

INFS 3770 – Final Exam Answer Sheet
Fall 2020

Name: _____

Exam Score ____ / 60

Visit: www.ersolve.com

Part 1 – 10 Points: Normalization.

Answer/Solve the following questions. *Record your answers below.*

- a. The table is susceptible to modification anomalies. Provide specific examples of insertion, deletion, and update anomalies one might encounter in this data (one of each).

Solution: In this table is not well structured, un-normalised and redundant data are available in the given table. We are analysing the given table for anomalies by using bottom-up approach. In the first observation, in the custName column, pilotSSN and pilotName column multiple entries are available which violates 1NF rule. By assuming pilotSSN and aircraftSign as a candidate key, there are multiple anomalies are exists.

Insertion anomalies: To insert new customer , if we are providing the customer detials, then we must have to enter correct information about the charter, aircraft detials and pilot details. For example if we are inserting the information about the customer Destination Ohio, then one customer can not assign two charter at a time and we can not insert null values for it.

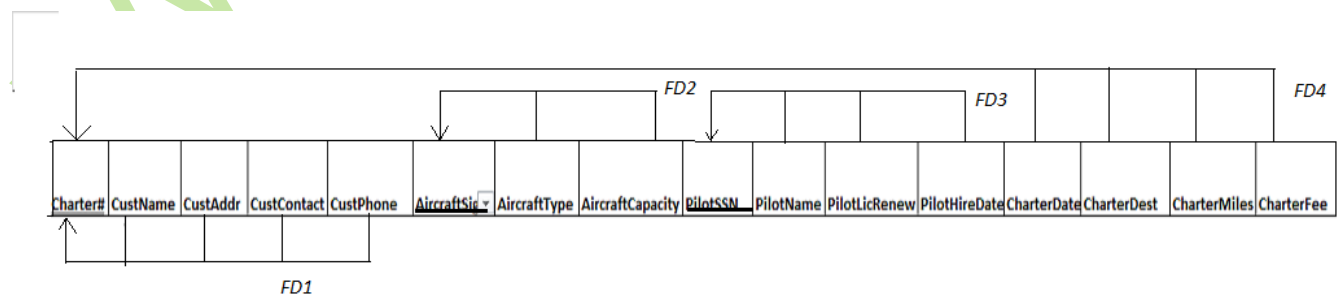
Deletion anomalies: If we want to delete the inoformation of the customer Destination Ohio, then we need to delete 4 records in the present table, and If we will delte four records then 4 charter's reocrd also will need to delete. So due to this deletion, we need to delete multiple records for the maintaining the data integrity.

Updation anomalies: When we want to update the one record of the aircraft, then we need to update many customer records to maintain the integrity of the data, for example if we are updating the aircraft KFS300 then we need to update the customer many customer's records in the database.

- b. Analyze the table structure and determine its state of normalization. Present your rationale and conclusion using the PRECISE methodology demonstrated in the class lecture.

Solution: As per the records in the given table, one patient has scheduled different journey on different -different date, and he is booked different-charter for the different-different destination. All the schedules have been fixed for the customer.

In the 1NF, we removed all the repeated data and assigned new date and charter to the customer which removes the redundancy and improve the integrity of the data.



In the 1NF, we remove all the repeated data, if there is the presence of any primary key or candidate key, then we will figure out the functional dependencies. By using the dependency diagram, we represent the table which is above.

Here FD4 violates 2NF and FD2 and FD3 violates 3NF because of the transitive dependencies. To remove 2NF dependencies we split it into the further tables.

- c. Normalize the data to 3NF. Present your solution in a set of **relational schemas** that meet 3NF requirements.

Solution: In the 3NF, normalization we remove all the transitive data which will be based on the given below,

Normalised form of the table are given below:

Customer:

<u>Customer Id</u>	CustName	CustAddr	custContact	custPhone
--------------------	----------	----------	-------------	-----------

Aircraft:

<u>aircraftSign</u>	aircraftType	aircraftCapacity
---------------------	--------------	------------------

Pilot:

<u>pilotSSN</u>	pilotName	pilotLicRenew	pilotHireDate
-----------------	-----------	---------------	---------------

Charter:

