SELECT Statement (4)

Dr Janusz R. Getta

School of Computing and Information Technology - University of Wollongong

SELECT statement (4)

Outline

Outer join queries

Left outer join queries

Right outer join queries

Full outer join queries

Grouping revisited

Outer join queries

Sample database

```
CREATE TABLE DEPARTMENT(                                    CREATE TABLE statement
 name               VARCHAR(50)        NOT NULL,
 code               CHAR(5)            NOT NULL,
 total_staff_number DECIMAL(2)         NOT NULL,
 chair              VARCHAR(50)            NULL,
 budget             DECIMAL(9,1)       NOT 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(                                        CREATE TABLE statement
 cnum               CHAR(7)            NOT NULL,
 title              VARCHAR(200)       NOT NULL,
 credits            DECIMAL(2)         NOT NULL,
 offered_by         VARCHAR(50)            NULL,
  CONSTRAINT course_pkey PRIMARY KEY(cnum),
  CONSTRAINT course_check1 CHECK (credits IN (6, 12)),
  CONSTRAINT course_fkey1 FOREIGN KEY(offered_by)
                    REFERENCES DEPARTMENT(name) ON DELETE CASCADE );
```

Outer join queries

Consider the following query: find the names of departments together with the titles of all courses offered by each department

```
SELECT name, title                          SELECT statement with JOIN operation
FROM DEPARTMENT JOIN COURSE
          ON name = offered_by;
```

What about the departments that offer no courses and what about the courses not assigned to any department ?

JOIN operation eliminates from both arguments the rows that cannot be joined with any row from the other argument

Sometimes, we would like to include into an answer the rows that cannot be joined

SELECT statement (4)

Outline

Outer join queries

Left outer join queries

Right outer join queries

Full outer join queries

Grouping revisited

Left outer join queries

Consider an extended version of the previous query: Find the names of departments together with the titles of all courses offered by each department and include the names of departments that offer no courses at all

SELECT statement with LEFT OUTER JOIN operation

```
SELECT name, title
FROM DEPARTMENT LEFT OUTER JOIN COURSE
              ON name = offered_by;
```

An operation of `LEFT OUTER JOIN` includes into the results all rows from the "left" argument of the operation

If a row from the "left" argument of the operation cannot be joined with any row from the "right" argument of the operation then it is extended with `NULL`s and it is appended to the result

Left outer join queries

What are the outcomes of the following `SELECT` statement ?

```
SELECT name, title
FROM DEPARTMENT LEFT OUTER JOIN COURSE
                ON name = offered_by;
```

Relational tables

name	code	t_s_n	chair	budget
Mathematics	MATH	7	Peter	200.0
Physics	PHY	5	Albert	100.1
Biology	BIOL	0	Mary	0.0

cnum	title	credits	offered_by
MATH101	Calculus	6	Mathematics
PHYS101	Mechanics	6	Physics
MATH300	Topology	12	NULL

The results of `LEFT OUTER JOIN` operation

Results of LEFT OUTER JOIN

name	code	t_s_n	chair	budget	cnum	title	credits	offered_by
Mathematics	MATH	7	Peter	200.0	MATH101	Calculus	6	Mathematics
Physics	PHY	5	Albert	100.1	PHYS101	Mechanics	6	Physics
Biology	BIOL	0	Mary	0.0	NULL	NULL	NULL	NULL

Left outer join queries

What are the outcomes of the following `SELECT` statement ?

```
SELECT name, title
FROM DEPARTMENT LEFT OUTER JOIN COURSE
              ON name = offered_by;
```

The final results of `PROJECTION` on the columns `name` and `title`

```
+------------+-----------+
| name       | title     |
+------------+-----------+
| Mathematics | Calculus |
| Physics    | Mechanics |
| Biology    | NULL      |
+------------+-----------+
```

SELECT statement (4)

Outline

Outer join queries

Left outer join queries

Right outer join queries

Full outer join queries

Grouping revisited

Right outer join queries

Consider the following query: find the names of departments together with the titles of all courses offered by each department and include the titles of all courses do not assigned to any department

```
                              SELECT statement with RIGHT OUTER JOIN operation
SELECT name, title
FROM DEPARTMENT RIGHT OUTER JOIN COURSE
ON name = offered_by;
```
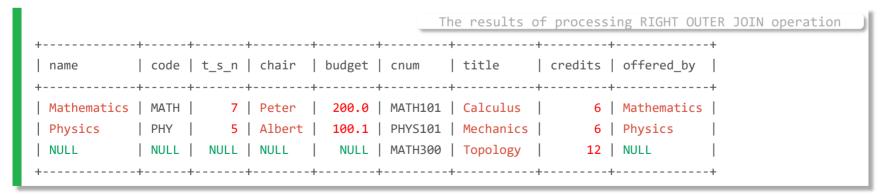
An operation of RIGHT OUTER JOIN includes into the results all rows from the "right" argument of the operation

If a row from the "right" argument of the operation cannot be joined with any row from the "left" argument of the operation then it is extended with NULLs and it is appended to the result

Right outer join queries

What are the outcomes of the following `SELECT` statement ?

