IZ - Laboratory to the lecture:
Oracle Database - programming
Task list No. 4

Task 47. Define the schema of the cats' database in a relational-object form. Propose and define examples of methods for each object type. Reference bindings should be defined using reference types. Fill relations from such a scheme with data from the reality of cats. Then, perform sample SQL queries using references (as joins), subqueries, grouping, and methods defined within types. One own example of task and query resolving this task should be provided for each mechanism (reference, subquery, grouping). Additionally, under this new database scheme, two selected non-trivial tasks from list 2 and two from list 3 should be carried out.

Task 48.*[1] Define the "overlay", in the form of object views, on the relational database of cats. Object types corresponding to relations are to contain examples of methods (they can be methods from Task 47). Model all reference associations using OID identifier and the MAKE_REF function. Perform for this database all SQL queries and PL/SQL blocks implemented as part of the task. 47.

Task 49. In connection with Poland's accession to the European Union, detailed records of hunted and consumed mice were necessary. Therefore, we need records for both the cat that hunted the mouse (along with the date of hunting) and the cat that ate it (together with the date of "payout"). In addition, the mouse's weight has become important (this weight must meet the EU standard (please set this standard). Worst of all, the data had to be completed backward, starting from January 1, 2004. Unfortunately, as it sometimes happens, there was a "slight" delay in implementing the recording program of hunted and eaten mice. By a strange coincidence, this record was only possible the day before the date of returning the current list of tasks.

Write a block (s) that will conduct these records, so:

a) Modify the database schema defining a new relation named Mice having the attributes: mouse_number (primary key), hunter (foreign key), eater (foreign key), mouse_weight, catch_date, release_date (always last Wednesday of the month),
b) Fill the Mice relation by artificially generated data, starting from January 1, 2004, to the day before this list's delivery date. The number of mice entered, caught in one month, is to be consistent with the number of mice that cats received as part of the "payout" that month (together with extra mice). When completing the data, it should be assumed that each cat can catch the monthly number of mice equal to the number of mice consumed. This number is an average per month for each cat (use any surpluses associated with rounding). The catch dates for the mice are to be spread evenly throughout the month. Each mouse's issue date is the last Wednesday of each month.

The solution should use internal dynamic SQL (creating a new relation) and bulk binding (filling the relation with generated data). The actual data about hunted mice should be saved from the current date. Preparing a procedure that will allow for writing mice caught during the day by a particular cat[2] in Mice relation is necessary. Another procedure for monthly cat' payments, considering their hierarchy's order, is also needed. The bulk binding should be used for both procedures.

[1] The list can be resolved and presented without this task, but at a loss of 2.5 point.

[2] data on mice caught during the day are available in an individual relation, which is owned by each cat