Database worksheets combined

Applied Databases - Week 2

1. Get school.sql from Moodle and import it into MySQL.

- Right Click and select “save as”
- mysql -u root -p <school.sql

2. What is the maximum length of data that can be inserted into the Name attribute of the subject table?

```
mysql> describe subject;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| Name          | varchar(15) | NO   | PRI | NULL    |       |
| Teacher       | varchar(20) | YES  |     | NULL    |       |
| OnLeavingCert | tinyint(1)  | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

3. What is the Primary Key of the teacher table?

```
mysql> describe teacher;
+------------+---------------+------+-----+---------+----------------+
| Field      | Type          | Null | Key | Default | Extra          |
+------------+---------------+------+-----+---------+----------------+
| tid        | int(11)       | NO   | PRI | NULL    | auto_increment |
| Name       | varchar(20)   | YES  |     | NULL    |                |
| level      | enum('J','L') | YES  |     | NULL    |                |
| experience | int(11)       | YES  |     | NULL    |                |
| dob        | date          | YES  |     | NULL    |                |
+------------+---------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)
```

4. What is the Primary Key of the subject table?

```
mysql> describe subject;
+---------------+-------------+------+-----+---------+-------+
| Field         | Type        | Null | Key | Default | Extra |
+---------------+-------------+------+-----+---------+-------+
| Name          | varchar(15) | NO   | PRI | NULL    |       |
| Teacher       | varchar(20) | YES  |     | NULL    |       |
| OnLeavingCert | tinyint(1)  | YES  |     | NULL    |       |
+---------------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

5. Show all data contained in the subject table.

```
mysql> select * from subject;
+-----------+--------------+---------------+
| Name      | Teacher      | OnLeavingCert |
+-----------+--------------+---------------+
| Biology   | Mr. Pasteur  |             1 |
| Colouring | Mr. Picasso  |             0 |
| English   | Mr. Kavanagh |             1 |
| French    | Ms. Dubois   |             1 |
| Maths     | Mr. Hawking  |             1 |
| Religion  | Fr. Lynch    |             1 |
| Spelling  | Ms.
Smith   |             0 |
+-----------+--------------+---------------+
7 rows in set (0.00 sec)
```

6. Show all names of all subjects that are on the leaving cert.

```
mysql> select * from subject
    -> where OnLeavingCert = '1';
+----------+--------------+---------------+
| Name     | Teacher      | OnLeavingCert |
+----------+--------------+---------------+
| Biology  | Mr. Pasteur  |             1 |
| English  | Mr. Kavanagh |             1 |
| French   | Ms. Dubois   |             1 |
| Maths    | Mr. Hawking  |             1 |
| Religion | Fr. Lynch    |             1 |
+----------+--------------+---------------+
5 rows in set (0.00 sec)
```

7. Show all name and experience of all teachers who are qualified to teach to Leaving Cert.

```
mysql> select * from teacher
    -> where level = 'L';
+-----+-------------+-------+------------+------------+
| tid | Name        | level | experience | dob        |
+-----+-------------+-------+------------+------------+
|   1 | Mr. Pasteur | L     |         15 | 1960-02-02 |
|   2 | Ms. Dubois  | L     |         22 | 1967-09-02 |
|   4 | Mr. Hawking | L     |         40 | 1951-02-19 |
|   7 | Fr. Lynch   | L     |         55 | 1939-03-31 |
+-----+-------------+-------+------------+------------+
4 rows in set (0.00 sec)
```

8. Show all details of all subjects who are taught by teachers whose title is not “Mr.”

```
mysql> select * from teacher
    -> where name NOT LIKE 'Mr.%'
    -> ;
+-----+------------+-------+------------+------------+
| tid | Name       | level | experience | dob        |
+-----+------------+-------+------------+------------+
```

```
|   2 | Ms. Dubois | L     |          22 | 1967-09-02 |
|   3 | Ms. Smith  | J     |           4 | 1980-03-23 |
|   7 | Fr. Lynch  | L     |          55 | 1939-03-31 |
+-----+------------+-------+-------------+------------+
3 rows in set (0.00 sec)
```

9. Show all details of all teachers who were born in January, February or March, and who can teach as far as Junior Cert only.

```
mysql> select * from teacher
    -> where (month(dob) != 1
    -> or month(dob) != 2
    -> or month(dob) != 3)
    -> and level = 'j';
+-----+---------------+-------+------------+------------+
| tid | Name          | level | experience | dob        |
+-----+---------------+-------+------------+------------+
|   3 | Ms. Smith     | J     |          4 | 1980-03-23 |
|   5 | Mr.
Kavanagh | J     |         50 | 1949-11-01 |
|   6 | Mr. Picasso   | J     |         42 | 1939-03-30 |
+-----+---------------+-------+------------+------------+
3 rows in set (0.00 sec)
```

10. Show all unique month names that teachers were born in.

```
mysql> SELECT distinct month(dob) mon FROM school.teacher;
+------+
| mon  |
+------+
|    2 |
|    9 |
|    3 |
|   11 |
+------+
4 rows in set (0.00 sec)
```

11. Show all details of all teachers, sorted by first by experience, then level.

```
mysql> SELECT * FROM school.teacher
    -> order by experience, level;
+-----+---------------+-------+------------+------------+
| tid | Name          | level | experience | dob        |
+-----+---------------+-------+------------+------------+
|   3 | Ms. Smith     | J     |          4 | 1980-03-23 |
|   1 | Mr. Pasteur   | L     |         15 | 1960-02-02 |
|   2 | Ms. Dubois    | L     |         22 | 1967-09-02 |
|   4 | Mr. Hawking   | L     |         40 | 1951-02-19 |
|   6 | Mr. Picasso   | J     |         42 | 1939-03-30 |
|   5 | Mr. Kavanagh  | J     |         50 | 1949-11-01 |
|   7 | Fr. Lynch     | L     |         55 | 1939-03-31 |
+-----+---------------+-------+------------+------------+
7 rows in set (0.00 sec)
```

12. Show all details of all subjects whose 3rd or 4th letter is “l”. Sort them by name.

```
mysql> SELECT * FROM school.subject
    -> where name like '__i%'
    -> or name like '___i%'
    -> order by name;
+----------+--------------+---------------+
| Name     | Teacher      | OnLeavingCert |
+----------+--------------+---------------+
| English  | Mr. Kavanagh |             1 |
| Religion | Fr. Lynch    |             1 |
+----------+--------------+---------------+
2 rows in set (0.00 sec)
```

