**OE NIK 2013**

**Táborhely Üzemeltető segédszoftver**

**Teljes dokumentáció**

**Lovas István K971UM**

# Tartalom Jegyzék:

Táborhely Üzemeltető segédszoftver dokumentáció.

# A Szoftver céljai és megszorításai

A web-es adatbázis-alkalmazás célja az, hogy a tábor személyzetének megkönnyítse a tábor lakóival kapcsolatos adminisztrációt. A nyilvántartásban tárolja a lakók, a tábori csoportok, illetve a tábori férőhelyekkel kapcsolatos legfontosabb adatokat. A rendszer adminisztrátorának, az SQL futtatását megkönnyítő szakasszal is rendelkezik, illetve egy olyan részegységgel, mely információt biztosít a rendszer környezetének OpenCl képességeiről.

# A Szoftverrel kapcsolatos technikai elvárások és megszorítások

Az elkészítés során a következő nyelveket és technológiákat alkalmaztam:

* Az alkalmazás az Oracle11g APEX környezetének felhasználásával készült.
  + Oracle11g:
    - Egy nagyvállalati környezetbe készült, és ilyen környezetben közkedvelt adatbázis kezelő eszköz.
  + APEX (Application Express):
    - Az Oracle RAD eszköze, mely adatbázis intenzív webes alkalmazások egyszerű elkészítésére hivatott.
* SQL - Structured Query Language (strukturált lekérdezőnyelv):
  + Mely egy olyan deklaratív nyelv mely az adatbázis kezelés megkönnyítésére hivatott.
* PL/SQL - Procedural Language/Structured Query Language:
  + Az Oracle által kifejlesztett programozási nyelv, mely az SQL procedurális eszközökkel való kiegészítése.
* Java:
  + Egy olyan általános célú, objektumorientált programozási nyelv, aminek tervezése és megalkotása során nagy figyelmet fordítottak a robosztus, hibatűrő, biztonságos működésre, és hogy lehetőleg kevés ráfordítással lehessen, könnyen hordozható, platform független, vállalati szintű programokat előállítani a segítségével.
  + A fenti előnyei, és a tény, hogy jól együttműködik az Oracle adatbázis kezelő rendszerrel, kiváló választássá tette az adott feladatra.
* Aparapi:
  + Egy olyan Java nyelvi programozási felület (API), mely segíti a külső gyorsító eszközök programozását.
  + Tervezésénél ügyeltek arra, hogy használata ne legyen idegen egy Java programozó számára. Ezt olyannyira sikeresen teljesítették, hogy egy egyszerű kliens megírása nem sokkal bonyolultabb egy Java szál indításánál.
  + Az Oracle bejelentette, hogy a Java 9 OpenCl képességeit az Aparapi mintájára, annak eredményeit felhasználva kívánják implementálni.
  + Az Aparapi ezért egy nagyon előremutató és jó eszköz a kívánt cél elérésére.
* NetBeans:
  + Egy nyílt forráskódú, de az Oracle által is koordinált fejlesztésű IDE, a Java kódok előállítását segítette.

# A Szoftver beüzemelése és tesztadatokkal való feltöltése

A program beüzemelésének lépései:

* A program vázának elkészítése az f102.sql APEX-be való importálásával kezdődik.
  + f102.sql
* A táblák elkészítése:
  + Csak scriptek segítségével:
    1. 1a\_create\_tables.sql
    2. 1b\_alter\_constraints.sql
    3. 2a\_init\_test\_Házak.sql
    4. 2b\_init\_test\_Csoportok.sql
    5. 2c\_init\_test\_Lakók.sql
  + Az APEX SQL Workshop > Utilities > Data Workshop eszközével
    1. Data Load > Text Data eszközzel importálandó fájlok:
       - csoportok.txt
       - házak.txt
       - lakók.txt
    2. 1b\_alter\_constraints.sql script futtatása, a megszorítások létrehozása végett
* A Java nyelvű külső rutin beüzemelése:

Az összes fent említett script, és importálandó fájl megtalálható a   
CD mellékélet gyökér\Scriptek\_Programok\_Fájlok\ mappában.

# Jogosultsági szintek a rendszerben

# Betétlap

BETÉTLAP

Lovas István Féléves Feladatához

1. Autentikációs adatok:

- adminisztrátor / adminisztrátor

- dolgozó / dolgozó

2. A Féléves Feladat kiemelkedő képességei:

- Gyorsító eszközök Java nyelvből történő elérése

# Táblaszerkezet tervezése:

## A feladat rövid leírása

* Egy ifjúsági tábor lakóival kapcsolatos fontosabb adatok tárolása.

## A tárolandó tulajdonságok leírása

|  |  |
| --- | --- |
| **A tárolandó adat** | **Rövid leírása** |
| név | a lakó neve |
| SzületésiDátum | a lakó születési dátuma |
| nem | a lakó neme |
| egészségbiztosítási\_szám | a lakó egészségbiztosítási száma |
| személyigazolvány\_szám | a lakó személyigazolvány száma |
| tartózkodas\_tól | az a dátum amikortól az adott lakó a táborban tartózkodik |
| tartózkodas\_ig | az a dátum ameddig az adott lakó a táborban tartózkodik |
| ország | a lakó származási országa |
| ház\_szám | a ház száma,a táboron belül a házak egytől számozódnak |
| ház\_megnevezés | a ház neve, a táboron belül lehet több azonos nevű ház is. |
| férőhely\_szám | egy szám, mely megmondja, hogy az adott ház hány személyt képes befogadni |
| csoport\_név | a tábor lakói csoportokat képeznek, táboron belül több csoportnak is lehet azonos a neve |
| csoportvezető | minden csoportnak csak egy vezetője van, de egy vezetőnek lehet több csoportja |
| csoport\_pontszám | egy-egy csoport által, a csapatversenyeken elért, összpontszámát tárolja |

## Az ősmodell:

* Lakók< név, SzületésiDátum, nem, egészségbiztosítási\_szám, személyigazolvány\_szám, tartózkodas\_tól, tartózkodas\_ig, ország, ház\_szám, ház\_megnevezés, férőhely\_szám, csoport\_név, csoportvezető, csoport\_pontszám >

