ПИ19-4 Деменчук Георгий КР

1. Создайте отображение базы данных студентов *Students_2021.sqlite*

In [203]:
```
from sqlalchemy.ext.automap import automap_base
from sqlalchemy.orm import Session
from sqlalchemy import create_engine
from datetime import datetime
engine = create_engine("sqlite:///Students_2021.sqlite")
Base = automap_base()
Base.prepare(engine, reflect=True)
session = Session(engine)
```

2. Напишите запрос, позволяющий получить из таблицы *exam_marks* значения столбца *mark* (экзаменационная оценка) для всех студентов, исключив из списка повторение одинаковых строк.

In [204]:
```
city,exam_marks,student,university,subject,lecturer,subj_lect = Base.classes.values()
```

In [205]:
```
from sqlalchemy.orm import Session
session = Session(engine)
```

In [206]:
```
for item in session.query(exam_marks.mark).distinct().all():
    print(item.mark)
```

```
1
4
3
5
2
None
```

3. Напишите запрос для получения списка студентов без определенного места жительства.

In [207]:
```
for item in session.query(student.name, student.surname).where(student.city == None).all():
    print(item)
```

```
('Ирина', 'Медведева')
('Оксана', 'Афанасьева')
```

4. Напишите запрос для получения списка студентов, проживающих в Воронеже и не получающих стипендию.

In [208]:
```
from sqlalchemy import and_
for item in session.query(student.name, student.surname, student.city, student.stipend).where(
    and_(
        student.city == "Воронеж",
        student.stipend == 0
    )
).all():
    print(item)
```

```
('Андрей', 'Павлов', 'Воронеж', 0)
```

5. Напишите запрос для получения списка университетов, расположенных в Москве и имеющих рейтинг меньший,

чем у НГУ. Значение рейтинга НГУ получите с помощью отдельного запроса или подзапроса.

In [209]:
```
from sqlalchemy import and_
subquery = session.query(university.rating).where(university.univ_name=="НГУ").subquery()
query1 = session.query(university.univ_name).where(
    and_(
        university.city == "Москва",
        university.rating < (subquery)
    )
)
for item in query1.all():
    print(item)
```

```
('ФмиУ',)
('МТУСИ',)
```

6. Напишите запрос, выполняющий вывод находящихся в таблице EXAM_MARKS

номеров предметов обучения, экзамены по которым сдавались между 1 и 21 марта 2020 г.

In [210]:
```
from datetime import datetime
first_date = datetime.strptime('2020-03-1', '%Y-%m-%d')
second_date = datetime.strptime('2020-03-21', '%Y-%m-%d')

for item in session.query(exam_marks.subj_id, exam_marks.mark, exam_marks.exam_date).where(
    exam_marks.exam_date.between(first_date, second_date)
).all():
    print(item)
```

```
(73, 4, datetime.datetime(2020, 3, 20, 0, 0))
(12, 1, datetime.datetime(2020, 3, 13, 0, 0))
(10, 3, datetime.datetime(2020, 3, 21, 0, 0))
```

7. Напишите запрос, который выполняет вывод названий предметов обучения, начинающихся на букву 'И'.

In [211]:
```
for item in session.query(subject.subj_id, subject.subj_name, subject.hour).where(
    subject.subj_name.like("И%")
).all():
    print(item)
```

```
(10, 'Информатика', 56)
(56, 'История', 34)
```

8. Напишите запрос, выбирающий сведения о студентах, у которых имена начинаются на букву 'И' или 'С'.

In [212]:
```
from sqlalchemy import or_
for item in session.query(student.name, student.surname).where(
    or_(
        student.name.like("И%"),
        student.name.like("С%")
    )
).all():
    print(item)
```

```
('Иван', 'Иванов')
('Ия', 'Соколова')
('Ирина', 'Медведева')
('Ирина', 'Сорокина')
('Ирина', 'Жданова')
('Иранда', 'Блохина')
('Ирон', 'Назаров')
('Иннокентий', 'Колобов')
('Ипат', 'Комаров')
```

9. Напишите запрос для получения списка предметов обучения, названия которых состоят из более одного слова.

In [213]:
```
for item in session.query(subject.subj_id, subject.subj_name, subject.hour).where(
    subject.subj_name.like("% %")
).all():
    print(item)
```

```
(12, 'Анализ данных', 42)
```

10. Напишите запрос для получения списка студентов, фамилии которых состоят из трех букв.

