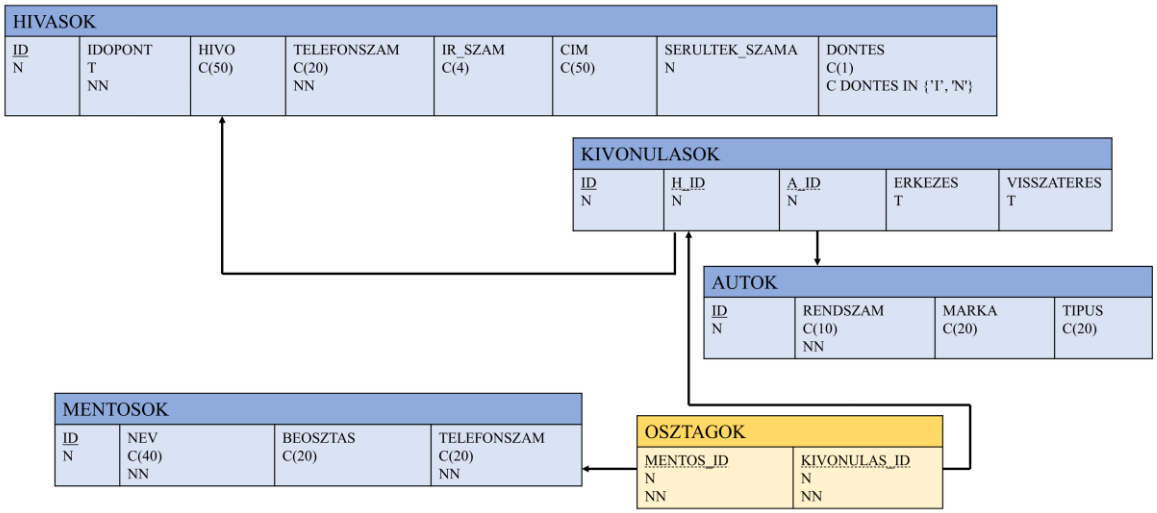
Adatbázis rendszerek II.
Második beadandó feladat

Palencsár Enikő
YD11NL

Gyakorlatvezető: dr. Kovács László
Gyakorlat időpontja: Hétfő 10:00
Képzés: BSc nappali

A feladat leírása:

A feladat egy mentőszolgálat adatbázis szerveroldalának megtervezése volt. Az adatbázis séma az alábbi:



HIVASOK: a beérkező hívások kezelésére

- ID
- IDOPONT: a hívás beérkezésének ideje (timestamp)
- HIVO: a hívó neve
- TELEFONSZAM: a telefonszám, amiről a hívás érkezik
- IR_SZAM: irányítószám, ahova ki kell vonulni
- CIM: cím, ahova ki kell vonulni
- SERULTEK_SZAMA: bejelentett sérültek/betegek száma
- DONTES: a diszpécser döntése, I ha szükséges a kivonulás, N, ha nem

KIVONULASOK: a helyszínre kivonulások naplózása

- ID
- H_ID: az adott helyszíni kivonulást elindító hívás id-je
- A_ID: a kivonuló autó id-je
- ERKEZES: az érkezés ideje (timestamp)
- VISSZATERES: a kiküldetésből való visszatérés ideje a mentőállomásra (ezután küldhető ki egy kocsi megint)
- plusz kikötés: a szereplő H_ID-A_ID párok egyediek, egy hívásra egy autó csak egyszer megy ki

AUTOK:

- ID
- RENDSZAM: az autó rendszáma
- MARKA: az autó márkája
- TIPUS: a mentőautó típusa (pl: rohamkocsi, gyerekmentő, esetkocsi..stb)

MENTOSOK:

- ID
- NEV: mentős neve
- BEOSZTAS: mentős beosztása (pl: mentőápoló, mentőorvos…stb)
- TELEFONSZAM: a mentős telefonszáma

OSZTAGOK:

- MENTOS_ID: a mentős azonosítója, aki kivonult
- KIVONULAS_ID: a kivonulás azonosítója (ebből tudjuk azt is, melyik kocsival vonult ki)
- ez a két mező együttesen adja a tábla elsődleges kulcsát

