```
# */Question 1/*
USE ischool_v2;

SELECT
    student_name,
    enrollment_count,
    credit_count,
    latest_end
FROM (
    SELECT
        CONCAT(p.lname, ', ', p.fname) AS student_name,
        COUNT(*) AS enrollment_count,
        SUM(c.credits) AS credit_count,
        TIME_FORMAT(MAX(cs.end_time), "%r") AS latest_end
    FROM
        people p
        JOIN enrollments e ON p.person_id = e.person_id
        JOIN course_sections cs ON e.section_id = cs.section_id
        JOIN courses c ON cs.course_id = c.course_id
    GROUP BY
        p.person_id
    HAVING
        COUNT(e.section_id) = 5
) AS subquery
ORDER BY
    student_name;



/* Question 2 */
WITH EnrollmentCounts AS (
    SELECT
        cs.course_id,
        COUNT(DISTINCT e.person_id) AS total_enrollment,
        COUNT(DISTINCT cs.section_id) AS section_count
    FROM
        course_sections cs
    JOIN
        enrollments e ON cs.section_id = e.section_id
    GROUP BY
        cs.course_id
    HAVING
        COUNT(DISTINCT e.person_id) > 0
)
SELECT
    CONCAT('INST', c.course_number) AS course_code,
    c.course_description,
    ec.section_count,
    ec.total_enrollment,
    ec.total_enrollment / ec.section_count AS average_enrollment
FROM
    courses c
JOIN
    EnrollmentCounts ec ON c.course_id = ec.course_id;

/* Extra Credit */
#For clarity and organization, I chose to use a CTE
#which allows me to establish a named temporary result set that separates the logic
```

of calculating enrollments from the primary query.
#By breaking down the query into logical steps, the CTE helps readability, making the overall process easier to understand and maintain.
#Furthermore, by segregating the aggregate logic for enrollments and section counts, utilizing a CTE simplifies debugging and any future expansions of the query.