Instructor Teaching Tracker

Brandon R. Russell

University of Maryland University College

Contents

[Statement of Work 2](#_Toc434172602)

[Entity Relationship Diagram 3](#_Toc434172603)

[Table Design 4](#_Toc434172604)

[Data Definition 5](#_Toc434172605)

[Insert Data 12](#_Toc434172606)

# Statement of Work

1. This database supports the Instructor Teaching Tracker application. It keeps track of Instructor, course, and section data, as well as classes taught and schedule. The purpose of this is to assist Instructors with scheduling/maintaining classes in an efficient manner.

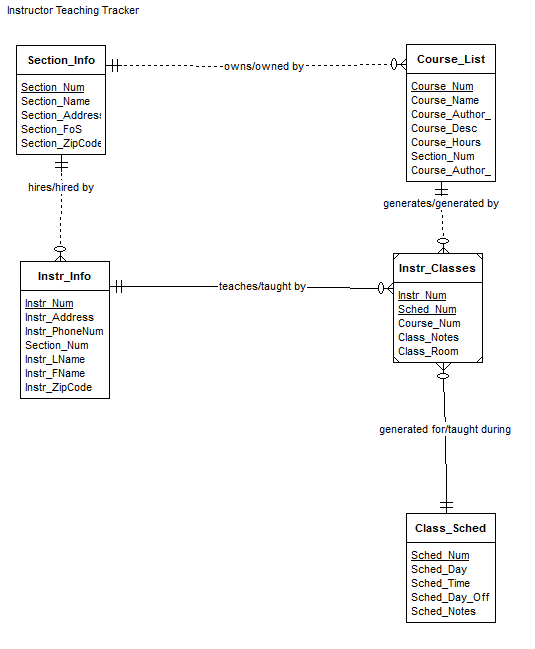
The Instructor table has the name, address, phone number and section number. The course list has the course name, author, description, hours and section number. The section table has the section number, name, address and field of study. The Instructor Classes table shows the Instructor, schedule, course, room number and Instructor notes. The class schedule gives the day/time a class is taught, a down day schedule and notes.

For this application a section can have many Instructors, but an Instructor can only belong to one section. A section can control many courses, but a course can belong to only one section. An instructor can teach many classes and a class can be taught by many instructors; this is handled with an associative entity. Many classes can be generated from one course, but each class can only be generated from a single course. One class can have only one schedule, while many classes can have the same schedule.

Assumptions: This application tracks classes taught by instructors and includes a class schedule. It is assumed classes will always meet a specific schedule criteria of Monday-Wednesday-Friday, Tuesday-Thursday, or Saturday. Down days are tracked for individual classes since this will vary based upon the specific class.

1. Database: Oracle 11g on Windows platform.
2. Software: ER Assistant, Oracle SQL developer, and Oracle SQL developer data modeler.
3. DDL – drop and create the database objects including tables, views, triggers, and sequences. DML – populate and query data.

# Entity Relationship Diagram



# Table Design

|  |  |  |
| --- | --- | --- |
| Section\_Info | | |
| Section\_Num | INTEGER | Not null - pk |
| Section\_Name | VARCHAR(100) |  |
| Section\_Address | VARCHAR(100) |  |
| Section\_FoS | VARCHAR(100) |  |
| Section\_ZipCode | VARCHAR(10) |  |

|  |  |  |
| --- | --- | --- |
| Course\_List | | |
| Course\_Num | INTEGER | Not null – pk |
| Course\_Name | VARCHAR(100) |  |
| Course\_Author\_FName | VARCHAR(20) |  |
| Course\_Author\_LName | VARCHAR(20) |  |
| Course\_Desc | VARCHAR(1000) |  |
| Course\_Hours | INTEGER |  |
| Section\_Num | INTEGER | Not null - fk |

|  |  |  |
| --- | --- | --- |
| Instr\_Info | | |
| Instr\_Num | INTEGER | Not null – pk |
| Instr\_Address | VARCHAR(100) |  |
| Instr\_PhoneNum | VARCHAR(20) |  |
| Section\_Num | INTEGER | Not null - fk |
| Instr\_LName | VARCHAR(20) |  |
| Instr\_FName | VARCHAR(20) |  |
| Instr\_ZipCode | VARCHAR(10) |  |

|  |  |  |
| --- | --- | --- |
| Class\_Sched | | |
| Sched\_Num | INTEGER | Not null - pk |
| Sched\_Day | VARCHAR(100) |  |
| Sched\_Time | VARCHAR(10) |  |
| Sched\_Day\_Off | VARCHAR(100) |  |
| Sched\_Notes | VARCHAR(1000) |  |

|  |  |  |
| --- | --- | --- |
| Instr\_Classes | | |
| Instr\_Num | INTEGER | Not null – pk/fk |
| Sched\_Num | INTEGER | Not null – pk/fk |
| Course\_Num | INTEGER | Not null - fk |
| Class\_Notes | VARCHAR(1000) |  |
| Class\_Room | VARCHAR(5) |  |

# Data Definition

SET echo on

SET serveroutput on

**/\* Drop tables, sequence, and other objects you create\*/**

DROP TABLE Instr\_Classes;

Table INSTR\_CLASSES dropped.