A sémát létrehozó SQL parancsok:

```
CREATE TABLE HIVASOK(
ID NUMBER PRIMARY KEY,
IDOPONT TIMESTAMP NOT NULL,
HIVO VARCHAR2(50),
TELEFONSZAM VARCHAR2(20) NOT NULL,
IR_SZAM CHAR(4),
CIM VARCHAR2(50),
SERULTEK_SZAMA NUMBER,
DONTES CHAR(1) CHECK DONTES='I' OR DONTES='N'
)

CREATE TABLE MENTOSOK(
ID NUMBER PRIMARY KEY,
NEV VARCHAR2(40) NOT NULL,
BEOSZTAS VARCHAR2(20),
TELEFONSZAM VARCHAR2(20) NOT NULL
)

CREATE TABLE AUTOK(
ID NUMBER PRIMARY KEY,
RENDSZAM CHAR(10) NOT NULL UNIQUE,
MARKA VARCHAR2(20),
TIPUS VARCHAR2(20)
)

CREATE TABLE KIVONULASOK(
ID NUMBER PRIMARY KEY,
H_ID NUMBER REFERENCES HIVASOK(ID),
A_ID NUMBER REFERENCES AUTOK(ID),
ERKEZES TIMESTAMP,
VISSZATERES TIMESTAMP,
UNIQUE(H_ID, A_ID)
)
```

```
CREATE TABLE OSZTAGOK(
KIVONULAS_ID NUMBER REFERENCES KIVONULASOK(ID),
MENTOS_ID NUMBER REFERENCES MENTOSOK(ID),
PRIMARY KEY(KIVONULAS_ID, MENTOS_ID)
);
```

PL/SQL csomag:

```
CREATE OR REPLACE PACKAGE MENTOSZOLGALAT AS
    PROCEDURE AUTO_BEOLVAS(PHIBA OUT INTEGER);
    PROCEDURE DISZPECSER_DONTES(PID NUMBER, PDONTES CHAR, PHIBA OUT INTEGER);
    PROCEDURE HIVAS_BEOLVAS(PHIBA OUT INTEGER);
    PROCEDURE HIVAS_FELTOLT_RANDOM(PNUM INTEGER, PHIBA OUT INTEGER);
    PROCEDURE HIVAS_NAPLOZ(PIDO TIMESTAMP, PTEL VARCHAR2, PID OUT INTEGER);
    PROCEDURE HIVAS_RESZLETEK(PID NUMBER, PHIVO VARCHAR2, PIR CHAR, PCIM
    VARCHAR2, PSERULT NUMBER, PHIBA OUT INTEGER);
    PROCEDURE HIVAS_SZURES_IDOSZAK(PKEZDET TIMESTAMP, PVEG TIMESTAMP, KURZOR
    OUT SYS_REFCURSOR);
    PROCEDURE HIVAS_SZURES_NEV(PNEV VARCHAR2, KURZOR OUT SYS_REFCURSOR);
    PROCEDURE KIVONULAS_FELTOLT_RANDOM(PHIBA OUT INTEGER);
    PROCEDURE KIVONULAS_FELVESZ(PHID NUMBER, PAID NUMBER, PERK TIMESTAMP,
    PVISSZA TIMESTAMP, PHIBA OUT INTEGER);
    PROCEDURE MENTOS_BEOLVAS(PHIBA OUT INTEGER);
    PROCEDURE OSZTAG_FELTOLT_RANDOM(PHIBA OUT INTEGER);
    PROCEDURE OSZTAG_FELVESZ (PKIVONULASID NUMBER, PMENTOSID NUMBER, PHIBA OUT
    INTEGER);
    FUNCTION HIVASOK_SZAMA(PNAP DATE) RETURN NUMBER;
    FUNCTION KIKULDETESEN_VOLT_E(PMENTOS_ID NUMBER, PKEZDET TIMESTAMP, PVEG
    TIMESTAMP) RETURN VARCHAR2;
    FUNCTION MAX_KIVONULASI_IDO_PERC RETURN INTEGER;
    FUNCTION MENTOS_TELEFONSZAM(PID NUMBER) RETURN VARCHAR2;
END;
```

