Adatbázis-kezelés

Takács Gábor

mérnökinformatikus, okl. mérnöktanár
Informatika Tanszék, mesteroktató

takacsg@sze.hu

http://takacsg.hu/

Emlékeztető:

Adattípus:

Az SQL 5 alapvető adattípust különböztet meg:

Adattípust	Leírás
Számszerű / numerikus	Olyan adatok tárolására szolgál, amelyekkel numerikus műveleteket végezhetünk.
Szöveges	Tetszőleges szöveges információ tárolására szolgál.
Dátum (idő) jellegű	Dátum és / vagy idő jellegű adatokat tároló adattípus.
Bináris vagy logikai	Kétértékű adatok tárolását végzi.
Szerkezet nélküli	

Adattípus:

BIT	Logikai
INT **SMALLINT** **TINYINT**	Egész
DECIMAL (x,[y])	Fixpontos valós(x az összes jegyek, y a tizedesjegyek száma)
FLOAT **REAL**	Lebegőpontos
DATE	Dátum
TIME	Idő
DATETIME	Dátum és időpont
CHAR (n), nCHAR,	Rögzített hosszúságú karaktersorozat
varCHAR (n), nvarCHAR (n)	Változó hossúságú karaktersorozat (szöveg, kép)

Műveletek az alapvető adattípusok körében

Matematikai műveletek	Előjelváltás (+,-) Számtani alapműveletek (+,-,*,/) Hatványozás (** vagy ^)
Karaktersorozatokkal végezhető műveletek	Összefűzés (+, ||, &)
Bináris és logikai műveletek	Tagadás (NOT), és (AND), vagy (OR)
Dátumok, időpontok és időtartamok körében használható műveletek	Számtani alapműveletek, függvények használatával

Műveletek részletesebben

Numerikus műveletek:

- Négy számtani alapművelet
- Számtani relációk
- Számtani függvények (pl. kerekítések, abszolút érték stb.)
- Ún. halmazfüggvények, amelyek nem egy-egy mezőre, hanem mindig egy adott valódi vagy származtatott oszlop összes mezőjére vonatkoznak egyszerre
- Egyéb matematikai függvények (logaritmus, hatványozás, szögfgv-ek stb.). (Ezek meglétét a szabvány nem írja elő)

Műveletek részletesebben

Karakteres műveletek:

- Minta szerinti keresés nev LIKE ’A%’;
- Összefűzés ’vnev’+’ ’+’knev’
- Csonkítás 1|02|2

Műveletek részletesebben

Műveletek dátumokkal és időpontokkal:

- Átalakítás dátummegadások, dátumformátumok között
- Részletek kiemelése (pl. nap sorszáma dátumból)
- Összehasonlítás
- Intervallumképzés
- Dátumok és intervallumok összeadása, kivonása

Műveletek részletesebben

Bináris és logikai műveletek:

- Negáció, komplemens képzése
- Logikai ÉS (*AND*)
- Logikai VAGY (*OR*)

Valamennyi adattípusnál használható

Összehasonlító műveletek:
Théta operátorok: =, <, >, <=, >=, <>

Összehasonlító műveletek tagadása:
NOT kulcsszó

<,>,= tagadása: ! Jel használható

PL: Nem egyenlő összehasonlító művelet:
!= vagy: <>, vagy: NOT =

Karaktersorozat összehasonlítása balról jobbra a karakterek belső kódja alapján történik.

Kis és nagybetűk belső kódja nem egyező!!
(konvertálás kisbetűsre, vagy nagybetűsre)

Predikátumok- feltételek előírására

- IS
- LIKE
- BETWEEN
- IN
- ANY
- ALL
- SOME
- EXIST
- UNIQUE

- A predikátumok kiválasztási, il. vizsgálati feltételekben használhatók

- Adatdefiníciós utasítások (CHECK), adatkezelő utasítások (WHERE, HAVING) záradékaiban

További CONSTRAINT osztályok…

- NOT NULL: Kötelező értéket megadni az oszlopban.
- CHECK: Kikényszeríti a tartományintegritást az oszlopban megadható értékek korlátozásával. Meghatároz egy logikai kifejezést, mely kiértékelődik, amikor megadunk egy oszlopértéket. Ha a kiértékelés hamis eredményt hoz, akkor nem engedi felvinni a rendszer az új értéket az oszlopba. Egy oszlophoz több CHECK constraint is rendelhető.
- UNIQUE: Biztosítja, hogy ha nem NULL egy adott oszlopérték, akkor egyedi.
- PRIMARY KEY: megadhat egy vagy több oszlopot, melyek értékei egyedien azonosítanak egy táblabeli sort. Nem engedi meg a NULL érték használatát. Táblánként csak egy elsődleges kulcs lehet. (de az akár lehet összetett is)
- FOREIGN KEY: Kapcsolatot biztosít két tábla között. Az egyik tábla idegen kulcsa a másik tábla jelölt kulcsára mutat. Megakadályozza olyan kulcsérték bevitelét, amely nem fordul elő a jelölt kulcs értékei között. ON DELETE záradékkal megadhatjuk, hogy mi történjen az idegenkulcsok soraival, ha a jelölt kulcs sora törlődik. ON UPDATE szintén.

Kifejezések kiértékelési sorrendje PRECEDENCIA szabály

- Zárójelek belülről kifelé
- Függvényhívások
- Matematikai, string és dátum műveletek
  - Előjelváltás
  - Hatványozás
  - Szorzás, osztás (multiplikatív műveletek)
  - Összeadás, kivonás (összevonási műveletek)
- Összehasonlító műveletek
- Logikai műveletek
  - Not,
  - AND,
  - OR

Azonos szintű műveletek: balról jobbra

Későbbiekben használatos jelmagyarázat

[ ] opcionális, nem kötelezően megadandó elem

<> Kötelezően megadandó elem

… Tetszés szerint ismétlődő elem v. utasításrész

| jellel elválasztott elemek közül egy használható

{ } Az elemek közül egyet kötelező megadni.

Egy oszlopra vonatkozó megszorítások

NULL az attribútum definíciójában arra utal, hogy az adat megadása nem kötelező, ez az alapértelmezés ezért a legritkább esetben írják ki.