## Függőségek:

1. Egy lakó pontos meghatározásához szükséges a neve és az egészségbiztosítási száma. **flakók** ={név, egészségbiztosítási\_szám } -> { SzületésiDátum, nem, személyigazolvány\_szám, tartózkodas\_tól, tartózkodas\_ig, ország, ház\_szám, ház\_megnevezés, férőhely\_szám, csoport\_név, csoportvezető, csoport\_pontszám }
2. Egy Csoport egyértelmű meghatározásához szükséges, a csoport neve, illetve a vezetője.  
   **fcsoportok** = { csoport\_név, csoportvezető } -> { csoport\_pontszám }
3. A házszámok különbözőek, ezért a házszámból egyértelműen képes megállapítani a rendszer, hogy melyik házról van szó. Megengedjük viszont, hogy több ház viselje ugyanazt a nevet.  
   **fházak** = { ház\_szám } -> { ház\_megnevezés, férőhely\_szám }

## A függőségek normalizálása

1. **flakók** normalizálása:
   1. 1NF-ben van, mert nincs összetett attribútuma
   2. 2NF-ben van, nincs olyan másodlagos attribútum, aminek a meghatározásához elegendő lenne az elsődleges attribútumok egy része.
   3. Nincs 3NF-ben, hisz **fcsoportok** és **fházak** belső függések, ezeket kiemelve:  
      **flakók1** = {név, egészségbiztosítási\_szám } -> { SzületésiDátum, nem, személyigazolvány\_szám, tartózkodas\_tól, tartózkodas\_ig, ország, ház\_szám, csoport\_név, csoportvezető }
   4. BCNF-ben van, hisz 3NF-ben van, és a baloldal egyetlen valódi részhalmaza sem függ másik elsődleges, vagy másodlagos attribútumtól.
2. **fcsoportok** normalizálása:
   1. 1NF-ben van, hisz nincs összetett attribútuma.
   2. 2NF-ben van, hisz a jobb oldal egy attribútumból áll, és az függ mindkét elsődleges attribútumtól.
   3. 3NF-ben van, hisz a jobb oldal csak egy elemből áll.
   4. BCNF-ben van, hisz az elsődleges attribútumok nem függnek egymástól, vagy a másodlagos attribútumtól.
3. **fházak** normalizálása:
   1. 1NF-ben van, hisz nincs összetett attribútuma.
   2. 2NF-ben van, hisz a baloldal csak egyetlen attribútumból áll.
   3. 3NF-ben van, hisz a másodlagos attribútumok között nincs összefüggés.
   4. BCNF-ben van, hisz 3NF-ben van, és a bal oldalán csak egyetlen attribútum áll.

## Az eredményfüggőségek összefoglalása

* **flakók1** = { név, egészségbiztosítási\_szám } -> { SzületésiDátum, nem, személyigazolvány\_szám, tartózkodas\_tól, tartózkodas\_ig, ország, ház\_szám, csoport\_név, csoportvezető }
* **fcsoportok** = { csoport\_név, csoportvezető } -> { csoport\_pontszám }
* **fházak** = { ház\_szám } -> { ház\_megnevezés, férőhely\_szám }

## Az eredmény adatmodell felírása

* Lakók < név, egészségbiztosítási\_szám, SzületésiDátum, nem, személyigazolvány\_szám, tartózkodas\_tól, tartózkodas\_ig, ország, *ház\_szám*, *csoport\_név, csoportvezető* >
  + Ahol a név és egészségbiztosítási\_szám attribútumok,   
    tehát a **Klakók** = { név, egészségbiztosítási\_szám } attribútum halmaz a Lakók relációra vonatkozó elsődleges kulcs.
  + A *ház\_szám attribútum a házak relációra vonatkozó idegen kulcs.*
  + *A* *csoport\_név és csoportvezető attribútumok pedig a csoportok relációra vonatkozó idegen kulcsok.*
* Csoportok *<*csoport\_név, csoportvezető, csoport\_pontszám>
  + A **Kcsoportok** = { csoport\_név, csoportvezető } attribútum halmaz a Csoportok relációra vonatkozó elsődleges kulcs.
* Házak < ház\_szám, ház\_megnevezés, férőhely\_szám >

A **Kházak** = { ház\_szám } attribútum halmaz a Házak relációra vonatkozó elsődleges kulcs.

# Táblák Implementációja

## A táblák szerkezete

A táblák tervben foglaltak szerint készültek.

A kívánt szöveges és szám mezők meghatározásakor, ahol tudtam, a lehető legnagyobb, még ésszerű foglalási méretet választani. Ezt néhol kutatás is megelőzte, például az ország név mező hosszának meghatározásakor számításba vettem ilyeneket, mint: The United Kingdom of Great Britain and Northern Ireland, mely 48 karakter hosszú. Az ország mezőt 50 hosszú VARCHAR2-ként vettem fel. A VARCHAR2 típusról tudni kell, hogy nem foglalódik le alapból a teljes meghatározott karakternyi méret (előző példában 50), ezért úgy láttam, érdemesebb túltervezni, ahol csak lehet. Az Egészségbiztosítási\_szám, illetve a Személyigazolvány\_szám fix hosszúságú szövegek, ennek megfelelő CHAR típusnak határoztam meg mind a kettőt. A szám típusoknál, mint a ház\_szám, illetve a férőhely\_szám három helyértéken tárol, a csoport pontszám egy tizedes jegy pontossággal, és öt egész helyértékkel tárol, ami a táborban minden bizonnyal elég lesz.

