Alex Rogoziński 280522

Programowanie Baz Danych

Laboratorium

Lista 4 - Raport

Spis treści

Zadania - bloki

Zadanie 1

A.

T-SQL

```
-------------------------------------------------------------------------------------------
-- TASK 7
-------------------------------------------------------------------------------------------
-- a)
DECLARE
    @przelozeni_num INT = 4,
    @dynamic_sql    NVARCHAR(MAX),
    @index          INT = 1,
    @index_nvarchar NVARCHAR(3);

-- Poczatek kwerendy
SET @dynamic_sql = 'SELECT k0.imie AS Imie';

-- Wyznaczenie danych do wypisania
WHILE @index <= @przelozeni_num
BEGIN
    SET @index_nvarchar = CAST(@index AS NVARCHAR);
    SET @dynamic_sql += N', ISNULL(k' + @index_nvarchar + N'.imie, '''') AS [Szef ' +
@index_nvarchar + N']';
    SET @index += 1;
END;

SET @dynamic_sql += N' FROM Kocury k0';

SET @index = 1;

-- Wyznaczenie zlaczen
WHILE @index <= @przelozeni_num
BEGIN
    SET @index_nvarchar = CAST(@index AS NVARCHAR);
    SET @dynamic_sql += N' LEFT JOIN Kocury k' + @index_nvarchar + N' ON k' + @index_nvarchar
                     + N'.pseudo = k' + CAST(@index - 1 AS NVARCHAR) + N'.szef';

    SET @index += 1;
END;

-- Dodanie warunkow poczatkowych i porzadku
SET @dynamic_sql += N'
    WHERE
        k0.funkcja IN (''KOT'', ''MILUSIA'')
    ORDER BY
        k0.imie;
    ';

-- Wywolanie
EXEC sp_executesql @dynamic_sql;
```

Imie	Szef 1	Szef 2	Szef 3	Szef 4	Szef 5
BELA	BOLEK	MRUCZEK			
DUDEK	PUCEK	MRUCZEK			
LATKA	PUCEK	MRUCZEK			
LUCEK	PUNIA	KOREK	MRUCZEK		
MICKA	MRUCZEK				
RUDA	MRUCZEK				
SONIA	KOREK	MRUCZEK			

((7 rows affected))

Dla liczby przełożonych = 5

Imie	Szef 1	Szef 2
BELA	BOLEK	MRUCZEK
DUDEK	PUCEK	MRUCZEK
LATKA	PUCEK	MRUCZEK
LUCEK	PUNIA	KOREK
MICKA	MRUCZEK	
RUDA	MRUCZEK	
SONIA	KOREK	MRUCZEK

((7 rows affected))

Dla liczby przełożonych = 2

B.

PL/SQL

```
-- TASK 7
-------------------------------------------------------------------------------------------------------
-- b)
UNDEFINE przelozeni_num_input;

DECLARE
    v_przelozeni_num NUMBER := &przelozeni_num_input;
    v_dynamic_sql    CLOB;
    v_pivot_for      CLOB   := '';
    v_cursor         NUMBER;
    v_column_cnt     NUMBER;
    v_desc_tab       DBMS_SQL.DESC_TAB;
    v_column_val     VARCHAR(4000);
    v_status         NUMBER;
BEGIN
    FOR i IN 1..v_przelozeni_num LOOP
        v_pivot_for := v_pivot_for || i || ' AS "Szef ' || i || '"';
        IF i < v_przelozeni_num THEN
            v_pivot_for := v_pivot_for || ', ';
        END IF;
    END LOOP;

    v_dynamic_sql :=
    '
    SELECT
        *
    FROM (
        SELECT
            CONNECT_BY_ROOT imie "Imie",
            LEVEL - 1               AS lvl,
            imie                    AS imie_szef
        FROM Kocury
        START WITH funkcja IN (''KOT'', ''MILUSIA'')
        CONNECT BY PRIOR szef = pseudo
    ) src
    PIVOT (
        MAX(imie_szef)
        FOR lvl IN (' || v_pivot_for ||')
    ) pvt
    ORDER BY
        "Imie"
    ';

    v_cursor := DBMS_SQL.OPEN_CURSOR;
```

```
    DBMS_SQL.PARSE(v_cursor, v_dynamic_sql, DBMS_SQL.NATIVE);
    DBMS_SQL.DESCRIBE_COLUMNS(v_cursor, v_column_cnt, v_desc_tab);

    FOR i IN 1..v_column_cnt LOOP
        DBMS_SQL.DEFINE_COLUMN(v_cursor, i, v_column_val, 4000);
    END LOOP;

    v_status := DBMS_SQL.EXECUTE(v_cursor);

    FOR i IN 1..v_column_cnt LOOP
        DBMS_OUTPUT.PUT(RPAD(v_desc_tab(i).col_name, 15));
    END LOOP;
    DBMS_OUTPUT.NEW_LINE;
    DBMS_OUTPUT.PUT_LINE(RPAD('-', 15 * v_column_cnt, '-'));

    WHILE DBMS_SQL.FETCH_ROWS(v_cursor) > 0 LOOP
        FOR i IN 1..v_column_cnt LOOP
            DBMS_SQL.COLUMN_VALUE(v_cursor, i, v_column_val);
            DBMS_OUTPUT.PUT(RPAD(NVL(v_column_val, ' '), 15));
        END LOOP;
        DBMS_OUTPUT.NEW_LINE;
    END LOOP;

    DBMS_SQL.CLOSE_CURSOR(v_cursor);
END;
/
```

```
Imie           Szef 1         Szef 2         Szef 3         Szef 4         Szef 5
-------------------------------------------------------------------------------------
BELA           BOLEK          MRUCZEK
DUDEK          PUCEK          MRUCZEK
LATKA          PUCEK          MRUCZEK
LUCEK          PUNIA          KOREK          MRUCZEK
MICKA          MRUCZEK
RUDA           MRUCZEK
SONIA          KOREK          MRUCZEK


PL/SQL procedure successfully completed.
```

Dla liczby przełożonych = 5

```
Imie           Szef 1         Szef 2
-----------------------------------------------
BELA           BOLEK          MRUCZEK
DUDEK          PUCEK          MRUCZEK
LATKA          PUCEK          MRUCZEK
LUCEK          PUNIA          KOREK
MICKA          MRUCZEK
RUDA           MRUCZEK
SONIA          KOREK          MRUCZEK


PL/SQL procedure successfully completed.
```