DROP TABLE Instr\_Info;

Table INSTR\_INFO dropped.

DROP TABLE Course\_List;

Table COURSE\_LIST dropped.

DROP TABLE Section\_Info;

Table SECTION\_INFO dropped.

DROP TABLE Class\_Sched;

Table CLASS\_SCHED dropped.

DROP SEQUENCE SectionNum\_Seq;

Sequence SECTIONNUM\_SEQ dropped.

DROP SEQUENCE InstrNum\_Seq;

Sequence INSTRNUM\_SEQ dropped.

DROP SEQUENCE CourseNum\_Seq;

Sequence COURSENUM\_SEQ dropped.

DROP SEQUENCE SchedNum\_Seq;

Sequence SCHEDNUM\_SEQ dropped.

**/\* Create 5 tables \*/**

CREATE TABLE Section\_Info

(

Section\_Num INTEGER NOT NULL,

Section\_Name VARCHAR (100),

Section\_Address VARCHAR (100),

Section\_FoS VARCHAR (100),

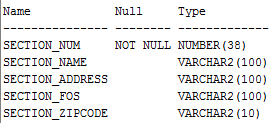
Section\_ZipCode VARCHAR(10),

CONSTRAINT pk\_section PRIMARY KEY (Section\_Num)

);

Table SECTION\_INFO created.

DESCRIBE Section\_Info;



CREATE TABLE Instr\_Info

(

Instr\_Num INTEGER NOT NULL,

Instr\_Address VARCHAR (100),

Instr\_PhoneNum VARCHAR (20),

Section\_Num INTEGER NOT NULL,

Instr\_LName VARCHAR (20),

Instr\_FName VARCHAR (20),

Instr\_ZipCode VARCHAR (10),

CONSTRAINT pk\_Instr PRIMARY KEY (Instr\_Num),

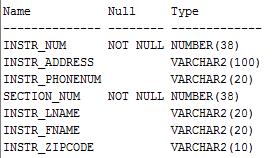
CONSTRAINT fk\_Instr\_Section FOREIGN KEY (Section\_Num)

REFERENCES Section\_Info

);

Table INSTR\_INFO created.

DESCRIBE Instr\_Info;



CREATE TABLE Course\_List

(

Course\_Num INTEGER NOT NULL,

Course\_Name VARCHAR (100),

Course\_Author\_FName VARCHAR (20),

Course\_Author\_LName VARCHAR(20),

Course\_Desc VARCHAR (1000),

Course\_Hours INTEGER,

Section\_Num INTEGER NOT NULL,

CONSTRAINT pk\_Course PRIMARY KEY (Course\_Num),

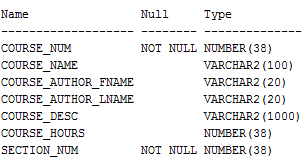
CONSTRAINT fk\_Course\_Section FOREIGN KEY (Section\_Num)

REFERENCES Section\_Info

);

Table COURSE\_LIST created.

DESCRIBE Course\_List;



CREATE TABLE Class\_Sched

(

Sched\_Num INTEGER NOT NULL,

Sched\_Day VARCHAR (100),

Sched\_Time VARCHAR (10),

Sched\_Day\_Off VARCHAR (100),

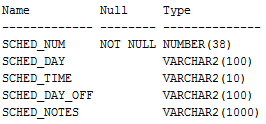
Sched\_Notes VARCHAR (1000),

CONSTRAINT pk\_Schedule PRIMARY KEY (Sched\_Num)

);

Table CLASS\_SCHED created.

