Demystifying SQL for Internal Auditors

Kate Head, CPA, CFE, CISA, CIG
University of South Florida

Joselyn De La Cruz-Rameau, Ed. D./ET
University of Texas-San Antonio

1

1

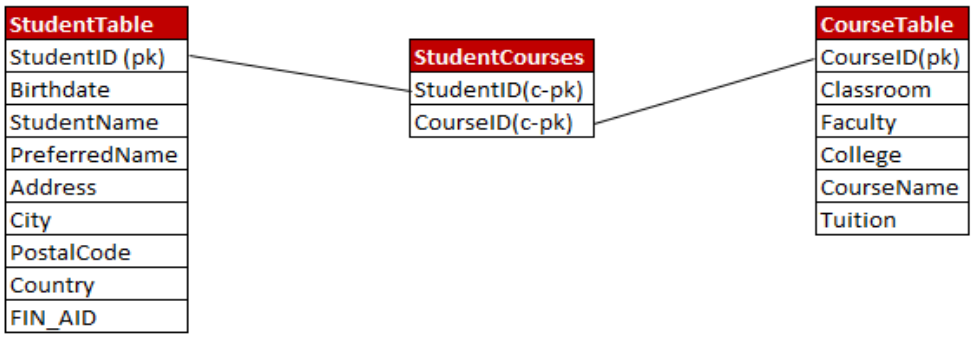
Topics of Breakout (Toad for Oracle)

- CREATE TABLE IN SQL (Data Types)
- INPUT RECORDS IN SQL
- DROP/TRUNCATE
- ALIAS
- RELATIONSHIPS
  - One-to-One, One-to-Many, Many-to-Many
- JOINS

2

2

Database Tables



3

3

CREATE Student Table

```
CREATE TABLE Student
(
STUDENTID           VARCHAR2(1 CHAR) PRIMARY KEY,
BIRTHDATE            DATE,
STUDENTNAME         VARCHAR2(55 CHAR),
PREFERREDNAME       VARCHAR2(50 CHAR),
ADDRESS             VARCHAR2(125 CHAR),
CITY                VARCHAR2(75 CHAR),
POSTALCODE          VARCHAR2(10 CHAR),
COUNTRY             VARCHAR2(35 CHAR),
FIN_AID             DECIMAL(15,2),
CONSTRAINT  pk_studentid PRIMARY KEY (StudentID)
)
;
```

4

4

CREATE StudentCourses Table

```
CREATE TABLE StudentCourses
(
STUDENTID           VARCHAR2(1 CHAR) PRIMARY KEY,
COURSEID            VARCHAR2 (10 CHAR) PRIMARY KEY ,
CONSTRAINT PK_STUDENTCOURSES PRIMARY KEY (STUDENTID, COURSEID) ,
CONSTRAINT FK_STUDENTID FOREIGN KEY (STUDENTID) REFERENCES STUDENT (STUDENTID) ,
CONSTRAINT FK_COURSETABLE FOREIGN KEY (COURSEID) REFERENCES COURSETABLE (COURSEID)
)
;
```