Dla liczby przełożonych = 2

Zadania - procedury

Zadanie 2

PL/SQL

a.

```
-------------------------------------------------------------------------------
-- TASK 11
-------------------------------------------------------------------------------
CREATE OR REPLACE FUNCTION GetFunctionCnt RETURN NUMBER
IS
    v_func_cnt NUMBER := 0;
BEGIN
    SELECT
        COUNT(DISTINCT funkcja)
    INTO
        v_func_cnt
    FROM
        Kocury;

    RETURN v_func_cnt;
END;
/

CREATE OR REPLACE FUNCTION GetSeparatorPodsumowanie (
    p_func_cnt IN NUMBER DEFAULT 0
) RETURN CLOB
IS
    v_dynamic_sql      CLOB;
    v_dynamic_func_sep CLOB   := '';
    v_index            NUMBER := 0;
BEGIN
    WHILE v_index < p_func_cnt LOOP
        v_dynamic_func_sep := CONCAT(v_dynamic_func_sep, 'RPAD(''-'', 10, ''-''),');
        v_index := v_index + 1;
    END LOOP;

    SELECT
    '
    SELECT
        RPAD(''Z'', 20, ''-''),
        RPAD(''-'', 6, ''-''),
        RPAD(''-'', 4, ''-''),
    '
    || v_dynamic_func_sep ||
    '
        RPAD(''-'', 7, ''-'')
    FROM dual

    UNION ALL

    SELECT
        ''ZJADA RAZEM'',
        '' '',
        '' '',
```

```
        '
    ||
    LISTAGG(
    '
        LPAD(SUM(DECODE(
            funkcja, ''' || funkcja || ''', przydzial_myszy + NVL(myszy_extra, 0), 0
        )), 10)',
        ', '
    ) WITHIN GROUP (ORDER BY funkcja DESC)
    ||
    '
        , LPAD(SUM(przydzial_myszy + NVL(myszy_extra, 0)), 7)
    FROM Kocury
    '
    INTO
        v_dynamic_sql
    FROM (
        SELECT DISTINCT
            funkcja
        FROM Kocury
    );

    RETURN v_dynamic_sql;
END GetSeparatorPodsumowanie;
/

-- a)
CREATE OR REPLACE PROCEDURE GetSumaSpozyciaA (
    p_rc OUT SYS_REFCURSOR
)
IS
    v_dynamic_sql CLOB;
BEGIN
    SELECT
        '
        SELECT
            RPAD(DECODE(k.plec, ''D'', b.nazwa, ''M'', '' ''), 20)      "NAZWA BANDY",
            RPAD(DECODE(k.plec, ''D'', ''Kotka'', ''M'', ''Kocor''), 6) "PLEC",
            LPAD(COUNT(*), 4)                                           "ILE",
        '
        ||
        LISTAGG(
            '
            LPAD(SUM(DECODE(
                k.funkcja, '''
                || funkcja || ''', k.przydzial_myszy + NVL(k.myszy_extra, 0),
                0
            )), 10) AS ' || funkcja, ', '
        ) WITHIN GROUP (ORDER BY funkcja DESC)
        ||
        '
        ,
            LPAD(SUM(k.przydzial_myszy + NVL(k.myszy_extra, 0)), 7)     "SUMA"
        FROM Kocury k
            INNER JOIN Bandy b ON b.nr_bandy = k.nr_bandy
        GROUP BY
            b.nazwa,
```

```
            k.plec

        UNION ALL'

        || GetSeparatorPodsumowanie(GetFunctionCnt)
    INTO
        v_dynamic_sql
    FROM (
        SELECT DISTINCT
            funkcja
        FROM Kocury
    );

    OPEN p_rc FOR v_dynamic_sql;
END GetSumaSpozyciaA;
/

VAR rc REFCURSOR
EXEC GetSumaSpozyciaA(:rc)
PRINT rc;
```

```
NAZWA BANDY         PLEC   ILE  SZEFUNIO   MILUSIA    LOWCZY     LAPACZ     KOT        DZIELCZY   BANDZIOR   SUMA
------------------- ------ ---- ---------- ---------- ---------- ---------- ---------- ---------- ---------- --------
                    Kocor     3          0          0         67         56          0          0         93      216
SZEFOSTWO           Kotka     2          0        136          0          0          0          0          0      136
                    Kocor     2          0          0          0          0         43          0         88      131
BIALI LOWCY         Kotka     2          0         55         61          0          0          0          0      116
LACIACI MYSLIWI     Kotka     2          0          0          0         51         40          0          0       91
                    Kocor     3          0          0         65         51         40          0          0      156
                    Kocor     2        136          0          0          0          0         50          0      186
CZARNI RYCERZE      Kotka     2          0         52         65          0          0          0          0      117
Z------------------ ------ ---- ---------- ---------- ---------- ---------- ---------- ---------- ---------- --------
ZJADA RAZEM                          136        243        258        158        123         50        181     1149

10 rows selected.
```

b.

```
-- b)
CREATE OR REPLACE PROCEDURE GetSumaSpozyciaB (
    p_rc OUT SYS_REFCURSOR
)
IS
    v_dynamic_sql CLOB;
BEGIN
    SELECT
        '
        SELECT
            RPAD(DECODE(plec, ''D'', nazwa, ''M'', '' ''), 20)         "NAZWA BANDY",
            RPAD(DECODE(plec, ''D'', ''Kotka'', ''M'', ''Kocor''), 6)  "PLEC",
            LPAD(ile, 4)                                               "ILE",
            '
            ||
            LISTAGG('LPAD(NVL(' || funkcja || ', 0), 10) AS ' || funkcja, ', ') WITHIN GROUP
(ORDER BY funkcja)
            ||
            ', LPAD('
            ||
            LISTAGG('NVL(' || funkcja || ', 0)', ' + ') WITHIN GROUP (ORDER BY funkcja)
            ||
            ', 7)    "SUMA"
        FROM (
            SELECT
                b.nazwa,
                k.plec,
                k.funkcja,
                k.przydzial_myszy + NVL(k.myszy_extra, 0) AS spozycie,
                COUNT(*) OVER (PARTITION BY nazwa, plec)  AS ile
            FROM Kocury k
                INNER JOIN Bandy b ON b.nr_bandy = k.nr_bandy
        )
        PIVOT (
            SUM(spozycie)
            FOR funkcja IN ('
            ||
            LISTAGG(
                '''' || funkcja || ''' AS ' || funkcja,
                ', '
            ) WITHIN GROUP (ORDER BY funkcja)
            ||
            ')
        )

        UNION ALL
        '
        || GetSeparatorPodsumowanie(GetFunctionCnt)
    INTO
        v_dynamic_sql
    FROM (
        SELECT DISTINCT
```

```
            funkcja
        FROM Kocury
    );

    OPEN p_rc FOR v_dynamic_sql;
END GetSumaSpozyciaB;
/

VAR rc REFCURSOR
EXEC GetSumaSpozyciaB(:rc)
PRINT rc;
```

```
NAZWA BANDY          PLEC    ILE  BANDZIOR   DZIELCZY   KOT        LAPACZ     LOWCZY     MILUSIA    SZEFUNIO   SUMA
-------------------- ------ ---- ---------- ---------- ---------- ---------- ---------- ---------- ---------- --------
BIALI LOWCY          Kotka     2          0          0          0          0         61         55          0      116
                     Kocor     2         88          0         43          0          0          0          0      131
CZARNI RYCERZE       Kotka     2          0          0          0          0         65         52          0      117
                     Kocor     3         93          0          0         56         67          0          0      216
LACIACI MYSLIWI      Kotka     2          0          0         40         51          0          0          0       91
                     Kocor     3          0          0         40         51         65          0          0      156
SZEFOSTWO            Kotka     2          0          0          0          0          0        136          0      136
                     Kocor     2          0         50          0          0          0          0        136      186
Z-------------------- ------ ---- ---------- ---------- ---------- ---------- ---------- ---------- ---------- --------
ZJADA RAZEM                             136        243        258        158        123         50        181     1149

10 rows selected.
```

