1. Using the "studenti" table from the studenti.sql script create a **view** that includes, for each student: last_name, first_name, section_code, section_name, year_of_study, group, and average; using this view, create additional views to solve the following problems (which can be easily solved using **analytical functions**):

a. For each student, display: last name, first name, group, average, the average of the section they belong to, the deviation from the section average (variance), the student's position within their year of study (ordered descending by average)
b. Students with the **top 3 averages in each group**.

2. The cursz table from the master schema can be used. If using your own instances, you may consider the following table in the current schema:

```
CREATE TABLE cursz (
zi number(2),
luna number(2),
an number(4),
valoare number(6,4),
moneda char(3) );
```

*(The table contains daily exchange rates over a period of time for multiple currencies. A script for inserts will be available for use.)*

(The MATCH_RECOGNIZE function is available starting from Oracle 12c, but not in the Express Edition.) For students using a version that does **not** support MATCH_RECOGNIZE, the query must be presented **with appropriate comments**, so the solution method can still be understood.

a. Identify the time intervals where the **exchange rate for EURO** decreased for **at least 10 consecutive days** (display number of days in the decreasing period and the difference in value from the start to the end of the period)
b. Create **another problem** that identifies a specific **pattern** (e.g., **V**, **W**, **M**, etc.) using data from this table.

3. Create a user-defined package that contains at least 3 functions/procedures, as follows:

a. Display data from the views created in point 1, with the ability to **filter results based on at least 2 input parameters**.
b. Include a **logging function** to insert records into a **log/journal table** each time a function/procedure from point a is executed. This should happen either **at the start and/or end of execution** to allow calculating the duration. The logging should work for both: successful executions and executions with errors. The **log table** must contain: the name of the executed function, execution duration, status (success or error) and error message (if any).

**Deadline:** April 29th