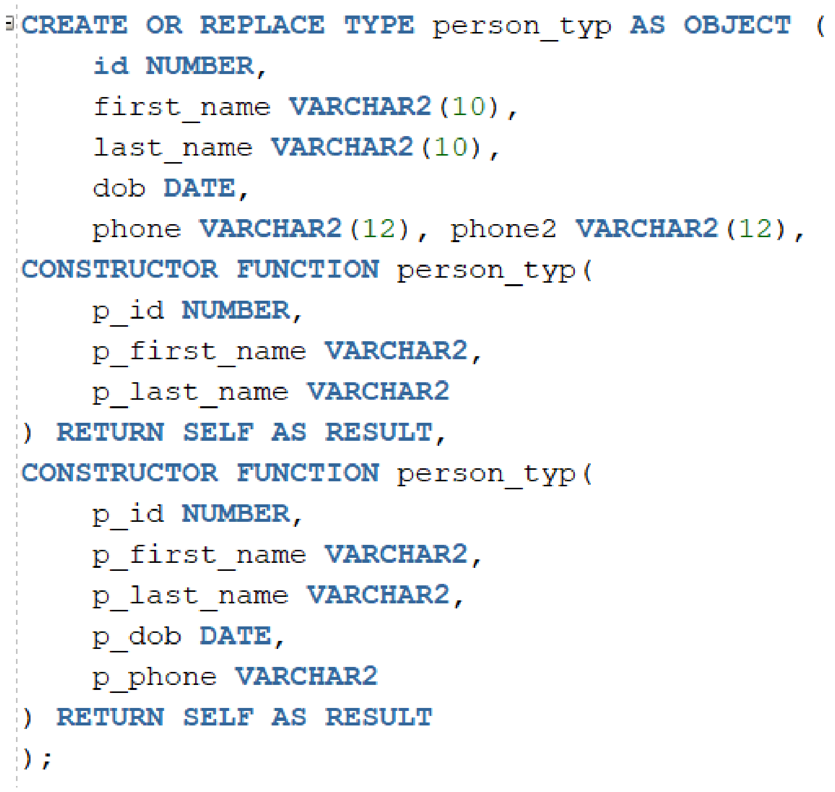


```
   ID MAKE            MODEL           C
----- --------------- --------------- -
    1 Toyota          MR2             Y
```

```
   ID MAKE            MODEL           S
----- --------------- --------------- -
    1 Harley-Davidson V-Rod           N
```

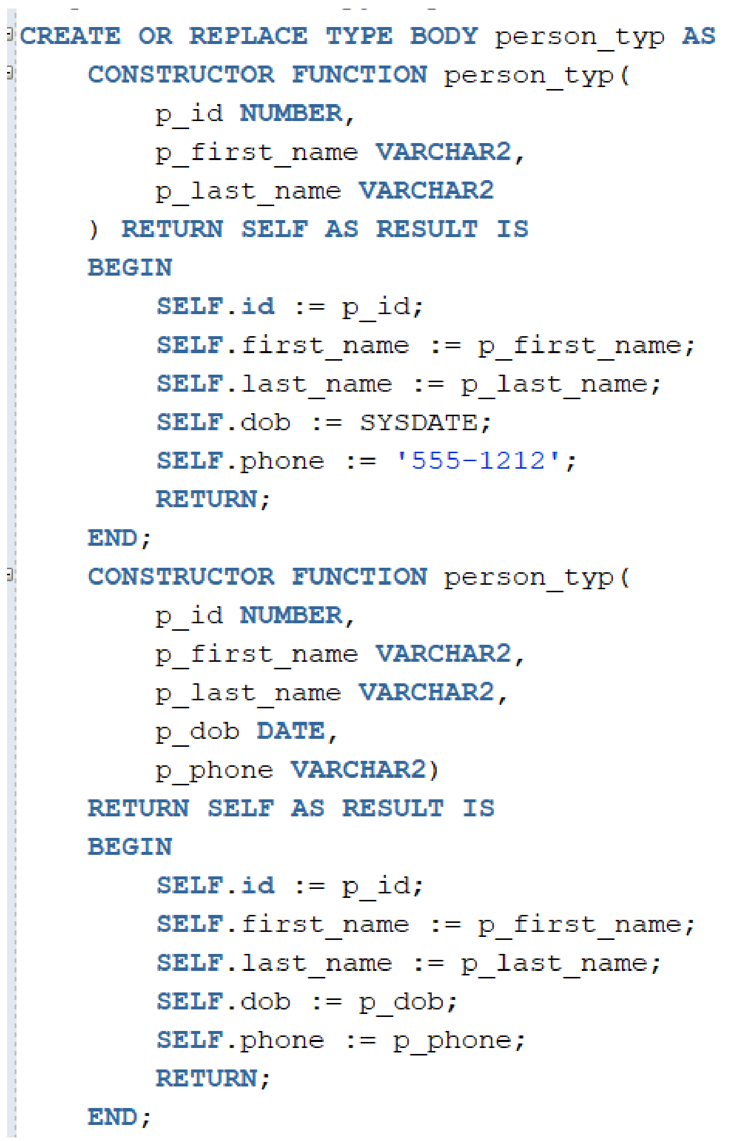
Методы - конструкторы

- Есть конструктор по умолчанию
- Можно создать дополнительные конструкторы
- Объявление и реализация компилируются отдельно

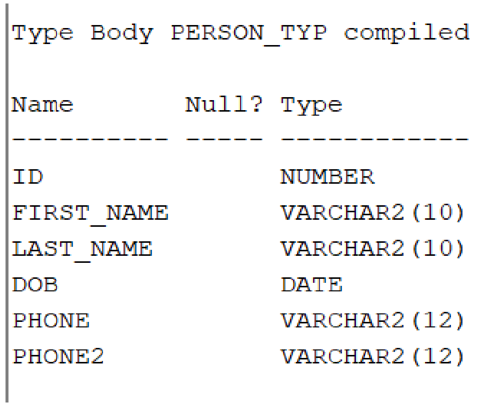


```
CREATE OR REPLACE TYPE person_typ AS OBJECT (
    id NUMBER,
    first_name VARCHAR2(10),
    last_name VARCHAR2(10),
    dob DATE,
    phone VARCHAR2(12), phone2 VARCHAR2(12),
CONSTRUCTOR FUNCTION person_typ(
    p_id NUMBER,
    p_first_name VARCHAR2,
    p_last_name VARCHAR2
) RETURN SELF AS RESULT,
CONSTRUCTOR FUNCTION person_typ(
    p_id NUMBER,
    p_first_name VARCHAR2,
    p_last_name VARCHAR2,
    p_dob DATE,
    p_phone VARCHAR2
) RETURN SELF AS RESULT
);
```

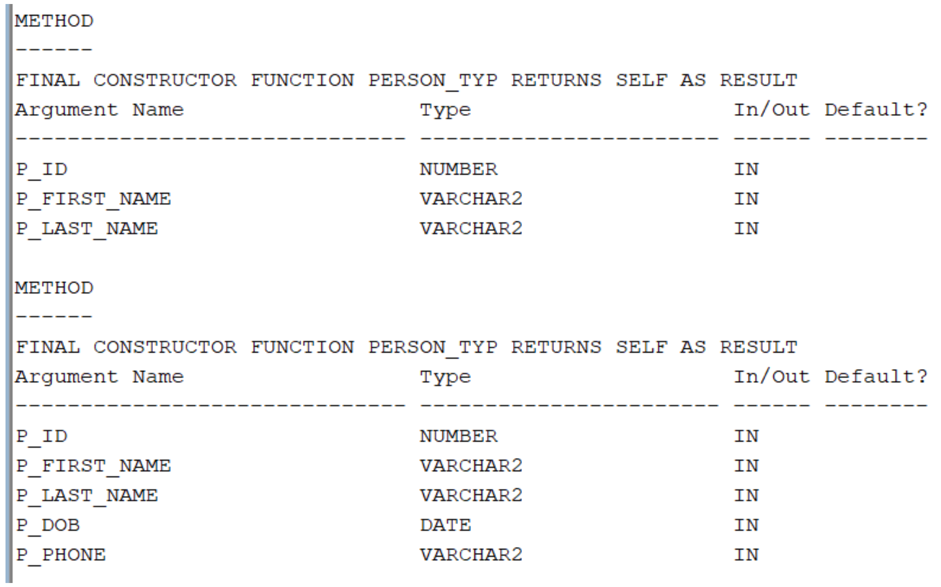
Методы - конструкторы



```
CREATE OR REPLACE TYPE BODY person_typ AS
    CONSTRUCTOR FUNCTION person_typ(
        p_id NUMBER,
        p_first_name VARCHAR2,
        p_last_name VARCHAR2
    ) RETURN SELF AS RESULT IS
    BEGIN
        SELF.id := p_id;
        SELF.first_name := p_first_name;
        SELF.last_name := p_last_name;
        SELF.dob := SYSDATE;
        SELF.phone := '555-1212';
        RETURN;
    END;
    CONSTRUCTOR FUNCTION person_typ(
        p_id NUMBER,
        p_first_name VARCHAR2,
        p_last_name VARCHAR2,
        p_dob DATE,
        p_phone VARCHAR2)
    RETURN SELF AS RESULT IS
    BEGIN
        SELF.id := p_id;
        SELF.first_name := p_first_name;
        SELF.last_name := p_last_name;
        SELF.dob := p_dob;
        SELF.phone := p_phone;
        RETURN;
    END;
```



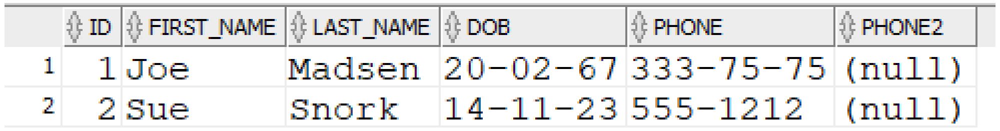
```
Type Body PERSON_TYP compiled

Name       Null? Type
---------- ----- ------------
ID               NUMBER
FIRST_NAME       VARCHAR2(10)
LAST_NAME        VARCHAR2(10)
DOB              DATE
PHONE            VARCHAR2(12)
PHONE2           VARCHAR2(12)
```



```
METHOD
------
FINAL CONSTRUCTOR FUNCTION PERSON_TYP RETURNS SELF AS RESULT
Argument Name                  Type                    In/Out Default?
------------------------------ ----------------------- ------ --------
P_ID                           NUMBER                  IN
P_FIRST_NAME                   VARCHAR2                IN
P_LAST_NAME                    VARCHAR2                IN

METHOD
------
FINAL CONSTRUCTOR FUNCTION PERSON_TYP RETURNS SELF AS RESULT
Argument Name                  Type                    In/Out Default?
------------------------------ ----------------------- ------ --------
P_ID                           NUMBER                  IN
P_FIRST_NAME                   VARCHAR2                IN
P_LAST_NAME                    VARCHAR2                IN
P_DOB                          DATE                    IN
P_PHONE                        VARCHAR2                IN
```

