Zaawansowane Bazy Danych

Grupa V, środa godzina 8.30

Projekt uczelni

Anna Kowalska 210230

Antonina Matuszek 210270

1. Schemat bazy danych

a) Założenia

Zaprojektowana baza danych ma pozwalać na zarządzanie wydziałem uczelni, umożliwia powiązanie studenta z grupą dziekańską oraz przedmiotami, na które chodzi oraz na określenie nauczycieli wykładających dany przedmiot i otrzymanych za dane przedmioty ocen.

Interpretacja relacji w bazie:

- teachers – dane wykładowców: id, imię, nazwisko, email, nr telefonu, data zatrudnienia, pensja.
- students – dane studentów: id, imię, nazwisko, nr telefonu, data urodzenia, id grupy.
- subjects – stanowi wykaz wszystkich przedmiotów wraz z informacją o prowadzącym przedmiot.
- buildings– zawiera nazwy budynków, w których znajdują się sale zajęciowe.
- classrooms– zawiera nazwy sal, w których znajdują się zajęcia.
- timetables – łączy studenta ze szczegółami jego zajęć (przedmioty i prowadzący).
- groups – zawiera maksymalną i minimalną możliwą liczbę uczestników grup dziekańskich.
- field_of_study – zawiera nazwy kierunków studiów na wydziale.
- academic_year - łączy oceny końcowe oraz grupę z rokiem akademickim.
- group_types – dane grup dziekańskich: id, nazwa, typ, rok akademicki, kierunek studiów.
- grades – zawiera wystawione oceny (id, nazwa, data wstawienia, przedmiot, student, notatka).
- albums – album studenta; stanowi powiązanie studenta z jego końcowymi ocenami.
- final_grades – zawiera końcowe oceny (id albumu, ocena, przedmiot, rok akademicki).

b) Skrypt tworzący bazę danych

```
if exists(select 1 from master.dbo.sysdatabases where name = 'UNIVERSITY') drop database UNIVERSITY
GO
CREATE DATABASE UNIVERSITY
GO

CREATE TABLE UNIVERSITY..teachers
   ( teacher_id    INT NOT NULL
   , CONSTRAINT     teacher_id_uk UNIQUE (teacher_id)
   , first_name     VARCHAR(20)
   , last_name      VARCHAR(25)
            CONSTRAINT     teacher_last_name_nn  NOT NULL
   , email          VARCHAR(25)
            CONSTRAINT     teacher_email_nn  NOT NULL
   , phone_number   VARCHAR(20)
   , hire_date      DATETIME
   , salary         MONEY
   , CONSTRAINT     teacher_salary_min CHECK (salary > 0)
   , CONSTRAINT     teacher_email_uk UNIQUE (email)
  ,CONSTRAINT PKTeacher_id PRIMARY KEY (teacher_id)
   ) ;

CREATE TABLE UNIVERSITY..students
   ( student_id    INT NOT NULL
          , CONSTRAINT     student_id_uk UNIQUE (student_id)
   , first_name     VARCHAR(20)
   , last_name      VARCHAR(25)
            CONSTRAINT     student_last_name_nn  NOT NULL
   , phone_number   VARCHAR(20)
   , birth_date     DATETIME
            CONSTRAINT     student_birth_date_nn  NOT NULL
          , CONSTRAINT student_id_pk PRIMARY KEY (student_id)
          , group_id    INT
          ) ;

CREATE TABLE UNIVERSITY..subjects
   ( subject_id    INT NOT NULL
          , CONSTRAINT     subject_id_uk UNIQUE (subject_id)
   , subject_name  VARCHAR(30)
          , main_teacher_id      INT
            CONSTRAINT  subject_name_nn  NOT NULL
          , CONSTRAINT PKSubject PRIMARY KEY (subject_id)
          , CONSTRAINT main_teacher_id_fk FOREIGN KEY (main_teacher_id) REFERENCES teachers (teacher_id)
   ) ;

CREATE TABLE UNIVERSITY..buildings
   ( building_id   INT NOT NULL
   , building_name   VARCHAR(20) NOT NULL
          , CONSTRAINT building_id_pk PRIMARY KEY (building_id )

   ) ;
```

```
        CREATE TABLE UNIVERSITY..classrooms
        ( classroom_id    INT NOT NULL
         , classroom_name  VARCHAR(20) NOT NULL
        , building_id   INT NOT NULL
        , CONSTRAINT classroom_id_pk PRIMARY KEY (classroom_id )
        , CONSTRAINT FKClassroomBuildings FOREIGN KEY (building_id ) REFERENCES buildings(building_id )
  ) ;

CREATE TABLE UNIVERSITY..timetables
  ( subject_id    INT NOT NULL
  , student_id  INT NOT NULL
        , teacher_id  INT NOT NULL
        , classroom_id    INT NOT NULL
        ,CONSTRAINT FKStudentTimeTable FOREIGN KEY (student_id) REFERENCES students(student_id)
        ,CONSTRAINT FKSubjectTimeTable FOREIGN KEY (subject_id) REFERENCES subjects(subject_id)
        ,CONSTRAINT FKTeacher FOREIGN KEY (teacher_id) REFERENCES teachers(teacher_id)
        ,CONSTRAINT FKTimeTableClassroom FOREIGN KEY (classroom_id) REFERENCES classrooms(classroom_id)
  ) ;

CREATE TABLE UNIVERSITY..groups
  ( group_id    INT NOT NULL
        , min_nr    INT NOT NULL
        , max_nr    INT NOT NULL
        , CONSTRAINT    group_id_uk UNIQUE (group_id)
  ) ;

ALTER TABLE UNIVERSITY..students ADD CONSTRAINT FKStudentGroup FOREIGN KEY (group_id) REFERENCES groups
(group_id);

CREATE TABLE UNIVERSITY..field_of_study
  ( field_of_study_id    INT NOT NULL
        , field_of_study_name  VARCHAR(30)
        , CONSTRAINT field_of_study_id_pk PRIMARY KEY (field_of_study_id )

  ) ;

CREATE TABLE UNIVERSITY..academic_year
  ( academic_year_id INT
        , academic_year_name   Datetime NOT NULL
        , constraint PKAcademicYear PRIMARY key (academic_year_id)
        ) ;
```

```
CREATE TABLE UNIVERSITY..group_types
  ( group_id    INT NOT NULL
        , group_name  VARCHAR(30)
          CONSTRAINT  group_name_nn  NOT NULL
        , group_type   VARCHAR(30)
        , academic_year INT NOT NULL
        , field_of_study_id INT NOT NULL
        , CONSTRAINT FKGroupTypeGroup FOREIGN KEY (group_id) REFERENCES groups (group_id)
        , CONSTRAINT TYPE_check_ CHECK(group_type IN('FULL TIME','PART TIME'))
        , CONSTRAINT FKGroupTypesAcademicYear foreign key (academic_year) references
academic_year(academic_year_id)
  ) ;

ALTER TABLE UNIVERSITY..group_types ADD CONSTRAINT FKGroupTypesFieldOfStudy FOREIGN KEY
(field_of_study_id ) REFERENCES field_of_study(field_of_study_id );


CREATE TABLE UNIVERSITY..grades (
        grade_id int NOT NULL IDENTITY(1,1),
        grade_name  int NOT NULL,
        grade_date DATETIME,
  subject_id  int NOT NULL,
        student_id  int NOT NULL,
        note VARCHAR(30) NOT NULL,
        constraint PKGrade PRIMARY key (grade_id),
        constraint FKSubjectG foreign key (subject_id) references subjects(subject_id),
        constraint FKStudentG foreign key (student_id) references students(student_id),
        CONSTRAINT grade_check_name CHECK(grade_name IN ('1', '2', '3', '4','5','6'))
);

GO

CREATE TABLE UNIVERSITY..albums
  ( album_id INT
        , student_id   INT
        , constraint PKAlbum PRIMARY key (album_id)
        ,constraint FKAlbumsStudent foreign key (student_id) references students(student_id)
        ) ;

CREATE TABLE UNIVERSITY..final_grades (
        album_id int NOT NULL,
        final_grade  int,
  subject_id  int NOT NULL,
        academic_year int NOT NULL,
        constraint FKFinalGradesSubject foreign key (subject_id) references subjects(subject_id),
        constraint FKFinalGradesAlbums foreign key (album_id) references albums(album_id),
        constraint FKFinalGradesAcademicYear foreign key (academic_year) references
academic_year(academic_year_id)
);
```