### A megvalósító script: 1a\_create\_tables.sql

ALTER SESSION SET NLS\_LANGUAGE= 'HUNGARIAN';

ALTER SESSION SET NLS\_TERRITORY= 'HUNGARY';

ALTER SESSION SET NLS\_DATE\_FORMAT= 'YYYY-MM-DD';

ALTER SESSION SET NLS\_DATE\_LANGUAGE = 'HUNGARIAN';

ALTER SESSION SET NLS\_ISO\_CURRENCY = 'HUNGARY';

ALTER SESSION SET NLS\_NUMERIC\_CHARACTERS = ',.';

ALTER SESSION SET NLS\_SORT = 'HUNGARIAN';

--SELECT \* FROM nls\_session\_parameters;

DROP TABLE Lakók;

CREATE TABLE Lakók (

Név VARCHAR2(30),

Egészségbiztosítási\_szám CHAR(9) UNIQUE,

SzületésiDátum DATE,

Nem CHAR(1) CHECK( Nem IN ('f','n') ),

Személyigazolvány\_szám CHAR(11) UNIQUE,

Tartózkodas\_tól DATE,

Tartózkodas\_ig DATE,

Ország VARCHAR2(50),

ház\_szám NUMBER(4),

Csoport\_név VARCHAR2(50),

Csoportvezető VARCHAR2(30)

);

DROP TABLE Csoportok;

CREATE TABLE Csoportok(

csoport\_név VARCHAR2(50),

csoportvezető VARCHAR2(30),

csoport\_pontszám NUMBER(5,1)

);

DROP TABLE Házak;

CREATE TABLE Házak (

ház\_szám NUMBER(3),

ház\_megnevezés VARCHAR2(25) NOT NULL,

férőhely\_szám NUMBER(3) NOT NULL

);

A fentiekben látható, hogy a Lakók táblában az Egészségbiztosítási\_szám, amellett, hogy az elsődleges kulcs része, még egyedinek is jelöltem. Egyedi továbbá a Személyigazolvány\_szám mező is. Az egészségbiztosítási szám, mivelhogy az elsődleges kulcs része, kötelező a kitöltése is. A Személyigazolvány\_szám kitöltése ellenben nem kötelező, hisz lehetséges, hogy kiskorú személy is tartózkodik a tábor területén.

## A táblák megszorításai

A terv alapján megalkottam az elsődleges, illetve az idegen kulcsokat.

### A megvalósító script: 1b\_alter\_constraints.sql

ALTER TABLE Lakók ADD CONSTRAINTS p\_Lakók PRIMARY KEY (Név,Egészségbiztosítási\_szám);

ALTER TABLE Csoportok ADD CONSTRAINTS p\_Csoportok PRIMARY KEY (csoport\_név,csoportvezető);

ALTER TABLE Házak ADD CONSTRAINTS p\_Házak PRIMARY KEY (ház\_szám);

ALTER TABLE Lakók ADD CONSTRAINTS f\_Lakók\_Csoportok FOREIGN KEY (Csoport\_név,Csoportvezető)

REFERENCES Csoportok (csoport\_név,csoportvezető);

ALTER TABLE Lakók ADD CONSTRAINTS f\_Lakók\_Házak FOREIGN KEY (ház\_szám)

REFERENCES Házak (ház\_szám);

# Implementáció részleteinek leírása

Általános megjegyzések

## Kezdőlap

### A komponens célja

A kezdőlap általános leírást nyújtani a webalkalmazás céljairól, és használatáról.

A kezdőoldal bárki számára látható.

### A megvalósításának fontosabb részletei

Az információt HTML egyszerű régiókban tárolja.  
Azt, hogy az oldalt bárki láthassa a Kezdőlap > Page Attributes > Security > Authentication választómezőjének Page Is Public értékre állításával értem el.

### Futási eredménye

## Táblák Megtekintése

### A komponens célja

Könnyen kezelhető eszköz a rendszerben szereplő táblák megjelenítésére.

Táblaválasztó súgószövege:  
Válasszon táblát, majd jelölje be a megjelenítendő oszlopokat, és ha szükséges, megadhat egy oszlopot, mely szerint az adatokat rendezi a rendszer.

Csak dolgozók láthatják.

### A megvalósításának fontosabb részletei

* A táblát egy legördülő listából lehet választani
  + értéklista lekérdezése:  
    select distinct table\_name, table\_name as "a" from user\_tables  
    where table\_name not like 'APEX%' AND table\_name not like 'DEMO%';
* A tábla kiválasztása után az adott tábla oszlopait checkbox segítségével lehet kiválasztani:
  + értéklista lekérdezése:  
    select column\_name as val, column\_name from all\_tab\_columns where table\_name = :P2\_TV
* A rendezést rádiógomb segítségével lehet kiválasztani:
  + értéklistája:  
    STATIC:Növekvő;ASC,Csökkenő;DESC,Nincs;NO
* A rendező oszlopot egy automatikus szövegkiegészítést nyújtó szövegdobozban lehet megadni
  + értéklista lekérdezése: megegyezik a checkbox lekérdezésével.
* Az oldal elküldése után két validáció futhat le, az egyik a kiválasztott tábla változtatásánál nulláz minden attól függő mezőt, míg a másik lekérdez gomb hatására ellenőrzi, hogy minden szükséges mezőt kitöltött-e a felhasználó.
* A lekérdezés szövegét egy folyamat állítja össze egy rejtett mezőbe.

DECLARE

tabla VARCHAR2(50);

oszlopok VARCHAR2(200);

sql\_str VARCHAR2(300);

rendez VARCHAR2(50);

BEGIN

:P2\_HID\_SQL\_CONTAINER := '';

tabla := :P2\_TV;

oszlopok:= apex\_util.table\_to\_string(apex\_util.string\_to\_table(:P2\_CHK\_OSZLOPOK,':'),',');

if oszlopok is null then

oszlopok := '\*';

end if;

if :P2\_RDO\_SORT != 'NO' then

rendez:='ORDER BY' || ' ' || :P2\_TB\_SORT\_FIELD || ' ' || :P2\_RDO\_SORT;

end if;

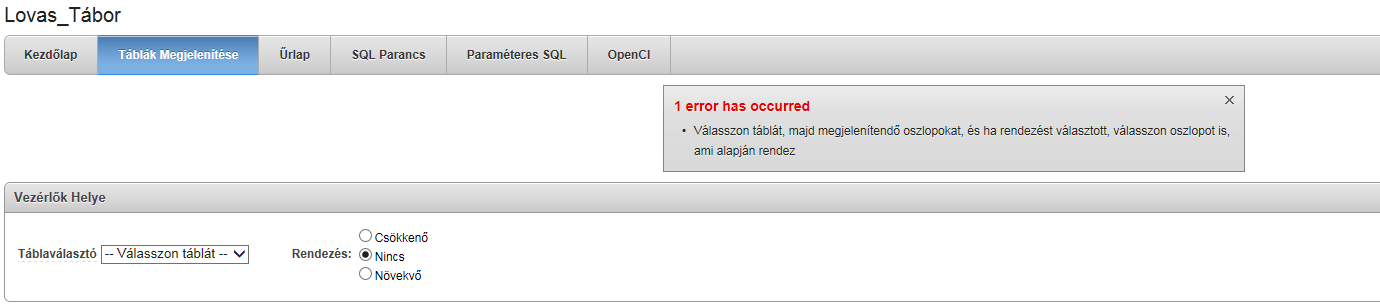
sql\_str := 'SELECT '|| oszlopok ||' FROM '|| tabla || ' ' || rendez;

