SQL 1 Laboratory

Author: Piotr Copek

Date: 02.04.2025

Task 1. Display the name of the student with identified by 5.

Solution:

```
SELECT STUDENT_NAME
FROM STUDENTS
WHERE STUDENT_ID = 5;
```

I selected the student's name where `STUDENT_ID` was equals 5.

Task 2. Display the names of subjects starting with C.

Solution:

```
SELECT SUBJECT_NAME
FROM SUBJECTS
WHERE SUBJECT_NAME LIKE 'C%';
```

I used `LIKE 'C%'` to find subject names starting with "C".

Task 3. Display the names of students majoring in ROBOTICS.

Solution:

```
SELECT S.STUDENT_NAME
FROM STUDENTS S
JOIN MAJORS M ON S.MAJOR_ID = M.MAJOR_ID
WHERE M.MAJOR_NAME = 'ROBOTICS';
```

I joined `STUDENTS` with `MAJORS` to filter students majoring in `ROBOTICS`.

Task 4. Display in alphabetical order the names of female students, who have grade 2 in any subject.

Solution:

```
SELECT DISTINCT S.STUDENT_NAME
FROM STUDENTS S
JOIN GRADES G ON S.STUDENT_ID = G.STUDENT_ID
WHERE S.GENDER = 'F' AND G.GRADE = 2
ORDER BY S.STUDENT_NAME ASC;
```

I selected distinct female students with grade 2, ordering them alphabetically.

Task 5. Prepare the list of lectures (subject with type='Lecture') given to COMPUTER CONSTRUCTION students.

Solution:

```
SELECT s.subject_name
FROM SUBJECTS s
JOIN SUBJECT_TYPES st ON s.sub_type_id = st.sub_type_id
JOIN MAJORS m ON s.major_id = m.major_id
WHERE st.sub_type_name = 'Lecture' AND m.major_name = 'COMPUTER CONSTRUCTION';
```

I joined `SUBJECTS`, `SUBJECT_TYPES`, and `MAJORS` to find lectures for `COMPUTER CONSTRUCTION` students.

Task 6. Display in reverse alphabetical order the names of male students who are in major PROCESS CONTROL or ELECTROTECHNOLOGY.

Solution:

```sql
SELECT S.STUDENT_NAME
FROM STUDENTS S
JOIN MAJORS M ON S.MAJOR_ID = M.MAJOR_ID
WHERE S.GENDER = 'M' AND M.MAJOR_NAME IN ('PROCESS CONTROL', 'ELECTROTECHNOLOGY')
ORDER BY S.STUDENT_NAME DESC;
```

I selected male students majoring in `PROCESS CONTROL` or `ELECTROTECHNOLOGY`, sorting names in reverse order.

Task 7. Display the names of subjects with at least one grade 2 given. The names mustn't repeat.

Solution:

```sql
SELECT DISTINCT SJ.SUBJECT_NAME
FROM SUBJECTS SJ
JOIN GRADES G ON SJ.SUBJECT_ID = G.SUBJECT_ID
WHERE G.GRADE = 2;
```

I used DISTINCT to retrieve subjects having at least one grade 2.

Task 8. Display the names of subjects with their parent subjects.

Solution:

```sql
SELECT s.subject_name, p.subject_name AS super_sub
FROM SUBJECTS s
JOIN SUBJECTS p ON s.super_sub_id = p.subject_id;
```

I self-joined `SUBJECTS` to retrieve subjects with their parent subjects.

Task 9. Display the number of COMPUTER SCIENCE students.

Solution:

```sql
SELECT COUNT(*) AS Number
FROM STUDENTS S
JOIN MAJORS M ON S.MAJOR_ID = M.MAJOR_ID
WHERE M.MAJOR_NAME = 'COMPUTER SCIENCE';
```

I counted students majoring in `COMPUTER SCIENCE` using `COUNT(*)`.

Task 10. Display an average of grades of student COLLEGE.

Solution:

```sql
SELECT AVG(G.GRADE) AS avg
FROM GRADES G
JOIN STUDENTS S ON G.STUDENT_ID = S.STUDENT_ID
WHERE S.STUDENT_NAME = 'COLLEGE';
```

I calculated the average grade of the student named `COLLEGE` using `AVG`.

Task 11. Display a number of students of MINING MACHINERY major.

Solution:

```sql
SELECT COUNT(*) AS cnt
FROM STUDENTS S
JOIN MAJORS M ON S.MAJOR_ID = M.MAJOR_ID
WHERE M.MAJOR_NAME = 'MINING MACHINERY';
```

I counted students majoring in `MINING MACHINERY`.

Task 12. Display the lowest and the highest grade got by each student having name starting with ‘B’.

Solution:

```
SELECT S.STUDENT_ID, S.STUDENT_NAME, MIN(G.GRADE) AS min, MAX(G.GRADE) AS max
FROM STUDENTS S
JOIN GRADES G ON S.STUDENT_ID = G.STUDENT_ID
WHERE S.STUDENT_NAME LIKE 'B%'
GROUP BY S.STUDENT_ID, S.STUDENT_NAME;
```

I retrieved the minimum and maximum grades for students whose names start with `B`.

Task 13. Display number of different names of students of MINING MACHINERY major.

Solution:

```
SELECT COUNT(DISTINCT STUDENT_NAME) AS cnt
FROM STUDENTS S
JOIN MAJORS M ON S.MAJOR_ID = M.MAJOR_ID
WHERE M.MAJOR_NAME = 'MINING MACHINERY';
```

I counted distinct student names in the `MINING MACHINERY` major using `COUNT(DISTINCT STUDENT_NAME)`.

Task 14. For each major display the dates of birth of the oldest students.

Solution:

```
SELECT M.MAJOR_ID, M.MAJOR_NAME, MIN(S.DATE_OF_BIRTH) AS "min(date_of_birth)"
FROM MAJORS M
JOIN STUDENTS S ON M.MAJOR_ID = S.MAJOR_ID
GROUP BY M.MAJOR_ID, M.MAJOR_NAME;
```

I found the oldest student's birthdate in each major using `MIN(DATE_OF_BIRTH)`.

Task 15. For each student with name starting with B display the number of grades that he or she got in each subject.

Solution:

```sql
SELECT S.STUDENT_NAME, SJ.SUBJECT_NAME, COUNT(*) AS "count(*)"
FROM STUDENTS S
JOIN GRADES G ON S.STUDENT_ID = G.STUDENT_ID
JOIN SUBJECTS SJ ON G.SUBJECT_ID = SJ.SUBJECT_ID
WHERE S.STUDENT_NAME LIKE 'B%'
GROUP BY S.STUDENT_NAME, SJ.SUBJECT_NAME;
```

I counted the number of grades each student with a name starting with `B` received per subject.

Task 16. Display the names of subjects which have more than 8 grades given.

Solution:

```sql
SELECT SJ.SUBJECT_NAME
FROM SUBJECTS SJ
JOIN GRADES G ON SJ.SUBJECT_ID = G.SUBJECT_ID
GROUP BY SJ.SUBJECT_NAME
HAVING COUNT(*) > 8;
```

I retrieved subjects with more than 8 recorded grades using `HAVING COUNT(*) > 8`.

Task 17. For each student from SOFTWARE major display the number of subjects in which he or she has grades (number of subjects not the number of grades!).

Solution:

```sql
SELECT S.STUDENT_ID, S.STUDENT_NAME, COUNT(DISTINCT G.SUBJECT_ID) AS Cnt
FROM STUDENTS S
JOIN GRADES G ON S.STUDENT_ID = G.STUDENT_ID
JOIN MAJORS M ON S.MAJOR_ID = M.MAJOR_ID
WHERE M.MAJOR_NAME = 'SOFTWARE'
GROUP BY S.STUDENT_ID, S.STUDENT_NAME;
```

I counted distinct subjects where `SOFTWARE` students received grades.

Task 18. Display the names of students who are younger than student CASAN.

Solution:

```sql
SELECT S1.STUDENT_NAME
FROM STUDENTS S1
WHERE S1.DATE_OF_BIRTH > (
    SELECT S2.DATE_OF_BIRTH
    FROM STUDENTS S2
    WHERE S2.STUDENT_NAME = 'CASAN'
);
```

I selected students younger than `CASAN` by comparing birthdates.

Task 19. Display the names of students whose average of grades is higher than student HAT.

Solution:

```
SELECT S.STUDENT_NAME
FROM STUDENTS S
JOIN GRADES G ON S.STUDENT_ID = G.STUDENT_ID
GROUP BY S.STUDENT_ID, S.STUDENT_NAME
HAVING AVG(G.GRADE) > (
    SELECT AVG(G2.GRADE)
    FROM GRADES G2
    JOIN STUDENTS S2 ON G2.STUDENT_ID = S2.STUDENT_ID
    WHERE S2.STUDENT_NAME = 'HAT'
);
```

I retrieved students whose average grade is higher than `HAT’s`.

Task 20. Display the names of majors having more students than COMPUTER SCIENCE.

Solution:

```
SELECT M.MAJOR_NAME
FROM MAJORS M
JOIN STUDENTS S ON M.MAJOR_ID = S.MAJOR_ID
GROUP BY M.MAJOR_ID, M.MAJOR_NAME
HAVING COUNT(*) > (
    SELECT COUNT(*)
    FROM STUDENTS S2
    JOIN MAJORS M2 ON S2.MAJOR_ID = M2.MAJOR_ID
    WHERE M2.MAJOR_NAME = 'COMPUTER SCIENCE'
);
```

I found majors with more students than `COMPUTER SCIENCE` using a `HAVING COUNT(*)` condition.