In [214]:
```
from sqlalchemy.sql.expression import func
for item in session.query(student.name, student.surname).where(
    func.length(student.surname) == 3
).all():
    print(item)
```

```
('Бернар', 'Шоу')
('Джон', 'Доу')
```

11. Составьте запрос для таблицы STUDENT таким образом, чтобы получить результат в следующем виде. Распечатайте первые 9 записей результата.

```
И. Иванов    1982-12-03
П. Петров    1980-12-01
В. Сидоров   1979-06-07
...
```

In [215]:
```
from sqlalchemy.sql.expression import func
for item in session.query(

    cast(func.substr(student.name,1,1), String(20))+"."+
        cast(student.surname, String(20))+"     "+
        cast(func.strftime('%Y-%m-%d',student.birthday),  String(20))
).all():
    print(item)
```

```
('И.Иванов    1982-12-03',)
('П.Петров    1980-12-01',)
('В.Сидоров   1979-06-07',)
('Б.Кузнецов    1981-12-08',)
('О.Зайцева   1981-05-01',)
('П.Котов    2021-02-28',)
('В.Белкин   1980-01-07',)
('Е.Сергеева    1997-07-04',)
('В.Кудряшова    2002-02-18',)
('В.Журавлева    1993-06-14',)
('С.Дементьева    1996-09-25',)
('П.Рожкова   1995-09-15',)
('И.Соколова   1995-06-25',)
('В.Семенова   1997-10-04',)
('И.Медведева    2000-08-22',)
('О.Афанасьева    2000-11-16',)
('Ф.Сергеева    2000-02-19',)
('В.Некрасова    1998-03-23',)
('О.Игнатьева    2000-03-08',)
('О.Казакова    2000-02-03',)
('Е.Шубина   1996-08-24',)
('А.Миронова    2000-10-08',)
('С.Кондрат    1995-02-23',)
('А.Пономарева    1995-10-07',)
('Л.Одинцова    1995-03-07',)
('Р.Гуляева    1999-07-23',)
('З.Игнатова    2000-09-26',)
('Д.Мельникова    1993-08-01',)
('С.Мишина    1997-05-22',)
('В.Агафонова    1994-10-29',)
('А.Афанасьева    1996-12-22',)
('В.Фролова    1999-06-02',)
('Н.Лихачева    1997-04-30',)
('Э.Лапина    1997-11-23',)
('И.Сорокина    1993-09-28',)
('Г.Пашкова    2001-08-15',)
('М.Казакова    1994-02-18',)
('Ф.Жукова   1998-10-06',)
('В.Зимина    1997-06-25',)
('И.Жданова    2002-03-02',)
('О.Валерия    2000-06-15',)
('К.Костина    1995-07-15',)
('И.Блохина    1997-03-23',)
('С.Мельникова    1998-03-24',)
('А.Щербакова    1995-01-24',)
('О.Васильева    1994-05-14',)
('В.Сорокина    1995-03-01',)
('Ф.Тимофеева    1993-08-01',)
('А.Вишнякова    2002-02-10',)
('Ф.Суравьева    1995-06-16',)
('М.Борисова    1995-08-07',)
('С.Никитин    1997-07-15',)
('Е.Калашников    1998-02-27',)
('К.Петров    1998-03-07',)
('В.Трофимов    1995-03-24',)
('Г.Бирюков    1999-10-16',)
('А.Антонов    1997-04-08',)
('В.Медведев    1995-04-27',)
('К.Морозов    1996-09-29',)
('И.Назаров    2001-12-15',)
('П.Дмитриев    1993-12-10',)
('К.Суханов    1996-11-22',)
('Л.Карпов    2001-06-27',)
('С.Захаров    1994-03-19',)
('А.Аксенов    1994-10-27',)
('А.Панов    1996-12-23',)
('И.Колобов    1993-07-23',)
('Г.Федоров    2000-11-16',)
('С.Лазарев    1998-05-22',)
('Н.Кондратьев    1999-10-28',)
('А.Бирюков    1994-10-15',)
('Р.Горский    1998-11-03',)
('А.Воронова    1999-02-04',)
('Ф.Гордеев    1997-05-16',)
('Т.Веселов    1994-04-28',)
('С.Юшин    2001-12-02',)
('В.Пономарев    2000-12-15',)
('П.Жуков    1993-09-11',)
('П.Кудрявцев    1995-12-14',)
('М.Васильев    2000-07-18',)
('А.Нестеров    1995-03-06',)
('Р.Кудряшов    1997-08-26',)
('Е.Ковалев    1995-11-28',)
('М.Мамонтов    2000-09-22',)
('Ю.Жуковский    1996-05-30',)
('Б.Хохлов    1996-07-12',)
('И.Комаров    2000-07-17',)
('Я.Ефимов    2000-09-23',)
('Ф.Кулаков    2001-03-14',)
('Т.Уваров    1996-12-16',)
('Р.Журавлев    1997-07-12',)
('А.Ширяев    1997-09-13',)
('М.Ершов    1995-05-28',)
('М.Бобылев    2001-09-05',)
('М.Бирюков    2000-11-14',)
('В.Ситников    1994-10-05',)
('Х.Агафонов    1997-03-21',)
('Д.Русаков    1993-08-07',)
('С.Симонов    2000-03-17',)
('К.Гуляев    1994-10-13',)
('П.Вячеслав    1997-05-14',)
('А.Баранов    2001-08-18',)
('А.Архипов    2000-01-28',)
('А.Владимиров    2003-05-21',)
('Е.Петрова    1999-05-21',)
('П.Петров    1997-05-21',)
('М.Иванова    2002-04-11',)
('В.Киселев    2003-02-10',)
('Ф.Миронова    2002-04-15',)
('А.Павлов    1979-11-05',)
('А.Петров    1981-08-05',)
('А.Лукин    1981-12-01',)
('Б.Шоу    1979-08-05',)
('Д.Доу    1979-08-05',)
```