:P2\_HID\_SQL\_CONTAINER := sql\_str;

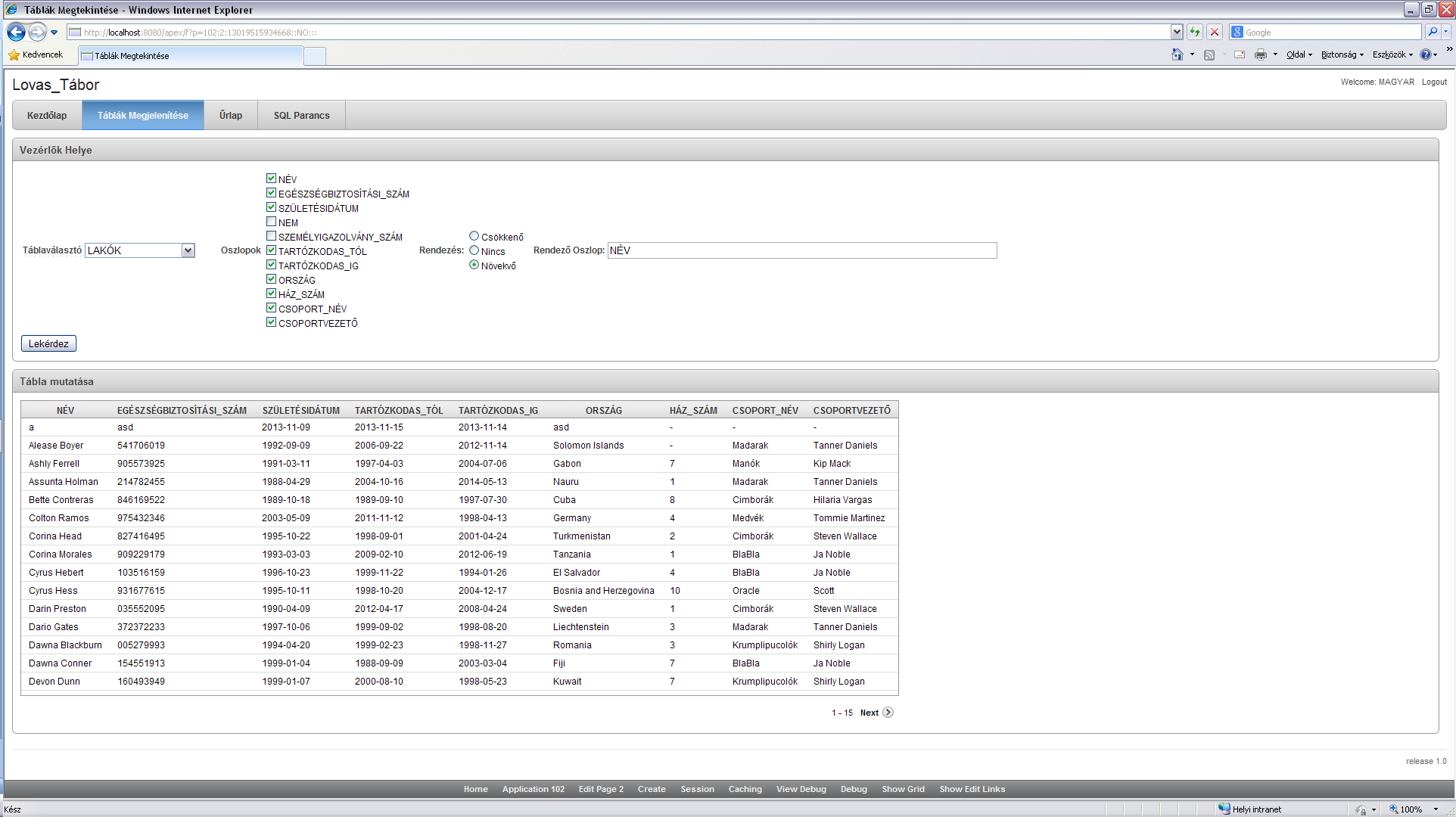
END;

### Futási eredménye

Az alábbi kép bemutatja azt az esetet, amikor a felhasználó nem tölti ki a megfelelő mezőket, látható a magyar nyelvű figyelmeztető szöveg.



Az alábbi kép a helyes működést ábrázolja.



## Űrlap

### A komponens célja

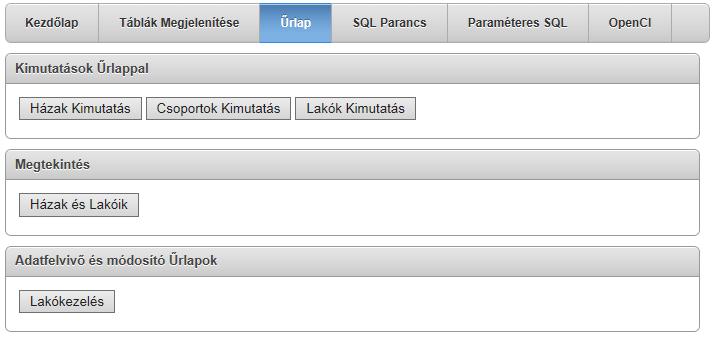
Innen érhetik el a dolgozók a rendszer által nyújtott űrlapokat (form) és kimutatásokat (report)

### A megvalósításának fontosabb részletei

A különböző alegységeket régiók különítik el, és a régiókban gombok vannak, melyek segítségével el lehet jutni a konkrét űrlaphoz vagy kimutatáshoz.

Az alegységeket külön-külön részletezem.