c) Skrypt wypełniający bazę danych przykładowymi danymi

```
USE UNIVERSITY;
----------------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..teachers VALUES ( 100, 'Nancy', 'Greenberg', 'NGREENBE', '515.124.4569', CAST('17-AUG-
1994'AS DATETIME), 1700);
INSERT INTO UNIVERSITY..teachers VALUES ( 110, 'Daniel', 'Faviet', 'DFAVIET', '515.124.4169', CAST('16-AUG-1994'AS
DATETIME), 2300);
INSERT INTO UNIVERSITY..teachers VALUES ( 120, 'John', 'Chen', 'JCHEN', '515.124.4269', CAST('28-SEP-1997'AS
DATETIME), 2100);
INSERT INTO UNIVERSITY..teachers VALUES ( 130, 'Luis', 'Popp', 'LPOPP', '515.124.4567', CAST('07-DEC-1999'AS
DATETIME), 2500);
INSERT INTO UNIVERSITY..teachers VALUES ( 140, 'Anastasia', 'Grey', 'AGREY', '515.124.3213', CAST('17-JAN-1999'AS
DATETIME), 2250);
INSERT INTO UNIVERSITY..teachers VALUES ( 150, 'Katniss', 'Everdeen', 'KEVER', 'NULL', CAST('12-FEB-1989'AS
DATETIME), 2670);
INSERT INTO UNIVERSITY..teachers VALUES ( 160, 'Nikogo', 'Nieucze', 'kasuj', 'NULL', CAST('12-FEB-1999'AS
DATETIME), 2670);
UPDATE teachers SET phone_number ='515.124.3222' WHERE teacher_id =150;

SELECT * FROM teachers;
-------------------------------------------------------------------
INSERT INTO UNIVERSITY..groups VALUES (10,1,250);  --id,min, max
INSERT INTO UNIVERSITY..groups VALUES (11,1,250);
INSERT INTO UNIVERSITY..groups VALUES (12,1,250);
INSERT INTO UNIVERSITY..groups VALUES (13,1,250);
INSERT INTO UNIVERSITY..groups VALUES (14,1,250);
INSERT INTO UNIVERSITY..groups VALUES (15,1,250);
INSERT INTO UNIVERSITY..groups VALUES (16,1,250);
INSERT INTO UNIVERSITY..groups VALUES (17,1,250);
INSERT INTO UNIVERSITY..groups VALUES (18,1,250);

INSERT INTO UNIVERSITY..groups VALUES (20,1,250);
INSERT INTO UNIVERSITY..groups VALUES (30,1,250);
INSERT INTO UNIVERSITY..groups VALUES (40,1,250);
SELECT * FROM groups;
-----------------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..field_of_study VALUES ( 1,'IT');
INSERT INTO UNIVERSITY..field_of_study VALUES ( 2,'Mathematics');
INSERT INTO UNIVERSITY..field_of_study VALUES ( 3,'Physics');

SELECT * FROM field_of_study;
-----------------------------------------------------------------------------------------------
```

```
INSERT INTO UNIVERSITY..academic_year VALUES (1, '2017/11/10');
SELECT * FROM academic_year;
-----------------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..group_types VALUES ( 10,'10','FULL TIME',1,1);
INSERT INTO UNIVERSITY..group_types VALUES ( 11,'11','FULL TIME',1,1);
INSERT INTO UNIVERSITY..group_types VALUES ( 12,'12','PART TIME',1,1);

INSERT INTO UNIVERSITY..group_types VALUES ( 13,'13','FULL TIME',1,1);
INSERT INTO UNIVERSITY..group_types VALUES ( 14,'14','FULL TIME',1,1);
INSERT INTO UNIVERSITY..group_types VALUES ( 15,'15','PART TIME',1,1);

INSERT INTO UNIVERSITY..group_types VALUES ( 16,'16','FULL TIME',1,1);
INSERT INTO UNIVERSITY..group_types VALUES ( 17,'17','FULL TIME',1,1);
INSERT INTO UNIVERSITY..group_types VALUES ( 18,'18','PART TIME',1,1);

INSERT INTO UNIVERSITY..group_types VALUES ( 20,'20','FULL TIME',1,2);
INSERT INTO UNIVERSITY..group_types VALUES ( 30,'30','FULL TIME',1,3);
INSERT INTO UNIVERSITY..group_types VALUES ( 40,'40','PART TIME',1,3);

SELECT * FROM group_types;
-----------------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..students VALUES ( 600, 'Susan', 'Mavris', '516.124.4567', CAST('13-DEC-1996'AS
DATETIME),10);
INSERT INTO UNIVERSITY..students VALUES ( 610, 'Hermann', 'Baer', '516.125.4547', CAST('01-JUL-1996'AS
DATETIME),10);
INSERT INTO UNIVERSITY..students VALUES ( 620, 'Shelley', 'Higgins', '513.124.4190', CAST('07-JUN-1997'AS
DATETIME),10);
INSERT INTO UNIVERSITY..students VALUES ( 630, 'William', 'Gietz', '519.128.4257', CAST('22-OCT-1998'AS
DATETIME),10);
INSERT INTO UNIVERSITY..students VALUES ( 640, 'Lilly', 'Collins', '516.125.1247', CAST('01-SEP-1998'AS
DATETIME),10);

INSERT INTO UNIVERSITY..students VALUES ( 650, 'Nastya', 'Kasnikov', '517.145.6747', CAST('19-SEP-1998'AS
DATETIME),11);
INSERT INTO UNIVERSITY..students VALUES ( 660, 'Rodion', 'Raskolnikov', '512.325.5555', CAST('30-MAR-1999'AS
DATETIME),11);
INSERT INTO UNIVERSITY..students VALUES ( 670, 'Harry', 'Potter', '511.225.6666', CAST('05-APR-1999'AS
DATETIME),11);
INSERT INTO UNIVERSITY..students VALUES ( 680, 'Peter', 'Parker', '516.725.7777', CAST('03-MAY-1999'AS
DATETIME),11);

INSERT INTO UNIVERSITY..students VALUES ( 690, 'Alexander', 'Grayson', '546.325.7757', CAST('09-FEB-1998'AS
DATETIME),12);
INSERT INTO UNIVERSITY..students VALUES ( 700, 'Bronagh', 'Murphy', '546.376.7457', CAST('18-OCT-1997'AS
DATETIME),12);
INSERT INTO UNIVERSITY..students VALUES ( 710, 'Tony', 'Stark', '946.325.7370', CAST('11-OCT-1997'AS
DATETIME),12);
INSERT INTO UNIVERSITY..students VALUES ( 720, 'Neal', 'Caffrey', '846.325.8757', CAST('23-JUN-1998'AS
DATETIME),12);
```

```
INSERT INTO UNIVERSITY..students VALUES ( 730, 'Geralt', 'Zrivii', '511.325.7757', CAST('09-JUN-1998'AS
DATETIME),13);
INSERT INTO UNIVERSITY..students VALUES ( 740, 'Jaskier', 'Bard', '512.325.7757', CAST('01-JUL-1998'AS
DATETIME),13);
INSERT INTO UNIVERSITY..students VALUES ( 750, 'Yennefer', 'Vengerberg', '513.325.7755', CAST('01-DEC-1998'AS
DATETIME),13);
INSERT INTO UNIVERSITY..students VALUES ( 760, 'Tris', 'Merigold', '514.325.7757', CAST('19-MAY-1998'AS
DATETIME),13);
INSERT INTO UNIVERSITY..students VALUES ( 770, 'Cirilla', 'Riannon', '516.325.7757', CAST('15-SEP-1998'AS