5

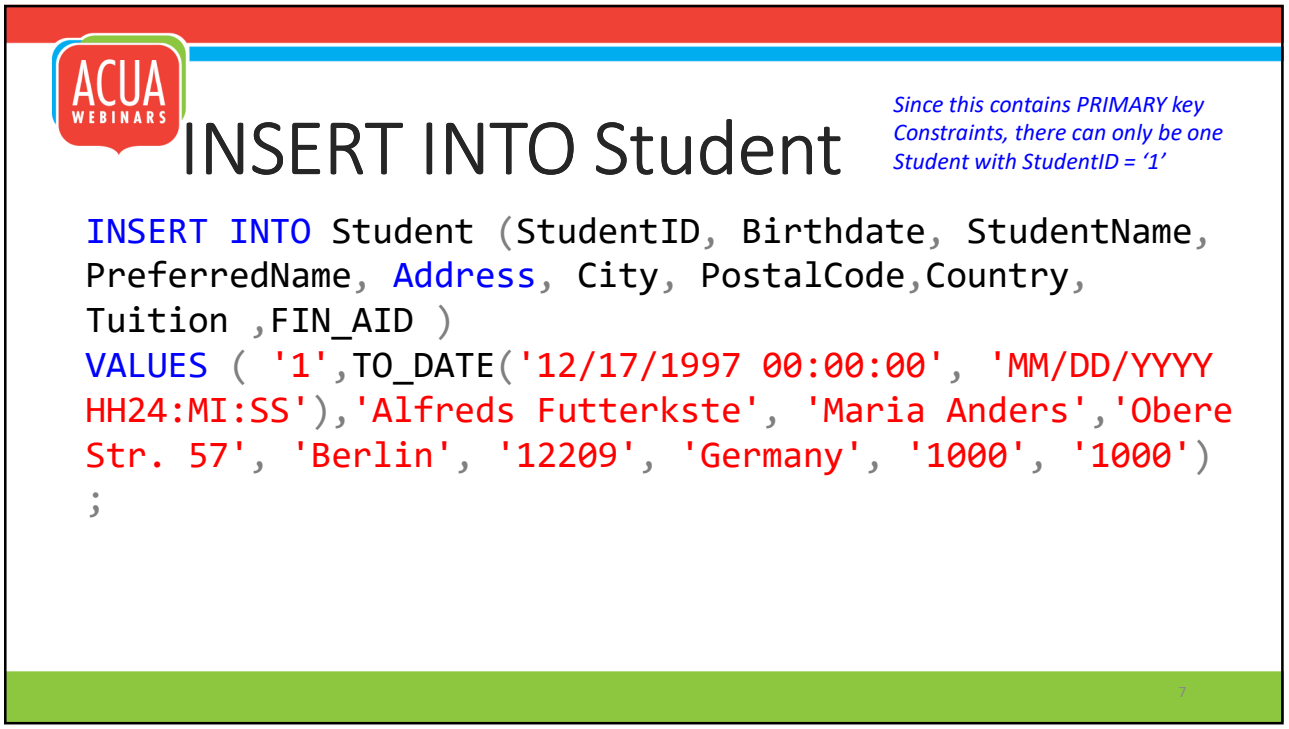
5

CREATE CourseTable

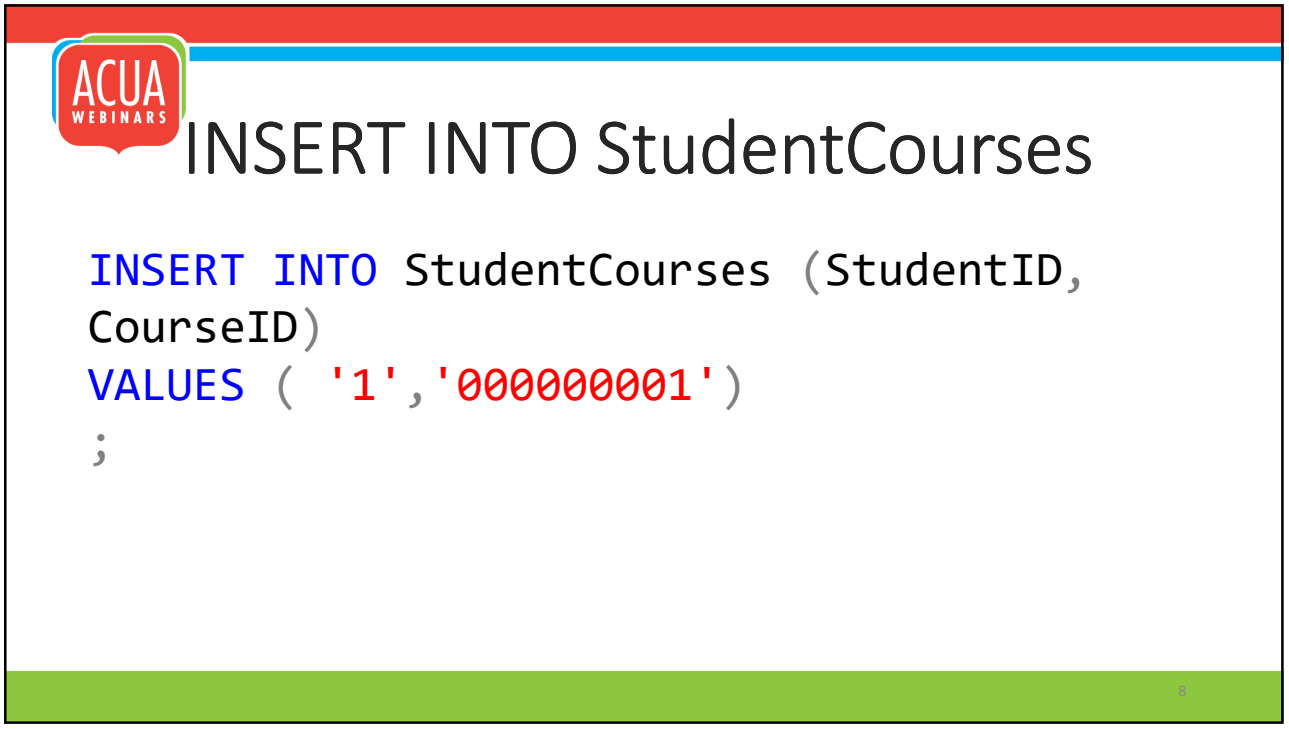
```
CREATE TABLE CourseTable
(
COURSEID            VARCHAR2 (10 CHAR),
CLASSROOM           VARCHAR2 (4 CHAR),
FACULTY             VARCHAR2 (75 CHAR),
COLLEGE             VARCHAR2 (75 CHAR),
COURSENAME          VARCHAR2   (125 CHAR),
TUITION             DECIMAL (10,2) , --DECIMAL(precision, scale)
CONSTRAINT  pk_COURSEID PRIMARY KEY (COURSEID)
)
;
```