- tábla feltöltés véletlen elemekkel
    - HIVASOK: HIVAS_FELTOLT_RANDOM eljárás
    - KIVONULASOK: KIVONULAS_FELTOLT_RANDOM
    - OSZTAGOK: OSZTAG_FELTOLT_RANDOM
- tábla feltöltés állományból
    - MENTOSOK: MENTOS_BEOLVAS
        - automatikusan generált PK
    - AUTOK: AUTO_BEOLVAS
        - automatikusan generált PK
    - HIVAS és KIVONULAS: HIVAS_BEOLVAS
        - a foreign key integritási feltétel miatt itt a fájlok tartalmazzák a primary key értékeket is
        - beolvasásuk emiatt együtt ajánlott – vagy minden adat legyen beolvasott, vagy minden adat véletlen generált

- új hívás felvitele
    - HIVAS_NAPLOZ
        - időpont és hívó szám felvitele, visszaadja az új hívás azonosítóját out paraméterként
- hívás kezelés adminisztrálása
    - HIVAS_RESZLETEK
        - hívó adatainak felvitele
    - DISZPECSER_DONTES
        - döntés felvitele a kivonulásról
- kivonulási napló feltöltése
    - KIVONULAS_FELVESZ
- függvény, mely egy paraméterként adott naphoz megadja a hívások számát.
    - HIVASOK_SZAMA
- eddig hívások lekérdezése személyre vagy időszakra, betegre szűrve
    - HIVAS_SZURES_IDOSZAK: két timestamp közötti hívások
    - HIVAS_SZURES_NEV: adott nevű személy hívásai

Extra feladatok:

- szűrések eredményének lekérése sys_refcursor szerkezetben out paraméterként, ennek feldolgozása JDBC-ben
- KIKULDETESEN_VOLT_E függvény
    - két adott timestamp között egy adott mentős kiküldetésen van/volt-e éppen, visszatérése ’TRUE’ ha igen, ’FALSE’, ha nem
    - egy mentős kiküldetésen van, ha be van osztva olyan kivonuláshoz, amihez tartozó bejelentő hívás időpontja és a kivonulás visszaérkezés időpontja közötti intervallumot metszi a kérdéses intervallum
    - ezt az osztagokat random generáló eljárás is ellenőrzi, mielőtt beoszt egy mentőst egy újabb kivonuláshoz, ha a mentős az adott időszakban már foglalt, akkor nem osztja be
- MAX_KIVONULASI_IDO_PERC függvény
    - mennyi volt a legtöbb időt igénylő kivonulás ideje (a hívás beérkezésétől a mentőautó helyszínre érkezéséig)
- MENTOS_TELEFONSZAM függvény
    - adott azonosítójú mentős telefonszámának lekérdezése

