Guzon, Jeanne Margaret L.
BS Information Technology 3-1

COMP 20213 Database Administration

CHAPTER 1

Introduction to Database Administration

ASSESSMENTS:

Answer the following questions:

1. How does a database administrator ensure database quality?

   - Data Administrators do database analysis and design. They choose the DBMS and related software tools, which requires an thorough evaluation of the suppliers and their software product. They optimize database speed by rebuilding, rearranging, and re-indexing the database on a regular basis. Database administrators safeguard the data security, privacy, and integrity of servers and databases. They are regularly backup and recovery data, as well as install and upgrade DBMS.

2. Why must a database administrator have a good knowledge of SQL?

   - Learning SQL will make your job as a database administrator easier since it will allow you to access databases more effectively and quickly instead of having to go through each one to find a record. As SQL servers fail, a database administrator who is well-versed in SQL will keep things running smoothly, and a certified data administrator will be able to think of solutions and handle the challenges. They will also feel more at ease engaging with other programs because SQL is used by the majority of office apps.

3. Give an example of multiple definitions of the same data entity in the database.

   - You have a shelf of notebooks, despite on how they look similar with each other, they have different content and subjects.

4. In your own words, explain how a database administrator can prevent excessive database downtime.

   - Database administrators should assess business and technological requirements to determine what should be changed or kept. They should also plan maintenance carefully and regularly so as not to disrupt daily operations. They must consider the best time to arrange a large system check vs. a minor issue check. They must maintain operations throughout migrations and upgrades. They must coexist between old and new systems. Database administrators must also backup and restore data so that they do not lose sight of what they will do if data is damaged or lost.

5. Each of the results of ineffective data administration above can be classified in any of the three most important aspects of database handling, security, availability, and quality. Identify to which aspect each result can be classified. For example, #8 ‘Embarrassment to the organization because of unauthorized access of data’ would fall under security aspect.

QUALITY	**#1** Multiple definitions of same data entity and /or inconsistent representations of the same data elements in databases

	#2 Missing key data elements whose loss eliminates the value of existing data
	#3 Low data quality levels due to inappropriate sources of data or timing of data transfers from one system to another
AVAILABILITY	**#4** Inadequate familiarity with existing data, including awareness of data location
	#5 Poor and inconsistent query response time, excessive database downtime
	#7 Lack of access to data due to damaged, sabotaged, or stolen files or due to hardware failures
SECURITY	**#6** Either stringent or inadequate controls to ensure agreed upon data privacy and security
	#7 Embarrassment to the organization because of unauthorized access of data

ACTIVITY (Optional):

Interview a DBA. From the interview list down the work that he does which relates to keeping the database secure and available at all times.

Guzon, Jeanne Margaret L.
BS Information Technology 3-1

COMP 20213 Database Administration

CHAPTER 2

Structured Query Language (SQL)

ASSESSMENTS:

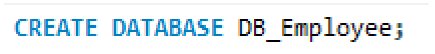
True or False

1. A column function can be used in a WHERE clause. **FALSE**
2. The HAVING and WHERE clauses can be used interchangeably. **FALSE**
3. There should only be 1 attribute in a GROUP BY. **FALSE**
4. The clauses in a SELECT statement should be coded in a specific order. **TRUE**
5. The use of '*' in a SELECT clause means all attributes from the table will be displayed as part of the output. **TRUE**

Programming

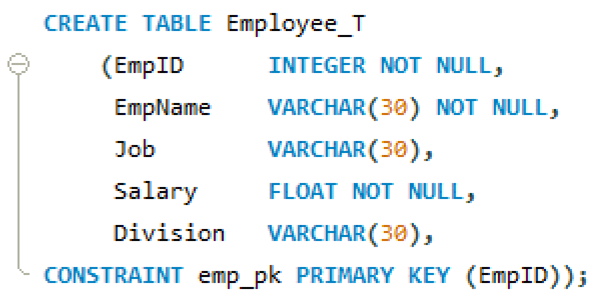
Use the Employee_T table (from the topic on GROUP BY) for the following problems. Create SQL programs for the following requirements. Take note of the syntax provided to be able to create your SQL code.

