Optimization process:

We first tried to answer the queries normally. Splitting the queries into sub-problems, and answering those first.
Afterwards we combined these into full queries which would give the correct answer.
Then we tried to optimise these queries, look at their runtimes, see why they are slow, or too slow. We prioritized queries that had to be executed multiple times, as slow run-times here would be too costly. Whilst doing this we also looked at sub-problems that were needed for multiple queries, to later on convert into views, such as number of passed courses, or active students.
If we weren't satisfied with the query optimizations, we resorted to using materialized views, to speed up the slow part of the query. Or had to leave out the query due to time constraints.

Chosen optimizations:

We decided to use multiple materialized views to greatly optimize lookups on multiple tables, and to precompute some values, like GPA, and accrued ECTS of a StudentRegistrationId. In short we had the following materialized views: Passed courses per student registration id, failed courses per student registration id, sum of the ECTS per student registration id, and a view with the student GPA per student registration id. All these are able to be made within time limit as most of the work is computed by the first view. The passed, and failed courses views are fairly large. Both hovering around ~1,5GB. The sum of the ECTS and the GPA views are considerably smaller, only ~350mb each.
Using these views we are able to answer one execution of Q1 in about ~3,5 seconds. And Q2 in ~16,5 seconds. Without these views Q1 costed above 10 seconds, and Q2 over 40.