Методы - конструкторы

```
-- object table
CREATE TABLE object_customers OF person_typ;

INSERT INTO object_customers
VALUES (person_typ(1, 'Joe', 'Madsen', '20-02-1967', '333-75-75'));
INSERT INTO object_customers
VALUES (person_typ(2, 'Sue', 'Snork'));
COMMIT;

SELECT * FROM object_customers;
```

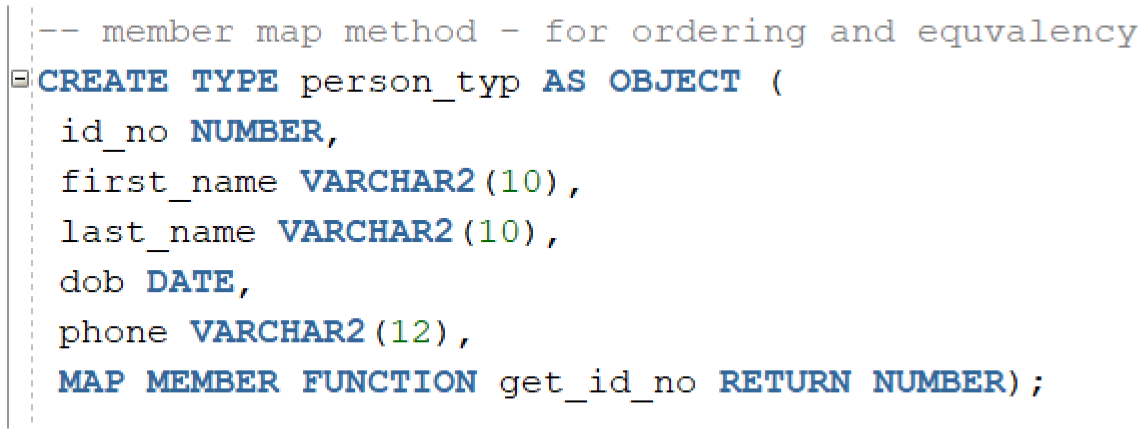


	ID	FIRST_NAME	LAST_NAME	DOB	PHONE	PHONE2
1	1	Joe	Madsen	20-02-67	333-75-75	(null)
2	2	Sue	Snork	14-11-23	555-1212	(null)

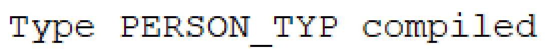
Методы сравнения

- Методы MAP – предназначены для сравнения, сортировки и UNION
- Методы ORDER – предназначены для сортировки по значениям полей

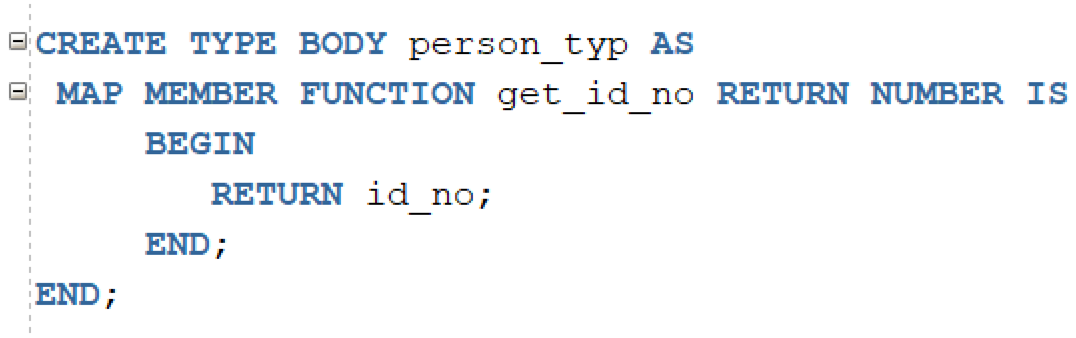
Метод сравнения MAP



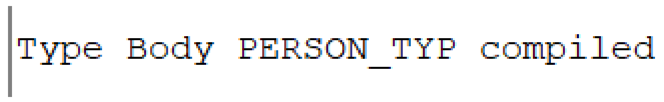
```
-- member map method - for ordering and equvalency
CREATE TYPE person_typ AS OBJECT (
 id_no NUMBER,
 first_name VARCHAR2(10),
 last_name VARCHAR2(10),
 dob DATE,
 phone VARCHAR2(12),
 MAP MEMBER FUNCTION get_id_no RETURN NUMBER);
```



```
Type PERSON_TYP compiled
```

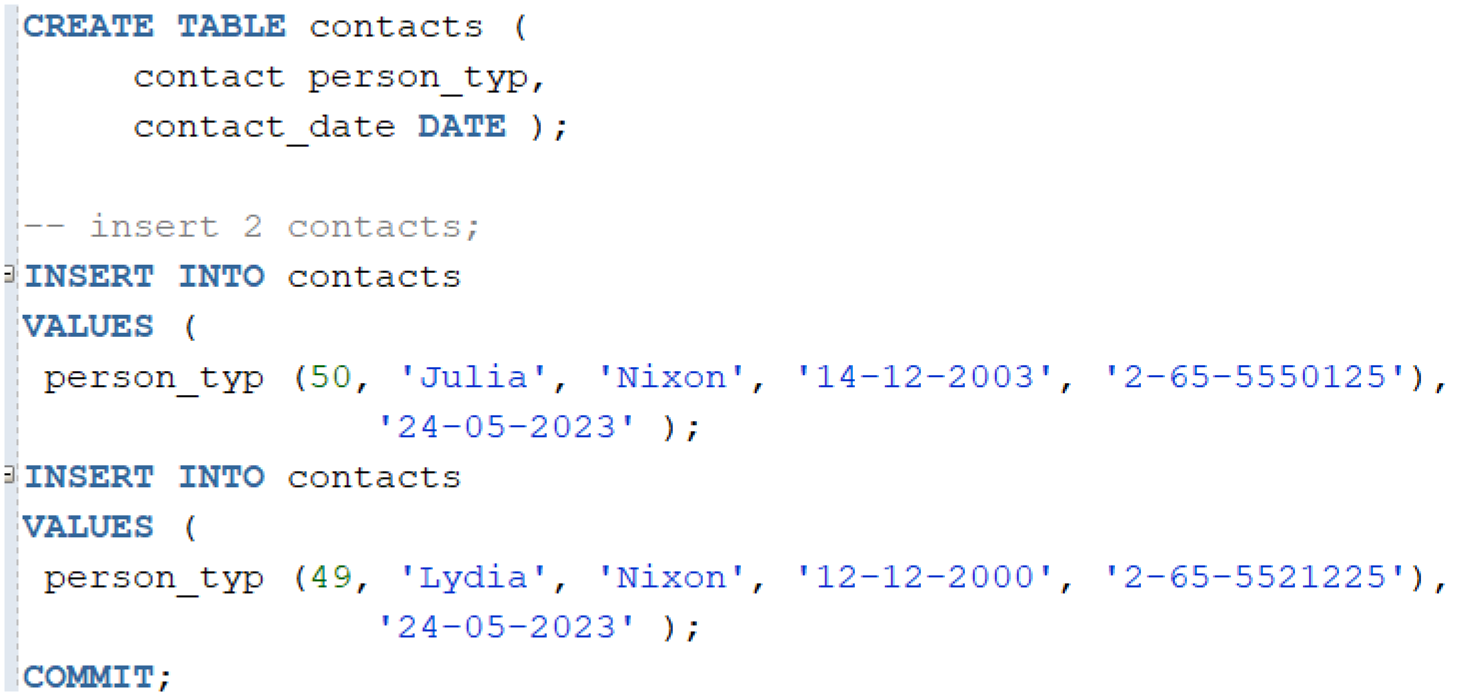


```
CREATE TYPE BODY person_typ AS
 MAP MEMBER FUNCTION get_id_no RETURN NUMBER IS
     BEGIN
        RETURN id_no;
     END;
END;
```

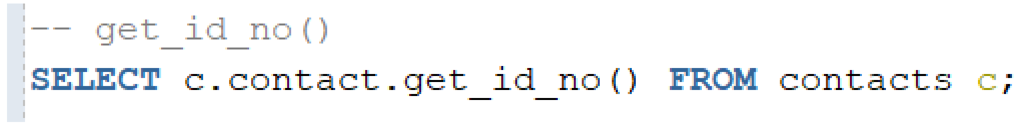


```
Type Body PERSON_TYP compiled
```

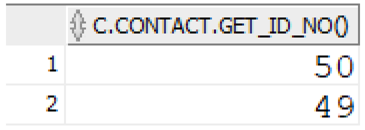
Метод сравнения MAP



```
CREATE TABLE contacts (
     contact person_typ,
     contact_date DATE );

-- insert 2 contacts;
INSERT INTO contacts
VALUES (
 person_typ (50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125'),
                '24-05-2023' );
INSERT INTO contacts
VALUES (
 person_typ (49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225'),
                '24-05-2023' );
COMMIT;
```

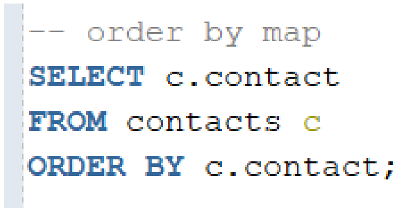


```
-- get_id_no()
SELECT c.contact.get_id_no() FROM contacts c;
```

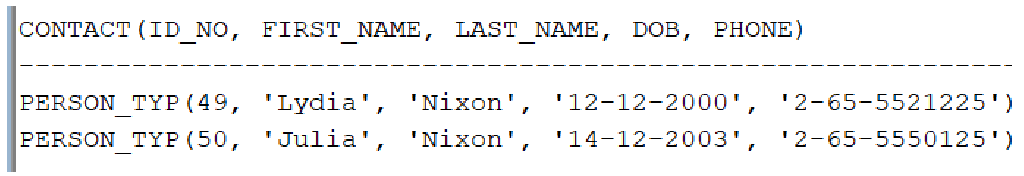


	C.CONTACT.GET_ID_NO()
1	50
2	49

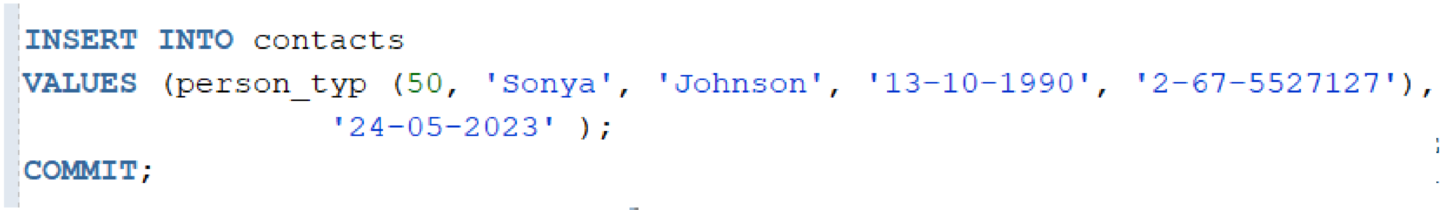
Метод сравнения MAP



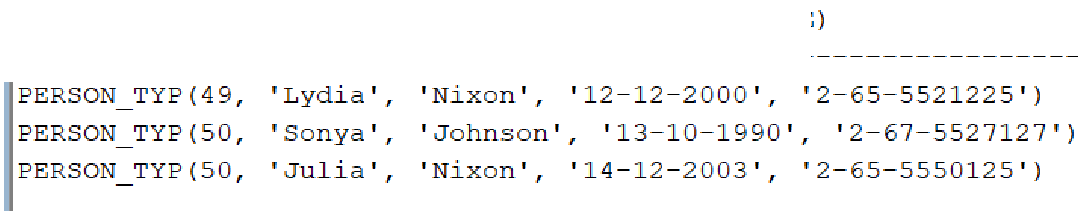
```
-- order by map
SELECT c.contact
FROM contacts c
ORDER BY c.contact;
```



