Relational Databases with MySQL Week 4 Coding Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: Using a text editor of your choice, write the queries that accomplishes the objectives listed below. Take screenshots of the queries and results and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document to the repository. Additionally, push an .sql file with all your queries to the same repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

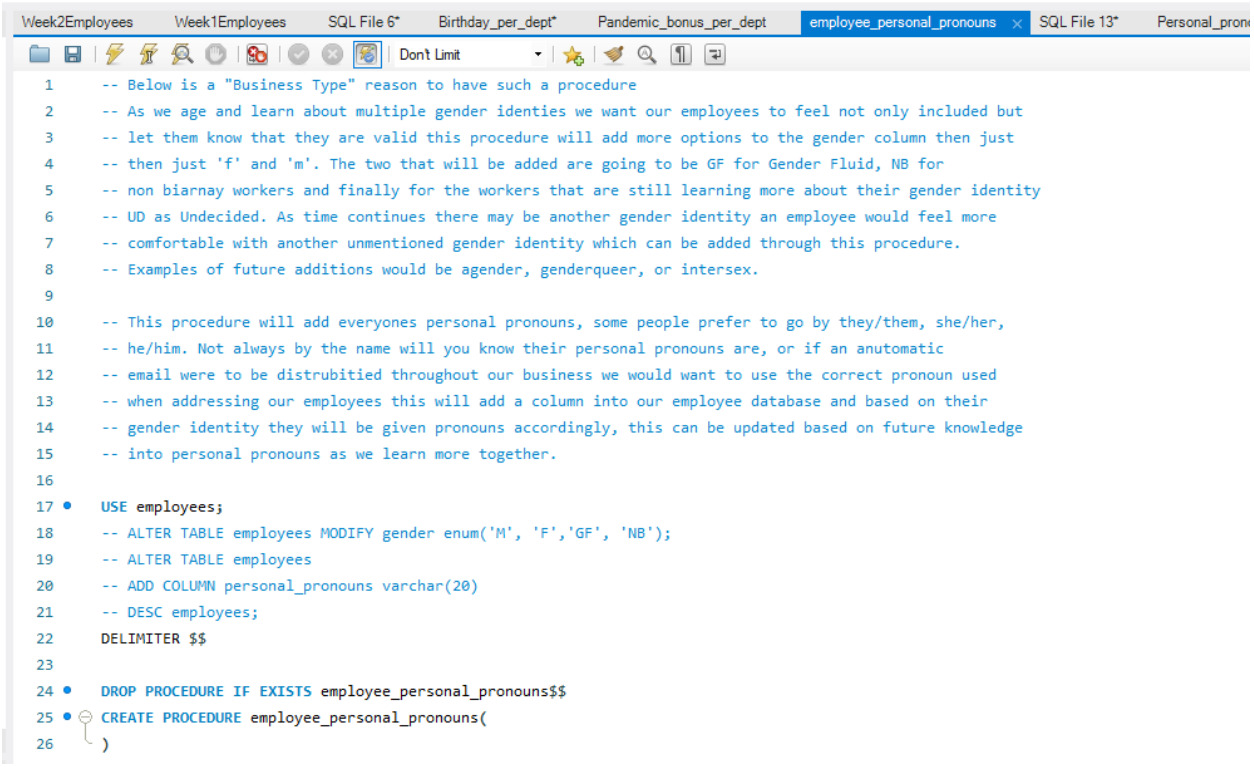
Coding Steps:

Write 5 stored procedures for the employees database.

Write a description of what each stored procedure does and how to use it.

Procedures should use constructs you learned about from your research assignment and be more than just queries.