NOT NULL az attribútum definíciójában arra utal, hogy az adat megadása kötelező, azaz nem vihető be olyan sor a relációban, ahol az így definiált adat nincs kitöltve.

PRIMARY KEY ez az oszlop a tábla elsődleges kulcsa.

UNIQUE ez az oszlop a tábla kulcsa.

CHECK(*feltétel*) csak feltételt kielégítő értékek kerülhetnek be az oszlopba.

[**FOREIGN KEY**] **REFERENCES** *reláció_név* [(*oszlop_név*)], ez az oszlop külső kulcs

Több oszlopra vonatkozó megszorítások

PRIMARY KEY(*oszlop1*[, *oszlop2*, ...]) ezek az oszlopok együtt alkotják az elsődleges kulcsot.

UNIQUE(*oszlop1*[, *oszlop2*, ...]) ezek az oszlopok együtt kulcsot alkotnak.

CHECK(feltétel) csak feltételt kielégítő sorok kerülhetnek be a táblába.

FOREIGN KEY (*oszlop1*[, *oszlop2*, ...]) **REFERENCES** *reláció*(*oszlop1*[, *oszlop2*, ...]), az oszlopok külső kulcsot alkotnak a megadott tábla oszlopaihoz.

Objektumok

Az objektum általánosan elfogadható definíciója: a számítástechnikai objektumok olyan tárolóterületek, amelyeket típusokba sorolunk, szimbolikus nevekkel látunk el és csak adott szabályok betartása mellett kezelhetünk.

- Adatséma
- Adattípus
- Jelkészlet
- Jelsorrend
- Oszloptípus
- Adattábla
- Nézettábla
- Index
- Adatértékszabály
- Hivatkozási függőségi szabály

Objektumok: **adattábla**, index tábla, nézettábla

Adattábla (Table):

Az adatbázis alapvető objektuma. Táblázatos formában tárolja az egyedtípus előfordulásokat. A táblázat oszlopai (mezői) az egyedhez definiált tulajdonságtípusok.

Oszloponként legalább meg kell adni a nevet, adattípust, méretet. Ezen kívül az SQL2 további mezőjellemzőket (indexelt, NULL érték használat, alapértelmezett érték, adatérték-szabály (CONSTRAINT), stb.) is támogat.

Hallgatók tábla

```
neptun      [nchar](6) NOT NULL,
nev         [nvarchar](50) NOT NULL,
telepules   [nvarchar](50) NOT NULL,
utca        [nvarchar](50) NOT NULL,
irsz        [nchar](10) NULL,
tel         [nvarchar](50) NULL,
szdat       [smalldatetime] NOT NULL,
fiu         [bit] NULL,
kforma      [nchar](10) NULL,
kdij        [money] NULL,
egyeb       [nvarchar](max) NULL,
```

Objektumok: adattábla, **index tábla**, nézettábla

Indextábla (INDEX):

Adattáblára alapozva definiálhatunk indexeket. Az indextábla az indexkulcsokat (kimásolva az adattáblából) és a kulcshoz tartozó sor (rekord) fizikai címét tartalmazza rendezett formában, így a kulcsszerinti keresés nagyon gyors.

A rendezettség iránya lehet növekvő, csökkenő. Az indexkulcs lehet összetett, több mező együtt adja az indexkulcsot. Egyedi indexek esetén egy tetszőleges kulcsérték csak egyszer fordul elő. Adattáblánként egy Primary Key (Elsődleges kulcs, egyedi index) definiálható, amely a táblák közötti kapcsolatok létrehozásánál játszik fontos szerepet. A primary key index definiálása az adattábla szerkezetének megadásakor, vagy módosításakor történik. A rendszer automatikusan kezeli az indextáblákat. Ha egy rendezettségre igény jelentkezik, akkor megnézi, hogy van-e olyan indexe. Ha igen, akkor használja, ha nem akkor elvégzi a szükséges rendezést. Az adattáblával együtt az indextáblák automatikusan karbantartódnak.

Objektumok: adattábla, index tábla, **nézettábla**

Nézettábla (VIEW):

Adattáblákra, korábban definiált nézettáblákra alapozva egy lekérdezést (SELECT) definiálhatunk. Csak a definíció kerül tárolásra. Amikor a nézettáblára hivatkozunk, akkor a rendszer a definíció alapján előállítja a lekérdezést. Ezután a lekérdezés eredménytáblában történő módosítás is. Lehetséges a nézettábla alapján az adattáblában történő módosítás is. Az adatséma jellemző objektumai a nézetek.

Adatok létrehozása, és kezelése az adatbázisban = SQL

Az adatok gyors és egyszerű kereshetősége létfontosságú napjainkban.

A relációs adatbázisok kezeléséhez teremtett megfelelő keretrendszert az egyik legelterjedtebb programozási nyelv, a Stuctured Query Language.

A relációs adatbázis-kezelő rendszer

- A relációsadatbázis-kezelő rendszer (angol rövidítéséből: RDBMS) egy olyan adatbázis-kezelő rendszer, amelynek logikai adatbázisát szoftverkomponensei kizárólag a relációs adatmodellek elvén épülnek fel, illetve kérdezhetők le.

- A relációsadatbázis-kezelő rendszerek szabványos adat hozzáférési és adatdefiníciós nyelve az **SQL**.

- FŐGONOSZ*: IBM

SQL

- Az SQL, azaz Structured Query Language (strukturált lekérdezőnyelv) relációsadatbázis-kezelők lekérdezési nyelve.

- Angol nyelvterületen 'eszkjuel' a kiejtése.
  A hagyományokhoz való hűség jegyében sokan 'szíkvel'-nek ejtik, ugyanis korábban Structured English Query Language (SEQUEL) volt az elnevezés, és ezt rövidítették le.

