Mérési jegyzőkönyv – Szoftver Laboratórium 5

1. mérés: Oracle, mint rendszer

Név:	**Szupera Zita**
Neptun kód:	**AWXUC6**
Feladat kódja:	**23-SZORAK**
Mérésvezető neve:	**Bacsi Tibor**
Mérés időpontja:	**2015-02-26 17:15**
Mérés helyszíne:	**HSZK**
Megoldottfeladatok:	**1,2,3,4,5,6**
Elérhetőpontszám (a megoldottnak jelölt feladatok max pontszámának összege pluszpontok nélkül):	**8,5p**

Mérési feladatok megoldása

1. Jelszóváltás

A jelszóváltoztatáshoz az alábbi parancsot használtam:

```
ALTER USER awxuc6 IDENTIFIED BY uj_jelszo;
```

2. Jogosultságok

2.1 Felhasználóként kapott jogosultságok:

Rendszer szintű jogok: `SELECT * FROM USER_SYS_PRIVS;`

Nem rendelkezem rendszer szintű jogosultságokkal.

Felhasználói szintű jogok: `SELECT * FROM USER_TAB_PRIVS;`

Inherit priviliges joggal rendelkezem

2.2 Szerepeken keresztül kapott jogosultságok:

Szerepeim: `SELECT * FROM USER_ROLE_PRIVS;`

HALLGATO_ROLE szerepem van.

HALLGATO_ROLE-al kapott további szerepek: `SELECT * FROM ROLE_ROLE_PRIVS WHERE ROLE = 'HALLGATO_ROLE';`

- CONNECT
- RESOURCE
- SELECT_CATALOG_ROLE

HALLGATO_ROLE-on keresztül kapott rendszer szintű jogok:

```
SELECT * FROM ROLE_SYS_PRIVS WHERE ROLE = 'HALLGATO_ROLE';
```

	ROLE	PRIVILEGE	ADMIN_OPTION	COMMON
1	HALLGATO_ROLE	ALTER SESSION	NO	NO
2	HALLGATO_ROLE	CREATE SYNONYM	NO	NO
3	HALLGATO_ROLE	CREATE VIEW	NO	NO
4	HALLGATO_ROLE	SELECT ANY DICTIONARY	NO	NO
5	HALLGATO_ROLE	CREATE DATABASE LINK	NO	NO

HALLGATO_ROLE-on keresztül kapott felhasználói privilégiumok:

```
SELECT * FROM ROLE_TAB_PRIVS WHERE ROLE = 'HALLGATO_ROLE';
```

	ROLE	OWNER	TABLE_NAME	COLUMN_NAME	PRIVILEGE	GRANTABLE	COMMON
1	HALLGATO_ROLE	OKTATAS	SZEMELYEK	(null)	SELECT	NO	NO
2	HALLGATO_ROLE	OKTATAS	VISITS	(null)	SELECT	NO	NO
3	HALLGATO_ROLE	OKTATAS	IGAZOLVANYOK	(null)	SELECT	NO	NO
4	HALLGATO_ROLE	OKTATAS	EMBER	(null)	SELECT	NO	NO
5	HALLGATO_ROLE	OKTATAS	SZEMELYEK2	(null)	SELECT	NO	NO
6	HALLGATO_ROLE	OKTATAS	CIM	(null)	SELECT	NO	NO
7	HALLGATO_ROLE	OKTATAS	SZEMELY	(null)	SELECT	NO	NO
8	HALLGATO_ROLE	OKTATAS	MNB_DEVIZA	(null)	SELECT	NO	NO

További szerepeken keresztül kapott rendszer szintű jogok:

```
SELECT * FROM ROLE_SYS_PRIVS
  WHERE ROLE IN ('CONNECT', 'RESOURCE', 'SELECT_CATALOG_ROLE')
  ORDER BY ROLE;
```