a.

```
-------------------------------------------------------------------------------------------
-- TASK 11
-------------------------------------------------------------------------------------------
-- a)
CREATE OR ALTER FUNCTION GetFunctionCnt () RETURNS INT
AS
BEGIN
    DECLARE
        @v_func_cnt INT;

    SELECT
        @v_func_cnt = COUNT(DISTINCT funkcja)
    FROM Kocury;

    RETURN @v_func_cnt;
END;
GO

CREATE OR ALTER FUNCTION GetSeparatorPodsumowanie (
    @p_func_cnt INT
) RETURNS NVARCHAR(MAX)
AS
BEGIN
    DECLARE
        @v_dynamic_sql      NVARCHAR(MAX) = N'',
        @v_dynamic_func_sep NVARCHAR(MAX) = N'',
        @v_index            INT           = 0;

    WHILE @v_index < @p_func_cnt
    BEGIN
        SET @v_dynamic_func_sep += N'REPLICATE(''-'', 10), '
        SET @v_index += 1
    END

    SELECT
        @v_dynamic_sql =
        N'
        SELECT
            RIGHT(''Z'' + REPLICATE(''-'', 19), 20),
            REPLICATE(''-'', 6),
            REPLICATE(''-'', 4),
        '
        + @v_dynamic_func_sep +
        N'
            REPLICATE(''-'', 7)

        UNION ALL

        SELECT
            ''ZJADA RAZEM'',
            '' '',
            '' '',
        '
        +
```

```
        STRING_AGG(
            N'
                RIGHT(
                    REPLICATE('' '', 10) +
                        CAST(SUM(CASE
                                    WHEN funkcja = ''' + funkcja + N'''
                                    THEN przydzial_myszy + ISNULL(myszy_extra, 0)
                                    ELSE 0
                                END) AS VARCHAR(50)),
                10) AS ' + funkcja,
            N', '
        ) WITHIN GROUP (ORDER BY funkcja DESC)
        +
        N'
            , RIGHT(SPACE(7) + CAST(SUM(przydzial_myszy + ISNULL(myszy_extra, 0)) AS
VARCHAR(50)), 7)
        FROM Kocury
        '
    FROM (
        SELECT DISTINCT
            funkcja
        FROM Kocury
    ) f;

    RETURN @v_dynamic_sql;
END;
GO

CREATE OR ALTER PROCEDURE GetSumaSpozyciaA
AS
BEGIN
    DECLARE
        @v_dynamic_sql NVARCHAR(MAX);

    SELECT
        @v_dynamic_sql =
        N'
        SELECT
            CASE k.plec
                WHEN ''D'' THEN b.nazwa
                WHEN ''M'' THEN '' ''
            END                                         [NAZWA BANDY],
            CASE k.plec
                WHEN ''D'' THEN ''Kotka''
                WHEN ''M'' THEN ''Kocor''
            END                                         [PLEC],
            RIGHT(REPLICATE('' '', 4) + COUNT(*), 4) [ILE],'
            +
            STRING_AGG(
                N'
                RIGHT(REPLICATE('' '', 10) +
                CAST(SUM(
                    CASE
                        WHEN k.funkcja = ''' + funkcja + '''
                            THEN k.przydzial_myszy + ISNULL(k.myszy_extra, 0)
                        ELSE 0
                    END
```

```
                ) AS VARCHAR), 10) [' + funkcja + ']',
                ', '
            )
            +
            N',
                RIGHT(REPLICATE('' '', 7) + CAST(SUM(k.przydzial_myszy + ISNULL(k.myszy_extra,
0)) AS VARCHAR), 7) [SUMA]
            FROM Kocury k
                INNER JOIN Bandy b ON b.nr_bandy = k.nr_bandy
            GROUP BY
                b.nazwa,
                k.plec

            UNION ALL
            '
            + dbo.GetSeparatorPodsumowanie(dbo.GetFunctionCnt())
    FROM (
        SELECT DISTINCT
            funkcja
        FROM Kocury
    ) f;

    PRINT @v_dynamic_sql;

    EXEC sp_executesql @v_dynamic_sql;
END;
GO

EXEC GetSumaSpozyciaA;
GO
```

NAZWA BANDY	PLEC	ILE	BANDZIOR	DZIELCZY	KOT	LAPACZ	LOWCZY	MILUSIA	SZEFUNIO	SUMA
BIALI LOWCY	Kotka	2	0	0	0	0	61	55	0	116
CZARNI RYCERZE	Kotka	2	0	0	0	0	65	52	0	117
LACIACI MYSLIWI	Kotka	2	0	0	40	51	0	0	0	91
SZEFOSTWO	Kotka	2	0	0	0	0	0	136	0	136
	Kocor	2	88	0	43	0	0	0	0	131
	Kocor	3	93	0	0	56	67	0	0	216
	Kocor	3	0	0	40	51	65	0	0	156
	Kocor	2	0	50	0	0	0	0	136	186
Z----------	------	----	----------	----------	----------	----------	----------	----------	----------	-------
ZJADA RAZEM			136	243	258	158	123	50	181	1149

((10 rows affected))

b.

```
-- b)
CREATE OR ALTER PROCEDURE GetSumaSpozyciaB
AS
BEGIN
    DECLARE
        @v_dynamic_sql NVARCHAR(MAX);

    SELECT
        @v_dynamic_sql =
        N'
        SELECT
            CASE plec
                WHEN ''D'' THEN nazwa
                WHEN ''M'' THEN '' ''
            END                                                 [NAZWA BANDY],
            CASE plec
                WHEN ''D'' THEN ''Kotka''
                WHEN ''M'' THEN ''Kocor''
            END                                                 [PLEC],
            RIGHT(REPLICATE('' '', 4) + CAST(ile AS VARCHAR), 4) [ILE],
        '
        +
        STRING_AGG(
            'RIGHT(REPLICATE('' '', 10) + CAST(ISNULL(' + QUOTENAME(funkcja) + ', 0) AS
VARCHAR), 10) ' + QUOTENAME(funkcja),
            ', '
        ) WITHIN GROUP (ORDER BY funkcja)
        +
        ',
            RIGHT(REPLICATE('' '', 7) + CAST(' +
        STRING_AGG('ISNULL(' + funkcja + ', 0)', ' + ') WITHIN GROUP (ORDER BY funkcja) +
        '   AS VARCHAR), 7)                                     [SUMA]
        FROM (
            SELECT
                b.nazwa,
                k.plec,
                k.funkcja,
                k.przydzial_myszy + ISNULL(k.myszy_extra, 0) AS spozycie,
                COUNT(*) OVER (PARTITION BY b.nazwa, k.plec) AS ile
            FROM Kocury k
                INNER JOIN Bandy b ON b.nr_bandy = k.nr_bandy
        ) src
        PIVOT (
            SUM(spozycie)
            FOR funkcja IN ('
            +
            STRING_AGG(
                QUOTENAME(funkcja),
                ', '
            ) WITHIN GROUP (ORDER BY funkcja)
            +
            ')
        ) pvt

        UNION ALL
```

```
        '
        + dbo.GetSeparatorPodsumowanie(dbo.GetFunctionCnt())
    FROM (
        SELECT DISTINCT
            funkcja
        FROM Kocury
    ) f;

PRINT @v_dynamic_sql;
    EXEC sp_executesql @v_dynamic_sql;
END;
GO

EXEC GetSumaSpozyciaB;
GO
```