SQL nyelv

Az SQL nyelvi elemeket 4 részre lehet bontani:

- adatdefiníciós (Data Definition Language, DDL),
- adatkezelési (Data Manipulation Language, DML),
- lekérdező (QUERY Language, QL)) és
- adatvezérlő (Data Control Language, DCL)

Utasítások használatának szabályai

- A nyelv szabad írásmódú (utasítások tetszés szerint tördelhetők)
- Záradékokat új sorba kell írni
- Az utasítások végét pontosvesszővel kell jelezni.
- Utasítás kiadható kisbetűs, nagybetűs és kevert formában is.
- Utasításokban elhelyezett listákban az elemeket vesszővel kell elválasztani.
- Utasítások egy részébe más utasítások is beágyazhatók.
- Utasítások alapszavai és egyéb elemei közé elválasztójelként szóközt vagy soremelést kell tenni.
- A zárójelek, vesszők betöltik az elválasztójel szerepét.
- Az elemek elválasztása szempontjából felesleges szóközök, zárójelpárok nem okoznak hibát.

Azonosítók használata

- Adatbázis,
- Adatbáziselemek,
- Alaptáblázatok,
- Nézetek oszlopainak megnevezésére használatosak

Az azonosítók **betűvel kezdődnek**, a második pozíciótól kezdve

- betűket,
- számjegyeket,
- aláhúzás jelet tartalmazhatnak.

Azonosítók egyediségé!!!

Interaktívan kiadott utasítások

- Az utasításokat **interpreter** dolgozza fel.
  - Szintaktikailag ellenőrzi
  - Értelmezi
  - Végrehajtja
  - Eredményeket megjeleníti.
- Ez a feldolgozási mód nem teszi lehetővé több utasítás együttes végrehajtását

Relációk létrehozása

- A relációk létrehozására a CREATE TABLE SQL utasítást használjuk, amelynek alakja:
  CREATE TABLE reláció_név (attribútum_név adattípus [(szélesség)] [NOT NULL].(attribútum_név adattípus [(szélesség)] [NOT NULL]....);

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

Példák

```
CREATE TABLE osztaly (
  oszt_azon      INT2 PRIMARY KEY,
  nev            VARCHAR(14) NOT NULL,
  varos          VARCHAR(13) NOT NULL,
  CONSTRAINT  unev UNIQUE(nev));

CREATE TABLE alkalmazott (
  alk_azon     INT2 PRIMARY KEY,
  nev          VARCHAR(10) NOT NULL,
  beosztas     VARCHAR(9) NOT NULL,
  fonok        INT2 CONSTRAINT fonok_korl REFERENCES alkalmazott (alk_azon),
  belepes      DATE NOT NULL,
  fizetes      FLOAT4 NOT NULL,
  jutalom      FLOAT4,
  oszt_azon    INT2 NOT NULL,
  CONSTRAINT alk_kulso_kulcs FOREIGN KEY (oszt_azon) REFERENCES
                 osztaly (oszt_azon));

CREATE TABLE fiz_oszt (
  f_oszt         INT2 PRIMARY KEY,
  min            FLOAT4 NOT NULL,
  max            FLOAT4 NOT NULL,
  CONSTRAINT  min_max CHECK (min < max));
```

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

Attribútum hozzáadása

A reláció újabb attribútummal való bővítésére az alábbi parancs szolgál:

ALTER TABLE reláció_név ADD attribútum_név adattípus [(szélesség)];

Az új attribútum a reláció utolsó oszlopa lesz. Ebben az esetben a NOT NULL módosító nem adható meg. Az attribútum értéke a már meglevő sorokban NULL (definiálatlan) lesz. Az SQL nyelv megkülönbözteti a nulla numerikus értéket és a NULL, még nem definiált értéket. Például a diákok adatai közé a születési évet felvehetjük a következőparanccsal:

ALTER TABLE Diak ADD szul_ev DATE;

Attribútum szélességének módosítása

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

Egy reláció attribútumának szélességét meg lehet növelni az ALTER TABLE paranccsal.

ALTER TABLE reláció_név MODIFY attribútum_név adattípus (új_szélesség) [NOT NULL];

Ha az attribútum csak NULL értékeket tartalmaz, akkor lehetőség van az adattípus módosítására és a szélesség csökkentésére is. Például a név attribútum szélességének megnövelése a következő paranccsal történhet.

ALTER TABLE Diak MODIFY nev CHAR (40) NOT NULL;

Egy attribútum szélességének csökkentésére illetve törlésére nincs lehetőségünk. Ez csak úgy érhető el, hogy a módosításoknak megfelelő üres relációt hozunk létre, amibe a szükséges adatokat átmásoljuk, majd az eredeti relációt töröljük.

Tábla törlése

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

Teljes relációk törlésére lehetőséget biztosít az SQL nyelv a

DROP TABLE reláció_név; utasítással.

Ezután a relációban tárolt valamennyi adat és a reláció definíciója is törlődik.

A diákok személyi adatait tartalmazó reláció törlése a következő paranccsal lehetséges:

DROP TABLE Diak;

Nézettáblák

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

A nézettáblázat az adatbázisban létező reláción vagy relációkon végrehajtott művelet eredményét tartalmazó olyan új táblázat, amely mögött a valóságban nem áll megfelelő táblázat. Nézettáblát a

CREATE VIEW nézettábla_név AS lekérdezés;

paranccsal hozhatunk létre.

A lekérdező utasítás formáit a lekérdező nyelv tárgyalása során részletezzük:
A 4/b osztály névsorát tartalmazó nézettáblázatot hoz létre a következő parancs.

CREATE VIEW 4b AS SELECT * FROM Diak WHERE osztaly = '4/b';

Akár több táblázatból is vehetünk oszlopokat a nézettáblába. A nézettábla segítségével a három relációra felbontott óra-rend relációt összevonhatjuk egy táblázatba a kényelmesebb kezelés érdekében.

A nézettáblák törlése a DROP VIEW nézettábla_név; paranccsal történik.

Indexek létrehozása

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

A relációkhoz indexeket is rendelhetünk, melyek helyes megválasztása esetén a lekérdezések felgyorsíthatók.
Az indexek létrehozására a következő utasítás szolgál:

CREATE [UNIQUE] INDEX index_név ON reláció (attribútum, attribútum, ...);

Egyedi index létrehozása
Az index létrehozásánál a UNIQUE módosító megadásával a reláció valamennyi sorában az index kulcsnak különbözőnek kell lennie. Általában a reláció kulcsok esetén használható csak (index kulcs = reláció kulcs).

