IZ - Laboratory to the lecture:
Oracle Database - programming
Task list No. 3

Task 34. Write a PL/SQL block that selects cats performing the function given on the keyboard. The only effect of block operation is to be a message informing whether or not a cat performing the given function has been found (if a cat is found, display the name of the appropriate function).

```
FUNCTION – FLAX
FLAX - function not found

FUNCTION - NICE
NICE - function found more than once

FUNCTION – BOSS
BOSS - function found once
```

Task 35. Write a PL/SQL block that displays the following information about the cat with a nickname entered from the keyboard (depending on the actual data):

- 'total annual mice ration> 700'
- 'name contains the letter A'
- 'May is the month of joining the herd'
- 'does not match the criteria'.

The above information is listed according to its hierarchy of importance. Precede each entry with the cat's name.

```
NICKNAME – TIGER
MRUCZEK: total annual mice ration > 700

NICKNAME – EAR
LATKA: name contains the letter A

NICKNAME – SMALL
DUDEK: May is the month of joining the herd

NICKNAME –NICKNAME ZERO
LUCEK: does not match the criteria
```

Task 36. Due to the high efficiency in hunting mice, SZEFUNIO decided to reward his subordinates. So he announced that he increases the individual mice ration of each cat by 10%, in order from the cat with the lowest ration to the cat with the highest ration. He also determined that the process of increasing the mice rations should be completed when the sum of all rations of all cats exceeds 1050. If for a cat, the mouse ration after the increase exceeds the maximum value due to the function (max_mice from the Functions relation), the mouse allocation after the increase should be equal to this value (max_mice). Write a PL/SQL block with a cursor that performs this task. The block is to operate until the sum of all rations actually exceeds 1050 (the number of "increase cycles" may be greater than 1 and thus the individual increase may be greater than 10%). Display the sum of mouse rations on the screen after completing the task along with the number of modifications in the Cats relation. Finally, roll back all changes.

```
Total ration 1057  Changes – 30
```

Task 37. Write a block that selects five cats with the highest total mouse allocation using FOR loop with cursor. Display the result on the screen.

```
No  Nickname    Eats
--------------------
1   TIGER       136
2   BALD         93
3   ZOMBIES      88
4   LOLA         72
5   CAKE         67
```

Task 38. Write a block that will implement version a. or b. of task 19 in a universal way (without having to take into account knowledge about the depth of the tree). The input value is to be the maximum number of supervisors to display.

```
NUMBER_OF_SUPERIORS - 5

name           |  chief 1         |  chief 2         |  chief 3
-------------- --- ------------- --- ------------- --- -------------
MICKA          |  MRUCZEK         |                  |
LUCEK          |  PUNIA           |  KOREK           |  MRUCZEK
SONIA          |  KOREK           |  MRUCZEK         |
LATKA          |  PUCEK           |  MRUCZEK         |
DUDEK          |  PUCEK           |  MRUCZEK         |
RUDA           |  MRUCZEK         |                  |
BELA           |  BOLEK           |  MRUCZEK         |

NUMBER_OF_SUPERIORS - 2

name           |  chief 1         |  chief 2
-------------- --- ------------- --- -------------
MICKA          |  MRUCZEK         |
LUCEK          |  PUNIA           |  KOREK
SONIA          |  KOREK           |  MRUCZEK
LATKA          |  PUCEK           |  MRUCZEK
DUDEK          |  PUCEK           |  MRUCZEK
RUDA           |  MRUCZEK         |
BELA           |  BOLEK           |  MRUCZEK
```

