Databases
Seminar 2

SQL – Data Manipulation Language

The SELECT Statement

Students[sid, sname, email, age, sgroup]
Courses[cid, cname, credits]
Exams[sid, cid, grade]

Q1. Find the 21-year-old students in the *Students* table.
```
SELECT *
FROM Students S
WHERE S.age = 21
```

Q2. Find the names and email addresses of all the 21-year-old students in the *Students* table. Eliminate duplicates.
```
SELECT DISTINCT S.sname, S.email
FROM Students S
WHERE S.age = 21
```

Range variable
- alias used for a table in a SQL query;
- needed when a relation appears more than once in the FROM clause (to solve the ambiguity);
- it is good style to always use range variables; compare the following versions of the same query:

Q3. Find the *"10"* grades (student name, course id).
```
SELECT S.sname, E.cid
FROM Students S, Exams E
WHERE S.sid = E.sid AND E.grade = 10

SELECT sname, cid
FROM Students, Exams
WHERE Students.sid = Exams.sid AND grade = 10
```

Arithmetic expressions and the LIKE operator
Q4. For all students whose name starts and ends with B and has at least 3 characters, retrieve the following data: student age, student age – 18, student age * 2.
```
SELECT S.age, age1 = S.age-18, 2*S.age AS age2
FROM Students S
WHERE S.sname LIKE 'B_%B'
```

- 'AS' and '=' can be used to name fields in the result set;
- the LIKE operator is used for string pattern matching:
    - '_' matches any one character;
    - '%' matches 0 or more arbitrary characters.

Set operations
UNION, INTERSECT, EXCEPT: compute the union / intersection / difference of any 2 union-compatible sets of tuples (results of SQL queries). Duplicate rows are eliminated.

Q5. Find the ids of students who are older than 20 or have a grade in the *Alg1* course.
```
SELECT S.sid
FROM Students S
WHERE S.age > 20
UNION
SELECT E.sid
FROM Exams E
```

```
WHERE E.cid = 'Alg1'
--UNION ALL doesn't eliminate duplicates
```

Q6. Find the ids of students who received a grade in both a 4 credits course and a 5 credits course.

```
SELECT E.sid
FROM Exams E, Courses C
WHERE E.cid = C.cid AND C.credits = 4
INTERSECT
SELECT E2.sid
FROM Exams E2, Courses C2
WHERE E2.cid = C2.cid AND C2.credits = 5
```

Q7. Find the ids of students who received a grade in a 4 credits course, but have no grades in 5 credits courses.

```
SELECT E.sid
FROM Exams E, Courses C
WHERE E.cid = C.cid AND C.credits = 4
EXCEPT
SELECT E2.sid
FROM Exams E2, Courses C2
WHERE E2.cid = C2.cid AND C2.credits = 5
```

<u>Nested queries</u>

- a query can contain another query (a subquery), e.g., in the WHERE, FROM, HAVING clauses;

Q8. Find the names of students who are not graded in *Alg1*.

```
SELECT S.sname
FROM Students S
WHERE S.sid NOT IN (SELECT E.sid
                FROM Exams E
                WHERE E.cid = 'Alg1')

SELECT S.sname
FROM Students S
WHERE NOT EXISTS
      (SELECT *
      FROM Exams E
      WHERE E.sid = S.sid AND E.cid = 'Alg1')
```

- the semantics of the nested query: the subquery is evaluated when testing the condition in the WHERE clause of the main query.

Q9. Find students who are older than some student called *Ion*.

```
SELECT *
FROM Students S
WHERE S.age > ANY (SELECT S2.age
                  FROM Students S2
                  WHERE S2.sname = 'Ion')
```

Q10. Find students who are older than all the students called *Ion*.

```
SELECT *
FROM Students S
WHERE S.age > ALL (SELECT S2.age
                  FROM Students S2
                  WHERE S2.sname = 'Ion')
```

SQL – Data Manipulation Language

The IN operator - it tests whether a value belongs to a set of elements; the latter can be explicitly specified or generated by a query.
The EXISTS operator - it tests whether a set is non-empty.
The ANY operator - it evaluates to true if the condition is true for at least one item in the subquery result.
The ALL operator - it evaluates to true if the condition is true for all the items in the subquery result.