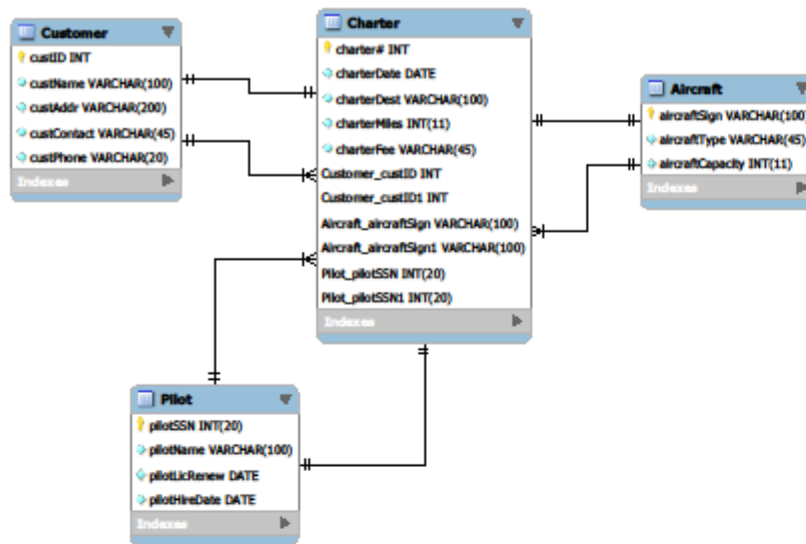
<u>Charter#</u>	charter Date	charter Dest	charter Miles	charter Fee	<u>Customer custID</u>	<u>Aircraft aircraftSign</u>	<u>Aircraft aircraftSign1</u>	<u>Pilot pilotSSN</u>	<u>Pilot pilotSSN1</u>
-----------------	--------------	--------------	---------------	-------------	------------------------	------------------------------	-------------------------------	-----------------------	------------------------

Part 2 –10 Points: ERD Modeling.

Provide a 'readable' screen shot, of your Visio drawing below. Also, upload your drawing (named yourLastName_Part2.pdf) to Blackboard. **Do not submit the .vsdx file!**

Insert a snippet of your Visio ERD here.

Solution: The entity relation diagram is given below, it full fills all the mentioned requirement of the data integrity



Part 3 - 10 points: Create Tables and Insert Data.

Create and name your files using the typical convention: yourLastName_Script.sql, and yourLastName_Spool.txt.

Upload your Script and Spool files to Blackboard.

Solution:

```
-- MySQL Script generated by MySQL Workbench
-- Wed Dec 2 15:23:59 2020
-- Model: New Model   Version: 1.0
-- MySQL Workbench Forward Engineering
```

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERR
OR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';
```

```
-----
-- Schema WeFlyU
-----
```

```
DROP SCHEMA IF EXISTS `WeFlyU` ;
```

```
-----
-- Schema WeFlyU
-----
```

```
CREATE SCHEMA IF NOT EXISTS `WeFlyU` DEFAULT CHARACTER SET utf8 COLLATE utf8_bin ;
SHOW WARNINGS;
USE `WeFlyU` ;
```

```
-----
-- Table `Aircraft`
-----
```

```
DROP TABLE IF EXISTS `Aircraft` ;
```

```
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `Aircraft` (
  `aircraftSign` VARCHAR(100) NOT NULL,
  `aircraftType` VARCHAR(45) NOT NULL,
  `aircraftCapacity` INT(11) NOT NULL,
  PRIMARY KEY (`aircraftSign`))
ENGINE = InnoDB;
```

```
SHOW WARNINGS;
```

```
-----
-- Table `Charter`
-----
```

```
DROP TABLE IF EXISTS `Charter` ;
```

```
SHOW WARNINGS;
```

```
CREATE TABLE IF NOT EXISTS `Charter` (  
  `charter#` INT NOT NULL,  
  `charterDate` DATE NOT NULL,  
  `charterDest` VARCHAR(100) NOT NULL,  
  `charterMiles` INT(11) NOT NULL,  
  `charterFee` VARCHAR(45) NOT NULL,  
  `Customer_custID` INT NOT NULL,  
  `Customer_custID1` INT NOT NULL,  
  `Aircraft_aircraftSign` VARCHAR(100) NOT NULL,  
  `Aircraft_aircraftSign1` VARCHAR(100) NOT NULL,  
  `Pilot_pilotSSN` INT(20) NOT NULL,  
  `Pilot_pilotSSN1` INT(20) NOT NULL,  
  PRIMARY KEY (`charter#`, `Customer_custID`, `Customer_custID1`, `Aircraft_aircraftSign`,  
  `Aircraft_aircraftSign1`, `Pilot_pilotSSN`, `Pilot_pilotSSN1`))  
ENGINE = InnoDB;
```