Screenshots:




```sql
-- Below is a "Business Type" reason to have such a procedure
-- As we age and learn about multiple gender identies we want our employees to feel not only included but
-- let them know that they are valid this procedure will add more options to the gender column then just
-- then just 'f' and 'm'. The two that will be added are going to be GF for Gender Fluid, NB for
-- non biarnay workers and finally for the workers that are still learning more about their gender identity
-- UD as Undecided. As time continues there may be another gender identity an employee would feel more
-- comfortable with another unmentioned gender identity which can be added through this procedure.
-- Examples of future additions would be agender, genderqueer, or intersex.

-- This procedure will add everyones personal pronouns, some people prefer to go by they/them, she/her,
-- he/him. Not always by the name will you know their personal pronouns are, or if an anutomatic
-- email were to be distrubitied throughout our business we would want to use the correct pronoun used
-- when addressing our employees this will add a column into our employee database and based on their
-- gender identity they will be given pronouns accordingly, this can be updated based on future knowledge
-- into personal pronouns as we learn more together.

USE employees;
-- ALTER TABLE employees MODIFY gender enum('M', 'F','GF', 'NB');
-- ALTER TABLE employees
-- ADD COLUMN personal_pronouns varchar(20)
-- DESC employees;
DELIMITER $$

DROP PROCEDURE IF EXISTS employee_personal_pronouns$$
CREATE PROCEDURE employee_personal_pronouns(
)
```

```sql
USE employees;
-- ALTER TABLE employees MODIFY gender enum('M', 'F','GF', 'NB');
-- ALTER TABLE employees
-- ADD COLUMN personal_pronouns varchar(20)
-- DESC employees;
DELIMITER $$

DROP PROCEDURE IF EXISTS employee_personal_pronouns$$
CREATE PROCEDURE employee_personal_pronouns(
)

BEGIN
    SELECT concat(last_name, " ", first_name ) AS "Employee", gender AS "Gender Identity", personal_pronouns AS "Personal Pronouns" FROM employees e
    ORDER BY last_name, first_name;
    UPDATE employees set personal_pronouns = 'She/Her'
    WHERE gender = 'F';
    UPDATE employees set personal_pronouns = 'He/Him'
    WHERE gender = 'M';
    UPDATE employees set personal_pronouns = 'They/Them'
    WHERE gender = 'GF';
    UPDATE employees set personal_pronouns = 'They/Them'
    WHERE gender = 'NB';
    END $$
DELIMITER ;
CALL employee_personal_pronouns();
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Employee	Gender Identity	Personal Pronouns
Aamodt Luigi	M	He/Him
Aamodt Luisa	F	She/Her
Aamodt Luise	F	She/Her
Aamodt Magy	M	He/Him
Aamodt Maha	M	He/Him
Aamodt Mahmut	M	He/Him
Aamodt Maik	M	He/Him
Aamodt Mani	M	He/Him
Aamodt Marek	F	She/Her
Aamodt Mariangiola	M	He/Him
Aamodt Marie	F	She/Her
Aamodt Mariusz	M	He/Him
Aamodt Marla	F	She/Her
Aamodt Masamitsu	M	He/Him
Aamodt Matk	M	He/Him
Aamodt Matk	M	He/Him
Aamodt Mats	F	She/Her
Aamodt Matt	M	He/Him
Aamodt Menkae	F	She/Her
Aamodt Mingsen	M	He/Him
Aamodt Mingzeng	M	He/Him
Aamodt Miomir	M	He/Him
Aamodt Mohammed	M	He/Him
Aamodt Mokhtar	M	He/Him
Aamodt Mostafa	F	She/Her
Aamodt Moto	F	She/Her
Aamodt Moty	M	He/Him
Aamodt Nahla	F	She/Her
Aamodt Narain	F	She/Her

Result 9

Output

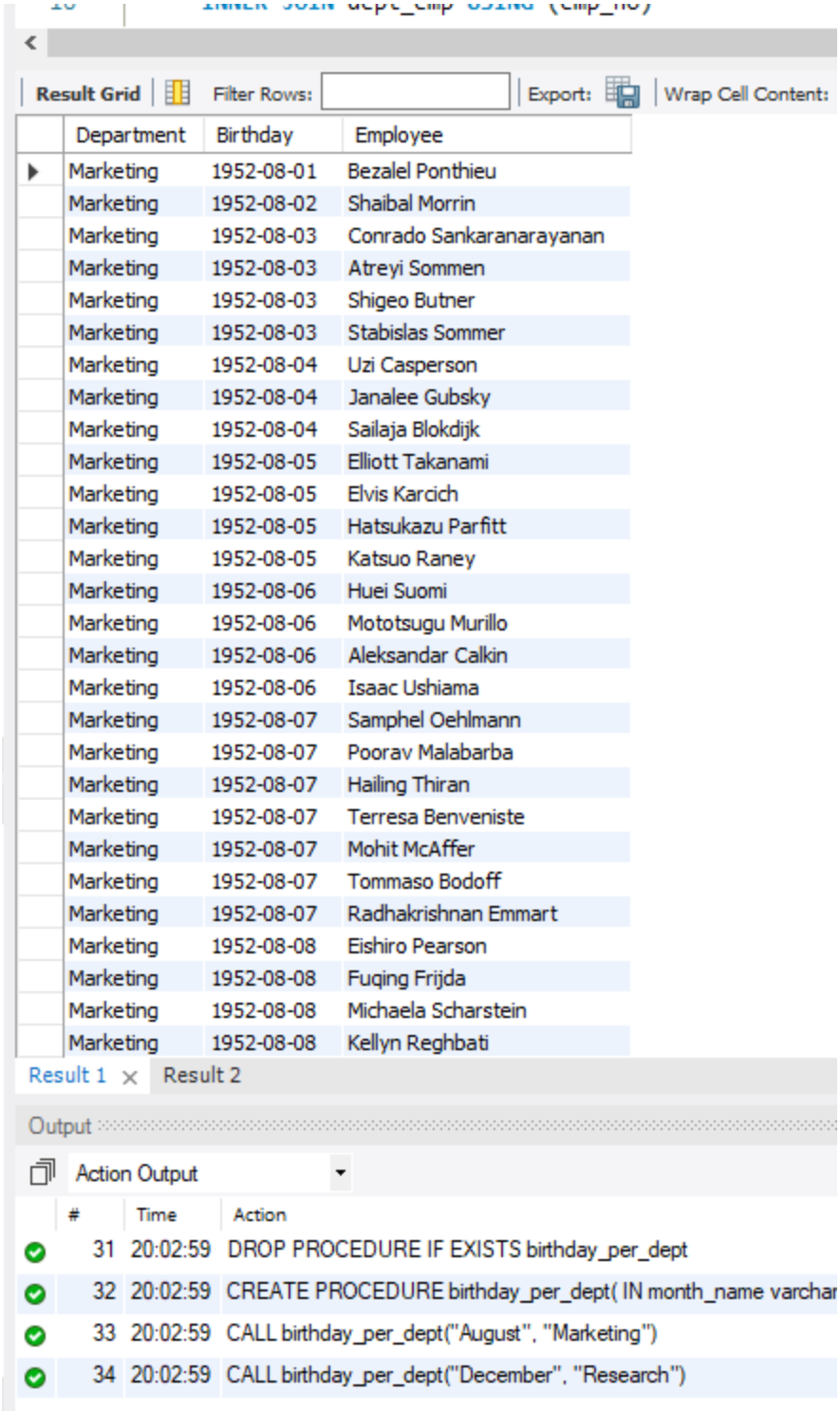
Action Output

#	Time	Action	Message
26	19:41:41	USE employees	0 row(s) affected
27	19:41:41	DROP PROCEDURE IF EXISTS employee_personal_pronouns	0 row(s) affected
28	19:41:41	CREATE PROCEDURE employee_personal_pronouns( ) -- UPDATE employees set personal_pronouns = 'She/Her' -- WHERE gender = 'F'; BEGIN SEL...	0 row(s) affected
29	19:41:41	CALL employee_personal_pronouns()	300023 row(s) returned

Week2Employees | Week1Employees | SQL File 6* | Birthday_per_dept | Pandemic_bonus_per_dept | employee_personal_pronouns | SQL File 13* | Personal_pronouns

Don't Limit

