Introduction to Database Systems

# Database for a School Management System

Made by: Gábor Csaba Attila ( DU4POE )

Project supervisor: Halász András

# Table of Contents

## Introduction

I would like to design a database for education system. Nowadays the educational institutions have to store a lot of data. So, databases will be inescapable for all school in the near future. I would like to set an example how the schools could build their own databases. With this database, the employees could find every data easily (for example: personal data). For the teachers, it can be useful too, they could store information about the courses, or homeworks in a retrievable way.

## Description

I'd like to store the following data:

School
School stores the essential data of an institution. It could be more premises in one school. The basic data are the following: name of the school, education-ID, contact data like telephone number, email, address, webpage.

Person
Person stores the basic data, which have all the people. It is a superclass. It's in IS-A relation with Teachers, Students and Parents.

Teachers
It is a subclass of Person, and has the basic data from there. So teachers have these data: name, place and date of birth, Social Security Number, gender, address, email, telephone number, age, date of start to work, work-length, education-ID.

Students
It is a subclass of Person. I will store the following data about a student: name, place and date of birth, Social Security Number, gender, address, email, telephone number, age, education-ID, date of start to learn, study-length.

Parents
It is also a subclass of Person. It doesn't have any specific data: name, place and date of birth, Social Security Number, gender, address, email, telephone number, age, ID.

Classes

The students are divided into classes. A class have some basic data: date of start, class ID.

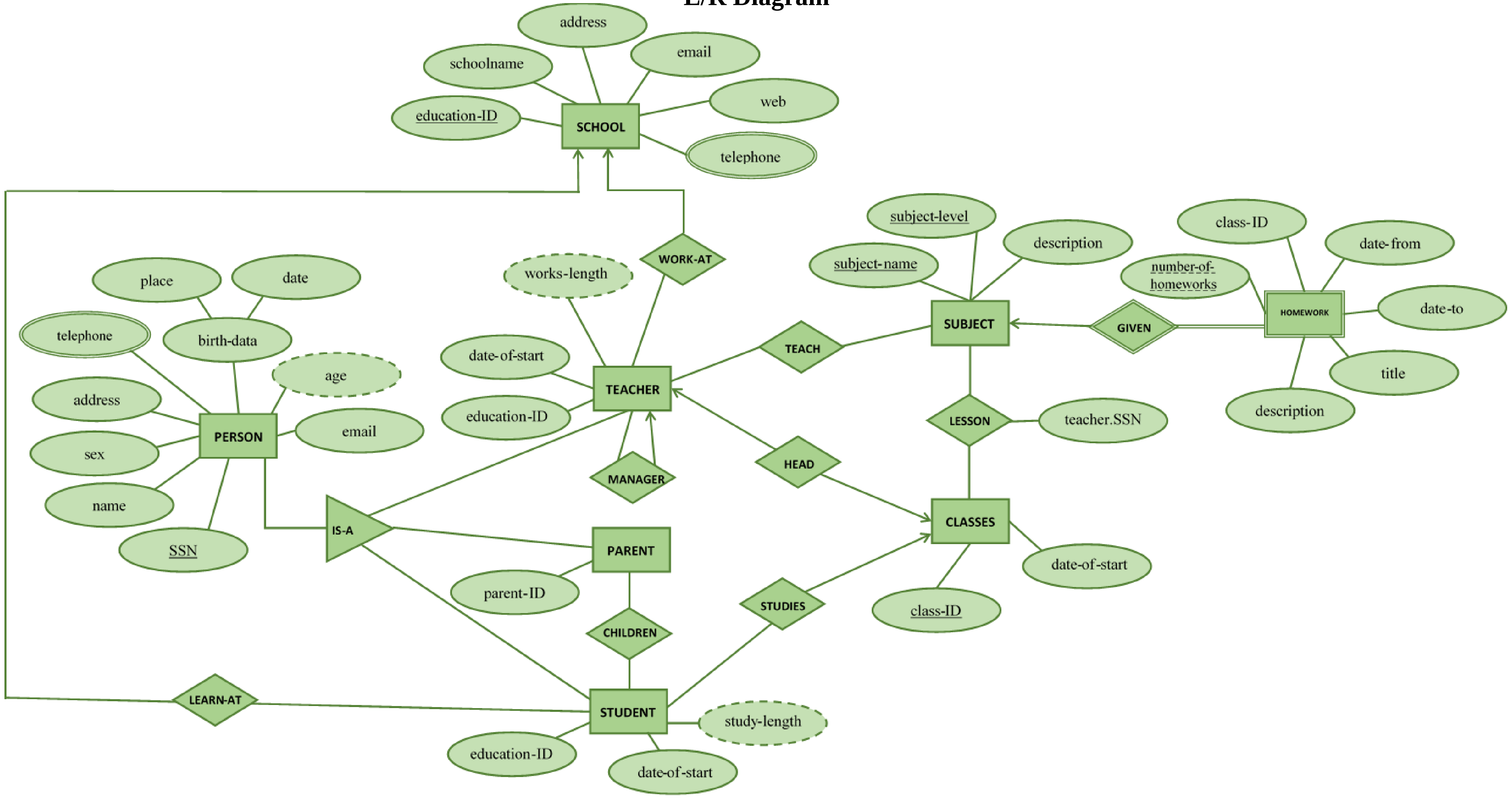
Subjects

A subject could be taught in more level, so we need to know the level of the subject to distinguish the subjects. These data will be stored: name, level, course information.

Homeworks

For each subject, could be given some homework. It has no identifying data, so it will be a weak entity of Subjects. The following data will be stored: class ID, number, date of publish, date of term, description.

## E/R Diagram



*1. Figure: ER Diagram*

Relationships and ISA-s

- teacher, student and parent *ISA* with person
- teacher has relationship with class – *one-to-one*
- teacher, student have relationships with school – *many-to-one*
- student has relationship with class – *many-to-one*
- parent has a relationship with student – *many-to-many*
- teacher has a relationship with subject – *many-to-many*
- subject has a relationship with class – *many-to-many*

## Relational model

SCHOOL (education_id, schoolname, address, email, web)

SCHOOL_TELEPHONE (school_id, telephone)

TEACHER (ssn, teacher_name, sex, address, birth_place, birth_date, email, education_id, date_of_start, manager_ssn, school_id)

PARENT (ssn, parent_name, sex, address, birth_place, birth_date, email, parent_id)

STUDENT (ssn, student_name, sex, address, birth_place, birth_date, email, education_id, date_of_start, scool_id, class_id)

TELEPHONE (telephone_number, user_ssn)

TEACH (teacher_id, subject_name, subject_level)

CLASSES (class_id, date_of_start, head_teacher)

SUBJECT (subject_name, subject_level, description)

HOMEWORK (subject_name, subject_level, number_of_homeworks, class_id, date_from, date_to, title, description)

CHILDREN (parent_id, child_id)

LESSON (class_id, subject_name, subject_level, teacher_id)

## SQL statements

### DROP

```
DROP TABLE homework             CASCADE CONSTRAINTS;
DROP TABLE lesson               CASCADE CONSTRAINTS;
DROP TABLE teach                CASCADE CONSTRAINTS;
DROP TABLE children             CASCADE CONSTRAINTS;
DROP TABLE student              CASCADE CONSTRAINTS;
DROP TABLE classes              CASCADE CONSTRAINTS;
DROP TABLE teacher              CASCADE CONSTRAINTS;
DROP TABLE school_telephone     CASCADE CONSTRAINTS;
DROP TABLE school               CASCADE CONSTRAINTS;
DROP TABLE parents              CASCADE CONSTRAINTS;
DROP TABLE telephone            CASCADE CONSTRAINTS;
DROP TABLE subject              CASCADE CONSTRAINTS;
```

*2. Figure: Drop statements*

With the DROP SQL statement, I can move a table or object table to the recycle bin or remove the table and all its data from the database entirely.

With CASCADE CONSTRAINTS, I can drop all referential integrity constraints that refer to primary and unique keys in the dropped table, otherwise the database returns an error and does not drop the table.[1]

[1] https://docs.oracle.com/cd/B19306_01/server.102/b14200/statements_9003.htm (visited: 31. October 2016)

CREATE

With CREATE TABLE statement, I can create relational tables, which are the basic structures to hold user data.[2]

DEFAULT

While creating the tables I used the DEFAULT expression to specify a value to be assigned to the column if an INSERT statement omits a value for the column. The data type of the expression must match the data type specified for the column. The column must also be large enough to hold this expression.[3]

DATA TYPES

I used the following data types:

- VARCHAR2(n) – stores variable-length character strings, n specify the maximum length in bytes (characters).
- SMALLINT – stores numbers in 2 bytes.
- DATE – stores year, month and day in this standard form DD-MMM-YY

NOT NULL

If it's important to store a column, I used the NOT NULL expression. It is default by the primary keys. For example, in the teacher table I store the education-ID, it isn't primary key, but each and every teacher must have an education-ID.

FOREIGN KEY

A foreign key in one table points to a primary key in another table.

PRIMARY KEY

The primary key constraint uniquely identifies each record in a database table. Primary keys must contain unique values. A primary key column cannot contain NULL values.