1. Create a database and name it DB_Employee



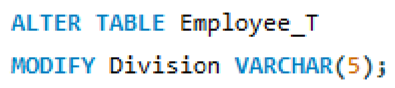
```
CREATE DATABASE DB_Employee;
```

2. Create Employee_T table in the database. Provide the appropriate data type/size for each attribute



```
CREATE TABLE Employee_T
    (EmpID      INTEGER NOT NULL,
     EmpName    VARCHAR(30) NOT NULL,
     Job        VARCHAR(30),
     Salary     FLOAT NOT NULL,
     Division   VARCHAR(30),
CONSTRAINT emp_pk PRIMARY KEY (EmpID));
```

3. Alter a property of an attribute in Employee_T. Increase the size of one of its attributes, any attribute.



```
ALTER TABLE Employee_T
MODIFY Division VARCHAR(5);
```

4. Add another attribute to Employee_T. The attribute is Date_Hired.

```
ALTER TABLE Employee_T
ADD Date_Hired DATE;
```

5. Insert 5 records in Employee_T. Provide your own data.

```
INSERT INTO Employee_T VALUES
(7, 'Cara Elyssa Dulay', 'MGR', 100000, 'ISD', '2015-11-19'),
(8, 'Ian Vincent Bartolome', 'RF', 25000, 'FIN', '2020-06-12'),
(9, 'Agatha Gwen Tagala', 'MGR', 25000, 'HR', '2018-01-17'),
(10, 'Patrick Sudaria', 'SMR', 50000, 'FIN', '2022-04-10'),
(11, 'John Dani Oicangi', 'SMR', 15000, 'HR', '2020-12-12');
```

6. List all records from Employee_T sorted by division

```
SELECT *
FROM Employee_T
ORDER BY Division;
```

7. List only the name of the employees who are managers

```
SELECT *
FROM Employee_T
WHERE Job = 'MGR';
```

8. List the names of the employees who are from ISD and whose salary is > 200

```
SELECT *
FROM Employee_T
WHERE Division = 'ISD'AND Salary > 200;
```

9. List all records of employees who are from ISD or FIN, and whose salary is > 200

```
SELECT *
FROM Employee_T
WHERE Division = 'ISD' OR Division = 'FIN' AND Salary > 200;
```

10. Update any field of any one record in Employee_T.

```
UPDATE Employee_T
SET Salary = 300
WHERE EmpID = 11;
```

11. Update the salary to 1000 of all the managers in the table

```
UPDATE Employee_T
SET Salary = 1000
WHERE Job = 'MGR';
```

12. Count how many employees there are in the table

```
SELECT COUNT(*)
From Employee_T;
```

13. Count how managers are there from the table.

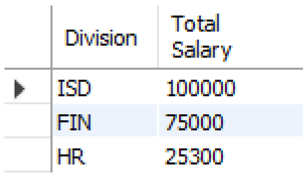
```
SELECT COUNT(*)
From Employee_T
WHERE Job = 'MGR';
```

14. Sum the salaries of all in the ISD department.

```
SELECT SUM(Salary)
From Employee_T
WHERE Division = 'ISD';
```

15. Sum up the salary per division. Show division in the output

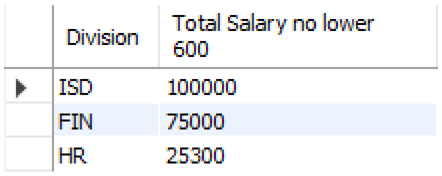
```
SELECT Division, SUM(Salary) AS "Total Salary"
FROM Employee_T
GROUP BY Division;
```



Division	Total Salary
ISD	100000
FIN	75000
HR	25300

16. Sum the salary per division and display only the total salaries which are greater than 600

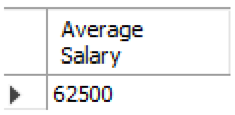
```
SELECT Division, SUM(Salary) AS "Total Salary no lower 600"
FROM Employee_T
GROUP BY Division
HAVING SUM(Salary)>600;
```



Division	Total Salary no lower 600
ISD	100000
FIN	75000
HR	25300

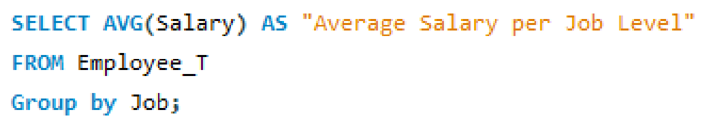
17. Display the average salary of the managers.

```
SELECT AVG(Salary) AS "Average Salary"
FROM Employee_T
WHERE Job = 'MGR';
```