```
1   -- The procedure will allow any department to look up the birthdays with the parameters dept_name and the
2   -- bday month so they could have a nice list to display at the beginning of every month organized by
3   -- date to keep up employee morale and celebrate our employees birthday, sepreating by department
4
5   USE employees;
6   DELIMITER $$
7
8   DROP PROCEDURE IF EXISTS birthday_per_dept$$
9   CREATE PROCEDURE birthday_per_dept(
10  IN month_name varchar(9),
11  IN department varchar(20)
12  )
13
14  BEGIN
15      SELECT dept_name AS "Department", birth_date AS "Birthday", concat(first_name, " ", last_name) AS "Employee" FROM employees e
16      INNER JOIN dept_emp USING (emp_no)
17      INNER JOIN departments d USING (dept_no)
18      WHERE month_name = MONTHNAME(birth_date)
19      AND dept_name = department
20      ORDER BY birth_date;
21  END $$
22  DELIMITER ;
23  CALL birthday_per_dept("August", "Marketing");
24  CALL birthday_per_dept("December", "Research");
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Department	Birthday	Employee
Marketing	1952-08-01	Bezalel Ponthieu
Marketing	1952-08-02	Shaibal Morrin
Marketing	1952-08-03	Conrado Sankaranarayanan
Marketing	1952-08-03	Atreyi Sommen
Marketing	1952-08-03	Shigeo Butner
Marketing	1952-08-03	Stabislas Sommer
Marketing	1952-08-04	Uzi Casperson
Marketing	1952-08-04	Janalee Gubsky
Marketing	1952-08-04	Sailaja Blokdijk
Marketing	1952-08-05	Elliott Takanami
Marketing	1952-08-05	Elvis Karcich
Marketing	1952-08-05	Hatsukazu Parfitt
Marketing	1952-08-05	Katsuo Raney
Marketing	1952-08-06	Huei Suomi
Marketing	1952-08-06	Mototsugu Murillo
Marketing	1952-08-06	Aleksandar Calkin
Marketing	1952-08-06	Isaac Ushiama
Marketing	1952-08-07	Samphel Oehlmann
Marketing	1952-08-07	Poorav Malabarba
Marketing	1952-08-07	Hailing Thiran
Marketing	1952-08-07	Terresa Benveniste
Marketing	1952-08-07	Mohit McAffer
Marketing	1952-08-07	Tommaso Bodoff
Marketing	1952-08-07	Radhakrishnan Emmart
Marketing	1952-08-08	Eishiro Pearson
Marketing	1952-08-08	Fuqing Frijda
Marketing	1952-08-08	Michaela Scharstein
Marketing	1952-08-08	Kellyn Reghbati