13. Show the name of all teachers who have 10, 15, 20, 25, 30, 35, 40, 45, 50, 55 or 60 years experience. Sort from youngest to oldest.

```
mysql> SELECT * FROM school.teacher
    -> where experience in (10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60);
+-----+---------------+-------+------------+------------+
| tid | Name          | level | experience | dob        |
+-----+---------------+-------+------------+------------+
|   1 | Mr. Pasteur   | L     |         15 | 1960-02-02 |
|   4 | Mr. Hawking   | L     |         40 | 1951-02-19 |
|   5 | Mr. Kavanagh  | J     |         50 | 1949-11-01 |
|   7 | Fr. Lynch     | L     |         55 | 1939-03-31 |
+-----+---------------+-------+------------+------------+
4 rows in set (0.00 sec)
```

Applied Databases - Week 3

1. Get employees.sql from Moodle and import it into MySQL.

```
MySQL - u root -p <employees.sql
```

2. Print out the emp_no, first_name and a capitalised version of the employees last_name, using the same column names that are in the table for the first 10 employees returned from the database.

```
mysql> SELECT emp_no, first_name, ucase(last_name) 'last_name'
    -> FROM employees.employees
    -> limit 10;
+--------+------------+-----------+
| emp_no | first_name | last_name |
+--------+------------+-----------+
|  10001 | Georgi     | FACELLO   |
```

```
|  10002 | Bezalel    | SIMMEL    |
|  10003 | Parto      | BAMFORD   |
|  10004 | Chirstian  | KOBLICK   |
|  10005 | Kyoichi    | MALINIAK  |
|  10006 | Anneke     | PREUSIG   |
|  10007 | Tzvetan    | ZIELINSKI |
|  10008 | Saniya     | KALLOUFI  |
|  10009 | Sumant     | PEAC      |
|  10010 | Duangkaew  | PIVETEAU  |
+--------+------------+-----------+
10 rows in set (0.00 sec)
```

3. Sort the employees table based on: • The length of last_name • Alphabetical order of last_name • The length of first_name • Alphabetical order of first_name

```
SELECT * FROM employees.employees
# order by length(last_name)
# order by last_name
# order by first_name
order by length(first_name);

SELECT * FROM employees.employees
order by length(last_name), last_name, length(first_name), first_name;
```

4. Show all details of the first 10 employees returned from the database and an extra column called Initials that shows the employee's initials.

```
mysql> SELECT emp_no,birth_date,first_name,last_name,gender,hire_date,
concat(left(first_name,1), left(last_name,1)) 'Initials'
    -> FROM employees.employees
    -> limit 10;
+--------+------------+------------+-----------+--------+------------+----------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  | Initials |
+--------+------------+------------+-----------+--------+------------+----------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 | GF       |
|  10002 | 1964-06-02 | Bezalel    | Simmel    | F      | 1985-11-21 | BS       |
|  10003 | 1959-12-03 | Parto      | Bamford   | M      | 1986-08-28 | PB       |
|  10004 | 1954-05-01 | Chirstian  | Koblick   | M      | 1986-12-01 | CK       |
|  10005 | 1955-01-21 | Kyoichi    | Maliniak  | M      | 1989-09-12 | KM       |
|  10006 | 1953-04-20 | Anneke     | Preusig   | F      | 1989-06-02 | AP       |
|  10007 | 1957-05-23 | Tzvetan    | Zielinski | F      | 1989-02-10 | TZ       |
|  10008 | 1958-02-19 | Saniya     | Kalloufi  | M      | 1994-09-15 | SK       |
|  10009 | 1952-04-19 | Sumant     | Peac      | F      | 1985-02-18 | SP       |
|  10010 | 1963-06-01 | Duangkaew  | Piveteau  | F      | 1989-08-24 | DP       |
+--------+------------+------------+-----------+--------+------------+----------+
10 rows in set (0.00 sec)
```

5. Show all details of all Females born in the 1950s and hired between September 1st 1988 and February 28th 1991.

```
mysql> SELECT * FROM employees.employees
    -> where gender = 'F'
    -> and (year(birth_date)>='1950' and year(birth_date)<='1959')
    -> and (hire_date>='1988-09-01' and hire_date<='1991-02-28')
    -> ;
```

```
+--------+------------+------------+------------+--------+------------+
| emp_no | birth_date | first_name | last_name  | gender | hire_date  |
+--------+------------+------------+------------+--------+------------+
|  10006 | 1953-04-20 | Anneke     | Preusig    | F      | 1989-06-02 |
|  10007 | 1957-05-23 | Tzvetan    | Zielinski  | F      | 1989-02-10 |
|  10011 | 1953-11-07 | Mary       | Sluis      | F      | 1990-01-22 |
|  10023 | 1953-09-29 | Bojan      | Montemayor | F      | 1989-12-17 |
|  10041 | 1959-08-27 | Uri        | Lenart     | F      | 1989-11-12 |
|  10063 | 1952-08-06 | Gino       | Leonhardt  | F      | 1989-04-08 |
|  10074 | 1955-08-28 | Mokhtar    | Bernatsky  | F      | 1990-08-13 |
|  10088 | 1954-02-25 | Jungsoon   | Syrzycki   | F      | 1988-09-02 |
|  10099 | 1956-05-25 | Valter     | Sullins    | F      | 1988-10-18 |
+--------+------------+------------+------------+--------+------------+
9 rows in set (0.03 sec)
```

6. Show the average salary from the salaries table formatted to two decimal places. E.g. 12345.6789 should become 12,345.68.

```
mysql> SELECT round(avg(salary),2) 'avg_salary'
FROM employees.salaries;

+------------+
| avg_salary |
+------------+
|   64417.59 |
+------------+
1 row in set (0.00 sec)
```

7. Show the emp_no and average salary for each employee formatted to two decimal places.

```
mysql> SELECT emp_no, round(avg(salary),2) avg_sal
    -> FROM employees.salaries
    -> group by emp_no
    -> limit 10;
+--------+----------+
| emp_no | avg_sal  |
+--------+----------+
|  10001 | 75388.94 |
|  10002 | 68854.50 |
|  10003 | 43030.29 |
|  10004 | 56512.25 |
|  10005 | 87275.77 |
|  10006 | 50514.92 |
|  10007 | 70826.71 |
|  10008 | 49307.67 |
|  10009 | 78284.56 |
|  10010 | 76723.00 |
+--------+----------+
10 rows in set (0.01 sec)
```

8. Show the emp_no and maximum salary for each employee formatted to two decimal places.

```
mysql> SELECT emp_no, cast(max(salary) as decimal(10,2)) max_sal
    -> FROM employees.salaries
```

```
    -> group by emp_no
    -> limit 10;
+--------+----------+
| emp_no | max_sal  |
+--------+----------+
|  10001 | 88958.00 |
|  10002 | 72527.00 |
|  10003 | 43699.00 |
|  10004 | 74057.00 |
|  10005 | 94692.00 |
|  10006 | 60098.00 |
|  10007 | 88070.00 |
|  10008 | 52668.00 |
|  10009 | 94443.00 |
|  10010 | 80324.00 |
+--------+----------+
10 rows in set (0.00 sec)
```

9. Show the emp_no and average salary formatted to two decimal places for the following employee numbers: 10001, 10021, 10033 and 10087. But only include in the average calculation salaries greater than 80,000.

```
mysql> SELECT emp_no,
    ->        avg(salary)
    -> FROM employees.salaries
    -> where emp_no in (10001, 10021, 10033, 10087)
    -> and salary > 80000
    -> group by emp_no;
+--------+-------------+
| emp_no | avg(salary) |
+--------+-------------+
|  10001 |  83745.5714 |
|  10021 |  83232.0000 |
|  10087 |  99015.2500 |
+--------+-------------+
3 rows in set (0.01 sec)
```

note: forgot to round to 2 decimal points

10. Show the emp_no and average salary rounded to the nearest whole number only for average salaries greater than 90,000.

```
mysql> SELECT emp_no, round(avg(salary)) as avg_sal
    -> FROM employees.salaries
    -> group by emp_no
    -> having avg_sal > 90000;
+--------+---------+
| emp_no | avg_sal |
+--------+---------+
|  10024 |   90572 |
|  10068 |  101224 |
|  10087 |   99015 |
+--------+---------+
3 rows in set (0.00 sec)
```

11. Show the following details, in the following order, for the first 15 employees, in emp_no order: ID, Title, Name, Surname, Gender. Title should be “Mr.” if the employee is Male, and “Ms.” if the employee is female.

```
mysql> SELECT emp_no as ID,
    ->     if(gender = 'M', 'Mr.','Ms.') as Gender,
    ->         first_name as Name,
    ->         last_name as Surname,
    ->         gender as Gender
    -> FROM employees.employees
    -> order by emp_no
    -> limit 15;
+-------+--------+-----------+-----------+--------+
| ID    | Gender | Name      | Surname   | Gender |
+-------+--------+-----------+-----------+--------+
| 10001 | Mr.    | Georgi    | Facello   | M      |
| 10002 | Ms.    | Bezalel   | Simmel    | F      |
| 10003 | Mr.    | Parto     | Bamford   | M      |
| 10004 | Mr.    | Chirstian | Koblick   | M      |
| 10005 | Mr.    | Kyoichi   | Maliniak  | M      |
| 10006 | Ms.    | Anneke    | Preusig   | F      |
| 10007 | Ms.    | Tzvetan   | Zielinski | F      |
| 10008 | Mr.    | Saniya    | Kalloufi  | M      |
| 10009 | Ms.    | Sumant    | Peac      | F      |
| 10010 | Ms.    | Duangkaew | Piveteau  | F      |
| 10011 | Ms.    | Mary      | Sluis     | F      |
| 10012 | Mr.    | Patricio  | Bridgland | M      |
| 10013 | Mr.    | Eberhardt | Terkki    | M      |
| 10014 | Mr.    | Berni     | Genin     | M      |
| 10015 | Mr.    | Guoxiang  | Nooteboom | M      |
+-------+--------+-----------+-----------+--------+
15 rows in set (0.00 sec)
```

12. Show the following details emp_no, the maximum salary for each employee, and the tax bracket the employee’s maximum salary is in (Tax Bracket).

Tax brackets are defined as follows:

Max Salary	Tax Bracket
Under 40,000	30%
Under 60,000	40%
Under 80,000	50%
Over 80,000	60%

```
mysql> SELECT emp_no as 'Employee Number',
    ->     max(salary) as 'Max Salary',
    ->         CASE
    ->           when max(salary) < 40000 then '30%'
    ->           when max(salary) < 60000 then '40%'
    ->           when max(salary) < 80000 then '50%'
    ->           else '60%'
    ->         END as 'Tax Bracket'
    -> FROM employees.salaries
    -> group by emp_no
    -> limit 15;
+-----------------+------------+-------------+
| Employee Number | Max Salary | Tax Bracket |
```

```
+-----------------+------------+-------------+
|           10001 |      88958 | 60%         |
|           10002 |      72527 | 50%         |
|           10003 |      43699 | 40%         |
|           10004 |      74057 | 50%         |
|           10005 |      94692 | 60%         |
|           10006 |      60098 | 50%         |
|           10007 |      88070 | 60%         |
|           10008 |      52668 | 40%         |
|           10009 |      94443 | 60%         |
|           10010 |      80324 | 60%         |
|           10011 |      56753 | 40%         |
|           10012 |      54794 | 40%         |
|           10013 |      68901 | 50%         |
|           10014 |      60598 | 50%         |
|           10015 |      40000 | 40%         |
+-----------------+------------+-------------+
15 rows in set (0.00 sec)
```

13. Show all details from the salaries table as well as a column entitled “Time” which states “Under 1 yr” if the employee has been on a particular salary for less than 365 days, otherwise states “Over 1 yr”.

```
SELECT *, if(datediff(to_date,from_date)<356, "under 1 year", "over 1 year") as Time
FROM employees.salaries
# having Time like '%under%'
;
```

Output

```
+--------+--------+------------+------------+-------------+
| emp_no | salary | from_date  | to_date    | Time        |
+--------+--------+------------+------------+-------------+
|  10001 |  60117 | 1986-06-26 | 1987-06-26 | over 1 year |
|  10001 |  62102 | 1987-06-26 | 1988-06-25 | over 1 year |
|  10001 |  66074 | 1988-06-25 | 1989-06-25 | over 1 year |
|  10001 |  66596 | 1989-06-25 | 1990-06-25 | over 1 year |
|  10001 |  66961 | 1990-06-25 | 1991-06-25 | over 1 year |
|  10001 |  71046 | 1991-06-25 | 1992-06-24 | over 1 year |
|  10001 |  74333 | 1992-06-24 | 1993-06-24 | over 1 year |
|  10001 |  75286 | 1993-06-24 | 1994-06-24 | over 1 year |
|  10001 |  75994 | 1994-06-24 | 1995-06-24 | over 1 year |
|  10001 |  76884 | 1995-06-24 | 1996-06-23 | over 1 year |
+--------+--------+------------+------------+-------------+
10 rows in set (0.00 sec)
```

14. Using a function show all columns from the employees table, and a column entitled “Age” which is the age the employee was when he or she was hired. The age should be rounded to 1 digit after the decimal place. For example, employee 10001 was 32.8 years old when he was hired.

HINT: Don’t for get to change the delimiter when writing the function and change it back to a semi-colon when the function is written.

Function code

```
CREATE FUNCTION `getage`(d1 date, d2 date) RETURNS float(5,1)
    DETERMINISTIC
BEGIN
   RETURN round(datediff(d2,d1)/365,1);
END
```

Query

```
SELECT *, getage(birth_date,hire_date) as Age
FROM employees.employees
limit 10;
```

Result

```
+--------+------------+------------+-----------+--------+------------+------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  | Age  |
+--------+------------+------------+-----------+--------+------------+------+
|  10001 | 1953-09-02 | Georgi     | Facello   | M      | 1986-06-26 | 32.8 |
|  10002 | 1964-06-02 | Bezalel    | Simmel    | F      | 1985-11-21 | 21.5 |
|  10003 | 1959-12-03 | Parto      | Bamford   | M      | 1986-08-28 | 26.8 |
|  10004 | 1954-05-01 | Chirstian  | Koblick   | M      | 1986-12-01 | 32.6 |
|  10005 | 1955-01-21 | Kyoichi    | Maliniak  | M      | 1989-09-12 | 34.7 |
|  10006 | 1953-04-20 | Anneke     | Preusig   | F      | 1989-06-02 | 36.1 |
|  10007 | 1957-05-23 | Tzvetan    | Zielinski | F      | 1989-02-10 | 31.7 |
|  10008 | 1958-02-19 | Saniya     | Kalloufi  | M      | 1994-09-15 | 36.6 |
|  10009 | 1952-04-19 | Sumant     | Peac      | F      | 1985-02-18 | 32.9 |
|  10010 | 1963-06-01 | Duangkaew  | Piveteau  | F      | 1989-08-24 | 26.2 |
+--------+------------+------------+-----------+--------+------------+------+
10 rows in set (0.00 sec)
```

15. Write a procedure that takes two parameters, one representing a year and the other a month. The procedure should return all employees hired in specified year and month.

Procedure

```
CREATE DEFINER=`jattie`@`%` PROCEDURE `hires`(y integer, m integer)
    DETERMINISTIC
BEGIN
  SELECT *
  FROM employees.employees
  where year(hire_date) = y
  and month(hire_date) = m;
END
```

Query

```
call hires(1988,9);
```

Result

```
+--------+------------+------------+-----------+--------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  |
+--------+------------+------------+-----------+--------+------------+
|  10034 | 1962-12-29 | Bader      | Swan      | M      | 1988-09-21 |
|  10035 | 1953-02-08 | Alain      | Chappelet | M      | 1988-09-05 |
|  10088 | 1954-02-25 | Jungsoon   | Syrzycki  | F      | 1988-09-02 |
+--------+------------+------------+-----------+--------+------------+
3 rows in set (0.01 sec)
```

16. Rewrite the above procedure so that if the month parameter is NULL the procedure returns all employees hired in the specified year. If the month is not NULL, the procedure works as it did previously.

HINT: To call a procedure with a NULL value for month (assuming in this case month is the second parameter) procedure_name(1985, NULL). To check if a parameter, e.g. m, is NULL say **IF M IS NULL THEN** To check if a parameter, e.g. m, is not NULL say **IF M IS NOT NULL THEN.**

Procedure code

```
CREATE DEFINER=`jattie`@`%` PROCEDURE `hires_2`(y integer, m integer)
    DETERMINISTIC
BEGIN
    if m is null then
        select * from employees
        where year(hire_date) = y;
    else
        select * from employees
        where year(hire_date) = y
        and month(hire_date) = m;
    end if;
END
```

Query

```
call hires_2(1988, null);
```

Result

```
mysql> call hires_2(1988, null);
+--------+------------+------------+-----------+--------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date  |
+--------+------------+------------+-----------+--------+------------+
|  10021 | 1960-02-20 | Ramzi      | Erde      | M      | 1988-02-10 |
|  10034 | 1962-12-29 | Bader      | Swan      | M      | 1988-09-21 |
|  10035 | 1953-02-08 | Alain      | Chappelet | M      | 1988-09-05 |
|  10039 | 1959-10-01 | Alejandro  | Brender   | M      | 1988-01-19 |
|  10052 | 1961-02-26 | Heping     | Nitsch    | M      | 1988-05-21 |
|  10065 | 1963-04-14 | Satosi     | Awdeh     | M      | 1988-05-18 |
|  10072 | 1952-05-15 | Hironoby   | Sidou     | F      | 1988-07-21 |
|  10088 | 1954-02-25 | Jungsoon   | Syrzycki  | F      | 1988-09-02 |
|  10099 | 1956-05-25 | Valter     | Sullins   | F      | 1988-10-18 |
```

```
+--------+------------+------------+-----------+--------+------------+
9 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Applied Databases - Week 4

1. Get garage.sql from Moodle and import it into MySQL.

```
mysql -u root -p <garage.sql
```

2. How are the tables in the database related?

```
show tables;
describe manufacturer;
show create table manufacturer;
show create table vehicle;
```

The *manufacturer* table contains a manufacturer code, name and description of the manufacturer of the vehicle.

The *vehicle* table contains the vehicle deteails with a reference to the manufacturer code described as a *foreign key constraint*. This foreign key links the two table with the manu_code present in both tables.

3. Show the manu_code, manu_name and the first 10 characters of the manu_details followed by three dots (...) for each manufacturer.

```
mysql> SELECT manu_code,
              manu_name,
              concat(left(manu_details,10),' ...') as 'manu_details'
              FROM garage.manufacturer;

+-----------+----------------+----------------+
| manu_code | manu_name      | manu_details   |
+-----------+----------------+----------------+
| FOR       | Ford           | The Ford M ... |
| GM        | General Motors | General Mo ... |
| NIS       | Nissan         | Nissan Mot ... |
| TOY       | Toyota         | Toyota Mot ... |
| VOL       | Volkswagen     | Volkswagen ... |
+-----------+----------------+----------------+
5 rows in set (0.00 sec)
```

4. Show the average length of the manu_name (displayed as “Length”) with 0 characters after the decimal point. HINT: Functions needed are avg(), length() and format().

```
mysql> SELECT format(avg(length(manu_name)),0)
    ->        as Length
    ->        FROM garage.manufacturer;

+--------+
```

```
| Length |
+--------+
| 8      |
+--------+
1 row in set (0.00 sec)
```

5. Show all details of all vehicles plus an extra column called “cost” which has the value 1.45 if the fuel is petrol otherwise has the value 1.30.