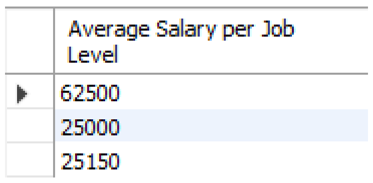


Average Salary
62500

18. Display the average salary per job level.

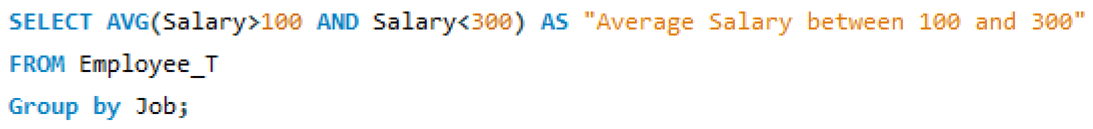


```
SELECT AVG(Salary) AS "Average Salary per Job Level"
FROM Employee_T
Group by Job;
```

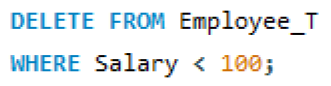


Average Salary per Job Level
62500
25000
25150

19. Display the average salary per job level where the average salary is between 100 to 300 20. Delete records from Employee_T if the salary is less than 100.



```
SELECT AVG(Salary>100 AND Salary<300) AS "Average Salary between 100 and 300"
FROM Employee_T
Group by Job;
```



```
DELETE FROM Employee_T
WHERE Salary < 100;
```

ACTIVITIES:

1. Create a different table, e.g. Student and practice the same set of questions above using this table. Identify the attributes that may be part of the table that you will create e.g. for a Student table, attributes may be student id, student name, birthdate, address, course

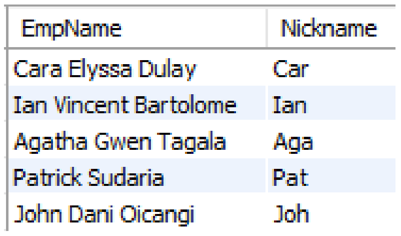
```
CREATE DATABASE activity3;

CREATE TABLE Student
        ( studID        INTEGER NOT NULL,
          studName      VARCHAR(30) NOT NULL,
          yearLvl       VARCHAR(30),
          age           INTEGER NOT NULL,
          course            VARCHAR(30),
CONSTRAINT stud_pk PRIMARY KEY (studID));

ALTER TABLE Student
MODIFY course VARCHAR(5);

ALTER TABLE Student
ADD birthday DATE;

INSERT INTO Student VALUES
(01, 'Cara Elyssa Dulay', '3rd', 25, 'BSIT', '2015-11-19'),
(02, 'Ian Vincent Bartolome', '2nd', 20, 'BSCS', '2020-06-12'),
(03, 'Agatha Gwen Tagala', '3rd', 26, 'BSIT', '2018-01-17'),
(04, 'Patrick Sudaria', '1st', 17, 'BSPHY', '2022-04-10'),
(05, 'John Dani Oicangi', '3rd', 22, 'BSME', '2020-12-12');
```

```
SELECT * FROM Student
ORDER BY course;

SELECT *
FROM Student
WHERE yearLvl = '3rd';

SELECT *
FROM Student
WHERE course = 'BSIT' OR course = 'BSCS' AND age > 20;

UPDATE Student
SET age = 18
WHERE studID = 04;

UPDATE Student
SET age = 25
WHERE yearLvl = '3rd';

SELECT COUNT(*)
From Student
WHERE yearLvl = '3rd';

SELECT SUM(age)
From Student
WHERE course = 'BSIT';

SELECT course, SUM(age) AS "Total age"
FROM Student
GROUP BY course;

SELECT course, SUM(age) AS "Total age no lower 20"
FROM Student
GROUP BY course
HAVING SUM(age)>20;

SELECT AVG(age) AS "Average age"
FROM Student
WHERE yearLvl = '3rd';

SELECT AVG(age) AS "Average age per Year Level"
FROM Student
Group by yearLvl;
```

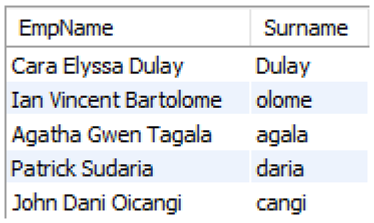
```
SELECT AVG(age>20 AND age<24) AS "Average age between 20 and 24"
FROM Student
Group by yearLvl;

DELETE FROM Student
WHERE age < 20;
```

2. Research on the other scalar functions and practice programming on how they are used. Examples of these are LEFT, RIGHT, RTRIM, LTRIM, ROUND, etc.

```
SELECT EmpName,
LEFT(EmpName,3) as 'Nickname'
FROM Employee_T;

SELECT EmpName,
Right(EmpName,5) as 'Surname'
FROM Employee_T;
```