Result 1 | Result 2

Output

Action Output

#	Time	Action
31	20:02:59	DROP PROCEDURE IF EXISTS birthday_per_dept
32	20:02:59	CREATE PROCEDURE birthday_per_dept(IN month_name varchar
33	20:02:59	CALL birthday_per_dept("August", "Marketing")
34	20:02:59	CALL birthday_per_dept("December", "Research")

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Department	Birthday	Employee
Research	1952-12-01	Selwyn Pews
Research	1952-12-01	Xiaopeng Nitsche
Research	1952-12-01	Yuqun Giarratana
Research	1952-12-01	Ziyad Matzen
Research	1952-12-01	Basant Constantine
Research	1952-12-01	Gill Clouatre
Research	1952-12-01	Zita Piazza
Research	1952-12-01	Sajjan Chachaty
Research	1952-12-02	Mani Polupanov
Research	1952-12-02	Keung Litzler
Research	1952-12-03	Anneke Harnett
Research	1952-12-03	Uli Plump
Research	1952-12-03	Tesuya Hettesheimer
Research	1952-12-04	Chikako Caine
Research	1952-12-05	Shin Felcyn
Research	1952-12-06	Adhemar Pelz
Research	1952-12-06	Jagoda Molberg
Research	1952-12-06	Yefim Godskesen
Research	1952-12-07	Arco Schwabacher
Research	1952-12-07	Siamak Juneja
Research	1952-12-07	Behnaam Picci
Research	1952-12-07	Luerbio Pintelas
Research	1952-12-07	Chuanti Heusch
Research	1952-12-07	Uta Unno
Research	1952-12-07	Vugranam Kakkar
Research	1952-12-08	Yaghout Dulay
Research	1952-12-08	Ghassan Olivero
Research	1952-12-08	Chanjung Peron

Result 1 | Result 2

Output

Action Output

#	Time	Action
31	20:02:59	DROP PROCEDURE IF EXISTS birthday_per_dept
32	20:02:59	CREATE PROCEDURE birthday_per_dept(IN month_name varchar(9), I
33	20:02:59	CALL birthday_per_dept("August", "Marketing")
34	20:02:59	CALL birthday_per_dept("December", "Research")

Week2Employees | Week1Employees | SQL File 6* | Birthday_per_dept | Pandemic_bonus_per_dept | employee_personal_pronouns | SQL File 13* | Personal_pronouns* | Employee_transition_update*