```
mysql> SELECT *, if(fuel='petrol', '1.45', '1.30') as cost
    -> FROM garage.vehicle;
+--------------+-----------+---------+----------+--------+--------+------+
| reg          | manu_code | mileage | price    | colour | fuel   | cost |
+--------------+-----------+---------+----------+--------+--------+------+
| 2003-LM-201  | TOY       |  170000 |  3500.50 | Red    | petrol | 1.45 |
| 2009-RN-12   | FOR       |   98242 |  2500.00 | Red    | petrol | 1.45 |
| 2010-G-13345 | TOY       |   50000 |  8599.00 | Silver | petrol | 1.45 |
| 2011-G-995   | FOR       |   33500 |  8500.00 | Blue   | petrol | 1.45 |
| 2011-WH-2121 | FOR       |   55998 | 14000.00 | Black  | diesel | 1.30 |
| 2014-WH-2189 | FOR       |   12553 | 11000.00 | Blue   | diesel | 1.30 |
| 2016-D-12345 | TOY       |    3456 | 15000.00 | Red    | petrol | 1.45 |
+--------------+-----------+---------+----------+--------+--------+------+
7 rows in set (0.00 sec)
```

6. Show all the reg, manu_code and associated manu_name for each vehicle.

```
mysql> SELECT gv.reg, gv.manu_code, gm.manu_name FROM garage.vehicle gv
    -> left join garage.manufacturer gm
    -> on gv.manu_code=gm.manu_code
    -> ;
+--------------+-----------+-----------+
| reg          | manu_code | manu_name |
+--------------+-----------+-----------+
| 2009-RN-12   | FOR       | Ford      |
| 2011-G-995   | FOR       | Ford      |
| 2011-WH-2121 | FOR       | Ford      |
| 2014-WH-2189 | FOR       | Ford      |
| 2003-LM-201  | TOY       | Toyota    |
| 2010-G-13345 | TOY       | Toyota    |
| 2016-D-12345 | TOY       | Toyota    |
+--------------+-----------+-----------+
7 rows in set (0.00 sec)
```

7. Show the manu_code and manu_name as well as associated reg, for each manufacturer who has vehicles listed in the vehicle table.

```
mysql> SELECT gm.manu_code, gm.manu_name, gv.reg
    -> FROM garage.manufacturer as gm
    -> inner join garage.vehicle as gv
    -> on gm.manu_code=gv.manu_code;

+-----------+-----------+--------------+
| manu_code | manu_name | reg          |
+-----------+-----------+--------------+
| FOR       | Ford      | 2009-RN-12   |
| FOR       | Ford      | 2011-G-995   |
| FOR       | Ford      | 2011-WH-2121 |
```

```
| FOR       | Ford      | 2014-WH-2189 |
| TOY       | Toyota    | 2003-LM-201  |
| TOY       | Toyota    | 2010-G-13345 |
| TOY       | Toyota    | 2016-D-12345 |
+-----------+-----------+--------------+
7 rows in set (0.00 sec)
```

8. Show the manu_code and manu_name as well as associated reg, for all manufacturers and if they have vehicles listed in the vehicle table, show the reg of it.

```
mysql> SELECT gm.manu_code, gm.manu_name, gv.reg
    -> FROM garage.manufacturer as gm
    -> left join garage.vehicle as gv
    -> on gm.manu_code=gv.manu_code;
+-----------+----------------+--------------+
| manu_code | manu_name      | reg          |
+-----------+----------------+--------------+
| FOR       | Ford           | 2009-RN-12   |
| FOR       | Ford           | 2011-G-995   |
| FOR       | Ford           | 2011-WH-2121 |
| FOR       | Ford           | 2014-WH-2189 |
| GM        | General Motors | NULL         |
| NIS       | Nissan         | NULL         |
| TOY       | Toyota         | 2003-LM-201  |
| TOY       | Toyota         | 2010-G-13345 |
| TOY       | Toyota         | 2016-D-12345 |
| VOL       | Volkswagen     | NULL         |
+-----------+----------------+--------------+
10 rows in set (0.00 sec)
```

9. Write a stored procedure called price_less_than that takes one parameter of type decimal(8,2) which represents the price of a vehicle:

```
price_less_than(p decimal(8,2))
```

The procedure should then return the following details for all vehicles where the price of the vehicle is less than p sorted by ascending price:

- Reg
- Manu_code
- Manu_name
- Mileage
- Price

Procedure

```
CREATE PROCEDURE `price_less_than`(p decimal(8,2))
    DETERMINISTIC
BEGIN
    SELECT gv.reg, gv.manu_code, gm.manu_name, gv.mileage, gv.price
    FROM garage.vehicle gv
    left join garage.manufacturer gm
    on gv.manu_code=gm.manu_code
    where gv.price < p
    order by gv.price;
```

```
END
```

Testing Procedure

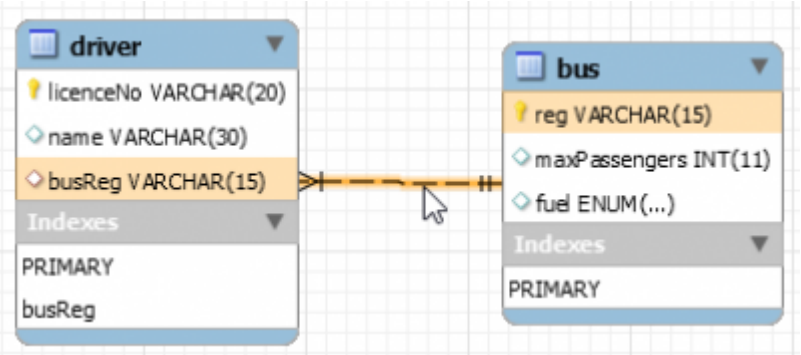
```
mysql> call price_less_than(15000);
+--------------+-----------+-----------+---------+----------+
| reg          | manu_code | manu_name | mileage | price    |
+--------------+-----------+-----------+---------+----------+
| 2009-RN-12   | FOR       | Ford      |   98242 |  2500.00 |
| 2003-LM-201  | TOY       | Toyota    |  170000 |  3500.50 |
| 2011-G-995   | FOR       | Ford      |   33500 |  8500.00 |
| 2010-G-13345 | TOY       | Toyota    |   50000 |  8599.00 |
| 2014-WH-2189 | FOR       | Ford      |   12553 | 11000.00 |
| 2011-WH-2121 | FOR       | Ford      |   55998 | 14000.00 |
+--------------+-----------+-----------+---------+----------+
6 rows in set (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Applied Databases - Week 5

1. Get bus.sql from Moodle and import it into MySQL.

```
MySQL -u root -p <bus.sql
```

2. How are the tables in the database related?




```
mysql> show create table driver;
+--------+-----------------------------------------------+
| Table  | Create Table                                  |
+--------+-----------------------------------------------+
| driver | CREATE TABLE `driver` (
  `licenceNo` varchar(20) NOT NULL,
  `name` varchar(30) DEFAULT NULL,
  `busReg` varchar(15) DEFAULT NULL,
  PRIMARY KEY (`licenceNo`),
  KEY `busReg` (`busReg`),
  CONSTRAINT `driver_ibfk_1`
  FOREIGN KEY (`busReg`) REFERENCES `bus` (`reg`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1 |
+--------+-----------------------------------------------+
1 row in set (0.00 sec)
```

The driver has a foreign key constraint busReg that requires a valid entry that already exists in the table bus.reg to be present. When the bus entry is deleted from thus bus table, the “ON DELET CASCADE” will also delete the driver associated with this bus entry.