```
SHOW WARNINGS;
```

```
-----  
-- Table `Customer`  
-----
```

```
DROP TABLE IF EXISTS `Customer` ;
```

```
SHOW WARNINGS;
```

```
CREATE TABLE IF NOT EXISTS `Customer` (  
  `custID` INT NOT NULL,  
  `custName` VARCHAR(100) NOT NULL,  
  `custAddr` VARCHAR(200) NOT NULL,  
  `custContact` VARCHAR(45) NOT NULL,  
  `custPhone` VARCHAR(20) NOT NULL,  
  PRIMARY KEY (`custID`))  
ENGINE = InnoDB;
```

```
SHOW WARNINGS;
```

```
CREATE UNIQUE INDEX `custID_UNIQUE` ON `Customer` (`custID` ASC) VISIBLE;
```

```
SHOW WARNINGS;
```

```
-----  
-- Table `Pilot`  
-----
```

```
DROP TABLE IF EXISTS `Pilot` ;
```

```
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `Pilot` (
  `pilotSSN` INT(20) NOT NULL,
  `pilotName` VARCHAR(100) NOT NULL,
  `pilotLicRenew` DATE NOT NULL,
  `pilotHireDate` DATE NOT NULL,
  PRIMARY KEY (`pilotSSN`))
ENGINE = InnoDB;

SHOW WARNINGS;
CREATE UNIQUE INDEX `pilotSSN_UNIQUE` ON `Pilot` (`pilotSSN` ASC) VISIBLE;
```

```
SHOW WARNINGS;
```

```
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

```
-----
-- Data for table `Aircraft`
-----
```

```
START TRANSACTION;
USE `WeFlyU`;
INSERT INTO `Aircraft` (`aircraftSign`, `aircraftType`, `aircraftCapacity`) VALUES ('DCM6304', 'BE350', 6);
INSERT INTO `Aircraft` (`aircraftSign`, `aircraftType`, `aircraftCapacity`) VALUES ('JMD100', 'Citation
650', 8);
INSERT INTO `Aircraft` (`aircraftSign`, `aircraftType`, `aircraftCapacity`) VALUES ('JXR120', 'Citation 550',
8);
INSERT INTO `Aircraft` (`aircraftSign`, `aircraftType`, `aircraftCapacity`) VALUES ('KFS300', 'BE90', 4);
INSERT INTO `Aircraft` (`aircraftSign`, `aircraftType`, `aircraftCapacity`) VALUES ('XYZ220', 'Falcoln 900',
20);
```

```
COMMIT;
```

```
-----
-- Data for table `Charter`
-----
```

```
START TRANSACTION;
USE `WeFlyU`;
INSERT INTO `Charter` (`charter#`, `charterDate`, `charterDest`, `charterMiles`, `charterFee`,
`Customer_custID`, `Customer_custID1`, `Aircraft_aircraftSign`, `Aircraft_aircraftSign1`,
`Pilot_pilotSSN`, `Pilot_pilotSSN1`) VALUES (1001, '2020-05-03', 'Nashville, TN', 728, '$13,461.00', 1, 1,
'KFS300', 'KFS300', 216231013, 216231013);
```

```
COMMIT;
```

-- Data for table `Customer`

START TRANSACTION;

USE `WeFlyU`;

