MySQL and SQL Part-II
More Querying Different Views

Amitabh Trehan
amitabhtrehan@softhome.net

September 12, 2003
1. Joins

Combining two tables:

types of Joins:

- Equijoin
- Natural Join
- Self Join
- Outer Join
Give the names of those students who have scored a 100 in any subject.
mysql> select distinct name from student s, Exam x 
where s.roll_no = x.roll_no and marks=100;
+---------------+
| name          |
| Harshad Mehta |
+---------------+
1 row in set (0.00 sec)
List Students by name, Subjects (by description) and their marks in every subject.
```sql
mysql> select name, course_name, marks from student s, Exam x, course c
    where s.roll_no=x.roll_no and
    x.course_code = c.course_code;

+---------------+-----------------+-------+
| name          | course_name     | marks |
| Bimal Jalan   | Business Ethics | 75    |
| Harshad Mehta | Business Ethics | 100   |
| Harshad Mehta | Share Swindling | 30    |
| Harshad Mehta | Cyber Fraud     | 100   |
| Sachin        | Business Ethics | 90    |
| Sachin        | Cyber Fraud     | 55    |
+---------------+-----------------+-------+
6 rows in set (0.00 sec)
```
List Roll Numbers having average > 50.
mysql> select roll_no from Exam
   ->  group by roll_no having avg(marks) > 50;

+------------+
| roll_no |
+------------+
| 2001MBA001 |
| 2001MBA007 |
| 2001MCA011 |
+------------+
3 rows in set (0.05 sec)
Name the Students having average > 50.
mysql> select name from student s, Exam x
    -> where s.roll_no=x.roll_no
    -> group by s.roll_no having avg(marks) > 50;

+---------------+
| name          |
|---------------+
| Bimal Jalan   |
| Harshad Mehta |
| Sachin        |
+---------------+
3 rows in set (0.04 sec)
Tell the courses along with slots for those students who do not have a phone.
mysql> select s.name, c.course_name, c.teach_slot from student s, Exam x, course c
   -> where s.roll_no=x.roll_no and
   -> c.course_code=x.course_code
   -> and phone_no is NULL;
+-------------+-----------------+------------+
| name | course_name | teach_slot |
|---------------------+---------------------|
| Bimal Jalan | Business Ethics | C |
+-------------+-----------------+------------+
1 row in set (0.00 sec)
Give roll_no of students who have got the same marks in any subject with the course and marks.
mysql> select x1.roll_no, x2.roll_no, x1.course_code, x1.marks, x2.course_code, x2.marks from Exam x1, Exam x2 where x1.marks=x2.marks;

+------------+------------+-------------+
| roll_no | roll_no | course_code |
|-----------+-----------+-------------|
+------------+------------+-------------+
| 2001MBA001 | 2001MBA001 | MBA001001 |
| 75 | MBA001001 | 75 |
| 2001MBA007 | 2001MBA007 | MBA001001 |
| 100 | MBA001001 | 100 |
| 2001MBA007 | 2001MBA007 | MBA007009 |
| 100 | MBA001001 | 100 |
| 2001MCA011 | 2001MCA011 | MBA001001 |
| 90 | MBA001001 | 90 |
| 2001MBA007 | 2001MBA007 | MBA007007 |
| 33 | MBA007007 | 33 |
| 2001MBA007 | 2001MBA007 | MBA001001 |
| 100 | MBA007009 | 100 |
| 2001MBA007 | 2001MBA007 | MBA007009 |
| 100 | MBA007009 | 100 |
| 2001MCA011 | 2001MCA011 | MBA007009 |
| 55 | MBA007009 | 55 |
+-----------------------------+

8 rows in set (0.00 sec)

Is this result proper?
The previous result was improper. Pair distinct roll numbers together.
mysql> select x1.roll_no, x2.roll_no, x1.course_code, x1.marks, x2.course_code, x2.marks from Exam x1, Exam x2 where x1.roll_no < x2.roll_no and x1.marks=x2.marks;

Empty set (0.00 sec)
Give the name and city of people who live in the same city.
mysql> select s1.name, s1.city, s2.name, s2.city
    from student s1, student s2 where
    s1.name < s2.name and s1.city=s2.city;

