Instructor Teaching Tracker: Security Project

Brandon R. Russell

University of Maryland University College

Abstract

If all we had were single-user offline database systems, the only security we would ever need would be physical security. For a time, long before the Internet existed, physical security was cybersecurity. Today, security is a critical part of software and database development. Before we can implement security, we need a plan. A security plan is the first step in implementing security and serves as a guide for implementation. After the plan is established, security policies are created to provide a finer level of detail on what needs to be protected and/or locked down. Once we have sufficient policies, we create procedures for each policy to implement to policy in the database. In this project we will see the description of the problem (lack of security in a production database), the creation of the necessary conceptual, logical, and physical models, development of a project timeline in the form of a Gantt chart, the implementation of the database definition and manipulation scripts, and the implementation of data control measures: Views, Virtual Private Database policies, and Oracle Label Security.

Keywords: Database, Instructor, Teaching, Tracker, Security, Oracle, Views, Virtual Private Database, Oracle Label Security

Contents

[Abstract 2](#_Toc529625671)

[Instructor Teaching Tracker: Security Project 6](#_Toc529625672)

[Statement of Work 6](#_Toc529625673)

[Database Name 6](#_Toc529625674)

[Database Description 6](#_Toc529625675)

[The Problem 7](#_Toc529625676)

[Security Plan 7](#_Toc529625677)

[Technical Details 8](#_Toc529625678)

[Timeline 9](#_Toc529625679)

[Data Model 10](#_Toc529625680)

[Conceptual 10](#_Toc529625681)

[User Roles. 10](#_Toc529625682)

[Assumptions. 10](#_Toc529625683)

[Business Rules. 10](#_Toc529625684)

[Logical 11](#_Toc529625685)

[Implementation 13](#_Toc529625686)

[DDL 13](#_Toc529625687)

[DML 20](#_Toc529625688)

[Security 36](#_Toc529625689)

[Policies 36](#_Toc529625690)

[Database Administrator (DBA) Account Policy. 37](#_Toc529625691)

[Username Policy. 37](#_Toc529625692)

[Password Policy. 37](#_Toc529625693)

[Connection Policy. 37](#_Toc529625694)

[Role Assignment Policy. 37](#_Toc529625695)

[Account Modify Policy. 37](#_Toc529625696)

[Course Management Policy. 37](#_Toc529625697)

[Section Management Policy. 37](#_Toc529625698)

[Class Schedule Modify Policy. 38](#_Toc529625699)

[Admin Support Policy. 38](#_Toc529625700)

[Admin Account Restriction Policy. 38](#_Toc529625701)

[Instructor View Policy. 38](#_Toc529625702)

[Instructor Modify Policy. 38](#_Toc529625703)

[Student View Policy. 38](#_Toc529625704)

[Student Modify Policy. 38](#_Toc529625705)

[Class Tentative Schedule Policy. 39](#_Toc529625706)

[Procedures 39](#_Toc529625707)

[Procedure Preparation. 39](#_Toc529625708)

[Database Administrator (DBA) Account Procedure. 41](#_Toc529625709)

[Username Procedure. 41](#_Toc529625710)

[Password and Connection Procedure. 42](#_Toc529625711)

[Role Assignment Procedure. 43](#_Toc529625712)

[Account Modify Procedure. 44](#_Toc529625713)

[Course Management Procedure. 48](#_Toc529625714)

[Section Management Procedure. 51](#_Toc529625715)

[Class Schedule Modify Procedure. 54](#_Toc529625716)

[Admin Support Procedure. 56](#_Toc529625717)

[Admin Account Restriction Procedure. 60](#_Toc529625718)

[Instructor View Procedure. 64](#_Toc529625719)

[Instructor Modify Procedure. 70](#_Toc529625720)

[Student View Procedure. 76](#_Toc529625721)

[Student Modify Procedure. 80](#_Toc529625722)

[Class Tentative Schedule Policy. 84](#_Toc529625723)

[Conclusion 93](#_Toc529625724)

[Lesson Learned 93](#_Toc529625725)

[Final Thoughts 94](#_Toc529625726)

[References 95](#_Toc529625727)

Instructor Teaching Tracker: Security Project

There was a time when database security consisted of physical security and not much else. If we look back to the 1980s, systems were mostly isolated and offline negating the need for a substantial investment in access control or encryption methods. Fast forward 30 years, we’ve seen the maturity of access control, i.e. role-based and discretionary, and advanced methods, such as encryption, to protect critical data (Lesov, 2010). The purpose of this research document is to take a fully functional Oracle database, our teaching tracker for a military training detachment, and infuse security mechanisms to protect the stored data. We will first develop a statement of work and a project timeline. We will then review the conceptual, logical, and physical data model. There will be a review of the code in place for building the database. We will then create a security plan, security policies, and procedures for implementation. Utilizing a combination of Oracle security methods, we will secure the training facilities data for faculty and students.

# Statement of Work

## **Database Name**

Instructor Teaching Tracker

Database Description

This database supports the Instructor Teaching Tracker application for Training Detachment 6. It keeps track of instructor, course, admin, student, and section data; additionally, it tracks classes taught, the schedule, and students signed up for classes. The overall purpose is assisting instructors, administrators, and students with creating, scheduling, maintaining, and monitoring personnel, course, and class data in an efficient manner. The instructor, administrator, and student tables hold all pertinent personal information, such as name, address, and phone number. The course list table is a comprehensive listing of all courses taught across the different sections of the school, and it contains the name, administrator, description, hours, and section number for each course. The section table provides basic information on the sections which instructors and courses belong to, and it holds the following information: section number, name, address, and field of study. The class schedule table holds the different schedules classes can follow and gives the day and time a class is taught, a down day schedule, and notes. The Instructor Classes table is at the core of the database, providing a complete schedule of all courses taught by all instructors, with schedules, and also the room number of the class and any instructor notes. The Student Class Signup table provides students the ability to sign up for available classes in the Instructor Classes table.

## The Problem

Training Detachment 6 is a military field training facility, employing instructors and support admin staff hosting a variable number of students annually. The teaching tracker application and associated database became a necessity due to the complexities of tracking a large number of instructors, courses, and students. The database was created in-house and no security was implemented within the initial design. In an effort to maintain the confidentiality of certain data, such as instructor’s phone numbers and addresses, and to ensure the integrity of the data, i.e. prevent students from changing course data, we need to hire a security consultant to redesign the database with security permissions added.

# Security Plan

One of the most important steps in this project is to create a sound security plan. This document will be an overarching guide on set requirements to be enforced through security policies to establish sound security within the database (Theriault & Heney, 1998). For our database, the Instructor Teaching Tracker, we will achieve the necessary security by separating user accounts into specific groups, i.e. students, instructors, and support administrators. Policies will be created to bolster a comprehensive approach to a sound security implementation within the current database. Policies will be enforced with procedures created with Views, Virtual Private Database (VPD) procedures, and Oracle Label Security (OLS). All procedures will be thoroughly tested and documented before operationalized. Any user in violation of this plan or subsequent policies are subject to account disablement.

# Technical Details

The database was built with Oracle 12c on a Windows platform. Oracle SQL Developer Data Modeler was used to design the Entity Relationship Diagram (ERD). All Data Definition Language (DDL), Data Manipulation Language (DML), Data Control Language (DCL), and Transaction Control Language (TCL) implementation scripts were rendered in Oracle SQL Developer or SQL Plus. The primary security functions used in this project are Views, VPD, and OLS. Views are used to enhance security by filtering rows and/or columns based on role assignment (Theriault & Heney, 1998). VPDs can also control data at the row and column-level and do so by dynamically assigning a predicate to a specific type of query to restrict or allow access to certain data. This is accomplished by defining a policy and creating a function to prepare the predicate (7 Using Oracle Virtual Private Database to Control Data Access, 2018). OLS is based on VPD but reduces the amount of custom code needed for implementation. It provides row-level security by labeling rows with a classification data label which can be compared to a user authorization label (1 Introduction to Oracle Label Security, 2017).

# Timeline

Although Gantt charts are well over 100 years old, they still prove to be an effective tool today for project management. The Gantt chart provides a project timeline in a simple two-axis format: the vertical axis shows activity and the horizontal its place in time (Wilson, 2003). We can see in figure 1 below, our project should take 57 days to complete with 6 different sections: scope, analysis, design, development, testing, and documentation.

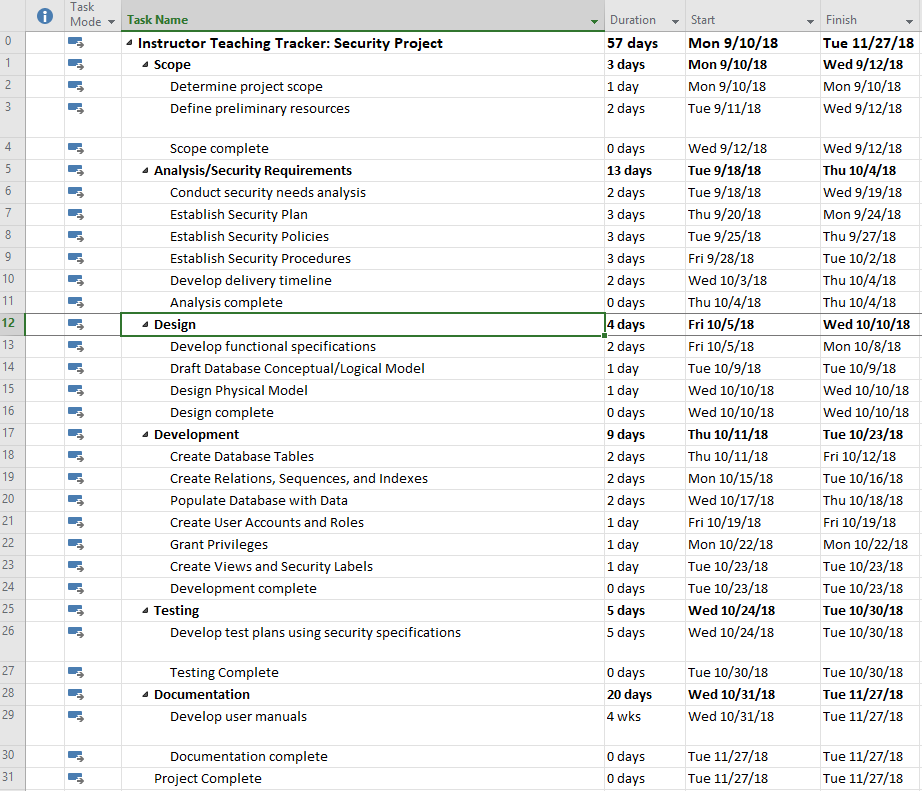


Figure 1. Project timeline created with Microsoft Project.