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (1, 'Nationwide Insurance', '100 Nationwide Blvd, Columbus, OH 43001', 'Katelyn Shanahan', '6146003311');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (2, 'Checkers Distributor', '123 Dussel Dr., Maumee, OH 43557', 'Carol Hall', '4193601220');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (3, 'City of Columbus', '10 State St., Columbus, OH 43061', 'Mitchell Handras', '4402552573');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (4, 'Sunwest Corp.', '439 S. Hampton Ave, Maumee, OH 43537', 'David McElroy', '4192442311');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (5, 'City of Columbus', '10 State St., Columbus, OH 43061', 'Mitchell Handras', '4402552573');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (6, 'Nationwide Insurance', '100 Nationwide Blvd, Columbus, OH 43001', 'Katelyn Shanahan', '6146003311');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (7, 'Martin Associates', '105 Erie St., Toledo, OH 43601', 'James Martin', '4192410000');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (8, 'Checkers Distributor', '123 Dussel Dr., Maumee, OH 43557', 'Carol Hall', '4193601220');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (9, 'Martin Associates', '105 Erie St., Toledo, OH 43601', 'James Martin', '4192410000');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (10, 'Martin Associates', '105 Erie St., Toledo, OH 43601', 'James Martin', '4192410000');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (11, 'Destination Ohio', '4323 Bertrol Ave, Columbus, OH 43014', 'Lesline Simpson', '6144731850');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (12, 'Destination Ohio', '4323 Bertrol Ave, Columbus, OH 43014', 'Lesline Simpson', '6144731850');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (13, 'City of Columbus', '10 State St., Columbus, OH 43061', 'Mitchell Handras', '4402552573');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (14, 'Nationwide Insurance', '100 Nationwide Blvd, Columbus, OH 43001', 'Katelyn Shanahan', '6146003311');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (15, 'Martin Associates', '105 Erie St., Toledo, OH 43601', 'James Martin', '4192410000');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (16, 'Checkers Distributor', '123 Dussel Dr., Maumee, OH 43557', 'Carol Hall', '4193601220');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (17, 'Martin Associates', '105 Erie St., Toledo, OH 43601', 'James Martin', '4192410000');

INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (18, 'Martin Associates', '105 Erie St., Toledo, OH 43601', 'James Martin', '4192410000');

```
INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (19, 'Destination Ohio', '4323 Bertrol Ave, Columbus, OH 43014', 'Lesline Simpson', '6144731850');
INSERT INTO `Customer` (`custID`, `custName`, `custAddr`, `custContact`, `custPhone`) VALUES (20, 'Destination Ohio', '4323 Bertrol Ave, Columbus, OH 43014', 'Lesline Simpson', '6144731850');
```

```
COMMIT;
```

```
-----
-- Data for table `Pilot`
-----
```

```
START TRANSACTION;
USE `WeFlyU`;
INSERT INTO `Pilot` (`pilotSSN`, `pilotName`, `pilotLicRenew`, `pilotHireDate`) VALUES (216841008, 'Hayden Ross', '2022-08-24', '2020-10-24');
INSERT INTO `Pilot` (`pilotSSN`, `pilotName`, `pilotLicRenew`, `pilotHireDate`) VALUES (216950321, 'Maggie Oliver', '2022-09-11', '1988-08-25');
```

```
COMMIT;
```

Part 4 – 30 points: SQL Queries.

For each query problem, Insert your SQL code (text for snip) in the space provided.

1. (1 point) List all rows and all columns of the **ex20Course** table.

Solution:

```
INSERT INTO ex20course(CourseNo, crsDesc, CrsUnits)VALUES ( 'FIN300','FUNDAMENTALS OF FINANCE',4);

INSERT INTO ex20course(CourseNo, crsDesc, CrsUnits)VALUES ( 'FIN450','PRINCIPLES OF INVESTMENTS',4);

INSERT INTO ex20course(CourseNo, crsDesc, CrsUnits)VALUES ( 'FIN480','CORPORATE FINANCE',4);

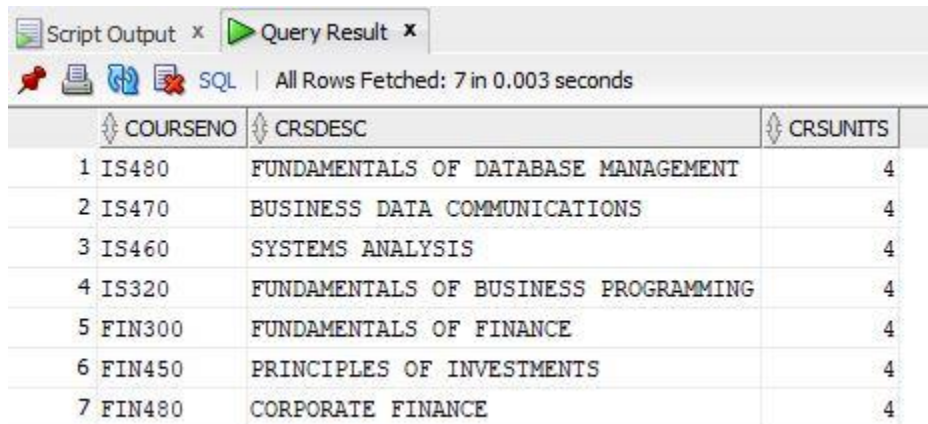
INSERT INTO ex20course(CourseNo, crsDesc, CrsUnits)VALUES ('IS320','FUNDAMENTALS OF BUSINESS PROGRAMMING',4 );

INSERT INTO ex20course(CourseNo, crsDesc, CrsUnits)VALUES ( 'IS460','SYSTEMS ANALYSIS',4);
```


