

20. SELECT (7)

SELECT statement (7)

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

1

20. SELECT (7)

Sample database

```
CREATE TABLE Department(
  name      VARCHAR2(50),
  code      CHAR(5),
  total_staff_number NUMBER(2)      NOT NULL,
  chair     VARCHAR2(50),
  budget    NUMBER(9,1)             NULL,
  CONSTRAINT dept_pkey PRIMARY KEY(name),
  CONSTRAINT dept_ckey1 UNIQUE(code),
  CONSTRAINT dept_ckey2 UNIQUE(chair),
  CONSTRAINT dept_check1
    CHECK (total_staff_number BETWEEN 1 AND 50) );
```

```
CREATE TABLE Course(
  c#        CHAR(7),
  title     VARCHAR2(200)          NOT NULL,
  credits   NUMBER(1)              NOT NULL,
  offered_by VARCHAR2(50)          NULL,
  CONSTRAINT course_pkey PRIMARY KEY(c#),
  CONSTRAINT course_check1
    CHECK (credits IN (6, 12) ),
  CONSTRAINT course_fkey1 FOREIGN KEY(offered_by)
    REFERENCES Department(name) );
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

2

20. SELECT (7)

Queries with ANY clause

Find the titles of all courses such that their credits are greater than credits of any database course

Course

c# | title | credits | offered by

```
SELECT title
FROM Course
WHERE credits >
```

ANY

```
( SELECT credits
  FROM Course
  WHERE title LIKE '%database%');
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

3

20. SELECT (7)

Computational model (queries with ANY clause)

```
SELECT <ATTRIBUTES>
FROM <TABLE_1>
WHERE ATTRIBUTE_1 > ANY
  (SELECT <ATTRIBUTES_1>
   FROM <TABLE_2>
   WHERE <CONDITION_2>);
```

```
forall rows t in <TABLE_2>
  if evaluate(<CONDITION_2>, t) then
    TEMP ← append(t.<ATTRIBUTES_2>)
  endif;
endforall;
forall rows s in <TABLE_1>
  if there exists t in TEMP such that s.ATTRIBUTE_1 > t then
    output(s.<ATTRIBUTES>);
  endif;
endforall;
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

4

20. SELECT (7)

Queries with ALL clause

Find the titles of all courses such that their credits are greater than credits of all (every) database courses

Course

c# | title | credits | offered by

```
SELECT title
FROM Course
WHERE credits >
```

ALL

```
( SELECT credits
  FROM Course
  WHERE title LIKE '%database%');
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

5

20. SELECT (7)

Computational model (queries with ALL clause)

```
SELECT <ATTRIBUTES>
FROM <TABLE_1>
WHERE ATTRIBUTE_1 > ANY
  (SELECT <ATTRIBUTES_1>
   FROM <TABLE_2>
   WHERE <CONDITION_2>);
```

```
forall rows t in <TABLE_2>
  if evaluate(<CONDITION_2>, t) then
    TEMP ← append(t.<ATTRIBUTES_2>)
  endif;
endforall;
forall rows s in <TABLE_1>
  if for each t in TEMP (s.ATTRIBUTE_1 > t) then
    output(s.<ATTRIBUTES>);
  endif;
endforall;
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

6

20. SELECT (7)

Queries with multiple column conditions

Find the titles of all courses such that their credits are the same as credits of CSCI235 course and they are offered by the same department as CSCI235 course

```
SELECT title
FROM Course
WHERE (credits, offered_by) =
      ( SELECT credits, offered_by
        FROM Course
        WHERE c# = 'CSCI235' )
AND c# <> 'CSCI235';
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

7

20. SELECT (7)

Nested queries revisited

Find the titles of all courses such that their credits are the same as credits of any database course and they are offered by the same department as any database course

```
SELECT title
FROM Course
WHERE credits IN
      ( SELECT credits
        FROM Course
        WHERE title LIKE '%database%' )
AND offered_by IN
      ( SELECT offered_by
        FROM Course
        WHERE title LIKE '%databases%' );
```

This query is NOT equivalent to a query with multiple column condition !!!