	ROLE	PRIVILEGE	ADMIN_OPTION	COMMON
1	CONNECT	CREATE SESSION	NO	NO
2	CONNECT	SET CONTAINER	NO	NO
3	RESOURCE	CREATE CLUSTER	NO	NO
4	RESOURCE	CREATE INDEXTYPE	NO	NO
5	RESOURCE	CREATE TYPE	NO	NO
6	RESOURCE	CREATE OPERATOR	NO	NO
7	RESOURCE	CREATE TABLE	NO	NO
8	RESOURCE	CREATE TRIGGER	NO	NO
9	RESOURCE	CREATE SEQUENCE	NO	NO
10	RESOURCE	CREATE PROCEDURE	NO	NO

További szerepeken keresztül kapott felhasználói jogosultságok:

Felhasználói privilégiumokat csak a SELECT_CATALOG_ROLE-on keresztül kaptam, amely eredményeként adatszótár nézeteket kérdezhetek le.

3. Fizikai tárolási struktúra

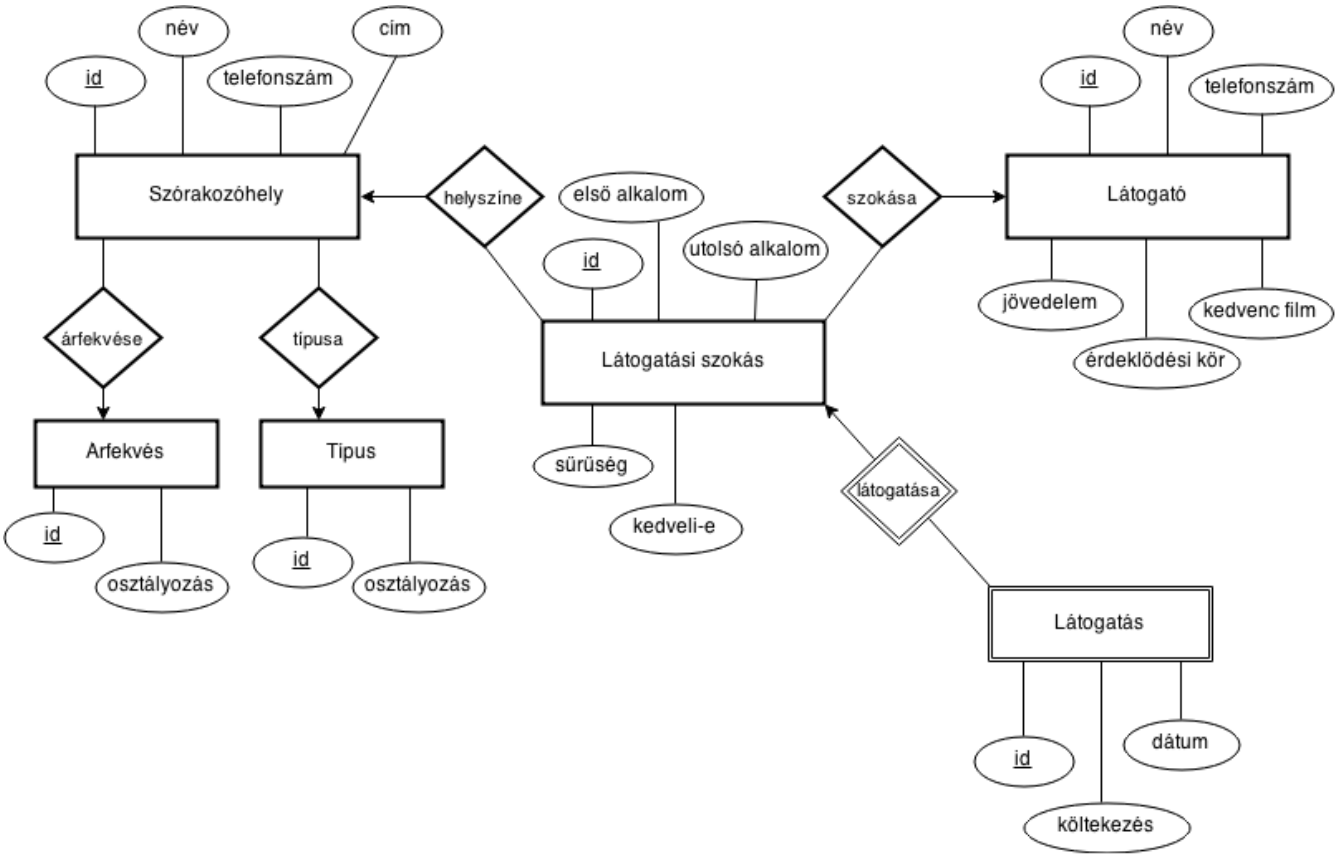
Az adatbázis táblahelyeinek tárolásához használt adatfájlok listázása: `SELECT * FROM DBA_DATA_FILES;`