# Data Model

## Conceptual

User Roles. There will be three different primary user roles: student, instructor, and support admin. Specific policies will be defined in the security policy section for how each user role can and cannot interact with the database.

Assumptions. This application tracks classes taught by instructors and includes a class schedule. It is assumed classes will always meet a specific schedule criterion, i.e. Monday-Wednesday-Friday, Tuesday-Thursday, or Saturday. Down days are tracked for individual classes since this will vary based upon the specific class. For the usernames, we are using the user’s first character of their first name concatenated with the full last name with their sequence id added to the end. There is still a chance of duplicate usernames since instructor, student, and admin information is stored in different tables, but for the purpose of this project we are assuming this will not occur due to the low number of users.

Business Rules. The business rules to follow for this database are as follows:

1. A section may hire zero or more instructors, but an instructor must belong to only one section.
2. A section may own zero or more courses, but a course must be owned by only one section.
3. A section may hire zero or more administrators, but an administrator must belong to only one section.
4. An administrator can control multiple courses, but a course can only be controlled by one administrator.
5. A course may generate zero or more classes, but a class can only be generated from one course.
6. An instructor may teach zero or more classes, but a class can only be taught by one instructor.
7. A schedule may be generated for zero or more classes, but a class must only have one schedule.
8. A class may have zero or more students.
9. A student can sign up for zero or more classes, so long as there is no schedule conflict.

## Logical

Figure 2 provides a logical rendering of the database design, while figure 3 provides a more detailed ERD of the database design.

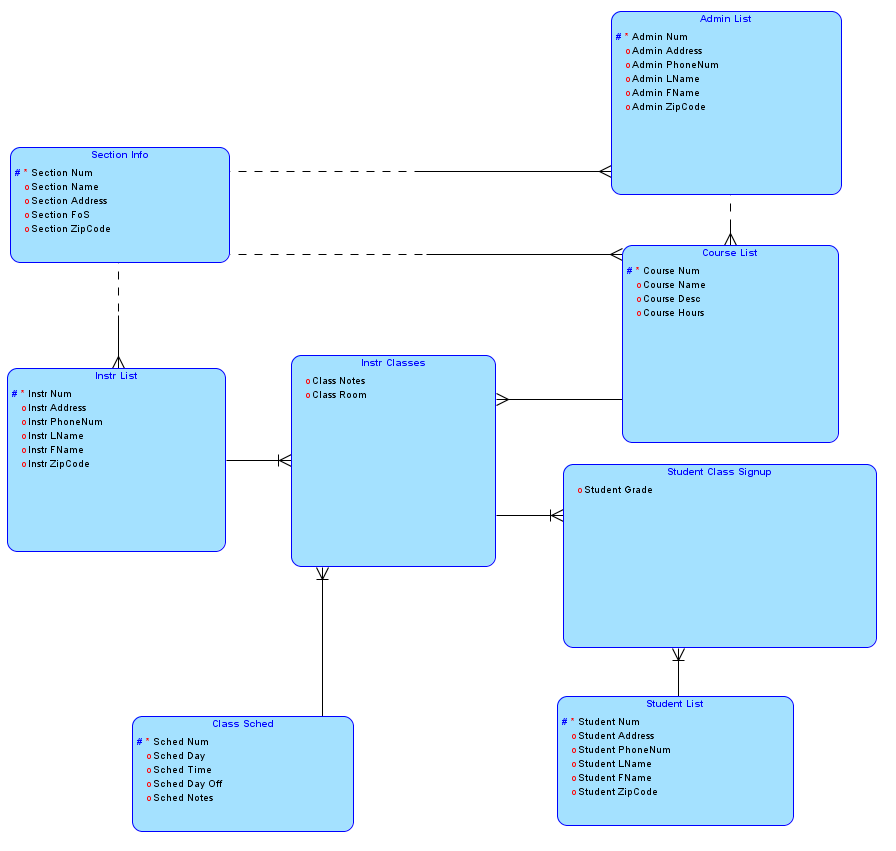


Figure 2. Logical model created with Oracle SQL Developer Data Modeler.

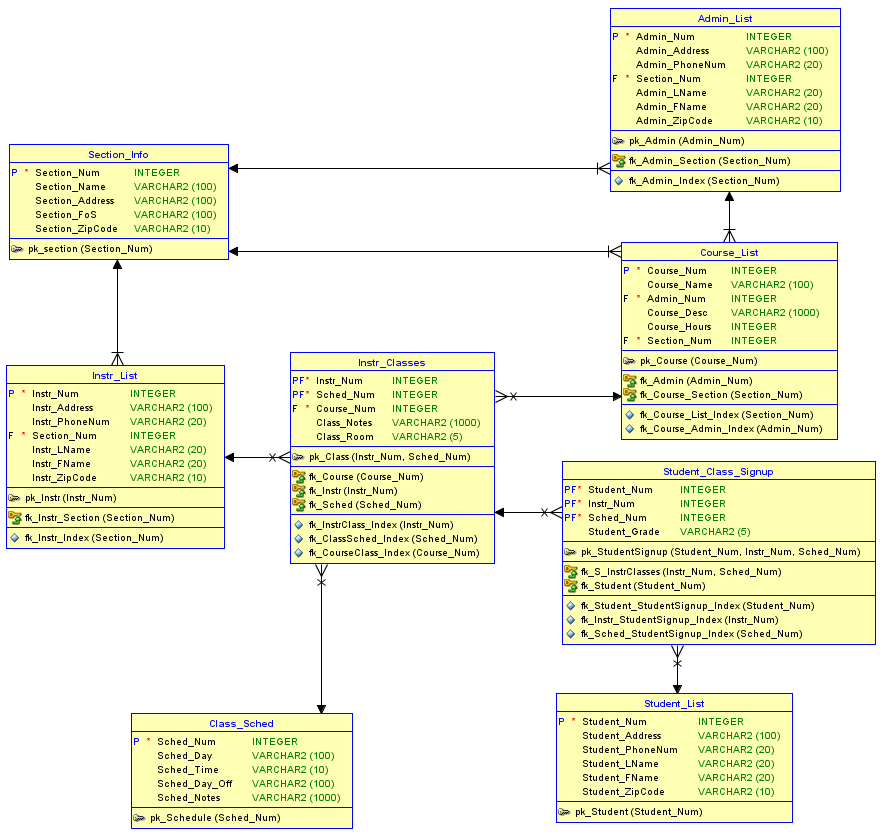


Figure 3. ERD created with Oracle SQL Developer Data Modeler.

# Implementation

This section consists of the physical implementation of the database, i.e. DDLs, used to create the database and the DMLs used to insert data into it.

## DDL

SQL> SET serveroutput on;

SQL> set sqlformat ansiconsole;

SQL>

SQL>

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

SQL> DROP TABLE Student\_Class\_Signup;

Table STUDENT\_CLASS\_SIGNUP dropped.

SQL> DROP TABLE Instr\_Classes;

Table INSTR\_CLASSES dropped.

SQL> DROP TABLE Student\_List;

Table STUDENT\_LIST dropped.

SQL> DROP TABLE Instr\_List;

Table INSTR\_LIST dropped.

SQL> DROP TABLE Course\_List;

Table COURSE\_LIST dropped.

SQL> DROP TABLE Admin\_List;

Table ADMIN\_LIST dropped.

SQL> DROP TABLE Section\_Info;

Table SECTION\_INFO dropped.

SQL> DROP TABLE Class\_Sched;

Table CLASS\_SCHED dropped.

SQL> DROP SEQUENCE SectionNum\_Seq;

Sequence SECTIONNUM\_SEQ dropped.

SQL> DROP SEQUENCE InstrNum\_Seq;

Sequence INSTRNUM\_SEQ dropped.

SQL> DROP SEQUENCE CourseNum\_Seq;

Sequence COURSENUM\_SEQ dropped.

SQL> DROP SEQUENCE AdminNum\_Seq;

Sequence ADMINNUM\_SEQ dropped.

SQL> DROP SEQUENCE SchedNum\_Seq;

Sequence SCHEDNUM\_SEQ dropped.

SQL> DROP SEQUENCE StudentNum\_Seq;

Sequence STUDENTNUM\_SEQ dropped.

SQL>

SQL> /\* Create tables \*/

SQL> CREATE TABLE Section\_Info

2 (

3 Section\_Num INTEGER NOT NULL,

4 Section\_Name VARCHAR (100),

5 Section\_Address VARCHAR (100),

6 Section\_FoS VARCHAR (100),

7 Section\_ZipCode VARCHAR(10),

8 CONSTRAINT pk\_section PRIMARY KEY (Section\_Num)

9 );

Table SECTION\_INFO created.

SQL>

SQL> DESCRIBE Section\_Info;

Name Null? Type

SECTION\_NUM NOT NULL NUMBER(38)

SECTION\_NAME VARCHAR2(100)

SECTION\_ADDRESS VARCHAR2(100)

SECTION\_FOS VARCHAR2(100)

SECTION\_ZIPCODE VARCHAR2(10)

SQL>

SQL> CREATE TABLE Admin\_List

2 (

3 Admin\_Num INTEGER NOT NULL,

4 Admin\_Address VARCHAR (100),

5 Admin\_PhoneNum VARCHAR (20),

6 Section\_Num INTEGER NOT NULL,

7 Admin\_LName VARCHAR (20),

8 Admin\_FName VARCHAR (20),

9 Admin\_ZipCode VARCHAR (10),

10 CONSTRAINT pk\_Admin PRIMARY KEY (Admin\_Num),

11 CONSTRAINT fk\_Admin\_Section FOREIGN KEY (Section\_Num)

12 REFERENCES Section\_Info

13 );

Table ADMIN\_LIST created.

SQL>

SQL> DESCRIBE Admin\_List;

Name Null? Type

ADMIN\_NUM NOT NULL NUMBER(38)

ADMIN\_ADDRESS VARCHAR2(100)

ADMIN\_PHONENUM VARCHAR2(20)

SECTION\_NUM NOT NULL NUMBER(38)

ADMIN\_LNAME VARCHAR2(20)

ADMIN\_FNAME VARCHAR2(20)

ADMIN\_ZIPCODE VARCHAR2(10)

SQL>

SQL>

SQL> CREATE TABLE Student\_List

2 (

3 Student\_Num INTEGER NOT NULL,

4 Student\_Address VARCHAR (100),

5 Student\_PhoneNum VARCHAR (20),

6 Student\_LName VARCHAR (20),

7 Student\_FName VARCHAR (20),

8 Student\_ZipCode VARCHAR (10),

9 CONSTRAINT pk\_Student PRIMARY KEY (Student\_Num)

10 );

Table STUDENT\_LIST created.