A csomag teljes generáló kódja:

```
CREATE OR REPLACE PACKAGE BODY MENTOSZOLGALAT AS
FUNCTION KIKULDETESEN_VOLT_E(PMENTOS_ID NUMBER, PKEZDET TIMESTAMP, PVEG
TIMESTAMP) RETURN VARCHAR2 AS
   PSZAMLALO NUMBER := 0;
BEGIN
   SELECT COUNT(*) INTO PSZAMLALO FROM HIVASOK H INNER JOIN KIVONULASOK K ON
   H.ID=K.H_ID INNER JOIN OSZTAGOK O ON O.KIVONULAS_ID = K.ID WHERE O.MENTOS_ID =
   PMENTOS_ID AND NOT (PVEG<H.IDOPONT OR PKEZDET>K.VISSZATERES);
   IF PSZAMLALO=0 THEN
      RETURN 'FALSE';
```

```
    ELSE
        RETURN 'TRUE';
    END IF;
END;

FUNCTION HIVASOK_SZAMA(PNAP DATE) RETURN NUMBER AS
    PHIVASDB NUMBER;
BEGIN
    SELECT COUNT(*) INTO PHIVASDB FROM HIVASOK WHERE IDOPONT<PNAP+INTERVAL '1' DAY
    AND IDOPONT>=PNAP;
    RETURN PHIVASDB;
END;

FUNCTION MAX_KIVONULASI_IDO_PERC RETURN INTEGER AS
    PPERC INTEGER := -1;
BEGIN
    SELECT MAX(ROUND((CAST(ERKEZES AS DATE)-CAST(IDOPONT AS DATE))*24*60)) INTO PPERC
    FROM HIVASOK H INNER JOIN KIVONULASOK K ON K.H_ID=H.ID;
    RETURN PPERC;
END;

FUNCTION MENTOS_TELEFONSZAM(PID NUMBER) RETURN VARCHAR2 AS
    PTEL MENTOSOK.TELEFONSZAM%TYPE;
BEGIN
    SELECT TELEFONSZAM INTO PTEL FROM MENTOSOK WHERE ID=PID;
    RETURN PTEL;
EXCEPTION
    WHEN OTHERS THEN RETURN NULL;
END;

PROCEDURE HIVAS_SZURES_IDOSZAK(PKEZDET TIMESTAMP, PVEG TIMESTAMP, KURZOR OUT
SYS_REFCURSOR) AS
BEGIN
    OPEN KURZOR FOR SELECT * FROM HIVASOK WHERE IDOPONT<PVEG AND IDOPONT>=PKEZDET;
END;

PROCEDURE HIVAS_SZURES_NEV(PNEV VARCHAR2, KURZOR OUT SYS_REFCURSOR) AS
BEGIN
    OPEN KURZOR FOR SELECT * FROM HIVASOK WHERE HIVO=PNEV;
END;

PROCEDURE OSZTAG_FELVESZ (PKIVONULASID NUMBER, PMENTOSID NUMBER, PHIBA OUT
INTEGER) AS
    PHIVAS TIMESTAMP;
    PVISSZATERES TIMESTAMP;
BEGIN
    PHIBA:=0;
```

```
    SELECT H.IDOPONT, K.VISSZATERES INTO PHIVAS, PVISSZATERES FROM HIVASOK H INNER JOIN
    KIVONULASOK K ON K.H_ID=H.ID WHERE K.ID=PKIVONULASID;
    IF KIKULDETESEN_VOLT_E(PMENTOSID, PHIVAS, PVISSZATERES) = 'TRUE' THEN
        RAISE_APPLICATION_ERROR(-20000, 'A mentős már máshol volt az adott pillanatban');
    END IF;
    INSERT INTO OSZTAGOK VALUES(PKIVONULASID, PMENTOSID);
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN PHIBA:=-1;
END OSZTAG_FELVESZ;

PROCEDURE OSZTAG_FELTOLT_RANDOM(PHIBA OUT INTEGER) AS
    PMENTOSID NUMBER;
    PKIVONULASID NUMBER;
    PSIKER INTEGER;
BEGIN
    PHIBA:=0;
    FOR I IN (SELECT * FROM KIVONULASOK) LOOP
        FOR J IN 1..3 LOOP
            SELECT ID INTO PMENTOSID FROM (SELECT ID FROM MENTOSOK ORDER BY
            DBMS_RANDOM.RANDOM) WHERE ROWNUM=1;
            PKIVONULASID:=I.ID;
            OSZTAG_FELVESZ(PKIVONULASID, PMENTOSID, PSIKER);
            IF PSIKER!=0 THEN
                PHIBA:=PHIBA+1;
            END IF;
        END LOOP;
    END LOOP;
EXCEPTION
    WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE DELETE_DATA AS
BEGIN
    DELETE FROM OSZTAGOK;
    DELETE FROM KIVONULASOK;
    DELETE FROM HIVASOK;
END;

PROCEDURE KIVONULAS_FELVESZ(PHID NUMBER, PAID NUMBER, PERK TIMESTAMP, PVISSZA
TIMESTAMP, PHIBA OUT INTEGER) AS
    PID NUMBER:=1;
    PDB NUMBER;
BEGIN
    SELECT COUNT(*) INTO PDB FROM KIVONULASOK;
    IF PDB>0 THEN
        SELECT MAX(ID)+1 INTO PID FROM KIVONULASOK;
    END IF;
```

```
    PHIBA:=PID;
    INSERT INTO KIVONULASOK VALUES(PID, PHID, PAID, PERK, PVISSZA);
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE KIVONULAS_BEOLVAS(PHIBA OUT INTEGER) AS
    PFAJL UTL_FILE.FILE_TYPE;
    PSOR VARCHAR2(200);
    PINDEX NUMBER :=0;
    PKIV KIVONULASOK%ROWTYPE;
BEGIN