NAZWA BANDY	PLEC	ILE	BANDZIOR	DZIELCZY	KOT	LAPACZ	LOWCZY	MILUSIA	SZEFUNIO	SUMA
BIALI LOWCY	Kotka	2	0	0	0	0	61	55	0	116
	Kocor	2	88	0	43	0	0	0	0	131
CZARNI RYCERZE	Kotka	2	0	0	0	0	65	52	0	117
	Kocor	3	93	0	0	56	67	0	0	216
LACIACI MYSLIWI	Kotka	2	0	0	40	51	0	0	0	91
	Kocor	3	0	0	40	51	65	0	0	156
SZEFOSTWO	Kotka	2	0	0	0	0	0	136	0	136
	Kocor	2	0	50	0	0	0	0	136	186
Z----------	------	----	----------	----------	----------	----------	----------	----------	----------	--------
ZJADA RAZEM			136	243	258	158	123	50	181	1149

((10 rows affected))

Zadania - wyzwalacze

Zadanie 3

PL/SQL

```
-------------------------------------------------------------------------------------------
-- TASK 3
-------------------------------------------------------------------------------------------
-- Definicja
CREATE OR REPLACE TRIGGER trg_autonum_bandy
BEFORE INSERT ON Bandy
FOR EACH ROW
BEGIN
    SELECT
        MAX(nr_bandy) + 1
    INTO
        :NEW.nr_bandy
    FROM Bandy;
END trg_autonum_bandy;
/

-- Wywołanie
INSERT INTO Bandy VALUES (0, 'Nowa banda', 'Wybrzeze', 'MALA');

SELECT * FROM Bandy;
```

```
Trigger TRG_AUTONUM_BANDY compiled


1 row inserted.


  NR_BANDY NAZWA               TEREN          SZEF_BANDY
---------- ------------------- -------------- --------------
         1 SZEFOSTWO           CALOSC         TYGRYS
         2 CZARNI RYCERZE      POLE           LYSY
         3 BIALI LOWCY         SAD            ZOMBI
         4 LACIACI MYSLIWI     GORKA          RAFA
         5 ROCKERSI            ZAGRODA
         6 Nowa banda          Wybrzeze       MALA

6 rows selected.
```

Zadanie 4

PL/SQL

```
-------------------------------------------------------------------------------
-- TASK 4
-------------------------------------------------------------------------------
-- Definicja relacji
CREATE TABLE Przekroczenia_przydzialow (
    kto      VARCHAR2(15),
    kiedy    DATE,
    komu     VARCHAR(15),
    operacja VARCHAR(6)
);

-- Definicja wyzwalacza
CREATE OR REPLACE TRIGGER trg_guard_przydzialy
BEFORE INSERT OR UPDATE ON Kocury
FOR EACH ROW
DECLARE
    v_funkcja_min Funkcje.min_myszy%TYPE;
    v_funkcja_max Funkcje.max_myszy%TYPE;
    v_operacja    VARCHAR(6) := 'INSERT';
BEGIN
    IF UPDATING THEN
        v_operacja := 'UPDATE';
    END IF;

    SELECT
        min_myszy,
        max_myszy
    INTO
        v_funkcja_min,
        v_funkcja_max
    FROM Funkcje
    WHERE
        funkcja = :NEW.funkcja;

    IF :NEW.przydzial_myszy NOT BETWEEN v_funkcja_min AND v_funkcja_max THEN
        INSERT INTO Przekroczenia_przydzialow VALUES (SYS.LOGIN_USER, SYSDATE, :NEW.pseudo,
v_operacja);
        :NEW.przydzial_myszy := :OLD.przydzial_myszy;
    END IF;
END trg_guard_przydzialy;
/

-- Wywołanie
UPDATE Kocury
SET przydzial_myszy = 600
WHERE
    pseudo IN ('RAFA', 'ZERO');

SELECT
    kto,
    TO_CHAR(kiedy, 'YYYY-MM-DD') "KIEDY",
    komu,
    operacja
```

```sql
FROM Przekroczenia_przydzialow;

SELECT
    pseudo,
    przydzial_myszy
FROM Kocury
WHERE
    pseudo IN ('RAFA', 'ZERO');

DROP TABLE Przekroczenia_przydzialow;

ROLLBACK;
```

```
Table PRZEKROCZENIA_PRZYDZIALOW created.


Trigger TRG_GUARD_PRZYDZIALY compiled


2 rows updated.


KTO             KIEDY      KOMU            OPERAC
--------------- ---------- --------------- ------
SYSTEM          2026-01-15 ZERO            UPDATE
SYSTEM          2026-01-15 RAFA            UPDATE


PSEUDO          PRZYDZIAL_MYSZY
--------------- ---------------
ZERO                         43
RAFA                         65
```