Don't Limit

```sql
-- Pandemic bonus, Due to some sections either being over worked due to the pandemic or understaffed or
-- both the bonus can be made by selecting the department, the current employeed workers and the
-- % of a bonus per employee salary they would want to give as a bonus
USE employees;
DELIMITER $$

DROP PROCEDURE IF EXISTS pandemic_bonus_per_dept$$
CREATE PROCEDURE pandemic_bonus_per_dept(
IN department varchar(20),
IN x decimal(2,2)
)

BEGIN
    SELECT dept_name AS "Department", concat(last_name, " ", first_name ) AS "Employee", format(salary,2) AS "Yearly Salary", format((salary * x),2) AS "Pandemic Bonus" FROM employees
    INNER JOIN salaries s USING (emp_no)
    INNER JOIN dept_emp USING (emp_no)
    INNER JOIN departments d USING (dept_no)
    WHERE s.to_date > now()
    AND dept_name = department
    ORDER BY last_name, first_name;
END $$
DELIMITER ;
CALL pandemic_bonus_per_dept("Marketing", .05);
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Department	Employee	Yearly Salary	Pandemic Bonus
Marketing	Aamodt Alexius	58,834.00	2,941.70
Marketing	Aamodt Gaetan	90,226.00	4,511.30
Marketing	Aamodt Hidefumi	62,885.00	3,144.25
Marketing	Aamodt Hidefumi	116,825.00	5,841.25
Marketing	Aamodt Mats	114,221.00	5,711.05
Marketing	Aamodt Phuoc	78,772.00	3,938.60
Marketing	Aamodt Roddy	59,222.00	2,961.10
Marketing	Aamodt Sreekrishna	59,087.00	2,954.35
Marketing	Acton Basim	93,826.00	4,691.30
Marketing	Acton Brigham	77,017.00	3,850.85
Marketing	Acton Fay	56,908.00	2,845.40
Marketing	Acton Jongsuk	54,466.00	2,723.30
Marketing	Acton Koichi	86,453.00	4,322.65
Marketing	Acton Kristen	83,890.00	4,194.50
Marketing	Acton Marsal	79,947.00	3,997.35
Marketing	Acton Mohan	86,526.00	4,326.30
Marketing	Acton Moss	75,782.00	3,789.10
Marketing	Acton Shim	89,527.00	4,476.35
Marketing	Acton Terresa	76,080.00	3,804.00
Marketing	Acton Werner	69,504.00	3,475.20
Marketing	Adachi Adit	92,544.00	4,627.20
Marketing	Adachi Elvis	67,789.00	3,389.45
Marketing	Adachi Fen	85,551.00	4,277.55
Marketing	Adachi Jamaludin	58,487.00	2,924.35
Marketing	Adachi Jenwei	115,087.00	5,754.35
Marketing	Adachi Kazuhira	75,732.00	3,786.60
Marketing	Adachi Kazuhiro	85,158.00	4,257.90
Marketing	Adachi Kristin	59,924.00	2,996.20
Marketing	Adachi Luise	85,356.00	4,267.80
Marketing	Adachi Sudharsan	76,535.00	3,826.75

Result 1

Output

Action Output

#	Time	Action	Message
35	20:08:54	USE employees	0 row(s) affected
36	20:08:54	DROP PROCEDURE IF EXISTS pandemic_bonus_per_dept	0 row(s) affected
37	20:08:54	CREATE PROCEDURE pandemic_bonus_per_dept(IN department varchar(20), IN x decimal(2,2)) BEGIN SELECT dept_name AS "Department", co...	0 row(s) affected
38	20:08:54	CALL pandemic_bonus_per_dept("Marketing", .05)	16252 row(s) returned

Week2Employees | Week1Employees | SQL File 6* | Birthday_per_dept | Pandemic_bonus_per_dept | employee_personal_pronouns | SQL File 13* | mandatory_overtime | S

Don't Limit

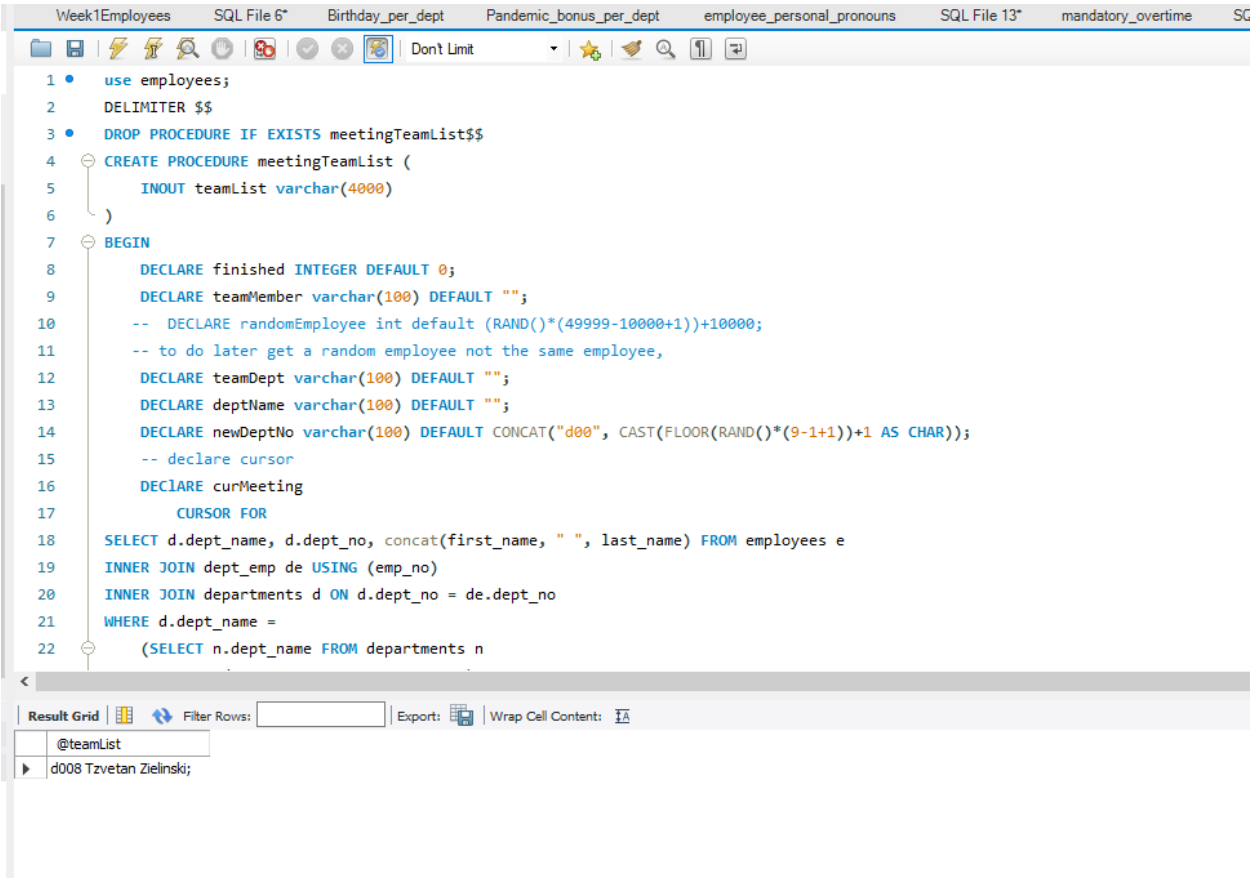
```
1   -- stored procedure with else if
2   -- mandatory overtime
3
4   USE employees;
5   DELIMITER $$
6   -- DESC employees;
7   DROP PROCEDURE IF EXISTS mandatory_overtime$$
8   CREATE PROCEDURE mandatory_overtime(
9   in scheudle_code varchar(20),
10  out hours_needed varchar(20)
11  )
12
13
14  BEGIN
15      IF scheudle_code = 'On Target'
16      THEN
17      SET hours_needed = 'No overtime needed';
18      ELSEIF scheudle_code = 'Behind Target'
19      THEN
20      SET  hours_needed = '2 hour of overtime for company needed';
21      ELSEIF scheudle_code = 'Missed Target'
22      THEN
23      SET hours_needed = '4 hour of overtime for company needed';
24      END IF;
25          END $$
```