    PFAJL := UTL_FILE.FOPEN('DIR1','kivonulasok.txt','R');
    PHIBA:=0;
    LOOP
        PINDEX :=0;
        UTL_FILE.GET_LINE(PFAJL,PSOR);
        BEGIN
            FOR I IN (SELECT REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) AS ADAT FROM dual CONNECT BY
            REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) IS NOT NULL) LOOP
                IF PINDEX = 0 THEN
                    PKIV.ID:=TO_NUMBER(I.ADAT);
                ELSIF PINDEX = 1 THEN
                    PKIV.H_ID:=TO_NUMBER(I.ADAT);
                ELSIF PINDEX = 2 THEN
                    PKIV.A_ID:=TO_NUMBER(I.ADAT);
                ELSIF PINDEX = 3 THEN
                    PKIV.ERKEZES:=TO_TIMESTAMP(I.ADAT, 'YYYY.MM.DD. HH24:MI');
                ELSIF PINDEX = 4 THEN
                    PKIV.VISSZATERES:=TO_TIMESTAMP(I.ADAT, 'YYYY.MM.DD. HH24:MI');
                ELSE
                     RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő struktúrájú adatok a fájlban');
                END IF;
                PINDEX:=PINDEX+1;
            END LOOP;
            IF PINDEX!=5 THEN
                 RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő struktúrájú adatok a fájlban');
            END IF;
            INSERT INTO KIVONULASOK VALUES(PKIV.ID, PKIV.H_ID, PKIV.A_ID, PKIV.ERKEZES,
            PKIV.VISSZATERES);
        EXCEPTION
            WHEN OTHERS THEN PHIBA:=PHIBA+1;
        END;
    END LOOP;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        UTL_FILE.FCLOSE(PFAJL);
```

```
        WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE KIVONULAS_FELTOLT_RANDOM(PHIBA OUT INTEGER) AS
    PHID NUMBER;
    PAID NUMBER;
    PERK TIMESTAMP;
    PVISSZA TIMESTAMP;
    PSIKER INTEGER;
BEGIN
    PHIBA:=0;
    FOR I IN (SELECT * FROM HIVASOK WHERE DONTES='I') LOOP
        PHID:=I.ID;
        SELECT ID INTO PAID FROM (SELECT ID FROM AUTOK ORDER BY DBMS_RANDOM.RANDOM)
        WHERE ROWNUM=1;
        PERK:=I.IDOPONT+NUMTODSINTERVAL(FLOOR(dbms_random.value()*40+5), 'MINUTE');
        PVISSZA:=PERK+NUMTODSINTERVAL(FLOOR(dbms_random.value()*60+30), 'MINUTE');
        KIVONULAS_FELVESZ(PHID, PAID, PERK, PVISSZA, PSIKER);
        IF PSIKER==-1 THEN
            PHIBA:=PHIBA+1;
        END IF;
        IF I.SERULTEK_SZAMA>3 THEN
            SELECT ID INTO PAID FROM (SELECT ID FROM AUTOK ORDER BY
            DBMS_RANDOM.RANDOM) WHERE ROWNUM=1;
            PERK:=I.IDOPONT+NUMTODSINTERVAL(FLOOR(dbms_random.value()*40+5), 'MINUTE');
            PVISSZA:=PERK+NUMTODSINTERVAL(FLOOR(dbms_random.value()*60+30), 'MINUTE');
            KIVONULAS_FELVESZ(PHID, PAID, PERK, PVISSZA, PSIKER);
            IF PSIKER!==-1 THEN
                PHIBA:=PHIBA+1;
            END IF;
        END IF;
    END LOOP;
EXCEPTION
    WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE HIVAS_BEOLVAS(PHIBA OUT INTEGER) AS
    PFAJL UTL_FILE.FILE_TYPE;
    PSOR VARCHAR2(200);
    PINDEX NUMBER :=0;
    PHIVAS HIVASOK%ROWTYPE;
BEGIN
    PFAJL := UTL_FILE.FOPEN('DIR1','hivasok.txt','R');
    PHIBA:=0;
    DELETE_DATA;
    LOOP
        PINDEX :=0;
        UTL_FILE.GET_LINE(PFAJL,PSOR);
```

```
        FOR I IN (SELECT REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) AS ADAT FROM dual CONNECT BY
        REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) IS NOT NULL) LOOP
            IF PINDEX = 0 THEN
                PHIVAS.ID:=TO_NUMBER(I.ADAT);
            ELSIF PINDEX = 1 THEN
                PHIVAS.IDOPONT:=TO_TIMESTAMP(I.ADAT, 'YYYY.MM.DD. HH24:MI');
            ELSIF PINDEX = 2 THEN
                PHIVAS.HIVO:=I.ADAT;