Zadanie 5

PL/SQL

a)

```
-------------------------------------------------------------------------------
-- TASK 5
-------------------------------------------------------------------------------
-- a)
-- Definicja pakietu
CREATE OR REPLACE PACKAGE pkg_wirus
AS
    g_mutex             BOOLEAN   := FALSE;
    g_delta_przydzialow NUMBER(3);
END pkg_wirus;
/

-- Definicja wyzwalaczy
CREATE OR REPLACE TRIGGER trg_przed
BEFORE UPDATE OF przydzial_myszy ON Kocury
FOR EACH ROW
WHEN (OLD.funkcja = 'MILUSIA')
BEGIN
    -- Zapamietaj delte przydzialow
    IF pkg_wirus.g_delta_przydzialow IS NULL THEN
        pkg_wirus.g_delta_przydzialow := :NEW.przydzial_myszy - :OLD.przydzial_myszy;
    END IF;

    -- Jesli obnizka, rzuc wyjatek
    IF pkg_wirus.g_delta_przydzialow < 0 THEN
        pkg_wirus.g_delta_przydzialow := NULL;
        RAISE_APPLICATION_ERROR(-20001, 'Nie mozna zmniejszyc przydzialu myszy.');
    END IF;
END trg_przed;
/

CREATE OR REPLACE TRIGGER trg_po
AFTER UPDATE OF przydzial_myszy ON Kocury
DECLARE
    v_przydzial_tygrys  Kocury.przydzial_myszy%TYPE;
    v_10_procent_tygrys NUMBER(3);
BEGIN
    IF pkg_wirus.g_delta_przydzialow IS NULL OR pkg_wirus.g_delta_przydzialow = 0 OR
pkg_wirus.g_mutex THEN
        RETURN;
    END IF;

    -- Blokada na update
    pkg_wirus.g_mutex := TRUE;

    -- Wez aktualny przydzial tygrysa i jego 10%
    SELECT
        przydzial_myszy,
        ROUND(0.1 * przydzial_myszy)
    INTO
        v_przydzial_tygrys,
        v_10_procent_tygrys
```

```
    FROM Kocury
    WHERE
        pseudo = 'TYGRYS';

    IF pkg_wirus.g_delta_przydzialow < v_10_procent_tygrys THEN
        -- Zwieksz przydzialy dla milus
        UPDATE Kocury
        SET
            przydzial_myszy = przydzial_myszy + v_10_procent_tygrys -
pkg_wirus.g_delta_przydzialow,
            myszy_extra     = NVL(myszy_extra, 0) + 5
        WHERE
            funkcja = 'MILUSIA';

        -- Zabierz Tygrysowi
        UPDATE Kocury
        SET przydzial_myszy = przydzial_myszy - v_10_procent_tygrys
        WHERE
            pseudo = 'TYGRYS';
    ELSE
        -- Dodaj extra Tygrysowi
        UPDATE Kocury
        SET myszy_extra = NVL(myszy_extra, 0) + 5
        WHERE
            pseudo = 'TYGRYS';
    END IF;

    pkg_wirus.g_delta_przydzialow := NULL;
    pkg_wirus.g_mutex             := FALSE;
EXCEPTION
    WHEN OTHERS THEN
        pkg_wirus.g_delta_przydzialow := NULL;
        pkg_wirus.g_mutex             := FALSE;
        RAISE;
END trg_po;
/

-- Wywołanie
SELECT
    pseudo,
    funkcja,
    przydzial_myszy,
    NVL(myszy_extra, 0) "EXTRA"
FROM Kocury
WHERE
    funkcja = 'MILUSIA'
    OR
    pseudo = 'TYGRYS'
ORDER BY
    funkcja;

UPDATE Kocury
SET przydzial_myszy = przydzial_myszy + 4
WHERE
    funkcja = 'MILUSIA';

SELECT
```

```
    pseudo,
    funkcja,
    przydzial_myszy,
    NVL(myszy_extra, 0) "EXTRA"
FROM Kocury
WHERE
    funkcja = 'MILUSIA'
    OR
    pseudo = 'TYGRYS'
ORDER BY
    funkcja;

ROLLBACK;
```

```
Package PKG_WIRUS compiled


Trigger TRG_PRZED compiled


Trigger TRG_PO compiled
```

```
PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33


Error starting at line : 87 in command -
UPDATE Kocury
SET przydzial_myszy = przydzial_myszy - 4
WHERE
    funkcja = 'MILUSIA'
Error at Command Line : 87 Column : 8
Error report -
SQL Error: ORA-20001: Nie mozna zmniejszyc przydzialu myszy.
ORA-06512: at "SYSTEM.TRIG_PRZED", line 11
ORA-04088: error during execution of trigger 'SYSTEM.TRIG_PRZED'

https://docs.oracle.com/error-help/db/ora-20001/

More Details :
https://docs.oracle.com/error-help/db/ora-20001/
https://docs.oracle.com/error-help/db/ora-06512/
https://docs.oracle.com/error-help/db/ora-04088/

PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33
```

Dla obniżki

```
PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33


4 rows updated.


PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 35         52
PUSZYSTA        MILUSIA                 30         40
LASKA           MILUSIA                 34         33
MALA            MILUSIA                 32         47
TYGRYS          SZEFUNIO                93         33
```

Dla podwyżki $< 10\%$

```
PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33


4 rows updated.


PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 65         47
PUSZYSTA        MILUSIA                 60         35
LASKA           MILUSIA                 64         28
MALA            MILUSIA                 62         42
TYGRYS          SZEFUNIO               103         38
```

Dla podwyżki $\geq 10\%$

b)

```
-- b)
--Definicja wyzwalacza
CREATE OR REPLACE TRIGGER trg_wirus
FOR UPDATE OF przydzial_myszy ON Kocury
COMPOUND TRIGGER
    v_delta_przydzialow Kocury.przydzial_myszy%TYPE;
    v_przydzial_tygrys  Kocury.przydzial_myszy%TYPE;
    v_10_procent_tygrys Kocury.przydzial_myszy%TYPE;
    v_za_malo           BOOLEAN := FALSE;

    BEFORE STATEMENT IS
    BEGIN
        -- Oblicz przdzial Tygrysa i jego 10%
        SELECT
            przydzial_myszy,
            ROUND(0.1 * przydzial_myszy)
        INTO
            v_przydzial_tygrys,
            v_10_procent_tygrys
        FROM Kocury
        WHERE
            pseudo = 'TYGRYS';
    END BEFORE STATEMENT;

    BEFORE EACH ROW IS
    BEGIN
        -- Zapamietaj delte przydzialow dla milus
        IF :OLD.funkcja = 'MILUSIA' AND v_delta_przydzialow IS NULL THEN
            v_delta_przydzialow := :NEW.przydzial_myszy - :OLD.przydzial_myszy;

            IF v_delta_przydzialow < 0 THEN
                RAISE_APPLICATION_ERROR(-20001, 'Nie mozna zmniejszyc przydzialu myszy.');
            END IF;

            IF v_delta_przydzialow < v_10_procent_tygrys THEN
                -- Zaktualizuj bonusowe przydzialy dla milus
                v_za_malo            := TRUE;
                :NEW.przydzial_myszy := :OLD.przydzial_myszy + v_10_procent_tygrys;
                :NEW.myszy_extra     := NVL(:NEW.myszy_extra, 0) + 5;
            END IF;
        END IF;
    END BEFORE EACH ROW;

    AFTER STATEMENT IS
    BEGIN
        IF v_delta_przydzialow != 0 THEN
            IF v_za_malo THEN
                -- Zabierz Tygrysowi
                UPDATE Kocury
                SET przydzial_myszy = przydzial_myszy - v_10_procent_tygrys
                WHERE
                    pseudo = 'TYGRYS';
            ELSE
                -- Dodaj extra Tygrysowi
                UPDATE Kocury
```

```
                SET myszy_extra = NVL(myszy_extra, 0) + 5
                WHERE
                    pseudo = 'TYGRYS';
            END IF;
        END IF;
    END AFTER STATEMENT;
END trg_wirus;
/

-- Wywołanie
SELECT
    pseudo,
    funkcja,
    przydzial_myszy,
    NVL(myszy_extra, 0) "EXTRA"
FROM Kocury
WHERE
    funkcja = 'MILUSIA'
    OR
    pseudo = 'TYGRYS'
ORDER BY
    funkcja;

UPDATE Kocury
SET przydzial_myszy = przydzial_myszy + 4
WHERE
    funkcja = 'MILUSIA';

SELECT
    pseudo,
    funkcja,
    przydzial_myszy,
    NVL(myszy_extra, 0) "EXTRA"
FROM Kocury
WHERE
    funkcja = 'MILUSIA'
    OR
    pseudo = 'TYGRYS'
ORDER BY
    funkcja;

ROLLBACK;
```

```
PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33


Error starting at line : 301 in command -
UPDATE Kocury
SET przydzial_myszy = przydzial_myszy - 4
WHERE
    funkcja = 'MILUSIA'
Error at Command Line : 301 Column : 8
Error report -
SQL Error: ORA-20001: Nie mozna zmniejszyc przydzialu myszy.
ORA-06512: at "SYSTEM.TRG_WIRUS", line 25
ORA-04088: error during execution of trigger 'SYSTEM.TRG_WIRUS'

https://docs.oracle.com/error-help/db/ora-20001/

More Details :
https://docs.oracle.com/error-help/db/ora-20001/
https://docs.oracle.com/error-help/db/ora-06512/
https://docs.oracle.com/error-help/db/ora-04088/

PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33


Rollback complete.
```