[2] https://docs.oracle.com/cd/B19306_01/server.102/b14200/statements_7002.htm (visited: 31. October 2016)
[3] https://docs.oracle.com/database/121/SQLRF/statements_7002.htm (visited: 31. October 2016)

The CREATE TABLE statements to this database are listed below.

```
CREATE TABLE school
 (
   education_id          VARCHAR2(8)          NOT NULL,
   schoolname            VARCHAR2(100)        NOT NULL,
   address               VARCHAR2(100)        NOT NULL,
   email                 VARCHAR2(50),
   web                   VARCHAR2(50),
   PRIMARY KEY(education_id)
 );
CREATE TABLE school_telephone
 (
   telephone             VARCHAR2(12)         NOT NULL,
   education_id          VARCHAR2(8)          NOT NULL,
   FOREIGN KEY(education_id) REFERENCES school(education_id),
   PRIMARY KEY(telephone, education_id)
 );
CREATE TABLE teacher
 (
   ssn                   VARCHAR2(9)          NOT NULL,
   teacher_name          VARCHAR2(50)         NOT NULL,
   sex                   VARCHAR2(6),
   address               VARCHAR2(100),
   birth_place           VARCHAR2(50),
   birth_date            DATE,
   email                 VARCHAR2(50),
   education_id          VARCHAR2(8)          NOT NULL,
   date_of_start         DATE                 DEFAULT CURRENT_DATE,
   manager_ssn           VARCHAR2(8),
   school_id             VARCHAR2(8)          NOT NULL,
   FOREIGN KEY(school_id) REFERENCES school(education_id),
   PRIMARY KEY(ssn)
 );
CREATE TABLE classes
 (
   class_id              VARCHAR2(3),
   date_of_start         DATE,
   head_teacher          VARCHAR2(9),
   FOREIGN KEY(head_teacher) REFERENCES teacher(ssn),
   PRIMARY KEY(class_id)
 );
CREATE TABLE parents
 (
   ssn                   VARCHAR2(9)          NOT NULL,
   parent_name           VARCHAR2(50)         NOT NULL,
   sex                   VARCHAR2(6),
   address               VARCHAR2(100),
   birth_place           VARCHAR2(50),
   birth_date            DATE,
   email                 VARCHAR2(50),
   parent_id             VARCHAR2(8),
   PRIMARY KEY(ssn)
 );
```

```
CREATE TABLE student
 (
   ssn                     VARCHAR2(9)              NOT NULL,
   student_name            VARCHAR2(50)             NOT NULL,
   sex                     VARCHAR2(6),
   address                 VARCHAR2(100),
   birth_place             VARCHAR2(50),
   birth_date              DATE,
   email                   VARCHAR2(50),
   education_id            VARCHAR2(8),
   date_of_start           DATE                     DEFAULT CURRENT_DATE,
   school_id               VARCHAR2(8),
   class_id                VARCHAR2(3),
   FOREIGN KEY(school_id) REFERENCES school(education_id),
   FOREIGN KEY(class_id) REFERENCES classes(class_id),
   PRIMARY KEY(ssn)
 );

CREATE TABLE children
 (
   parent_id               VARCHAR2(9)              NOT NULL,
   child_id                VARCHAR2(9)              NOT NULL,
   FOREIGN KEY(parent_id) REFERENCES parents(ssn),
   FOREIGN KEY(child_id) REFERENCES student(ssn),
   PRIMARY KEY(parent_id, child_id)
 );

CREATE TABLE telephone
 (
   telephone_number        VARCHAR2(12)             NOT NULL,
   user_ssn                VARCHAR2(9)              NOT NULL,
   PRIMARY KEY(telephone_number, user_ssn)
 );

CREATE TABLE subject
 (
   subject_name            VARCHAR2(50)             NOT NULL,
   subject_level           SMALLINT                 NOT NULL,
   description             VARCHAR2(1000),
   PRIMARY KEY(subject_name, subject_level)
 );

CREATE TABLE teach
 (
   teacher_id              VARCHAR2(9)              NOT NULL,
   subject_name            VARCHAR2(50)             NOT NULL,
   subject_level           SMALLINT                 NOT NULL,
   FOREIGN KEY(teacher_id) REFERENCES teacher(ssn),
   PRIMARY KEY(teacher_id, subject_name, subject_level)
 );
```

```
CREATE TABLE lesson
 (
   class_id              VARCHAR2(3)         NOT NULL,
   subject_name          VARCHAR2(50)        NOT NULL,
   subject_level         SMALLINT            NOT NULL,
   teacher_id            VARCHAR2(9)         NOT NULL,
   FOREIGN KEY(class_id) REFERENCES classes(class_id),
   PRIMARY KEY(class_id, subject_name, subject_level)
 );

CREATE TABLE homework
 (
   subject_name             VARCHAR2(50)  NOT NULL,
   subject_level            SMALLINT      NOT NULL,
   number_of_homeworks      SMALLINT      NOT NULL,
   class_id                 VARCHAR2(3),
   date_from                DATE          DEFAULT CURRENT_DATE,
   date_to                  DATE          DEFAULT CURRENT_DATE + 1,
   title                    VARCHAR2(50)  NOT NULL,
   description              VARCHAR2(1000),
   FOREIGN KEY(class_id) REFERENCES classes(class_id),
   PRIMARY KEY(subject_name, subject_level, number_of_homeworks)
 );
```

*3. Figure: Create table statements*

CONSTRAINT
While creating the tables I used SQL constraints to specify rules for the data in tables.

The constraints, what I used:

- NUT NULL
- DEFAULT
- FOREIGN KEY
- *PRIMARY KEY – what is NOT NULL and UNIQUE*

INSERT
The list of insert statements is at the attachments, on the 32. page.

VIEWS
A view is a virtual table based on the result-set of an SQL statement. I created a view for a class to demonstrate, how we can use it.

```
CREATE VIEW email_of_class AS
SELECT student_name, address, email
FROM student
WHERE student.class_id = '8/B';
```

*4. Figure: Create view statement*

ALTER TABLE
This statement is used to add, delete, or modify columns in an existing table.

```
ALTER TABLE classes
ADD head_teacher_name VARCHAR2(50) DEFAULT 'Karlne Purczeld Erika';
ALTER TABLE classes
MODIFY date_of_start DATE;
```

*5. Figure: Alter table statements*

UPDATE
With the update statement, we can update existing records in a table.

```
UPDATE classes
   SET head_teacher_name = 'Gaborne Szilagyi Erzsebet'
   WHERE class_id IN (SELECT class_id
                      FROM classes, teacher
                      WHERE classes.head_teacher = teacher.ssn
                            AND teacher.teacher_name = ‘Gaborne Szilagyi Erzsebet’ );
UPDATE classes
   SET head_teacher_name = 'Eordogh Krisztina’
   WHERE class_id IN (SELECT class_id
                      FROM classes, teacher
                      WHERE classes.head_teacher = teacher.ssn
                            AND teacher.teacher_name = ‘Eordogh Krisztina’ );
```

*6. Figure: Update statements*

## RA and SQL queries

### Simple queries

Find all students and give their names, whose head teacher is Eordogh Krisztina!
*Add meg az összes hallgató nevét, akinek az osztályfőnöke Eördögh Krisztina!*

$$\pi_{student\_name}\left(student \bowtie \pi_{class\_id}\left(classes \bowtie \pi_{ssn}\left(\sigma_{teacher\_name="Eordogh\ Krisztina"}(teacher)\right)\right)\right)$$

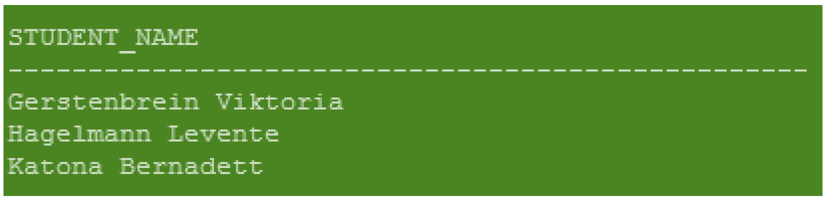
**SELECT** student_name

**FROM** student, classes, teacher

**WHERE** teacher.teacher_name = ‘Eordogh Krisztina’

  **AND** teacher.ssn = classes.head_teacher

  **AND** classes.class_id = student.class_id**;**



```
STUDENT_NAME
--------------------------------------------------
Gerstenbrein Viktoria
Hagelmann Levente
Katona Bernadett
```

*7. Figure: Result of the SQL statement above*

Find all teachers, who teach History in the level 8!