Index törlése
Az indexek törlése a DROP INDEX index_név ON [reláció] paranccsal történik. Az indexeket létrehozásuktól a megszüntetésükig az adatbázis-kezelő automatikusan frissíti, a módosításoknak megfelelően. Ha növeljük az indexek számát, a módosítások végrehajtási ideje növekszik.

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

Új sorok beszúrása az adattáblába

Az SQL adatmanipulációs része biztosítja a relációk feltöltését, az attribútumok módosítását és a sorok törlését. A relációk feltöltésére az INSERT SQL parancs szolgál, melynek általános alakja a következő:

INSERT INTO táblanév [(mező_név_1, mező_név_2, ...)]VALUES (érték, érték, ...);

Egy utasítás segítségével egy sor adható meg az adott relációhoz.
Az attribútum nevek megadása csak akkor kötelező, ha nem minden attribútumhoz rendelünk értéket, vagy az attribútumok értékét nem a definiálás sorrendjében adjuk meg.
A NOT NULL megjelöléssel definiált attribútumok megadása kötelező az INSERT parancsnál, ellenkező esetben hibaüzenetet kapunk.
Az attribútumok és a VALUES után álló értékek sorrendjének és típusának meg kell felelnie egymásnak. A definiálatlan értékekre a NULL érték is beállítható, amely nem azonos a szám (numerikus) típusú nullával.

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

A mezők tartalmának módosítása

A relációkban szereplő mezők tartalmát az UPDATE utasítással módosíthatjuk.

UPDATE reláció_név SET attribútum_név = érték, attribútum_név = érték, ...[WHERE feltétel];

Az UPDATE utasítás segítségével egyidőben a relációk több sorát is módosíthatjuk.
A SET után adhatjuk meg a módosítandó attribútumot és értékeit.
A WHERE után egy feltétel adható meg, az utasítás csak a reláció azon sorain dolgozik, amelyekre a feltétel értéke igaz.
A WHERE rész el is maradhat, ekkor a reláció összes sorára vonatkozik az UPDATE parancs.

tábla létrehozása	CREATE
attribútummal való bővítés, attribútum módosítása	ALTER
tábla törlése	DROP
nézettábla létrehozása	CREATE VIEW
index létrehozása	CREATE INDEX
index törlése	DROP INDEX
feltöltés	INSERT INTO
mező módosítása	UPDATE
sor törlése	DELETE
lekérdezés	SELECT

Sorok törlése

A relációk sorait a DELETE parancs segítségével törölhetjük.

DELETE FROM reláció_név [WHERE feltétel];

A feltételben az UPDATE parancshoz hasonlóan egy zárójelek közé tett lekérdező utasítás is megadható. A WHERE alparancs elmaradása esetén a reláció összes sora törlődik.

DDL (Data Definition Language)

Az SQL nyelvi elemeket 4 részre lehet bontani:

- adatdefiníciós (Data Definition Language, DDL),
- adatkezelési (Data Manipulation Language, DML),
- lekérdező (QUERY Language, QL)) és
- adatvezérlő (Data Control Language, DCL)

Adatbázis és táblái létrehozása.
Relációk, indexek szerkezetének kialakítása, módosítása.

CREATE – létrehozás

ALTER – módosítás

DROP - törlés

DML
(Data Manipulation Language)

Az SQL nyelvi elemeket 4 részre lehet bontani:

- adatdefiníciós (Data Definition Language, DDL),
- adatkezelési (Data Manipulation Language, DML),
- lekérdező (QUERY Language, QL)) és
- adatvezérlő (Data Control Language, DCL)

- Relációk adatainak felvitele, módosítása, vagy törlése

INSERT – adatok beszúrása

UPDATE – adatok módosítása, frissítése

DELETE – adatok törlése relációból

QL (QUERY Language)

Az SQL nyelvi elemeket 4 részre lehet bontani:

- adatdefiníciós (Data Definition Language, DDL),
- adatkezelési (Data Manipulation Language, DML),
- lekérdező (QUERY Language, QL)) és
- adatvezérlő (Data Control Language, DCL)

- Relációk adatainak lekérdezése

SELECT (DISTINCT/TOP) mezőnevek, függvények, összefűzés
FROM tábla nevek
WHERE feltétel(ek)
GROUP BY csoportosítási mezők
HAVING csoportosítási feltételek, utószűrés
ORDER BY rendezési mezők

DCL (Data Control Language)

Felhasználók jogosultságait lehet vele paraméterezni, azaz hogy egy adatbázis esetében milyen műveleteket használhat egy felhasználó

GRANT – jogok adása

REVOKE – jogok visszaállítása

Bevezetés az SQL-be

http://sqlfiddle.com/

Bevezetés az SQL-be

SÉMA (SCRIPT)

```
create table termekek(
id int primary key identity,
megnevezes nvarchar(80),
mennyiseg int,
reszleg varchar (3)
)
insert termekek (megnevezes, mennyiseg, reszleg) values ('Borsodi', 3, 'ALK')
insert termekek (megnevezes, mennyiseg, reszleg) values ('Amstel', 8, 'AUT')
insert termekek (megnevezes, mennyiseg, reszleg) values ('Coronita', 6, 'ALK')
insert termekek (megnevezes, mennyiseg, reszleg) values ('Kőbányai', 6, 'AUT')
insert termekek (megnevezes, mennyiseg, reszleg) values ('Kozel', 24, 'ALK')
insert termekek (megnevezes, mennyiseg, reszleg) values ('Stella Artois', 20, 'AUT')
```

- SELECT
- FROM
- WHERE
- GROUP BY
- HAVING
- ORDER BY

Relációs adatbázis kezelő fő részei

- **Lekérdezés feldolgozó**

A magas szintű programnyelven megfogalmazott (pl.: SQL nyelv) lekérdezéseket, adatbázis-műveleteket egyszerű utasítások sorozatává alakítja.