12. Напишите запрос для получения списка студентов, фамилии которых начинаются на 'Ков' или на 'Куз'.

In [216]:
```
from sqlalchemy import or_
for item in session.query(student.surname, student.name).where(
    or_(
        student.surname.like("Ков%"),
        student.surname.like("Куз%")
    )
).all():
    print(item)
```

```
('Кузнецов', 'Борис')
('Ковалев', 'Ефим')
```

13. Напишите запрос для получения списка предметов, названия которых оканчиваются на 'ия'.

In [217]:
```
for item in session.query(subject.subj_id, subject.subj_name, subject.hour).where(
    subject.subj_name.like("%ия")
).all():
    print(item)
```

```
(56, 'История', 34)
```

14. Напишите запрос для выбора из таблицы EXAM_MARKS записей, для которых отсутствуют значения оценок (поле MARK).

In [218]:
```
for item in session.query(exam_marks.subj_id, exam_marks.mark).where(
    exam_marks.mark == None
).all():
    print(item)
```

```
(10, None)
```

15. Составьте запрос, выводящий фамилии, имена студентов и величину получаемых ими стипендий,

при этом значения стипендий должны быть увеличены в 100 раз.

In [219]:
```
from sqlalchemy import cast, Integer
for item in session.query(student.surname, student.name, cast((student.stipend * 100), Integer)).all():
    print(item)
```

```
('Иванов', 'Иван', 15000)
('Петров', 'Петр', 20000)
('Сидоров', 'Вадим', 15000)
('Кузнецов', 'Борис', 0)
('Зайцева', 'Ольга', 25000)
('Котов', 'Павел', 15000)
('Белкин', 'Вадим', 25000)
('Сергеева', 'Елизавета', 15000)
('Кудряшова', 'Вера', 10000)
('Журавлева', 'Вера', 0)
('Дементьева', 'Софья', 15000)
('Рожкова', 'Пелагея', 0)
('Соколова', 'Ия', 10000)
('Семенова', 'Вероника', 0)
('Медведева', 'Ирина', 10000)
('Афанасьева', 'Оксана', 25000)
('Сергеева', 'Фекла', 10000)
('Некрасова', 'Вероника', 25000)
('Игнатьева', 'Ольга', 15000)
('Казакова', 'Ольга', 25000)
('Шубина', 'Елена', 0)
('Миронова', 'Анна', 20000)
('Кондрат', 'Снежана', 20000)
('Пономарева', 'Алина', 25000)
('Одинцова', 'Лидия', 10000)
('Гуляева', 'Рузиля', 0)
('Игнатова', 'Зинаида', 10000)
('Мельникова', 'Дарья', 10000)
('Мишина', 'Светлана', 15000)
('Агафонова', 'Василий', 15000)
('Афанасьева', 'Анжелика', 0)
('Фролова', 'Василиса', 10000)
('Лихачева', 'Нина', 15000)
('Лапина', 'Элеонора', 15000)
('Сорокина', 'Ирина', 20000)
('Пашкова', 'Галина', 15000)
('Казакова', 'Мария', 15000)
('Жукова', 'Фекла', 15000)