3. Add the following drivers:

- “Mary”
- “Bob” – licence number “RN2423”
- “Sean” – licence number “FF88345” who drives bus “191-G-123”
- What happens and why?

```
mysql> insert into driver (name) values("Mary");
ERROR 1364 (HY000): Field 'licenceNo' doesn't have a default value
```

This insert fails because the licenseNo field is a primary key value and cannot be NULL and fails when no value is assigned during the insert.

```
mysql> insert into driver (licenceNo,name) values("RN2423","Bob");
Query OK, 1 row affected (0.01 sec)

mysql> select * from driver;
+-----------+------+------------+
| licenceNo | name | busReg     |
+-----------+------+------------+
| F2233     | Alan | 191-G-123  |
| L23423    | John | 12-G-1323  |
| RN2423    | Bob  | NULL       |
| X98983    | Tom  | 161-D-1323 |
+-----------+------+------------+
4 rows in set (0.00 sec)
```

The second query works since Bo is added with a licensee number and the minimum criteria is being met.

```
mysql> insert into driver (licenceNo,name,busReg) values("FF88345","Sean","191-G-123");
Query OK, 1 row affected (0.01 sec)

mysql> select * from driver;
+-----------+------+------------+
| licenceNo | name | busReg     |
+-----------+------+------------+
| F2233     | Alan | 191-G-123  |
| FF88345   | Sean | 191-G-123  |
| L23423    | John | 12-G-1323  |
| RN2423    | Bob  | NULL       |
| X98983    | Tom  | 161-D-1323 |
+-----------+------+------------+
5 rows in set (0.00 sec)
```

The last insert also works since all the fields are populated with valid values.

4. Add the following buses:

- “12-G-1323” that can hold up to 34 passengers and runs on “Diesel”
- “171-G-885” that can hold up to 84 passengers and runs on “Petrol”
- “191-D-45890” that can hold up to 120 passengers and runs on “Ethanol”
- What happens and why?

```
mysql> select * from bus;
+------------+---------------+----------+
| reg        | maxPassengers | fuel     |
```

```
+------------+--------------+----------+
| 12-G-1323  |           34 | Petrol   |
| 161-D-1323 |           80 | Diesel   |
| 162-D-3433 |          120 | Electric |
| 191-G-123  |           56 | Diesel   |
+------------+--------------+----------+
4 rows in set (0.00 sec)

mysql> insert into bus values("12-G-1323",34,"Diesel");
ERROR 1062 (23000): Duplicate entry '12-G-1323' for key 'PRIMARY'
```

The bus already exists in the table and the primary key constraint prevents duplication in this reg column.

```
mysql> insert into bus values("171-G885",84,"Petrol");
Query OK, 1 row affected (0.01 sec)

mysql> select * from bus;
+------------+--------------+----------+
| reg        | maxPassengers | fuel     |
+------------+--------------+----------+
| 12-G-1323  |           34 | Petrol   |
| 161-D-1323 |           80 | Diesel   |
| 162-D-3433 |          120 | Electric |
| 171-G885   |           84 | Petrol   |
| 191-G-123  |           56 | Diesel   |
+------------+--------------+----------+
5 rows in set (0.00 sec)
```

All the criteria entered mets the requirements and is successfully added to the table.

```
mysql> insert into bus values("191-D-45890",120,"Ethanol");
ERROR 1265 (01000): Data truncated for column 'fuel' at row 1
mysql> select * from bus;
+------------+--------------+----------+
| reg        | maxPassengers | fuel     |
+------------+--------------+----------+
| 12-G-1323  |           34 | Petrol   |
| 161-D-1323 |           80 | Diesel   |
| 162-D-3433 |          120 | Electric |
| 171-G885   |           84 | Petrol   |
| 191-G-123  |           56 | Diesel   |
+------------+--------------+----------+
5 rows in set (0.00 sec)
```

Ethanol is not in the enumeration list of fuels and is therefore rejected.

```
1. CREATE TABLE `bus` (
2.   `reg` varchar(15) NOT NULL,
3.   `maxPassengers` int(11) DEFAULT NULL,
4.   `fuel` enum('Diesel','Petrol','Electric') DEFAULT 'Diesel',
5.   PRIMARY KEY (`reg`)
6. ) ENGINE=InnoDB DEFAULT CHARSET=latin1 |
```

5. Update driver's licences that contain the letters "F" or "R" to have the letters "T-" before their current licence number.

```
Update driver
  set licenceNo = concat("T-", licenceNo)
  where licenceNo like "F%"
```

```
  or licenceNo like "R%";

select * from driver;
+-----------+------+------------+
| licenceNo | name | busReg     |
+-----------+------+------------+
| L23423    | John | 12-G-1323  |
| T-F2233   | Alan | 191-G-123  |
| T-FF88345 | Sean | 191-G-123  |
| T-RN2423  | Bob  | NULL       |
| X98983    | Tom  | 161-D-1323 |
+-----------+------+------------+
5 rows in set (0.00 sec)
```

6. Delete driver “Alan”.

What happens and why?

```
mysql> select * from driver;
+-----------+------+------------+
| licenceNo | name | busReg     |
+-----------+------+------------+
| L23423    | John | 12-G-1323  |
| T-F2233   | Alan | 191-G-123  |
| T-FF88345 | Sean | 191-G-123  |
| T-RN2423  | Bob  | NULL       |
| X98983    | Tom  | 161-D-1323 |
+-----------+------+------------+
5 rows in set (0.00 sec)

mysql> select * from bus;
+------------+---------------+----------+
| reg        | maxPassengers | fuel     |
+------------+---------------+----------+
| 12-G-1323  |            34 | Petrol   |
| 161-D-1323 |            80 | Diesel   |
| 162-D-3433 |           120 | Electric |
| 171-G885   |            84 | Petrol   |
| 191-G-123  |            56 | Diesel   |
+------------+---------------+----------+
5 rows in set (0.00 sec)

mysql> delete from driver
    -> where name = "Alan";
Query OK, 1 row affected (0.01 sec)

mysql> select * from driver;
+-----------+------+------------+
| licenceNo | name | busReg     |
+-----------+------+------------+
| L23423    | John | 12-G-1323  |
| T-FF88345 | Sean | 191-G-123  |
| T-RN2423  | Bob  | NULL       |
| X98983    | Tom  | 161-D-1323 |
+-----------+------+------------+
4 rows in set (0.00 sec)

mysql> select * from bus;
+------------+---------------+----------+
| reg        | maxPassengers | fuel     |
+------------+---------------+----------+
| 12-G-1323  |            34 | Petrol   |
```

```
| 161-D-1323 |            80 | Diesel   |
| 162-D-3433 |           120 | Electric |
| 171-G885   |            84 | Petrol   |
| 191-G-123  |            56 | Diesel   |
+------------+---------------+----------+
5 rows in set (0.00 sec)
```

Alan was deleted and nothing else happens. Assuming the question is asked to highlight the fworking of the foreign key constraint busReg and the "ON DELET CASCADE" clause.