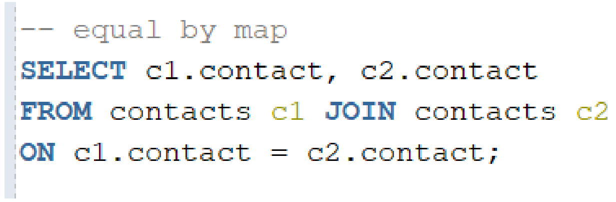
```
CONTACT(ID_NO, FIRST_NAME, LAST_NAME, DOB, PHONE)
-----------------------------------------------------------------
PERSON_TYP(49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225')
PERSON_TYP(50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125')
```



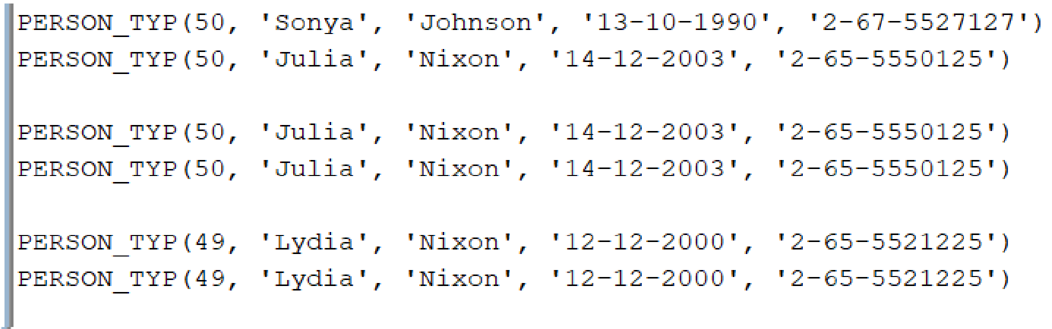
```
INSERT INTO contacts
VALUES (person_typ (50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127'),
                '24-05-2023' );
COMMIT;
```



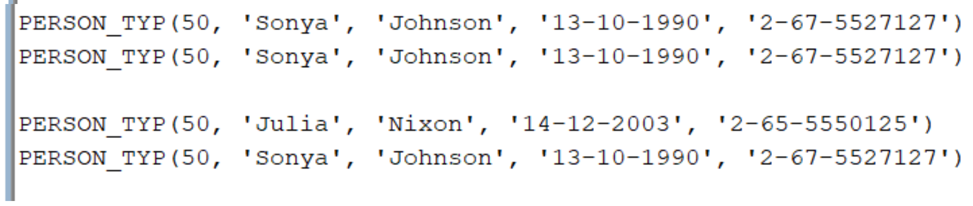
```
:)
------------------
PERSON_TYP(49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225')
PERSON_TYP(50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127')
PERSON_TYP(50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125')
```



```
-- equal by map
SELECT c1.contact, c2.contact
FROM contacts c1 JOIN contacts c2
ON c1.contact = c2.contact;
```

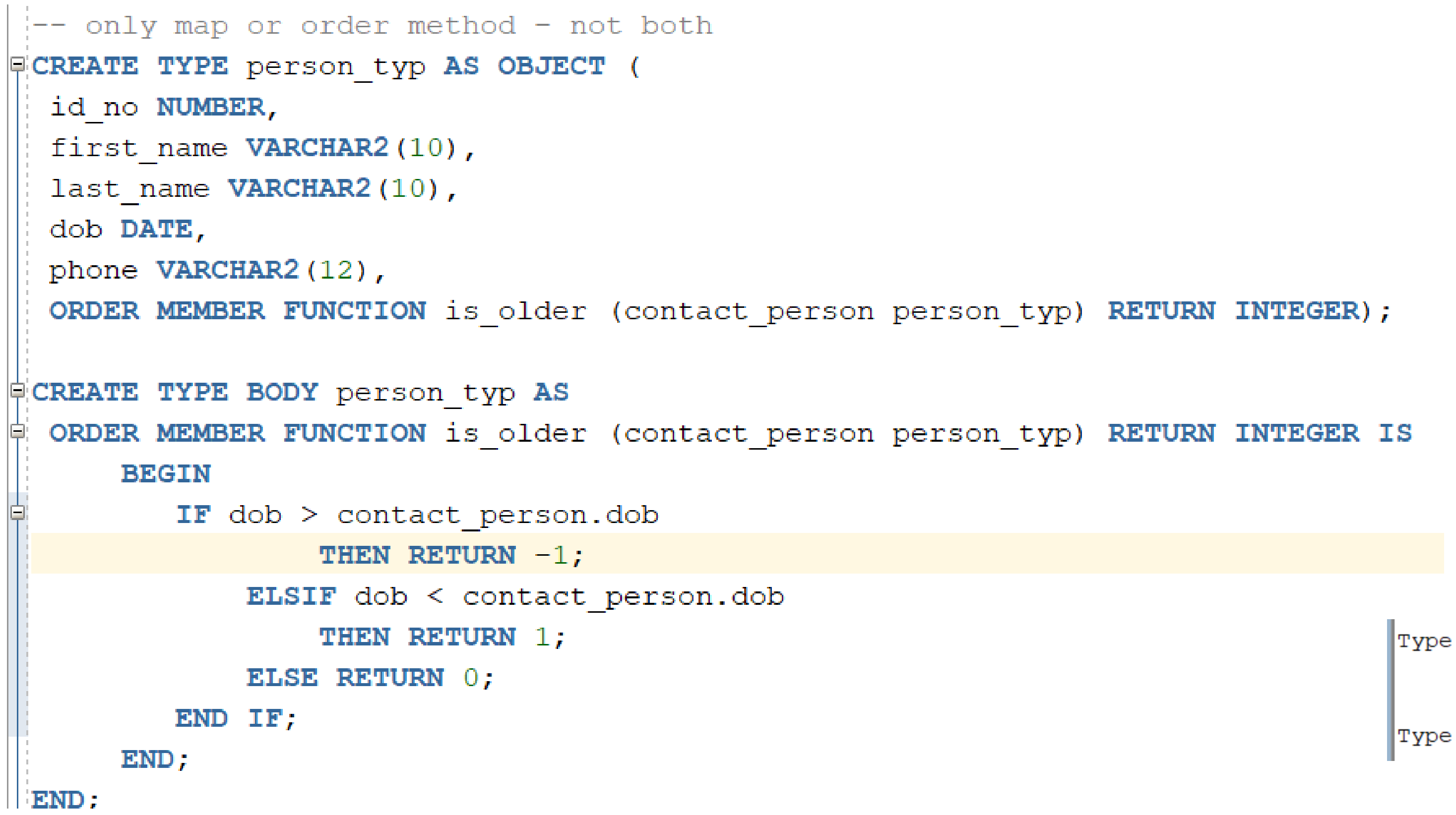


```
PERSON_TYP(50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127')
PERSON_TYP(50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125')

PERSON_TYP(50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125')
PERSON_TYP(50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125')

PERSON_TYP(49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225')
PERSON_TYP(49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225')
```

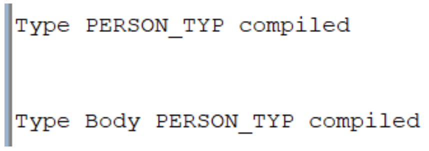


```
PERSON_TYP(50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127')
PERSON_TYP(50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127')

PERSON_TYP(50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125')
PERSON_TYP(50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127')
```

Метод сравнения ORDER



```
-- only map or order method - not both
CREATE TYPE person_typ AS OBJECT (
 id_no NUMBER,
 first_name VARCHAR2(10),
 last_name VARCHAR2(10),
 dob DATE,
 phone VARCHAR2(12),
 ORDER MEMBER FUNCTION is_older (contact_person person_typ) RETURN INTEGER);

CREATE TYPE BODY person_typ AS
 ORDER MEMBER FUNCTION is_older (contact_person person_typ) RETURN INTEGER IS
     BEGIN
        IF dob > contact_person.dob
                THEN RETURN -1;
            ELSIF dob < contact_person.dob
                THEN RETURN 1;
            ELSE RETURN 0;
        END IF;
     END;
END;
```



```
Type PERSON_TYP compiled


Type Body PERSON_TYP compiled
```

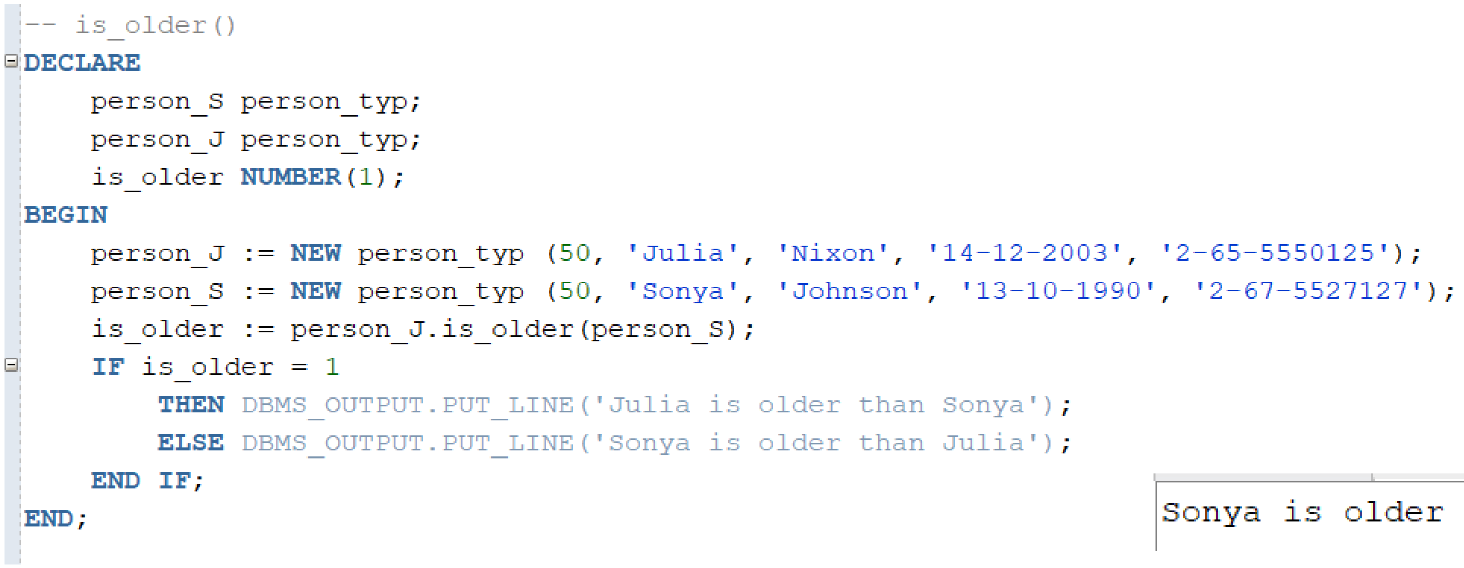
Метод сравнения ORDER

```sql
CREATE TABLE contacts (
     contact person_typ,
     contact_date DATE );

-- insert 2 contacts;
INSERT INTO contacts VALUES (
person_typ (50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125'), '24-05-2023' );
INSERT INTO contacts VALUES (
person_typ (49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225'), '24-05-2023' );
INSERT INTO contacts VALUES (
person_typ (50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127'), '24-05-2023' );
COMMIT;
```