DESCRIBE Class\_Sched;



CREATE TABLE Instr\_Classes

(

Instr\_Num INTEGER NOT NULL,

Sched\_Num INTEGER NOT NULL,

Course\_Num INTEGER NOT NULL,

Class\_Notes VARCHAR (1000),

Class\_Room VARCHAR (5),

CONSTRAINT pk\_Class PRIMARY KEY (Instr\_Num, Sched\_Num),

CONSTRAINT fk\_Instr FOREIGN KEY (Instr\_Num)

REFERENCES Instr\_Info

ON DELETE CASCADE,

CONSTRAINT fk\_Sched FOREIGN KEY (Sched\_Num)

REFERENCES Class\_Sched

ON DELETE CASCADE,

CONSTRAINT fk\_Course FOREIGN KEY (Course\_Num)

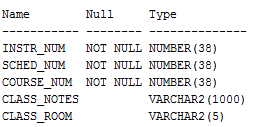
REFERENCES Course\_List

ON DELETE CASCADE

);

Table INSTR\_CLASSES created.

DESCRIBE Instr\_Classes;



**/\* Create indexes on foreign keys\*/**

CREATE INDEX fk\_Course\_List on Course\_List(Section\_Num);

Index FK\_COURSE\_LIST created.

CREATE INDEX fk\_Instr on Instr\_Info(Section\_Num);

Index FK\_INSTR created.

CREATE INDEX fk\_InstrClass on Instr\_Classes(Instr\_Num);

Index FK\_INSTRCLASS created.

CREATE INDEX fk\_ClassSched on Instr\_Classes(Sched\_Num);

Index FK\_CLASSSCHED created.

CREATE INDEX fk\_CourseClass on Instr\_Classes(Course\_Num);

Index FK\_COURSECLASS created.

**/\* Create 2 views \*/**

**/\*This view shows all classes Instructor 1 is teaching on Schedule 1 \*/**

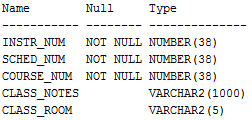
CREATE OR REPLACE VIEW Instr\_Teaching\_View AS

SELECT \* from Instr\_Classes

WHERE Instr\_Num ='1' AND Sched\_Num ='1';

View INSTR\_TEACHING\_VIEW created.

DESCRIBE Instr\_Teaching\_View;



**/\*This view will show which section each Instructor belongs to\*/**

CREATE OR REPLACE VIEW Section\_Instructors\_View AS

SELECT section\_num, s.section\_name, i.instr\_fname, i.instr\_lname

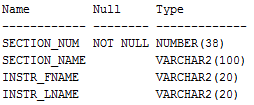
FROM section\_info s

INNER JOIN instr\_info i

USING (section\_num);

View SECTION\_INSTRUCTORS\_VIEW created.

DESCRIBE Section\_Instructors\_View;



**/\* Create trigger \*/**

**/\*This trigger will display a message when a row is added to Instr\_Classes\*/**

CREATE OR REPLACE TRIGGER SchedClass\_Trigger AFTER INSERT ON Instr\_Classes

FOR EACH ROW

BEGIN

dbms\_output.put\_line ('A class has been added!!');

END;

/

Trigger SCHEDCLASS\_TRIGGER compiled

**/\* Create sequence\*/**

CREATE SEQUENCE SectionNum\_Seq

START WITH 1

INCREMENT BY 1;

Sequence SECTIONNUM\_SEQ created.

CREATE SEQUENCE InstrNum\_Seq

START WITH 1

INCREMENT BY 1;

Sequence INSTRNUM\_SEQ created.

CREATE SEQUENCE CourseNum\_Seq

START WITH 1

INCREMENT BY 1;

Sequence COURSENUM\_SEQ created.

CREATE SEQUENCE SchedNum\_Seq

START WITH 1

INCREMENT BY 1;

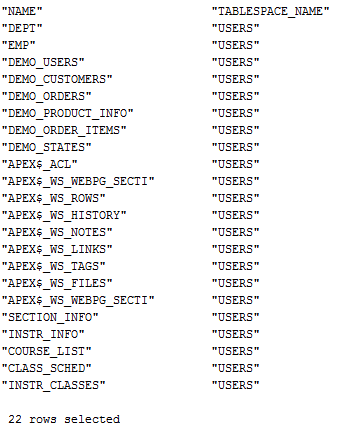
Sequence SCHEDNUM\_SEQ created.

**/\* Data Dictionary query \*/**

PURGE recyclebin;

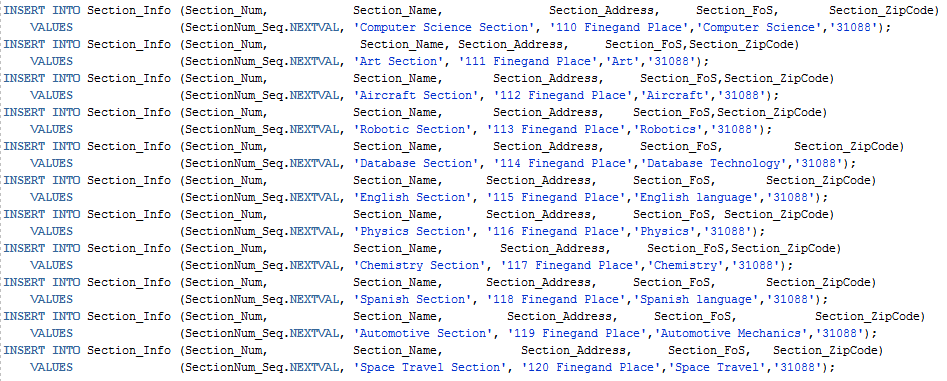
**/\*The next query returns the first 20 characters of all table names and then their tablespace names\*/**

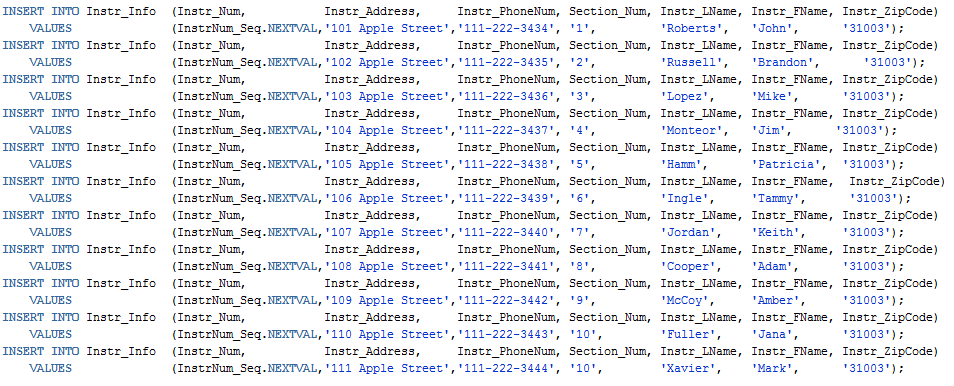
SELECT /\*fixed\*/ substr(table\_name, 1,20) as name, tablespace\_name from user\_tables;