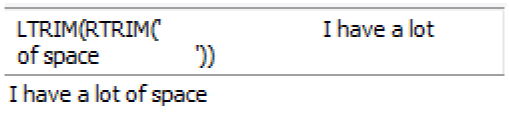
EmpName	Nickname
Cara Elyssa Dulay	Car
Ian Vincent Bartolome	Ian
Agatha Gwen Tagala	Aga
Patrick Sudaria	Pat
John Dani Oicangi	Joh



EmpName	Surname
Cara Elyssa Dulay	Dulay
Ian Vincent Bartolome	olome
Agatha Gwen Tagala	agala
Patrick Sudaria	daria
John Dani Oicangi	cangi

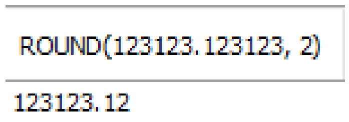
LEFT() and RIGHT() functions return the specified number of characters from the specified side end of the string.

```
SELECT LTRIM(RTRIM('                         I have a lot of space              '));
```



LTRIM(RTRIM(' I have a lot of space '))
I have a lot of space

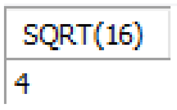
LTRIM() and RTRIM() functions return to remove unnecessary spaces from the specified side end of the string.



ROUND(123123.123123, 2)
123123.12

```
SELECT ROUND(123123.123123, 2)
```

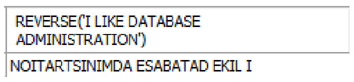
ROUND() function rounds a number to a specified number of decimal places.



SQRT(16)
4

```
SELECT SQRT(16);
```

SQRT() function returns the square root of a number.



REVERSE('I LIKE DATABASE ADMINISTRATION')
NOITARTSINIMDA ESABATAD EKIL I

```
SELECT REVERSE('I LIKE DATABASE ADMINISTRATION');
```

REVERSE() function returns a reversed string

Guzon, Jeanne Margaret L. COMP 20213 Database Administration
BS Information Technology 3-1

CHAPTER 3

Integrity Constraints

ASSESSMENTS / ACTIVITIES:

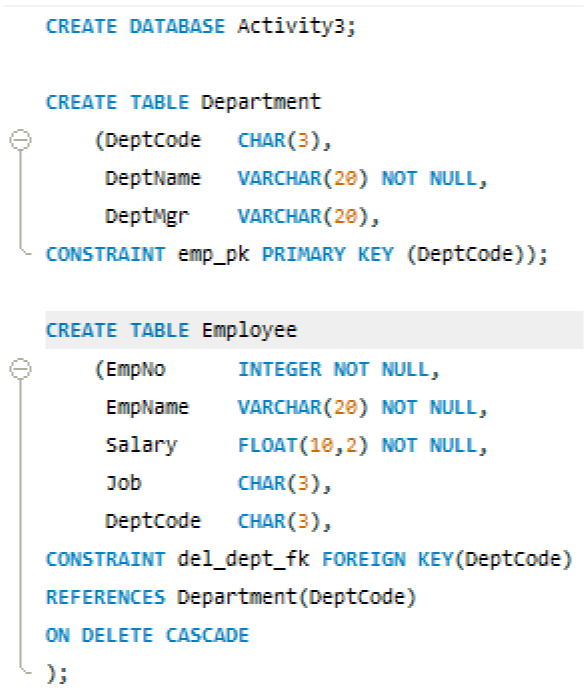
1. Create table Employee using below attributes with the data type/sizes provided

Employee Number	INT, must not be null
Employee Name	Variable Character : Size (20) must not be null
Salary Number	Size (10,2) must not be null
Job Level	Character : Size (3)
Dept code	Character : Size (2)

2. Create table Department using below attributes with the data type/sizes provided.

Dept code	Character : Size (2) must not be null
Dept Name	Variable Character : Size (20) must not be null
Dept Mgr	Variable Character : Size (30)

a. The constraint in Department table should be that every time a dept code is deleted In Department table, corresponding record with same department is likewise deleted in Employee table.



```
CREATE DATABASE Activity3;

CREATE TABLE Department
    (DeptCode    CHAR(3),
     DeptName    VARCHAR(20) NOT NULL,
     DeptMgr     VARCHAR(20),
CONSTRAINT emp_pk PRIMARY KEY (DeptCode));

CREATE TABLE Employee
    (EmpNo       INTEGER NOT NULL,
     EmpName     VARCHAR(20) NOT NULL,
     Salary      FLOAT(10,2) NOT NULL,
     Job         CHAR(3),
     DeptCode    CHAR(3),
CONSTRAINT del_dept_fk FOREIGN KEY(DeptCode)
REFERENCES Department(DeptCode)
ON DELETE CASCADE
);
```