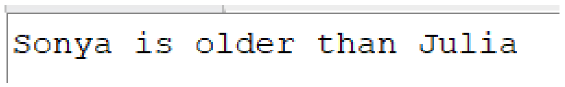
```sql
-- order by
SELECT c.contact
FROM contacts c
ORDER BY c.contact;
```

```
CONTACT(ID_NO, FIRST_NAME, LAST_NAME, DOB, PHONE)
--------------------------------------------------------------
PERSON_TYP(50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125')
PERSON_TYP(49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225')
PERSON_TYP(50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127')
```

Метод сравнения ORDER



```
-- is_older()
DECLARE
    person_S person_typ;
    person_J person_typ;
    is_older NUMBER(1);
BEGIN
    person_J := NEW person_typ (50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125');
    person_S := NEW person_typ (50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127');
    is_older := person_J.is_older(person_S);
    IF is_older = 1
        THEN DBMS_OUTPUT.PUT_LINE('Julia is older than Sonya');
        ELSE DBMS_OUTPUT.PUT_LINE('Sonya is older than Julia');
    END IF;
END;
```

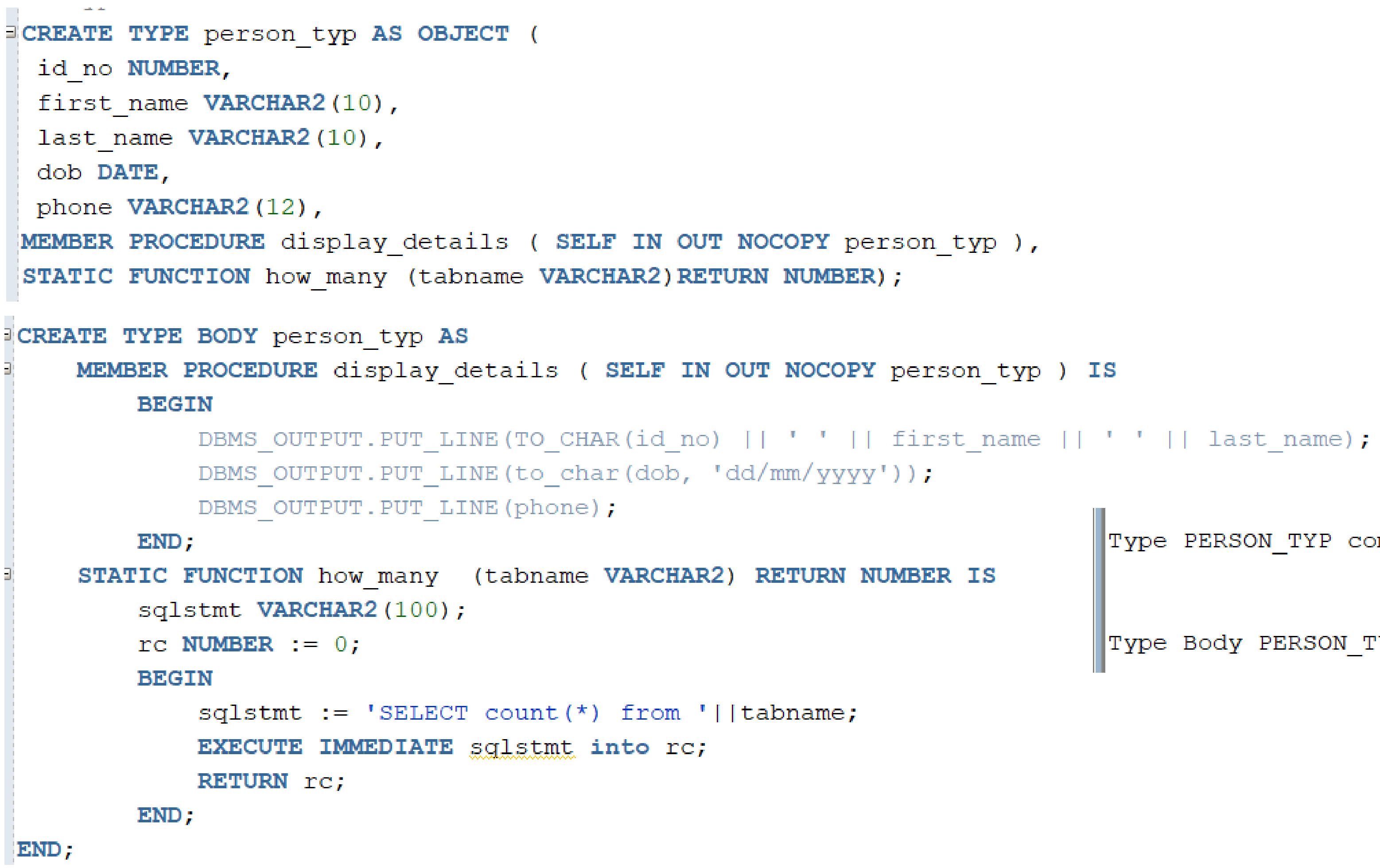


```
Sonya is older than Julia
```

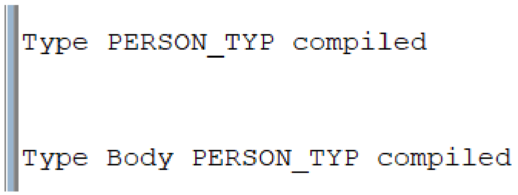
Статические методы

- Статические методы относятся к типу в целом, а не к экземпляру

Статические методы

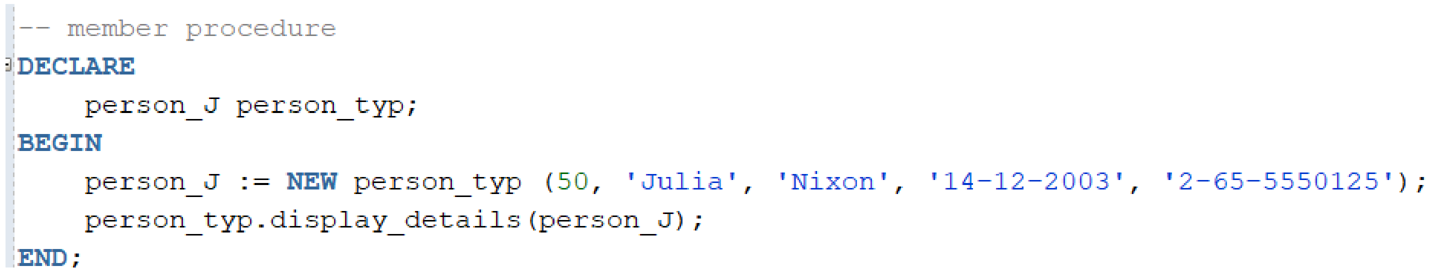


```
CREATE TYPE person_typ AS OBJECT (
 id_no NUMBER,
 first_name VARCHAR2(10),
 last_name VARCHAR2(10),
 dob DATE,
 phone VARCHAR2(12),
MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY person_typ ),
STATIC FUNCTION how_many (tabname VARCHAR2)RETURN NUMBER);

CREATE TYPE BODY person_typ AS
    MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY person_typ ) IS
        BEGIN
            DBMS_OUTPUT.PUT_LINE(TO_CHAR(id_no) || ' ' || first_name || ' ' || last_name);
            DBMS_OUTPUT.PUT_LINE(to_char(dob, 'dd/mm/yyyy'));
            DBMS_OUTPUT.PUT_LINE(phone);
        END;
    STATIC FUNCTION how_many  (tabname VARCHAR2) RETURN NUMBER IS
        sqlstmt VARCHAR2(100);
        rc NUMBER := 0;
        BEGIN
            sqlstmt := 'SELECT count(*) from '||tabname;
            EXECUTE IMMEDIATE sqlstmt into rc;
            RETURN rc;
        END;
END;
```

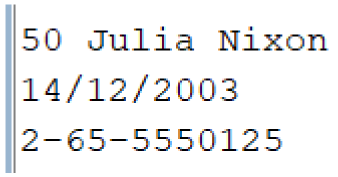


```
Type PERSON_TYP compiled


Type Body PERSON_TYP compiled
```

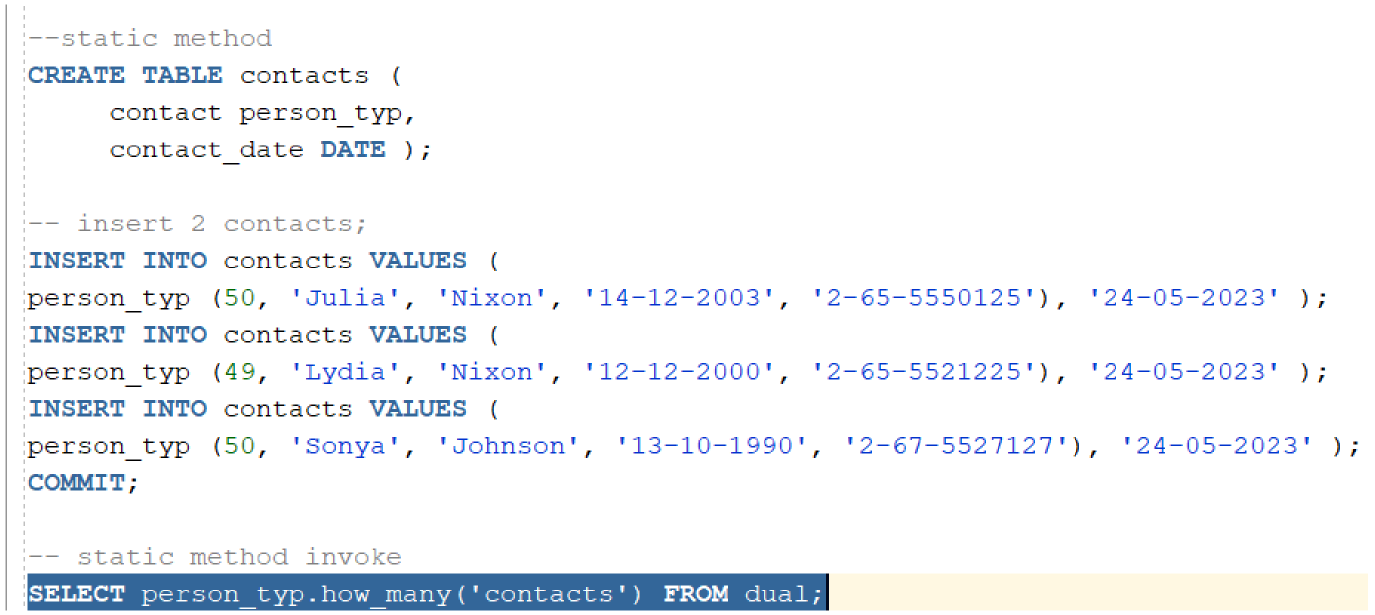
Статические методы



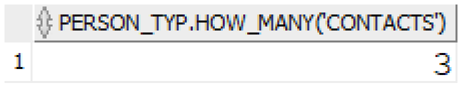
```
-- member procedure
DECLARE
    person_J person_typ;
BEGIN
    person_J := NEW person_typ (50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125');
    person_typ.display_details(person_J);
END;
```



```
50 Julia Nixon
14/12/2003
2-65-5550125
```