- **Tárkezelő**

Az adatbázis-kezelők általában közvetlenül felügyelik az adatok lemezen való tárolását.
A tárkezelő két részből áll:

- **Fájlkezelő:** Nyilvántartja a fájlok lemezen való elhelyezkedését és beolvassa egy állomány blokkjait a puffer kezelő kérésére. Az állományok lemezblokkokból épülnek fel, méretük általában 4, 8, vagy 16kb.
- **Puffer kezelő:** A memóriát kezeli. A lemezről beolvasott blokkokat egy memóriaterületen (puffer) tárolja. A különböző műveletek innen kezelik az adatokat. Ha betelt a puffer terület, akkor a „nem használt” blokkokat visszaírja a lemezre, így a felszabaduló memóriaterületre új blokkokat olvas. „Kérésre” a blokkokat szintén lemezre írja.

Relációs adatbázis kezelő fő részei

Tranzakció-kezelő

Tranzakció: Olyan műveletek csoportja, amelyeket egymás után egy egységként kell végrehajtani. Vagy az összes műveletet végrehajtjuk, vagy egyet sem.

A tranzakció-kezelő feladata a tranzakciók helyes lefutásának biztosítása.

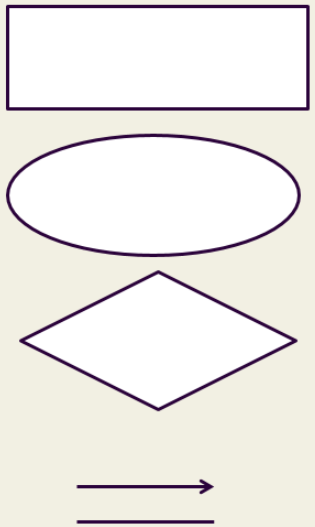
Alapvető elvárások egy tranzakció-kezelővel szemben:

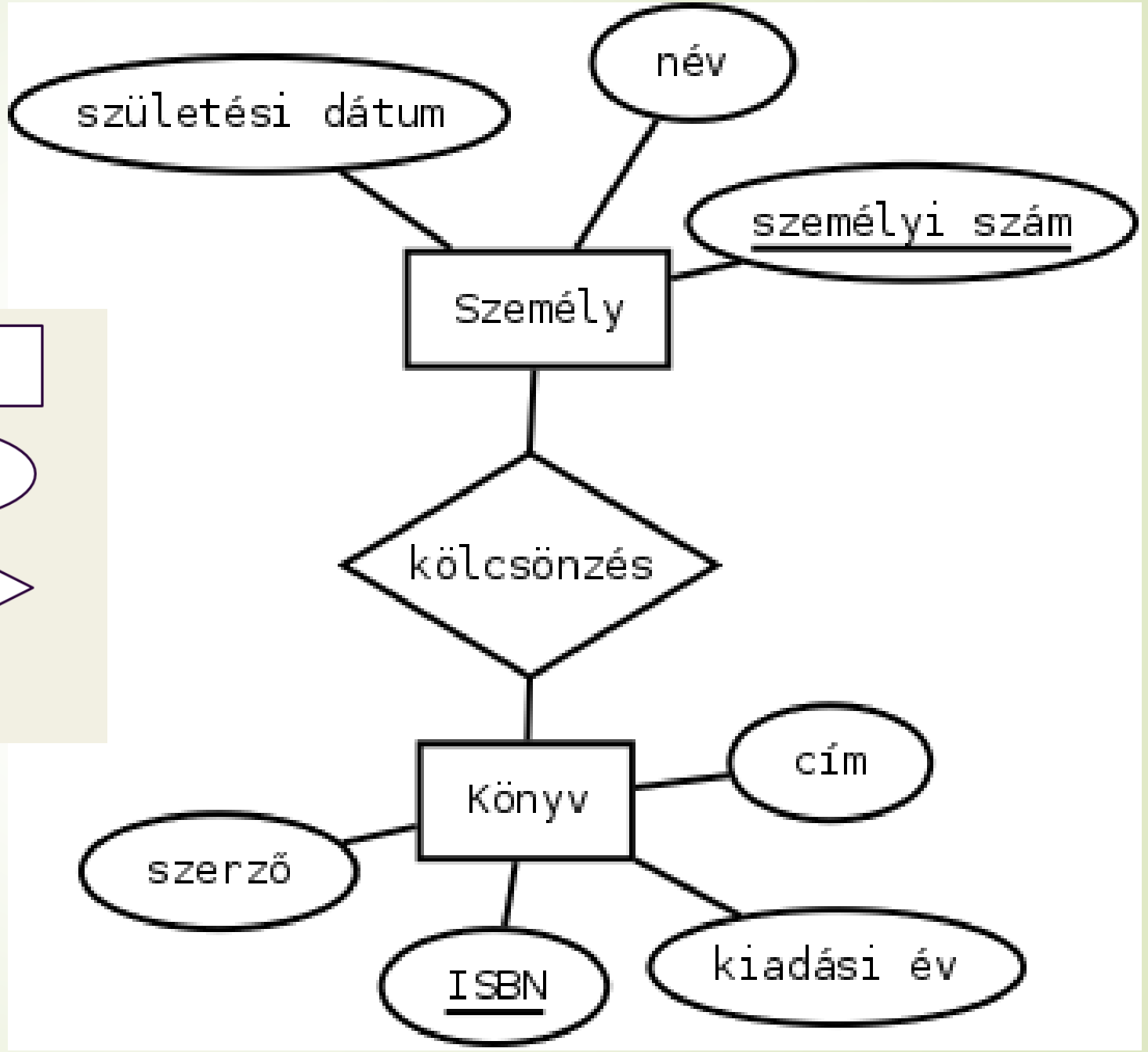
- **Atomosság:** A tranzakció vagy teljes egészében hajtódjon végre, vagy egyáltalán ne. Pl.: pénzautomata...
- **Konzisztencia/Következetesség:** Az adatok megfelelnek bizonyos elvárásoknak. Pl.: Egy repülőgép-helyfoglalási adatbázisban feltétel, hogy egyetlen ülőhelyet se rendeljünk hozzá két különböző utashoz. A feltételt megsérthetjük egy rövid időre a tranzakció alatt (Pl.: Amíg az utasokat áthelyezzük az ülőhelyek között), a tranzakció-kezelő biztosítja, hogy a tranzakciók befejeződése után az adatbázis ismét következetes állapotba kerüljön.
- **Izoláció/Elkülönítés: Az egyidejűleg futó tranzakciók egymásra hatását el kell különíteni egymástól**, mintha egymás után futnának. Ha két ügyintéző ugyanarra a járatra ad el jegyet és a járaton már csak egy hely van, akkor az egyik kérést teljesíteni kell, a másikat elutasítani.
- **Tartósság:** Ha egy tranzakció befejezte a munkáját akkor annak eredménye nem veszhet el rendszerhiba esetén, akkor sem, ha a rendszer közvetlenül a tranzakció befejezése után hibásodik meg.