SQL>

SQL> DESCRIBE Student\_List;

Name Null? Type

STUDENT\_NUM NOT NULL NUMBER(38)

STUDENT\_ADDRESS VARCHAR2(100)

STUDENT\_PHONENUM VARCHAR2(20)

STUDENT\_LNAME VARCHAR2(20)

STUDENT\_FNAME VARCHAR2(20)

STUDENT\_ZIPCODE VARCHAR2(10)

SQL>

SQL>

SQL> CREATE TABLE Instr\_List

2 (

3 Instr\_Num INTEGER NOT NULL,

4 Instr\_Address VARCHAR (100),

5 Instr\_PhoneNum VARCHAR (20),

6 Section\_Num INTEGER NOT NULL,

7 Instr\_LName VARCHAR (20),

8 Instr\_FName VARCHAR (20),

9 Instr\_ZipCode VARCHAR (10),

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

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

12 REFERENCES Section\_Info

13 );

Table INSTR\_LIST created.

SQL>

SQL> DESCRIBE Instr\_List;

Name Null? Type

INSTR\_NUM NOT NULL NUMBER(38)

INSTR\_ADDRESS VARCHAR2(100)

INSTR\_PHONENUM VARCHAR2(20)

SECTION\_NUM NOT NULL NUMBER(38)

INSTR\_LNAME VARCHAR2(20)

INSTR\_FNAME VARCHAR2(20)

INSTR\_ZIPCODE VARCHAR2(10)

SQL>

SQL>

SQL> CREATE TABLE Course\_List

2 (

3 Course\_Num INTEGER NOT NULL,

4 Course\_Name VARCHAR (100),

5 Admin\_Num INTEGER NOT NULL,

6 Course\_Desc VARCHAR (1000),

7 Course\_Hours INTEGER,

8 Section\_Num INTEGER NOT NULL,

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

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

11 REFERENCES Section\_Info,

12 CONSTRAINT fk\_Admin FOREIGN KEY (Admin\_Num)

13 REFERENCES Admin\_List

14

15 );

Table COURSE\_LIST created.

SQL>

SQL> DESCRIBE Course\_List;

Name Null? Type

COURSE\_NUM NOT NULL NUMBER(38)

COURSE\_NAME VARCHAR2(100)

ADMIN\_NUM NOT NULL NUMBER(38)

COURSE\_DESC VARCHAR2(1000)

COURSE\_HOURS NUMBER(38)

SECTION\_NUM NOT NULL NUMBER(38)

SQL>

SQL>

SQL> CREATE TABLE Class\_Sched

2 (

3 Sched\_Num INTEGER NOT NULL,

4 Sched\_Day VARCHAR (100),

5 Sched\_Time VARCHAR (10),

6 Sched\_Day\_Off VARCHAR (100),

7 Sched\_Notes VARCHAR (1000),

8 CONSTRAINT pk\_Schedule PRIMARY KEY (Sched\_Num)

9 );

Table CLASS\_SCHED created.

SQL>

SQL> DESCRIBE Class\_Sched;

Name Null? Type

SCHED\_NUM NOT NULL NUMBER(38)

SCHED\_DAY VARCHAR2(100)

SCHED\_TIME VARCHAR2(10)

SCHED\_DAY\_OFF VARCHAR2(100)

SCHED\_NOTES VARCHAR2(1000)

SQL>

SQL> CREATE TABLE Instr\_Classes

2 (

3

4 Instr\_Num INTEGER NOT NULL,

5 Sched\_Num INTEGER NOT NULL,

6 Course\_Num INTEGER NOT NULL,

7 Class\_Notes VARCHAR (1000),

8 Class\_Room VARCHAR (5),

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

10 CONSTRAINT fk\_Instr FOREIGN KEY (Instr\_Num)

11 REFERENCES Instr\_List

12 ON DELETE CASCADE,

13 CONSTRAINT fk\_Sched FOREIGN KEY (Sched\_Num)

14 REFERENCES Class\_Sched

15 ON DELETE CASCADE,

16 CONSTRAINT fk\_Course FOREIGN KEY (Course\_Num)

17 REFERENCES Course\_List

18 ON DELETE CASCADE

19 );

Table INSTR\_CLASSES created.

SQL>

SQL> DESCRIBE Instr\_Classes;

Name Null? Type

INSTR\_NUM NOT NULL NUMBER(38)

SCHED\_NUM NOT NULL NUMBER(38)

COURSE\_NUM NOT NULL NUMBER(38)

CLASS\_NOTES VARCHAR2(1000)

CLASS\_ROOM VARCHAR2(5)

SQL>

SQL>

SQL> CREATE TABLE Student\_Class\_Signup

2 (

3 Student\_Num INTEGER NOT NULL,

4 Instr\_Num INTEGER NOT NULL,

5 Sched\_Num INTEGER NOT NULL,

6 Student\_Grade VARCHAR (5),

7 CONSTRAINT pk\_StudentSignup PRIMARY KEY (Student\_Num, Instr\_Num, Sched\_Num),

8 CONSTRAINT fk\_Student FOREIGN KEY (Student\_Num)

9 REFERENCES Student\_List

10 ON DELETE CASCADE,

11 CONSTRAINT fk\_S\_InstrClasses FOREIGN KEY (Instr\_Num, Sched\_Num)

12 REFERENCES Instr\_Classes(Instr\_Num, Sched\_Num)

13 ON DELETE CASCADE

14 );

Table STUDENT\_CLASS\_SIGNUP created.

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

SQL>

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

Index FK\_COURSE\_LIST\_INDEX created.

SQL> CREATE INDEX fk\_Course\_Admin\_Index on Course\_List(Admin\_Num);

Index FK\_COURSE\_ADMIN\_INDEX created.

SQL> CREATE INDEX fk\_Instr\_Index on Instr\_List(Section\_Num);

Index FK\_INSTR\_INDEX created.

SQL> CREATE INDEX fk\_Admin\_Index on Admin\_List(Section\_Num);

Index FK\_ADMIN\_INDEX created.

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

Index FK\_INSTRCLASS\_INDEX created.

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

Index FK\_CLASSSCHED\_INDEX created.

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

Index FK\_COURSECLASS\_INDEX created.

SQL> CREATE INDEX fk\_Student\_StudentSignup\_Index on Student\_Class\_Signup(Student\_Num);

Index FK\_STUDENT\_STUDENTSIGNUP\_INDEX created.

SQL> CREATE INDEX fk\_Instr\_StudentSignup\_Index on Student\_Class\_Signup(Instr\_Num);

Index FK\_INSTR\_STUDENTSIGNUP\_INDEX created.

SQL> CREATE INDEX fk\_Sched\_StudentSignup\_Index on Student\_Class\_Signup(Sched\_Num);

Index FK\_SCHED\_STUDENTSIGNUP\_INDEX created.

SQL>

SQL> /\* Create trigger \*/

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

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

2 FOR EACH ROW

3 BEGIN

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

5 END;

6 /

Trigger SCHEDCLASS\_TRIGGER compiled

SQL> /\* Create sequence\*/

SQL>

SQL> CREATE SEQUENCE SectionNum\_Seq

2 START WITH 1

3 INCREMENT BY 1;

Sequence SECTIONNUM\_SEQ created.

SQL>

SQL> CREATE SEQUENCE AdminNum\_Seq

2 START WITH 1

3 INCREMENT BY 1;

Sequence ADMINNUM\_SEQ created.

SQL>

SQL> CREATE SEQUENCE InstrNum\_Seq

2 START WITH 1

3 INCREMENT BY 1;

Sequence INSTRNUM\_SEQ created.

SQL>

SQL> CREATE SEQUENCE CourseNum\_Seq

2 START WITH 1

3 INCREMENT BY 1;

Sequence COURSENUM\_SEQ created.

SQL>

SQL> CREATE SEQUENCE SchedNum\_Seq

2 START WITH 1

3 INCREMENT BY 1;

Sequence SCHEDNUM\_SEQ created.

SQL>

SQL> CREATE SEQUENCE StudentNum\_Seq

2 START WITH 1

3 INCREMENT BY 1;

Sequence STUDENTNUM\_SEQ created.

## DML

SQL>

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

SQL>

SQL>

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Computer Science Section', '110 Finegand Place','Computer Science','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS,Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Art Section', '111 Finegand Place','Art','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS,Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Aircraft Section', '112 Finegand Place','Aircraft','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS,Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Robotic Section', '113 Finegand Place','Robotics','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Database Section', '114 Finegand Place','Database Technology','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'English Section', '115 Finegand Place','English language','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Physics Section', '116 Finegand Place','Physics','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS,Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Chemistry Section', '117 Finegand Place','Chemistry','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Spanish Section', '118 Finegand Place','Spanish language','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Automotive Section', '119 Finegand Place','Automotive Mechanics','31088');

1 row inserted.

SQL> INSERT INTO Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (SectionNum\_Seq.NEXTVAL, 'Space Travel Section', '120 Finegand Place','Space Travel','31088');

1 row inserted.

SQL>

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'501 Orange Park','111-333-3434', '1', 'Evans', 'Bob', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'502 Orange Park','111-333-3435', '2', 'Johnson', 'Mike', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'503 Orange Park','111-333-3436', '3', 'Jenkins', 'Mary', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'504 Orange Park','111-333-3437', '4', 'Russell', 'Jim', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'505 Orange Park','111-333-3438', '5', 'Bargueno', 'Patricia', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'506 Orange Park','111-333-3439', '6', 'Hopkins', 'John', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'507 Orange Park','111-333-3440', '7', 'Smith', 'Will', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'508 Orange Park','111-333-3441', '8', 'Brown', 'Adam', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'509 Orange Park','111-333-3442', '9', 'Williams', 'Shane', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'510 Orange Park','111-333-3443', '10', 'Robins', 'Amber', '31004');

1 row inserted.

SQL> INSERT INTO Admin\_List (Admin\_Num, Admin\_Address, Admin\_PhoneNum, Section\_Num, Admin\_LName, Admin\_FName, Admin\_ZipCode)

2 VALUES (AdminNum\_Seq.NEXTVAL,'511 Orange Park','111-333-3444', '4', 'Hawkins', 'Tim', '31004');

1 row inserted.