6

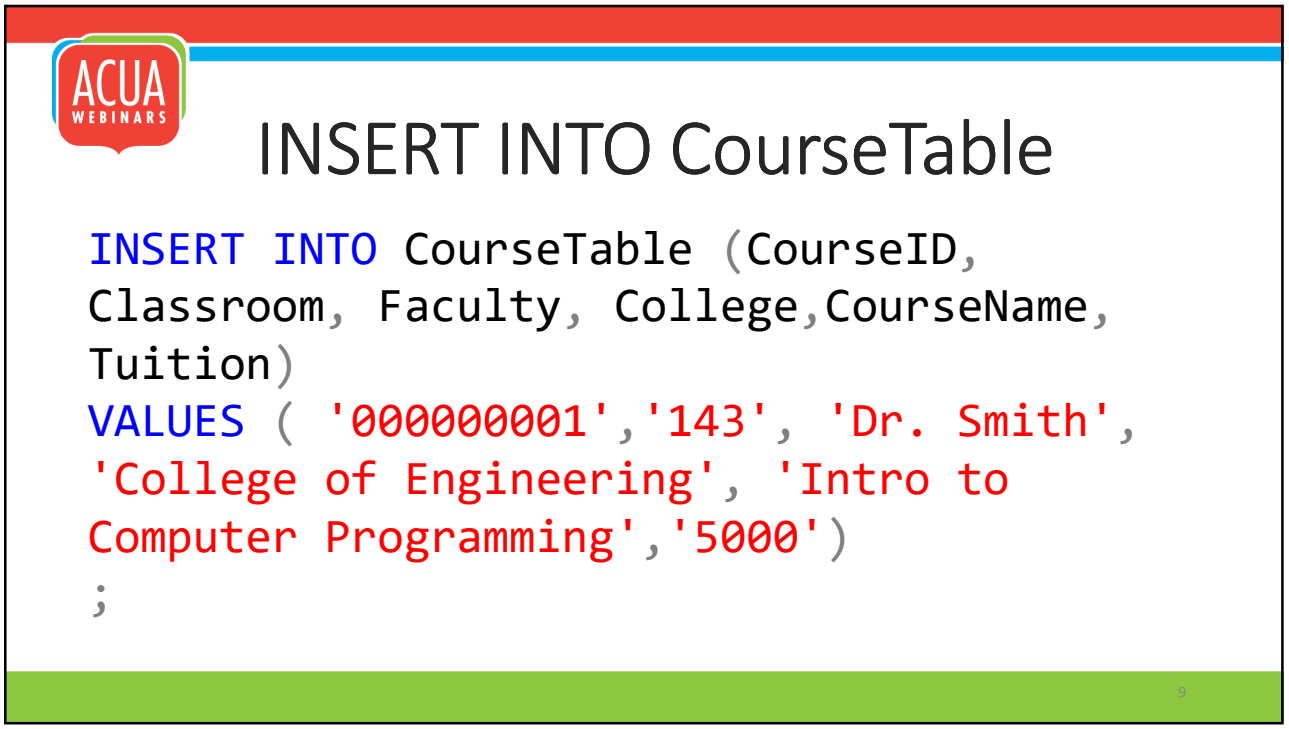
6


ACUA
WEBINARS
INSERT INTO Student
Since this contains PRIMARY key Constraints, there can only be one Student with StudentID = '1'
INSERT INTO Student (StudentID, Birthdate, StudentName, PreferredName, Address, City, PostalCode,Country, Tuition ,FIN_AID)
VALUES ('1',TO_DATE('12/17/1997 00:00:00', 'MM/DD/YYYY HH24:MI:SS'),'Alfreds Futterkste', 'Maria Anders','Obere Str. 57', 'Berlin', '12209', 'Germany', '1000', '1000')
;
7