INFS 3770 – Final Exam Answer Sheet

```
INSERT INTO ex20course(CourseNo, crsDesc, CrsUnits) VALUES ( 'IS470','BUSINESS  
DATA COMMUNICATIONS',4);
```

```
INSERT INTO ex20course(CourseNo, crsDesc, CrsUnits)VALUES ( 'IS480','FUNDAMENTALS OF  
DATABASE MANAGEMENT',4 );
```



Script Output x Query Result x

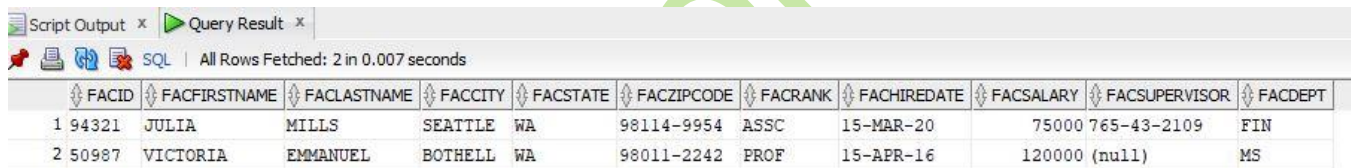
SQL | All Rows Fetched: 7 in 0.003 seconds

	COURSENO	CRSDESC	CRSUNITS
1	IS480	FUNDAMENTALS OF DATABASE MANAGEMENT	4
2	IS470	BUSINESS DATA COMMUNICATIONS	4
3	IS460	SYSTEMS ANALYSIS	4
4	IS320	FUNDAMENTALS OF BUSINESS PROGRAMMING	4
5	FIN300	FUNDAMENTALS OF FINANCE	4
6	FIN450	PRINCIPLES OF INVESTMENTS	4
7	FIN480	CORPORATE FINANCE	4

2. (2 points) List all faculty members with a salary > \$70,000.00 Include the columns facID, facfirstname, faclastname, and facsalary.

Solution:

```
SELECT * FROM ex20faculty WHERE FacSalary > 70000.0;
```



Script Output x Query Result x

SQL | All Rows Fetched: 2 in 0.007 seconds

	FACID	FACFIRSTNAME	FACLASTNAME	FACCITY	FACSTATE	FACZIPCODE	FACRANK	FACHIREDATE	FACSalary	FACSUPERVISOR	FACDEPT
1	94321	JULIA	MILLS	SEATTLE	WA	98114-9954	ASSC	15-MAR-20	75000	765-43-2109	FIN
2	50987	VICTORIA	EMMANUEL	BOTHELL	WA	98011-2242	PROF	15-APR-16	120000	(null)	MS

3. (4 points) List the offerno, offerterm, offyear, courseno, crsdesc, crsunits, offnumenrolled from the ex20Offering and ex20Course tables. Additionally, calculate the tuition using the formula of crsUnits times offnumEnrolled times \$335 (per credit hours). Only display rows where the offnumenrolled is greater than 0. Order the data by offerno.

Solution:

```
SELECT o.offerno,o.offterm, o.offyear,c.courseno, c.crsdesc, c.crsunits,  
o.offnumenrolled,(c.crsunits*o.offnumenrolled*335)as "Tution" FROM ex20offering o,ex20course c where  
o.offnumenrolled>0;
```