('Зимина', 'Виктория', 10000)
('Жданова', 'Ирина', 20000)
('Валерия', 'Овчинникова', 20000)
('Костина', 'Кира', 0)
('Блохина', 'Иранда', 25000)
('Мельникова', 'Светлана', 25000)
('Щербакова', 'Агата', 25000)
('Васильева', 'Оксана', 0)
('Сорокина', 'Валерия', 15000)
('Тимофеева', 'Фаина', 15000)
('Вишнякова', 'Анжела', 20000)
('Суравьева', 'Фаина', 15000)
('Борисова', 'Марина', 15000)
('Никитин', 'Самуил', 25000)
('Калашников', 'Евгений', 20000)
('Петров', 'Касьян', 20000)
('Трофимов', 'Владислав', 25000)
('Бирюков', 'Герасим', 0)
('Антонов', 'Анатолий', 10000)
('Медведев', 'Владимир', 20000)
('Морозов', 'Кирилл', 25000)
('Назаров', 'Ирон', 20000)
('Дмитриев', 'Павел', 20000)
('Суханов', 'Константин', 25000)
('Карпов', 'Лука', 20000)
('Захаров', 'Софон', 15000)
('Аксенов', 'Арсений', 25000)
('Панов', 'Арсений', 25000)
('Колобов', 'Иннокентий', 20000)
('Федоров', 'Гостомысл', 10000)
('Лазарев', 'Святослав', 15000)
('Кондратьев', 'Николай', 15000)
('Бирюков', 'Архип', 20000)
('Горский', 'Радмил', 10000)
('Воронова', 'Антонина', 0)
('Гордеев', 'Филипп', 20000)
('Веселов', 'Тимур', 15000)
('Юшин', 'Савва', 0)
('Пономарев', 'Варфоломей', 20000)
('Жуков', 'Пров', 0)
('Кудрявцев', 'Пахом', 15000)
('Васильев', 'Матвей', 10000)
('Нестеров', 'Арсений', 15000)
('Кудряшов', 'Роман', 25000)
('Ковалев', 'Ефим', 20000)
('Мамонтов', 'Максим', 0)
('Жуковский', 'Юрий', 15000)
('Хохлов', 'Борис', 0)
('Комаров', 'Ипат', 0)
('Ефимов', 'Яков', 20000)
('Кулаков', 'Феофан', 15000)
('Уваров', 'Тимур', 0)
('Журавлев', 'Рома', 20000)
('Ширяев', 'Андрей', 15000)
('Ершов', 'Митрофан', 25000)
('Бобылев', 'Михаил', 10000)
('Бирюков', 'Мартин', 10000)
('Ситников', 'Викентий', 15000)
('Агафонов', 'Хасан', 20000)
('Русаков', 'Давид', 25000)
('Симонов', 'Семен', 10000)
('Гуляев', 'Кузьма', 15000)
('Вячеслав', 'Пахомов', 25000)
('Баранов', 'Александр', 22000)
('Архипов', 'Андрей', 18000)
('Владимиров', 'Анатолий', 20000)
('Петрова', 'Екатерина', 25000)
('Петров', 'Вадим', 25000)
('Иванова', 'Мария', 20000)
('Киселев', 'Владимир', 21000)
('Миронова', 'Федор', 20000)
('Павлов', 'Андрей', 0)
('Петров', 'Антон', 20000)
('Лукин', 'Артем', 20000)
('Шоу', 'Бернар', 12000)
('Доу', 'Джон', 12000)
```

16. Составьте запрос для таблицы UNIVERSITY таким образом,

чтобы выходная таблица содержала всего один столбец в следующем виде: Код-10; ВГУ-г.ВОРОНЕЖ; Рейтинг=296.