Task 39. Write a PL/SQL block loading three parameters representing the band number, band name and hunting site. The script is to prevent entering existing parameter values by handling the appropriate exceptions. Entering the band number <= 0 is also an exceptional situation. In the event of an exceptional situation, an appropriate message should be displayed on the screen. If the parameters are correct, create a new band in the Bands relation. The change should be roll backed at the end.

```
BAND_NO - 2
BAND_NAME - BLACK KNIGHTS
BAND_SITE - FIELD
2, BLACK KNIGHTS, FIELD: already exists

BAND_NO - 6
BAND_NAME - SUPERIORS
BAND_SITE - CELLAR
SUPERIORS: already exists
```

```
BAND_NO - 7
BAND_NAME - PINTO HUNTERS
BAND_SITE -HILLOCK
PINTO HUNTERS, HILLOCK: already exists

BAND_NO - 0
BAND_NAME - NEW
BAND_SITE - CELLAR
Band No <=0 !

BAND_NO - 6
BAND_NAME - NEW
BAND_SITE - CELLAR
Band NEW created
```

Task 40. Convert block from Task 39 into a procedure named new_band, stored in the database.

Task 41. Define a trigger that ensures that the number of the new band will always be 1 greater than the highest number of the existing band. Check the result of the trigger operation using the procedure from Task 40.

```
EXEC new_band(7,'NEW','CELLAR');
Band NEW created

SELECT * FROM Bands;
```

BAND_NO	NAME	SITE	BAND_CHIEF
6	NEW	CELLAR	
1	SUPERIORS	WHOLE AREA	TIGER
2	BLACK KNIGHTS	FIELD	BALD
3	WHITE HUNTERS	ORCHARD	ZOMBIES
4	PINTO HUNTERS	HILLOCK	REEF
5	ROCKERSI	FARM	

Task 42. Female cats with function Nice decided to take care of their maters. So they hired IT to enter the virus into the Tiger's system. Now, with every attempt to change the mice ration for Nice for a plus (for minus there is no question at all) by a value less than 10% of the mice ration of Tiger, the regret of Nice caused by this must be comforted by an increase of their ration by this value as well as an increase in extra mice by 5. Additionally, Tiger is to be punished by the loss of these 10%. However, if the increase is satisfactory for Nice, the Tiger ration extra of mice is to increase by 5.

Propose two solutions of this task that will bypass the basic limitation for a row trigger activated by a DML command, ie the inability to read or change the relation on which the operation (DML command) activates this trigger. In the first (classic) solution, use several triggers and memory in the form of a package specification dedicated to the task, in the second, use the COMPOUND trigger.

Provide examples of how triggers work, and then remove any changes they make.

Task 43. Write a block that will carry out Task 33 in a universal way (without having to take into account knowledge of the functions performed by cats).

Task 44. The tiger was concerned about the inexplicable reduction in mice supplies. So he decided to introduce a head tax that would top up the pantry. So he ordered that every cat is obliged to donate 5% (rounded up) of their total mice income. In addition, from what will remain:

- cats without subordinates give two mice for their incompetence during
- applying for promotion,
- cats without enemies give one mouse for too much agreeableness,
- cats pay an additional tax, the form of which is determined by the contractor of the task.

Write a function whose parameter is the cat's nickname, determining the cat's head tax due. This function together with the procedure from the Task 40 should be included in the one package and then used to determine the tax for all cats.

–

Task 45. The Tiger noticed strange changes in the value of his private mice ration (see Task 42). He was not worried about the positive changes but those in the negative were, in his opinion, inadmissible. So he motivated one of his spies to act and thanks to this he discovered Nice's evil practices (Task 42). So he instructed his computer scientist to construct a mechanism which would write in the Extra_additions relation (see Lectures - part 2) -10 mice for each Nice (minus ten) when changing to plus any of the Nice mice ration made by an operator other than he himself. Propose such a mechanism, in lieu of the computer scientist. In the solution use the LOGIN_USER function (which returns the user name activating the trigger) and elements of dynamic SQL.

Task 46. Write a trigger which will prevent the cat from being allocated a number of mice outside the range (min_mice, max_mice) specified for each function in the Functions relation. Each attempt to go beyond the applicable range is to be additionally monitored in a separate relation (who, when, which cat, which operation).