Dla obniżki

```
PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33


4 rows updated.

PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 35         52
PUSZYSTA        MILUSIA                 24         35
LASKA           MILUSIA                 28         28
MALA            MILUSIA                 26         42
TYGRYS          SZEFUNIO                93         33


Rollback complete.
```

Dla podwyżki $< 10\%$

```
Trigger TRG_WIRUS compiled


PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 25         47
PUSZYSTA        MILUSIA                 20         35
LASKA           MILUSIA                 24         28
MALA            MILUSIA                 22         42
TYGRYS          SZEFUNIO               103         33


4 rows updated.

PSEUDO          FUNKCJA    PRZYDZIAL_MYSZY      EXTRA
--------------- ---------- --------------- ----------
LOLA            MILUSIA                 65         47
PUSZYSTA        MILUSIA                 60         35
LASKA           MILUSIA                 64         28
MALA            MILUSIA                 62         42
TYGRYS          SZEFUNIO               103         38


Rollback complete.
```

Dla podwyżki $\geq 10\%$

Zadanie 6

PL/SQL

```
-------------------------------------------------------------------------------
-- TASK 6
-------------------------------------------------------------------------------
-- Definicja relacji
CREATE TABLE Dodatki_extra (
    pseudo        VARCHAR2(15),
    dodatek_extra NUMBER(3)
);

-- Definicja wyzwalacza
CREATE OR REPLACE TRIGGER trg_dokumentuj_podwyzki_milus
FOR UPDATE OF przydzial_myszy ON Kocury
COMPOUND TRIGGER
    v_czy_update_milusia BOOLEAN := FALSE;
    v_czy_podwyzka       BOOLEAN := FALSE;
    v_dynamic_sql        CLOB;

    BEFORE EACH ROW IS
    BEGIN
        v_czy_update_milusia := :OLD.funkcja = 'MILUSIA';
        v_czy_podwyzka       := :OLD.przydzial_myszy < :NEW.przydzial_myszy;
    END BEFORE EACH ROW;

    AFTER STATEMENT IS
    BEGIN
        IF v_czy_update_milusia AND SYS.LOGIN_USER != 'TYGRYS' AND v_czy_podwyzka THEN
            v_dynamic_sql := v_dynamic_sql ||
                '
                INSERT INTO Dodatki_extra (pseudo, dodatek_extra)
                SELECT
                    pseudo,
                    -10
                FROM Kocury
                WHERE
                    funkcja = ''MILUSIA''
                ';

            EXECUTE IMMEDIATE v_dynamic_sql;
        END IF;
    END AFTER STATEMENT;


END trg_dokumentuj_podwyzki_milus;
/

-- Wywołanie
SELECT
    *
FROM Dodatki_extra;

UPDATE Kocury
SET przydzial_myszy = przydzial_myszy + 2
WHERE
```

```
    pseudo = 'LOLA';

SELECT
    *
FROM Dodatki_extra;

ROLLBACK;

DROP TABLE Dodatki_extra;

DROP TRIGGER trg_dokumentuj_podwyzki_milus;
```

```
Table DODATKI_EXTRA created.


Trigger TRG_DOKUMENTUJ_PODWYZKI_MILUS compiled

no rows selected

1 row updated.


PSEUDO          DODATEK_EXTRA
--------------- -------------
LOLA                      -10
PUSZYSTA                  -10
MALA                      -10
LASKA                     -10
```

Zadanie 7

PL/SQL

a)

```
-------------------------------------------------------------------------------
-- TASK 7
-------------------------------------------------------------------------------
-- a)
BEGIN
    DECLARE
        v_table_count NUMBER;
    BEGIN
        SELECT
            COUNT(*)
        INTO
            v_table_count
        FROM USER_TABLES
        WHERE
            TABLE_NAME = 'MYSZY';

        IF v_table_count != 0 THEN
            EXECUTE IMMEDIATE 'DROP TABLE Myszy CASCADE CONSTRAINTS';
            DBMS_OUTPUT.PUT_LINE('Usunieto tabele `Myszy`.');
        END IF;

    EXECUTE IMMEDIATE
    '
        CREATE TABLE Myszy (
        nr_myszy       NUMBER       CONSTRAINT pk_myszy PRIMARY KEY,
        lowca          VARCHAR2(15) CONSTRAINT fk_lowca REFERENCES Kocury(pseudo),
        zjadacz        VARCHAR2(15) CONSTRAINT fk_zjadacz REFERENCES Kocury(pseudo),
        waga_myszy     NUMBER,
        data_zlowienia DATE,
        data_wydania   DATE
    )
    ';

    DBMS_OUTPUT.PUT_LINE('Utworzono tabele `Myszy`.');
END;
/
```

```
Usunieto tabele `Myszy`.
Utworzono tabele `Myszy`.
```