*Keresd meg az összes tanárt, aki történelmet tanít a 8. évfolyamon!*

$$\pi_{teacher\_name}\left(teacher \bowtie \pi_{ssn}\left(\sigma_{subject\_name="History"\ and\ level\ =\ 8}(teach)\right)\right)$$

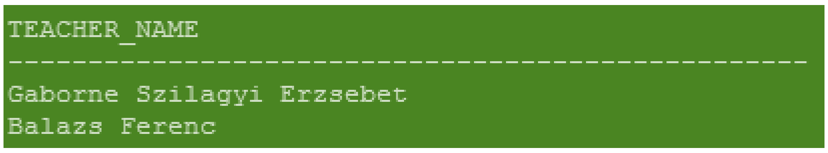
**SELECT** teacher_name

**FROM** teacher, teach

**WHERE** teach.subject_name = ‘History’

  **AND** teach.level = 8

  **AND** teach.teacher_id = teacher.ssn**;**



```
TEACHER_NAME
--------------------------------------------------
Gaborne Szilagyi Erzsebet
Balazs Ferenc
```

*8. Figure: Result of the SQL statement above*

Find teachers, who can retire in 10 years (We supposed, that the retirement age is 65)! Sort they decreasing order.

*Keresd meg azokat a tanárokat, akik 10 éven belül nyugdíjba mehetnek (Tegyük fel, hogy 65 évesen mehetnek nyugdíjba)! Rendezd őket életkoruk alapján csökkenő sorrendbe.*

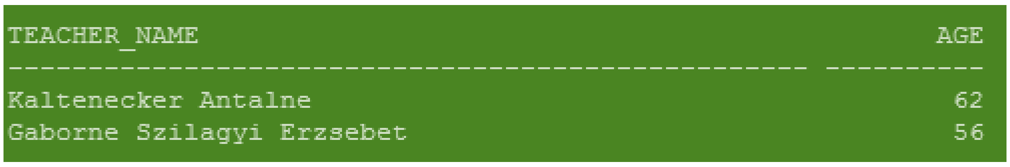
$$\pi_{teacher\_name,\ age}\left(\sigma_{65\text{-}age<10}\left(teacher\right)\right)$$

**SELECT** teacher_name, CAST( (TO_DATE(CURRENT_DATE,'YYYY-MM-DD') – TO_DATE(birth_date,'YYYY-MM-DD') ) / 365 AS INT) as age

**FROM** teacher

**WHERE** 65 - CAST( (TO_DATE(CURRENT_DATE,'YYYY-MM-DD') – TO_DATE(birth_date,'YYYY-MM-DD') ) / 365 AS INT) < 10

**ORDER BY** age desc;



```
TEACHER_NAME                                              AGE
-------------------------------------------------- ----------
Kaltenecker Antalne                                        62
Gaborne Szilagyi Erzsebet                                  56
```

*9. Figure: Result of the SQL statement above*

Find the full names and telephone numbers of all students, whose first name is Viktoria!

*Keresd meg azoknak a tanulóknak a teljes nevét és telefonszámát, akiket Viktóriának hívnak!*

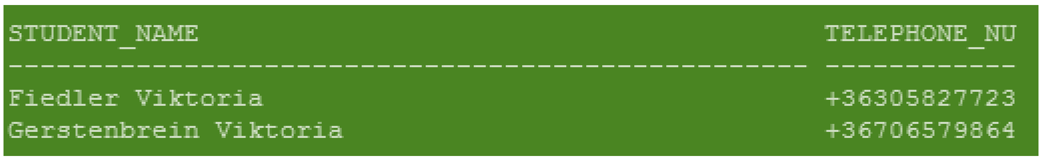
$$\pi_{student\_name,\ telephone\_number}\left(telephone \bowtie \sigma_{student\_name\ LIKE\ "Viktoria"}(student)\right)$$

**SELECT** student_name, telephone_number

**FROM** student, telephone

**WHERE** student.ssn = telephone.user_ssn

**AND** student.student_name LIKE ‘%Viktoria%’**;**



```
STUDENT_NAME                                         TELEPHONE_NU
---------------------------------------------------- ------------
Fiedler Viktoria                                     +36305827723
Gerstenbrein Viktoria                                +36706579864
```

*10. Figure: Result of the SQL statement above*

Find the manager of Kopjar Mate!

*Add meg Kopjár Máté felettesét!*

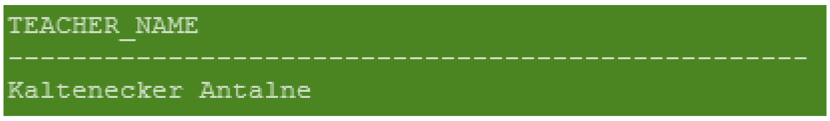
$$\pi_{manager.\ teacher\_name}\left(\rho_{manager}(teacher) \times \sigma_{teacher\_name="Kopjar\ Mate"}(teacher)\right)$$

**SELECT** manager.teacher_name

**FROM** teacher, teacher AS manager

**WHERE** teacher.teacher_name = ‘Kopjar Mate’

**AND** teacher.manager_ssn = manager.ssn**;**



```
TEACHER_NAME
----------------------------------------------------
Kaltenecker Antalne
```

*11. Figure: Result of the SQL statement above*

Give the number of teachers in every school!

*Add meg a tanárok számát tagozatonként / iskolánként!*

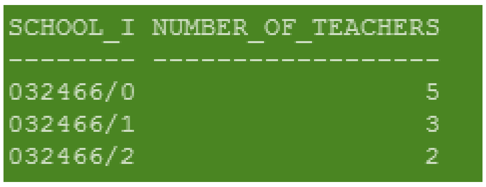
$$_{school\_id}g_{count(ssn)}(teacher)$$

**SELECT** school_id, COUNT(ssn) as number_of_teachers

**FROM** teachers

**GROUP BY** school_id

**ORDER BY** number_of_teachers desc**;**


```
SCHOOL_I NUMBER_OF_TEACHERS
-------- ------------------
032466/0                  5
032466/1                  3
032466/2                  2
```

*12. Figure: Result of the SQL statement above*

Give the number of students in each class from the school with 032466/0 education ID, where are at least 3 students! Put them in descending order.

*Add meg azon osztályok létszámát a 032466/0 OM kódú iskolából, amelyikben legalább három tanuló van! Tedd az eredményt csökkenő sorrendbe.*

$$_{class\_id}g_{count(ssn)}\left(\sigma_{school\_id="032466/0"}(student)\right)$$

**SELECT** class_id, COUNT(*) as number_of_students

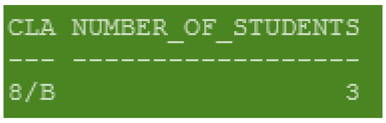
**FROM** student

**WHERE** school_id = ‘032466/0’

**GROUP BY** class_id

**HAVING** COUNT(*) >= 3

**ORDER BY** number_of_students desc**;**



```
CLA NUMBER_OF_STUDENTS
--- ------------------
8/B                  3
```

*13. Figure: Result of the SQL statement above*

Find the names of the students and teachers, who doesn’t live in Dunaharaszti!

*Add meg azoknak a tanulóknak és tanároknak a nevét, akik nem Dunaharasztin élnek!*

$$\pi_{student\_name}\left(\sigma_{address\ NOT\ LIKE\ "Dunaharaszti"}(student)\right) \cup \pi_{teacher\_name}\left(\sigma_{address\ NOT\ LIKE\ "Dunaharaszti"}(teacher)\right)$$

**SELECT** student_name names

**FROM** student

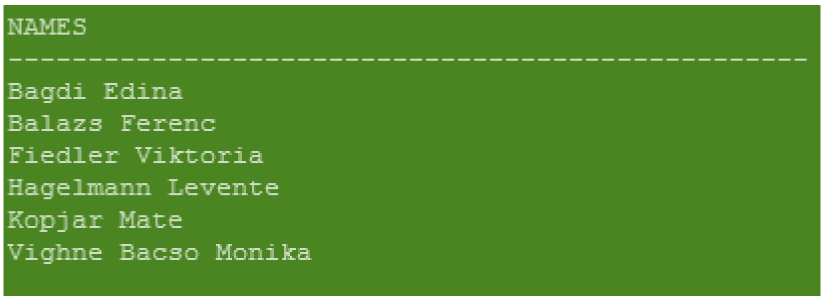
**WHERE** address NOT LIKE ‘%Dunaharaszti%’

**UNION**

**SELECT** teacher_name

**FROM** teacher

**WHERE** address NOT LIKE ‘%Dunaharaszti%’**;**



```
NAMES
--------------------------------------------------
Bagdi Edina
Balazs Ferenc
Fiedler Viktoria
Hagelmann Levente
Kopjar Mate
Vighne Bacso Monika
```

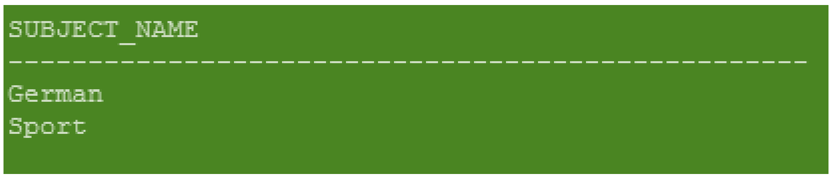
*14. Figure: Result of the SQL statement above*

Give the name of the subjects in level 1, what Drahosne Akocsi Ancilla doesn't teach in level 1!