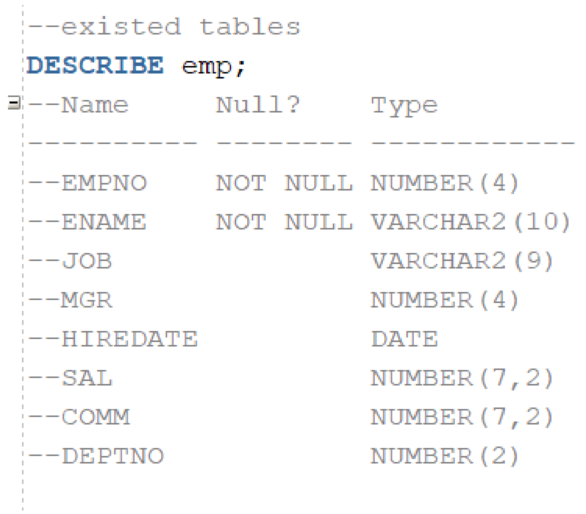
```
--static method
CREATE TABLE contacts (
     contact person_typ,
     contact_date DATE );

-- insert 2 contacts;
INSERT INTO contacts VALUES (
person_typ (50, 'Julia', 'Nixon', '14-12-2003', '2-65-5550125'), '24-05-2023' );
INSERT INTO contacts VALUES (
person_typ (49, 'Lydia', 'Nixon', '12-12-2000', '2-65-5521225'), '24-05-2023' );
INSERT INTO contacts VALUES (
person_typ (50, 'Sonya', 'Johnson', '13-10-1990', '2-67-5527127'), '24-05-2023' );
COMMIT;

-- static method invoke
SELECT person_typ.how_many('contacts') FROM dual;
```



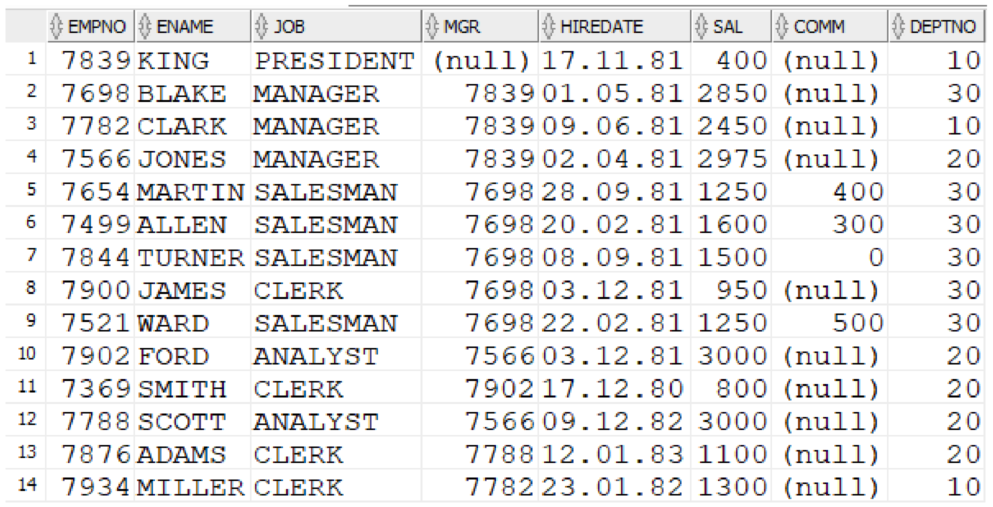
	PERSON_TYP.HOW_MANY('CONTACTS')
1	3

Объектные представления

- Цель – создание аналога ORM:
- На основе уже существующих таблиц создать объекты

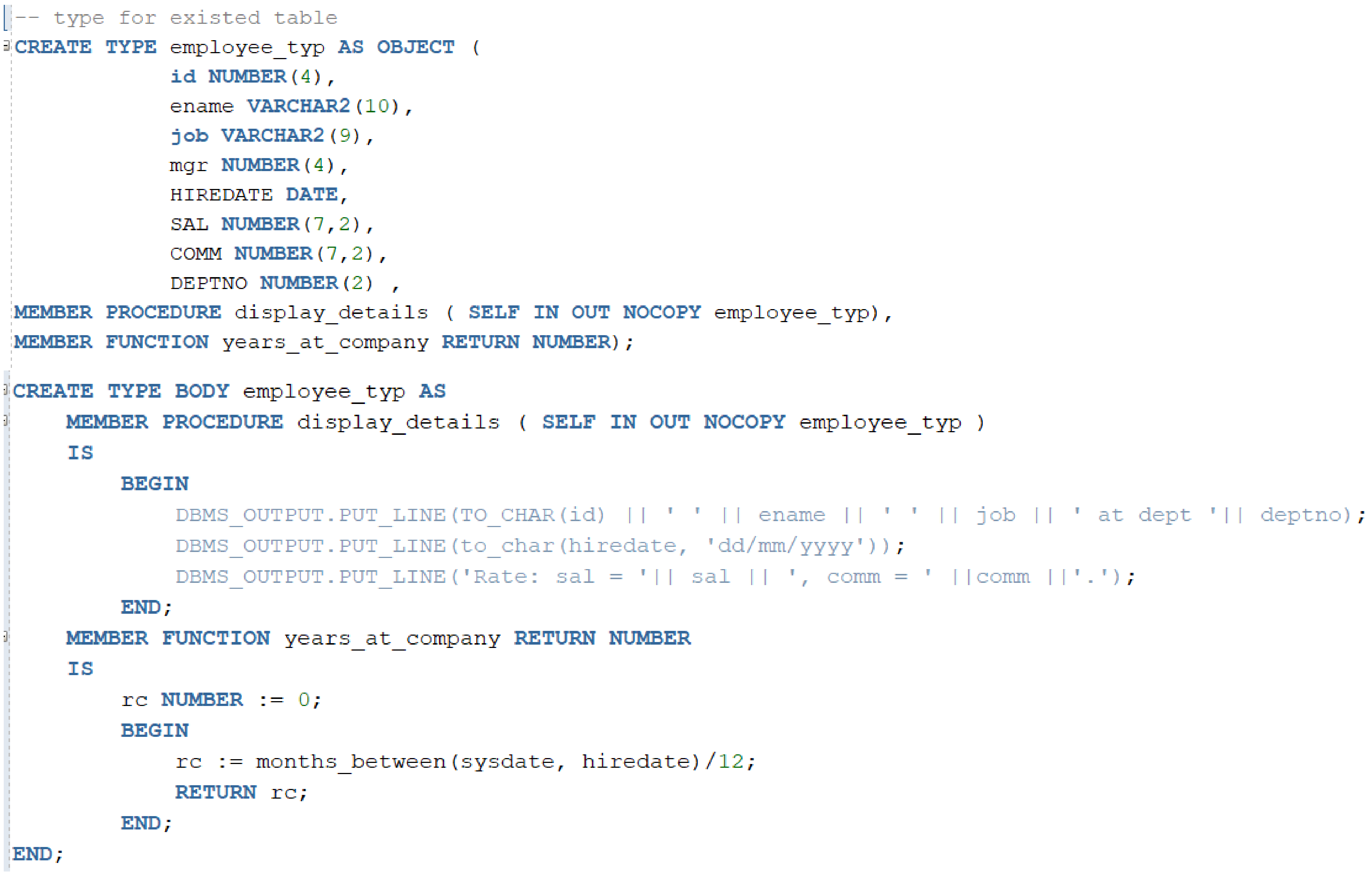


```
--existed tables
DESCRIBE emp;
--Name       Null?     Type
---------- -------- ------------
--EMPNO      NOT NULL NUMBER(4)
--ENAME      NOT NULL VARCHAR2(10)
--JOB                 VARCHAR2(9)
--MGR                 NUMBER(4)
--HIREDATE            DATE
--SAL                 NUMBER(7,2)
--COMM                NUMBER(7,2)
--DEPTNO              NUMBER(2)
```



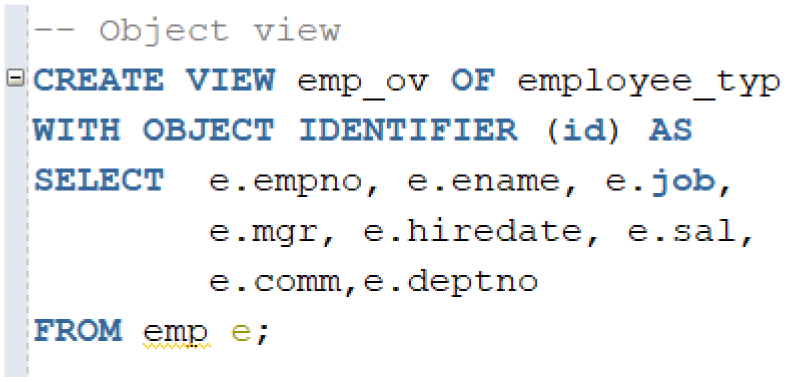
	EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
1	7839	KING	PRESIDENT	(null)	17.11.81	400	(null)	10
2	7698	BLAKE	MANAGER	7839	01.05.81	2850	(null)	30
3	7782	CLARK	MANAGER	7839	09.06.81	2450	(null)	10
4	7566	JONES	MANAGER	7839	02.04.81	2975	(null)	20
5	7654	MARTIN	SALESMAN	7698	28.09.81	1250	400	30
6	7499	ALLEN	SALESMAN	7698	20.02.81	1600	300	30
7	7844	TURNER	SALESMAN	7698	08.09.81	1500	0	30
8	7900	JAMES	CLERK	7698	03.12.81	950	(null)	30
9	7521	WARD	SALESMAN	7698	22.02.81	1250	500	30
10	7902	FORD	ANALYST	7566	03.12.81	3000	(null)	20
11	7369	SMITH	CLERK	7902	17.12.80	800	(null)	20
12	7788	SCOTT	ANALYST	7566	09.12.82	3000	(null)	20
13	7876	ADAMS	CLERK	7788	12.01.83	1100	(null)	20
14	7934	MILLER	CLERK	7782	23.01.82	1300	(null)	10