A lekérdezés eredményeként láthatjuk, hogy mely táblahelyekhez mely adatfájloz tartoznak, továbbá, hogy ezeknek mekkora mérete, bővíthetők-e, és mi a státuszuk.

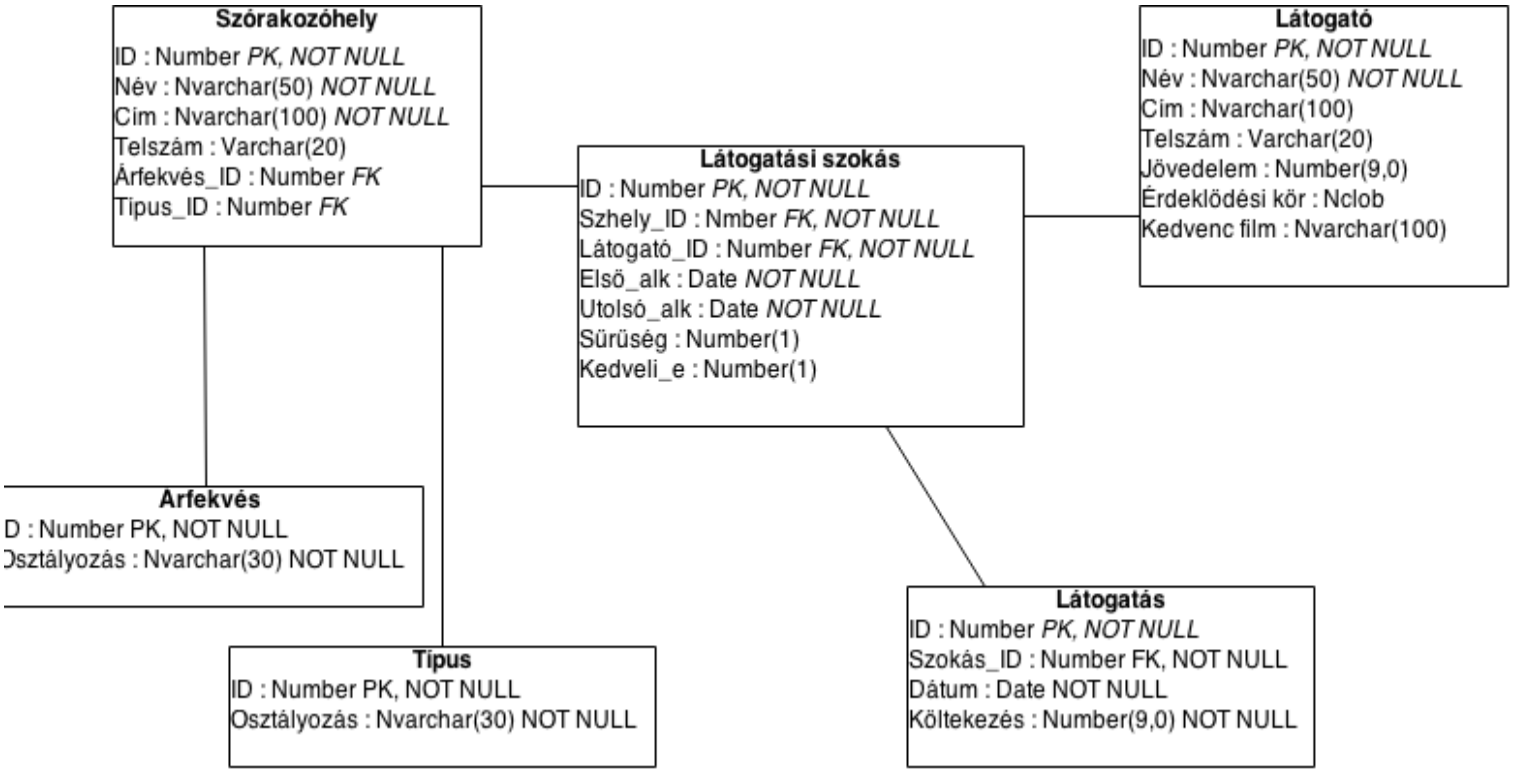
Az OKT táblahelyhez két adatfájl tartozik (okt01.dbf, okt02.dbf), mindkettő állapota AVAILABLE, méretük 64000 és 12800 blokk (blokkméret: 8192 byte = BYTES/BLOCKS). Azt első adatfájl nem bővíthető, kihasználtsága 100%-os, hiszen ha nem lenne az, akkor nem lenne szükség második fájlra. A második fájl kihasználságát úgy kapjuk meg, ha a BLOCKS mező értékét elosztjuk a MAXBLOCK mező értékével: 12800 / 4194302 = 0.3%.