*Add meg azoknak az elsős tantárgyaknak a nevét, melyeket nem tanít az első évfolyamon Drahosné Akócsi Ancilla!*

$$\pi_{subject\_name}\left(\sigma_{subject\_level=1}(subject)\right) - \pi_{subject\_name}\left(\sigma_{teacher_{name}="Drahosne\ Akocsi\ Ancilla"}(teacher) \bowtie \sigma_{subject\_level=1}(teach)\right)$$

```
SELECT subject_name
FROM subject
WHERE subject_level = 1
        AND subject_name NOT IN (SELECT subject_name
                                 FROM teacher, teach
                                 WHERE teacher.ssn = teach.teacher_id
                                       AND subject_level = 1
                                       AND teacher_name = 'Drahosne
                                                 Akocsi Ancilla');
```



```
SUBJECT_NAME
--------------------------------------------------
German
Sport
```

*15. Figure: Result of the SQL statement above*

Find the director of Hunyadi Janos Ground School! Give the name and telephone number of the director.

*Add meg a Hunyadi János Általános Iskola igazgatóját, jelenítsd meg a nevét, és telefonszámát.*

$$\pi_{teacher\_name,\ telephone\_number}\left(telephone \bowtie \sigma_{mgr.ssn=teacher.manager\_ssn}\left(teacher \times \rho_{mgr}(teacher)\right)\right)$$

**SELECT** teacher.teacher_name, telephone_number

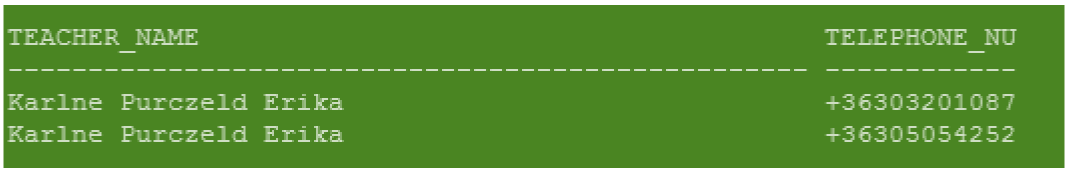
**FROM** teacher, telephone

**WHERE** teacher.ssn **IN (SELECT** mgr.manager_ssn

**FROM** teacher, teacher mgr

**WHERE** mgr.ssn = teacher.manager_ssn)

**AND** telephone.user_ssn = teacher.ssn;



```
TEACHER_NAME                                       TELEPHONE_NU
-------------------------------------------------- ------------
Karlne Purczeld Erika                              +36303201087
Karlne Purczeld Erika                              +36305054252
```

*16. Figure: Result of the SQL statement above*

More complex queries

Find the name of the teacher, who gave the homework to class 8/B, what they should hand in last!

*Add meg annak a tanárnak a nevét, aki a 8/B osztálynak a legkésőbb beadandó házi feladatát adta!*

$$\pi_{teacher\_name}\left(teacher \bowtie lesson \bowtie_{date\_to} g_{max(date\_to)}\left(\sigma_{class\_id="8/B"}(homework)\right)\right)$$

**SELECT** teacher_name

**FROM** homework, lesson, teacher

**WHERE** homework.class_id = lesson.class_id

**AND** homework.subject_name = lesson.subject_name

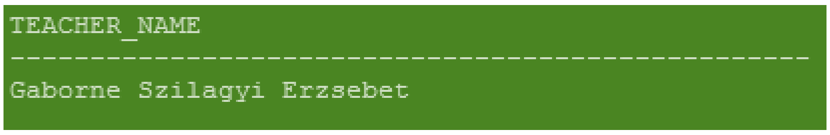
**AND** homework.subject_level = lesson.subject_level

**AND** teacher.ssn = lesson.teacher_id

**AND** date_to = **SOME (SELECT** MAX(HW.date_to)

**FROM** homework HW

**WHERE** HW.class_id = ‘8/B’ **);**



```
TEACHER_NAME
--------------------------------------------------
Gaborne Szilagyi Erzsebet
```

*17. Figure: Result of the SQL statement above*

Give the name of the teachers, who teach every subject in level 1.

*Add meg az összes olyan tanár nevét, aki minden tantárgyat tanít az első évfolyamon.*

$$\pi_{teacher\_name}\left(teacher \bowtie \left(\pi_{teacher\_id,\ subject\_name}\left(\sigma_{subject\_level=1}(teach)\right) \div \pi_{subject\_name}\left(\sigma_{subject\_level=1}(subject)\right)\right)\right)$$

**SELECT** teacher_name

**FROM** teach, teacher

**WHERE** subject_level = 1

**AND** teacher.ssn = teach.teacher_id

**AND** subject_name **IN**

**(SELECT** subject_name

**FROM** subject

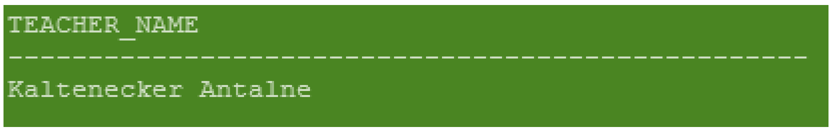
**WHERE** subject_level = 1 **)**

**GROUP BY** teacher_name

**HAVING** COUNT(*) = **(SELECT** COUNT(*)

**FROM** subject

**WHERE** subject_level = 1 **);**


```
TEACHER_NAME
--------------------------------------------------
Kaltenecker Antalne
```

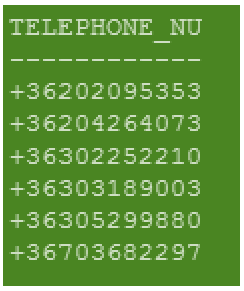
*18. Figure: Result of the SQL statement above*

Give the telephone numbers of parents in the class of Gaborne Szilagyi Erzsebet!

*Add meg a Gáborné Szilágyi Erzsébet osztályába járó gyerekek szüleinek a telefonszámát!*

$$\pi_{telephone\_number}\left(telephone \bowtie \pi_{parent\_id}\left(children \bowtie \pi_{ssn}\left(student \bowtie \pi_{class\_id}\left(classes \bowtie \pi_{ssn}\left(\sigma_{teacher\_name="Gaborne\ Szilagyi\ Erzsebet"}(teacher)\right)\right)\right)\right)\right)$$

```
SELECT telephone_number
FROM telephone
WHERE user_ssn IN (SELECT children.parent_id
                   FROM children, teacher, classes, student
                   WHERE teacher.teacher_name = ’Gaborne Szilagyi Erzsebet’
                         AND teacher.ssn = classes.head_teacher
                         AND student.class_id = class.class_id
                         AND children.child_id = student.ssn);
```



```
TELEPHONE_NU
------------
+36202095353
+36204264073
+36302252210
+36303189003
+36305299880
+36703682297
```

*19. Figure: Result of the SQL statement above*

Give the names of all teachers, who gave exactly 3 homework in 1 subject!

*Add meg azoknak a tanároknak a nevét, akik egy tantárgyból pontosan három házi feladatot adtak fel!*

$$\pi_{teacher\_name}\left(teacher \bowtie lesson \bowtie \pi_{subject\_name,\ subject\_level}\left( {}_{subject\_name,\ subject\_level}g_{count(*)}(homework)\right)\right)$$

**SELECT** teacher_name

**FROM** teacher, lesson

**WHERE** teacher.ssn = lesson.teacher_id

    **AND** lesson.subject_name **IN (SELECT** homework.subject_name

        **WHERE** lesson.class_id = homework.class_id

        **FROM** homework

        **GROUP BY** subject_name, subject_level

        **HAVING** COUNT(*) = 3 **)**

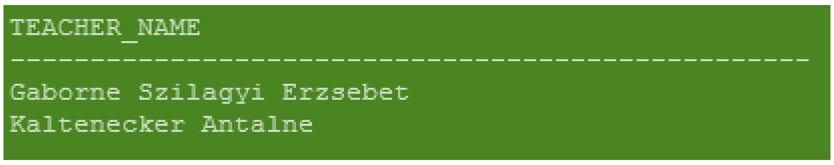
    **AND** lesson.subject_level **IN (SELECT** homework.subject_level

        **FROM** homework

        **WHERE** lesson.class_id = homework.class_id

        **GROUP BY** subject_name, subject_level

        **HAVING** COUNT(*) = 3 **);**



```
TEACHER_NAME
--------------------------------------------------
Gaborne Szilagyi Erzsebet
Kaltenecker Antalne
```

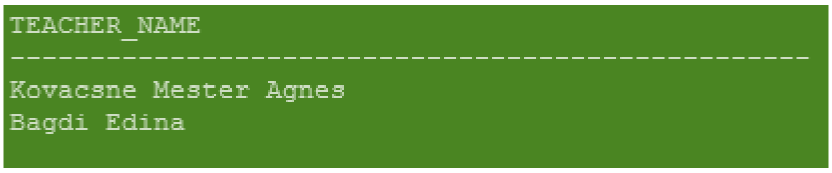
*20. Figure: Result of the SQL statement above*

Give the names of teachers, who teach what Karlne Pruczeld Erika!