Объектные представления

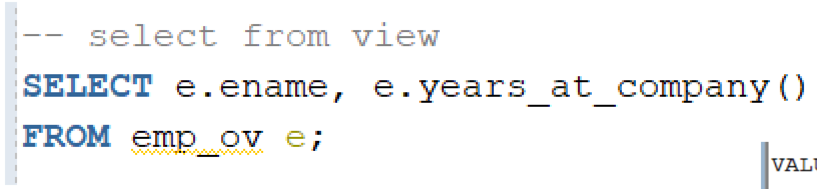


```
-- type for existed table
CREATE TYPE employee_typ AS OBJECT (
            id NUMBER(4),
            ename VARCHAR2(10),
            job VARCHAR2(9),
            mgr NUMBER(4),
            HIREDATE DATE,
            SAL NUMBER(7,2),
            COMM NUMBER(7,2),
            DEPTNO NUMBER(2)  ,
MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY employee_typ),
MEMBER FUNCTION years_at_company RETURN NUMBER);

CREATE TYPE BODY employee_typ AS
    MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY employee_typ )
    IS
        BEGIN
            DBMS_OUTPUT.PUT_LINE(TO_CHAR(id) || ' ' || ename || ' ' || job || ' at dept '|| deptno);
            DBMS_OUTPUT.PUT_LINE(to_char(hiredate, 'dd/mm/yyyy'));
            DBMS_OUTPUT.PUT_LINE('Rate: sal = '|| sal || ', comm = ' ||comm ||'.');
        END;
    MEMBER FUNCTION years_at_company RETURN NUMBER
    IS
        rc NUMBER := 0;
        BEGIN
            rc := months_between(sysdate, hiredate)/12;
            RETURN rc;
        END;
END;
```

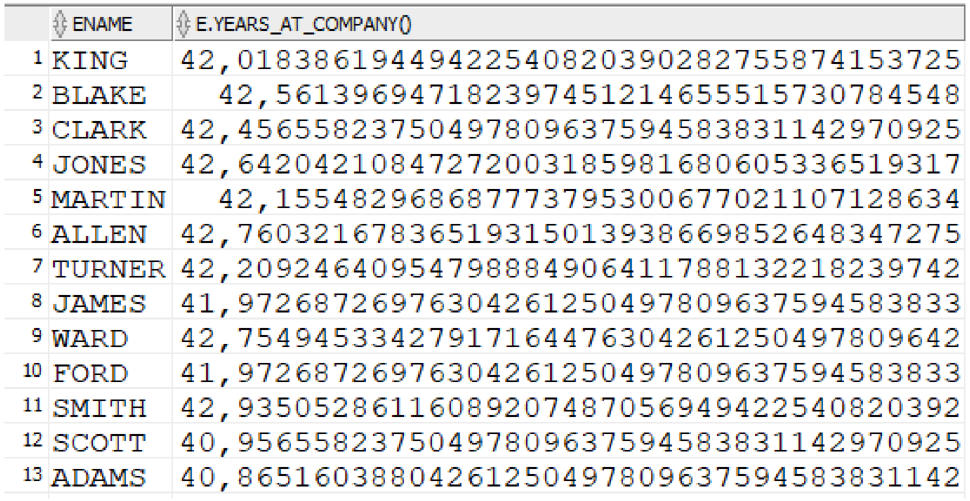
Объектные представления



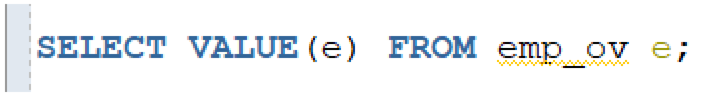
```
-- Object view
CREATE VIEW emp_ov OF employee_typ
WITH OBJECT IDENTIFIER (id) AS
SELECT  e.empno, e.ename, e.job,
        e.mgr, e.hiredate, e.sal,
        e.comm,e.deptno
FROM emp e;
```



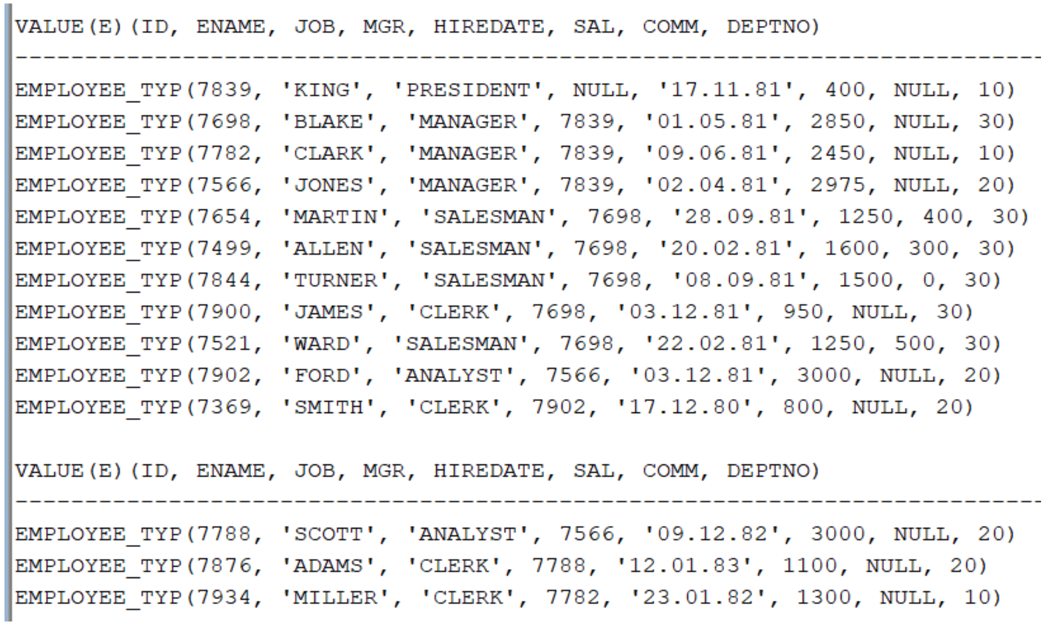
```
-- select from view
SELECT e.ename, e.years_at_company()
FROM emp_ov e;
```



	ENAME	E.YEARS_AT_COMPANY()
1	KING	42,01838619449422540820390282755874153725
2	BLAKE	42,561396947182397451214655515730784548
3	CLARK	42,45655823750497809637594583831142970925
4	JONES	42,64204210847272003185981680605336519317
5	MARTIN	42,155482968687773795300677021107128634
6	ALLEN	42,76032167836519315013938669852648347275
7	TURNER	42,20924640954798884906411788132218239742
8	JAMES	41,97268726976304261250497809637594583833
9	WARD	42,75494533427917164476304261250497809642
10	FORD	41,97268726976304261250497809637594583833
11	SMITH	42,93505286116089207487056949422540820392
12	SCOTT	40,95655823750497809637594583831142970925
13	ADAMS	40,86516038804261250497809637594583831142



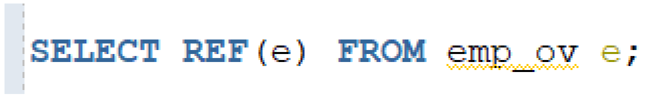
```
SELECT VALUE(e) FROM emp_ov e;
```



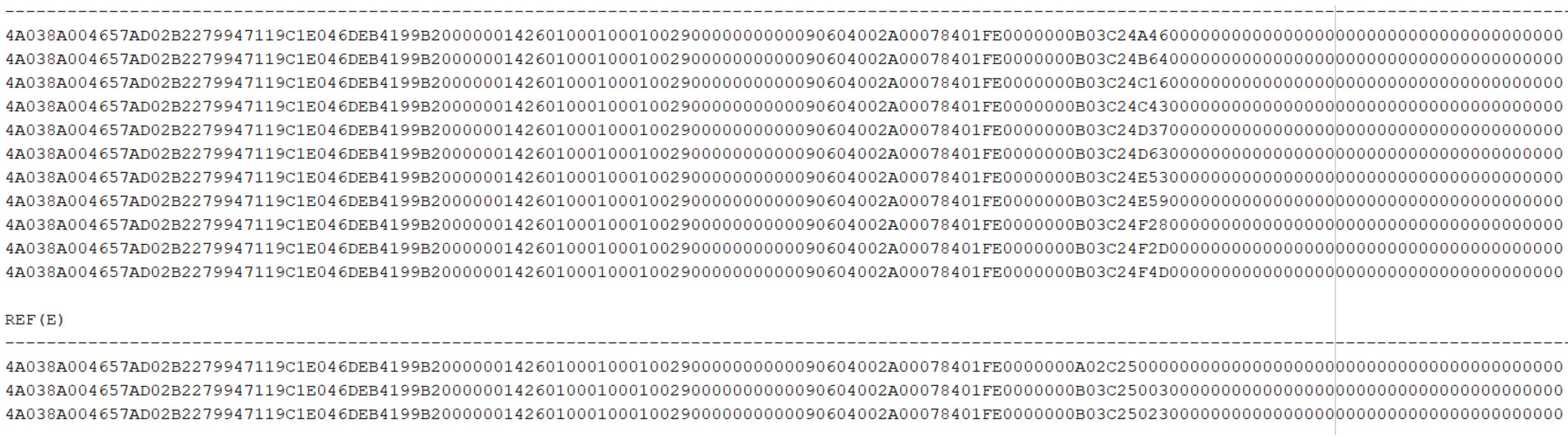
```
VALUE(E)(ID, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO)
--------------------------------------------------------------------------
EMPLOYEE_TYP(7839, 'KING', 'PRESIDENT', NULL, '17.11.81', 400, NULL, 10)
EMPLOYEE_TYP(7698, 'BLAKE', 'MANAGER', 7839, '01.05.81', 2850, NULL, 30)
EMPLOYEE_TYP(7782, 'CLARK', 'MANAGER', 7839, '09.06.81', 2450, NULL, 10)
EMPLOYEE_TYP(7566, 'JONES', 'MANAGER', 7839, '02.04.81', 2975, NULL, 20)
EMPLOYEE_TYP(7654, 'MARTIN', 'SALESMAN', 7698, '28.09.81', 1250, 400, 30)
EMPLOYEE_TYP(7499, 'ALLEN', 'SALESMAN', 7698, '20.02.81', 1600, 300, 30)
EMPLOYEE_TYP(7844, 'TURNER', 'SALESMAN', 7698, '08.09.81', 1500, 0, 30)
EMPLOYEE_TYP(7900, 'JAMES', 'CLERK', 7698, '03.12.81', 950, NULL, 30)
EMPLOYEE_TYP(7521, 'WARD', 'SALESMAN', 7698, '22.02.81', 1250, 500, 30)
EMPLOYEE_TYP(7902, 'FORD', 'ANALYST', 7566, '03.12.81', 3000, NULL, 20)
EMPLOYEE_TYP(7369, 'SMITH', 'CLERK', 7902, '17.12.80', 800, NULL, 20)

VALUE(E)(ID, ENAME, JOB, MGR, HIREDATE, SAL, COMM, DEPTNO)
--------------------------------------------------------------------------
EMPLOYEE_TYP(7788, 'SCOTT', 'ANALYST', 7566, '09.12.82', 3000, NULL, 20)
EMPLOYEE_TYP(7876, 'ADAMS', 'CLERK', 7788, '12.01.83', 1100, NULL, 20)
EMPLOYEE_TYP(7934, 'MILLER', 'CLERK', 7782, '23.01.82', 1300, NULL, 10)
```