EK modell

Adatmodell sematikus ábrája

- Egyedhalmaz (táblák)
- Attribútum (sorok)
- Kapcsolat (közös kulcs)
- Összeköttetés





CODD szabályai

- Efgar F. Codd 12 szabály néven elhíresült 13 megállapítása a relációs adatbázis-kezelő rendszerek legfontosabb ismérveit rögzítik. (1985-ből)

- **A 0. szabály**
  Ahhoz, hogy egy rendszer **relációs**nak, **adatbázis**nak, és **kezelő rendszer**nek legyen nevezhető, elfogadható – a rendszernek a relációkkal foglalkozó adottságait kizárólag az adatbázis kezelésére kell használnia.

- **1. Az egységes megjelenésű információ szabálya**
  Ez a szabály egyszerűen csak annyit ír elő, hogy az adatbázisban szereplő összes információt egy, és csak egy megadott formában lehet ábrázolni, nevezetesen a sorokba szedett táblázatokon belül egy-egy oszlop pozícióban. (vízszintes és függőleges koordináta-rendszerben)

CODD szabályai

- **2. Garantált lokalizálhatóság szabálya**

  Elsődleges kulcs alapkövetelménye: Azt mondja ki, hogy az adatbázisban minden egyes skaláris értékre logikailag úgy kell hivatkozni, hogy megadjuk az azt tartalmazó táblázat és az oszlop nevét, valamint a megfelelő sor (azt tartalmazó sor) elsődleges kulcsának az értékét.

- **3. A NULL értékek egységes kezelése**

  Az adatbázis-kezelő rendszernek olyan módszerrel kell támogatnia a „hiányzó és nem felhasználható információt" amely egységes, és eltér az összes „rendes" érték kezelésétől (például numerikus értékek esetében, „nullától vagy más számtól különböző"-ként), továbbá független az adattípustól. Ebbe az is beletartozik, hogy ezeknek a reprezentációknak a kezelését a szoftvernek módszeresen kell végeznie.

CODD szabályai

- **4. A relációs modell alapján aktív online katalógust kell üzemben tartani**
  A rendszernek támogatnia kell egy online, beépített katalógust, amely a szokásos lekérdező nyelvet használó feljogosított felhasználók előtt nyitva áll.

- **5. A teljes körű „adatnyelv" szabálya**
  A rendszernek legalább egy olyan relációs nyelvet kell támogatnia, amelynek
  - (a) lineáris a szintaxisa,
  - (b) interaktívan és alkalmazási programokon belül is lehet használni, továbbá
  - (c) támogatja az adat definiáló műveleteket (beleértve az adatok megjelenítési képeinek meghatározására szolgálókat), az adatmódosító (manipulációs) műveleteket (frissítés és visszakeresés is), biztonsági és jósági (integritási) korlátokat, valamint a tranzakció kezelési műveleteket (begin, commit, és rollback: elkezdés, jóváhagyás és visszagörgetés).

CODD szabályai

- **6. A nézetek frissítésének szabálya**

  A rendszernek képesnek kell lennie az adatok elméletileg frissíthető minden nézetének frissítésére.

- **7. Magas szintű beszúrás, frissítés és törlés**

  A rendszernek támogatnia kell az INSERT, UPDATE, és DELETE (új adat, módosítás, törlés) operátorok halmaz szintű, egyidejű működését.

- **8. Fizikai szintű adatfüggetlenség**

  A fizikai adatfüggetlenség akkor áll fenn, ha az alkalmazások (programok) és a felhasználók adatelérési módja független az adatok tényleges (fizikai) tárolási és elérési módjától.

CODD szabályai

- **9. Logikai szintű adatfüggetlenség**

  Logikai adatfüggetlenség akkor áll fenn, ha az adatbázis logikai szerkezetének megváltoztatása nem igényli az adatbázist használó alkalmazások (programok) megváltoztatását.

- **10. Jóság (integritás) függetlenség**

  Az adatok jóságának (érvényességének) korlátait az adatfeldolgozási programoktól függetlenül kell tudni meghatározni, és azokat katalógusban kell nyilvántartani. Legyen lehetséges a szóban forgó korlátokat – úgy és amikor szükséges – megváltoztatni, anélkül hogy a meglévő alkalmazásokat szükségtelen módon zavarnánk.

CODD szabályai

- **11. Elosztástól való függetlenség**

  A meglévő alkalmazások működése zavartalan kell, hogy maradjon

  (a) amikor sor kerül az adatbázis kezelő szoftver elosztott változtatásnak bevezetésére a rendszerben

  (b) amikor a meglévő elosztott adatokat a rendszer újra szétosztja

- **12. Megkerülhetetlenség szabálya**

  Ha a rendszernek van egy alacsony szintű (egyszerre egy rekordot érintő) interfésze, akkor azt az interfészt ne lehessen a rendszer megkerülésére használni, például a relációs biztonsági vagy jósági (integritás védelmi) korlátok megsértésével.

Normálformák: 1NF

(első normálforma)

Cél: **Teljes funkcionális függőségek legyenek. Kialakítása**

- Egyedeinket -> egyedtípusokba rendezzük
- Tulajdonságaikat → tulajdonságtípusokba rendezzük.

Feltételek:

- Minden rekord különbözik
- Rekordonként megegyezik a mezők száma, és sorrendje
- Nincsenek többértékű mezők.