*Add meg azoknak a tanároknak a nevét, akik tanítják azokat a tárgyakat, melyet Karlné Purczeld Erika tanít!*

$$teach \div \pi_{subject\_name}\left(teach \bowtie \pi_{ssn}\left(\sigma_{teacher\_name="Karlne\ Purczeld\ Erika"}(teacher)\right)\right)$$

```
SELECT teacher_name
FROM teach, teacher
WHERE teacher.ssn = teach.teacher_id
        AND teacher_name != 'Karlne Purczeld Erika'
        AND subject_name IN (SELECT subject_name
                             FROM teach, teacher
                             WHERE teacher.ssn = teach.teacher_id
                                     AND teacher.teacher_name = 'Karlne
                                                   Purczeld Erika' )
GROUP BY teacher_name
HAVING COUNT(*) = (SELECT COUNT(*)
                   FROM teach, teacher
                   WHERE teacher.ssn = teach.teacher_id
                           AND teacher.teacher_name = 'Karlne Purczeld
                                                   Erika' );
```



```
TEACHER_NAME
--------------------------------------------------
Kovacsne Mester Agnes
Bagdi Edina
```

*21. Figure: Result of the SQL statement above*

## Normal forms

The database should be normalized to avoid anomalies. Below I summarize the main normal forms. I will check my tables and decide in which normal form they are.

First Normal Form (1NF)[4]
In First Normal Form, any row must not have a column in which more than one value is saved, like separated with commas. Rather than that, we must separate such data into multiple rows.

Second Normal Form (2NF)
A database is in 2NF if it is in First Normal Form and all non-key attributes are fully functional dependent on any candidate key.

Third Normal Form (3NF)
A database is in Third Normal Form if it is in 2NF and there is no transitive functional dependency.

Boyce-Codd Normal Form (BCNF)[5]
A database is in Boyce-Codd Normal Form if it is in 3NF and attributes depend only on any super key.

BCNF is one of the most important normal form, because there is always a lossless decomposition in BCNF.

Other normal forms
A list about other normal forms, which are not discussed in this project.

Elementary Key Normal Form (EKNF)
EKNF tables are also in Third Normal Form. This happens when there is more than one unique composite key and they overlap. Such cases can cause redundant information in the overlapping column(s).

Fourth Normal Form (4NF)
4NF is the next level of normalization after Boyce-Codd Normal Form. Whereas the Second, Third, and Boyce-Codd Normal Forms are concerned with functional

[4] http://www.oracle.com/technetwork/issue-archive/2011/11-sep/o51sql-453459.html (visited: 10. December 2016)
[5] https://www.ischool.utexas.edu/~wyllys/DMPAMaterials/normstep.html (visited: 10. December 2016)

dependencies, 4NF is concerned with a more general type of dependency known as a multivalued dependency.

Fifth Normal Form (5NF)
A table is in the 5NF if and only if every non-trivial join dependency in it is implied by the candidate keys.

Sixth Normal Form (6NF)
A table is in Sixth Normal Form if and only if it satisfies no nontrivial join dependencies at all - where, as before, a join dependency is trivial if and only if at least one of the projections involved is taken over the set of all attributes of the table concerned.

Domain-Key Normal Form (DKNF)
The domain/key normal form is achieved when every constraint on the relation is a logical consequence of the definition of keys and domains, and enforcing key and domain restraints and conditions causes all constraints to be met.

The Third Normal Form, Boyce-Codd Normal Form, Fourth Normal Form and Fifth Normal Form are special cases of the domain/key normal form.

## Normalization

I will check the normal forms of all the tables. First, I write down the Functional Dependencies on the given table. After it, I can define in what normal form is the table. If it is necessary, I will decompose the table to reach the BCNF normal form.

SCHOOL (<u>education_id</u>, schoolname, address, email, web)

$F_{SCHOOL}$ = {education_id → schoolname, education_id → address, education_id → email, education_id → web}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

SCHOOL_TELEPHONE (school_id, telephone)

$F_{SCHOOL\_TELEPHONE}$ = {school_id → telephone}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

TEACHER (ssn, teacher_name, sex, address, birth_place, birth_date, email, education_id, date_of_start, manager_ssn, school_id)

$F_{TEACHER}$ = {ssn → teacher_name, ssn → sex, ssn → address, ssn → birth_place, ssn → birth_date, ssn → email, ssn → education_id, ssn → date_of_start, ssn → manager_ssn, ssn → school_id, education_id → ssn, education_id → teacher_name, education_id → sex, education_id → address, education_id → birth_place, education_id → birth_date, education_id → email, education_id → date_of_start, education_id → manager_ssn, education_id → school_id}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

PARENT (ssn, parent_name, sex, address, birth_place, birth_date, email, id)

$F_{PARENT}$ = {ssn → parent_name, ssn → sex, ssn → birth_place, ssn → birth_date, ssn → email, ssn → id, id → ssn, id → parent_name, id → sex, id → address, id → birth_place, id → email}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

STUDENT (ssn, student_name, sex, address, birth_place, birth_date, email, education_id, date_of_start, school_id, class_id)

F<sub>STUDENT</sub> = {ssn → parent_name, ssn → sex, ssn → address, ssn → birth_place, ssn → birth_date, ssn → email, ssn → education_id, ssn → date_of_start, ssn → school_id, ssn → class_id, education_id → ssn, education_id → name, education_id → sex, education_id → address, education_id → birth_place, education_id → birth_date, education_id → email, education_id → date_of_start, education_id → school_id, education_id → class_id}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

TELEPHONE (telephone-number, user_ssn)

F<sub>TELEPHONE</sub> = Ø

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

TEACH (teacher_id, subject_name, subject_level)

F<sub>TEACH</sub> = Ø

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

CLASS (class_id, date_of_start, head_teacher)

F<sub>CLASS</sub> = {class_id → date_of_start, class_id → head_teacher, head_teacher → class_id, head_teacher → date_of_start}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

SUBJECT (subject_name, subject_level, description)

F<sub>SUBJECT</sub> = {subject_name & subject_level → description}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

HOMEWORK (subject_name, subject_level, number_of_homeworks, class_id, date_from, date_to, title, description)

F<sub>HOMEWORK</sub> = {subject_name & subject_level & number_of_homeworks → class_id, subject_name & subject_level & number_of_homeworks → date_from, subject_name & subject_level & number_of_homeworks → date_to, subject_name & subject_level & number_of_homeworks → title, subject_name & subject_level & number_of_homeworks → description}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

CHILDREN (parent_id, child_id)

F<sub>CHILDREN</sub> = Ø

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

LESSON (class_id, subject_name, subject_level, teacher_id)

F<sub>LESSON</sub> = {class_id & subject_name & subject_level → teacher_id}

| Normal Forms | Fulfilled |
|---|---|
| 1NF | **YES** |
| 2NF | **YES** |
| 3NF | **YES** |
| BCNF | **YES** |

All tables, so the database is in Boyce-Codd Normal Form.

## Triggers

The triggers are special proceedings in a database. A trigger can include SQL an PL/SQL (Procedural Language / Standard Query Language) statements. This procedural code is automatically executed after /before instead of insert, update, delete.[6]

The trigger can allow or reject the changes on a table, logging, or can make other changes on the table, too.

In my sample database, I created a logging trigger. It means that, when I insert a row in the given table, update, or delete a row, the trigger will add a new row into my special logging table with some information about the action. You can see the code of the trigger below.

First, I created the logging table.

```
DROP TABLE classes_log CASCADE CONSTRAINTS;

CREATE TABLE classes_log

(

   log_timestamp TIMESTAMP,

   action VARCHAR2(20),

   class_id VARCHAR2(3),

   FOREIGN KEY(class_id) REFERENCES classes(class_id),

   PRIMARY KEY(log_timestamp)

);
```

*22. Figure: Create table statement of the logging statement*

[6] https://docs.oracle.com/cd/A57673_01/DOC/server/doc/SCN73/ch15.htm (visited: 10. December 2016)

```
CREATE OR REPLACE TRIGGER log_classes_events
            AFTER
                        INSERT OR
                        UPDATE OR
                        DELETE
            ON classes
            FOR EACH ROW
BEGIN
    CASE
                        WHEN INSERTING THEN
                                    INSERT INTO classes_log VALUES
                                    (
                                            systimestamp,
                                            'INSERT',
                                            :NEW.class_id
                                    );
                        WHEN UPDATING THEN
                                    INSERT INTO classes_log VALUES
                                    (
                                            systimestamp,
                                            'UPDATE',
                                           :NEW.class_id
                                   );
                        WHEN DELETING THEN
                                    INSERT INTO classes_log VALUES
                                    (
                                            systimestamp,
                                            'DELETE',
                                            :OLD.class_id
                                   );
            END CASE;
END;
/
```