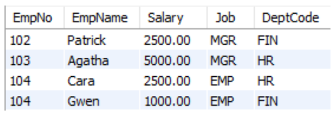
b. Populate Employee and Department tables

i. Take note that dept code that you will use in Employee table must be dept code which are in the Department table

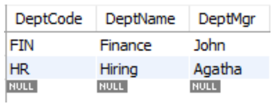
```
INSERT INTO Department VALUES
('ISD', 'Operations' , 'Patrick'),
('FIN', 'Finance', 'John'),
('HR', 'Hiring', 'Agatha');

INSERT INTO Employee VALUES
(101, 'John', 1000, 'MGR', 'ISD'),
(102, 'Patrick', 2500, 'MGR', 'FIN'),
(103, 'Agatha', 5000, 'MGR', 'HR'),
(104, 'Cara', 2500, 'EMP' , 'HR'),
(104, 'Gwen', 1000, 'EMP', 'FIN');
```

ii. Delete 1 record in the Department table. Take note of what happened in Employee table

```
DELETE FROM Department
WHERE DeptCode = 'ISD';
```



EmpNo	EmpName	Salary	Job	DeptCode
102	Patrick	2500.00	MGR	FIN
103	Agatha	5000.00	MGR	HR
104	Cara	2500.00	EMP	HR
104	Gwen	1000.00	EMP	FIN



DeptCode	DeptName	DeptMgr
FIN	Finance	John
HR	Hiring	Agatha
NULL	NULL	NULL

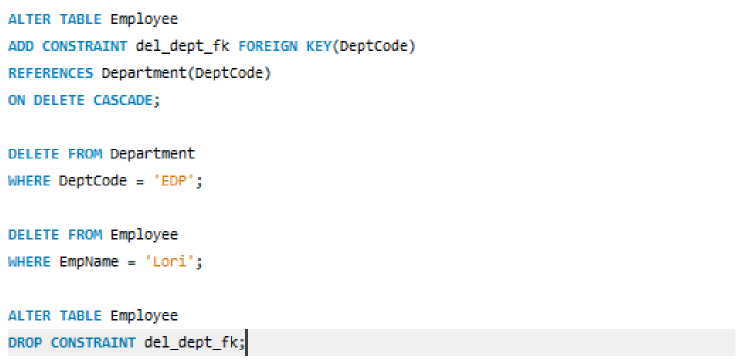
c. Revise constraint in Department table so that if a dept code is updated in Department table, it is likewise updated in Employee table.

i. Update the value of a department code. For example, say one of the department codes is 'EDU', update it to 'EDT. Take note of what happened for corresponding records in Employee table.

```
ALTER TABLE Employee
DROP CONSTRAINT del_dept_fk;

ALTER TABLE Employee
ADD CONSTRAINT update_dept_fk FOREIGN KEY(DeptCode) REFERENCES Department(DeptCode) ON UPDATE CASCADE;

INSERT INTO Department VALUES
('EDU', 'EDUCATION','Ian');

INSERT INTO Employee VALUES
(104, 'Ian', 1000, 'EMP', 'EDU');

UPDATE Department
Set DeptCode = 'EDT'
WHERE DeptCode = 'EDU';
```