### Futási eredménye



## Kimutatások Űrlappal

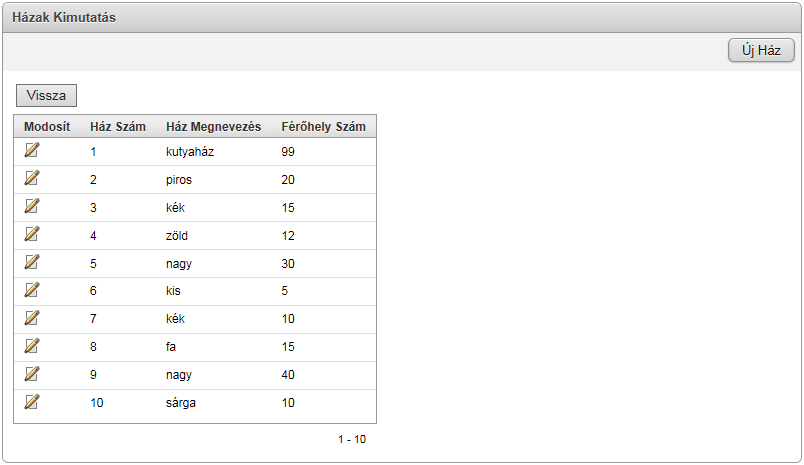
### A komponens célja

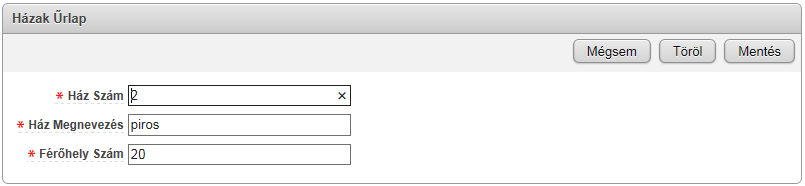
Az alegység egyszerű felületet biztosít a házak, csoportok, és lakók kezelésére.

### A megvalósításának fontosabb részletei

Az alegység összes lapja az APEX Form on Table with Report eszközével készült. A kimutatási oldal és az űrlap oldal kapcsolatát mindenhol a rowid segítségével teremtem meg, mely kényelmesebb kezelést biztosít összetett kulcsok, esetén, továbbá kiküszöböli a környezet szöveges kulcsok getes paraméterként való küldésénél fellépő karakterkódolási problémáját. Az űrlap oldalakon, amennyiben dátum mező nevében ékezet volt, a generált item nevét átírtam ékezetmentesre, a DateTime picker ilyenkor fellépő hibájának kiküszöbölése végett. A generált szövegdobozokat ahol lehetett, és célszerű volt, legördülő listára cseréltem.

### Futási eredménye





A fenti képeken a Ház Kimutatás látható, a Csoportok illetve a Lakók Kimutatás is hasonlóképp működik.

## Megtekintés -> Házak és lakóik

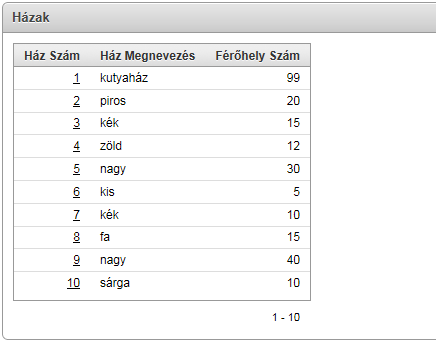
### A komponens célja

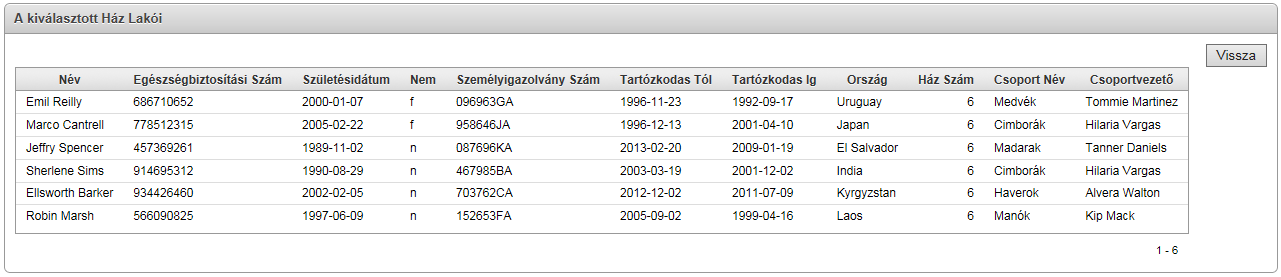
Az alegység házak és a kiválasztott ház lakóinak egyszerű megjelenítését szolgálja.

### A megvalósításának fontosabb részletei

Droll Down módszert használtam, az APEX Wizard Report eszközének segítségével. A kapcsolatot megteremtő házszám mező kiválasztott értékét égy rejtett mezőben tároltam.

### Futási eredménye





## Adatfelvivő és módosító Űrlapok -> Lakókezelés

### A komponens célja

A lakókról tárolt nagy mennyiségű információra és a lakók nagy számára való tekintettel, a lakók kezelésére, ez az alrendszer, egy a Lakók Kimutatásnál látottnál hatékonyabb eszközt biztosít a dolgozók részére.

### A megvalósításának fontosabb részletei

A megvalósítás során az APEX Tabular Form eszközét használtam. A generált szövegdobozokat ahol lehetett, és célszerű volt, legördülő listára cseréltem. A rendszerben egy csoportot vezetője és neve együttesen határoz meg, ez a szakasz viszont a felhasználók már meglévő csoportokhoz rendelésére is használható. Ezért úgy döntöttem, hogy a már meglévő csoportnév - csoportvezető kombinációkat egyetlen legördülő listában jelenítem meg a felhasználó munkájának megkönnyítésére.

