UNIVERSITY COMPUTER SCIENCE • EXERCISE SHEET

8.2 Relational algebra and SQL

Name: _____ Class: _____ Date: _____ **Total: 24 marks**

KEY WORDS projection • selection

Objective

Read queries as pipelines and predict their exact outputs.

1 Key knowledge and worked examples

■ The course query

Join Course and Enrollment, select credits = 15, group by title, keep groups ≥ 2, project title with the count: executed result `Databases | 3` ✓.

■ Worked example: join flavors

The plain join drops the 2 empty courses; `LEFT JOIN` keeps 6 rows, two with $n = 0$ – `COUNT(e.student_id)` skips the NULLs the misses produce ✓.

2 Practice

2.1 Name the operator: `WHERE credits = 15`. [3]

2.2 Name the operator: `SELECT title FROM ….` [3]

2.3 WHERE versus HAVING, one line each. [3]

3 Exam-style questions

3.1 The sample schema: Student(8 rows), Course(6), Enrollment(14); Databases (course 1, 15 credits) has 3 enrollments; Compilers and Topology have none.

Write the result table (rows and values) of the course query with HAVING COUNT(*) ≥ 2. Write the result of the LEFT JOIN version listing every course with its count, ordered by title. State the row count of `SELECT student_id FROM Enrollment` and of its DISTINCT version on data where 8 students enrolled exactly 14 times with 6 repeat enrollees. [8]

3.2 Explain, in clause order, what this query does at each step:

```
SELECT d.name, COUNT(*) AS n
FROM Department d JOIN Course c ON d.dept_id = c.dept_id
WHERE c.credits >= 12
GROUP BY d.name
ORDER BY n DESC;
```

Name the clause that would change if only departments with at least 2 such courses should appear, and write that clause. State the result row count before and after that change (2 departments, 4 courses of ≥ 12 credits split 2 and 2). [7]

Solutions

2.1

- Selection σ (a WHERE filter on rows) ✓.

2.2

- Projection π (keeping the named columns) ✓.

2.3

- WHERE filters rows before grouping; HAVING filters groups after ✓.

3.1

- Course query result: one row – `Databases | 3` ✓.
- LEFT JOIN: Algorithms 2, Calculus 5, Compilers 0, Databases 3, Linear Algebra 4, Topology 0 – 6 rows ✓.
- `SELECT student_id FROM Enrollment`: 14 rows ✓.
- DISTINCT: 8 rows – the 6 repeats collapse ✓.

3.2

- FROM/JOIN builds Department x Course rows following the foreign key ✓.
- WHERE keeps courses with credits ≥ 12 ✓.
- GROUP BY partitions by department name; SELECT projects name and the count; ORDER BY sorts descending ✓.
- The change is a HAVING clause: `HAVING COUNT(*) >= 2` ✓.
- Before: 2 result rows (one per department); after: still 2 rows – each department has exactly 2 such courses, so both survive the filter ✓.