SQL>

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 1','222-333-3434', 'Gilbert', 'Sarah', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 2','222-333-3435', 'Myers', 'James', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 3','222-333-3436', 'Howard', 'Rico', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 4','222-333-3437', 'Bush', 'Donald', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 5','222-333-3438', 'Simmons', 'Bill', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 6','222-333-3439', 'Aultman', 'Richard', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 7','222-333-3440', 'Ruger', 'Victoria', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 8','222-333-3441', 'Thomas', 'Melissa', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 9','222-333-3442', 'Synder', 'Jennifer', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 10','222-333-3443', 'Baker', 'Elizabeth', '31005');

1 row inserted.

SQL> INSERT INTO Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (StudentNum\_Seq.NEXTVAL,'Dorm 11','222-333-3444', 'Pines', 'Matt', '31005');

1 row inserted.

SQL>

SQL>

SQL>

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'101 Apple Street','111-222-3434', '1', 'Roberts', 'John', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'102 Apple Street','111-222-3435', '2', 'Russell', 'Brandon', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'103 Apple Street','111-222-3436', '3', 'Lopez', 'Mike', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'104 Apple Street','111-222-3437', '4', 'Monteor', 'Jim', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'105 Apple Street','111-222-3438', '5', 'Hamm', 'Patricia', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'106 Apple Street','111-222-3439', '6', 'Ingle', 'Tammy', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'107 Apple Street','111-222-3440', '7', 'Jordan', 'Keith', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'108 Apple Street','111-222-3441', '8', 'Cooper', 'Adam', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'109 Apple Street','111-222-3442', '9', 'McCoy', 'Amber', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'110 Apple Street','111-222-3443', '10', 'Fuller', 'Jana', '31003');

1 row inserted.

SQL> INSERT INTO Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (InstrNum\_Seq.NEXTVAL,'111 Apple Street','111-222-3444', '10', 'Xavier', 'Mark', '31003');

1 row inserted.

SQL>

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Computer Science',1, 'Intro to Computer Science','3', '1');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Art',2, 'Intro to Art','3', '2');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Aircraft mechanics',3, 'Intro to Aircraft mechanics','3', '3');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Robotics',4, 'Intro to Robotics','3', '4');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Databases',5, 'Intro to Databases','3', '5');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Advanced English',6, 'Advanced English','3', '6');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Physics',7, 'Intro to Physics','3', '7');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Chemistry',8, 'Intro to Chemistry','3', '8');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Intro to Spanish',9, 'Intro to Spanish','3', '9');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Automotive mechanics Intro',10, 'Automotive mechanics Intro','3', '10');

1 row inserted.

SQL> INSERT INTO Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (CourseNum\_Seq.NEXTVAL, 'Advanced Robotics',11, 'Advanced Robotics','6', '4');

1 row inserted.

SQL>

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Mon-Tues-Wed','0900-1100','Oct. 13th', 'This is one of the main schedules');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Mon-Tues-Thur','0900-1100','Oct. 13th', 'This is one of the main schedules');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Mon-Wed','1000-1200','Oct. 14th', 'This is one of the main schedules');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Mon-Wed-Fri','1000-1200','Oct. 15th', 'This is an alternative schedule');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Mon-Tues','0800-0900','Oct. 16th', 'This is an alternative schedule');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Tues-Wed','1300-1500','Oct. 17th', 'This is an alternative schedule');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Thur-Fri','1300-1500','Oct. 18th', 'This is an alternative schedule');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Wed-Fri','1600-1800','Oct. 19th', 'This is an alternative schedule');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Sat', '0900-1100','Oct. 20th', 'Weekend Schedule');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Sun', '0900-1100','Oct. 21st', 'Weekend Schedule');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Mon', '0900-0930','6 days a week', 'Beta');

1 row inserted.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (SchedNum\_Seq.NEXTVAL,'Tues', '1100-1115','6 days a week', 'Beta');

1 row inserted.

SQL>

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

2 VALUES ('1', '1', '1', 'Computer one broke','A');

1 row inserted.

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

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

1 row inserted.

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

2 VALUES ('3', '3', '3', 'Instructor Chair broke','C');

1 row inserted.

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

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

1 row inserted.

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

2 VALUES ('5', '5', '5', 'Computer two broke','E');

1 row inserted.

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

2 VALUES ('6', '6', '6', 'No whiteboard','F');

1 row inserted.

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

2 VALUES ('7', '7', '7', 'station 3 missing keyboard','G');

1 row inserted.

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

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

1 row inserted.

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

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

1 row inserted.

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

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

1 row inserted.

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

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

1 row inserted.

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

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

1 row inserted.

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

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

1 row inserted.

SQL>

SQL>

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (1, 1, 1, 'A');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (1, 2, 2, 'B');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (2, 3, 3, 'A');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (3, 3, 3, 'C');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num,Student\_Grade)

2 VALUES (4, 4, 4, 'C');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (5, 5, 5, 'B');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (6, 6, 6, 'D');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (7, 7, 7, 'F');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (8, 8, 8, '');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (9, 9, 9, 'A');

1 row inserted.

SQL> INSERT INTO Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (10, 10, 10, 'A');

1 row inserted.

SQL>

SQL> commit;

Commit complete.

SQL>

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

SQL>

SQL> SELECT \* FROM Section\_Info;

SECTION\_NUM SECTION\_NAME SECTION\_ADDRESS SECTION\_FOS SECTION\_ZIPCODE

1 Computer Science Section 110 Finegand Place Computer Science 31088

2 Art Section 111 Finegand Place Art 31088

3 Aircraft Section 112 Finegand Place Aircraft 31088

4 Robotic Section 113 Finegand Place Robotics 31088

5 Database Section 114 Finegand Place Database Technology 31088

6 English Section 115 Finegand Place English language 31088

7 Physics Section 116 Finegand Place Physics 31088

8 Chemistry Section 117 Finegand Place Chemistry 31088

9 Spanish Section 118 Finegand Place Spanish language 31088

10 Automotive Section 119 Finegand Place Automotive Mechanics 31088

11 Space Travel Section 120 Finegand Place Space Travel 31088

11 rows selected.

SQL> SELECT \* FROM Instr\_List;

INSTR\_NUM INSTR\_ADDRESS INSTR\_PHONENUM SECTION\_NUM INSTR\_LNAME INSTR\_FNAME INSTR\_ZIPCODE

1 101 Apple Street 111-222-3434 1 Roberts John 31003

2 102 Apple Street 111-222-3435 2 Russell Brandon 31003

3 103 Apple Street 111-222-3436 3 Lopez Mike 31003

4 104 Apple Street 111-222-3437 4 Monteor Jim 31003

5 105 Apple Street 111-222-3438 5 Hamm Patricia 31003

6 106 Apple Street 111-222-3439 6 Ingle Tammy 31003

7 107 Apple Street 111-222-3440 7 Jordan Keith 31003

8 108 Apple Street 111-222-3441 8 Cooper Adam 31003

9 109 Apple Street 111-222-3442 9 McCoy Amber 31003

10 110 Apple Street 111-222-3443 10 Fuller Jana 31003

11 111 Apple Street 111-222-3444 10 Xavier Mark 31003

11 rows selected.

SQL> SELECT \* FROM Admin\_List;

ADMIN\_NUM ADMIN\_ADDRESS ADMIN\_PHONENUM SECTION\_NUM ADMIN\_LNAME ADMIN\_FNAME ADMIN\_ZIPCODE

1 501 Orange Park 111-333-3434 1 Evans Bob 31004

2 502 Orange Park 111-333-3435 2 Johnson Mike 31004

3 503 Orange Park 111-333-3436 3 Jenkins Mary 31004

4 504 Orange Park 111-333-3437 4 Russell Jim 31004

5 505 Orange Park 111-333-3438 5 Bargueno Patricia 31004

6 506 Orange Park 111-333-3439 6 Hopkins John 31004

7 507 Orange Park 111-333-3440 7 Smith Will 31004

8 508 Orange Park 111-333-3441 8 Brown Adam 31004

9 509 Orange Park 111-333-3442 9 Williams Shane 31004

10 510 Orange Park 111-333-3443 10 Robins Amber 31004

11 511 Orange Park 111-333-3444 4 Hawkins Tim 31004

11 rows selected.

SQL> SELECT \* FROM Student\_List;

STUDENT\_NUM STUDENT\_ADDRESS STUDENT\_PHONENUM STUDENT\_LNAME STUDENT\_FNAME STUDENT\_ZIPCODE

1 Dorm 1 222-333-3434 Gilbert Sarah 31005

2 Dorm 2 222-333-3435 Myers James 31005

3 Dorm 3 222-333-3436 Howard Rico 31005

4 Dorm 4 222-333-3437 Bush Donald 31005

5 Dorm 5 222-333-3438 Simmons Bill 31005

6 Dorm 6 222-333-3439 Aultman Richard 31005

7 Dorm 7 222-333-3440 Ruger Victoria 31005

8 Dorm 8 222-333-3441 Thomas Melissa 31005

9 Dorm 9 222-333-3442 Synder Jennifer 31005

10 Dorm 10 222-333-3443 Baker Elizabeth 31005

11 Dorm 11 222-333-3444 Pines Matt 31005

11 rows selected.

SQL> SELECT \* FROM Course\_List;

COURSE\_NUM COURSE\_NAME ADMIN\_NUM COURSE\_DESC COURSE\_HOURS SECTION\_NUM

1 Intro to Computer Science 1 Intro to Computer Science 3 1

2 Intro to Art 2 Intro to Art 3 2

3 Intro to Aircraft mechanics 3 Intro to Aircraft mechanics 3 3

4 Intro to Robotics 4 Intro to Robotics 3 4

5 Intro to Databases 5 Intro to Databases 3 5

6 Advanced English 6 Advanced English 3 6

7 Intro to Physics 7 Intro to Physics 3 7

8 Intro to Chemistry 8 Intro to Chemistry 3 8

9 Intro to Spanish 9 Intro to Spanish 3 9

10 Automotive mechanics Intro 10 Automotive mechanics Intro 3 10

11 Advanced Robotics 11 Advanced Robotics 6 4

11 rows selected.

SQL> SELECT \* FROM Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule

9 Sat 0900-1100 Oct. 20th Weekend Schedule

10 Sun 0900-1100 Oct. 21st Weekend Schedule

11 Mon 0900-0930 6 days a week Beta

12 Tues 1100-1115 6 days a week Beta

12 rows selected.