Query Result - x

SQL | All Rows Fetched: 42 in 0.01 seconds

	OFFERNO	OFFTERM	OFFYEAR	COURSENO	CRSDESC	CRSUNITS	OFFNUMENROLLED	Tuition
1	9876	SPRING	2020	IS480	FUNDAMENTALS OF DATABASE MANAGEMENT	4	6	8040
2	9876	SPRING	2020	IS470	BUSINESS DATA COMMUNICATIONS	4	6	8040
3	9876	SPRING	2020	IS460	SYSTEMS ANALYSIS	4	6	8040
4	9876	SPRING	2020	IS320	FUNDAMENTALS OF BUSINESS PROGRAMMING	4	6	8040
5	9876	SPRING	2020	FIN300	FUNDAMENTALS OF FINANCE	4	6	8040
6	9876	SPRING	2020	FIN450	PRINCIPLES OF INVESTMENTS	4	6	8040
7	9876	SPRING	2020	FIN480	CORPORATE FINANCE	4	6	8040
8	4321	FALL	2020	IS480	FUNDAMENTALS OF DATABASE MANAGEMENT	4	5	6700
9	4321	FALL	2020	IS470	BUSINESS DATA COMMUNICATIONS	4	5	6700
10	4321	FALL	2020	IS460	SYSTEMS ANALYSIS	4	5	6700

4. (3 points) List Faculty whose salary (facsalary) is greater than the average salary of all faculty. Include faculty name, facSalary and the avgSalary (of all faculty).

Solution:

```
select f.facid, CONCAT(CONCAT(f.facfirstname, ' '), f.faclastname) as "name", f.facsalary, AVG(f.facsalary) AS "Avg Salary" from ex20faculty f where f.facsalary > AVG(f.facsalary) group by f.facid;
```

5. (6 points) List any Faculty member that was not scheduled to teach a course (No offering). Display the facid, facfirstname, faclastname columns from the ex20Faculty table.

Method 1: IN with Nested Query

Method 2: Outer Join

```
SELECT f.facid, f.facfirstname, f.faclastname FROM ex20faculty f WHERE f.facid not in (select courseno from ex20course);
```

Script Output x Query Result x

SQL | All Rows Fetched: 6 in 0.01 seconds

	FACID	FACFIRSTNAME	FACLASTNAME
1	94321	JULIA	MILLS
2	83210	CRISTOPHER	COLAN
3	05432	LEONARD	VINCE
4	61098	LEONARD	FIBON
5	72109	NICKI	MACON
6	50987	VICTORIA	EMMANUEL

```
SELECT f.facid, f.facfirstname, f.faclastname FROM ex20faculty f FULL OUTER JOIN ex20course c ON c.courseno = f.facid ORDER BY f.facfirstname;
```

Script Output x Query Result x

SQL | All Rows Fetched: 6 in 0.01 sec

	FACID	FACFIRSTNAME	FACLASTNAME
1	94321	JULIA	MILLS
2	83210	CRISTOPHER	COLAN
3	05432	LEONARD	VINCE
4	61098	LEONARD	FIBON
5	72109	NICKI	MACON
6	50987	VICTORIA	EMMANUEL

6. (2 points) Write the SQL statement to modify the data in the ex20Faculty table to update the salary of professors with a rank of "PROF" by 10%. Then, identify the SQL statement would you use to save the changes made by the query.

Solution:

```
UPDATE ex20faculty SET facsalary = round(100*(count(*) / sum(count(*) over ()),2)
WHERE facrank = 'prof' ;
```

7. (3 points) List students who have registered for Spring 2020 and the count of how many courses they have registered for. Only display the students enrolled in 3 or more courses. Include the stdID, first and last name, stdMajor and the count of enrolled courses. Order the data by last name.

```
select s.stdid,s.stdfirstname,s.stdlastname, s.stdmajor from ex20student s where s.stdid = (select courseno from ex20coursecourse);
```

8. (3 points) List **ALL** faculty members (facID, firstname and lastname) and count how many courses they have taught (ex20Offering) and count how many (different) terms they have taught in (also in ex20Offering). Order the result set by last name.

```
SELECT f.facid, f.facfirstname, f.faclastname FROM ex20faculty f WHERE f.facid in (select ex20course.courseno from ex20course);
```

9. (6 points) Create a View named "**exp9**" for the following query definition. List courses that have been offered with a count of how many students (across all terms) were enrolled and a count of how many faculty were assigned to teach the

courses (offerings). **Include the code to create the view** – and the code needed to execute your view to display the result set.

```
SELECT c.courseno, c.coursedesc, s.stdcount FROM ex20coursecourse c, ex20studentf;
```

www.ersolve.com