In [220]:
```
from sqlalchemy import cast, String
for item in session.query("Код-"+cast(university.univ_id, String(20))+"; "+
        cast(university.univ_name, String(20))+"-г."+
        cast(university.city, String(20)) +"; Рейтинг="+
        cast(university.rating, String(20)) +".").all():
    print(item)
```

```
('Код-10; ВГУ-г.Воронеж; Рейтинг=296.',)
('Код-11; НГУ-г.Новосибирск; Рейтинг=345.',)
('Код-14; БГУ-г.Белгород; Рейтинг=326.',)
('Код-15; ТГУ-г.Томск; Рейтинг=368.',)
('Код-18; ВГМА-г.Воронеж; Рейтинг=327.',)
('Код-22; МГУ-г.Москва; Рейтинг=400.',)
('Код-32; РГУ-г.Ростов; Рейтинг=416.',)
('Код-44; ФмиУ-г.Москва; Рейтинг=330.',)
('Код-45; МГТУ-г.Москва; Рейтинг=372.',)
('Код-46; Политех-г.Санкт-Петербург; Рейтинг=300.',)
('Код-47; КФУ-г.Казань; Рейтинг=330.',)
('Код-48; УЛГУ-г.Ульяновск; Рейтинг=231.',)
('Код-49; МТУСИ-г.Москва; Рейтинг=295.',)
```

17. Напишите запрос для подсчета количества студентов, сдававших экзамен по предмету обучения с идентификатором 10.

In [221]:
```
from sqlalchemy import cast, String
for item in session.query(
    func.count(exam_marks.student_id)
).where(exam_marks.subj_id == 10).all():
    print(item)

print("-------")
from sqlalchemy import cast, String
for item in session.query(exam_marks.student_id
).where(exam_marks.subj_id == 10).all():
    print(item)
```

```
(10,)
-------
(97,)
(71,)
(92,)
(105,)
(32,)
(55,)
(15,)
(12,)
(12,)
(55,)
```

18. Напишите запрос, который позволяет подсчитать в таблице EXAM_MARKS количество различных предметов обучения.

In [222]:
```
for item in session.query(exam_marks.subj_id).distinct().all():
    print(item)
print("-------")
session.query(exam_marks.subj_id).distinct().count()
```

```
(10,)
(73,)
(56,)
(12,)
(22,)
(13,)
(94,)
(43,)
(11,)
-------
```

Out[222]: 9

19. Напишите запрос, который для каждого студента выполняет выборку его идентификатора и минимальной из полученных им оценок.

In [223]:
```
for item in session.query(exam_marks.student_id, func.min(exam_marks.mark)).group_by(
    exam_marks.student_id
).all():
    print(item)
```

```
(1, 1)
(6, 4)
(10, 5)
(12, 2)
(15, 5)
(32, 4)
(55, 5)
(62, 4)
(64, 4)
(65, 1)
(71, 3)
(73, 1)
(76, 2)
(77, 1)
(79, 1)
(82, 5)
(83, 1)
(85, 1)
(88, 5)
(90, 4)
(91, 5)
(92, 5)
(97, 1)
(99, 5)
(101, 2)
(103, 2)
(105, 3)
(110, 1)
(116, 1)
(117, 4)
(123, 2)
(126, 1)
(128, 3)
(149, 2)
(203, 3)
```

20. Напишите запрос, который для каждого предмета обучения выводит наименование предмета и максимальное значение номера семестра, в котором этот предмет преподается.

In [224]:
```
for item in session.query(subject.subj_name,
        func.max(subject.semester.label("max_semester"))).group_by(subject.subj_name).all():
    print(item)
```

```
('Анализ данных', 1)
('Английский', 3)
('Информатика', 1)
('История', 4)
('Математика', 2)
('ОБЖ', 2)
('Программирование', 2)
('Физика', 1)
('Физкультура', 5)
```

21. Напишите запрос, который для каждого конкретного дня сдачи экзамена выводит данные о количестве студентов, сдававших экзамен в этот день.