SQL>

SQL> SELECT \* FROM Instr\_Classes;

INSTR\_NUM SCHED\_NUM COURSE\_NUM CLASS\_NOTES CLASS\_ROOM

1 1 1 Computer one broke A

2 2 2 Room Ready B

3 3 3 Instructor Chair broke C

4 4 4 A/C not working D

5 5 5 Computer two broke E

6 6 6 No whiteboard F

7 7 7 station 3 missing keyboard G

8 8 8 Room Ready H

9 9 9 Room Ready I

10 10 10 Room Ready J

10 9 10 Room Ready J

9 8 10 Room Ready X

1 2 9 Room Ready Z

13 rows selected.

SQL> SELECT \* FROM Student\_Class\_Signup;

STUDENT\_NUM INSTR\_NUM SCHED\_NUM STUDENT\_GRADE

1 1 1 A

1 2 2 B

2 3 3 A

3 3 3 C

4 4 4 C

5 5 5 B

6 6 6 D

7 7 7 F

8 8 8

9 9 9 A

10 10 10 A

11 rows selected.

Connection created by CONNECT script command disconnected

# Security

## Policies

Security policies are a subset of the security plan, defined in a previous section, and provide a more detailed specification of what requirements will be enforced to promote database security hardening (Theriault & Heney, 1998). The following policies will be implemented in the Instructor Teaching Tracker database:

Database Administrator (DBA) Account Policy. As recommended by Oracle, we will create a separate Instructor DBA account, vice using the “SYS” or “SYSTEM” accounts, to execute all DDL, DML, DCL, and TCL for this database (7.3.1 SYS and SYSTEM Users, 2017). The only exceptions will be the use of the LBACSYS account for implementing OLS and individual admin, instructor, and student accounts for testing.

Username Policy. Effective immediately, all database users will be assigned a username. The username will consist of the first letter of the first name, followed by the full last name with no space, followed by the row identification number with no space.

Password Policy. Per the recommendation of (Grassi, et al., 2017), user passwords will be at least 8 characters in length, using a combination of available Unicode characters, and should avoid common words or repetitive characters.

Connection Policy. All users are allowed to create a session to the database.

Role Assignment Policy. The admin\_role will be assigned to all administrative support staff. The instructor\_role will be assigned to all instructors. The student\_role will be assigned to all students.

Account Modify Policy. Administrators are the only authorized users to add or remove users from the instructor and student tables.

Course Management Policy. Administrators are in charge of adding, removing, and updating course table data.

Section Management Policy. Administrators are in charge of adding, removing, and updating section table data.

Class Schedule Modify Policy. Administrators are the only authorized user to add, remove, and update the class schedule table data. However, instructors can update the notes section.

Admin Support Policy. Administrators are authorized to insert, update, and delete all database tables (except the admin personnel table), in the event other users need assistance. For example, instructors rely on administrators to add and remove classes from their schedule, and students would need an administrator to disenroll from a class.

Admin Account Restriction Policy. Administrators can view the admin table and update their personal information. The system administrators are the only ones able to add or remove from the admin table.

Instructor View Policy. Instructors are authorized to view only their personal info in the instructor list table. They may also view their section info, their class schedule, course information on all courses, and a complete list of available schedules to assign their classes to. They may also view basic information on students enrolled in their classes.

Instructor Modify Policy. Instructors are only authorized to update their personal information, their class notes, student grades of students enrolled in their classes, and may also remove students from their classes.

Student View Policy. Students are only authorized to view their personal information, the instructor class schedule, classes they are enrolled in, and their grade for classes they are enrolled in.

Student Modify Policy. Students are only authorized to update their personal information, and they may also enroll themselves in classes.

Class Tentative Schedule Policy. Only the Instructor DBA and school administrators may view and/or create tentative class schedules, i.e. alpha and beta test schedules. Tentative schedules will be regarded as highly sensitive data, while approved schedules will be regarded as sensitive data. Instructors may view, but not create tentative class schedules. Students may view approved class schedules.

## Procedures

Procedure Preparation. The following shows the preparation SQL code needed before procedure implementation:

SQL> SET SERVEROUTPUT ON;

SQL> set sqlformat ansiconsole;

SQL>

SQL> /\*Connect to Instructor DBA\*/

SQL> connect InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

SQL> /\* Drop users, roles, policies, and other objects you create\*/

SQL> DROP ROLE student\_role;

Role STUDENT\_ROLE dropped.

SQL> DROP ROLE instructor\_role;

Role INSTRUCTOR\_ROLE dropped.

SQL> DROP ROLE admin\_role;

Role ADMIN\_ROLE dropped.

SQL>

SQL> BEGIN

2 DBMS\_RLS.DROP\_POLICY(

3 object\_schema => 'InstructorDBA',

4 object\_name => 'Admin\_List',

5 policy\_name => 'Hide\_Admin\_Info'

6 );

7 END;

8 /

PL/SQL procedure successfully completed.

SQL>

SQL> CONNECT lbacsys/brr1wik7

Connected.

SQL> show user;

USER is "LBACSYS"

SQL>

SQL> BEGIN

2 SA\_SYSDBA.DROP\_POLICY(

3 policy\_name => 'Sched\_OLS\_POL'

4 );

5 END;

6 /

PL/SQL procedure successfully completed.

SQL> connect InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

SQL> /\*Instructor account drop\*/

SQL> BEGIN

2 FOR x IN (SELECT User\_Name FROM Instr\_List)

3 LOOP

4 EXECUTE IMMEDIATE 'DROP USER '||x.User\_Name;

5 END LOOP;

6 /\*Admin account drop\*/

7 FOR x IN (SELECT User\_Name FROM Admin\_List)

8 LOOP

9 EXECUTE IMMEDIATE 'DROP USER '||x.User\_Name;

10 END LOOP;

11 /\*Student account drop\*/

12 FOR x IN (SELECT User\_Name FROM Student\_List)

13 LOOP

14 EXECUTE IMMEDIATE 'DROP USER '||x.User\_Name;

15 END LOOP;

16 END;

17 /

PL/SQL procedure successfully completed.

SQL>

SQL> ALTER TABLE Admin\_List DROP COLUMN User\_Name;

Table ADMIN\_LIST altered.

SQL> ALTER TABLE Instr\_List DROP COLUMN User\_Name;

Table INSTR\_LIST altered.

SQL> ALTER TABLE Student\_List DROP COLUMN User\_Name;

Table STUDENT\_LIST altered.

SQL>

SQL>

Database Administrator (DBA) Account Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Database Administrator Account Procedure\*/

SQL> connect sys/brr1wik7 as sysdba;

Connected.

SQL> show user;

USER is "SYS"

SQL> DROP USER InstructorDBA cascade;

User INSTRUCTORDBA dropped.

SQL> Create user InstructorDBA identified by brr1wik7;

User INSTRUCTORDBA created.

SQL> grant create session to InstructorDBA;

Grant succeeded.

SQL> grant dba to InstructorDBA;

Grant succeeded.

SQL> grant execute on dbms\_rls to InstructorDBA;

Grant succeeded.

SQL> connect InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

Username Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Username Procedure - Create usernames for every user\*/

SQL> ALTER TABLE Admin\_List ADD User\_Name VARCHAR2(45);

Table ADMIN\_LIST altered.

SQL> ALTER TABLE Instr\_List ADD User\_Name VARCHAR2(45);

Table INSTR\_LIST altered.

SQL> ALTER TABLE Student\_List ADD User\_Name VARCHAR2(45);

Table STUDENT\_LIST altered.

SQL>

SQL> UPDATE Admin\_List a SET User\_Name =

2 (SELECT CONCAT(UPPER(CONCAT(SUBSTR(Admin\_FName,1,1),Admin\_LName)), Admin\_Num)

3 FROM Admin\_List b WHERE a.Admin\_Num = b.Admin\_Num);

11 rows updated.

SQL>

SQL> UPDATE Instr\_List a SET User\_Name =

2 (SELECT CONCAT(UPPER(CONCAT(SUBSTR(Instr\_FName,1,1),Instr\_LName)), Instr\_Num)

3 FROM Instr\_List b WHERE a.Instr\_Num = b.Instr\_Num);

11 rows updated.

SQL>

SQL> UPDATE Student\_List a SET User\_Name =

2 (SELECT CONCAT(UPPER(CONCAT(SUBSTR(Student\_FName,1,1),Student\_LName)), Student\_Num)

3 FROM Student\_List b WHERE a.Student\_Num = b.Student\_Num);

11 rows updated.

SQL>

SQL>

Password and Connection Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Password Procedure - Create Instructor user accounts\*/

SQL> DECLARE

2 s\_num INTEGER;

3 begin

4 s\_num :=0;

5 for x in (SELECT User\_Name FROM Instr\_List)

6 loop

7 execute immediate 'CREATE USER '||x.User\_Name||' IDENTIFIED BY TheSecPass'||s\_num;

8 s\_num := s\_num +1;

9 end loop;

10 /\*Create Admin user accounts\*/

11 s\_num :=0;

12 for x in (SELECT User\_Name FROM Admin\_List)

13 loop

14 execute immediate 'CREATE USER '||x.User\_Name||' IDENTIFIED BY TheSecPass'||s\_num;

15 s\_num := s\_num +1;

16 end loop;

17 /\*Create Student user accounts\*/

18 s\_num :=0;

19 for x in (SELECT User\_Name FROM Student\_List)

20 loop

21 execute immediate 'CREATE USER '||x.User\_Name||' IDENTIFIED BY TheSecPass'||s\_num;

22 s\_num := s\_num +1;

23 end loop;

24 /\*Connection Prodcedure - grant create session\*/

25 /\*Instructor account\*/

26 for x in (SELECT User\_Name FROM Instr\_List)

27 loop

28 execute immediate 'GRANT CREATE SESSION TO '||x.User\_Name;

29 end loop;

30 /\*Admin account\*/

31 for x in (SELECT User\_Name FROM Admin\_List)

32 loop

33 execute immediate 'GRANT CREATE SESSION TO '||x.User\_Name;

34 end loop;

35 /\*Student account\*/

36 for x in (SELECT User\_Name FROM Student\_List)

37 loop

38 execute immediate 'GRANT CREATE SESSION TO '||x.User\_Name;

39 end loop;

40 end;

41 /

PL/SQL procedure successfully completed.