SELECT statement with RIGHT OUTER JOIN operation

```
SELECT name, title
FROM DEPARTMENT RIGHT OUTER JOIN COURSE
            ON name = offered_by;
```

The releational tables

name	code	t_s_n	chair	budget
Mathematics	MATH	7	Peter	200.0
Physics	PHY	5	Albert	100.1
Biology	BIOL	0	Mary	0.0

cnum	title	credits	offered_by
MATH101	Calculus	6	Mathematics
PHYS101	Mechanics	6	Physics
MATH300	Topology	12	NULL

The results of `RIGHT OUTER JOIN` operation

The results of processing RIGHT OUTER JOIN operation

name	code	t_s_n	chair	budget	cnum	title	credits	offered_by
Mathematics	MATH	7	Peter	200.0	MATH101	Calculus	6	Mathematics
Physics	PHY	5	Albert	100.1	PHYS101	Mechanics	6	Physics
NULL	NULL	NULL	NULL	NULL	MATH300	Topology	12	NULL

Right outer join queries

What are the outcomes of the following `SELECT` statement ?

```
SELECT name, title
FROM DEPARTMENT RIGHT OUTER JOIN COURSE
               ON name = offered_by;
```

The final results of `PROJECTION` on the columns `name` and `title`

```
+------------+-----------+
| name       | title     |
+------------+-----------+
| Mathematics | Calculus  |
| NULL        | Topology  |
| Physics     | Mechanics |
+------------+-----------+
```

SELECT statement (4)

Outline

Outer join queries

Left outer join queries

Right outer join queries

Full outer join queries

Grouping revisited

Full outer join queries

Full outer join SQL query is equivalent to a union of the results from left outer join and right outer join

Consider the following query: find the names of departments together with the titles of all courses offered by each department, and ...

... include the names of departments that offer no courses and ...

... include the titles of courses not offered by any department

SELECT statement with FULL OUTER JOIN operation

```
SELECT name, title
FROM DEPARTMENT FULL OUTER JOIN COURSE
              ON name = offered_by;
```

Full outer join queries

MySQL does not support full outer join

Full outer join can be simulated in MySQL by `UNION` of `SELECT` statements that implement left outer join and right outer join

SELECT statement with LEFT OUTER JOIN and RIGHT OUTER JOIN operation equivalent to FULL OUTER JOIN operation

```
SELECT name, title
FROM DEPARTMENT LEFT OUTER JOIN COURSE
              ON name = offered_by
UNION
SELECT name, title
FROM DEPARTMENT RIGHT OUTER JOIN COURSE
              ON name = offered_by;
```

SELECT statement (4)

Outline

Outer join queries

Left outer join queries

Right outer join queries

Full outer join queries

Grouping revisited

Grouping revisited

Do you remember the following query: Find the names of all departments together with the total number of all courses offered by each department

SELECT statement with GROUP BY clause and aggregation function COUNT

```
SELECT offered_by, COUNT(*)
FROM COURSE
GROUP BY offered_by;
```

What about the departments that offer no courses ?

Find the names of all departments together with the total number of all courses offered by each department and list the names of departments that offer no courses with a value 0

SELECT statement with LEFT OUTER JOIN operation, GROUP BY clause, and aggregation function COUNT

```
SELECT name, count(title)
FROM DEPARTMENT LEFT OUTER JOIN COURSE
          ON DEPARTMENT.name = COURSE.offered_by
GROUP BY name;
```

Grouping revisited

Find the names of all departments together with the total number of all courses offered by each department and list the names of departments that offer no courses with a value 0

```
                  SELECT statement with LEFT OUTER JOIN operation, GROUP BY clause, and aggregation function COUNT

SELECT name, count(title)
FROM DEPARTMENT LEFT OUTER JOIN COURSE
                ON DEPARTMENT.name = COURSE.offered_by

GROUP BY name;
```

Note, that:

- we must use a relational table DEPARTMENT to get the names of all departments

- we must use LEFT OUTER JOIN (or RIGHT OUTER join if a name of relational table DEPARTMENT is on the right hand side of OUTER JOIN operation) to include the names of departments that offer no courses

- we must count the values in a column title with COUNT(title) in the results of LEFT OUTER JOIN (and not the rows with COUNT(*))

References

T. Connoly, C. Begg, Database Systems, A Practical Approach to Design, Implementation, and Management, Chapters 6.3.7 Multi-table Queries, Pearson Education Ltd, 2015

D. Darmawikarta, SQL for MySQL A Beginner's Tutorial, Chapter 6, pages 62 - 63 Brainy Software Inc. First Edition: June 2014

How to ... ? Cookbook, How to implement queries in SQL ? (Part 2) Recipe 6.2 How to implement outer joins ?