```
13
14  BEGIN
15      IF scheudle_code = 'On Target'
16      THEN
17      SET hours_needed = 'No overtime needed';
18      ELSEIF scheudle_code = 'Behind Target'
19      THEN
20      SET  hours_needed = '2 hour of overtime for company needed';
21      ELSEIF scheudle_code = 'Missed Target'
22      THEN
23      SET hours_needed = '4 hour of overtime for company needed';
24      END IF;
25          END $$
26  DELIMITER ;
27  CALL mandatory_overtime ('On Target', @hours_needed);
28  select @hours_needed;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

@hours_needed
No overtime needed

Result 8


Week1Employees
SQL File 6*
Birthday_per_dept
Pandemic_bonus_per_dept
employee_personal_pronouns
SQL File 13*
mandatory_overtime
Don't Limit
use employees;
DELIMITER $$
DROP PROCEDURE IF EXISTS meetingTeamList$$
CREATE PROCEDURE meetingTeamList (
INOUT teamList varchar(4000)
)
BEGIN
DECLARE finished INTEGER DEFAULT 0;
DECLARE teamMember varchar(100) DEFAULT "";
-- DECLARE randomEmployee int default (RAND()*(49999-10000+1))+10000;
-- to do later get a random employee not the same employee,
DECLARE teamDept varchar(100) DEFAULT "";
DECLARE deptName varchar(100) DEFAULT "";
DECLARE newDeptNo varchar(100) DEFAULT CONCAT("d00", CAST(FLOOR(RAND()*(9-1+1))+1 AS CHAR));
-- declare cursor
DEClARE curMeeting
CURSOR FOR
SELECT d.dept_name, d.dept_no, concat(first_name, " ", last_name) FROM employees e
INNER JOIN dept_emp de USING (emp_no)
INNER JOIN departments d ON d.dept_no = de.dept_no
WHERE d.dept_name =
(SELECT n.dept_name FROM departments n
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
@teamList
d008 Tzvetan Zielinski;

Week1Employees | SQL File 6* | Birthday_per_dept | Pandemic_bonus_per_dept | employee_personal_pronouns | SQL File 1