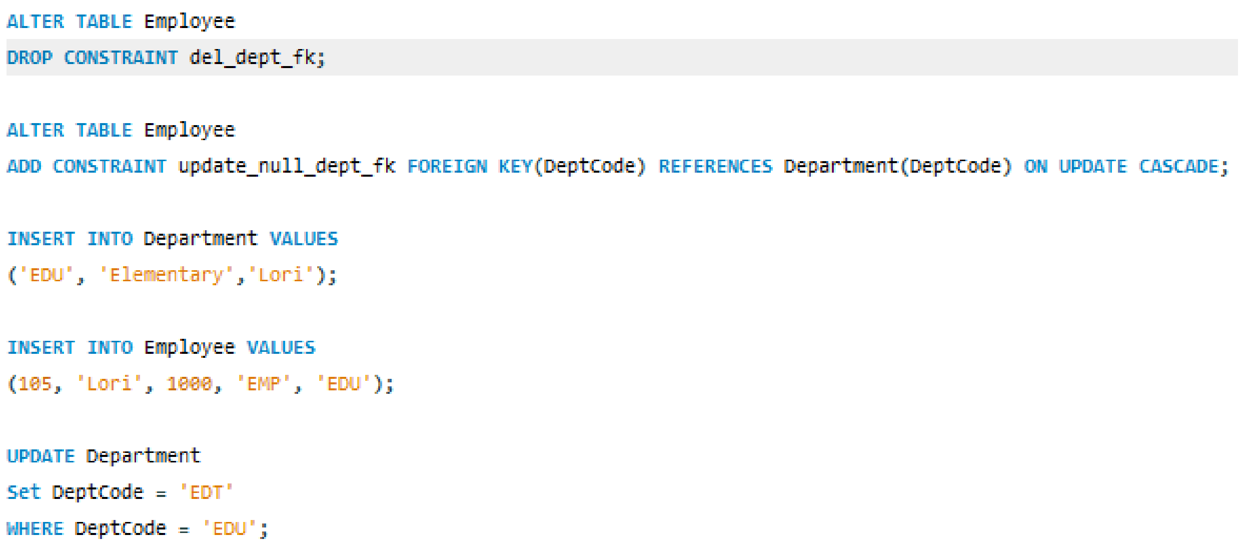
d. Revise constraint in Department table so that if a dept code is updated in Department table, it is updated to null in Employee table.

deleting the previous record



```
ALTER TABLE Employee
ADD CONSTRAINT del_dept_fk FOREIGN KEY(DeptCode)
REFERENCES Department(DeptCode)
ON DELETE CASCADE;

DELETE FROM Department
WHERE DeptCode = 'EDP';

DELETE FROM Employee
WHERE EmpName = 'Lori';

ALTER TABLE Employee
DROP CONSTRAINT del_dept_fk;
```

i. Update the value of a department code. For example, say one of the department codes is ‘EDU’, update it to ‘EDT. Take note of what happened to the corresponding records in Employee table.



```
ALTER TABLE Employee
DROP CONSTRAINT del_dept_fk;

ALTER TABLE Employee
ADD CONSTRAINT update_null_dept_fk FOREIGN KEY(DeptCode) REFERENCES Department(DeptCode) ON UPDATE CASCADE;

INSERT INTO Department VALUES
('EDU', 'Elementary','Lori');

INSERT INTO Employee VALUES
(105, 'Lori', 1000, 'EMP', 'EDU');

UPDATE Department
Set DeptCode = 'EDT'
WHERE DeptCode = 'EDU';
```

Guzon, Jeanne Margaret L.
BS Information Technology 3-1

CHAPTER 4

Working with Multiple Tables

ASSESSMENTS:

Answer the following questions:

1. What are the rules that must be satisfied in order to perform a union, intersect, and except?
   - Tables should be joined by a common attribute. They must have the same number of output columns with compatible data types.
2. What type of Join must be used if we want to display only all matching records from 2 tables
   - Inner Join
3. How many records will a cross join output if 1 table has 5 records while the second table has 6 records?
   - 30 records
4. When do you use an inner join over an outer join?
   - if you only want to keep information from both tables that are related to each other
5. What is the difference between a left join and a right join?
   - Use a left join if you want all of the rows from the first table and any matching rows from the second table.
   - Use a right join if you want all of the rows from the second table as well as any matching rows from the first table.

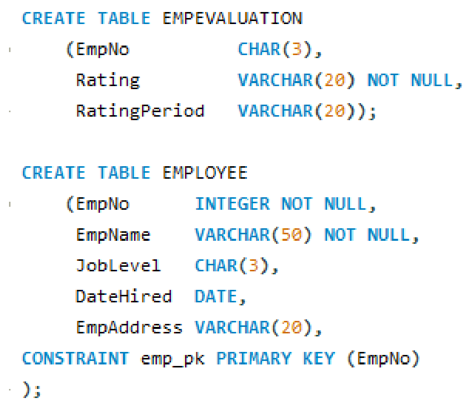
ACTIVITIES:

Use the 2 tables below

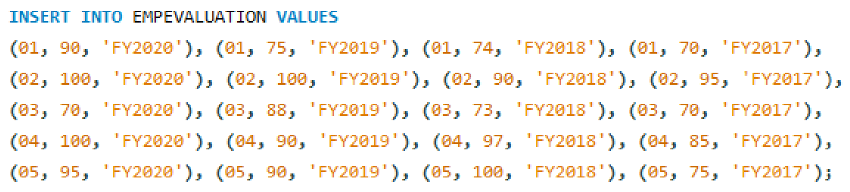
EMPLOYEE(EmpNo, EName, JobLevel, DateHired, EmpAddress) → Employee Master File