Személyek							
SzemIgSzam	Nev	IRSZ	Varos	Utca	Szam	Vezetekes	Mikor
101558 AI	Virág Borbála	3100	Salgótarján	Füsti u 4.	+36/20/324 7889	Nem	08:00-16:00
234511 ZK	Tóth Áron	3300	Eger	Gornai u. 1.	+36/30/678 6634	Nem	00:00-24:00
113297 UL	Tóth Áron	1610	Budapest	Kerek u. 11	+36/1/320 5122	Igen	18:00-22:00
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3.	+36/36/321 321 +36/70/788 9130 +36/20/315 5551	Igen Nem Nem	08:00-13:00 09:30-18:00 08:00-20:00
656456 ZY	Bende Aladár	3000	Hatvan	Cukor út 12.	+36/36/321 321 +36/30/368 5552	Igen Nem	08:00-13:00 08:00-20:00

Normálformák: 1NF
(első normálforma)

A jó megoldást a következő változat nyújtja. Ebben a táblában a több telefonnal rendelkező személyeket többször is eltároltuk, annak megfelelően, hogy hány telefonnal rendelkeznek. Így elértük, hogy ne legyenek többértékű mezők.

Személyek							
SzemIgSzam	Nev	IRSZ	Varos	Utca	Szam	Vezetekes	Mikor
101558 AI	Virág Borbála	3100	Salgótarján	Füsti u 4.	+36/20/324 7889	Nem	08:00-16:00
234511 ZK	Tóth Áron	3300	Eger	Gornai u. 1.	+36/30/678 6634	Nem	00:00-24:00
113297 UL	Tóth Áron	1610	Budapest	Kerek u. 11	+36/1/320 5122	Igen	18:00-22:00
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3.	+36/36/321 321	Igen	08:00-13:00
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3.	+36/70/788 9130	Nem	09:30-18:00
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3.	+36/20/315 5551	Nem	08:00-20:00
656456 ZY	Bende Aladár	3001	Hatvan	Cukor út 12.	+36/36/321 321	Igen	08:00-13:00
656456 ZY	Bende Aladár	3001	Hatvan	Cukor út 12.	+36/30/368 5552	Nem	08:00-20:00
060811 DI	Örök Virág	3100	Salgótarján	Főút 32.	+36/32/932928	Igen	19:00-20:30
060811 TF	Medve Bálint	3300	Eger	Kesre u 41.	+36/36/555 325	Igen	18:00-20:00

Normálformák: 2NF
(második normálforma)

A 2NF előfeltétele, hogy

- adatbázisunk minden táblája legalább 1NF-ben legyen!
- a reláció minden nem elsődleges attribútuma teljes funkcionális függőségben van az összes reláció kulccsal.

Tehát a táblákban **megszűntetjük az esetleges részleges funkcionális függéseket.**

Személyek

SzemIgSzam	Nev	IRSZ	Varos	Utca
101558 AI	Virág Borbála	3100	Salgótarján	Füsti u 4.
234511 ZK	Tóth Áron	3300	Eger	Gornai u. 1.
113297 UL	Tóth Áron	1610	Budapest	Kerek u. 11
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3.
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3
656456 ZY	Bende Aladár	3000	Hatvan	Cukor út 12.
656456 ZY	Bende Aladár	3001	Hatvan	Cukor út 12
060811 DI	Örök Virág	3100	Salgótarján	Főút 32.
060811 TF	Medve Bálint	3300	Eger	Kesre u 41.

Telefonok

Szam	Vezetekes	Mikor
+36/20/324 7889	Nem	08:00-16:00
+36/30/678 6634	Nem	00:00-24:00
+36/1/320 5122	Igen	18:00-22:00
+36/36/321 321	Igen	08:00-13:00
+36/70/788 9130	Nem	09:30-18:00
+36/20/315 5551	Nem	08:00-20:00
+36/36/321 321	Igen	08:00-13:00
+36/30/368 5552	Nem	08:00-20:00
+36/32/932928	Igen	19:00-20:30
+36/36/555 325	Igen	18:00-20:00

Normálformák: 2NF
(második normálforma)

Személyek

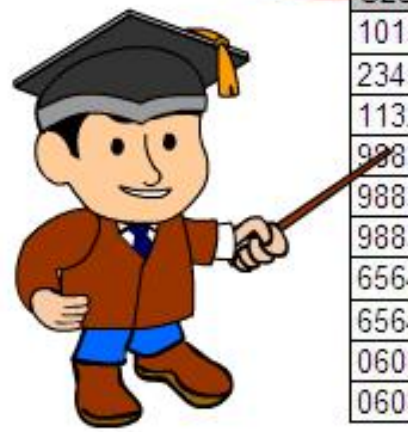
SzemIgSzam	Nev	IRSZ	Varos	Utca
101558 AI	Virág Borbála	3100	Salgótarján	Füsti u 4.
234511 ZK	Tóth Áron	3300	Eger	Gornai u. 1.
113297 UL	Tóth Áron	1610	Budapest	Kerek u. 11
988326 AX	Árva Viola	3024	Selyp	Nagy tér 3.
656456 ZY	Bende Aladár	3001	Hatvan	Cukor út 12.
060811 DI	Örök Virág	3100	Salgótarján	Főút 32.
060811 TF	Medve Bálint	3300	Eger	Kesre u 41.

Telefonok

Szam	Vezetekes	Mikor
+36/20/324 7889	Nem	08:00-16:00
+36/30/678 6634	Nem	00:00-24:00
+36/1/320 5122	Igen	18:00-22:00
+36/70/788 9130	Nem	09:30-18:00
+36/20/315 5551	Nem	08:00-20:00
+36/36/321 321	Igen	08:00-13:00
+36/30/368 5552	Nem	08:00-20:00
+36/32/932928	Igen	19:00-20:30
+36/36/555 325	Igen	18:00-20:00

SzemelyekTelefonok

SzemIgSzam	Szam
101558 AI	+36/20/324 7889
234511 ZK	+36/30/678 6634
113297 UL	+36/1/320 5122
988326 AX	+36/36/321 321
988326 AX	+36/70/788 9130
988326 AX	+36/20/315 5551
656456 ZY	+36/36/321 321
656456 ZY	+36/30/368 5552
060811 DI	+36/32/932928
060811 TF	+36/36/555 325



Normálformák: 3NF
(harmadik normálforma)

A 3NF előfeltétele, hogy :

- adatbázisunk minden táblája legalább 2NF-ben legyen!
- A reláció nem tartalmaz funkcionális függőséget a nem elsődleges attribútumok között.