*23. Figure: The code of my logging trigger*

# Attachments

## INSERT statements

```
INSERT INTO school VALUES
  ( '032466/0',
     'Hunyadi Janos Ground School Upper Classes',
     '2330 Dunaharaszti, Foldvari utca 15.',
     'hunyadidh@pr.hu',
     'http://hunyadidh.hu'  );
INSERT INTO school VALUES
  ( '032466/1',
     'Hunyadi Janos Ground School Lower Classes',
     '2330 Dunaharaszti, Fo ut 69.',
     'hunyadidh@pr.hu',
     'http://hunyadidh.hu' );
INSERT INTO school VALUES
  ( '032466/2',
     'Hunyadi Janos Ground School Member Institution',
     '2330 Dunaharaszti, Fo ut 268.',
     'hunyadidh@pr.hu',
     'http://hunyadidh.hu' );
INSERT INTO school VALUES
  ( '032467/0',
     'II. Rakoczi Ferenc Ground School',
     '2330 Dunaharaszti, Rakoczi ut 28.',
     'frakoczi@pr.hu',
     'http://frakoczi.hu' );
INSERT INTO school VALUES
  ( '032468/0',
     'Korosi Csoma Sandor Ground School',
     '2330 Dunaharaszti, Eotvos utca 52.',
     'hunyadidh@pr.hu',
     'http://csomaiskola.hu' );
INSERT INTO teacher VALUES
  ( '259175019',
     'Karlne Purczeld Erika',
     'female',
     '2330 Dunaharaszti, Foldvari utca 17',
     'Budapest',
     TO_DATE('1975-12-01','YYYY-MM-DD'),
     'karerika@gmail.com',
     '74555773959',
     TO_DATE('1992-09-01','YYYY-MM-DD'),
     NULL,
     '032466/0’ );
```

```
INSERT INTO teacher VALUES
 ( '259175020',
    'Vighne Bacso Monika',
    'female',
    '2338 Aporka, Dunasor utca 17',
    'Budapest',
    TO_DATE('1975-10-09','YYYY-MM-DD'),
    'vighma@indamail.hu',
    '74555773960',
    TO_DATE('1994-09-01','YYYY-MM-DD'),
    '259175019',
    '032466/0' );
INSERT INTO teacher VALUES
 ( '259175021',
    'Gaborne Szilagyi Erzsebet',
    'female',
    '2330 Dunaharaszti, Bartal Antal utca 2/C',
    'Gyula',
    TO_DATE('1960-12-29','YYYY-MM-DD'),
    'gaborneszilagyi@gmail.com',
    '74555773961',
    TO_DATE('1989-09-01','YYYY-MM-DD'),
    '25917020',
    '032466/0' );
INSERT INTO teacher VALUES
 ( '259175022',
    'Kovacsne Mester Agnes',
    'female',
    '2330 Dunaharaszti, Kazinczy Ferenc utca 15',
    'Budapest',
    TO_DATE('1970-09-10','YYYY-MM-DD'),
    'amester@pr.hu',
    '74555773962',
    TO_DATE('2000-09-01','YYYY-MM-DD'),
    '259170521',
    '032466/0' );
INSERT INTO teacher VALUES
 ( '259175023',
    'Balazs Ferenc',
    'male',
    '1215 Budapest, Bajcsy-Zsilinszky ut 40',
    'Budapest',
    TO_DATE('1986-06-14','YYYY-MM-DD'),
    'balazsferenc@gmail.com',
    '74555773963',
    TO_DATE('2016-09-01','YYYY-MM-DD'),
    '259170521',
    '032466/0' );
```

```
INSERT INTO teacher VALUES
 ( '259175024',
   'Kaltenecker Antalne',
   'female',
   '2330 Dunaharaszti, Semmelweis utca 5',
   'Budapest',
   TO_DATE('1955-04-17','YYYY-MM-DD'),
   'kalteneckerkati@gmail.com',
   '74555773964',
   TO_DATE('1994-09-01','YYYY-MM-DD'),
   '259175019',
   '032466/1' );
INSERT INTO teacher VALUES
 ( '259175025',
   'Bagdi Edina',
   'female',
   '2310 Szigetszentmiklos, Szent Miklos utja 3',
   'Gyula',
   TO_DATE('1984-07-26','YYYY-MM-DD'),
   'bagdie@gmail.com',
   '74555773965',
   TO_DATE('2007-09-01','YYYY-MM-DD'),
   '259175024',
   '032466/1' );
INSERT INTO teacher VALUES
 ( '259175026',
   'Kopjar Mate',
   'male',
   '2310 Szigetszentmiklos, Szent Miklos utja 3',
   'Gyula',
   TO_DATE('1984-07-26','YYYY-MM-DD'),
   'kopjarmate@gmail.com',
   '74555773965',
   TO_DATE('2007-09-01','YYYY-MM-DD'),
   '259175024',
   '032466/1' );
INSERT INTO teacher VALUES
 ( '259175027',
   'Drahosne Akocsi Ancilla',
   'female',
   '2330, Dunaharaszti, Kolcsey utca 22',
   'Budapest',
   TO_DATE('1965-03-04','YYYY-MM-DD'),
   'akocsiancilla@pr.hu',
   '74555773966',
   TO_DATE('1997-09-01','YYYY-MM-DD'),
   '259175019',
   '032466/2' );
```

```
INSERT INTO teacher VALUES
  ( '259175028',
     'Eordogh Krisztina',
     'female',
     '2330 Dunaharaszti, Szent Laszlo utca 55',
     'Budapest',
     TO_DATE('1974-02-10','YYYY-MM-DD'),
     'ecilike@gmail.com',
     '74555773967',
     TO_DATE('2007-09-01','YYYY-MM-DD'),
     '259175027',
     '032466/2' );
INSERT INTO school_telephone VALUES
  ( '+3624531020',
     '032466/0' );
INSERT INTO school_telephone VALUES
  ( '+36704912411',
     '032466/0' );
INSERT INTO school_telephone VALUES
  ( '+3624531031',
     '032466/1' );
INSERT INTO school_telephone VALUES
  ( '+36704912412',
     '032466/1' );
INSERT INTO school_telephone VALUES
  ( '+3624531040',
     '032466/2' );
INSERT INTO school_telephone VALUES
  ( '+36704912413',
     '032466/2' );
INSERT INTO school_telephone VALUES
  ( '+3624370253',
     '032467/0' );
INSERT INTO school_telephone VALUES
  ( '+36306765205',
     '032467/0' );
INSERT INTO school_telephone VALUES
  ( '+3624260374',
     '032468/0' );
INSERT INTO school_telephone VALUES
  ( '+3624260044',
     '032468/0' );
INSERT INTO parents VALUES
  (
     '670374911',
     'Becsakna Nagy Agnes',
     'female',
     '2330 Dunaharaszti, Szent Laszlo utca 30',
     'Budapest',
     TO_DATE('1976-08-01','YYYY-MM-DD'),
     'nagnes08@gmail.com',
     '404127KA'
  );
```

```
INSERT INTO parents VALUES
  (
     '670374912',
     'Becsak Tamas',
     'male',
     '2330 Dunaharaszti, Szent Laszlo utca 30',
     'Budapest',
     TO_DATE('1975-09-17','YYYY-MM-DD'),
     'btamas@gmail.com',
     '404137KA'
  );
INSERT INTO parents VALUES
  (
     '670374913',
     'Benedek Almos',
     'male',
     '2330 Dunaharaszti, Csengeri utca 2',
     'Budapest',
     TO_DATE('1978-11-04','YYYY-MM-DD'),
     'almosbenedek@invitel.hu',
     '404147KA'
  );
INSERT INTO parents VALUES
  (
     '670374914',
     'Benedekne Szavu Iren',
     'female',
     '2330 Dunaharaszti, Csengeri utca 2',
     'Budapest',
     TO_DATE('1977-01-20','YYYY-MM-DD'),
     'irenbenedek@invitel.hu',
     '404157KA'
  );
INSERT INTO parents VALUES
  (
     '670374915',
     'Fiedler Gyula',
     'male',
     '2315 Szigethalom, Fiumei ut 2',
     'Budapest',
     TO_DATE('1968-07-19','YYYY-MM-DD'),
     'fiedlergyula@gmail.com',
     '404167KA'
  );
INSERT INTO parents VALUES
  (
     '670374916',
     'Fiedlerne Szakacs Rita',
     'female',
     '2315 Szigethalom, Fiumei ut 2',
     'Budapest',
     TO_DATE('1971-06-10','YYYY-MM-DD'),
     'zitafiedler@gmail.com',
     '404177KA'
  );
```

```
INSERT INTO parents VALUES
  (
    '670374917',
    'Gerstenbrein Jozsef',
    'male',
    '2330 Dunaharaszti, Danko Pista utca 15',
    'Budapest',
    TO_DATE('1970-11-10','YYYY-MM-DD'),
    'gerstjozsef@freemail.hu',
    '404187KA'
  );
INSERT INTO parents VALUES
  (
    '670374918',
    'Meiszter Margit',
    'female',
    '2330 Dunaharaszti, Fo ut 120',
    'Budapest',
    TO_DATE('1971-12-06','YYYY-MM-DD'),
    'margitka71@freemail.hu',
    '404197KA'
  );
INSERT INTO parents VALUES
  (
    '670374919',
    'Hagelmann Zsolt',
    'male',
    '2336 Dunavarsany, Arnyas utca 33',
    'Budapest',
    TO_DATE('1971-04-04','YYYY-MM-DD'),
    'hagi71@freemail.hu',
    '404207KA'
  );
INSERT INTO parents VALUES
  (
    '670374920',
    'Marsal Mariann',
    'female',