DATETIME),13);

INSERT INTO UNIVERSITY..students VALUES ( 780, 'Daenerys', 'Targaryen', '516.325.7157', CAST('11-JUN-1997'AS
DATETIME),14);
INSERT INTO UNIVERSITY..students VALUES ( 790, 'Jon', 'Snow', '516.325.7337', CAST('18-AUG-1997'AS
DATETIME),14);
INSERT INTO UNIVERSITY..students VALUES ( 800, 'Arya', 'Stark', '516.325.7227', CAST('20-FEB-1997'AS
DATETIME),14);
INSERT INTO UNIVERSITY..students VALUES ( 810, 'Tyrion', 'Lannister', '516.325.7057', CAST('22-SEP-1997'AS
DATETIME),14);
INSERT INTO UNIVERSITY..students VALUES ( 820, 'Cersei', 'Lannister', '516.325.7887', CAST('05-MAR-1997'AS
DATETIME),14);
INSERT INTO UNIVERSITY..students VALUES ( 830, 'Sansa', 'Stark', '516.325.0007', CAST('09-APR-1997'AS
DATETIME),14);
INSERT INTO UNIVERSITY..students VALUES ( 840, 'Khal', 'Drogo', '516.325.5557', CAST('26-JUL-1997'AS
DATETIME),14);

INSERT INTO UNIVERSITY..students VALUES ( 850, 'Joffrey', 'Baratheon', '516.005.5557', CAST('26-OCT-1996'AS
DATETIME),15);
INSERT INTO UNIVERSITY..students VALUES ( 860, 'Petyr', 'Baelish', '516.895.5557', CAST('22-JUL-1996'AS
DATETIME),15);
INSERT INTO UNIVERSITY..students VALUES ( 870, 'Sandor', 'Clegane', '516.355.5557', CAST('24-SEP-1998'AS
DATETIME),15);
INSERT INTO UNIVERSITY..students VALUES ( 880, 'Melisandre', 'Red', '516.225.5557', CAST('13-JUL-1994'AS
DATETIME),15);
INSERT INTO UNIVERSITY..students VALUES ( 890, 'Margaery', 'Tyrell', '516.995.5557', CAST('07-AUG-1997'AS
DATETIME),15);

INSERT INTO UNIVERSITY..students VALUES ( 900, 'Theon', 'Greyjoy', '516.325.1157', CAST('21-OCT-1997'AS
DATETIME),16);
INSERT INTO UNIVERSITY..students VALUES ( 910, 'Brienne', 'Tarthu', '516.325.5117', CAST('22-DEC-1997'AS
DATETIME),16);
INSERT INTO UNIVERSITY..students VALUES ( 920, 'Jaime', 'Lannister', '516.325.5511', CAST('25-DEC-1997'AS
DATETIME),16);
INSERT INTO UNIVERSITY..students VALUES ( 930, 'Ygritte', 'Wildone', '516.325.1117', CAST('06-JUL-1997'AS
DATETIME),16);

INSERT INTO UNIVERSITY..students VALUES ( 940, 'Robb', 'Stark', '516.325.1122', CAST('05-MAY-1996'AS
DATETIME),17);
INSERT INTO UNIVERSITY..students VALUES ( 950, 'Daario', 'Naharis', '333.325.1117', CAST('06-AUG-1996'AS
DATETIME),17);
INSERT INTO UNIVERSITY..students VALUES ( 960, 'Oberyn', 'Martell', '446.325.1117', CAST('13-DEC-1997'AS
DATETIME),17);
INSERT INTO UNIVERSITY..students VALUES ( 970, 'Patrick', 'Jane', '555.325.1117', CAST('15-SEP-1998'AS
DATETIME),17);
INSERT INTO UNIVERSITY..students VALUES ( 980, 'Teresa', 'Lisbon', '566.325.1117', CAST('24-JUL-1996'AS
DATETIME),17);
```

```
INSERT INTO UNIVERSITY..students VALUES ( 990, 'Grace', 'Van Pelt', '999.325.1117', CAST('06-JUL-1998'AS
DATETIME),18);
INSERT INTO UNIVERSITY..students VALUES ( 1000, 'Kimball', 'Cho', '589.325.1117', CAST('11-AUG-1998'AS
DATETIME),18);
INSERT INTO UNIVERSITY..students VALUES ( 1010, 'Wayne', 'Rigsby', '578.325.1117', CAST('27-FEB-1998'AS
DATETIME),18);
INSERT INTO UNIVERSITY..students VALUES ( 1020, 'Erica', 'Flynn', '515.325.1117', CAST('30-APR-1998'AS DATETIME),18);

INSERT INTO UNIVERSITY..students VALUES ( 1030, 'Sherlock', 'Holmes', '515.742.1117', CAST('30-APR-1998'AS
DATETIME),20);
INSERT INTO UNIVERSITY..students VALUES ( 1040, 'Oliver', 'Queen', '515.722.1117', CAST('30-APR-1997'AS
DATETIME),20);
INSERT INTO UNIVERSITY..students VALUES ( 1050, 'Felicity', 'Smoak', '515.244.1117', CAST('30-APR-1999'AS
DATETIME),20);
INSERT INTO UNIVERSITY..students VALUES ( 1060, 'Roy', 'Harper', '515.888.1117', CAST('30-APR-1996'AS DATETIME),20);
INSERT INTO UNIVERSITY..students VALUES ( 1070, 'Thea', 'Queen', '515.555.1117', CAST('30-APR-1996'AS DATETIME),20);
INSERT INTO UNIVERSITY..students VALUES ( 1080, 'John', 'Diggle', '515.000.1117', CAST('30-APR-1998'AS DATETIME),20);

INSERT INTO UNIVERSITY..students VALUES ( 1090, 'Sara', 'Lance', '515.355.1117', CAST('30-APR-1997'AS DATETIME),30);
INSERT INTO UNIVERSITY..students VALUES ( 1100, 'Malcolm', 'Merlyn', '515.222.1117', CAST('30-APR-1997'AS
DATETIME),30);
INSERT INTO UNIVERSITY..students VALUES ( 1110, 'Slade', 'Wilson', '525.775.1117', CAST('30-APR-1996'AS
DATETIME),30);
INSERT INTO UNIVERSITY..students VALUES ( 1120, 'Isabel', 'Rochev', '515.835.1117', CAST('30-APR-1996'AS
DATETIME),30);
INSERT INTO UNIVERSITY..students VALUES ( 1130, 'Liv', 'Moore', '522.382.1117', CAST('30-APR-1999'AS DATETIME),30);
INSERT INTO UNIVERSITY..students VALUES ( 1140, 'Blaine', 'DeBeers', '555.245.1117', CAST('30-APR-1999'AS
DATETIME),30);

INSERT INTO UNIVERSITY..students VALUES ( 1150, 'Ed', 'Sheeran', '524.177.7217', CAST('20-MAY-1997'AS DATETIME),40);
INSERT INTO UNIVERSITY..students VALUES ( 1160, 'Selena', 'Gomez', '523.137.7217', CAST('22-JUL-1992'AS
DATETIME),40);

SELECT * FROM students;
-----------------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..subjects VALUES ( 1, 'SYSOPY',100);
INSERT INTO UNIVERSITY..subjects VALUES ( 2, 'ZSBD',110);
INSERT INTO UNIVERSITY..subjects VALUES ( 3, 'TP',120);
INSERT INTO UNIVERSITY..subjects VALUES ( 4, 'WDI',100);
INSERT INTO UNIVERSITY..subjects VALUES ( 5, 'PROKOMP',130);
INSERT INTO UNIVERSITY..subjects VALUES ( 6, 'POBI',130);
INSERT INTO UNIVERSITY..subjects VALUES ( 7, 'ENGLISH',140);
INSERT INTO UNIVERSITY..subjects VALUES ( 8, 'PE',140);
INSERT INTO UNIVERSITY..subjects VALUES ( 9, 'EMBEDDED',120);
INSERT INTO UNIVERSITY..subjects VALUES ( 10, 'CAD',150);

SELECT * FROM subjects;
```