Role Assignment Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Role Assignment Procedure\*/

SQL> /\*Create roles\*/

SQL> CREATE ROLE admin\_role;

Role ADMIN\_ROLE created.

SQL> CREATE ROLE instructor\_role;

Role INSTRUCTOR\_ROLE created.

SQL> CREATE ROLE student\_role;

Role STUDENT\_ROLE created.

SQL> /\*Assign roles\*/

SQL> begin

2 for x in (SELECT User\_Name FROM Admin\_List)

3 loop

4 execute immediate 'GRANT admin\_role TO '||x.User\_Name;

5 end loop;

6 for x in (SELECT User\_Name FROM Instr\_List)

7 loop

8 execute immediate 'GRANT instructor\_role TO '||x.User\_Name;

9 end loop;

10 for x in (SELECT User\_Name FROM Student\_List)

11 loop

12 execute immediate 'GRANT student\_role TO '||x.User\_Name;

13 end loop;

14 end;

15 /

PL/SQL procedure successfully completed.

Account Modify Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Account Modify Procedure - Admins have full rights to the instructor and student tables.\*/

SQL> GRANT SELECT, INSERT, DELETE, UPDATE ON Instr\_List TO admin\_role;

Grant succeeded.

SQL> GRANT SELECT, INSERT, DELETE, UPDATE ON Student\_List TO admin\_role;

Grant succeeded.

SQL>

SQL> /\*Test Account Modify Procedure\*/

SQL> /\*Test admin access\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SAVEPOINT Admin\_Account\_Modify\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Instr\_List;

INSTR\_NUM INSTR\_ADDRESS INSTR\_PHONENUM SECTION\_NUM INSTR\_LNAME INSTR\_FNAME INSTR\_ZIPCODE USER\_NAME

1 101 Apple Street 111-222-3434 1 Roberts John 31003 JROBERTS1

2 102 Apple Street 111-222-3435 2 Russell Brandon 31003 BRUSSELL2

3 103 Apple Street 111-222-3436 3 Lopez Mike 31003 MLOPEZ3

4 104 Apple Street 111-222-3437 4 Monteor Jim 31003 JMONTEOR4

5 105 Apple Street 111-222-3438 5 Hamm Patricia 31003 PHAMM5

6 106 Apple Street 111-222-3439 6 Ingle Tammy 31003 TINGLE6

7 107 Apple Street 111-222-3440 7 Jordan Keith 31003 KJORDAN7

8 108 Apple Street 111-222-3441 8 Cooper Adam 31003 ACOOPER8

9 109 Apple Street 111-222-3442 9 McCoy Amber 31003 AMCCOY9

10 110 Apple Street 111-222-3443 10 Fuller Jana 31003 JFULLER10

11 111 Apple Street 111-222-3444 10 Xavier Mark 31003 MXAVIER11

11 rows selected.

SQL> SELECT \* FROM InstructorDBA.Student\_List;

STUDENT\_NUM STUDENT\_ADDRESS STUDENT\_PHONENUM STUDENT\_LNAME STUDENT\_FNAME STUDENT\_ZIPCODE USER\_NAME

1 Dorm 1 222-333-3434 Gilbert Sarah 31005 SGILBERT1

2 Dorm 2 222-333-3435 Myers James 31005 JMYERS2

3 Dorm 3 222-333-3436 Howard Rico 31005 RHOWARD3

4 Dorm 4 222-333-3437 Bush Donald 31005 DBUSH4

5 Dorm 5 222-333-3438 Simmons Bill 31005 BSIMMONS5

6 Dorm 6 222-333-3439 Aultman Richard 31005 RAULTMAN6

7 Dorm 7 222-333-3440 Ruger Victoria 31005 VRUGER7

8 Dorm 8 222-333-3441 Thomas Melissa 31005 MTHOMAS8

9 Dorm 9 222-333-3442 Synder Jennifer 31005 JSYNDER9

10 Dorm 10 222-333-3443 Baker Elizabeth 31005 EBAKER10

11 Dorm 11 222-333-3444 Pines Matt 31005 MPINES11

11 rows selected.

SQL> UPDATE InstructorDBA.Instr\_List SET Instr\_LName = 'test';

11 rows updated.

SQL> UPDATE InstructorDBA.Student\_List SET Student\_LName = 'test';

11 rows updated.

SQL> INSERT INTO InstructorDBA.Student\_List (Student\_Num, Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_ZipCode)

2 VALUES (999,'Dorm 11','222-333-3444', 'Pines', 'Matt', '31005');

1 row inserted.

SQL> INSERT INTO InstructorDBA.Instr\_List (Instr\_Num, Instr\_Address, Instr\_PhoneNum, Section\_Num, Instr\_LName, Instr\_FName, Instr\_ZipCode)

2 VALUES (888,'101 Apple Street','111-222-3434', '1', 'Roberts', 'John', '31003');

1 row inserted.

SQL> SELECT \* FROM InstructorDBA.Instr\_List;

INSTR\_NUM INSTR\_ADDRESS INSTR\_PHONENUM SECTION\_NUM INSTR\_LNAME INSTR\_FNAME INSTR\_ZIPCODE USER\_NAME

1 101 Apple Street 111-222-3434 1 test John 31003 JROBERTS1

2 102 Apple Street 111-222-3435 2 test Brandon 31003 BRUSSELL2

3 103 Apple Street 111-222-3436 3 test Mike 31003 MLOPEZ3

4 104 Apple Street 111-222-3437 4 test Jim 31003 JMONTEOR4

5 105 Apple Street 111-222-3438 5 test Patricia 31003 PHAMM5

6 106 Apple Street 111-222-3439 6 test Tammy 31003 TINGLE6

7 107 Apple Street 111-222-3440 7 test Keith 31003 KJORDAN7

8 108 Apple Street 111-222-3441 8 test Adam 31003 ACOOPER8

9 109 Apple Street 111-222-3442 9 test Amber 31003 AMCCOY9

10 110 Apple Street 111-222-3443 10 test Jana 31003 JFULLER10

11 111 Apple Street 111-222-3444 10 test Mark 31003 MXAVIER11

888 101 Apple Street 111-222-3434 1 Roberts John 31003

12 rows selected.

SQL> SELECT \* FROM InstructorDBA.Student\_List;

STUDENT\_NUM STUDENT\_ADDRESS STUDENT\_PHONENUM STUDENT\_LNAME STUDENT\_FNAME STUDENT\_ZIPCODE USER\_NAME

1 Dorm 1 222-333-3434 test Sarah 31005 SGILBERT1

2 Dorm 2 222-333-3435 test James 31005 JMYERS2

3 Dorm 3 222-333-3436 test Rico 31005 RHOWARD3

4 Dorm 4 222-333-3437 test Donald 31005 DBUSH4

5 Dorm 5 222-333-3438 test Bill 31005 BSIMMONS5

6 Dorm 6 222-333-3439 test Richard 31005 RAULTMAN6

7 Dorm 7 222-333-3440 test Victoria 31005 VRUGER7

8 Dorm 8 222-333-3441 test Melissa 31005 MTHOMAS8

9 Dorm 9 222-333-3442 test Jennifer 31005 JSYNDER9

10 Dorm 10 222-333-3443 test Elizabeth 31005 EBAKER10

11 Dorm 11 222-333-3444 test Matt 31005 MPINES11

999 Dorm 11 222-333-3444 Pines Matt 31005

12 rows selected.

SQL> DELETE FROM InstructorDBA.Instr\_List;

12 rows deleted.

SQL> DELETE FROM InstructorDBA.Student\_List;

12 rows deleted.

SQL> ROLLBACK TO Admin\_Account\_Modify\_Procedure;

Rollback complete.

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Instr\_List;

Error starting at line : 165 in command -

SELECT \* FROM InstructorDBA.Instr\_List

Error at Command Line : 165 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Student\_List;

Error starting at line : 166 in command -

SELECT \* FROM InstructorDBA.Student\_List

Error at Command Line : 166 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Instr\_List;

Error starting at line : 170 in command -

SELECT \* FROM InstructorDBA.Instr\_List

Error at Command Line : 170 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Student\_List;

Error starting at line : 171 in command -

SELECT \* FROM InstructorDBA.Student\_List

Error at Command Line : 171 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

Course Management Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Course Management Procedure - Admin have full rights to course information\*/

SQL> GRANT SELECT, INSERT, DELETE, UPDATE ON Course\_List TO admin\_role;

Grant succeeded.

SQL>

SQL> /\*Test Course Management Procedure\*/

SQL> /\*Test admin access\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SAVEPOINT Course\_Account\_Modify\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Course\_List;

COURSE\_NUM COURSE\_NAME ADMIN\_NUM COURSE\_DESC COURSE\_HOURS SECTION\_NUM

1 Intro to Computer Science 1 Intro to Computer Science 3 1

2 Intro to Art 2 Intro to Art 3 2

3 Intro to Aircraft mechanics 3 Intro to Aircraft mechanics 3 3

4 Intro to Robotics 4 Intro to Robotics 3 4

5 Intro to Databases 5 Intro to Databases 3 5

6 Advanced English 6 Advanced English 3 6

7 Intro to Physics 7 Intro to Physics 3 7

8 Intro to Chemistry 8 Intro to Chemistry 3 8

9 Intro to Spanish 9 Intro to Spanish 3 9

10 Automotive mechanics Intro 10 Automotive mechanics Intro 3 10

11 Advanced Robotics 11 Advanced Robotics 6 4

11 rows selected.

SQL> UPDATE InstructorDBA.Course\_List SET Course\_Name = 'test';

11 rows updated.

SQL> INSERT INTO InstructorDBA.Course\_List (Course\_Num, Course\_Name, Admin\_Num , Course\_Desc, Course\_Hours, Section\_Num)

2 VALUES (8888, 'Intro to Computer Science',1, 'Intro to Computer Science','3', '1');

1 row inserted.

SQL> SELECT \* FROM InstructorDBA.Course\_List;

COURSE\_NUM COURSE\_NAME ADMIN\_NUM COURSE\_DESC COURSE\_HOURS SECTION\_NUM

1 test 1 Intro to Computer Science 3 1

2 test 2 Intro to Art 3 2

3 test 3 Intro to Aircraft mechanics 3 3

4 test 4 Intro to Robotics 3 4

5 test 5 Intro to Databases 3 5

6 test 6 Advanced English 3 6

7 test 7 Intro to Physics 3 7

8 test 8 Intro to Chemistry 3 8

9 test 9 Intro to Spanish 3 9

10 test 10 Automotive mechanics Intro 3 10

11 test 11 Advanced Robotics 6 4

8888 Intro to Computer Science 1 Intro to Computer Science 3 1

12 rows selected.