Объектные представления

- Все строки таблиц имеют объектные ссылки



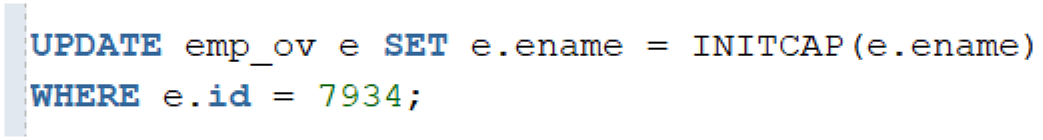
```sql
SELECT REF(e) FROM emp_ov e;
```



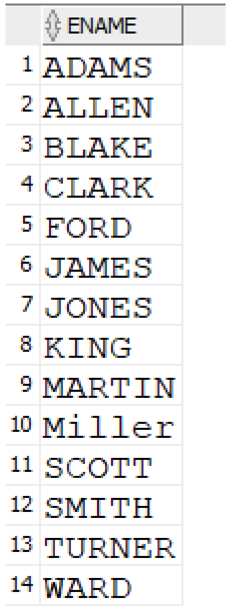
```
-------------------------------------------------------------------------------------------------------------------------------------------------------
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24A4600000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24B6400000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24C1600000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24C4300000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24D3700000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24D6300000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24E5300000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24E5900000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24F2800000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24F2D00000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C24F4D00000000000000000000000000000000000000

REF(E)
-------------------------------------------------------------------------------------------------------------------------------------------------------
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000A02C2500000000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C2500300000000000000000000000000000000000000
4A038A004657AD02B2279947119C1E046DEB4199B200000001426010001000100290000000000090604002A00078401FE0000000B03C2502300000000000000000000000000000000000000
```

Объектные представления

- DML операции производятся так же



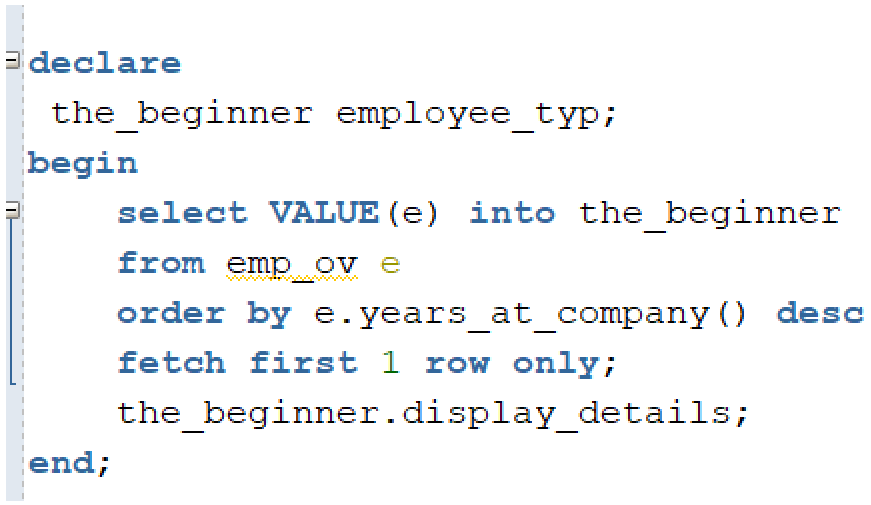
```
UPDATE emp_ov e SET e.ename = INITCAP(e.ename)
WHERE e.id = 7934;
```



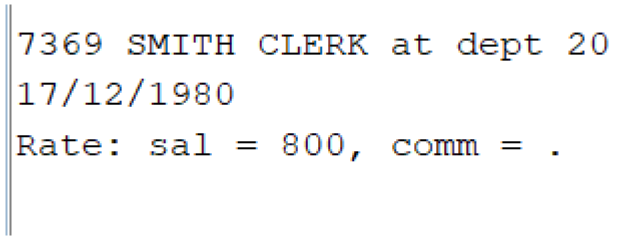
	ENAME
1	ADAMS
2	ALLEN
3	BLAKE
4	CLARK
5	FORD
6	JAMES
7	JONES
8	KING
9	MARTIN
10	Miller
11	SCOTT
12	SMITH
13	TURNER
14	WARD

Объектные представления

- В PL/SQL объектные представления также могут использоваться

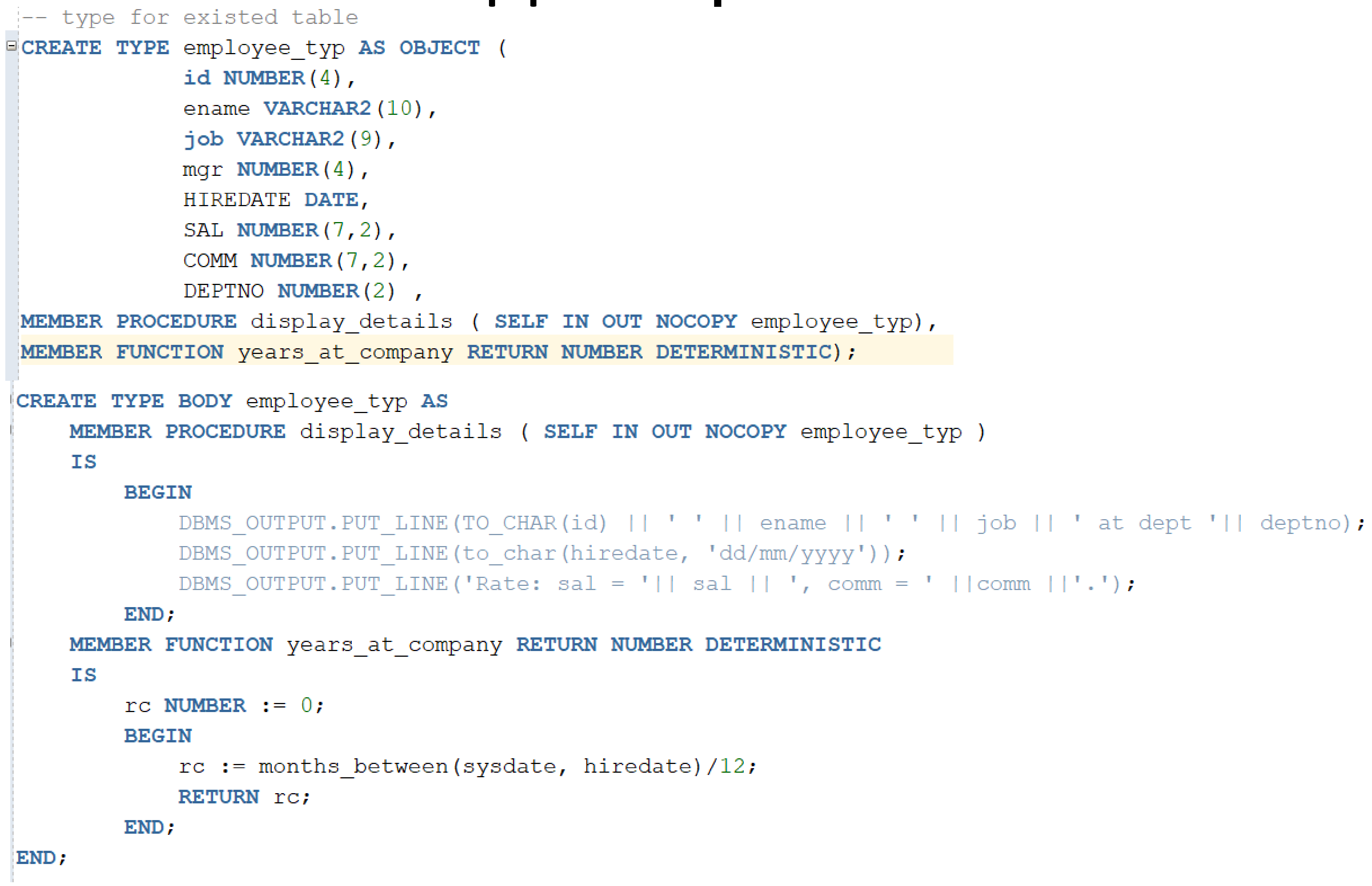


```
declare
 the_beginner employee_typ;
begin
    select VALUE(e) into the_beginner
    from emp_ov e
    order by e.years_at_company() desc
    fetch first 1 row only;
    the_beginner.display_details;
end;
```



```
7369 SMITH CLERK at dept 20
17/12/1980
Rate: sal = 800, comm = .
```

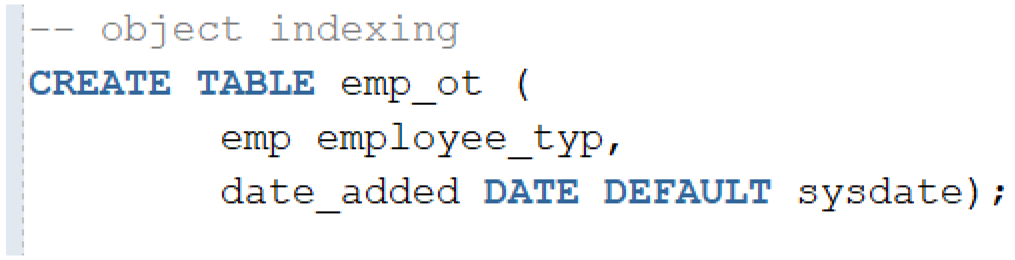
Индексирование



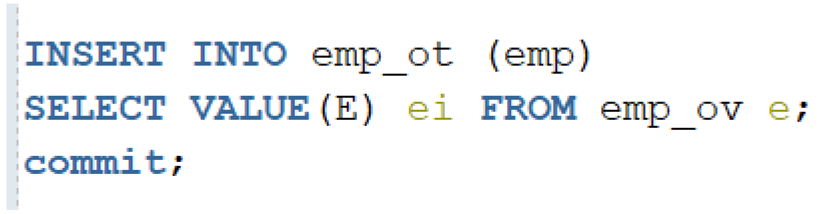
```
-- type for existed table
CREATE TYPE employee_typ AS OBJECT (
            id NUMBER(4),
            ename VARCHAR2(10),
            job VARCHAR2(9),
            mgr NUMBER(4),
            HIREDATE DATE,
            SAL NUMBER(7,2),
            COMM NUMBER(7,2),
            DEPTNO NUMBER(2)  ,
MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY employee_typ),
MEMBER FUNCTION years_at_company RETURN NUMBER DETERMINISTIC);

CREATE TYPE BODY employee_typ AS
    MEMBER PROCEDURE display_details ( SELF IN OUT NOCOPY employee_typ )
    IS
        BEGIN
            DBMS_OUTPUT.PUT_LINE(TO_CHAR(id) || ' ' || ename || ' ' || job || ' at dept '|| deptno);
            DBMS_OUTPUT.PUT_LINE(to_char(hiredate, 'dd/mm/yyyy'));
            DBMS_OUTPUT.PUT_LINE('Rate: sal = '|| sal || ', comm = ' ||comm ||'.');
        END;
    MEMBER FUNCTION years_at_company RETURN NUMBER DETERMINISTIC
    IS
        rc NUMBER := 0;
        BEGIN
            rc := months_between(sysdate, hiredate)/12;
            RETURN rc;
        END;
END;
```