So in this case nothing happens but if the bus assigned to Alan was deleted Alan would have disappeared too. Is this right?

7. Delete bus "161-d-1323".

What happens and why?

```
mysql> select * from bus;
+------------+---------------+----------+
| reg        | maxPassengers | fuel     |
+------------+---------------+----------+
| 12-G-1323  |            34 | Petrol   |
| 161-D-1323 |            80 | Diesel   |
| 162-D-3433 |           120 | Electric |
| 171-G885   |            84 | Petrol   |
| 191-G-123  |            56 | Diesel   |
+------------+---------------+----------+
5 rows in set (0.00 sec)

mysql> select * from driver;
+-----------+------+------------+
| licenceNo | name | busReg     |
+-----------+------+------------+
| L23423    | John | 12-G-1323  |
| T-FF88345 | Sean | 191-G-123  |
| T-RN2423  | Bob  | NULL       |
| X98983    | Tom  | 161-D-1323 |
+-----------+------+------------+
4 rows in set (0.00 sec)

mysql> delete from bus
    -> where reg = "161-D-1323";
Query OK, 1 row affected (0.01 sec)

mysql> select * from bus;
+------------+---------------+----------+
| reg        | maxPassengers | fuel     |
+------------+---------------+----------+
| 12-G-1323  |            34 | Petrol   |
| 162-D-3433 |           120 | Electric |
| 171-G885   |            84 | Petrol   |
| 191-G-123  |            56 | Diesel   |
+------------+---------------+----------+
4 rows in set (0.00 sec)

mysql> select * from driver;
+-----------+------+-----------+
| licenceNo | name | busReg    |
```

```
+-----------+------+-----------+
| L23423    | John | 12-G-1323 |
| T-FF88345 | Sean | 191-G-123 |
| T-RN2423  | Bob  | NULL      |
+-----------+------+-----------+
3 rows in set (0.00 sec)
```

So bus “161-D-1323” and driver “Tom” was deleted because of the “ON DELETE CASCADE” foreign key reference applied on the driver table.

8. Get bus2.sql from Moodle and import it into MySQL.

To use this database type use bus2;

```
MySQL -u root -p <bus2.mysql
```

9. Delete bus “161-d-1323”.

What happens and why?

```
mysql> use bus2;
Database changed
mysql> select * from driver;
+-----------+------+------------+
| licenceNo | name | busReg     |
+-----------+------+------------+
| F2233     | Alan | 191-G-123  |
| L23423    | John | 12-G-1323  |
| X98983    | Tom  | 161-D-1323 |
+-----------+------+------------+
3 rows in set (0.00 sec)

mysql> select * from bus;
+------------+--------------+----------+
| reg        | maxPassengers | fuel     |
+------------+--------------+----------+
| 12-G-1323  |           34 | Petrol   |
| 161-D-1323 |           80 | Diesel   |
| 162-D-3433 |          120 | Electric |
| 191-G-123  |           56 | Diesel   |
+------------+--------------+----------+
4 rows in set (0.00 sec)

mysql> delete from bus
    -> where reg = "161-D-1323";
Query OK, 1 row affected (0.01 sec)

mysql> select * from driver;
+-----------+------+-----------+
| licenceNo | name | busReg    |
+-----------+------+-----------+
| F2233     | Alan | 191-G-123 |
| L23423    | John | 12-G-1323 |
| X98983    | Tom  | NULL      |
+-----------+------+-----------+
3 rows in set (0.00 sec)
```

```
mysql> select * from bus;
+-----------+--------------+----------+
| reg       | maxPassengers | fuel     |
+-----------+--------------+----------+
| 12-G-1323 |            34 | Petrol   |
| 162-D-3433 |           120 | Electric |
| 191-G-123 |            56 | Diesel   |
+-----------+--------------+----------+
3 rows in set (0.00 sec)
```

The behavior changed by setting driver Tom busReg foreign key to NULL. This happened because of the amended foreign key clause “ON DELETE SET NULL”

```
mysql> show create table driver;
CREATE TABLE `driver` (
  `licenceNo` varchar(20) NOT NULL,
  `name` varchar(30) DEFAULT NULL,
  `busReg` varchar(15) DEFAULT NULL,
  PRIMARY KEY (`licenceNo`),
  KEY `busReg` (`busReg`),
  CONSTRAINT `driver_ibfk_1` FOREIGN KEY (`busReg`)
  REFERENCES `bus` (`reg`) ON DELETE SET NULL
```

10. Get employees2.sql from Moodle and import it into MySQL.

To use this database type use employees2;

```
MySQL -u root -p <employees2.sql
use employees2;
Database changed
```

11. Show the emp_no, first_name and last_name of employees born in the average year.

The average year should be rounded down to the nearest whole number.

For example,

- 1949.1 becomes 1949.
- 1949.9 becomes 1949.
- 1949.0 becomes 1949.

```
mysql> SELECT emp_no, first_name, last_name, year(birth_date) as bd
    -> FROM employees2.employees
    -> where year(birth_date) in (
    ->   SELECT truncate(avg(year(birth_date)),0) FROM employees2.employees
    ->   )
    -> ;
+--------+------------+--------------+------+
| emp_no | first_name | last_name    | bd   |
+--------+------------+--------------+------+
|  10007 | Tzvetan    | Zielinski    | 1957 |
|  10045 | Moss       | Shanbhogue   | 1957 |
|  10054 | Mayumi     | Schueller    | 1957 |
```

```
|  10080 | Premal     | Baek         | 1957 |
|  10094 | Arumugam   | Ossenbruggen | 1957 |
+--------+------------+--------------+------+
5 rows in set (0.00 sec)
```

or with rounding instead ....