EMPEVALUATION(EmpNo, Rating, RatingPeriod) → Table of all over-all ratings of employee for all rating periods

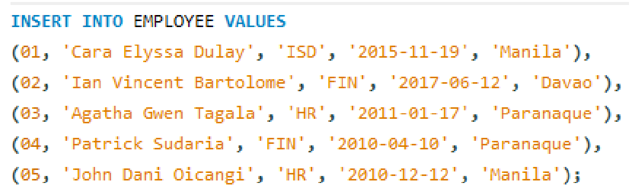
Guzon, Jeanne Margaret L.
BS Information Technology 3-1



```
CREATE TABLE EMPEVALUATION
    (EmpNo          CHAR(3),
     Rating         VARCHAR(20) NOT NULL,
     RatingPeriod   VARCHAR(20));

CREATE TABLE EMPLOYEE
    (EmpNo      INTEGER NOT NULL,
     EmpName    VARCHAR(50) NOT NULL,
     JobLevel   CHAR(3),
     DateHired  DATE,
     EmpAddress VARCHAR(20),
CONSTRAINT emp_pk PRIMARY KEY (EmpNo)
);
```



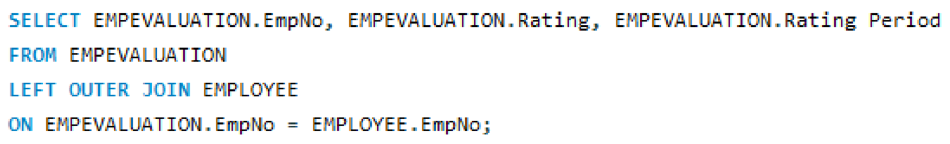
```
INSERT INTO EMPEVALUATION VALUES
(01, 90, 'FY2020'), (01, 75, 'FY2019'), (01, 74, 'FY2018'), (01, 70, 'FY2017'),
(02, 100, 'FY2020'), (02, 100, 'FY2019'), (02, 90, 'FY2018'), (02, 95, 'FY2017'),
(03, 70, 'FY2020'), (03, 88, 'FY2019'), (03, 73, 'FY2018'), (03, 70, 'FY2017'),
(04, 100, 'FY2020'), (04, 90, 'FY2019'), (04, 97, 'FY2018'), (04, 85, 'FY2017'),
(05, 95, 'FY2020'), (05, 90, 'FY2019'), (05, 100, 'FY2018'), (05, 75, 'FY2017');
```



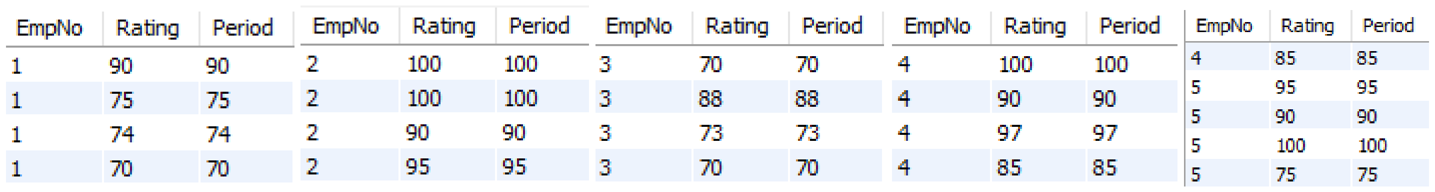
```
INSERT INTO EMPLOYEE VALUES
(01, 'Cara Elyssa Dulay', 'ISD', '2015-11-19', 'Manila'),
(02, 'Ian Vincent Bartolome', 'FIN', '2017-06-12', 'Davao'),
(03, 'Agatha Gwen Tagala', 'HR', '2011-01-17', 'Paranaque'),
(04, 'Patrick Sudaria', 'FIN', '2010-04-10', 'Paranaque'),
(05, 'John Dani Oicangi', 'HR', '2010-12-12', 'Manila');
```

Use join, intersect, union, or except for the following problems:

1. Display the employee number, name, and his ratings for the different rating periods.

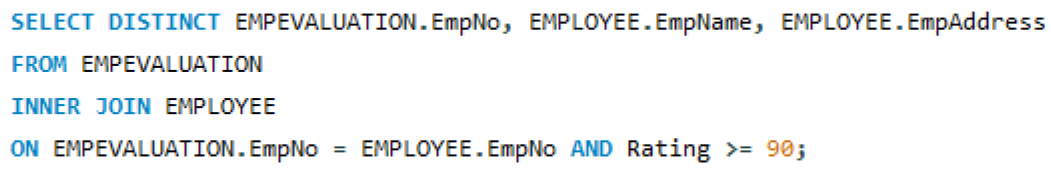


```
SELECT EMPEVALUATION.EmpNo, EMPEVALUATION.Rating, EMPEVALUATION.Rating Period
FROM EMPEVALUATION
LEFT OUTER JOIN EMPLOYEE
ON EMPEVALUATION.EmpNo = EMPLOYEE.EmpNo;
```

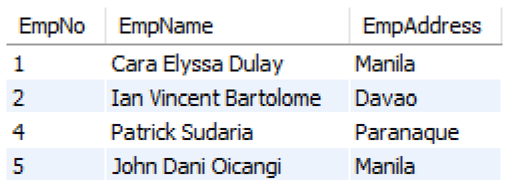


EmpNo	Rating	Period
1	90	90
1	75	75
1	74	74
1	70	70
2	100	100
2	100	100
2	90	90
2	95	95
3	70	70
3	88	88
3	73	73
3	70	70
4	100	100
4	90	90
4	97	97
4	85	85
4	85	85
5	95	95
5	90	90
5	100	100
5	75	75

2. Display the employee number, name, and address of employees who got 90 and above in any of the rating periods.



```
SELECT DISTINCT EMPEVALUATION.EmpNo, EMPLOYEE.EmpName, EMPLOYEE.EmpAddress
FROM EMPEVALUATION
INNER JOIN EMPLOYEE
ON EMPEVALUATION.EmpNo = EMPLOYEE.EmpNo AND Rating >= 90;
```



EmpNo	EmpName	EmpAddress
1	Cara Elyssa Dulay	Manila
2	Ian Vincent Bartolome	Davao
4	Patrick Sudaria	Paranaque
5	John Dani Oicangi	Manila

3. Display the employee number, name, and address of employees who got between 70-75 during ‘FY2018’ rating period.

```
SELECT EMPEVALUATION.EmpNo, EMPLOYEE.EmpName, EMPLOYEE.EmpAddress
FROM EMPLOYEE
RIGHT OUTER JOIN EMPEVALUATION
ON EMPEVALUATION.EmpNo = EMPLOYEE.EmpNo
WHERE EMPEVALUATION.Rating > 70 AND EMPEVALUATION.Rating < 75 AND EMPEVALUATION.RatingPeriod = 'FY2018';
```

```
EmpNo  EmpName            EmpAddress
1      Cara Elyssa Dulay  Manila
3      Agatha Gwen Tagala Paranaque
```

4. Create two solutions/programs for this problem. For the first, use Intersect, for the second, use Join. Display only the employee number of those who were hired after 12/31/2018 whose rating is 90 and above for rating period ‘FY2019’

Intersect

```
SELECT EmpNo
FROM EMPLOYEE
WHERE EMPLOYEE.DateHired >= '2018-12-31'
HAVING EmpNo IN(SELECT EmpNo FROM EMPEVALUATION WHERE EMPEVALUATION.Rating >=90 AND EMPEVALUATION.RatingPeriod = 'FY2019');
```

```
EmpNo
4
NULL
```

Join

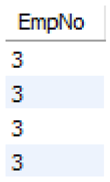
```
SELECT EMPEVALUATION.EmpNo
FROM EMPEVALUATION
INNER JOIN EMPLOYEE
ON EMPEVALUATION.EmpNo = EMPLOYEE.EmpNo
AND EMPEVALUATION.Rating >=90 AND EMPLOYEE.DateHired >= '2018-12-31' AND EMPEVALUATION.RatingPeriod = 'FY2019';
```

```
EmpNo
4
```

5. Display all employee numbers who did not get a rating of 90 in any of the rating periods.

```
SELECT EMPEVALUATION.EmpNo
FROM EMPEVALUATION
WHERE EmpNo NOT IN (SELECT EMPEVALUATION.EmpNo FROM EMPEVALUATION WHERE EMPEVALUATION.Rating >= 90);
```

Guzon, Jeanne Margaret L. COMP 20213 Database Administration
BS Information Technology 3-1



EmpNo
3
3
3
3