            ELSIF PINDEX = 3 THEN
                PHIVAS.TELEFONSZAM:=I.ADAT;
            ELSIF PINDEX = 4 THEN
                PHIVAS.IR_SZAM:=I.ADAT;
            ELSIF PINDEX = 5 THEN
                PHIVAS.CIM:=I.ADAT;
            ELSIF PINDEX = 6 THEN
                PHIVAS.SERULTEK_SZAMA:=TO_NUMBER(I.ADAT);
            ELSIF PINDEX = 7 THEN
                PHIVAS.DONTES:=I.ADAT;
            ELSE
                RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő struktúrájú adatok a fájlban');
            END IF;
            PINDEX:=PINDEX+1;
        END LOOP;
        IF PINDEX!=8 THEN
            RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő struktúrájú adatok a fájlban');
        END IF;
        INSERT INTO HIVASOK VALUES(PHIVAS.ID, PHIVAS.IDOPONT, PHIVAS.HIVO,
        PHIVAS.TELEFONSZAM, PHIVAS.IR_SZAM, PHIVAS.CIM, PHIVAS.SERULTEK_SZAMA,
        PHIVAS.DONTES);
    END LOOP;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        UTL_FILE.FCLOSE(PFAJL);
        KIVONULAS_BEOLVAS(PHIBA);
        IF PHIBA!=0 THEN
            ROLLBACK;
        ELSE
            COMMIT;
        END IF;
    WHEN OTHERS THEN
        PHIBA:=-1;
        ROLLBACK;
END;

PROCEDURE HIVAS_FELVESZ(PIDOPONT TIMESTAMP, PHIVO VARCHAR2, PTEL VARCHAR2, PIR CHAR,
PCIM VARCHAR2, PSERULT NUMBER, PDONTES CHAR, PID OUT NUMBER) AS
    PDB NUMBER;
BEGIN
```

```
    PID:=1;
    SELECT COUNT(*) INTO PDB FROM HIVASOK;
    IF PDB>0 THEN
        SELECT MAX(ID)+1 INTO PID FROM HIVASOK;
    END IF;
    INSERT INTO HIVASOK VALUES(PID, PIDOPONT, PHIVO, PTEL, PIR, PCIM, PSERULT, PDONTES);
    COMMIT;
END;

PROCEDURE DISZPECSER_DONTES(PID NUMBER, PDONTES CHAR, PHIBA OUT INTEGER) AS
    CURSOR C1 IS SELECT * FROM HIVASOK WHERE ID=PID FOR UPDATE;
    REKORD HIVASOK%ROWTYPE;
BEGIN
    PHIBA:=1;
    OPEN C1;
    LOOP
        FETCH C1 INTO REKORD;
        EXIT WHEN C1%NOTFOUND;
        UPDATE HIVASOK SET DONTES=PDONTES WHERE CURRENT OF C1;
        PHIBA:=PHIBA-1;
    END LOOP;
    CLOSE C1;
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE HIVAS_RESZLETEK(PID NUMBER, PHIVO VARCHAR2, PIR CHAR, PCIM VARCHAR2,
PSERULT NUMBER, PHIBA OUT INTEGER) AS
BEGIN
    SELECT 1-COUNT(*) INTO PHIBA FROM HIVASOK WHERE ID=PID;
    UPDATE HIVASOK SET HIVO=PHIVO, IR_SZAM=PIR, CIM=PCIM, SERULTEK_SZAMA=PSERULT
    WHERE ID=PID;
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE HIVAS_NAPLOZ(PIDO TIMESTAMP, PTEL VARCHAR2, PID OUT INTEGER) AS
BEGIN
    HIVAS_FELVESZ(PIDO, NULL, PTEL, NULL, NULL, NULL, NULL, PID);
    COMMIT;
EXCEPTION
    WHEN OTHERS THEN PID:=-1;
END;

PROCEDURE HIVAS_FELTOLT_RANDOM(PNUM INTEGER, PHIBA OUT INTEGER) AS
    PID NUMBER;
```

```
    PIDOPONT TIMESTAMP;
    PHIVO VARCHAR2(50);
    PTEL VARCHAR2(20);
    PIR CHAR(4);
    PCIM VARCHAR2(50);
    PSERULT NUMBER;
    PDONTES CHAR(1);
BEGIN
    PHIBA:=0;
    FOR I IN 1..PNUM LOOP
        PIDOPONT:=TO_TIMESTAMP(TO_CHAR(SYSTIMESTAMP, 'YYYY.MM.DD.
        HH24:MI'),'YYYY.MM.DD. HH24:MI')-
        NUMTODSINTERVAL(FLOOR(dbms_random.value()*525600), 'MINUTE');
        PHIVO:='Kiss Kamil'||I;
        PTEL:='+3670123'||FLOOR(DBMS_RANDOM.VALUE()*9000+1000);
        PIR:='35'|| FLOOR(DBMS_RANDOM.VALUE()*20+20);
        PCIM:='Mucsaröcsöge, Fő utca '|| FLOOR(DBMS_RANDOM.VALUE()*100+1);
        IF DBMS_RANDOM.VALUE()<0.5 THEN
            PSERULT:=FLOOR(DBMS_RANDOM.VALUE()*10+2);
        ELSE
            PSERULT:=1;
        END IF;
        IF DBMS_RANDOM.VALUE()<0.8 THEN
            PDONTES:='I';
        ELSE
            PDONTES:='N';
        END IF;