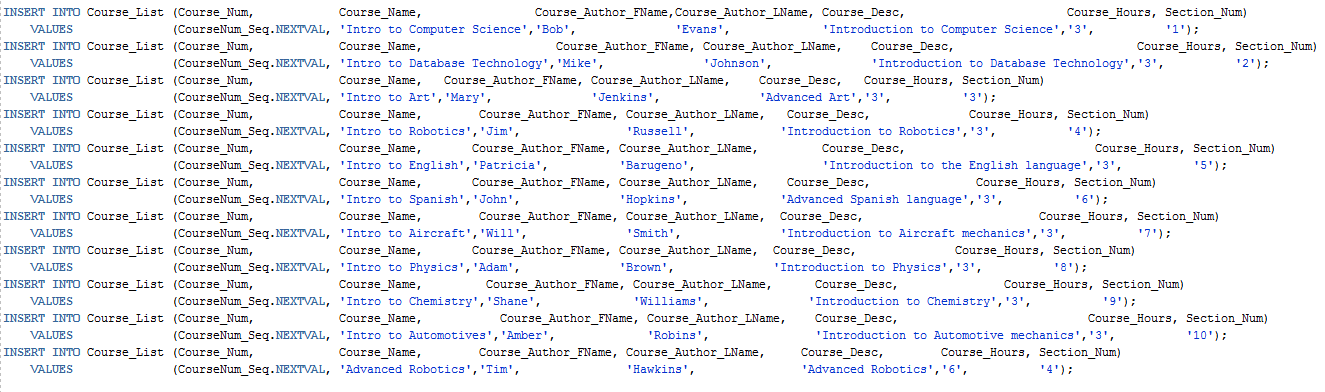


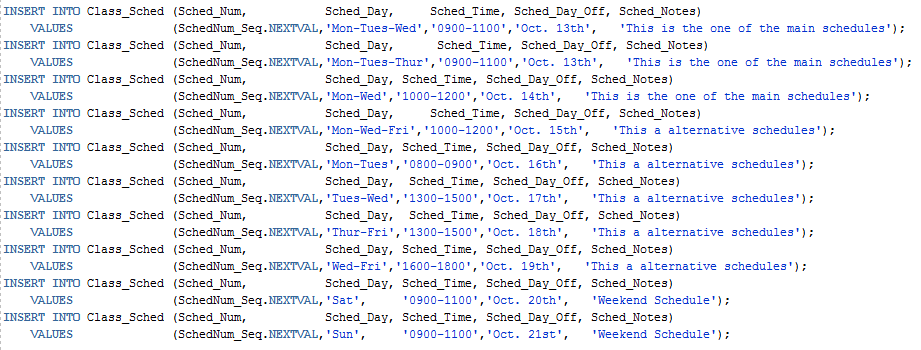
# Insert Data

**/\* Insert 10 or more rows into each table \*/**









INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('1', '1', '1', 'Computer one is broke in this room','A');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('2', '2', '2', 'Room Ready','B');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('3', '3', '3', 'The Instructor Chair is broke','C');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('4', '4', '4', 'A/C is not working','D');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('5', '5', '5', 'Computer two is broke in this room','E');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('6', '6', '6', 'No whiteboard in this room','F');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('7', '7', '7', 'Student station 3 is missing a keyboard','G');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('8', '8', '8', 'Room Ready','H');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('9', '9', '9', 'Room Ready','I');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('10', '10', '10', 'Room Ready','J');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('10', '9', '10', 'Room Ready','J');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('9', '8', '10', 'Room Ready','X');

1 row inserted.

A class has been added!!

INSERT INTO Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

VALUES ('1', '2', '9', 'Room Ready','Z');

1 row inserted.

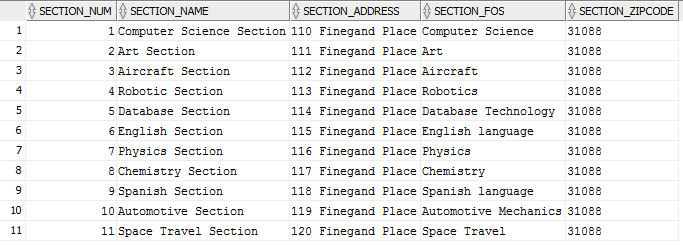
A class has been added!!

commit;

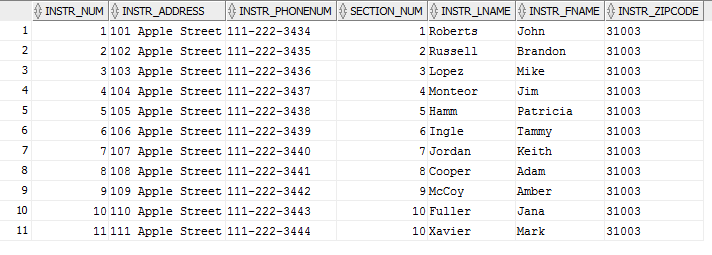
Commit complete.

**/\* Verify that each table has 10 or more rows of data \*/**

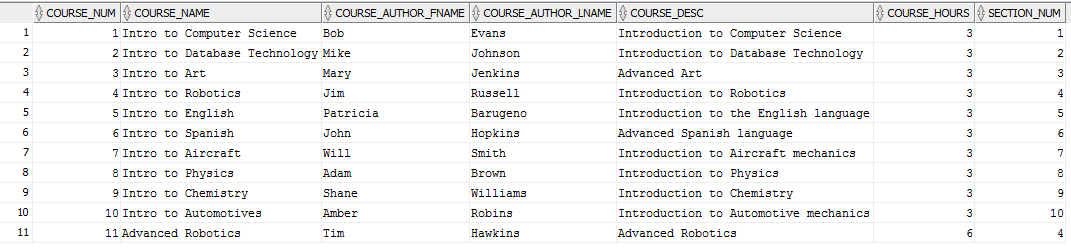
SELECT /\*fixed\*/ \* FROM Section\_Info;



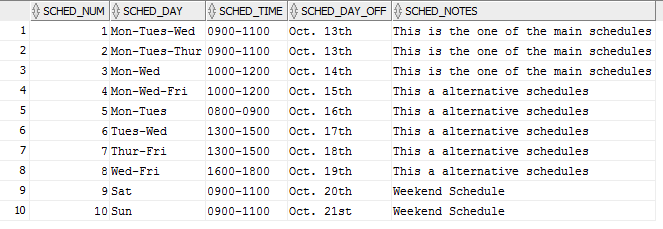
SELECT /\*fixed\*/ \* FROM Instr\_Info;



SELECT /\*fixed\*/ \* FROM Course\_List;



SELECT /\*fixed\*/ \* FROM Class\_Sched;



SELECT /\*fixed\*/ \* FROM Instr\_Classes;