In [225]:
```
for item in session.query(
    func.count(func.distinct(exam_marks.student_id)).label("Кол-во студентов"),
    exam_marks.exam_date.label("Дата экзамена"),
    exam_marks.exam_id.label("id экзамена")
).group_by(exam_marks.exam_date).all():
    print(item)
```

```
(1, datetime.datetime(1999, 6, 17, 0, 0), 238)
(1, datetime.datetime(1999, 6, 22, 0, 0), 639)
(2, datetime.datetime(2000, 1, 5, 0, 0), 43)
(1, datetime.datetime(2000, 1, 23, 0, 0), 34)
(1, datetime.datetime(2006, 1, 12, 0, 0), 145)
(1, datetime.datetime(2019, 5, 11, 0, 0), 18)
(1, datetime.datetime(2019, 6, 8, 0, 0), 95)
(1, datetime.datetime(2019, 7, 4, 0, 0), 25)
(1, datetime.datetime(2019, 7, 6, 0, 0), 30)
(1, datetime.datetime(2019, 7, 24, 0, 0), 14)
(1, datetime.datetime(2019, 8, 14, 0, 0), 16)
(1, datetime.datetime(2019, 8, 30, 0, 0), 23)
(1, datetime.datetime(2019, 9, 2, 0, 0), 12)
(1, datetime.datetime(2019, 10, 28, 0, 0), 28)
(1, datetime.datetime(2019, 12, 17, 0, 0), 31)
(1, datetime.datetime(2019, 12, 25, 0, 0), 10)
(1, datetime.datetime(2019, 12, 31, 0, 0), 20)
(1, datetime.datetime(2020, 2, 10, 0, 0), 13)
(1, datetime.datetime(2020, 2, 22, 0, 0), 9)
(1, datetime.datetime(2020, 3, 13, 0, 0), 8)
(1, datetime.datetime(2020, 3, 20, 0, 0), 3)
(1, datetime.datetime(2020, 3, 21, 0, 0), 22)
(1, datetime.datetime(2020, 4, 6, 0, 0), 6)
(1, datetime.datetime(2020, 5, 4, 0, 0), 17)
(1, datetime.datetime(2020, 5, 10, 0, 0), 5)
(1, datetime.datetime(2020, 5, 13, 0, 0), 21)
(1, datetime.datetime(2020, 7, 26, 0, 0), 1)
(1, datetime.datetime(2020, 8, 10, 0, 0), 4)
(1, datetime.datetime(2020, 8, 18, 0, 0), 26)
(1, datetime.datetime(2020, 9, 9, 0, 0), 32)
(1, datetime.datetime(2020, 9, 21, 0, 0), 27)
(1, datetime.datetime(2020, 10, 24, 0, 0), 29)
(1, datetime.datetime(2020, 12, 3, 0, 0), 100)
(1, datetime.datetime(2020, 12, 4, 0, 0), 11)
(1, datetime.datetime(2020, 12, 23, 0, 0), 2)
(1, datetime.datetime(2021, 2, 10, 0, 0), 7)
(1, datetime.datetime(2021, 3, 1, 0, 0), 15)
(1, datetime.datetime(2021, 3, 7, 0, 0), 19)
(1, datetime.datetime(2021, 3, 14, 0, 0), 33)
(1, datetime.datetime(2021, 3, 29, 0, 0), 24)
```

22. Напишите запрос, выдающий средний балл для каждого студента.

In [226]:
```
for item in session.query(
        exam_marks.student_id,
        func.avg(exam_marks.mark)
).group_by(exam_marks.student_id).all():
    print(item)
```

```
(1, 1.0)
(6, 4.0)
(10, 5.0)
(12, 4.0)
(15, 5.0)
(32, 4.0)
(55, 5.0)
(62, 4.0)
(64, 4.0)
(65, 1.0)
(71, 3.0)
(73, 1.0)
(76, 2.0)
(77, 1.0)
(79, 1.0)
(82, 5.0)
(83, 1.0)
(85, 2.5)
(88, 5.0)
(90, 4.0)
(91, 5.0)
(92, 5.0)
(97, 1.0)
(99, 5.0)
(101, 2.0)
(103, 2.0)
(105, 3.0)
(110, 1.0)
(116, 1.0)
(117, 4.0)
(123, 2.0)
(126, 1.0)
(128, 4.0)
(149, 2.0)
(203, 3.0)
```

23. Напишите запрос, выдающий средний балл для каждого экзамена.

In [227]:
```
for item in session.query(
    exam_marks.exam_id,
    func.avg(exam_marks.mark)
).group_by(exam_marks.exam_id).all():
    print(item)
```

```
(1, 1.0)
(2, 4.0)
(3, 4.0)
(4, 1.0)
(5, 3.0)
(6, 4.0)
(7, 1.0)
(8, 1.0)
(9, 5.0)
(10, 1.0)
(11, 2.0)
(12, 5.0)
(13, 2.0)
(14, 1.0)
(15, 4.0)
(16, 5.0)
(17, 5.0)
(18, 2.0)
(19, 3.0)
(20, 3.0)
(21, 2.0)
(22, 3.0)
(23, 4.0)
(24, 2.0)
(25, 1.0)
(26, 1.0)
(27, 5.0)
(28, 5.0)
(29, 1.0)
(30, 5.0)
(31, 1.0)
(32, 1.0)
(33, 2.0)
(34, 4.0)
(43, 4.0)
(75, 5.0)
(95, 2.0)
(100, 5.0)
(145, 5.0)
(238, 5.0)
(639, None)
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: