Homework 1

You are given the following schema of a University, where primary keys of each table are
underlined and symbol (#) symbolises foreign keys:

Student (student_id, first_name, last_name, date_of_birth)
Course (course_id, #professor_id, course_name, credits, programme)
Exam (#student_id, #course_id, exam_date, grade)
Professor (professor_id, first_name, last_name, research_field)

1. **PART (60%): Use SQL to write the following queries:**
   a) Find the last names of students born after January 1, 2000

```
SELECT l_name
FROM Student
WHERE date_of_birth > '2000-01-01';
```

b) Find the student last names, dates and grades of exams they passed, where a grade was
greater than or equal to 8

```
SELECT St.l_name, Ex.exam_date, Ex.grade
FROM Student JOIN Exam ON St.student_id = Ex.student.id
WHERE grade >= 8;
```

c) Find the first and last names of students who took to at least one exam from "Computer
Science" programme

```
SELECT St.f_name, St.l_name
FROM Student, Course
WHERE St.student_id = Ex.student.id AND Ex.course_id = C.course_id
AND C.programme = 'Computer Science';
```

d) List professor IDs of top 5 professors who teach the most courses

```
SELECT prof_id, COUNT(*) as 'Teached courses'
FROM Course
GROUP BY prof_id
ORDER BY 'Teached courses' DESC
LIMIT 5;
```

e) Calculate the average credits for courses in each programme

```
SELECT programme, AVG(credits)
FROM Course
GROUP BY programme;
```

f) Find the surnames of students who have never taken an exam from “Mathematics” programme

```
SELECT St.l_name
FROM Student St
WHERE St_id NOT IN (
SELECT *
FROM Exams Ex
JOIN Cources Cour ON Ex.Course_id = Cour_id
WHERE Cour.programme = 'Mathematics' )
```

2. **PART (20 %): Use RELATIONAL ALGEBRA to write the following queries:**
   a) Find names and surnames of professors, who teach a course in "Biology" programme

   Solution:

   $\pi_{f\_name,\, l\_name}(\sigma_{programme=''Biology''})(Professor \ \bowtie Course)$

   b) Find the student IDs and names of students who have not taken any exam (from any course)

Solution:

$$\pi_{student\_id,\ f\_name}((Students) - \pi_{student\_id, f\_name}Courses)$$

c) Find student IDs of students who have passed every exam from "Data Science" programme

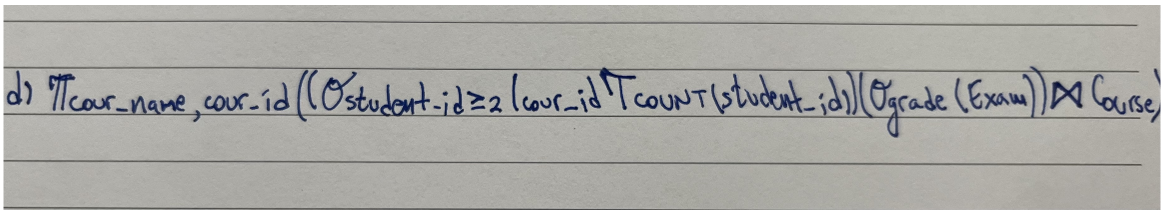
/*I'm uncertain about how this task should be done.*/

Solution:

$$\pi_{student\_id}(\sigma programme =' \ DataScience' \wedge)()$$

d) Find the course IDs and names of courses that have been passed by at least two different
students (If a course was taken, it means that a student wrote an exam on that course)

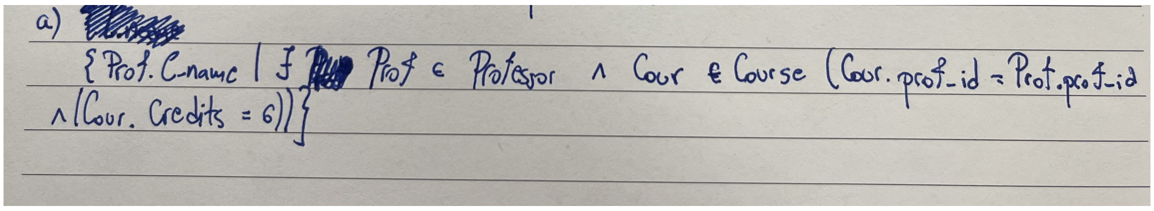
Solution:



d) $\pi_{cour\_name, cour\_id}((\sigma_{student\_id \geq 2}(_{cour\_id}\Upsilon_{COUNT(student\_id)})(\sigma_{grade}(Exam)) \bowtie Course)$

3. **PART (20 %): Use DOMAIN or TUPLE RELATIONAL CALCULUS to write the following queries:**
   a) Find the last names of professors who have taught a course with 6 credits

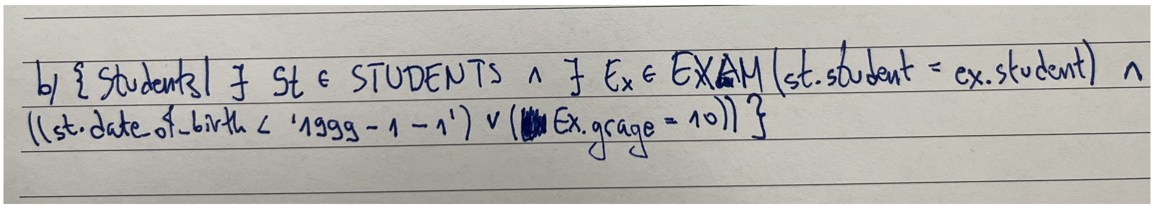
   Solution:



a) $\{Prof.C\_name \mid \exists\ Prof \in Professor \wedge Cour \in Course\ (Cour.prof\_id = Prof.prof\_id$
$\wedge (Cour.Credits = 6))\}$

b) Find all student IDs of students who are born before January 1, 1999 or have passed at least
one exam with a 10

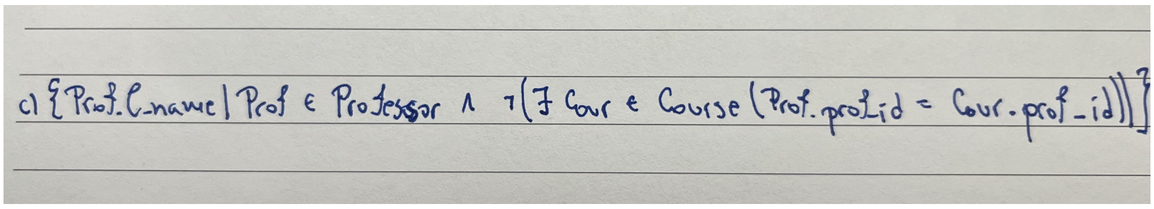
Solution:



b) $\{ Students \mid \exists\ St \in STUDENTS \wedge \exists\ Ex \in EXAM\,(st.student = ex.student) \wedge$
$((st.date\_of\_birth < '1999-1-1') \vee (Ex.grage = 10))\}$

c) Find the last names of professors who have never taught a course

Solution:



c) $\{ Prof.l\_name \mid Prof \in Professor \wedge \neg(\exists\ Cour \in Course\,(Prof.prof\_id = Cour.prof\_id))\}$

d) Find the student ids of student(s) who took the latest (most recent) exam

Solution: