Nested Select Statement

- **Nested queries**
  - `SELECT` contains another `SELECT`
- **Two options**
  - `SELECT` returning a single value
  - `SELECT` returning a table

Nested Select Statement

- **Simple case**
  - In WHERE we compare the value of an attribute with the value returned by the `SELECT`
  - operators: `>, <, =, >=, <=, <>, LIKE, IS NULL`

- **`"Scalar" SELECT`**
  - SELECT returns only one tuple with a single attribute
  - Aggregation function

Nested Select Statement

- **Example**: Return the student with the highest value for studentID

```
SELECT studentID, surame, name
FROM Students
WHERE studentID =  (SELECT max(studentID)
                     FROM Students);
```

max(studentID)
88888

For each tuple of Students, the value of studentID is compared with the number 88888

Nested Select Statement

- **Conditions on values that are not scalar (collection of tuples)**
  - Compare the value of an attribute with the result of a generic SELECT (collection of tuples)
  - operators: standard operators combined with `ANY, ALL`
- **`ANY`**
  - “any element of the collection”; ex: `=ANY`, or `IN`
- **`ALL`**
  - “all elements of the collection”; ex: `>ALL`

Nested Select Statement

- **Example**: the student with the highest studentID (without aggregation functions)

```
SELECT studentID, surname, name
FROM Students
WHERE studentID >= ALL (SELECT studentID
                        FROM Students);
```

For each tuple of Students, the value of studentID is compared with all studentIDs in Students

studentID
111
222
333
444
77777
88888

Nested Select Statement & ORDER BY

  - NO `ORDER BY` in nested `SELECT`

- **Semantics of NESTED SELECT**
  - Any time it is necessary to test the condition, the result of the internal SELECT is computed
  - The process may be repeated for more intervals
  - in practice: store a temporary table

Nested Select Statement

- **Note**: Nested queries may substitute JOIN
- **Example**: The list of marks for courses of bachelor program

```
SELECT mark
FROM Exams
WHERE course = ANY (SELECT code
                    FROM Courses
                    WHERE program=‘bachelor’);
```

code
PR1
ASD

Same semantics of the JOIN

Nested Select Statement

- **Note**: Nested Queries may substitute intersection and difference
- **Example**: surname and name of full professors who are not supervisioning any thesis

```
SELECT surname, name
FROM teachers
WHERE role=‘full’ AND
code <>  ALL (SELECT DISTINCT supervisor
                    FROM Students
                    WHERE  supervisor IS NOT NULL);
```

supervisor
FT
VC

<>ALL equal to NOT IN

= ANY equal to IN

Nested QUERIES in DML and DDL

- **DML**
  - `DELETE`, `UPDATE`

- **DDL**
  - `CHECK` (<condition>)
  - <condition>: syntax and semantics equal to the condition in `WHERE`

Nested QUERIES in DML and DDL

- **Example**: It is possible to do exams only for courses having a teacher

```
CREATE TABLE EXAMS (
   student integer REFERENCES Students(studentID)
                  ON DELETE cascade
                  ON UPDATE cascade,
   course char(3) REFERENCES Courses(code),
   mark integer,
   laud bool,
   CHECK (mark>=18 and mark<=30),
   CHECK (not laud or mark=30),
   PRIMARY KEY (student, course));
```

Constraint

```
CHECK (course = ANY
      (SELECT code
       FROM Courses
       WHERE teacher IS NOT NULL))
```

Views

- **Views**
  - Temporary table that is not part of the schema

- **Example**: **Students with the highest average of the mark**
  - For computing the average for each student it is necessary a grouping
  - Nested Condition on the groups
  - We cannot put nested select in `HAVING` (only in `WHERE`)

Views

- Solution with views

```
CREATE  VIEW AVG_Students AS
SELECT studentID, surname, name, avg(mark) as AVG
FROM Exams JOIN Students  ON student=studentID
GROUP  BY studentID, surname, name;
```

```
SELECT studentID, surname, name
FROM AVG_Students
WHERE AVG =  (SELECT max(AVG)
                         FROM AVG_Students );
```

AVG_Students

StudentID	Surname	Name	AVG
111	Rossi	Mario	20,7
222	Neri	Paolo	24,5
333	Rossi	Maria	25,8
444	Pinco	Palla	19,6
77777	Bruno	Pasquale	26
88888	Pinco	Pietro	26