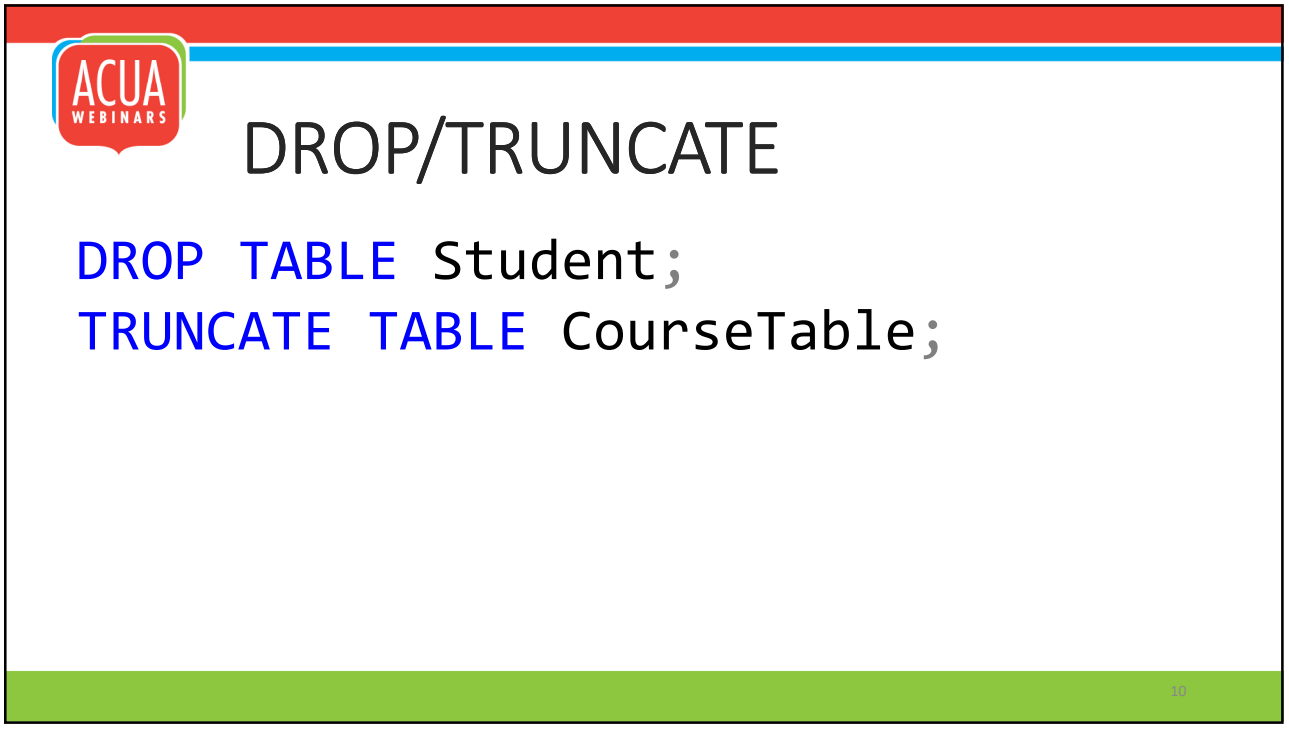
7


ACUA
WEBINARS
INSERT INTO StudentCourses
INSERT INTO StudentCourses (StudentID, CourseID)
VALUES ('1','000000001')
;
8

8


ACUA
WEBINARS
INSERT INTO CourseTable
INSERT INTO CourseTable (CourseID, Classroom, Faculty, College,CourseName, Tuition)
VALUES ('000000001','143', 'Dr. Smith', 'College of Engineering', 'Intro to Computer Programming','5000')
;
9

9


ACUA
WEBINARS
DROP/TRUNCATE
DROP TABLE Student;
TRUNCATE TABLE CourseTable;
10

10

ALIAS

```
SELECT StudentID, StudentName, PreferredName,
Address, City, PostalCode, Country, Tuition
FROM Student ST --the alias is ST
GROUP BY StudentID, StudentName, PreferredName,
Address, City, PostalCode, Country, Tuition
HAVING StudentID >1
ORDER BY StudentName DESC;
```