4. Modellezés

E-R diagram:



Adatbázis:



Kényszerek:

Az ábrán is feltűntetett kényszereken kívül az alábbiak szerepelnék még az adatbázisban:

- Szórakozóhely tábla:

    - Unique: A névnek és a címnek közösen egyedinek kell lennie
- Látogató tábla
    - Unique: A névnek és a telefonszámnak együtt egyedinek kell lennie
    - Check: A jövedelemenek 15 000 – 200 000 000 között kell lenie
- Látogatási szokás tábla
    - Check: a sűrűségnek 1 és 7 között kell lennie
    - Check: Kedveli-e: 0 vagy 1 kell legyen (csak PL/SQL-ben van Bool típus)
    - Check: Az első alakalomnak az utolsó alkalomnál korábbi dátumnak kell lennie
- Típus és Árfekvés táblák
    - Unique: az osztályzás mezőnek egyedinek kell lennie

Egyéb megjegyzések:

- A PK mezőket mindenhol tábla szintű szekvenciák segítségével automatikusan generáltam
- A Kedveli-e alapértéke 0
- A Szórakozóhelyben található Típus és Árfekvés idegen kulcs kényszereknél 'No Action' opciót állítottam törlésre, azaz nem lehet egy típus vagy árfekvés mezőt törölni addig, amíg van rájuk hivatkozás.
- A Látogatási szokság táblában a Szórakozóhelyre és a Látogatóra vannak idegen kulcsok, itt a 'Cascade' opciót, tehát, ha egy szórakozóhelyet vagy látogatót törölnek, akkor a hozzátartozó szokások is törlődnek, ugyanez történik a szoksáshoz tartozó látogatásokkal.
- Mivel egy látogató felvételekor csak a név megadása kötelező ezért nem lehet biztosítani, hogy egy személy csak egyszer szerepeljen a Látogató táblában

Tervezői döntések:

- Azonosításra mindenhol id mezőket használtam, mert azt gondolom, hogy ez a legtisztább módja az azonosításnak (illetve kihasználva az Oracle által biztosított szekvenciákat hatékony és egyszerű módja is).
- A látogatási szokások tárolására egy egyedhalmazt hoztam létre, ennek példányaihoz tartoznak látogatások, amelyek önmagukban - a hozzájuk tartozó szokás ismerete nélkül – nem igazán értelmezhetőek, ezért a Látogatás egy gyenge egyedhalmaz.
- A típus adatelem leírására egy külön táblát készítettem, a típust magát az osztályzás mező tartalmazza.
- A szórkozóhelyek osztályozására – típus és árfekvés meghatározása – külön egyedhalmazokat vettem fel. A létrejött adatbázisban idegen kulcsok segítségével már könnyen tudom biztosítani, hogy a Szórakozóhelyhet csak érvényes (vagy legalábbis olyan, ami szerepel az adott táblában) típus ill. Árfekvés tartozzon. Mivel az idegen kulcsok felvehetnek null értéket is, ezért az opcionalitás is biztosított.