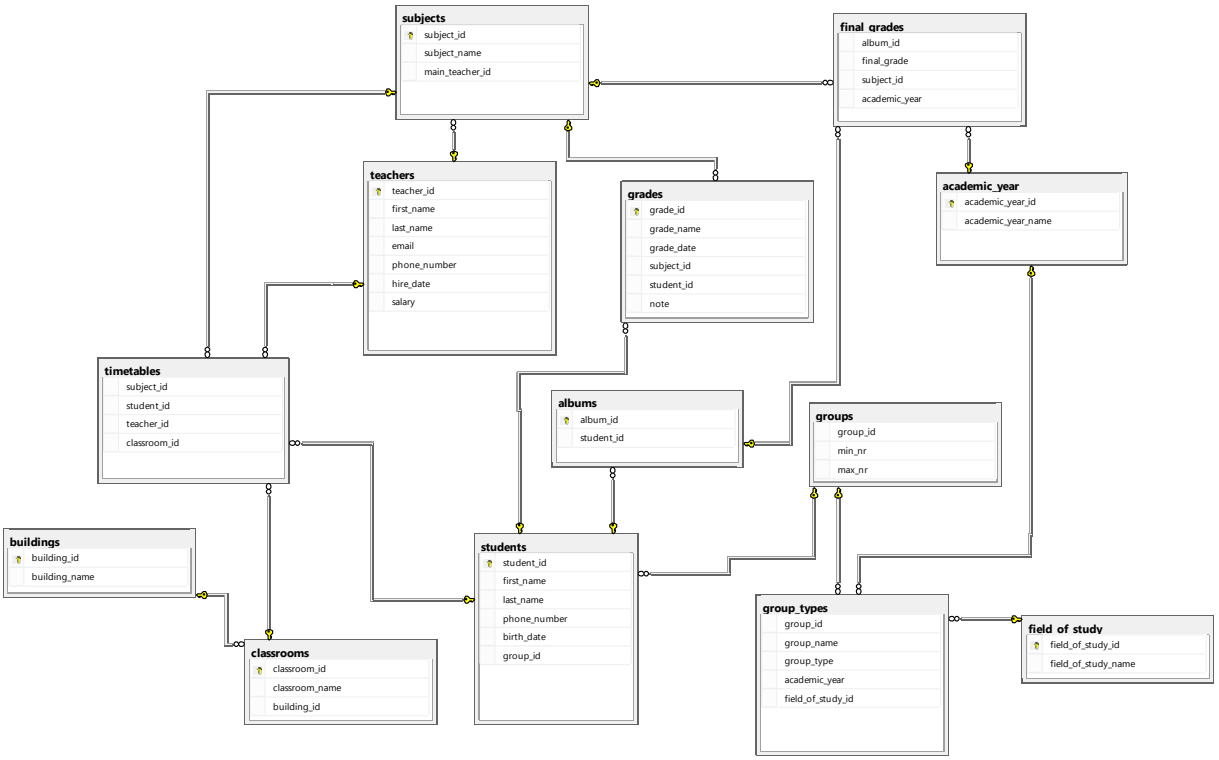
```
INSERT INTO UNIVERSITY..buildings VALUES ( 1, 'B9');
INSERT INTO UNIVERSITY..buildings VALUES ( 2, 'CTI');
INSERT INTO UNIVERSITY..buildings VALUES ( 3, 'B14');

SELECT * FROM buildings;
-----------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..classrooms VALUES ( 1, 'F10',1);
INSERT INTO UNIVERSITY..classrooms VALUES ( 2, 'AULA',3);
INSERT INTO UNIVERSITY..classrooms VALUES ( 3, 'F.04',3);
INSERT INTO UNIVERSITY..classrooms VALUES ( 4, '409',2);
SELECT * FROM classrooms;
-----------------------------------------------------------------------------------------

INSERT INTO UNIVERSITY..timetables VALUES ( 1, 600, 100,1);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 600, 110,1);
INSERT INTO UNIVERSITY..timetables VALUES ( 3, 600, 120,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 600, 130,3);

INSERT INTO UNIVERSITY..timetables VALUES ( 1, 610, 100,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 5, 610, 150,3);
INSERT INTO UNIVERSITY..timetables VALUES ( 6, 610, 140,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 7, 610, 130,1);

INSERT INTO UNIVERSITY..timetables VALUES ( 8, 620, 100,1);
INSERT INTO UNIVERSITY..timetables VALUES ( 10, 620, 110,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 9, 620, 120,3);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 620, 130,2);

INSERT INTO UNIVERSITY..timetables VALUES ( 1, 630, 100,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 630, 110,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 3, 630, 120,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 630, 130,2);

INSERT INTO UNIVERSITY..timetables VALUES (10, 640, 150,1);
INSERT INTO UNIVERSITY..timetables VALUES ( 9, 640, 140,1);
INSERT INTO UNIVERSITY..timetables VALUES ( 3, 640, 130,1);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 640, 120,1);

INSERT INTO UNIVERSITY..timetables VALUES ( 8, 650, 110,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 7, 650, 100,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 3, 650, 120,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 650, 130,2);

INSERT INTO UNIVERSITY..timetables VALUES ( 6, 660, 150,3);
INSERT INTO UNIVERSITY..timetables VALUES ( 5, 660, 140,3);
INSERT INTO UNIVERSITY..timetables VALUES ( 3, 660, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 660, 120,1);
```

```
INSERT INTO UNIVERSITY..timetables VALUES ( 5, 670, 100,1);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 670, 110,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 3, 670, 120,3);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 670, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 680, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 690, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 710, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 720, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 730, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 740, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 750, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 760, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 770, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 780, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 790, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 2, 800, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 3, 860, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 870, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 4, 800, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 6, 880, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 7, 890, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 900, 110,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 5, 910, 120,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 8, 920, 120,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 9, 930, 150,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 10, 940, 140,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 950, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 960, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 970, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 980, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 990, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1000, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1010, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1020, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1040, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1050, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1060, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1070, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1080, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1090, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1100, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1110, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1120, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1130, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1140, 130,2);
INSERT INTO UNIVERSITY..timetables VALUES ( 1, 1070, 130,2);


SELECT * FROM timetables;
```

```
INSERT INTO UNIVERSITY..grades VALUES (1,'2017/11/10',1,600,'For Homework'); --grade,date,subject,student
INSERT INTO UNIVERSITY..grades VALUES (3,'2017/12/14',1,600,'Test');
INSERT INTO UNIVERSITY..grades VALUES (2,'2018/02/01',5,610, 'Mid test');
INSERT INTO UNIVERSITY..grades VALUES (3,'2017/12/19',8,620,'Speech');
INSERT INTO UNIVERSITY..grades VALUES (4,'2017/11/10',1,630,'For Homework');
INSERT INTO UNIVERSITY..grades VALUES (5,'2017/12/17',10,640,'For Homework');
INSERT INTO UNIVERSITY..grades VALUES (3,'2018/02/01',4,640, 'Mid test');
INSERT INTO UNIVERSITY..grades VALUES (5,'2017/12/03',4,640, 'For Homework');
INSERT INTO UNIVERSITY..grades VALUES (5,'2018/02/01',3,650, 'Mid test');
INSERT INTO UNIVERSITY..grades VALUES (4,'2018/02/01',3,660, 'Mid test');
INSERT INTO UNIVERSITY..grades VALUES (3,'2018/02/01',3,670, 'Mid test');
INSERT INTO UNIVERSITY..grades VALUES (5,'2018/06/01',5,670, 'For Homework');
INSERT INTO UNIVERSITY..grades VALUES (4,'2018/05/30',3,670, 'Project');

SELECT * FROM grades;
-----------------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..albums VALUES (1,600);
INSERT INTO UNIVERSITY..albums VALUES (2,610);
INSERT INTO UNIVERSITY..albums VALUES (3,620);
INSERT INTO UNIVERSITY..albums VALUES (4,630);
INSERT INTO UNIVERSITY..albums VALUES (5,640);
INSERT INTO UNIVERSITY..albums VALUES (6,650);
INSERT INTO UNIVERSITY..albums VALUES (7,660);
INSERT INTO UNIVERSITY..albums VALUES (8,670);
INSERT INTO UNIVERSITY..albums VALUES (9,680);
INSERT INTO UNIVERSITY..albums VALUES (10,690);
INSERT INTO UNIVERSITY..albums VALUES (11,700);
INSERT INTO UNIVERSITY..albums VALUES (12,710);
INSERT INTO UNIVERSITY..albums VALUES (13,720);
INSERT INTO UNIVERSITY..albums VALUES (14,730);
INSERT INTO UNIVERSITY..albums VALUES (15,740);
INSERT INTO UNIVERSITY..albums VALUES (16,750);
INSERT INTO UNIVERSITY..albums VALUES (17,760);
INSERT INTO UNIVERSITY..albums VALUES (18,770);
INSERT INTO UNIVERSITY..albums VALUES (19,780);
INSERT INTO UNIVERSITY..albums VALUES (20,790);
INSERT INTO UNIVERSITY..albums VALUES (21,800);
INSERT INTO UNIVERSITY..albums VALUES (22,810);
INSERT INTO UNIVERSITY..albums VALUES (23,820);
INSERT INTO UNIVERSITY..albums VALUES (24,830);
INSERT INTO UNIVERSITY..albums VALUES (25,840);
INSERT INTO UNIVERSITY..albums VALUES (26,850);
INSERT INTO UNIVERSITY..albums VALUES (27,860);
INSERT INTO UNIVERSITY..albums VALUES (28,870);
INSERT INTO UNIVERSITY..albums VALUES (29,880);
INSERT INTO UNIVERSITY..albums VALUES (30,890);
INSERT INTO UNIVERSITY..albums VALUES (31,900);
INSERT INTO UNIVERSITY..albums VALUES (32,910);
INSERT INTO UNIVERSITY..albums VALUES (33,920);
INSERT INTO UNIVERSITY..albums VALUES (34,930);
```

```
INSERT INTO UNIVERSITY..albums VALUES (35,940);
INSERT INTO UNIVERSITY..albums VALUES (36,950);
INSERT INTO UNIVERSITY..albums VALUES (37,960);
INSERT INTO UNIVERSITY..albums VALUES (38,970);
INSERT INTO UNIVERSITY..albums VALUES (39,980);
INSERT INTO UNIVERSITY..albums VALUES (40,990);
INSERT INTO UNIVERSITY..albums VALUES (41,1000);
INSERT INTO UNIVERSITY..albums VALUES (42,1010);
INSERT INTO UNIVERSITY..albums VALUES (43,1020);
INSERT INTO UNIVERSITY..albums VALUES (44,1030);
INSERT INTO UNIVERSITY..albums VALUES (45,1040);
INSERT INTO UNIVERSITY..albums VALUES (46,1050);
INSERT INTO UNIVERSITY..albums VALUES (47,1060);
INSERT INTO UNIVERSITY..albums VALUES (48,1070);
INSERT INTO UNIVERSITY..albums VALUES (49,1080);
INSERT INTO UNIVERSITY..albums VALUES (50,1090);
INSERT INTO UNIVERSITY..albums VALUES (51,1100);
INSERT INTO UNIVERSITY..albums VALUES (52,1110);
INSERT INTO UNIVERSITY..albums VALUES (53,1120);
INSERT INTO UNIVERSITY..albums VALUES (54,1130);
INSERT INTO UNIVERSITY..albums VALUES (55,1140);
INSERT INTO UNIVERSITY..albums VALUES (56,1150);
INSERT INTO UNIVERSITY..albums VALUES (57,1160);

SELECT * FROM albums;
-----------------------------------------------------------------------------------------
INSERT INTO UNIVERSITY..final_grades VALUES (1,5,1,1);
INSERT INTO UNIVERSITY..final_grades VALUES (1,4,3,1);
INSERT INTO UNIVERSITY..final_grades VALUES (1,4,2,1);

INSERT INTO UNIVERSITY..final_grades VALUES (2,4,1,1);
INSERT INTO UNIVERSITY..final_grades VALUES (2,5,2,1);

INSERT INTO UNIVERSITY..final_grades VALUES (2,4,3,1);

INSERT INTO UNIVERSITY..final_grades VALUES (3,1,1,1);

SELECT * FROM final_grades;
```

d) Schemat bazy danych



2. Zapytania

a) Zapytanie 1. Odnaleźć imię, nazwisko datę urodzenia, numer telefonu, kierunek studiów oraz nazwę grupy tych studentów, którzy studiują zaocznie fizykę.

```
SELECT DISTINCT students.first_name+' '+ students.last_name  AS 'Student',
students.birth_date AS 'Birth Date',
students.phone_number AS 'Phone Number', group_name AS 'Group',
group_type AS 'Type', field_of_study_name AS 'Field of study'
FROM students, groups, group_types,field_of_study
WHERE students.group_id = groups.group_id
AND group_types.group_id = groups.group_id
AND field_of_study.field_of_study_id = group_types.field_of_study_id
AND group_types.group_type = 'PART TIME'
AND field_of_study.field_of_study_name = 'Physics';
```

Wynik zapytania 1.:

	Student	Birth Date	Phone Num...	Group	Type	Field of study
1	Ed Sheeran	1997-05-20 00:00:00.000	524.177.7217	40	PART TIME	Physics
2	Selena Gomez	1992-07-22 00:00:00.000	523.137.7217	40	PART TIME	Physics

b) Zapytanie 2. Odnaleźć imię, nazwisko, przedmioty z prowadzącymi oraz budynki i klasy, w których się odbywają, dla studenta o nazwisku 'Raskolnikov'.

```
SELECT students. first_name+' '+ students.last_name  AS 'Student',
subject_name AS 'Subject',teachers.first_name +' '+ teachers.last_name AS 'Teacher' ,
classroom_name AS 'Classrooms', building_name AS 'Buildings'
FROM students, timetables, subjects,teachers, classrooms, buildings
WHERE students.student_id = timetables.student_id AND timetables.subject_id =
subjects.subject_id
AND teachers.teacher_id = timetables.teacher_id
AND timetables.classroom_id = classrooms.classroom_id
AND buildings.building_id = classrooms.building_id
AND students.last_name = 'Raskolnikov';
```

Wynik zapytania 2.:

	Student	Subject	Teacher	Classroo...	Buildings
1	Rodion Raskolnikov	POBI	Katniss Everdeen	F.04	B14
2	Rodion Raskolnikov	PROKOMP	Anastasia Grey	F.04	B14
3	Rodion Raskolnikov	TP	Luis Popp	AULA	B14
4	Rodion Raskolnikov	WDI	John Chen	F10	B9

c) Zapytanie 3. Odnaleźć imię, nazwisko prowadzących przedmioty, nazwę przedmiotu wraz z liczbą zapisanych na nie studentów. Wynik uporządkować rosnąco po liczbie studentów.

```
SELECT DISTINCT subject_name AS 'Taught Subject' , teachers.first_name +' '+
teachers.last_name AS 'Main Teacher',
count(students.student_id) AS 'Number Of Students'
FROM students, timetables, subjects, teachers
WHERE students.student_id = timetables.student_id AND timetables.subject_id =
subjects.subject_id
AND teachers.teacher_id = subjects.main_teacher_id
GROUP BY subject_name, teachers.first_name,teachers.last_name
ORDER BY 'Number Of Students' ASC;
```

Wynik zapytania 3.:

	Taught Subject	Main Teacher	Number Of Students
1	CAD	Katniss Everdeen	3
2	EMBEDDED	John Chen	3
3	ENGLISH	Anastasia Grey	3
4	PE	Anastasia Grey	3
5	POBI	Luis Popp	3
6	PROKOMP	Luis Popp	4
7	TP	John Chen	7
8	WDI	Nancy Greenberg	8
9	ZSBD	Daniel Faviet	8
10	SYSOPY	Nancy Greenberg	32

d) Zapytanie 4. Odnaleźć imię, nazwisko, przedmioty i notatki do ocen wraz z datą ich wstawienia dla tych studentów, którzy dostali ocenę cząstkową '5'.

```
SELECT students.student_id, students. first_name+' '+ students.last_name  AS 'Student',
grade_name AS 'Grade', subject_name AS 'Subject', note AS 'Note', grade_date AS 'Date'
FROM students, grades, subjects
WHERE students.student_id = grades.student_id
AND subjects.subject_id=grades.subject_id
AND grade_name='5';
```

Wynik zapytania 4.:

	student...	Student	Grade	Subject	Note	Date
1	640	Lilly Collins	5	CAD	For Homework	2017-12-17 00:00:00.000
2	640	Lilly Collins	5	WDI	For Homework	2017-12-03 00:00:00.000
3	650	Nastya Kasnikov	5	TP	Mid test	2018-02-01 00:00:00.000
4	670	Harry Potter	5	PROKOMP	For Homework	2018-06-01 00:00:00.000

e) Zapytanie 5. Wyznaczyć średnią ocen w przedmiocie dla studentów na podstawie ich ocen cząstkowych.

```
SELECT students. first_name+' '+ students.last_name  AS 'Student',
 ROUND(AVG(CAST(grades.grade_name AS FLOAT)), 2)  AS 'Average Grade', subject_name AS
'Subject'
FROM students, grades, subjects
WHERE students.student_id = grades.student_id
AND subjects.subject_id=grades.subject_id
GROUP BY subject_name, students. first_name, students.last_name, students.student_id
ORDER BY students.last_name , students.student_id , subject_name;
```

Wynik zapytania 5.:

	Student	Average Grade	Subject
1	Hermann Baer	2	PROKOMP
2	Lilly Collins	5	CAD
3	Lilly Collins	4	WDI
4	William Gietz	4	SYSOPY
5	Shelley Higgins	3	PE
6	Nastya Kasnikov	5	TP
7	Susan Mavris	2	SYSOPY
8	Harry Potter	5	PROKOMP
9	Harry Potter	3,5	TP
10	Rodion Raskolnikov	4	TP

f) Zapytanie 6. Odnaleźć imię i nazwisko studenta z najwyższą średnią ocen na roku na podstawie ocen końcowych.

```
SELECT TOP 1 students.student_id AS 'Id', students. first_name+' '+ students.last_name  AS
'Student',
 ROUND(AVG(CAST(final_grades.final_grade AS FLOAT)), 2)  AS 'Student Average'
FROM students, albums,final_grades
WHERE students.student_id = albums.student_id AND  albums.album_id
=final_grades.album_id
GROUP BY students. first_name, students.last_name, students.student_id
ORDER BY avg(final_grades.final_grade) DESC;
```

Wynik zapytania 6:

	Id	Student	Student Average
1	680	Peter Parker	5

g) Zapytanie 7. Wyznaczyć średnią ocen dla przedmiotu na podstawie ocen końcowych. Wynik posortować malejąco.

```
SELECT subject_name AS 'Subject', ROUND(AVG(CAST(final_grades.final_grade AS FLOAT)), 2)
AS 'Subject Average Grade'
FROM students, albums,final_grades, subjects
WHERE students.student_id = albums.student_id  AND  albums.album_id
=final_grades.album_id
AND subjects.subject_id=final_grades.subject_id
GROUP BY subject_name
ORDER BY 'Subject Average Grade' DESC;
```

Wynik zapytania 7:

	Subject	Subject Average Grade
1	SYSOPY	4,67
2	ZSBD	4,5
3	TP	4

h) Zapytanie 8. Wyznaczyć średnią ocen dla każdej z grup w przedmiocie 'Sysopy' na podstawie ocen końcowych. Wynik posortować malejąco.

```
SELECT subject_name AS 'Subject',groups.group_id AS 'Group',
ROUND(AVG(CAST(final_grades.final_grade AS FLOAT)), 2) AS 'Subject Average Grade'
FROM students, albums,final_grades, subjects, groups
WHERE students.student_id = albums.student_id  AND  albums.album_id
=final_grades.album_id
AND subjects.subject_id=final_grades.subject_id AND students.group_id = groups.group_id
AND subject_name = 'Sysopy'
GROUP BY groups.group_id, subject_name
ORDER BY 'Subject Average Grade' DESC;
```

Wynik zapytania 8:

	Subject	Group	Subject Average Grade
1	SYSOPY	11	5
2	SYSOPY	10	4,5

i) Zapytanie 9. Wyznaczyć liczbę studentów w każdej z grup dziekańskich.

```
SELECT group_name AS 'Group', count(student_id) AS 'Number of students'
FROM students, groups , group_types
WHERE students.group_id = groups.group_id
AND group_types.group_id = groups.group_id
GROUP BY  group_types. group_name, groups.group_id
ORDER BY groups.group_id ;
```

Wynik zapytania 9:

	Group	Number of students
1	10	5
2	11	4
3	12	4
4	13	5
5	14	7
6	15	5
7	16	4
8	17	5
9	18	4
10	20	6
11	30	6
12	40	2

j) Zapytanie 10. Odnaleźć imię i nazwisko, nr telefonu, email, datę zatrudnienia i pensję wykładowców zatrudnionych po '1995/01/01'.

```
SELECT first_name +' '+ last_name AS 'Teacher' , phone_number AS 'Phone Number' ,
email AS 'Email', hire_date AS 'Hire Date' , salary AS 'Salary'
FROM teachers
WHERE hire_date > '1995/01/01'
ORDER BY first_name, last_name;
```

Wynik zapytania 10:

	Teacher	Phone Number	Email	Hire Date	Salary
1	Anastasia Grey	515.124.3213	AGREY	1999-01-17 00:00:00.000	2250,00
2	John Chen	515.124.4269	JCHEN	1997-09-28 00:00:00.000	2100,00
3	Luis Popp	515.124.4567	LPOPP	1999-12-07 00:00:00.000	2500,00

k) Zapytanie 11. Odnaleźć najwcześniej zatrudnionego wykładowcę.

```
SELECT first_name +' '+ last_name AS 'First Hired Teacher' , hire_date AS 'Hire Date'
FROM teachers
WHERE hire_date= (SELECT min (hire_date)FROM teachers );
```

Wynik zapytania 11:

	First Hired Teacher	Hire Date
1	Katniss Everdeen	1989-02-12 00:00:00.000

l) Zapytanie 12. Odnaleźć najpóźniej zatrudnionego wykładowcę.

```
SELECT first_name +' '+ last_name AS 'Last Hired Teacher' , hire_date AS 'Hire Date'

FROM teachers

WHERE hire_date= (SELECT max(hire_date)FROM teachers );
```

Wynik zapytania 12:

	Last Hired Teacher	Hire Date
1	Luis Popp	1999-12-07 00:00:00.000

m) Zapytanie 13. Odnaleźć studentów niezapisanych na żaden przedmiot.

```
SELECT DISTINCT students.student_id AS 'Student id', students.first_name+' '+ students.last_name  AS
'Student'
FROM students
EXCEPT
        (
                SELECT DISTINCT students.student_id,students.first_name+' '+ students.last_name  AS
'Student'
                FROM students, timetables
                WHERE students.student_id = timetables.student_id
        )
```

Wynik zapytania 13:

	Student id	Student
1	700	Bronagh Murphy
2	810	Tyrion Lannister
3	820	Cersei Lannister
4	830	Sansa Stark
5	840	Khal Drogo
6	850	Joffrey Baratheon
7	1030	Sherlock Holmes
8	1150	Ed Sheeran
9	1160	Selena Gomez

n) Zapytanie 14. Odnaleźć oceny końcowe studenta o nazwisku 'Baer'.

```
SELECT  students. first_name+' '+ students.last_name  AS 'Student',subject_name AS
'Subject',
final_grades.final_grade AS 'Final Grade'
FROM students, albums,final_grades, subjects
WHERE students.student_id = albums.student_id  AND  albums.album_id
=final_grades.album_id
AND subjects.subject_id=final_grades.subject_id
AND students.last_name = 'Baer'
GROUP BY subject_name,students. first_name,students.last_name , final_grades.final_grade
ORDER BY 'Final Grade' DESC;
```

Wynik zapytania 14:

	Student	Subject	Final Grade
1	Hermann Baer	ZSBD	5
2	Hermann Baer	SYSOPY	4
3	Hermann Baer	TP	4

o) Zapytanie 15. Odnaleźć imię, nazwisko prowadzących przedmioty wraz z liczbą prowadzonych przez nich przedmiotów.

```
SELECT  teachers.first_name +' '+ teachers.last_name AS 'Teacher',
count(DISTINCT timetables.subject_id) AS 'Number Of Subjects'
FROM timetables, subjects, teachers
WHERE teachers.teacher_id = timetables.teacher_id AND timetables.subject_id =
subjects.subject_id
GROUP BY  teachers.first_name,teachers.last_name
ORDER BY 'Number Of Subjects' ASC;
```

Wynik zapytania 15:

	Teacher	Number Of Subjects
1	Katniss Everdeen	4
2	Daniel Faviet	4
3	Nancy Greenberg	4
4	Anastasia Grey	4
5	John Chen	5
6	Luis Popp	6

p) Zapytanie 16. Odnaleźć wykładowców, którzy nie uczą żadnych przedmiotów.

```
SELECT DISTINCT teachers.teacher_id AS 'Teachers id', teachers.first_name+' '+ teachers.last_name   AS
'Teachers'
FROM teachers
EXCEPT
        (
                SELECT DISTINCT teachers.teacher_id,teachers.first_name+' '+ teachers.last_name
                FROM teachers, timetables
                WHERE teachers.teacher_id = timetables.teacher_id
        )
```

Wynik zapytania 16:

	Teachers id	Teachers
1	160	Nikogo Nieucze

r) Zapytanie 17. Wypisać z użyciem kursora nazwiska wszystkich studentów z grupy '20' oraz podać liczbę urodzonych przed rokiem

```
DECLARE @imię varchar(40), @nazwisko varchar(40), @student_id int, @group_id int, @num int, @licz int , @birth_date
datetime;
set @num=20
set @licz=0
DECLARE kursor_studenci_w_grupie CURSOR FOR
SELECT first_name, last_name, student_id, group_id, birth_date FROM students;
OPEN kursor_studenci_w_grupie
    FETCH NEXT FROM kursor_studenci_w_grupie
    INTO @imię, @nazwisko ,@student_id,@group_id,@birth_date

    WHILE @@FETCH_STATUS = 0
    BEGIN
        IF (@group_id = @num)
                        BEGIN
        PRINT 'Id: ' + CAST(@student_id AS VARCHAR) + ' Imię: ' + CAST(@imię AS VARCHAR) + '  Nazwisko: ' + @nazwisko
                                + '  Urodziny: ' + CAST(@birth_date AS VARCHAR)
                            IF (@birth_date < '1997/01/01')
                            BEGIN
                            set @licz = @licz + 1
                            END
                        END
        FETCH NEXT FROM kursor_studenci_w_grupie
        INTO @imię, @nazwisko ,@student_id,@group_id,@birth_date
                        END
                        PRINT 'Liczba osob w grupie urodzonych przed 1997: ' + CAST(@licz AS VARCHAR)

CLOSE kursor_studenci_w_grupie;
DEALLOCATE kursor_studenci_w_grupie;
```

Wynik zapytania 17:

```
Id: 1030  Imię: Sherlock   Nazwisko: Holmes   Urodziny: Apr 30 1998 12:00AM
Id: 1040  Imię: Oliver   Nazwisko: Queen   Urodziny: Apr 30 1997 12:00AM
Id: 1050  Imię: Felicity   Nazwisko: Smoak   Urodziny: Apr 30 1999 12:00AM
Id: 1060  Imię: Roy   Nazwisko: Harper   Urodziny: Apr 30 1996 12:00AM
Id: 1070  Imię: Thea   Nazwisko: Queen   Urodziny: Apr 30 1996 12:00AM
Id: 1080  Imię: John   Nazwisko: Diggle   Urodziny: Apr 30 1998 12:00AM
Liczba osob urodzonych przed 1997: 2
```

3. Funkcje

a) Funkcja podająca dla każdej grupy, ile procent wszystkich studentów znajduje się w tej grupie.

```
USE UNIVERSITY;
IF OBJECT_ID ('group_perc') IS NOT NULL
drop function group_perc;
go
create function group_perc (@group_id varchar(15)) returns float
as
begin
       declare @percentage float
       declare @in_group float
       declare @all float
       select @all = count(*) from students
       select @in_group = count (*) from students as s
                                          where group_id in (select g.group_id from
groups as g
                                                                                    where
g.group_id = @group_id)

set @percentage = 100 * @in_group / @all
return @percentage
end
go
```

Przykładowe wywołanie nr 1:

```
select distinct gt.group_name AS 'Group',cast(round(dbo.group_perc(g.group_id), 1) as
varchar)

+ '%' as 'Percentage' from groups g, group_types gt where g.group_id = gt.group_id AND
g.group_id =10 group by gt.group_name, g.group_id;
```

Wynik wywołania:

	Group	Percentage
1	10	8.6%

Przykładowe wywołanie nr 2:

```
select distinct gt.group_name AS 'Group',cast(round(dbo.group_perc(g.group_id), 1)
as varchar)

+ '%' as 'Percentage' from groups g, group_types gt where g.group_id = gt.group_id
```

Wynik wywołania:

	Group	Percentage
1	10	8.6%
2	11	6.9%
3	12	6.9%
4	13	10.3%
5	14	12.1%
6	15	8.6%
7	16	6.9%
8	17	8.6%
9	18	6.9%
10	20	10.3%
11	30	10.3%
12	40	3.4%

4. Procedury

a) Procedura ADD_STUDENT przypisuje losowo wybrany numer przy dodawaniu nowego studenta.

```
if exists(select 1 from sys.objects where TYPE='P' and name = 'ADD_STUDENT')
drop procedure ADD_STUDENT;
go
create procedure ADD_STUDENT
	@first_name varchar(25),
		@last_name varchar(25),
		@phone_number varchar(15),
		@birth_date DATETIME,
		@group_id int
as
begin
	declare @student_id varchar(4) = 1;
		declare @numer int, @flaga int = 1;
		while(@flaga = 1)
		begin
			SELECT @numer = (600+ CONVERT(INT, (700 - 600 +1) * RAND()))
			set @flaga = 0

			DECLARE kursor CURSOR FOR
			SELECT @student_id FROM students
			OPEN kursor
			FETCH NEXT FROM kursor INTO @student_id
			WHILE @@FETCH_STATUS = 0
			  BEGIN
				if(cast(@numer as varchar) = @student_id)
				begin
					set @flaga = 1;
					break;
				end
				FETCH NEXT FROM kursor INTO @student_id
			  END
			CLOSE kursor
			DEALLOCATE kursor

			if(@flaga = 0)
				set @student_id = cast(@numer as varchar);
		end

	insert into students values(@student_id, @first_name, @last_name, @phone_number,
@birth_date, @group_id);
end
```

Przykładowe wywołanie:

```
begin
        exec UNIVERSITY..ADD_STUDENT 'John', 'Green', '0-22-111-7532', '03-FEB-1998', '13'
end
```

Losowo dobrane student_id dla wprowadzonego studenta:

	student_id	first_name	last_name	phone_number	birth_date	group_id
1	696	John	Green	0-22-111-7532	1998-02-03 00:00:00.000	13

b) Procedura MODIFY_SALARY wprowadza podwyżkę pensji o dany procent wykładowcy o podanym id.

```
if exists(select 1 from sys.objects where TYPE='P' and name = 'MODIFY_SALARY')
drop procedure MODIFY_SALARY
go
create procedure MODIFY_SALARY
      @teacher_id  int = 100,
                  @percentage int = 10
as
begin
          if(@teacher_id IN (select teacher_id  from teachers))
          begin
                    update teachers
           set salary += salary *(@percentage)/100
                                    where teacher_id = @teacher_id
                                    print ( 'Wprowadzono podwyzke o ' + cast(@percentage as
                                    varchar) + ' procent' + ' dla wykładowcy' + cast(@teacher_id as
                                    varchar))
          end
          else
          print('Nieprawidlowe id nauczyciela')

end
go
```

Przykładowe wywołania:

```
begin
        exec UNIVERSITY..MODIFY_SALARY
        exec UNIVERSITY..MODIFY_SALARY 150, 50
end
```

Wynik wywołania:

```
(1 row(s) affected)
Wprowadzono podwyzke o 10 procent dla wykładowcy 100

(1 row(s) affected)
Wprowadzono podwyzke o 50 procent dla wykładowcy 150
```

c) Procedura DELETE_TEACHER usuwa wykładowców, którzy nie uczą żadnych przedmiotów. Niemożliwe jest usunięcie wykładowcy, który uczy jakiegoś przedmiotu.

```
if exists(select 1 from sys.objects where TYPE='P' and name = 'DELETE_TEACHER')
drop procedure DELETE_TEACHER;
go
create procedure DELETE_TEACHER
@teacher_id int = 530
as
begin
                if(@teacher_id IN (select teacher_id from teachers)
                AND @teacher_id NOT IN
                                (
                                        SELECT DISTINCT teachers.teacher_id
                                         FROM timetables, teachers
                                        WHERE teachers.teacher_id = timetables.teacher_id )
                                )
                begin
                delete from teachers where teacher_id = @teacher_id
                print('Teacher '+ cast(@teacher_id as varchar) +' was deleted')
                end
                else print('Teacher '+ cast(@teacher_id as varchar) +' cannot be deleted')
end
```

Przykładowe wywołania:

```
begin
        exec UNIVERSITY..DELETE_TEACHER 160
        exec UNIVERSITY..DELETE_TEACHER 150
end
```

Wynik wywołania:

```
(1 row(s) affected)
Teacher 160 was deleted
Teacher 150 cannot be deleted
```

d) Procedura ADD_ TEACHER przypisuje losowo wybrany numer przy dodawaniu nowego wykładowcy.

```
if exists(select 1 from sys.objects where TYPE='P' and name = 'ADD_TEACHER')
drop procedure ADD_TEACHER
go
create procedure ADD_TEACHER
      @first_name varchar(25), @last_name varchar(25),@email  varchar(25),@phone_number varchar(15),
     @hire_date DATETIME, @salary money
as
begin
          declare @teacher_id int = 1;
                    declare @numer int, @flaga int = 1;
                    while(@flaga = 1)
                    begin
                              SELECT @numer = (100+ CONVERT(INT, (200 - 100 +1) * RAND()))
                              set @flaga = 0
                              DECLARE kursor CURSOR FOR
                              SELECT teacher_id FROM teachers
                              OPEN kursor
                              FETCH NEXT FROM kursor INTO @teacher_id
                              WHILE @@FETCH_STATUS = 0
                                BEGIN
                                        if(cast(@numer as varchar) = @teacher_id)
                                        begin
                                                  set @flaga = 1;
                                                  break;
                                        end
                                        FETCH NEXT FROM kursor INTO @teacher_id
                                END
                              CLOSE kursor
                              DEALLOCATE kursor
                              if(@flaga = 0)
                                        set @teacher_id = cast(@numer as varchar)
                    end

insert into teachers values(@teacher_id, @first_name, @last_name, @email, @phone_number, @hire_date, @salary);
end
```

Przykładowe wywołania:

```
begin
        exec UNIVERSITY..ADD_TEACHER 'Tomasz', 'Makowski', 'TMAK', '0-98-457 7532', '22-FEB-2018', '2000'
end
```

Losowo dobrane teacher_id dla wprowadzonego wykładowcy:

	teacher_id	first_name	last_name	email	phone_number	hire_date	salary
1	183	Tomasz	Makowski	TMAK	0-98-457 7532	2018-02-22 00:00:00.000	2000,00

5. Wyzwalacze

a) Wyzwalacz GRADE_ADD_CHECK uniemożliwia wprowadzenie oceny cząstkowej z przedmiotu, na który dany student nie chodzi.

```
IF OBJECT_ID ('GRADE_ADD_CHECK' , 'TR') IS NOT NULL
  DROP trigger GRADE_ADD_CHECK ;
go
create trigger GRADE_ADD_CHECK
 on grades instead of insert
as
begin
        declare @student_id int = (select student_id from inserted);
        declare @subject_id int = (select subject_id from inserted);
        declare @sub_id int = (SELECT  t.subject_id FROM students s, timetables t
                                                WHERE s.student_id = t.student_id AND @student_id =
t.student_id AND @subject_id = t.subject_id)
        declare @note VARCHAR(30) = (select note from inserted),
                        @grade int = (select grade_name from inserted),
                @date datetime = (select grade_date from inserted),
                        @name VARCHAR(45) = (select first_name + ' ' + last_name from students where
student_id = @student_id),
                        @subject_name VARCHAR(30) = (select subject_name from subjects where subject_id
= @subject_id)

        if(@subject_id = @sub_id)
                begin
                INSERT INTO UNIVERSITY..grades VALUES (@grade,@date,@subject_id,@student_id,@note)
                print 'Zapisano ' + 'Student ' + @name + ' na przedmiot: ' + @subject_name + '.';
                end
                else
                print 'Student ' + @name + ' nie jest zapisany na przedmiot: ' + @subject_name + '.'
end
```

Próba wprowadzenia ocen dla 2 studentów niezapisanych na dany przedmiot i 1 zapisanego:

```
INSERT INTO UNIVERSITY..grades VALUES (2,'2018/02/01',1,640,'Test'); -- nie jest zapisany
 INSERT INTO UNIVERSITY..grades VALUES (5,'2018/02/01',1,660,'Test'); -- nie jest zapisany
 INSERT INTO UNIVERSITY..grades VALUES (5,'2018/02/01',3,640,'Project');  -- jest zapisany
```

Komunikat:

```
Student Lilly Collins nie jest zapisany na przedmiot: SYSOPY.

(1 row(s) affected)
Student Rodion Raskolnikov nie jest zapisany na przedmiot: SYSOPY.

(1 row(s) affected)

(1 row(s) affected)
Zapisano Student Lilly Collins na przedmiot: TP.

(1 row(s) affected)
```

b) Wyzwalacz TEACHER_HIRE_DATE, który po wstawieniu wykładowcy bez daty zatrudnienia, wstawia aktualną.

```
-- IF OBJECT_ID (TEACHER_HIRE_DATE , 'TR') IS NOT NULL
-- DROP trigger TEACHER_HIRE_DATE ;
go
create trigger TEACHER_HIRE_DATE
on teachers
after insert
as
declare @teacher_id int
set @teacher_id = (select teacher_id from inserted)
if (select teachers.hire_date from teachers where teachers.teacher_id =
@teacher_id) is null
begin
update teachers
set hire_date = GETDATE()
where teachers.teacher_id = @teacher_id
end
```

Wstawienie wykładowcy bez pola "hire_date":

```
INSERT INTO UNIVERSITY..teachers VALUES ( 200, 'Niemam', 'Daty', '','515.124.4567', NULL, 2670);
```

Zaktualizowana informacja w polu hire_date:

	teacher_id	first_name	last_na...	email	phone_number	hire_date	salary
1	200	Niemam	Daty		515.124.4567	2018-06-01 20:26:21.820	2670,00

c) Wyzwalacz ENTER_FINAL_GRADES, który po dodaniu do tabeli academic_year nowego roku akademickiego wprowadza 'final grade' jako zaokrągloną średnią ocen cząstkowych.

```
IF OBJECT_ID ('ENTER_FINAL_GRADES' , 'TR') IS NOT NULL
 DROP trigger ENTER_FINAL_GRADES ;
go
CREATE TRIGGER ENTER_FINAL_GRADES
ON academic_year
INSTEAD OF INSERT
AS
BEGIN --start kursor_studenci
	DECLARE  @student_id int
	DECLARE kursor_studenci CURSOR FOR
	SELECT student_id FROM students;
	OPEN kursor_studenci
		FETCH NEXT FROM kursor_studenci
		INTO @student_id
		WHILE @@FETCH_STATUS = 0
		BEGIN --start kursor_przedmioty
			DECLARE @subject_id int
			DECLARE kursor_przedmioty CURSOR FOR
			SELECT subject_id FROM subjects;
			OPEN kursor_przedmioty
				FETCH NEXT FROM kursor_przedmioty
				INTO @subject_id
				WHILE @@FETCH_STATUS = 0
				BEGIN
					IF EXISTS (SELECT student_id FROM grades WHERE @student_id = student_id AND
							@subject_id = subject_id)
					BEGIN
						DECLARE @final_grade int,
						@album_id int = (SELECT album_id FROM albums WHERE student_id =
						@student_id),
						@academic_year int = (SELECT TOP 1 academic_year_id FROM academic_year
						ORDER BY academic_year_id DESC);

						SELECT @final_grade = ROUND(AVG(grade_name),0) FROM grades
							WHERE @student_id = student_id AND @subject_id = subject_id

						INSERT INTO final_grades VALUES (@album_id, @final_grade,
						@subject_id,@academic_year);
					END

					FETCH NEXT FROM kursor_przedmioty
					INTO @subject_id
				END
			CLOSE kursor_przedmioty;
```

```
                    DEALLOCATE kursor_przedmioty; --koniec kursor_przedmioty

                    FETCH NEXT FROM kursor_studenci
                    INTO @student_id
                END
        CLOSE kursor_studenci;
        DEALLOCATE kursor_studenci; --koniec kursor_studenci

        DECLARE @academic_year_id INT = (SELECT academic_year_id FROM inserted),
                    @academic_year_name DATETIME = (SELECT academic_year_name FROM inserted);
        INSERT INTO academic_year VALUES(@academic_year_id, @academic_year_name);
END
go
        INSERT INTO academic_year VALUES(2, '2018/10/15');
        SELECT* from final_grades
```

Tabela z ocenami końcowym 'final_grade' przed dodaniem do tabeli academic_year nowego roku akademickiego.

	album_id	final_grade	subject_id	academic_year
1	1	5	1	1
2	1	4	3	1
3	1	4	2	1
4	2	4	1	1
5	2	5	2	1
6	2	4	3	1
7	3	1	1	1

Tabela z ocenami końcowym 'final_grade' przed dodaniem do tabeli academic_year nowego roku akademickiego.

	album_id	final_grade	subject_id	academic_year
1	1	5	1	1
2	1	4	3	1
3	1	4	2	1
4	2	4	1	1
5	2	5	2	1
6	2	4	3	1
7	3	1	1	1
8	1	2	1	1
9	2	2	5	1
10	3	3	8	1
11	4	4	1	1
12	5	4	4	1
13	5	5	10	1
14	6	5	3	1
15	7	4	3	1
16	8	3	3	1
17	8	5	5	1

d) Wyzwalacz DELETE_STUDENT, który przy próbie usunięcia studenta w pierwszej kolejności wypisuje go ze wszystkich przedmiotów, na które jest zapisany, a następnie trwale usuwa go z listy studentów.

```
SELECT* from students WHERE student_id = 600
IF OBJECT_ID ('DELETE_STUDENT' , 'TR') IS NOT NULL
 DROP trigger DELETE_STUDENT ;
go
CREATE TRIGGER DELETE_STUDENT
ON students
INSTEAD OF DELETE
AS
BEGIN

	DECLARE @student_id int = (SELECT student_id FROM deleted),
			@ilosc int,
			@first_name VARCHAR(20) = (SELECT first_name FROM deleted),
			@last_name VARCHAR(25) = (SELECT last_name FROM deleted)

	SET @ilosc = (SELECT count(subject_id)
	FROM timetables		WHERE student_id = @student_id)

	DELETE FROM timetables
	WHERE student_id = @student_id

	DELETE FROM final_grades
	WHERE album_id = (SELECT album_id FROM albums WHERE student_id = @student_id)

	DELETE FROM grades
	WHERE student_id = @student_id

	DELETE FROM albums
	WHERE student_id = @student_id

	DELETE FROM students
	WHERE student_id = @student_id

	print 'Studenta ' + @first_name + ' ' + @last_name + ' wypisano z ' + cast(@ilosc as VARCHAR) + ' przedmiotów, a
nastepnie usunięto z listy studentów.'

END
go
	DELETE FROM students
	WHERE student_id = 600;

	SELECT* from students WHERE student_id = 600
```

Wynik zapytania o dane studenta przed jego usunięciem.

	student_id	first_name	last_name	phone_number	birth_date	group_id
1	600	Susan	Mavris	516.124.4567	1996-12-13 00:00:00.000	10

Wynik zapytania o dane studenta po jego usunięciu.

	student...	first_na...	last_na...	phone_num...	birth_d...	group...

e) Wyzwalacz UPDATE_GRADE, który po każdej modyfikacji w tabeli "grades" wprowadza aktualną datą modyfikacji do pola "grade_date".

```
SELECT* from grades WHERE grade_id = 5

IF OBJECT_ID ('UPDATE_GRADE' , 'TR') IS NOT NULL
 DROP trigger UPDATE_GRADE ;
go
CREATE TRIGGER UPDATE_GRADE
ON grades
AFTER UPDATE
AS
BEGIN
	DECLARE @grade_id int = (SELECT grade_id FROM inserted)

	UPDATE grades
	SET grade_date = GETDATE()
	WHERE @grade_id = grade_id;
END
go
	UPDATE grades
	SET grade_name = 1
	WHERE grade_id = 5;

	SELECT* from grades WHERE grade_id = 5
```

Tabela 'grades' przed modyfikacją.

	grade_id	grade_name	grade_date	subject_id	student_id	note
1	5	4	2017-11-10 00:00:00.000	1	630	For Homework

Tabela 'grades' po modyfikacji, z zaktualizowaną datą modyfikacji.

	grade_id	grade_name	grade_date	subject_id	student...	note
1	5	1	2018-06-05 20:39:43.450	1	630	For Homework