# Assignment 4B

## CS20B008

## Group - 17

- **Query**

  Get the roll number, name and cgpa of students whose name starts with ‘A’, have male advisors, were not enrolled in the year 2001 and never failed in any course they took. (S grade: 10, A grade: 9, B grade: 8, C grade: 7, D grade: 6, E grade: 5, U grade: Fail)

```
8   select s.rollNo, s.name, SUM(
9   (case
10      when e.grade = 'E' THEN 5
11      when e.grade = 'D' THEN 6
12      when e.grade = 'C' THEN 7
13      when e.grade = 'B' THEN 8
14      when e.grade = 'A' THEN 9
15      when e.grade = 'S' THEN 10
16  end)*(c.credits)
17  ) / SUM(c.credits) as cgpa
18  from student as s, course as c, enrollment as e, professor as p
19  where s.advisor = p.empId and p.sex = 'male' and (s.year != 2001)
20  and e.grade != 'U' and e.courseId = c.courseId and e.rollNo = s.rollNo and s.name like 'A%'
21  group by s.rollNo
22  order by cgpa desc;
```

| rollNo | name | cgpa |
|---|---|---|
| 89188 | Anse | 9.0000 |
| 79487 | Androutsopoulos | 8.6000 |
| 65676 | Aufr | 8.5500 |
| 96615 | Anty | 8.5263 |
| 31080 | Aschoff | 8.2903 |
| 69471 | Aly | 8.2800 |
| 81150 | Atkins | 8.1538 |
| 14621 | Azevedo | 8.1429 |
| 94726 | Ailamaki | 8.1200 |
| 11152 | Al-Tahat | 8.0000 |
| 13081 | Alqui | 7.8571 |
| 58326 | Afim | 7.8462 |
| 20084 | Adda | 7.7778 |
| 81876 | Arora | 7.6364 |
| 46655 | Advani | 7.3333 |
| 42092 | Arinb | 7.1429 |
| 69222 | Albuquerque | 7.0000 |
| 87222 | Allard | 6.9545 |
| 67024 | Aufr | 6.9333 |
| 30334 | Arakawa | 6.8750 |
| 77588 | Aguilar | 6.4706 |
| 83622 | Achilles | 6.4545 |
| 22050 | Alioto | 6.3333 |
| 14499 | Axte | 5.8571 |
| 77664 | Apostolov | 5.8000 |
| 24325 | Álvarez | 5.7273 |

❖ EXPLAIN command output before creating index

| id | select_ty... | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | SIMPLE | p | NULL | ALL | PRIMARY | NULL | NULL | NULL | 52 | 10.00 | Using where; Using temporary; Using filesort |
| 1 | SIMPLE | s | NULL | ref | PRIMARY,deptNo,advisor | advisor | 23 | academic_insti.p.empId | 40 | 10.00 | Using where |
| 1 | SIMPLE | e | NULL | ref | PRIMARY,courseId | PRIMARY | 22 | academic_insti.s.rollNo | 6 | 90.00 | Using where |
| 1 | SIMPLE | c | NULL | eq_ref | PRIMARY | PRIMARY | 34 | academic_insti.e.courseId | 1 | 100.00 | NULL |

MySQL engine accesses the relations in the following order:

1. professor
2. student
3. enrollment
4. course

It goes through

**52 x 40 x 6 x 1 = 12,480 rows**

to get the final output of the given query.

Also, notice that the percentage of rows filtered is quite low for professor(10%) and student(10%), hence we need to create some indices to increase the filtration percentages and reduce the total number of rows accessed.

❖ EXPLAIN command output after creating index

We create the following indexes:

```
5 •  create index s1 on student(name ASC, year ASC);
6 •  create index p1 on professor(sex DESC);
```

This is the output that we get:

| id | select_ty... | table | partitions | type | possible_keys | key | key_len | ref | rows | filtered | Extra |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | SIMPLE | s | NULL | range | PRIMARY,deptNo,advisor,s1 | s1 | 86 | NULL | 105 | 90.00 | Using index condition; Using where; Using temporary; Using filesort |
| 1 | SIMPLE | p | NULL | eq_ref | PRIMARY,p1 | PRIMARY | 22 | academic_insti.s.advisor | 1 | 50.00 | Using where |
| 1 | SIMPLE | e | NULL | ref | PRIMARY,courseId | PRIMARY | 22 | academic_insti.s.rollNo | 6 | 90.00 | Using where |
| 1 | SIMPLE | c | NULL | eq_ref | PRIMARY | PRIMARY | 34 | academic_insti.e.courseId | 1 | 100.00 | NULL |

After creating the indices, the MySQL engine goes through

**105 x 1 x 6 x 1 = 630 rows**

and the filtering percentage has also increased(90% for student and 50% for professor) significantly. As we can see the total number of rows accessed decreased to 630 from 12,480.

This shows a huge improvement in the performance using indexing.