JOIN operations

Students

sid	sname	email	age	sgroup
1234	Ada	a@cs.ro	20	921
1235	Razvan	r@cs.ro	21	921
1236	Monica	m@cs.ro	20	922

Courses

cid	cname	credits
Alg1	Algorithms 1	7
DB1	Databases 1	6
DB2	Databases 2	6

Exams

sid	cid	grade
1234	Alg1	9
1235	Alg1	10
1237	DB2	9

JOIN operator	Example query	Result
INNER JOIN	`Q11. SELECT S.sname, C.cname FROM Students S INNER JOIN Exams E ON S.sid = E.sid INNER JOIN Courses C ON E.cid = C.cid`	sname | cname Ada | Algorithms 1 Razvan | Algorithms 1
LEFT OUTER JOIN (e.g., students with no grades should also appear in the result set)	`Q12. SELECT S.sname, C.cname FROM Students S LEFT OUTER JOIN Exams E ON S.sid = E.sid LEFT OUTER JOIN Courses C ON E.cid = C.cid`	sname | cname Ada | Algorithms 1 Razvan | Algorithms 1 Monica | NULL
RIGHT OUTER JOIN (e.g., also find the grades given by mistake to nonexistent students)	`Q13. SELECT S.sname, C.cname FROM Students S RIGHT OUTER JOIN Exams E ON S.sid = E.sid INNER JOIN Courses C ON E.cid = C.cid`	sname | cname Ada | Algorithms 1 Razvan | Algorithms 1 NULL | Databases 2
FULL OUTER JOIN (LEFT + RIGHT OUTER JOIN)	`Q14. SELECT S.sname, C.cname FROM Students S FULL OUTER JOIN Exams E ON S.sid = E.sid FULL OUTER JOIN Courses C ON E.cid = C.cid`	

sname	cname
Ada	Algorithms 1
Razvan	Algorithms 1
NULL	Databases 2
NULL	Databases 1
Monica	NULL

Obs. The following queries return the same result set:
```
SELECT *
FROM Students S INNER JOIN Exams E ON S.sid = E.sid

SELECT *
FROM Students S, Exams E
WHERE S.sid = E.sid
```

Aggregation operators

- `COUNT(*)`
- `COUNT([DISTINCT] A)`
- `SUM([DISTINCT] A)`
- `AVG([DISTINCT] A)`
- `MAX(A)`
- `MIN(A)`

, where A is an attribute name in a table.

- evaluated on a set of values, corresponding to a group of records;
- NULL values: seminar discussion – *SELECT COUNT(*), SELECT COUNT(A), SELECT COUNT(DISTINCT A)*;
- expressions.

Q15. Find the number of students.
```
SELECT COUNT(*)
FROM Students S
```

Q16. Find the average and minimum age for group 924.
```
SELECT AVG(S.age), MIN(S.age)
FROM Students S
WHERE S.sgroup = 924
```

Q17. Find the number of groups that have at least one student called *Mihai*.
```
SELECT COUNT(DISTINCT S.sgroup)
FROM Students S
WHERE S.sname = 'Mihai'
```

Q18. Find the name and age of the oldest student.
```
SELECT S.sname, S.age
FROM Students S
WHERE S.age = ANY
      (SELECT MAX(S2.age)
      FROM Students S2)
```

SQL – Data Manipulation Language

GROUP BY

Q19. For each 6 credits course, find the number of grades and their average.

```
SELECT C.cid, COUNT(*) AS no_gr, AVG(grade) AS gr_avg
FROM Exams E, Courses C
WHERE E.cid = C.cid AND C.credits = 6
GROUP BY C.cid
```