    '2336 Dunavarsany, Arnyas utca 33',
    'Budapest',
    TO_DATE('1972-12-29','YYYY-MM-DD'),
    'marsalmariann@freemail.hu',
    '404217KA'
  );
INSERT INTO parents VALUES
  (
    '670374921',
    'Katona Tamas',
    'male',
    '2330 Dunaharaszti, Moricz Zsigmond utca 5',
    'Budapest',
    TO_DATE('1970-03-24','YYYY-MM-DD'),
    'harison34@gmail.com',
    '404227KA'
  );
```

```
INSERT INTO parents VALUES
  (
     '670374922',
     'Katona-Wolf Szilvia',
     'female',
     '2330 Dunaharaszti, Moricz Zsigmond utca 5',
     'Budapest',
     TO_DATE('1974-10-05','YYYY-MM-DD'),
     'wolfszilvi@gmail.com',
     '404237KA'
  );
INSERT INTO classes (class_id, date_of_start, head_teacher) VAlUES
  (
     '8/B',
     TO_DATE('2009-09-01','YYYY-MM-DD'),
     '259175021'
  );
INSERT INTO classes (class_id, date_of_start, head_teacher) VALUES
  (
     '1/A',
     TO_DATE('2016-09-01','YYYY-MM-DD'),
     '259175028'
  );
INSERT INTO student VALUES
  (
     '427710021',
     'Becsak Daniel',
     'male',
     '2330 Dunaharaszti, Szent Laszlo utca 30',
     'Budapest',
     TO_DATE('2002-08-14','YYYY-MM-DD'),
     'dani.becsak@gmail.com',
     '72169405748',
     '2009-09-01',
     '032466/0',
     '8/B'
  );
INSERT INTO student VALUES
  (
     '427710022',
     'Benedek Jennifer',
     'female',
     '2330 Dunaharaszti, Csengeri utca 2',
     'Budapest',
     TO_DATE('2003-03-30','YYYY-MM-DD'),
     'jeni.benedek@gmail.com',
     '72169405749',
     '2009-09-01',
     '032466/0',
     '8/B'
  );
```

```
INSERT INTO student VALUES
  (
     '427710023',
     'Fiedler Viktoria',
     'female',
     '2315 Szigethalom, Fiumei ut 2',
     'Budapest',
     TO_DATE('2002-08-22','YYYY-MM-DD'),
     'fiedler.viki@gmail.com',
     '72169405750',
     '2009-09-01',
     '032466/0',
     '8/B'
  );
INSERT INTO student VALUES
  (
     '427710024',
     'Gerstenbrein Viktoria',
     'female',
     '2330 Dunaharaszti, Danko Pista utca 15',
     'Budapest',
     TO_DATE('2003-03-24','YYYY-MM-DD'),
     'gerstenbrein.viki@gmail.com',
     '72169405751',
     '2009-09-01',
     '032466/1',
     '1/A'
  );
INSERT INTO student VALUES
  (
     '427710025',
     'Hagelmann Levente',
     'male',
     '2336 Dunavarsany, Arnyas utca 33',
     'Budapest',
     TO_DATE('2003-02-13','YYYY-MM-DD'),
     'hagelmann.levente@gmail.com',
     '72169405752',
     '2009-09-01',
     '032466/1',
     '1/A'
  );
INSERT INTO student VALUES
  (
     '427710026',
     'Katona Bernadett',
     'female',
     '2330 Dunaharaszti, Moricz Zsigmond utca 5',
     'Budapest',
     TO_DATE('2002-03-16','YYYY-MM-DD'),
     'katona.berni@gmail.com',
     '72169405753',
     '2009-09-01',
     '032466/1',
     '1/A'
  );
```

```
INSERT INTO children VALUES
 (
    '670374911',
    '427710021'
 );
INSERT INTO children VALUES
 (
    '670374912',
    '427710021'
 );
INSERT INTO children VALUES
 (
    '670374913',
    '427710022'
 );
INSERT INTO children VALUES
 (
    '670374914',
    '427710022'
 );
INSERT INTO children VALUES
 (
    '670374915',
    '427710023'
 );
INSERT INTO children VALUES
 (
    '670374916',
    '427710023'
 );
INSERT INTO children VALUES
 (
    '670374917',
    '427710024'
 );
INSERT INTO children VALUES
 (
    '670374918',
    '427710024'
 );
INSERT INTO children VALUES
 (
    '670374919',
    '427710025'
 );
INSERT INTO children VALUES
 (
    '670374920',
    '427710025'
 );
INSERT INTO children VALUES
 (
    '670374921',
    '427710026'
 );
```

```
INSERT INTO children VALUES
  (
     '670374922',
     '427710026'
  );
INSERT INTO telephone VALUES
  (
     '+36305774441',
     '427710021'
  );
INSERT INTO telephone VALUES
  (
     '+36309237919',
     '427710022'
  );
INSERT INTO telephone VALUES
  (
     '+36305827723',
     '427710023'
  );
INSERT INTO telephone VALUES
  (
     '+36706579864',
     '427710024'
  );
INSERT INTO telephone VALUES
  (
     '+362044445198',
     '427710025'
  );
INSERT INTO telephone VALUES
  (
     '+36306063636',
     '427710026'
  );
INSERT INTO telephone VALUES
  (
     '+36303201087',
     '259175019'
  );
INSERT INTO telephone VALUES
  (
     '+36305054252',
     '259175019'
  );
INSERT INTO telephone VALUES
  (
     '+36705773644',
     '259175020'
  );
INSERT INTO telephone VALUES
  (
     '+36203932926',
     '259175021'
  );
```

```
INSERT INTO telephone VALUES
 (
    '+36204545797',
    '259175022'
 );
INSERT INTO telephone VALUES
 (
    '+36706277704',
    '259175023'
 );
INSERT INTO telephone VALUES
 (
    '+36209928686',
    '259175024'
 );
INSERT INTO telephone VALUES
 (
    '+36307371171',
    '259175025'
 );
INSERT INTO telephone VALUES
 (
    '+36705696414',
    '259175026'
 );
INSERT INTO telephone VALUES
 (
    '+3637117767',
    '259175027'
 );
INSERT INTO telephone VALUES
 (
    '+36202574242',
    '259175028'
 );
INSERT INTO telephone VALUES
 (
    '+36703682297',
    '670374911'
 );
INSERT INTO telephone VALUES
 (
    '+36302252210',
    '670374912'
 );
INSERT INTO telephone VALUES
 (
    '+36204264073',
    '670374913'
 );
INSERT INTO telephone VALUES
 (
    '+36303189003',
    '670374914'
 );
```

```
INSERT INTO telephone VALUES
  (
    '+36202095353',
    '670374915'
  );
INSERT INTO telephone VALUES
  (
    '+36305299880',
    '670374916'
  );
INSERT INTO telephone VALUES
  (
    '+36209272282',
    '670374917'
  );
INSERT INTO telephone VALUES
  (
    '+36304493132',
    '670374918'
  );
INSERT INTO telephone VALUES
  (
    '+36303648484',
    '670374919'
  );
INSERT INTO telephone VALUES
  (
    '+36705644250',
    '670374920'
  );
INSERT INTO telephone VALUES
  (
    '+36703912025',
    '670374921'
  );
INSERT INTO telephone VALUES
  (
    '+36703912025',
    '670374922'
  );
INSERT INTO subject VALUES
  (
    'Grammar',
    8,
    'magyar nyelvtan 8. osztályosok részére'
  );
INSERT INTO subject VALUES
  (
    'German',
    8,
    'német nyelv 8. osztályosok részére'
  );
INSERT INTO subject VALUES
  (
    'Mathematic',
    8,
    'matematika 8. osztályosok részére'
  );
```

```
INSERT INTO subject VALUES
 (
   'History',
   8,
   'történelem 8. osztályosok részére'
 );
INSERT INTO subject VALUES
 (
   'Geography',
   8,
   'földrajz 8. osztályosok részére'
 );
INSERT INTO subject VALUES
 (
   'Geography',
   1,
   'környezetismeret 1. osztályosok részére'
 );
INSERT INTO subject VALUES
 (
   'Reading',
   1,
   'olvasás tanítása 1. osztályosok részére'
 );
INSERT INTO subject VALUES
 (
   'Sport',
   1,
   'testnevelés 1. osztályosok részére'
 );
INSERT INTO subject VALUES
 (
   'German',
   1,
   'német nyelv 1. osztályosok részére'
 );
INSERT INTO teach VALUES
 (
   '259175019',
   'German',
   8
 );
INSERT INTO teach VALUES
 (
   '259175019',
   'German',
   1
 );
INSERT INTO teach VALUES
 (
   '259175020',
   'Grammar',
   8
 );
```

```
INSERT INTO teach VALUES
 (
    '259175021',
    'Grammar',
    8
 );
INSERT INTO teach VALUES
 (
    '259175021',
    'History',
    8
 );
INSERT INTO teach VALUES
 (
    '259175022',
    'German',
    8
 );
INSERT INTO teach VALUES