+---------------+--------+--------+--------+
| name | city | name | city |
+---------------+--------+--------+--------+
| Harshad Mehta | Mumbai | Sachin | Mumbai |
+---------------+--------+--------+--------+
1 row in set (0.03 sec)
Give Names of people pairwise and the subjects for those who have got more than 90 marks in any subject.
mysql> select s1.name, s2.name, x1.marks, x1.course_code, x2.marks, x2.course_code from student s1, student s2, Exam x1, Exam x2 -> where x1.marks>=90 and x2.marks>=90 -> and x1.roll_no < x2.roll_no -> and s1.roll_no=x1.roll_no and -> s2.roll_no=x2.roll_no;

+---------------+--------+-------+
| name          | name   | marks |
|               | course_code | marks| course_code |
+---------------+--------+-------+
| Harshad Mehta | Sachin | 100   |
| MBA001001     | 100    | MBA001001 |
| Harshad Mehta | Sachin | 100   |
| MBA007009     | 90     | MBA001001 |
+---------------+--------+-------+
2 rows in set (0.00 sec)
Give Names of people who have got the same marks across any subject.
mysql> select s1.name, s2.name, x1.marks, x1.course_code, x2.course_code from student s1, student s2, Exam x1, Exam x2 where x1.marks=x2.marks and x1.roll_no < x2.roll_no and s1.roll_no=x1.roll_no and s2.roll_no=x2.roll_no;

Empty set (0.16 sec)
Let us add another record to the table Exam:

```sql
mysql> insert into Exam values
-> ("2001MBA001","MBA007007",90);
Query OK, 1 row affected (0.06 sec)
```

Now we run the query Again.
mysql> select s1.name, s2.name, x1.marks, x1.course_code, x2.course_code from student s1, student s2, Exam x1, Exam x2 where x1.marks= x2.marks and x1.roll_no < x2.roll_no and s1.roll_no=x1.roll_no and s2.roll_no=x2.roll_no;

+-------------+--------+-------+-------------+-------------+
| name | name | marks | course_code | course_code |
+-------------+--------+-------+-------------+-------------+
| Bimal Jalan | Sachin | 90 | MBA007007 | MBA001001 |
+-------------+--------+-------+-------------+-------------+
1 row in set (0.00 sec)
Using certain functions

Let us add another record:

```sql
mysql> insert into student values 
    -> ("2002MLA001","Saurav G",
    -> "Sahara India", "Calcutta",
    -> NULL,"1983-09-17");
Query OK, 1 row affected (0.03 sec)
```

Count the number of Teenagers in the University.
mysql> select count(*) from student
->     where year(curdate())-year(DOB) > 12
and
->     year(curdate())-year(DOB) < 20;

+----------+
| count(*) |
+----------+
|    1     |
+----------+
1 row in set (0.00 sec)

Is the answer correct?
No, Saurav G is 19 years old but the difference in years is 20.
So, How do we find out the Age? (in **MySQL**)

Hint: reduce 1 from age if the person’s birthday has still to come this year.
mysql> SELECT name, DOB, CURDATE(),
        -> (YEAR(CURDATE()) - YEAR(DOB))
        -> - (RIGHT(CURDATE(),5) < RIGHT(DOB,5))
        -> AS age
        -> FROM student;

+---------------+------------+------------+------+
| name          | DOB        | CURDATE()  | age  |
| Bimal Jalan   | 1960-01-05 | 2003-09-11 | 43   |
| Harshad Mehta | 1971-12-24 | 2003-09-11 | 31   |
| Sachin        | 1984-04-24 | 2003-09-11 | 19   |
| Saurav G      | 1983-09-17 | 2003-09-11 | 19   |
+---------------+------------+------------+------+
**Explanation:**

**YEAR()**  
Gives year part of a date

**RIGHT()**  
rightmost five characters  
here they are MM-DD part  
of the date.

**CURDATE()**  
Current Date.

**(RIGHT(CURDATE(),5)<RIGHT(DOB,5))**  
evaluates to 1 or 0  
thus making the proper adjustment.
So, now count the number of teenagers in the University.
mysql> select count(*) from student
    where (YEAR(CURDATE())-YEAR(DOB))
    - (RIGHT(CURDATE(),5)<RIGHT(DOB,5)) > 12
    and (YEAR(CURDATE())-YEAR(DOB))
    - (RIGHT(CURDATE(),5)<RIGHT(DOB,5)) < 20;

+----------+
| count(*) |
+----------+
| 2        |
+----------+
1 row in set (0.00 sec)
Outer Joins

- Required when certain tuples are required even when they do not satisfy the join condition.
- Three types:
  - Left Outer Join
  - Right Outer Join
  - Full Outer Join
Give the Names of all the students and their marks in subjects in which they have appeared?
in MySQL and SQL2

```sql
select name, course_code, marks from student left join Exam on student.roll_no = Exam.roll_no;
```