SQL> DELETE FROM InstructorDBA.Course\_List;

12 rows deleted.

SQL> ROLLBACK TO Course\_Account\_Modify\_Procedure;

Rollback complete.

SQL>

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Course\_List;

Error starting at line : 195 in command -

SELECT \* FROM InstructorDBA.Course\_List

Error at Command Line : 195 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Course\_List;

Error starting at line : 200 in command -

SELECT \* FROM InstructorDBA.Course\_List

Error at Command Line : 200 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

SQL>

Section Management Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Section Management Procedure - Admins have full rights to section information\*/

SQL> GRANT SELECT, INSERT, DELETE, UPDATE ON Section\_Info TO admin\_role;

Grant succeeded.

SQL>

SQL> /\*Test Section Management Procedure\*/

SQL> /\*Test admin access\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SAVEPOINT Section\_Account\_Modify\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Section\_Info;

SECTION\_NUM SECTION\_NAME SECTION\_ADDRESS SECTION\_FOS SECTION\_ZIPCODE

1 Computer Science Section 110 Finegand Place Computer Science 31088

2 Art Section 111 Finegand Place Art 31088

3 Aircraft Section 112 Finegand Place Aircraft 31088

4 Robotic Section 113 Finegand Place Robotics 31088

5 Database Section 114 Finegand Place Database Technology 31088

6 English Section 115 Finegand Place English language 31088

7 Physics Section 116 Finegand Place Physics 31088

8 Chemistry Section 117 Finegand Place Chemistry 31088

9 Spanish Section 118 Finegand Place Spanish language 31088

10 Automotive Section 119 Finegand Place Automotive Mechanics 31088

11 Space Travel Section 120 Finegand Place Space Travel 31088

11 rows selected.

SQL> UPDATE InstructorDBA.Section\_Info SET Section\_Name = 'test';

11 rows updated.

SQL> INSERT INTO InstructorDBA.Section\_Info (Section\_Num, Section\_Name, Section\_Address, Section\_FoS, Section\_ZipCode)

2 VALUES (9999, 'Space Travel Section', '120 Finegand Place','Space Travel','31088');

1 row inserted.

SQL> SELECT \* FROM InstructorDBA.Section\_Info;

SECTION\_NUM SECTION\_NAME SECTION\_ADDRESS SECTION\_FOS SECTION\_ZIPCODE

1 test 110 Finegand Place Computer Science 31088

2 test 111 Finegand Place Art 31088

3 test 112 Finegand Place Aircraft 31088

4 test 113 Finegand Place Robotics 31088

5 test 114 Finegand Place Database Technology 31088

6 test 115 Finegand Place English language 31088

7 test 116 Finegand Place Physics 31088

8 test 117 Finegand Place Chemistry 31088

9 test 118 Finegand Place Spanish language 31088

10 test 119 Finegand Place Automotive Mechanics 31088

11 test 120 Finegand Place Space Travel 31088

9999 Space Travel Section 120 Finegand Place Space Travel 31088

12 rows selected.

SQL> DELETE FROM InstructorDBA.Section\_Info WHERE Section\_Num = 9999;

1 row deleted.

SQL> ROLLBACK TO Section\_Account\_Modify\_Procedure;

Rollback complete.

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Section\_Info;

Error starting at line : 224 in command -

SELECT \* FROM InstructorDBA.Section\_Info

Error at Command Line : 224 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Section\_Info;

Error starting at line : 228 in command -

SELECT \* FROM InstructorDBA.Section\_Info

Error at Command Line : 228 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

Class Schedule Modify Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Class Schedule Modify Procedure - Admins have full rights to the class schedule\*/

SQL> GRANT SELECT, INSERT, DELETE, UPDATE ON Class\_Sched TO admin\_role;

Grant succeeded.

SQL>

SQL> /\*Test Class Schedule Management Procedure\*/

SQL> /\*Test admin access\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SAVEPOINT Schedule\_Account\_Modify\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

12 rows selected.

SQL> UPDATE InstructorDBA.Class\_Sched SET Sched\_Notes = 'test';

12 rows updated.

SQL> INSERT INTO InstructorDBA.Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes)

2 VALUES (9999,'Sun', '0900-1100','Oct. 21st', 'Weekend Schedule');

1 row inserted.

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th test 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th test 30

3 Mon-Wed 1000-1200 Oct. 14th test 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th test 30

5 Mon-Tues 0800-0900 Oct. 16th test 30

6 Tues-Wed 1300-1500 Oct. 17th test 30

7 Thur-Fri 1300-1500 Oct. 18th test 30

8 Wed-Fri 1600-1800 Oct. 19th test 30

9 Sat 0900-1100 Oct. 20th test 30

10 Sun 0900-1100 Oct. 21st test 30

11 Mon 0900-0930 6 days a week test 40

12 Tues 1100-1115 6 days a week test 40

9999 Sun 0900-1100 Oct. 21st Weekend Schedule

13 rows selected.

SQL> DELETE FROM InstructorDBA.Class\_Sched WHERE Sched\_Num = 9999;

1 row deleted.

SQL> ROLLBACK TO Schedule\_Account\_Modify\_Procedure;

Rollback complete.

SQL>

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

Error starting at line : 252 in command -

SELECT \* FROM InstructorDBA.Class\_Sched

Error at Command Line : 252 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

Error starting at line : 257 in command -

SELECT \* FROM InstructorDBA.Class\_Sched

Error at Command Line : 257 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

Admin Support Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Admin Support Procedure - Admins have the ability to assist with adding classes for instructors or signing up students

SQL> for classes if necessary.\*/

SQL> GRANT SELECT, INSERT, DELETE, UPDATE ON Instr\_Classes TO admin\_role;

Grant succeeded.

SQL> GRANT SELECT, INSERT, DELETE, UPDATE ON Student\_Class\_Signup TO admin\_role;

Grant succeeded.

SQL>

SQL> /\*Test Admin Support Schedule Management Procedure\*/

SQL> /\*Test admin access\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SAVEPOINT Admin\_Support\_Account\_Modify\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Instr\_Classes;

INSTR\_NUM SCHED\_NUM COURSE\_NUM CLASS\_NOTES CLASS\_ROOM

1 1 1 Computer one broke A

2 2 2 Room Ready B

3 3 3 Instructor Chair broke C

4 4 4 A/C not working D

5 5 5 Computer two broke E

6 6 6 No whiteboard F

7 7 7 station 3 missing keyboard G

8 8 8 Room Ready H

9 9 9 Room Ready I

10 10 10 Room Ready J

10 9 10 Room Ready J

9 8 10 Room Ready X

1 2 9 Room Ready Z

13 rows selected.

SQL> SELECT \* FROM InstructorDBA.Student\_Class\_Signup;

STUDENT\_NUM INSTR\_NUM SCHED\_NUM STUDENT\_GRADE

1 1 1 A

1 2 2 B

2 3 3 A

3 3 3 C

4 4 4 C

5 5 5 B

6 6 6 D

7 7 7 F

8 8 8

9 9 9 A

10 10 10 A

11 rows selected.

SQL> UPDATE InstructorDBA.Instr\_Classes SET Class\_Notes = 'test';

13 rows updated.

SQL> UPDATE InstructorDBA.Student\_Class\_Signup SET Student\_Grade = 'test';

11 rows updated.

SQL>

SQL> INSERT INTO InstructorDBA.Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

2 VALUES ('8', '9', '9', 'Room Ready','Z');

1 row inserted.

SQL>

SQL>

SQL> INSERT INTO InstructorDBA.Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES (10, 9, 9, 'A');

1 row inserted.

SQL>

SQL> SELECT \* FROM InstructorDBA.Instr\_Classes;

INSTR\_NUM SCHED\_NUM COURSE\_NUM CLASS\_NOTES CLASS\_ROOM

1 1 1 test A

2 2 2 test B

3 3 3 test C

4 4 4 test D

5 5 5 test E

6 6 6 test F

7 7 7 test G

8 8 8 test H

9 9 9 test I

10 10 10 test J

10 9 10 test J

9 8 10 test X

1 2 9 test Z

8 9 9 Room Ready Z

14 rows selected.

SQL> SELECT \* FROM InstructorDBA.Student\_Class\_Signup;

STUDENT\_NUM INSTR\_NUM SCHED\_NUM STUDENT\_GRADE

1 1 1 test

1 2 2 test

2 3 3 test

3 3 3 test

4 4 4 test

5 5 5 test

6 6 6 test

7 7 7 test

8 8 8 test

9 9 9 test

10 10 10 test

10 9 9 A

12 rows selected.

SQL>

SQL> DELETE FROM InstructorDBA.Instr\_Classes WHERE Instr\_Num = 8 AND Sched\_Num = 9;

1 row deleted.

SQL> DELETE FROM InstructorDBA.Student\_Class\_Signup WHERE Student\_Num = 10 AND Instr\_Num = 9 AND Sched\_Num = 9;

1 row deleted.

SQL> ROLLBACK TO Admin\_Support\_Account\_Modify\_Procedure;

Rollback complete.

SQL>

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL>

SQL> INSERT INTO InstructorDBA.Instr\_Classes (Instr\_Num, Sched\_Num, Course\_Num, Class\_Notes, Class\_Room)

2 VALUES ('8', '9', '9', 'Room Ready','Z');

Error starting at line : 295 in command -

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

VALUES ('8', '9', '9', 'Room Ready','Z')

Error at Command Line : 295 Column : 27

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> DELETE FROM InstructorDBA.Student\_Class\_Signup WHERE Student\_Num = 1 AND Instr\_Num = 1 AND Sched\_Num = 1;

Error starting at line : 301 in command -

DELETE FROM InstructorDBA.Student\_Class\_Signup WHERE Student\_Num = 1 AND Instr\_Num = 1 AND Sched\_Num = 1

Error at Command Line : 301 Column : 27

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

SQL>

Admin Account Restriction Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Admin Account Restriction Procedure - Admins can only select & update the Admin\_List table.\*/

SQL> GRANT SELECT, UPDATE (Admin\_Address, Admin\_PhoneNum, Admin\_LName, Admin\_FName, Admin\_ZipCode) ON Admin\_List TO admin\_role;

Grant succeeded.