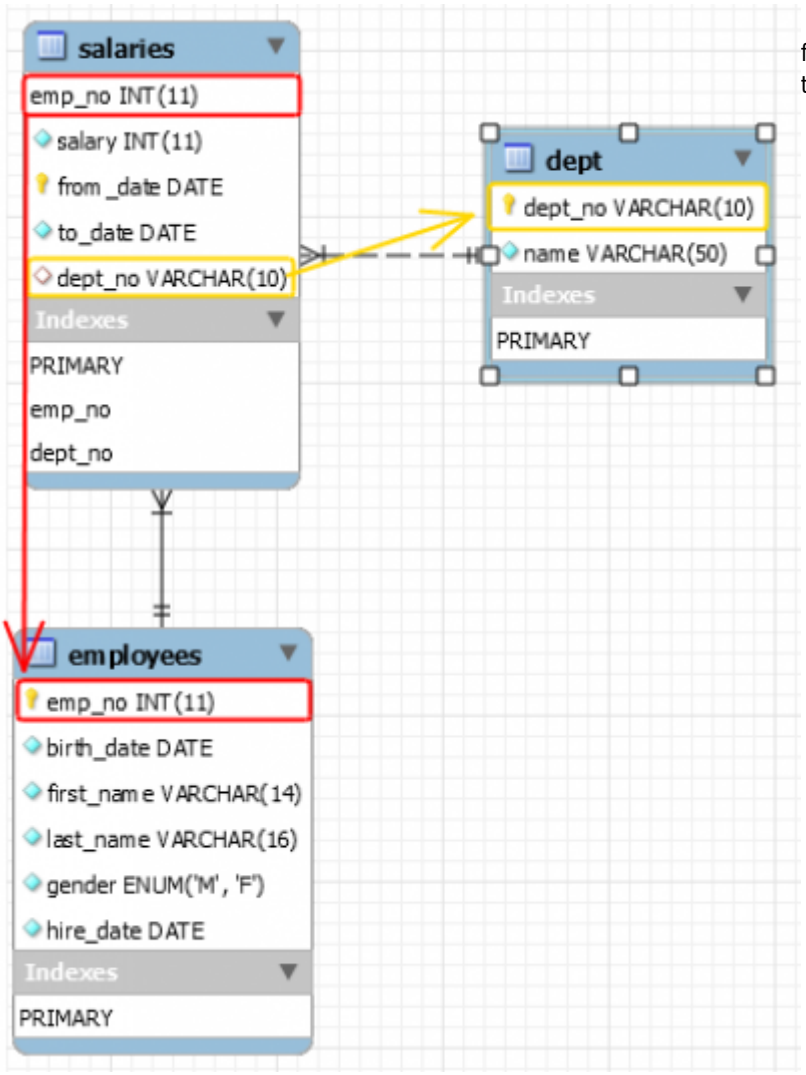
```
mysql> SELECT emp_no, first_name, last_name, year(birth_date) as bd
    -> FROM employees2.employees
    -> where year(birth_date) in (
    ->   SELECT round(avg(year(birth_date)),0) FROM employees2.employees
    ->   )
    -> ;
+--------+------------+-----------+------+
| emp_no | first_name | last_name | bd   |
+--------+------------+-----------+------+
|  10008 | Saniya     | Kalloufi  | 1958 |
|  10017 | Cristinel  | Bouloucos | 1958 |
|  10024 | Suzette    | Pettey    | 1958 |
|  10025 | Prasadram  | Heyers    | 1958 |
|  10030 | Elvis      | Demeyer   | 1958 |
|  10050 | Yinghua    | Dredge    | 1958 |
|  10071 | Hisao      | Lipner    | 1958 |
+--------+------------+-----------+------+
7 rows in set (0.00 sec)
```

12. Show the emp_no, first_name, last_name and name of the department each employee is in.

First determine the relationships....

```
mysql> show tables;

+----------------------+
| Tables_in_employees2 |
+----------------------+
| dept                 |
| employees            |
| salaries             |
+----------------------+
3 rows in set (0.00 sec)

mysql> show create table dept;

CREATE TABLE `dept` (
  `dept_no` varchar(10) NOT NULL,
  `name` varchar(50) NOT NULL,
  PRIMARY KEY (`dept_no`)
)

mysql> show create table employees;
CREATE TABLE `employees` (
  `emp_no` int(11) NOT NULL,
  `birth_date` date NOT NULL,
  `first_name` varchar(14) NOT NULL,
  `last_name` varchar(16) NOT NULL,
  `gender` enum('M','F') NOT NULL,
  `hire_date` date NOT NULL,
  PRIMARY KEY (`emp_no`)
)
```

```
1 row in set (0.00 sec)

mysql> show create table salaries;

CREATE TABLE `salaries` (
  `emp_no` int(11) NOT NULL,
  `salary` int(11) NOT NULL,
  `from_date` date NOT NULL,
  `to_date` date NOT NULL,
  `dept_no` varchar(10) DEFAULT NULL,
  PRIMARY KEY (`emp_no`,`from_date`),
  KEY `emp_no` (`emp_no`),
  KEY `dept_no` (`dept_no`),
  CONSTRAINT `salaries_ibfk_1` FOREIGN KEY (`emp_no`) REFERENCES `employees` (`emp_no`) ON
DELETE CASCADE,
  CONSTRAINT `salaries_ibfk_2` FOREIGN KEY (`dept_no`) REFERENCES `dept` (`dept_no`) ON DELETE
CASCADE
)1 row in set (0.00 sec)
```



So the master table is salaries and contains foreign key to the employees and departments tables.

Construct the query

```
SELECT DISTINCT(ee.emp_no), ee.first_name, ee.last_name, ed.name AS department
FROM employees2.employees AS ee
LEFT JOIN employees2.salaries AS es
ON ee.emp_no=es.emp_no
LEFT JOIN employees2.dept AS ed
```

```
ON es.dept_no=ed.dept_no
```

Concatenated Query Result

emp_no	first_name	last_name	department
10001	Georgi	Facello	Human Resources
10002	Bezalel	Simmel	Human Resources
10003	Parto	Bamford	Human Resources
...			
10028	Domenick	Tempesti	Human Resources
10029	Otmar	Herbst	Human Resources
10030	Elvis	Demeyer	Human Resources
10031	Karsten	Joslin	Research & Development
10032	Jeong	Reistad	Research & Development
10033	Arif	Merlo	Research & Development
...			
10068	Charlene	Brattka	Research & Development
10069	Margareta	Bierman	Research & Development
10070	Reuven	Garigliano	Research & Development
10071	Hisao	Lipner	Sales
10072	Hironoby	Sidou	Sales
10073	Shir	McClurg	Sales
...			
10098	Sreekrishna	Servieres	Sales
10099	Valter	Sullins	Sales
10100	Hironobu	Haraldson	Sales

100 rows in set (0.00 sec)

Last update: **2019/03/04 12:14**