Tehát : a táblákban megszűntetjük a **tranzitív függéseket.**

Személyek

SzemIgSzam	Nev	Utca	IRSZ
101558 AI	Virág Borbála	Füsti u 4.	3100
234511 ZK	Tóth Áron	Gornai u. 1.	3300
113297 UL	Tóth Áron	Kerek u. 11	1610
988326 AX	Árva Viola	Nagy tér 3.	3024
656456 ZY	Bende Aladár	Cukor út 12.	3001
060811 DI	Örök Virág	Főút 32.	3100
060811 TF	Medve Bálint	Kesre u 41.	3300

Varosok

IRSZ	Varos
3100	Salgótarján
3300	Eger
1610	Budapest
3024	Selyp
3001	Hatvan

SzemelyekTelefonok

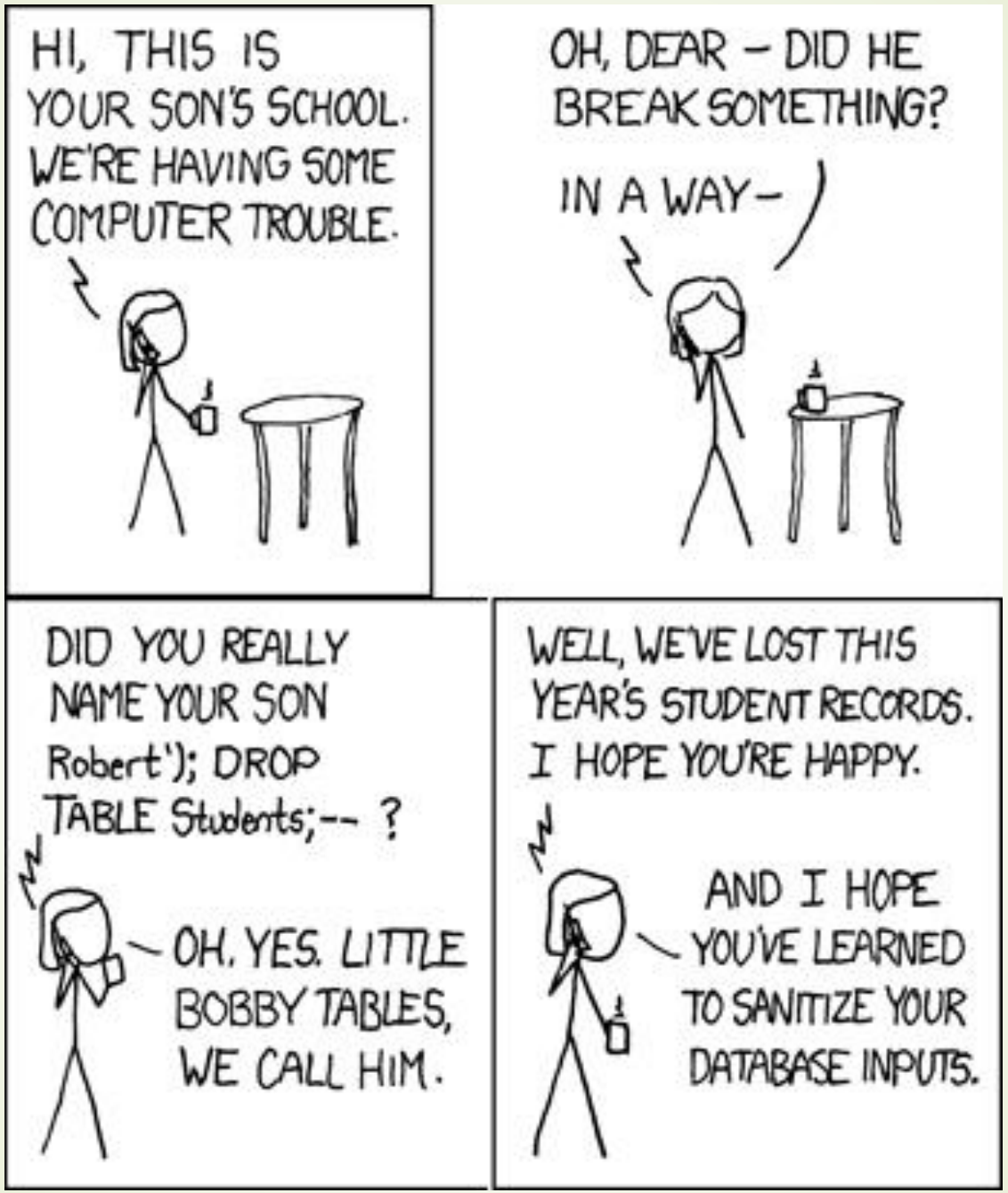
SzemIgSzam	Szam
101558 AI	+36/20/324 7889
234511 ZK	+36/30/678 6634
113297 UL	+36/1/320 5122
988326 AX	+36/36/321 321
988326 AX	+36/70/788 9130
988326 AX	+36/20/315 5551
656456 ZY	+36/36/321 321
656456 ZY	+36/30/368 5552
060811 DI	+36/32/932928
060811 TF	+36/36/555 325

Telefonok

Szam	Vezetekes	Mikor
+36/20/324 7889	Nem	08:00-16:00
+36/30/678 6634	Nem	00:00-24:00
+36/1/320 5122	Igen	18:00-22:00
+36/70/788 9130	Nem	09:30-18:00
+36/20/315 5551	Nem	08:00-20:00
+36/36/321 321	Igen	08:00-13:00
+36/30/368 5552	Nem	08:00-20:00
+36/32/932928	Igen	19:00-20:30
+36/36/555 325	Igen	18:00-20:00

Google

sql injection licence plate


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR - DID HE BREAK SOMETHING?
IN A WAY -
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

Köszönöm a figyelmet!