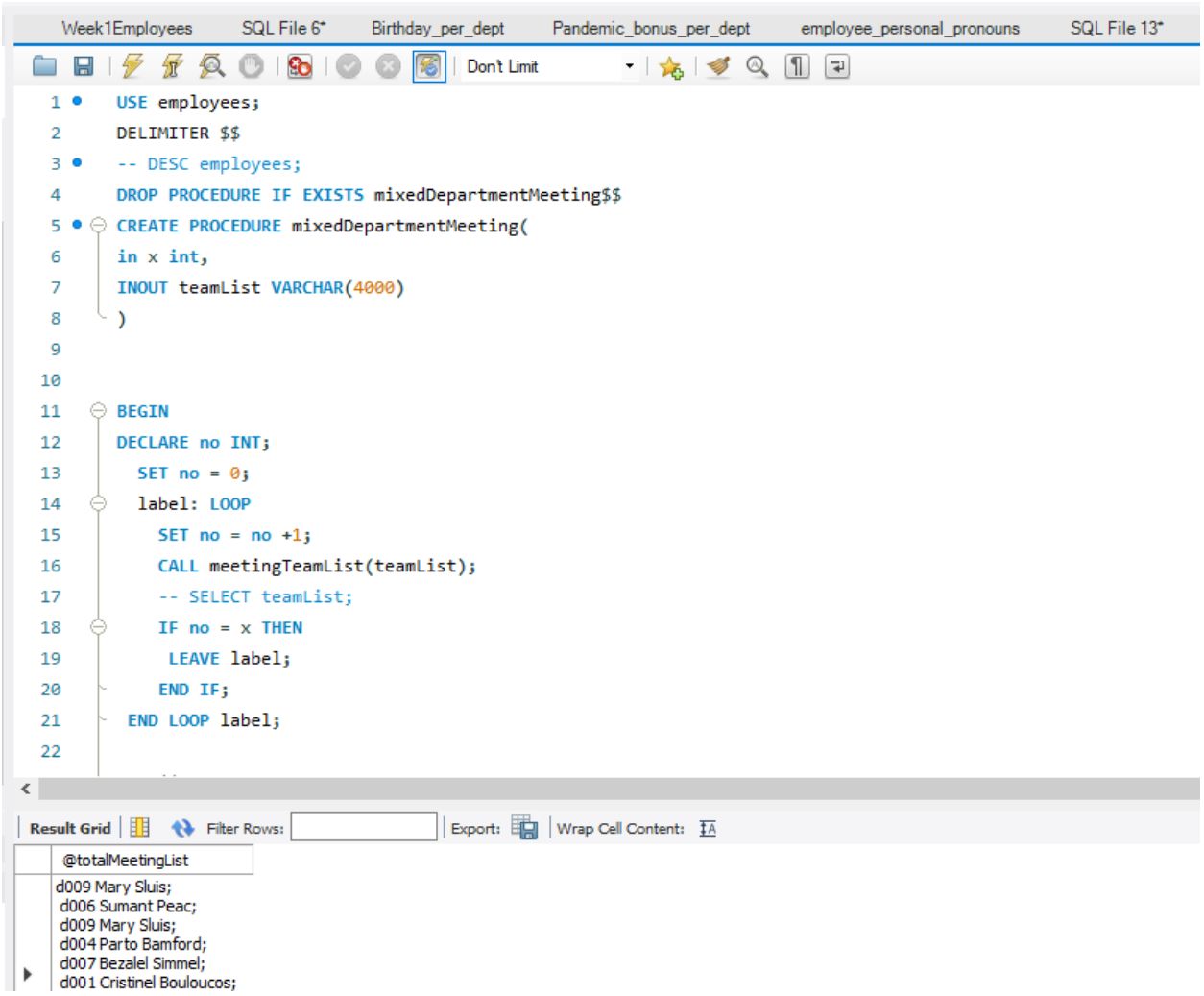
Don't Limit

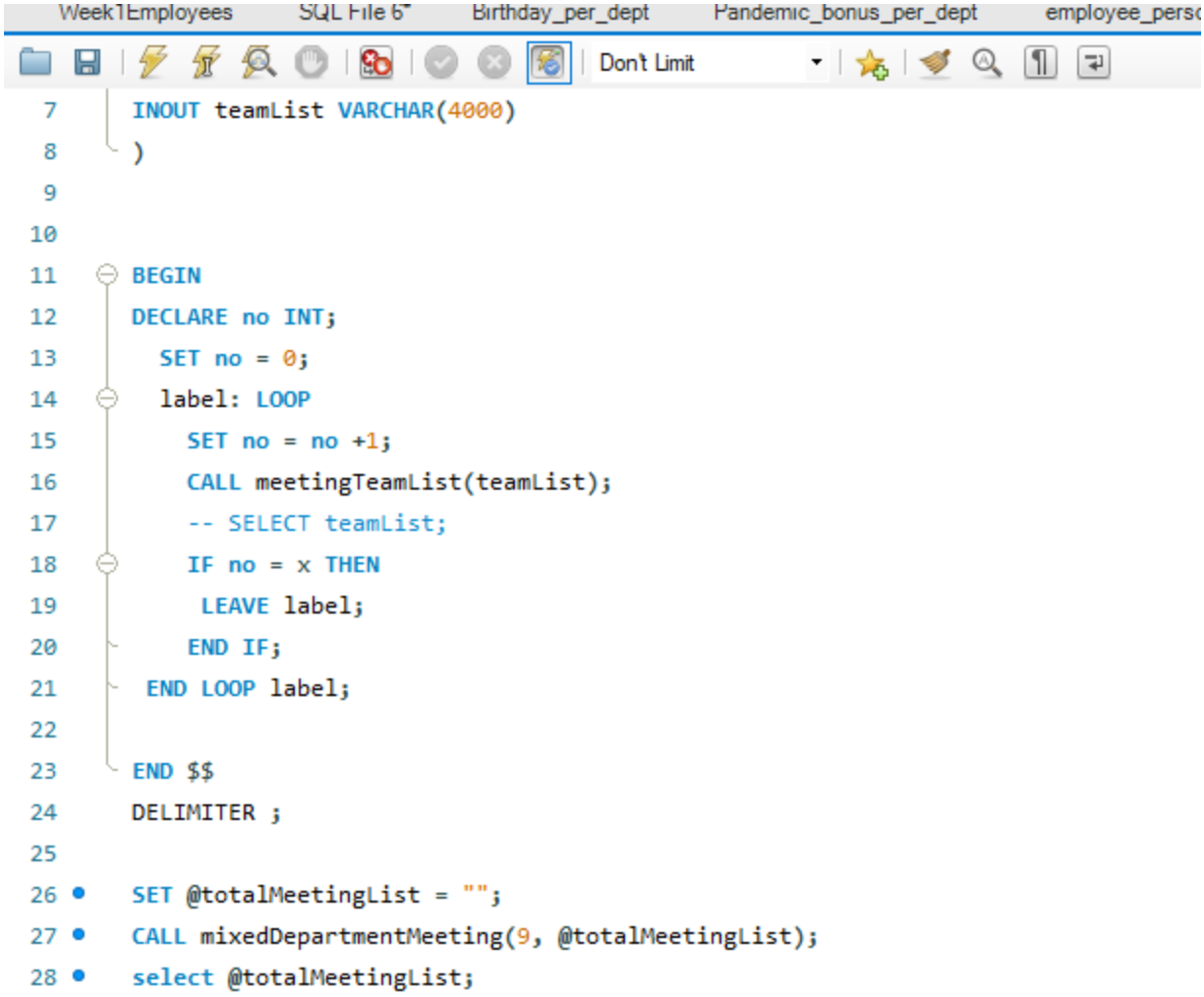
```
19  INNER JOIN dept_emp de USING (emp_no)
20  INNER JOIN departments d ON d.dept_no = de.dept_no
21  WHERE d.dept_name =
22      (SELECT n.dept_name FROM departments n
23       WHERE n.dept_no = newDeptNo LIMIT 1)
24       LIMIT 1;
25      -- declare NOT FOUND handler
26      DECLARE CONTINUE HANDLER
27      FOR NOT FOUND SET finished = 1;
28      OPEN curMeeting;
29      getTeam: LOOP
30          FETCH curMeeting INTO deptName, teamDept, teamMember;
31          IF (finished = 1) THEN
32              LEAVE getTeam;
33          END IF;
34          -- build meeting list
35          SET teamList = CONCAT(teamDept, " ", teamMember,";\n",teamList);
36      END LOOP getTeam;
37      CLOSE curMeeting;
38  END$$
39  DELIMITER ;
40
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

@teamList
d008 Tzvetan Zielinski;

```
34          -- build meeting list
35          SET teamList = CONCAT(teamDept, " ", t
36      END LOOP getTeam;
37      CLOSE curMeeting;
38  END$$
39  DELIMITER ;
40