b)

```
-- b)
DECLARE
    v_count NUMBER;
    v_name CONSTANT VARCHAR2(13) := 'SEQ_MYSZY_NUM';
BEGIN
    SELECT
        COUNT(*)
    INTO
        v_count
    FROM USER_SEQUENCES
    WHERE
        SEQUENCE_NAME = v_name;

    IF v_count != 0 THEN
        EXECUTE IMMEDIATE 'DROP SEQUENCE ' || v_name;
    END IF;

    EXECUTE IMMEDIATE
    '
        CREATE SEQUENCE ' || v_name || '
        START WITH 1
        INCREMENT BY 1
        NOCACHE
    ';
END;
/

-- Wypełnianie historii wstecz az do biezacej daty lub daty przed dniem oddania listy
CREATE OR REPLACE PROCEDURE WypenijHistorieMyszy IS
    -- Definicje typów
    TYPE t_pseudo_tab IS TABLE OF Kocury.pseudo%TYPE;
    TYPE t_myszy_tab  IS TABLE OF Myszy%ROWTYPE;

    -- Definicje tabel
    v_kocury t_pseudo_tab;
    v_myszy  t_myszy_tab := t_myszy_tab();

    -- Definicje dat
    v_data_koniec_final CONSTANT DATE := TO_DATE('2026-01-30', 'YYYY-MM-DD');

    v_data_start     DATE := TO_DATE('2004-01-01', 'YYYY-MM-DD');
    v_data_koniec    DATE := CASE WHEN SYSDATE > v_data_koniec_final THEN v_data_koniec_final
ELSE SYSDATE END;
    v_miesiac        DATE;
    v_ostatnia_sroda DATE;

    -- Pozostałe zmienne obliczeniowe
    v_myszy_miesiecznie_total NUMBER;
    v_liczba_kotow            NUMBER;
    v_srednio_na_kota         NUMBER;
    v_reszta                  NUMBER;
    v_dni_w_miesiacu          NUMBER;
    v_licznik                 NUMBER := 0;
BEGIN
    -- Zebranie pseudo kocurów w kolejnosci dołączenia do stada
```

```
    SELECT DISTINCT
        pseudo
    BULK COLLECT INTO v_kocury
    FROM Kocury
    START WITH szef IS NULL
    CONNECT BY PRIOR pseudo = szef;

    -- Jesli kocurów nie ma, wyjscie
    v_liczba_kotow := v_kocury.COUNT;
    IF v_liczba_kotow = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Brak kotow - brak danych do uzupelnienia.');
        RETURN;
    END IF;

    -- Obliczenie łącznej liczby myszy do rozdzielenia miesiecznie
    SELECT
        SUM(przydzial_myszy + NVL(myszy_extra, 0))
    INTO
        v_myszy_miesiecznie_total
    FROM Kocury;

    v_miesiac := TRUNC(v_data_start, 'MM');

    -- Główna pętla po miesiącach
    WHILE v_miesiac <= v_data_koniec LOOP
        v_dni_w_miesiacu  := LAST_DAY(v_miesiac) - v_miesiac + 1;
        v_ostatnia_sroda  := NEXT_DAY(LAST_DAY(v_miesiac) - 7, 'WEDNESDAY');
        v_srednio_na_kota := FLOOR(v_myszy_miesiecznie_total / v_liczba_kotow);
        v_reszta          := v_myszy_miesiecznie_total - v_srednio_na_kota * v_liczba_kotow;

        v_myszy.DELETE;

        -- Rozdzielenie myszy pomiędzy kocury
        FOR loop_var_i IN 1..v_liczba_kotow LOOP
            DECLARE
                v_myszy_na_kota NUMBER := v_srednio_na_kota;
            BEGIN
                -- Reszta dzielona po jednej myszy na kota
                IF v_reszta > 0 THEN
                    v_myszy_na_kota := v_myszy_na_kota + 1;
                    v_reszta        := v_reszta - 1;
                END IF;

                -- Wypełnienie tablicy myszy do wstawienia
                FOR loop_var_j IN 1..v_myszy_na_kota LOOP
                    v_licznik := v_licznik + 1;

                    v_myszy.EXTEND;
                    v_myszy(v_myszy.COUNT).nr_myszy       := seq_myszy_num.NEXTVAL;
                    v_myszy(v_myszy.COUNT).lowca          := v_kocury(loop_var_i);
                    v_myszy(v_myszy.COUNT).zjadacz        := v_kocury(loop_var_i);
                    v_myszy(v_myszy.COUNT).waga_myszy     := ROUND(DBMS_RANDOM.VALUE(15, 55),
2);
                    v_myszy(v_myszy.COUNT).data_zlowienia := v_miesiac +
TRUNC(DBMS_RANDOM.VALUE(0, TO_NUMBER(TO_CHAR(v_ostatnia_sroda, 'DD'))));
                    v_myszy(v_myszy.COUNT).data_wydania   := v_ostatnia_sroda;
                END LOOP;
```

```
            END;
        END LOOP;

        -- Wstawienie danych co miesiąc
        IF v_myszy.COUNT > 0 THEN
                FORALL idx IN v_myszy.FIRST..v_myszy.LAST
                INSERT INTO Myszy VALUES v_myszy(idx);
        END IF;

        v_miesiac := ADD_MONTHS(v_miesiac, 1);
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Historia myszy uzupelniona pomyslnie.');
END WypenijHistorieMyszy;
/

BEGIN
    WypenijHistorieMyszy;
END;
/

-- Sprawdzenie wyników
SELECT
    nr_myszy,
    lowca,
    zjadacz,
    waga_myszy,
    TO_CHAR(data_zlowienia, 'YYYY-MM-DD') "DATA_ZLOWIENIA",
    TO_CHAR(data_wydania, 'YYYY-MM-DD')   "DATA_WYDANIA"
FROM Myszy;

-- Rejestrowanie dziennej historii łowów dla danego kocura
CREATE OR REPLACE PROCEDURE RejestrujDzienneLowy (
    p_pseudo Kocury.pseudo%TYPE
) IS
    -- Definicje typów
    TYPE t_daty_tab IS TABLE OF DATE;
    TYPE t_wagi_tab IS TABLE OF NUMBER;

    -- Definicje tabel
    v_daty t_daty_tab;
    v_wagi t_wagi_tab;

    -- Pozostałe zmienne
    v_pseudo_liczba NUMBER;
BEGIN
    SELECT
        COUNT(*)
    INTO
        v_pseudo_liczba
    FROM Kocury
    WHERE
        pseudo = p_pseudo;

    IF v_pseudo_liczba = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Brak kocurow o wskazanym pseudonimie: ' || p_pseudo);
        RETURN;
```

```
    END IF;

    EXECUTE IMMEDIATE
     '
        SELECT
            data_zlowienia,
            waga_myszy
        FROM ' || 'DzienneLowy_' || p_pseudo || '
        WHERE
            pseudo = ' || p_pseudo || '
    '
    BULK COLLECT INTO
        v_daty,
        v_wagi;

    IF v_daty.COUNT = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Brak nowych zlowionych myszy dla: ' || p_pseudo);
        RETURN;
    END IF;

    FORALL idx IN v_wagi.FIRST..v_wagi.LAST
    INSERT INTO Myszy VALUES (
        seq_myszy_num.NEXTVAL,
        p_pseudo,
        NULL, -- Zjadacz jeszcze nie okreslony
        v_wagi(idx),
        v_daty(idx),
        NULL -- Data wydania jeszcze nie okreslona
    );
END RejestrujDzienneLowy;
/

-- Przyznanie myszy kocurom
CREATE OR REPLACE PROCEDURE MiesiecznaWyplataMyszy
IS
    -- Definicje typów
    TYPE t_kot_info IS RECORD (
        pseudo              Kocury.pseudo%TYPE,
        limit_myszy         NUMBER,
        przydzielone_myszy  NUMBER
    );

    TYPE t_koty_tab      IS TABLE OF t_kot_info;
    TYPE t_nr_myszy_tab  IS TABLE OF Myszy.nr_myszy%TYPE;
    TYPE t_pseudo_tab    IS TABLE OF Kocury.pseudo%TYPE;
    TYPE t_date_tab      IS TABLE OF DATE;

    -- Definicje tabel
    v_koty               t_koty_tab;
    v_dostepne_myszy     t_nr_myszy_tab;
    v_wynik_id_myszy     t_nr_myszy_tab := t_nr_myszy_tab();
    v_wynik_zjadacze     t_pseudo_tab   := t_pseudo_tab();

    -- Definicje dat
    v_ostatnia_sroda     DATE := NEXT_DAY(LAST_DAY(SYSDATE) - 7, 'WEDNESDAY');

    -- Pozostałe zmienne
```

```
    v_idx_mysz           PLS_INTEGER := 1;
    v_idx_kot            PLS_INTEGER := 1;
    v_pelne_koty_z_rzedu PLS_INTEGER := 0;
    v_liczba_kotow       PLS_INTEGER;
    v_liczba_myszy       PLS_INTEGER;
BEGIN
    IF TRUNC(SYSDATE) != TRUNC(v_ostatnia_sroda) THEN
        RETURN;
    END IF;

    -- Zebranie kocurów zgodnie z hierarchią
    SELECT DISTINCT
        pseudo,
        przydzial_myszy + NVL(myszy_extra, 0),
        0
    BULK COLLECT INTO v_koty
    FROM Kocury
    START WITH szef IS NULL
    CONNECT BY PRIOR pseudo = szef;

    v_liczba_kotow := v_koty.COUNT;

    EXECUTE IMMEDIATE
    '
    SELECT
        COUNT(*)
    INTO
        v_dostepne_myszy
    FROM Myszy
    WHERE
        zjadacz IS NULL;
    ';

    v_liczba_myszy := v_dostepne_myszy.COUNT;

    IF v_liczba_kotow = 0 OR v_liczba_myszy = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Brak kotow lub myszy - brak danych do przyznania myszy.');
        RETURN;
    END IF;

    -- Główna pętla przydziału myszy
    LOOP
        EXIT WHEN v_idx_mysz > v_liczba_myszy OR v_pelne_koty_z_rzedu >= v_liczba_kotow;

        -- Przydzielenie myszy aktualnemu kocurowi
        IF v_koty(v_idx_kot).przydzielone_myszy < v_koty(v_idx_kot).limit_myszy THEN
            v_wynik_id_myszy.EXTEND;
            v_wynik_zjadacze.EXTEND;

            v_wynik_id_myszy(v_wynik_id_myszy.COUNT) := v_dostepne_myszy(v_idx_mysz);
            v_wynik_zjadacze(v_wynik_zjadacze.COUNT) := v_koty(v_idx_kot).pseudo;

            v_koty(v_idx_kot).przydzielone_myszy     := v_koty(v_idx_kot).przydzielone_myszy +
1;
            v_idx_mysz                               := v_idx_mysz + 1;
            v_pelne_koty_z_rzedu                     := 0;
        ELSE
```

```
            -- Jesli kot pelny, zwieksz licznik pelnych kotow z rzedu
            v_pelne_koty_z_rzedu := v_pelne_koty_z_rzedu + 1;
        END IF;

        -- Jesli myszy do rozdysponowania wciaz jeszcze sa, wroc na poczatek
        v_idx_kot := v_idx_kot + 1;
        IF v_idx_kot > v_liczba_kotow THEN
            v_idx_kot := 1;
        END IF;
    END LOOP;

    -- Przyznanie nadmiarowych myszy Tygrysowi
    WHILE v_idx_mysz <= v_liczba_myszy LOOP
        v_wynik_id_myszy.EXTEND;
        v_wynik_zjadacze.EXTEND;

        v_wynik_id_myszy(v_wynik_id_myszy.COUNT) := v_dostepne_myszy(v_idx_mysz);
        v_wynik_zjadacze(v_wynik_zjadacze.COUNT) := 'TYGRYS';
        v_idx_mysz                               := v_idx_mysz + 1;
    END LOOP;

    -- Aktualizacja tabeli
    IF v_wynik_zjadacze.COUNT = 0 THEN
        DBMS_OUTPUT.PUT_LINE('Nie rozdano myszy.');
        RETURN;
    END IF;

    FORALL idx IN 1..v_wynik_id_myszy.COUNT
    UPDATE Myszy
    SET
        zjadacz      = v_wynik_zjadacze(idx),
        data_wydania = SYSDATE
    WHERE
        nr_myszy = v_wynik_id_myszy(idx);
END;
/

BEGIN
    MiesiecznaWyplataMyszy;
END;
/
```

```
PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Procedure WYPENIJHISTORIEMYSZY compiled
Historia myszy uzupelniona pomyslnie.

PL/SQL procedure successfully completed.

Procedure REJESTRUJDZIENNELOWY compiled

Procedure MIESIECZNAWYPLATAMYSZY compiled

PL/SQL procedure successfully completed.
```