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

8

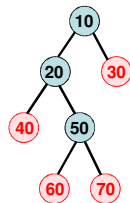
20. SELECT (7)

NULLs returned by subqueries

Employee Find all employees who are not managers

e# | name | manager#

e#	name	manager#
10	John	NULL
20	Peter	10
30	Mary	10
40	Mike	20
50	Kate	20
60	Greg	50
70	Phil	50



© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

9

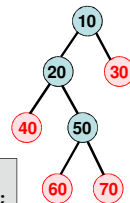
20. SELECT (7)

NULLs returned by subqueries

Employee Find all employees who are not managers

e# | name | manager#

```
SELECT *
FROM Employee
WHERE e# NOT IN
      ( SELECT manager#
        FROM Employee );
```



© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

10

20. SELECT (7)

NULLs returned by subqueries

Find all employees who are not managers

```
SELECT *
FROM Employee
WHERE e# NOT IN
      ( SELECT manager#
        FROM Employee );
```

No rows selected !!!

NULL value means "any number" !!!

manager#

NULL
10
10
20
20
50
50

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

11

20. SELECT (7)

NULLs returned by subqueries

Find all employees who are not managers

```
SELECT *
FROM Employee
WHERE e# NOT IN
      ( SELECT manager#
        FROM Employee
        WHERE manager# IS NOT NULL );
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

12

20. SELECT (7)

Queries with three-valued logic

NOT	TRUE	FALSE	UNKNOWN
	FALSE	TRUE	UNKNOWN

OR	TRUE	FALSE	UNKNOWN
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	UNKNOWN
UNKNOWN	TRUE	UNKNOWN	UNKNOWN

AND	TRUE	FALSE	UNKNOWN
TRUE	TRUE	FALSE	UNKNOWN
FALSE	FALSE	FALSE	FALSE
UNKNOWN	UNKNOWN	FALSE	UNKNOWN

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

13

20. SELECT (7)

Queries with three-valued logic

Employee

e# | name | manager#

e#	name	manager#
10	John	NULL
20	Peter	10
30	Mary	10
40	Mike	20
50	Kate	20
60	Greg	50
70	Phil	50

```
SELECT e#
FROM Employee
WHERE manager# = 10;
```

e#
--
20
30

```
SELECT e#
FROM Employee
WHERE manager# <> 10;
```

e#
--
40
50
60
70

Negation of UNKNOWN is equal to UNKNOWN !!!

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

14

20. SELECT (7)

NVL row function

Find the average number of employees per department

Department NULL → 0

name | code | total staff number | chair | budget

```
SELECT AVG( NVL(total_staff_number, 0) )
FROM Department;
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

15

20. SELECT (7)

COALESCE function

Find the average number of employees per department

Department NULL → 0

name | code | total staff number | chair | budget

```
SELECT AVG( COALESCE(total_staff_number, 0) )
FROM Department;
```

```
COALESCE(exp1, exp2, ..., expn)
Return the first non-NULL expression.
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

16

20. SELECT (7)

CASE statement

```
SELECT ename, salary,
CASE
  WHEN salary < 100 THEN 0
  WHEN salary < 200 THEN 0.2
  WHEN salary < 300 THEN 0.4
  WHEN salary < 400 THEN 0.5
  ELSE 0.7
END TAX_RATE
FROM Employee;
```

```
SELECT AVG(
CASE
  WHEN total_staff_number IS NULL THEN 0
  ELSE total_staff_number
END )
FROM Department;
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

17

20. SELECT (7)

How to implement complex queries ?

Find the titles of all courses such that their credits are the same as credits of CSCI235 course and they are offered by the same department as CSCI235 course

```
SELECT title
FROM ( SELECT credits, offered_by
      FROM Course
      WHERE c# = 'CSCI235'
      ) C235 JOIN Course
      ON C235.credits = Course.credits AND
      C235.offered_by = Course.offered_by
```

© Janusz R. Getta

CSCI235/MCS9235 Databases, SCSSE, Spring 2009

18

20. SELECT (7)

How to implement complex queries ?

Find the names of all departments that offer more than 1 course

```
SELECT Grouped.offered_by
FROM ( SELECT offered_by, count(*) total
      FROM Course
      GROUP BY offered_by
      ) Grouped
WHERE Grouped.total > 1;
```

20. SELECT (7)

How to implement complex queries ?

Find the names of all departments that have budget higher than 10000 and such that offer at least one 12 credit points course

```
SELECT D1K.name
FROM ( SELECT name
      FROM Department
      WHERE budget > 10000 ) D1K JOIN
( SELECT offered_by
  FROM Course
  WHERE credits >= 12 ) C12
ON D1K.name = C12.offered_by;
```

20. SELECT (7)

References

Elmasri R., Navathe S. B., *Database Systems*, chapter 6

Ramakrishnan R., Gehrke J., *Database Management Systems*, chapter 5.4.3, 5.6,

<https://sai.uow.edu.au/oradocs/>
SQL Reference, **SELECT** statement