41  SET @teamList = "";
42  CALL meetingTeamList(@teamList);
43  SELECT @teamList;
```


Week1Employees
SQL File 6*
Birthday_per_dept
Pandemic_bonus_per_dept
employee_personal_pronouns
SQL File 13*
Don't Limit
1 USE employees;
2 DELIMITER $$
3 -- DESC employees;
4 DROP PROCEDURE IF EXISTS mixedDepartmentMeeting$$
5 CREATE PROCEDURE mixedDepartmentMeeting(
6 in x int,
7 INOUT teamList VARCHAR(4000)
8)
9
10
11 BEGIN
12 DECLARE no INT;
13 SET no = 0;
14 label: LOOP
15 SET no = no +1;
16 CALL meetingTeamList(teamList);
17 -- SELECT teamList;
18 IF no = x THEN
19 LEAVE label;
20 END IF;
21 END LOOP label;
22
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
@totalMeetingList
d009 Mary Sluis;
d006 Sumant Peac;
d009 Mary Sluis;
d004 Parto Bamford;
d007 Bezalel Simmel;
d001 Cristinel Bouloucos;



@totalMeetingList
d009 Mary Sluis; d006 Sumant Peac; d009 Mary Sluis; d004 Parto Bamford; d007 Bezalel Simmel; d001 Cristinel Bouloucos; d008 Tzvetan Zielinski; d001 Cristinel Bouloucos; d005 Georgi Facello;

URL to GitHub Repository:

https://github.com/JoleneMel/Week4StoredProcedures