 (
    '259175022',
    'German',
    1
 );
INSERT INTO teach VALUES
 (
    '259175022',
    'Grammar',
    8
 );
INSERT INTO teach VALUES
 (
    '259175023',
    'Geography',
    8
 );
INSERT INTO teach VALUES
 (
    '259175023',
    'Geography',
    1
 );
INSERT INTO teach VALUES
 (
    '259175023',
    'Mathematic',
    8
 );
INSERT INTO teach VALUES
 (
    '259175023',
    'History',
    8
 );
```

```
INSERT INTO teach VALUES
 (
    '259175024',
    'German',
    1
 );
INSERT INTO teach VALUES
 (
    '259175024',
    'Sport',
    1
 );
INSERT INTO teach VALUES
 (
    '259175024',
    'Reading',
    1
 );
INSERT INTO teach VALUES
 (
    '259175024',
    'Geography',
    1
 );
INSERT INTO teach VALUES
 (
    '259175025',
    'German',
    1
 );
INSERT INTO teach VALUES
 (
    '259175025',
    'German',
    8
 );
INSERT INTO teach VALUES
 (
    '259175025',
    'Sport',
    1
 );
INSERT INTO teach VALUES
 (
    '259175025',
    'Reading',
    1
 );
INSERT INTO teach VALUES
 (
    '259175026',
    'German',
    1
 );
```

```
INSERT INTO teach VALUES
 (
   '259175026',
   'Geography',
   1
 );
INSERT INTO teach VALUES
 (
   '259175027',
   'Reading',
   1
 );
INSERT INTO teach VALUES
 (
   '259175027',
   'Geography',
   1
 );
INSERT INTO teach VALUES
 (
   '259175028',
   'Reading',
   1
 );
INSERT INTO teach VALUES
 (
   '259175028',
   'German',
   1
 );
INSERT INTO teach VALUES
 (
   '259175028',
   'Sport',
   1
 );
INSERT INTO lesson VALUES
 (
   '8/B',
   'Grammar',
   8,
   '259175021'
 );
INSERT INTO lesson VALUES
 (
   '8/B',
   'History',
   8,
   '259175021'
 );
INSERT INTO lesson VALUES
 (
   '8/B',
   'German',
   8,
   '259175022'
 );
```

```
INSERT INTO lesson VALUES
 (
   '8/B',
   'Geography',
   8,
   '259175023'
 );
INSERT INTO lesson VALUES
 (
   '8/B',
   'Mathematic',
   8,
   '259175023'
 );
INSERT INTO lesson VALUES
 (
   '1/A',
   'Reading',
   1,
   '259175024'
 );
INSERT INTO lesson VALUES
 (
   '1/A',
   'Geography',
   1,
   '259175026'
 );
INSERT INTO lesson VALUES
 (
   '1/A',
   'Sport',
   1,
   '259175025'
 );
INSERT INTO lesson VALUES
 (
   '1/A',
   'Mathematic',
   1,
   '259175024'
 );
INSERT INTO homework VALUES
 (
   'Grammar',
   8,
   1,
   '8/B',
   TO_DATE('2016-09-15','YYYY-MM-DD'),
   TO_DATE('2016-09-22','YYYY-MM-DD'),
   'Ismétlés',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_magyar_1.pdf'
 );
```

```
INSERT INTO homework VALUES
  (
    'Grammar',
    8,
    2,
    '8/B',
    TO_DATE('2016-09-22','YYYY-MM-DD'),
    TO_DATE('2016-09-29','YYYY-MM-DD'),
    'Szavak',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_magyar_2.pdf'
  );
INSERT INTO homework VALUES
  (
    'Grammar',
    8,
    3,
    '8/B',
    TO_DATE('2016-09-29','YYYY-MM-DD'),
    TO_DATE('2016-10-06','YYYY-MM-DD'),
    'Mondatok',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_magyar_3.pdf'
  );
INSERT INTO homework VALUES
  (
    'German',
    8,
    1,
    '8/B',
    TO_DATE('2016-09-06','YYYY-MM-DD'),
    TO_DATE('2016-09-13','YYYY-MM-DD'),
    'Wiederholung',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_nemet_1.pdf'
  );
INSERT INTO homework VALUES
  (
    'German',
    8,
    2,
    '8/B',
    TO_DATE('2016-09-07','YYYY-MM-DD'),
    TO_DATE('2016-09-14','YYYY-MM-DD'),
    'Wiederholung 2',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_nemet_2.pdf'
  );
INSERT INTO homework VALUES
  (
    'German',
    8,
    3,
    '8/B',
    TO_DATE('2016-09-06','YYYY-MM-DD'),
    TO_DATE('2016-09-13','YYYY-MM-DD'),
    'Wiederholung 3',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_nemet_3.pdf'
  );
```

```
INSERT INTO homework VALUES
 (
   'German',
   8,
   4,
   '8/B',
   TO_DATE('2016-09-06','YYYY-MM-DD'),
   TO_DATE('2016-09-13','YYYY-MM-DD'),
   'Wiederholung 4',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_nemet_4.pdf'
 );
INSERT INTO homework VALUES
 (
   'Geography',
   8,
   1,
   '8/B',
   TO_DATE('2016-09-12','YYYY-MM-DD'),
   TO_DATE('2016-09-19','YYYY-MM-DD'),
   'Egyenletek ismétlése',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_matek_1.pdf'
 );
INSERT INTO homework VALUES
 (
   'Geography',
   8,
   2,
   '8/B',
   TO_DATE('2016-09-12','YYYY-MM-DD'),
   TO_DATE('2016-09-19','YYYY-MM-DD'),
   'Szöveges feladatok',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_8_b_matek_2.pdf'
 );
INSERT INTO homework VALUES
 (
   'Reading',
   1,
   1,
   '1/A',
   TO_DATE('2016-09-12','YYYY-MM-DD'),
   TO_DATE('2016-09-19','YYYY-MM-DD'),
   'Gyakorlás',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_olvasas_1.pdf'
 );
INSERT INTO homework VALUES
 (
   'Reading',
   1,
   2,
   '1/A',
   TO_DATE('2016-09-13','YYYY-MM-DD'),
   TO_DATE('2016-09-20','YYYY-MM-DD'),
   'Gyakorlás 2',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_olvasas_2.pdf'
 );
```

```
INSERT INTO homework VALUES
 (
    'Reading',
    1,
    3,
    '1/A',
    TO_DATE('2016-09-20','YYYY-MM-DD'),
    TO_DATE('2016-09-27','YYYY-MM-DD'),
    'Gyakorlás 3',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_olvasas_3.pdf'
 );
INSERT INTO homework VALUES
 (
    'Geography',
    1,
    1,
    '1/A',
    TO_DATE('2016-09-12','YYYY-MM-DD'),
    TO_DATE('2016-09-19','YYYY-MM-DD'),
    'Gyakorlás',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_kornyezet_1.pdf'
 );
INSERT INTO homework VALUES
 (
    'German',
    1,
    1,
    '1/A',
    TO_DATE('2016-09-10','YYYY-MM-DD'),
    TO_DATE('2016-09-17','YYYY-MM-DD'),
    'Der',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_nemet_1.pdf'
 );
INSERT INTO homework VALUES
 (
    'German',
    1,
    2,
    '1/A',
    TO_DATE('2016-09-10','YYYY-MM-DD'),
    TO_DATE('2016-09-17','YYYY-MM-DD'),
    'Die',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_nemet_2.pdf'
 );
INSERT INTO homework VALUES
 (
    'German',
    1,
    3,
    '1/A',
    TO_DATE('2016-09-10','YYYY-MM-DD'),
    TO_DATE('2016-09-17','YYYY-MM-DD'),
    'Das',
    'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_nemet_3.pdf'
 );
```

```
INSERT INTO homework VALUES
 (
   'German',
   1,
   4,
   '1/A',
   TO_DATE('2016-09-10','YYYY-MM-DD'),
   TO_DATE('2016-09-17','YYYY-MM-DD'),
   'Wiederholung',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_nemet_4.pdf'
 );
INSERT INTO homework VALUES
 (
   'German',
   1,
   5,
   '1/A',
   TO_DATE('2016-09-10','YYYY-MM-DD'),
   TO_DATE('2016-09-17','YYYY-MM-DD'),
   'Wiederholung',
   'A feladatok itt vannak: http://hunyadidh.hu/hf_1_a_nemet_5.pdf'
 );
```

*24. Figure: Insert statements*

## Digital Attachments

The SQL statements are also attached in .sql files to the project, with this files you are able to create my sample database.

The attached files:

- GABOR_CSABA_ATTILA_SQL.ZIP
    - GABOR_CSABA_ATTILA_FULL.SQL
    - GABOR_CSABA_ATTILA_DROP.SQL
    - GABOR_CSABA_ATTILA_CREATE.SQL
    - GABOR_CSABA_ATTILA_INSERT.SQL
    - GABOR_CSABA_ATTILA_SELECT.SQL
    - GABOR_CSABA_ATTILA_TRIGGER.SQL

To the project show, I made a Prezi, what you can watch here: https://goo.gl/Pi75X0.