DDL kód értelmezése:

A táblákat az SQL Developer tervezője segítségével hoztam létre, az általa generált kódot értelmezem röviden az alábbiakban.

```
CREATE TABLE ARFEKVES
(
  ID NUMBER NOT NULL
, OSZTALYOZAS NVARCHAR2(30) NOT NULL
, CONSTRAINT ARFEKVES_PK PRIMARY KEY
  (
    ID
  )
  ENABLE
);
```

Ez a parancs egy táblát hoz létre, felsorolva az egyes oszlopokat, azok típusát továbbá látható még a NOT NULL kényszer (ha van). Ezen a táblán pedig az ID mező egy PRIMARY_KEY, ami szintén egy kényszerként jelenik meg, az Enable pedig arra utal, hogy a kényszer engedélyezve van.

```
ALTER TABLE SZORAKOZOHELY
ADD CONSTRAINT SZORAKOZOHELY_FK1_ARFEKVES FOREIGN KEY
(
  ARFEKVES_ID
)
REFERENCES ARFEKVES
(
  ID
)
ENABLE
);
```

A további kényszerek felvétele az `ALERT TABLE` utasítás segítségével történik, ebben a példában egy idegen kulcs felvétele látható. A SZORAKOZOHELY_FK1_ARFEKVES nevű kényszere az ARFEKVES ARFEKES_ID mezőjére hivatkozik az ID mezőjén keresztül.

```
CREATE SEQUENCE LATOGATASI_SZOKAS_SEQ;

CREATE TRIGGER LATOGATASI_SZOKAS_TRG
BEFORE INSERT ON LATOGATASI_SZOKAS
FOR EACH ROW
BEGIN
  <<COLUMN_SEQUENCES>>
  BEGIN
    IF INSERTING AND :NEW.ID IS NULL THEN
      SELECT LATOGATASI_SZOKAS_SEQ.NEXTVAL INTO :NEW.ID FROM SYS.DUAL;
    END IF;
  END COLUMN_SEQUENCES;
END;
```

Mivel az elsődleges kulcsokat (ID mező) táblánként egyedi szekvenciák segítségével automatikusan generálom, ezért sok ilyen utasítás is keletkezett az adatbázisom létrehozásakor. Az első sorban a szekvencia létrehozása történik. Utána pedig egy trigger definiálása látható, amely beszúráskor a szekvencia következő elemét adja értékül az ID mezőnek amennyiben annak értéke NULL.

5. Együttműködés sémák között: jogok

Üzlettársaim: C7HEZO, TZLZAK

A jogokat az alábbi paranccsal adtam:

```
GRANT SELECT, UPDATE ON SZORAKOZOHELY TO TZLZAK, C7HEZO;
```

Magyarázat

A jogosultságokat táblánként külön parancsban osztottam ki, mert az Oracle nem enged egy utasításban több táblára is privilégiumokat adni.

SQL script futtatása:

A script azokat a jogosultságokat listázza ki, amelyeket mi adtunk vagy kaptunk. A script elején az oszlopok megjelenésére vonatkozó állításokat találunk, ezután egy `select` utasítás következik, amelyen belül `initcap(grantable) grant_opt` sor elsőre nem feltétlen egyértelmű, itt az történik, hogy a `grantable` nevű mező tartalmát formázzuk (`initcap()`), és megmondjuk, hogy az eredménytáblázatban az oszlop `grant_opt` néven jelenjen meg.