A Csoportok legördülő lista értéklistáját az alábbi lekérdezés adja:

SELECT "CSOPORT\_NÉV" || ' - ' || "CSOPORTVEZETŐ" AS "Csoportnev", "CSOPORT\_NÉV" || ' - ' || "CSOPORTVEZETŐ" AS "Csoportval" FROM csoportok

Ennek megfelelően a feldolgozást is át kellett írni:

Declare

csap\_n Varchar(500);

csap\_v Varchar(500);

idx NUMBER(4);

len NUMBER(4);

BEGIN

csap\_n := '';

csap\_v := '';

for i in 1..apex\_application.g\_f02.count loop

--Coportnév - Csopotrvezető

csap\_n := '';

csap\_v := '';

idx := instr(apex\_application.g\_f11(i),'-');

csap\_n := SUBSTR(apex\_application.g\_f11(i),1,idx-2);

len := length(apex\_application.g\_f11(i));

len := len - (idx+1);

csap\_v := SUBSTR(apex\_application.g\_f11(i),idx+2,len);

if apex\_application.g\_frowid(i) is not null then

--Update

UPDATE Lakók

set

NÉV = apex\_application.g\_f02(i),

EGÉSZSÉGBIZTOSÍTÁSI\_SZÁM = apex\_application.g\_f03(i),

SZÜLETÉSIDÁTUM = TO\_DATE(apex\_application.g\_f04(i),'YYYY-MM-DD'),

NEM = apex\_application.g\_f05(i),

SZEMÉLYIGAZOLVÁNY\_SZÁM = apex\_application.g\_f06(i),

TARTÓZKODAS\_TÓL = TO\_DATE(apex\_application.g\_f07(i),'YYYY-MM-DD'),

TARTÓZKODAS\_IG = TO\_DATE(apex\_application.g\_f08(i),'YYYY-MM-DD'),

ORSZÁG = apex\_application.g\_f09(i),

HÁZ\_SZÁM = apex\_application.g\_f10(i),

CSOPORT\_NÉV = csap\_n,

CSOPORTVEZETŐ = csap\_v

WHERE rowid = apex\_application.g\_frowid(i);

else

--insert

INSERT INTO Lakók

(

Név,

Egészségbiztosítási\_szám,

SzületésiDátum,

Nem,

Személyigazolvány\_szám,

Tartózkodas\_tól,

Tartózkodas\_ig,

Ország,

ház\_szám,

Csoport\_név,

Csoportvezető

)

VALUES

(

apex\_application.g\_f02(i),

apex\_application.g\_f03(i),

TO\_DATE(apex\_application.g\_f04(i),'YYYY-MM-DD'),

apex\_application.g\_f05(i),

apex\_application.g\_f06(i),

TO\_DATE(apex\_application.g\_f07(i),'YYYY-MM-DD'),

TO\_DATE(apex\_application.g\_f08(i),'YYYY-MM-DD'),

apex\_application.g\_f09(i),

apex\_application.g\_f10(i),

csap\_n,

csap\_v

)

;

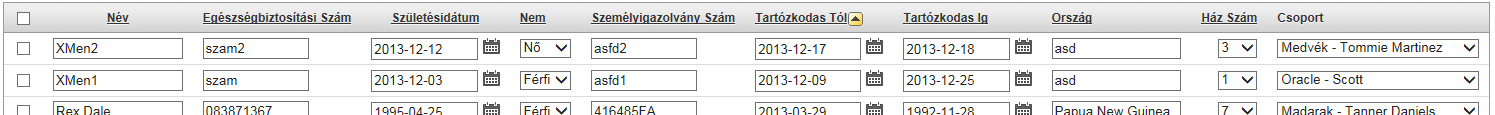
end if;

end loop;

END;

### Futási eredménye





Az fenti képek bemutatják, két sor egyszerre történő sikeres felvételét.

## SQL Parancs

### A komponens célja

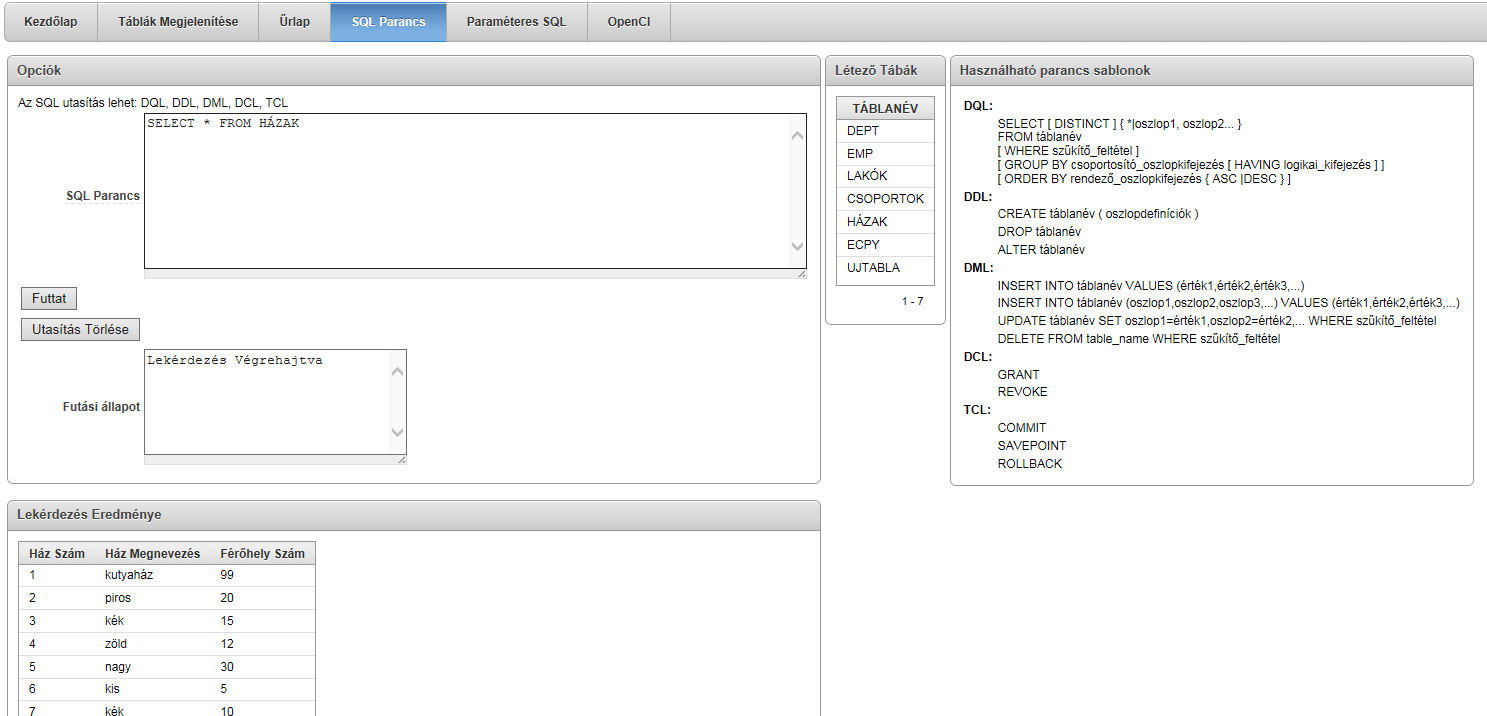
Az adminisztrátoroknak biztosít SQL parancsfuttatási lehetőséget. A komponens a parancs megírását parancssablonokkal és a felhasználó tábláinak a megjelenítésével segíti.