OR

```sql
select name, course_code, marks from student left join Exam using (roll_no);
```
in Oracle — using (+)

select name, course_code, marks from student, Exam where student.roll_no(+) = Exam.roll_no;

in Sybase — using *

select name, course_code, marks from student, Exam where student.roll_no *= Exam.roll_no;
Join Syntax

table_reference, table_reference

A CROSS JOIN is also used:
table_reference [CROSS] JOIN table_reference

table_reference INNER JOIN table_reference [join_condition]

table_reference STRAIGHT_JOIN table_reference

table_reference LEFT [OUTER] JOIN table_reference [join_condition]

table_reference LEFT [OUTER] JOIN table_reference [join_condition]

table_reference NATURAL [LEFT [OUTER]] JOIN table_reference

table_reference RIGHT [OUTER] JOIN table_reference [join_condition]

table_reference RIGHT [OUTER] JOIN table_reference [join_condition]

A NATURAL JOIN is also used:
table_reference NATURAL [RIGHT [OUTER]] JOIN table_reference

2. SubQueries

It is also possible to combine multiple Queries together. This is called SubQueries.

Subqueries may be of the following types:

- **Nested SubQueries**: Queries Nested (Queries within Queries within Queries within ..), higher level queries accessing results from lower level queries; Very Fast (Faster thann Joins).

- **Correlated Subqueries**: Inner queries depend on outer queries for values; Slow.
Subqueries can also be classified on basis of operators used to introduce them:

- Unmodified comparison operators, queries return single value.
- Using *in*, *any* or *all* with/without *not* keywords.
- Using *exists* keyword.
Unmodified operators

• Query introduced using comparison operators =, !=, >, >=, <, <=, !>, !<.

• The subquery must return a single value only.

Which people have got more marks than roll_no 2001MBA001 in the subject MBA001001?

select roll_no from Exam where course_code="MBA001001" and marks >
( select marks from Exam where roll_no="2001MBA001" and course_code="MBA001001" );
Using in, any, all and not

- Subquery which is accessed with above keywords can return a list of zero or more values

Which people study a subject which roll_no 2001MBA001 also studies?

```sql
select roll_no from Exam where course_code in
(select course_code from Exam where roll_no="2001MBA001");
```
Which people do not share any subject with roll_no 2001MBA001?
select roll_no from Exam where course_code != any
( select course_code from Exam where roll_no="2001MBA001" );
Correlated Subqueries

- Inner Subquery depends on outer for values.

In which subjects have students scored more than their average?
select roll_no, course_code from Exam M1 where marks > (select avg(marks) from Exam M2 where M1.roll_no = M2.roll_no);
Using **exists**

- Subquery functions as an existence test. Tests for existence of rows returned by inner query.

Which Students have not given any Exam?
select roll_no from student where not exists
  (select * from Exam where roll_no=student.roll_no);
Restrictions on SubQueries

Some restrictions on Subqueries may be:

- The *select-list* of an inner query using comparison operator can have only one expression or column name and must return a single value only.

- The *distinct* and *group by* clauses cannot be included in the same query.

- Subqueries cannot include the *order by* or the *into* keyword.
Some Sub-Warm-up exercises

How many people have better average than Bimal Jalan?
select count(*) from Exam group by roll_no having avg(marks) > (select avg(marks) from Exam x, student s where s.roll_no=x.roll_no and name="Bimal Jalan");
Which students are older in age than Sachin?
select name from student
where DOB < 
(select DOB from student
where name="Sachin");
Which people (in which subject?) have scored better marks than Bimal Jalan in any subject?
select roll_no, marks from Exam x1 
where marks > 
(select marks from Exam x2, student s 
where x2.roll_no=s.roll_no 
and name="Bimal Jalan"
and x1.course_code = 
x2.course_code);
3. Your Point of View

- **View** is a virtual table. It has no data of its own but can be operated upon just like any other table. It shows the data of tables on which it is based called **Base Tables**.

- **Data according to User**: Different views of same table according to need.

- **Data Security**: Different views of the data to different people based on access rights.

- **Logical Data Independence**: Suppose Procedures were written for a table which was later split. These can be preserved by recreating the old table as a new view.

- Will be available in **MySQL** from version 5.1.
4. SQL Views

Create View <View Name> as <Table Expression>

Create a View StudentMarks
Which has Name, Course and their Marks in the Course
create view StudentMarks as
select name, course_name, marks from student s,
Exam x, Course c
where s.roll_no = x.roll_no
and x.course_code = c.course_code