Индексирование

- Индексирование может быть реализовано по атрибутам или методам

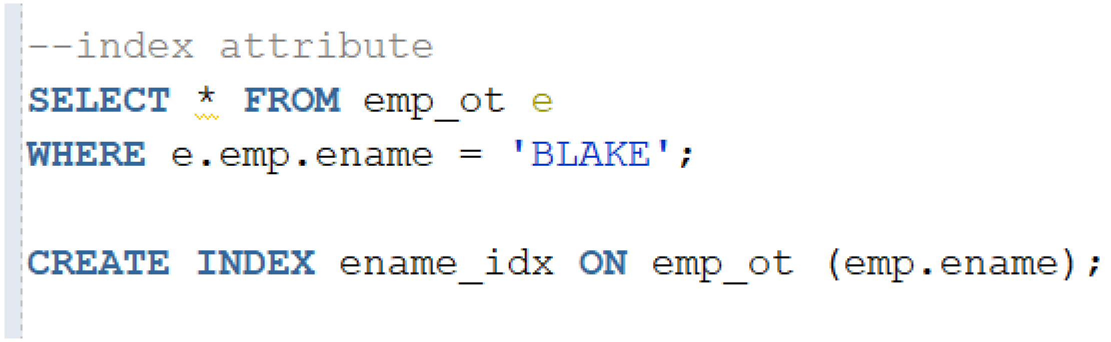


```
-- object indexing
CREATE TABLE emp_ot (
        emp employee_typ,
        date_added DATE DEFAULT sysdate);
```

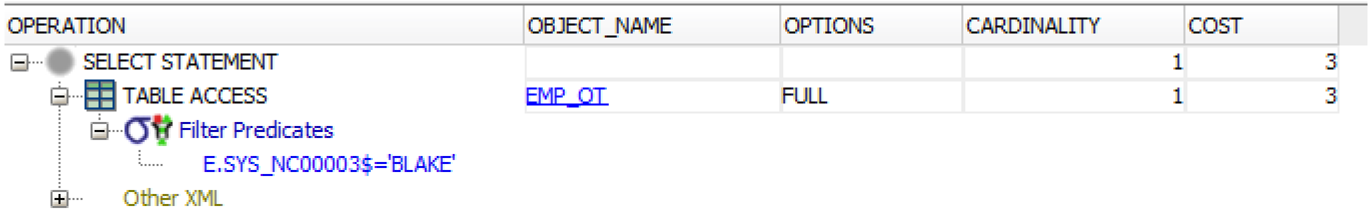


```
INSERT INTO emp_ot (emp)
SELECT VALUE(E) ei FROM emp_ov e;
commit;
```

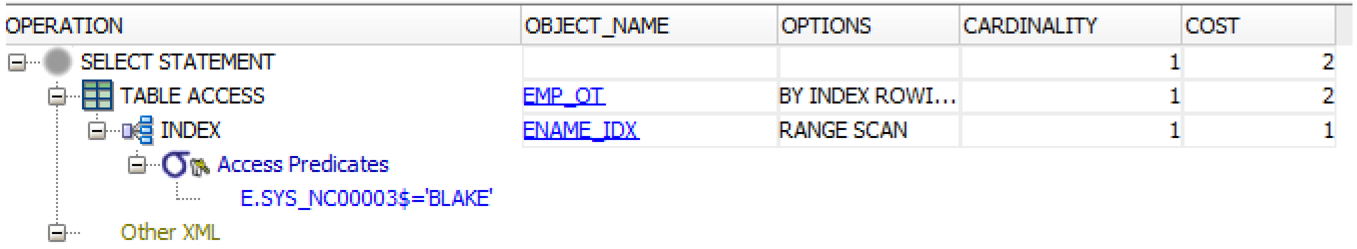
Индексирование по атрибуту



```
--index attribute
SELECT * FROM emp_ot e
WHERE e.emp.ename = 'BLAKE';

CREATE INDEX ename_idx ON emp_ot (emp.ename);
```

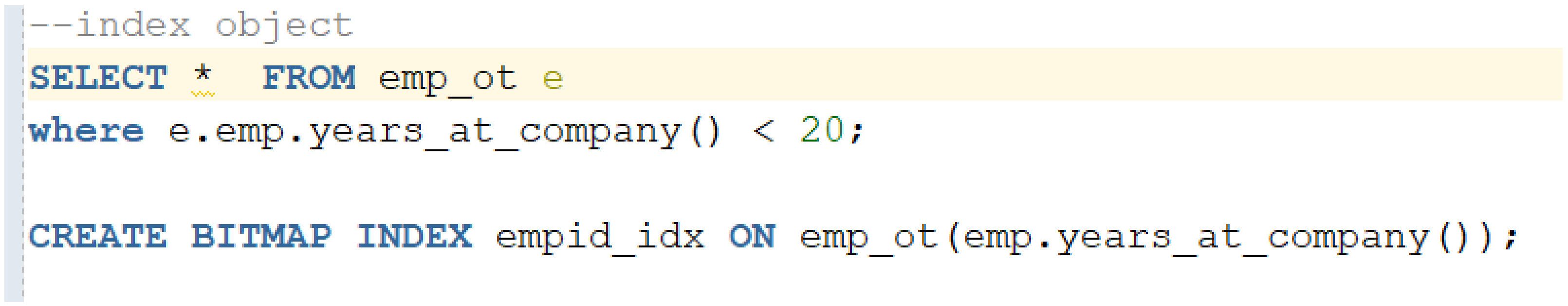


OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			1	3
TABLE ACCESS	EMP_OT	FULL	1	3
Filter Predicates				
E.SYS_NC00003$='BLAKE'				
Other XML				

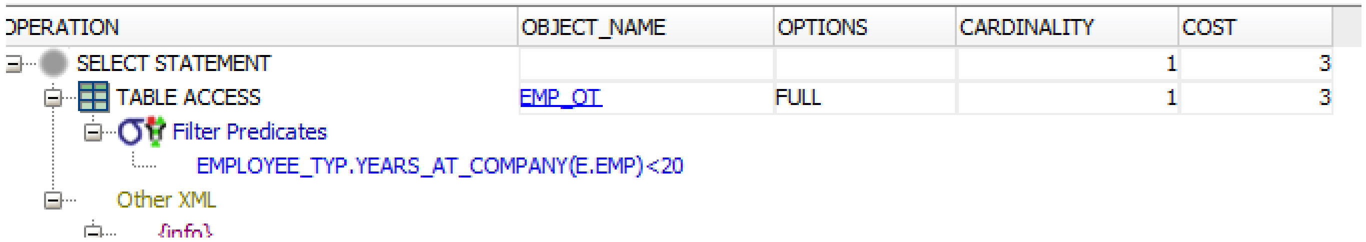


OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			1	2
TABLE ACCESS	EMP_OT	BY INDEX ROWI...	1	2
INDEX	ENAME_IDX	RANGE SCAN	1	1
Access Predicates				
E.SYS_NC00003$='BLAKE'				
Other XML				

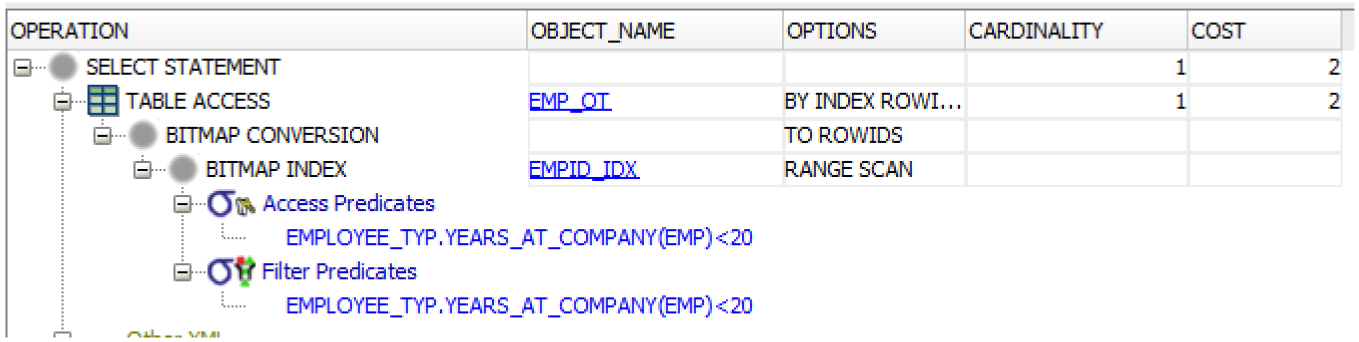
Индексирование по методу



```
--index object
SELECT *  FROM emp_ot e
where e.emp.years_at_company() < 20;

CREATE BITMAP INDEX empid_idx ON emp_ot(emp.years_at_company());
```



```
PERATION                                    OBJECT_NAME   OPTIONS   CARDINALITY   COST
⊟ SELECT STATEMENT                                                            1      3
  ⊟ TABLE ACCESS                            EMP_OT        FULL                1      3
    ⊟ Filter Predicates
        EMPLOYEE_TYP.YEARS_AT_COMPANY(E.EMP)<20
  ⊟ Other XML
    ⊟ {info}
```



```
OPERATION                                   OBJECT_NAME   OPTIONS            CARDINALITY   COST
⊟ SELECT STATEMENT                                                                     1      2
  ⊟ TABLE ACCESS                            EMP_OT        BY INDEX ROWI...             1      2
    ⊟ BITMAP CONVERSION                                   TO ROWIDS
      ⊟ BITMAP INDEX                        EMPID_IDX     RANGE SCAN
        ⊟ Access Predicates
            EMPLOYEE_TYP.YEARS_AT_COMPANY(EMP)<20
        ⊟ Filter Predicates
            EMPLOYEE_TYP.YEARS_AT_COMPANY(EMP)<20
```

Вопросы?