A futtatás eredménye:

	GRANTOR	GRANTEE	TABLE_NAME	PRIVILEGE	GRANT_OPT
1	AWXUC6	C7HEZO	ARFEKVES	SELECT	No
2	AWXUC6	C7HEZO	ARFEKVES	UPDATE	No
3	AWXUC6	C7HEZO	LATOGATAS	SELECT	No
4	AWXUC6	C7HEZO	LATOGATAS	UPDATE	No
5	AWXUC6	C7HEZO	LATOGATASI_SZOKAS	SELECT	No
6	AWXUC6	C7HEZO	LATOGATASI_SZOKAS	UPDATE	No
7	AWXUC6	C7HEZO	LATOGATO	SELECT	No
8	AWXUC6	C7HEZO	LATOGATO	UPDATE	No
9	AWXUC6	C7HEZO	SZORAKOZOHELY	SELECT	No
10	AWXUC6	C7HEZO	SZORAKOZOHELY	UPDATE	No
11	AWXUC6	C7HEZO	TIPUS	SELECT	No
12	AWXUC6	C7HEZO	TIPUS	UPDATE	No
13	AWXUC6	PUBLIC	AWXUC6	INHERIT PRIVILEGES	No
14	AWXUC6	TZLZAK	ARFEKVES	SELECT	No
15	AWXUC6	TZLZAK	ARFEKVES	UPDATE	No
16	AWXUC6	TZLZAK	LATOGATAS	SELECT	No
17	AWXUC6	TZLZAK	LATOGATAS	UPDATE	No
18	AWXUC6	TZLZAK	LATOGATASI_SZOKAS	SELECT	No
19	AWXUC6	TZLZAK	LATOGATASI_SZOKAS	UPDATE	No
20	AWXUC6	TZLZAK	LATOGATO	SELECT	No
21	AWXUC6	TZLZAK	LATOGATO	UPDATE	No
22	AWXUC6	TZLZAK	SZORAKOZOHELY	SELECT	No
23	AWXUC6	TZLZAK	SZORAKOZOHELY	UPDATE	No
24	AWXUC6	TZLZAK	TIPUS	SELECT	No
25	AWXUC6	TZLZAK	TIPUS	UPDATE	No
26	TZLZAK	AWXUC6	HELYFOGLALAS	SELECT	No
27	TZLZAK	AWXUC6	HELYFOGLALAS	UPDATE	No
28	TZLZAK	AWXUC6	ULES	SELECT	No
29	TZLZAK	AWXUC6	ULES	UPDATE	No
30	TZLZAK	AWXUC6	UTAS	SELECT	No
31	TZLZAK	AWXUC6	UTAS	UPDATE	No

A kimeneten jól látszik, hogy C7HEZO-nak és TZLZAK-nak SELECT és UPDATE jogaik vannak a tábláimon, nekem TZLZAK tábláin vannak hasonló jogaim (C7HEZO még valószínűleg nem tart ennél a feladatnál). A 13-ik sort nem értettem elsőre, némi utánanézés után az derült ki, hogy az INHERIT [ANY] PRIVILIGES a 12c verzió újítása, és biztonsági célok miatt került be. (Ha jól értelmezet azért volt szükség erre az újításra, mert PL/SQL kódok futtatásakor a függvény megörökölheti a futattó jogait, ami ahhoz vezethez, hogy a kód írója jogosultságokat adhat magának kihasználva a futtató privilégiumait. Ezt megakadályozandó megvonhatjuk felhasználóktól az INHERIT PRIVILIGES jogot, ami kompatibilitási okok miatt alapesetben engedélyezve van.)

6. Tranzakciókezelés: zárak

A zárak megfigyeléséhez használt utasítások:

```
SELECT locked.OBJECT_ID, objects.OBJECT_NAME, locked.LOCKED_MODE,
locked.SESSION_ID
  FROM V$LOCKED_OBJECT locked, USER_OBJECTS objects
    WHERE locked.ORACLE_USERNAME = 'AWXUC6'
        and locked.OBJECT_ID = objects.OBJECT_ID;
```

`A V$LOCKED_OBJECT` táblából kiolvashatom azon objektumaim azonsítóját, amelyek táblaszinten vannak zárolva (jelen esetben feltehetjük, hogy ezeket mind az általam vizsgált utasítás zárolja), a `USER_OBJECTS` tábla segítségével az azonosítókhoz nevet is rendelhetek.