### A megvalósításának fontosabb részletei

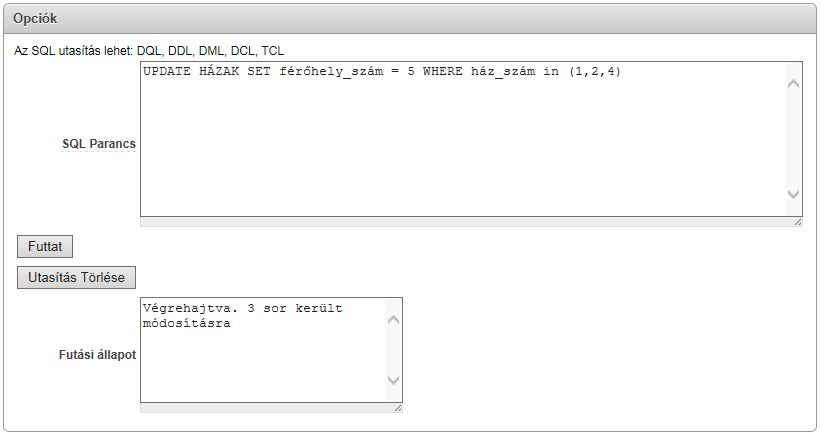
A felhasználói parancs végrehajtására az EXECUTE IMMEDIATE PL/SQL parancsot használtam. Amennyiben a parancs eleje, nagybetűssé alakítva, SELECT, akkor a lekérdezést átadom egy Riportnak, ellenkező esetben az SQL%ROWNUM segítségével megadom a módosított sorok számát.

### Futási eredménye

Egy lekérdezés eredménye:



Egy módosító SQL szkript végrehajtása:



## Paraméteres SQL

### A komponens célja

Az adminisztrátoroknak biztosít dinamikus SQL összeállítási, és futtatási módot.

### A megvalósításának fontosabb részletei

Az SQL Típusa legördülő lista alapján jelennek meg, az adott parancs összeállítására szolgáló beviteli elemek. A kötelezően kitöltendő beviteli mezők kitöltését ellenőrzöm, az összetett parancs helyességét viszont az adatbázis által kiváltott kivételek segítségével ellenőrzöm. Ezeket a kivételeket elkapás után minden esetben továbbdobom, hogy a felületen is megjelenjen, hisz ez az alegység adminisztrátoroknak szól, kik így minden információt megkaphatnak, mely segítheti feladatuk hibamentes megoldását. Az összes rész az SQL dinamikus összeállításával és annak EXECUTE IMMEDIATE paranccsal való futtatásával végzi a dolgát, kivéve az alább részletezett „Új sor felvétele a kiválasztott táblába” nevű részfunkció.

#### Új sor felvétele a kiválasztott táblába

Egy kiválasztott tábla hatására dinamikusan listázásra kerül minden oszlopnak megfelelően egy beviteli mező:

FOR i in (SELECT DISTINCT column\_name from USER\_TAB\_COLUMNS where table\_name = :P15\_ACT\_TABLE)

LOOP

htp.p('<div style="min-width:200px; display: inline-block;">' || i.column\_name ||' </div> '||APEX\_ITEM.TEXT(50) || '<br />' );

END LOOP;

htp.p('<br />');

A feldolgozásnál felhasználom az Oracle DBMS\_SQL csomagját:

DECLARE

str VARCHAR2(1000);

vals VARCHAR2(500);

TYPE idxt IS TABLE OF NUMBER INDEX BY PLS\_INTEGER;

j NUMBER(2);

idx\_index NUMBER(2);

inx\_es idxt;

curid INTEGER;

ret INTEGER;

BEGIN

if :P15\_ACT\_TABLE is null

then

raise\_application\_error(-20001, 'Adjon meg egy kezelendő táblát (Kijelőlt tábla)');

end if;

str := 'INSERT INTO ' || :P15\_ACT\_TABLE || '( ';

vals := 'VALUES ( ';

j := 0;

idx\_index := 0;

FOR i in (SELECT DISTINCT column\_name from USER\_TAB\_COLUMNS where table\_name = :P15\_ACT\_TABLE)

LOOP

j := j + 1;

if apex\_application.g\_f50(j) is not null

then

if idx\_index > 0

then

str := str || ', ';

vals := vals || ', ';

end if;

idx\_index := idx\_index + 1;

str := str || i.column\_name;

vals := vals || ':p'|| idx\_index;

inx\_es(idx\_index) := j;

end if;

END LOOP;

str := str || ' ) ';

str := str || vals || ' )';

curid := DBMS\_SQL.OPEN\_CURSOR;

DBMS\_SQL.PARSE(curid, str, DBMS\_SQL.NATIVE);

for i in inx\_es.FIRST..inx\_es.LAST

loop

DBMS\_SQL.BIND\_VARIABLE(curid, 'p' || i, apex\_application.g\_f50(inx\_es(i)));

end loop;

ret := DBMS\_SQL.EXECUTE(curid);

DBMS\_SQL.CLOSE\_CURSOR(curid);

:P15\_GEN := str;

:P15\_EREDMENY := 'Sikeres végrehajtás';

EXCEPTION WHEN OTHERS THEN

IF DBMS\_SQL.IS\_OPEN(curid) THEN

DBMS\_SQL.CLOSE\_CURSOR(curid);

END IF;

