DATA ANALYTICS REFERENCE DOCUMENT	
Document Title:	Appliead Databases week 3
Document No.:	1549571853
Author(s):	Gerhard van der Linde
Contributor(s):	

REVISION HISTORY

Revision	Details of Modification(s)	Reason for modification	Date	By
0	Draft release	Document description here	2019/02/07 20:37	Gerhard van der Linde

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)
```