11

11

RETRIEVE RECORDS

```
SELECT * FROM Student;
SELECT * FROM CourseTable;
SELECT * FROM StudentCourses;
```

12

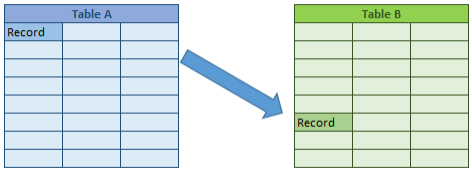
12

Understand how to write and document SQL Code (Table Relationships)

Table Relationships

A *one-to-one* (1:1)

Relationship means that each record in Table A relates to one, and only one, record in Table B, and each record in Table B relates to one, and only one, record in Table A.



13

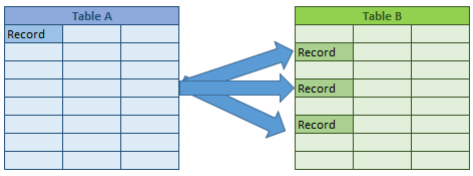
13

Understand how to write and document SQL Code (Table Relationships)

Table Relationships

A *one-to-many* (1:N)

Relationship means that each record instance in Table A, there exists zero, one, or many instances in Table B.



14

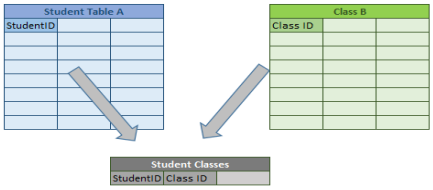
14

Understand how to write and document SQL Code (Table Relationships)

Table Relationships

Many-to-many (N:N) relationship.

Relationship in which multiple instances on Table A exist in with multiple instances in Table B. This constitutes a *many-to-many* (N:N) relationship.



15

15

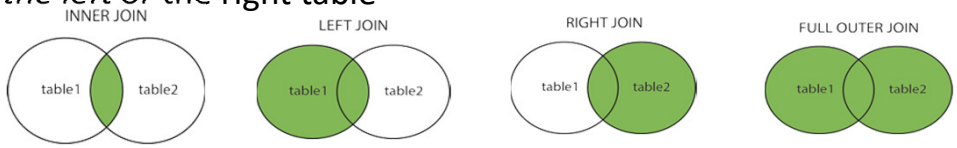
JOINS

16

16

Understand how to write and document SQL Code

- **Inner Join:** returns records that are matching in both tables
- **Left (Outer) Join:** Returns all records from the left table and only the matching from the right table.
- **Right (Outer) Join:** Returns all records from the right table, and the matched records from the left table.
- **Full (Outer) Join:** Returns all records when there is a match in either the left or the right table



17

17

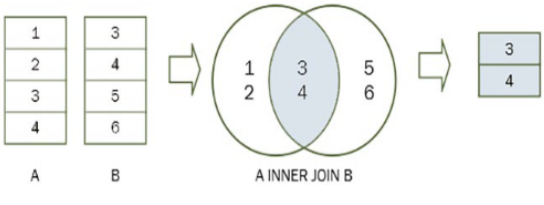
Understand how to write and document SQL Code

- **Inner Join:**

SQL INNER JOIN (sometimes called SIMPLE JOIN)

The Inner Join only returns data that appears in both data sets. With the above example of student and course tables, this join would return only students who have enrolled in courses, thus have a record in the Student table and one or more records in the Course table.

```
SELECT <select_list>
FROM Table_A A
INNER JOIN Table_B B
ON A.Key = B.Key
```



18

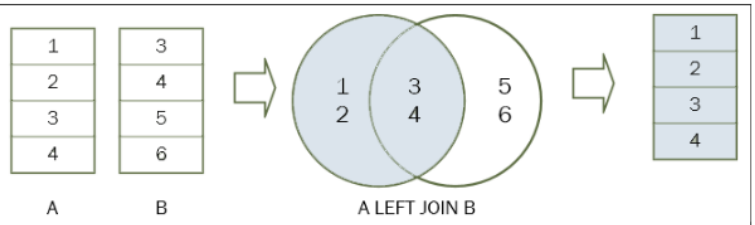
18

Understand how to write and document SQL Code

SQL LEFT OUTER JOIN (sometimes called LEFT JOIN)