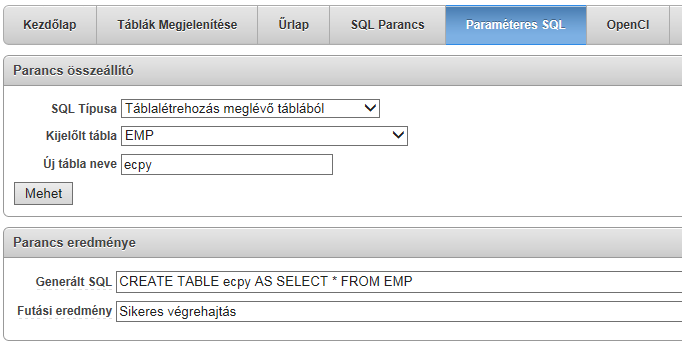
:P15\_EREDMENY := 'A végrehajtás ismeretlen okból megszakadt, ellenőrizze a lekérdezést és próbálja újra';

RAISE;

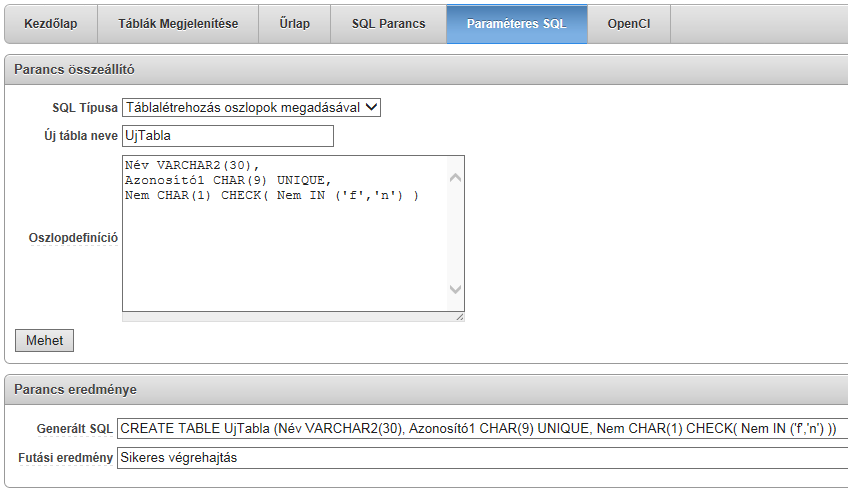
END;

### Futási eredménye

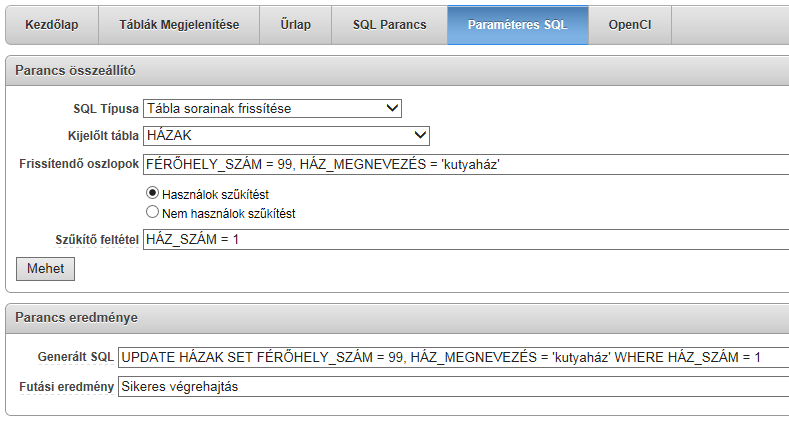
#### Táblalétrehozás meglévő táblából rész futása



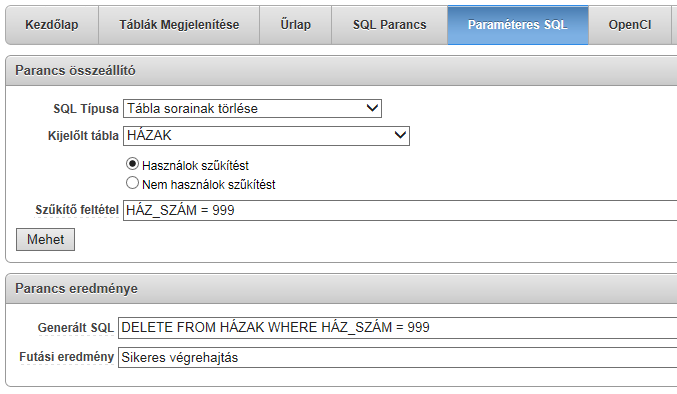
#### Táblalétrehozás oszlopok megadásával rész futása



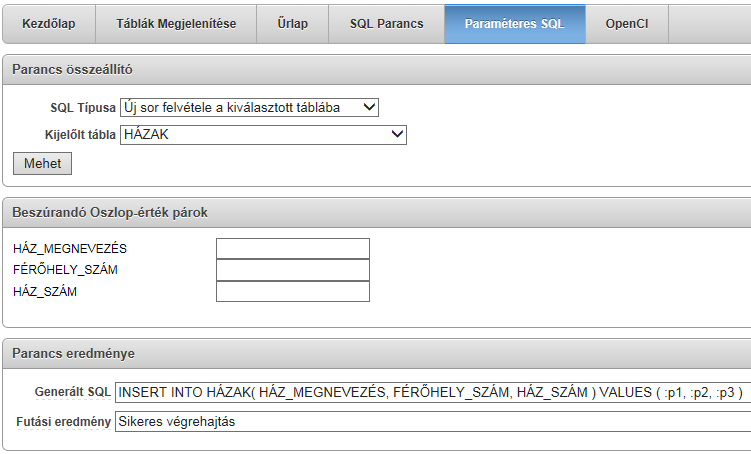
#### Tábla sorainak frissítése rész futása



#### Tábla sorainak törlése



#### Új sor felvétele a kiválasztott táblába



## OpenCL

### A komponens célja

### A megvalósításának fontosabb részletei

### Futási eredménye

# Irodalomjegyzék

A feladat elkészítése során különböző internetes segédanyagokat, és az órai segédleteket, mint az APEX könyv, illetve a példaprogramok, használtam.

# CD Melléklet