        HIVAS_FELVESZ(PIDOPONT, PHIVO,PTEL, PIR, PCIM, PSERULT, PDONTES, PID);
    END LOOP;
EXCEPTION
    WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE MENTOS_FELVESZ(PNEV VARCHAR2, PBEOSZTAS VARCHAR2, PTELEFONSZAM
VARCHAR2) AS
    PID NUMBER:=1;
    PDB NUMBER;
BEGIN
    SELECT COUNT(*) INTO PDB FROM MENTOSOK;
    IF PDB>0 THEN
        SELECT MAX(ID)+1 INTO PID FROM MENTOSOK;
    END IF;
    INSERT INTO MENTOSOK VALUES(PID, PNEV, PBEOSZTAS, PTELEFONSZAM);
    COMMIT;
END;

PROCEDURE MENTOS_BEOLVAS(PHIBA OUT INTEGER) AS
    PFAJL UTL_FILE.FILE_TYPE;
```

```
    PSOR VARCHAR2(200);
    PINDEX NUMBER :=0;
    PNEV VARCHAR2(40);
    PBEOSZT VARCHAR2(20);
    PTEL VARCHAR2(20);
BEGIN
    PFAJL := UTL_FILE.FOPEN('DIR1','mentosok.txt','R');
    PHIBA:=0;
    LOOP
        UTL_FILE.GET_LINE(PFAJL,PSOR);
        BEGIN
            PINDEX :=0;
            FOR I IN (SELECT REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) AS ADAT FROM dual CONNECT BY
            REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) IS NOT NULL) LOOP
                IF PINDEX = 0 THEN
                    PNEV:=I.ADAT;
                ELSIF PINDEX = 1 THEN
                    PBEOSZT:=I.ADAT;
                ELSIF PINDEX = 2 THEN
                    PTEL:=I.ADAT;
                ELSE
                    RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő struktúrájú adatok a fájlban');
                END IF;
                PINDEX:=PINDEX+1;
            END LOOP;
            IF PINDEX!=3 THEN
                RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő struktúrájú adatok a fájlban');
            END IF;
            MENTOS_FELVESZ(PNEV, PBEOSZT, PTEL);
        EXCEPTION
            WHEN OTHERS THEN PHIBA:=PHIBA+1;
        END;
    END LOOP;
EXCEPTION
    WHEN NO_DATA_FOUND THEN UTL_FILE.FCLOSE(PFAJL);
    WHEN OTHERS THEN PHIBA:=-1;
END;

PROCEDURE AUTO_FELVESZ(PRENDSZAM VARCHAR2, PMARKA VARCHAR2, PTIPUS VARCHAR2) AS
    PID NUMBER:=1;
    PDB NUMBER;
BEGIN
    SELECT COUNT(*) INTO PDB FROM AUTOK;
    IF PDB>0 THEN
        SELECT MAX(ID)+1 INTO PID FROM AUTOK;
    END IF;
    INSERT INTO AUTOK VALUES(PID, PRENDSZAM, PMARKA, PTIPUS);
    COMMIT;
```

```
END;

PROCEDURE AUTO_BEOLVAS(PHIBA OUT INTEGER) AS
    PFAJL UTL_FILE.FILE_TYPE;
    PSOR VARCHAR2(200);
    PINDEX NUMBER :=0;
    PRSZ VARCHAR2(40);
    PMARKA VARCHAR2(20);
    PTIPUS VARCHAR2(20);
BEGIN
    PHIBA:=0;
    PFAJL := UTL_FILE.FOPEN('DIR1','autok.txt','R');
    LOOP
        PINDEX :=0;
        UTL_FILE.GET_LINE(PFAJL,PSOR);
        BEGIN
            FOR I IN (SELECT REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) AS ADAT FROM dual CONNECT BY
            REGEXP_SUBSTR (PSOR, '[^;]+', 1, level) IS NOT NULL) LOOP
                IF PINDEX = 0 THEN
                    PRSZ:=I.ADAT;
                ELSIF PINDEX = 1 THEN
                    PMARKA:=I.ADAT;
                ELSIF PINDEX = 2 THEN
                    PTIPUS:=I.ADAT;
                ELSE
                    RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő fájl szerkezet.');
                END IF;
                PINDEX:=PINDEX+1;
            END LOOP;
            IF PINDEX!=3 THEN
                RAISE_APPLICATION_ERROR(-20000, 'Nem megfelelő fájl szerkezet.');
            END IF;
            AUTO_FELVESZ(PRSZ, PMARKA, PTIPUS);
        EXCEPTION
            WHEN OTHERS THEN PHIBA:=PHIBA+1;
        END;
    END LOOP;
EXCEPTION
    WHEN NO_DATA_FOUND THEN UTL_FILE.FCLOSE(PFAJL);
    WHEN OTHERS THEN PHIBA:=-1;
END;

END;
```