A SESSION_ID a következő parancshoz szükséges:

```
SELECT TYPE, LMODE FROM V$LOCK WHERE SID = 46;
```

Az első utasításból tehát megtudtam, hogy melyik session zárolja az objektumaimat, ennek a felhasználásával már ki tudom listázni a konkrét zárakat is (ez a második utasítás). Ami számomra most érdekes az a zár típusa (itt már nem csak a táblaszintű zárak vannak feltűntetve) és a módja.

Ezen utasításokkal az alábbi következtetésekre jutottam:

- AE: ez a zár mindig jelen volt, nem DML zár, emiatt nem foglalkoztam vele.
- `INSERT INTO ARFEKVES (OSZTALYOZAS) VALUES (alacsony);` és `INSERT INTO TIPUS (OSZTALYOZAS) VALUES ('bowling');`

  Az `INSERT` utasítás végrehajtásakkor a beszúrandó sor mindig zárolódik (TX/X típusú kizárólagos zárral), továbbá az adott táblára is kerül zár (TM/SX típusú, amely egy megengedőbb fajta táblazár, egyszerre több `INSERT`, `UPDATE`, `SELECT` vagy `DELETE` utasítás is használhatja az így zárolt táblát).

  Ezek a zárak tehát minden beszúráskor megjelennek.

  Ezeken kívül még a SZORAKOZOHELY táblára került egy megosztható táblazár (TM/S, legmegengedőbb táblazár), amelyben található idegen kulcs az ARFEKVES/TIPUS táblára. Az Oracle dokumentációja szerint a zárra a gyermektáblán ilyenkor az esetleges szerkezeti módosítások elkerülése miatt van szükség.

- `INSERT INTO SZORAKOZOHELY (NEV, CIM, TIPUS_ID) VALUES ('Jó Hely', 'Szeged, belváros', 6);`

A beszúrás következtében zárolódik a SZORAKOZOHELY tábla, a beszúrandó soron pedig egy sor szintű zár található.

A LATOGATASI_SZOKAS tábla az előbbi utasításnál látottakhoz hasonlóan a szülő-gyermek viszony miatt zárolódik (itt most a SZORAKOZOHELY a szülő tábla).

Érdekes, hogy bár a fenti utasításban nem szúrok be árfevkésre vonatkozó adatot (idegen kulcsot) mégis - a TIPUS melett - az ARFEKVES tábla is zárolódik. Ennek egyik oka, hogy az idegen kulcsoknak minden esetben létező rekordokra kell mutatniuk (vagy semmire) ezért egy szórakozóhely felvételekor fontos, hogy például ne törölhessék azokat a sorokat, amelyekre az új szórakozóhely hivatkozna. Továbbá valószínűleg a szülő táblán történő szerkezeti módosítások elkeülése is cél a zárolásnál, ami megmagyarázza, hogy miért kerül zár az ARFEKVES-re is. (A táblákra egyébként TM/SX zárak kerülnek, erről a típusról az első utasításnál már volt szó).

- `INSERT INTO LATOGATO (NEV) VALUES ('Ádi');`

  A beszúrás miatt zárolódik a LATOGATO tábla és annak megfelelő sora.

  A LATOGATASI_SZOKAS-ra pedig az előző utasításnál látottak miatt került zár.

- ```
  INSERT INTO LATOGATASI_SZOKAS
  (SZHELY_ID, LATOGATO_ID, ELSO_ALK, UTOLSO_ALK) VALUES (
              6,
              2,
              TO_DATE('2012.  09.  01.', 'YYYY.  MM.  DD.'),
              TO_DATE('2015. 03. 02.', 'YYYY. MM. DD.'));
  ```

  A korábbi utasítások után itt más semmi váratalan nem történik.

  A beszúrás következtében a LATOGATASI_SZOKAS tábla és annak megfelelő sora zárolódik.

  A LATOGATO-ra és a SZORAKOZOHELY-ra idegen kulcsokkal hivatkozunk a táblában, ezért került rájuk zár (lásd előző utasítás TIPUS és ARFEKVES).

  A LATOGATAS táblával pedig ugyanaz történt, mint az első beszúrásnál a SZORAKOZOHELY táblával.

Vélemény(ek) a mérésről

Vélemény, építő jellegű kritika.
  ```