The Left Outer Join returns all data in table A and only the matching records from table B. In our example, this Join would return all students in the student table and matching course information for the students in the student table. In other words, the results would list all students, but if a student was not assigned to any courses the course-related fields in the output would be blank.

```
SELECT
<select_list>
FROM Table_A A
LEFT JOIN
Table_B B
ON A.Key =
B.Key
```



19

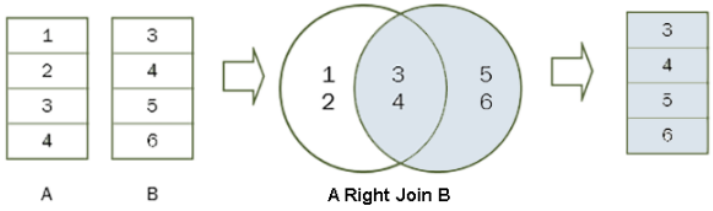
19

Understand how to write and document SQL Code

SQL RIGHT OUTER JOIN (sometimes called RIGHT JOIN)

The Right Outer Join returns all data in table B and the matching records from table A. This Join would return all courses from the course table and matching student information for the courses in the courses table. In other words, the results would list all courses, but if there were not students in a course the student-related fields in the output would be blank.

```
SELECT
<select_list>
FROM Table_A A
RIGHT JOIN
Table_B B
ON A.Key =
B.Key
```



20

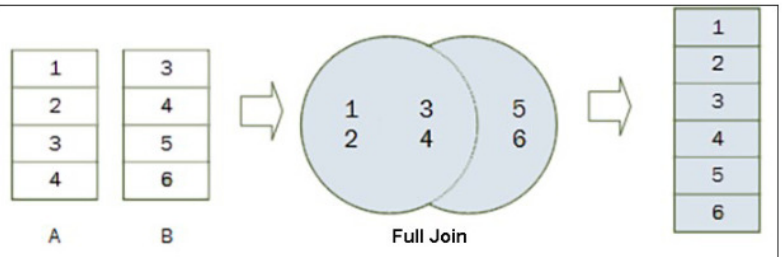
20

Understand how to write and document SQL Code

SQL FULL OUTER JOIN (sometimes called FULL JOIN)

The Full Outer Join returns all data in both table A and table B. This Join would return all students from the student table and courses from the course table. It is important to understand that relationships (one-to-one, one-to-many, many-to-many) differ from Joins.

```
SELECT
<select_list>
FROM Table_A A
FULL OUTER
JOIN Table_B B
ON A.Key =
B.Key
```



21

21

Resources

1. Getting Started with SQL for Data Analysis Kick Starter(To be published by ACUA)
2. Aggregate Functions
https://docs.microsoft.com/en-us/sql/t-sql/functions/aggregate-functions-transact-sql?view=sql-server-ver15
3. SQL: Joins
https://www.techonthenet.com/sql/joins.php
4. SQL Tutorial
https://www.w3schools.com/sql/
5. Structured Query Language
https://www.computerworld.com/article/2595492/structured-query-language.html
6. Visual Representation of SQL Joins. The Code Project
https://www.codeproject.com/Articles/33052/Visual-Representation-of-SQL-Joins
7. Taylor, A. G. (2019). SQL for Dummies (9th Edition). ISBN 978-1-119-52707-7
8. Forta, B. (2019). Sams Teach Yourself SQL in 10 minutes (5th Edition); ISBN-13: 978-0-13-518279-6; ISBN-10: 0-13-518279-4
9. Stephens, R., Plew, R. and Jones, A.D. (2013). Sams Teach Yourself SQL in 24 hours (5th Edition); ISBN-13: 978-0-372-33541-9; ISBN-10: 0-672-33541-7

22

22

ACUA Data Analytics Team

Data Analytics Strategic Chair
Tiffany Yordan
tyordan@princeton.edu

Data Analytics Faculty Chair
Joselyn De La Cruz- Rameau
Joselyn.Rameau@utsa.edu

Data Analytics Kick Starter Chair
Christine Heise
Christine.Heise@usnh.edu

23

23

Thank You

Kate Head, CPA, CFE, CISA, CIG	**Joselyn De La Cruz-Rameau, Ed. D./ET**
University of South Florida Office of Internal Audit (813) 974-3737 Khead@usf.edu	University of Texas-San Antonio Audit and Consulting (210) 458-4237 joselyn.rameau@utsa.edu

24

24