Fragment wygenerowanych przydziałów myszy:

```
  NR_MYSZY LOWCA           ZJADACZ         WAGA_MYSZY DATA_ZLOWI DATA_WYDAN
---------- --------------- --------------- ---------- ---------- ----------
    304470 PUSZYSTA        PUSZYSTA             31,37 2026-01-01 2026-01-28
    304471 PUSZYSTA        PUSZYSTA             39,81 2026-01-11 2026-01-28
    304472 PUSZYSTA        PUSZYSTA             23,51 2026-01-24 2026-01-28
    304473 PUSZYSTA        PUSZYSTA             32,69 2026-01-28 2026-01-28
    304474 PUSZYSTA        PUSZYSTA             20,89 2026-01-16 2026-01-28
    304475 PUSZYSTA        PUSZYSTA             52,12 2026-01-14 2026-01-28
    304476 PUSZYSTA        PUSZYSTA             24,93 2026-01-13 2026-01-28
    304477 PUSZYSTA        PUSZYSTA             47,58 2026-01-17 2026-01-28
    304478 PUSZYSTA        PUSZYSTA             44,24 2026-01-03 2026-01-28
    304479 PUSZYSTA        PUSZYSTA             32,91 2026-01-24 2026-01-28
    304480 PUSZYSTA        PUSZYSTA             20,35 2026-01-11 2026-01-28

  NR_MYSZY LOWCA           ZJADACZ         WAGA_MYSZY DATA_ZLOWI DATA_WYDAN
---------- --------------- --------------- ---------- ---------- ----------
    304481 PUSZYSTA        PUSZYSTA             39,06 2026-01-13 2026-01-28
    304482 PUSZYSTA        PUSZYSTA             20,66 2026-01-21 2026-01-28
    304483 PUSZYSTA        PUSZYSTA             52,77 2026-01-16 2026-01-28
    304484 PUSZYSTA        PUSZYSTA             29,18 2026-01-01 2026-01-28
    304485 PUSZYSTA        PUSZYSTA              41,4 2026-01-15 2026-01-28

304,485 rows selected.
```