JDBC hívások:

A csomag interfészéhez tartozó eljárások és függvények tesztelésére írt java programban bejelentkezést követően lehetőség van autók és mentősök beolvasására fájlból, hívások és

kivonulások együttes beolvasására fájlból (az eddigi tartalom törlését követően, ugyanis ezek a fájlok már konkrét kulcsértékeket is tartalmaznak), hívások, kivonulások és osztagok random generálására. Ezen kívül 3 lépésben (napló, részletek, döntés a kivonulásról) fel lehet venni új hívást, továbbá új kivonulást is az autóban tartózkodó mentősökkel együtt (az osztagok táblába is rekordok kerülnek). Lehetőség van lekérdezni a leghosszabb kivonulás hosszát, az adott napi hívások számát, az adott időszakhoz vagy bejelentő személyhez tartozó hívások adatait, egy adott mentős telefonszámát, valamint azt, hogy egy adott mentős egy adott időszakban volt-e kiküldetésen.

A program osztályai:

- AmbulanceDB: a tárolt eljárások és függvények hívását, az adatbázis kapcsolódást és az eredmények feldolgozását tartalmazza
- Runner: program belépési pont
- MenuHandler: a konzolos felület kezelőosztálya
- InputHandler: az alapvető típusok beolvasását kezelő segédosztály