SQL> /\*Use VPD to create policy restricting admin to only be able to view and update their information. Only the InstructorDBA user

SQL> can update this table.\*/

SQL> CREATE OR REPLACE FUNCTION Get\_Admin\_Name (

2 schema\_v IN VARCHAR2,

3 tbl\_v IN VARCHAR2)

4

5 RETURN VARCHAR2 IS

6 BEGIN

7 RETURN ('User\_Name = USER OR USER = ''InstructorDBA''') ;

8 END Get\_Admin\_Name;

9 /

Function GET\_ADMIN\_NAME compiled

SQL> BEGIN

2 DBMS\_RLS.ADD\_POLICY (

3 object\_schema => 'InstructorDBA',

4 object\_name => 'Admin\_List',

5 policy\_name => 'Hide\_Admin\_Info',

6 policy\_function => 'Get\_Admin\_Name',

7 statement\_types => 'update',

8 update\_check => TRUE);

9 END;

10 /

PL/SQL procedure successfully completed.

SQL>

SQL> /\*Test Admin restriction policy\*/

SQL> /\*Test admin account\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SAVEPOINT Admin\_Restriction\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Admin\_List;

ADMIN\_NUM ADMIN\_ADDRESS ADMIN\_PHONENUM SECTION\_NUM ADMIN\_LNAME ADMIN\_FNAME ADMIN\_ZIPCODE USER\_NAME

1 501 Orange Park 111-333-3434 1 Evans Bob 31004 BEVANS1

2 502 Orange Park 111-333-3435 2 Johnson Mike 31004 MJOHNSON2

3 503 Orange Park 111-333-3436 3 Jenkins Mary 31004 MJENKINS3

4 504 Orange Park 111-333-3437 4 Russell Jim 31004 JRUSSELL4

5 505 Orange Park 111-333-3438 5 Bargueno Patricia 31004 PBARGUENO5

6 506 Orange Park 111-333-3439 6 Hopkins John 31004 JHOPKINS6

7 507 Orange Park 111-333-3440 7 Smith Will 31004 WSMITH7

8 508 Orange Park 111-333-3441 8 Brown Adam 31004 ABROWN8

9 509 Orange Park 111-333-3442 9 Williams Shane 31004 SWILLIAMS9

10 510 Orange Park 111-333-3443 10 Robins Amber 31004 AROBINS10

11 511 Orange Park 111-333-3444 4 Hawkins Tim 31004 THAWKINS11

11 rows selected.

SQL> UPDATE InstructorDBA.Admin\_List SET Admin\_Address = 'Changeit!';

1 row updated.

SQL> UPDATE InstructorDBA.Admin\_List SET Admin\_Num = 1234555;

Error starting at line : 338 in command -

UPDATE InstructorDBA.Admin\_List SET Admin\_Num = 1234555

Error at Command Line : 338 Column : 22

Error report -

SQL Error: ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without

the necessary privileges.

\*Action: Ask your database administrator or designated security

administrator to grant you the necessary privileges

SQL> SELECT \* FROM InstructorDBA.Admin\_List;

ADMIN\_NUM ADMIN\_ADDRESS ADMIN\_PHONENUM SECTION\_NUM ADMIN\_LNAME ADMIN\_FNAME ADMIN\_ZIPCODE USER\_NAME

1 Changeit! 111-333-3434 1 Evans Bob 31004 BEVANS1

2 502 Orange Park 111-333-3435 2 Johnson Mike 31004 MJOHNSON2

3 503 Orange Park 111-333-3436 3 Jenkins Mary 31004 MJENKINS3

4 504 Orange Park 111-333-3437 4 Russell Jim 31004 JRUSSELL4

5 505 Orange Park 111-333-3438 5 Bargueno Patricia 31004 PBARGUENO5

6 506 Orange Park 111-333-3439 6 Hopkins John 31004 JHOPKINS6

7 507 Orange Park 111-333-3440 7 Smith Will 31004 WSMITH7

8 508 Orange Park 111-333-3441 8 Brown Adam 31004 ABROWN8

9 509 Orange Park 111-333-3442 9 Williams Shane 31004 SWILLIAMS9

10 510 Orange Park 111-333-3443 10 Robins Amber 31004 AROBINS10

11 511 Orange Park 111-333-3444 4 Hawkins Tim 31004 THAWKINS11

11 rows selected.

SQL> ROLLBACK TO Admin\_Restriction\_Procedure;

Rollback complete.

SQL> /\*Test instructor account\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Admin\_List;

Error starting at line : 344 in command -

SELECT \* FROM InstructorDBA.Admin\_List

Error at Command Line : 344 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM Admin\_List;

Error starting at line : 349 in command -

SELECT \* FROM Admin\_List

Error at Command Line : 349 Column : 15

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test InstructorDBA account\*/

SQL> CONNECT InstructorDBA/brr1wik7

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL> SELECT \* FROM Admin\_List;

ADMIN\_NUM ADMIN\_ADDRESS ADMIN\_PHONENUM SECTION\_NUM ADMIN\_LNAME ADMIN\_FNAME ADMIN\_ZIPCODE USER\_NAME

1 501 Orange Park 111-333-3434 1 Evans Bob 31004 BEVANS1

2 502 Orange Park 111-333-3435 2 Johnson Mike 31004 MJOHNSON2

3 503 Orange Park 111-333-3436 3 Jenkins Mary 31004 MJENKINS3

4 504 Orange Park 111-333-3437 4 Russell Jim 31004 JRUSSELL4

5 505 Orange Park 111-333-3438 5 Bargueno Patricia 31004 PBARGUENO5

6 506 Orange Park 111-333-3439 6 Hopkins John 31004 JHOPKINS6

7 507 Orange Park 111-333-3440 7 Smith Will 31004 WSMITH7

8 508 Orange Park 111-333-3441 8 Brown Adam 31004 ABROWN8

9 509 Orange Park 111-333-3442 9 Williams Shane 31004 SWILLIAMS9

10 510 Orange Park 111-333-3443 10 Robins Amber 31004 AROBINS10

11 511 Orange Park 111-333-3444 4 Hawkins Tim 31004 THAWKINS11

11 rows selected.

SQL>

Instructor View Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Instructor View Procedure - Personal info, section info, class schedule with course info, name and student number enrolled in classes.

SQL>Also, complete class schedule without student names.\*/

SQL> /\*Show personal info\*/

SQL> CREATE OR REPLACE VIEW Instr\_Personal\_Info AS

2 SELECT \* FROM Instr\_List

3 WHERE Instr\_List.User\_Name = USER WITH CHECK OPTION;

View INSTR\_PERSONAL\_INFO created.

SQL> GRANT SELECT ON Instr\_Personal\_Info TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Show section info\*/

SQL> CREATE OR REPLACE VIEW Instr\_Section\_Info AS

2 SELECT \* FROM Section\_Info

3 INNER JOIN Instr\_List USING (Section\_Num)

4 WHERE Instr\_List.User\_Name = USER WITH CHECK OPTION;

View INSTR\_SECTION\_INFO created.

SQL> GRANT SELECT ON Instr\_Section\_Info TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Show class schedules info\*/

SQL> GRANT SELECT ON Class\_Sched TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Show course info\*/

SQL> GRANT SELECT ON Course\_List TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Show class schedule and course info\*/

SQL> CREATE OR REPLACE VIEW Instr\_Class\_Course\_Info AS

2 SELECT Course\_Name, Course\_Desc, Course\_Hours, Admin\_LName, Admin\_FName,

3 Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes,

4 Class\_Notes, Class\_Room FROM Instr\_List INNER JOIN Instr\_Classes USING (Instr\_Num)

5 INNER JOIN Class\_Sched USING (Sched\_Num)

6 INNER JOIN Course\_List USING (Course\_Num)

7 INNER JOIN Admin\_List USING (Admin\_Num)

8 WHERE Instr\_List.User\_Name = USER WITH CHECK OPTION;

View INSTR\_CLASS\_COURSE\_INFO created.

SQL> GRANT SELECT ON Instr\_Class\_Course\_Info TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Show students enrolled in classes\*/

SQL> CREATE OR REPLACE VIEW Instr\_Student\_Class AS

2 SELECT Course\_Name, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Class\_Room,

3 Student\_Num, Student\_LName, Student\_FName, Student\_Grade FROM Instr\_List INNER JOIN Instr\_Classes USING (Instr\_Num)

4 INNER JOIN Class\_Sched USING (Sched\_Num)

5 INNER JOIN Course\_List USING (Course\_Num)

6 INNER JOIN Student\_Class\_Signup USING (Instr\_Num, Sched\_Num)

7 INNER JOIN Student\_List USING (Student\_Num)

8 WHERE Instr\_List.User\_Name = USER WITH CHECK OPTION;

View INSTR\_STUDENT\_CLASS created.

SQL> GRANT SELECT ON Instr\_Student\_Class TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Show entire class schedule\*/

SQL> CREATE OR REPLACE VIEW Instr\_All\_Class AS

2 SELECT Instr\_LName, Instr\_FName, Course\_Name, Course\_Desc, Course\_Hours, Admin\_LName, Admin\_FName,

3 Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes,

4 Class\_Notes, Class\_Room FROM Instr\_List INNER JOIN Instr\_Classes USING (Instr\_Num)

5 INNER JOIN Class\_Sched USING (Sched\_Num)

6 INNER JOIN Course\_List USING (Course\_Num)

7 INNER JOIN Admin\_List USING (Admin\_Num);

View INSTR\_ALL\_CLASS created.

SQL> GRANT SELECT ON Instr\_All\_Class TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Test Instructor View Policy\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Instr\_Personal\_Info;

INSTR\_NUM INSTR\_ADDRESS INSTR\_PHONENUM SECTION\_NUM INSTR\_LNAME INSTR\_FNAME INSTR\_ZIPCODE USER\_NAME

3 103 Apple Street 111-222-3436 3 Lopez Mike 31003 MLOPEZ3

SQL> SELECT \* FROM InstructorDBA.Instr\_Section\_Info;

SECTION\_NUM SECTION\_NAME SECTION\_ADDRESS SECTION\_FOS SECTION\_ZIPCODE INSTR\_NUM INSTR\_ADDRESS INSTR\_PHONENUM INSTR\_LNAME INSTR\_FNAME INSTR\_ZIPCODE USER\_NAME

3 Aircraft Section 112 Finegand Place Aircraft 31088 3 103 Apple Street 111-222-3436 Lopez Mike 31003 MLOPEZ3

SQL> SELECT \* FROM InstructorDBA.Instr\_Class\_Course\_Info;

COURSE\_NAME COURSE\_DESC COURSE\_HOURS ADMIN\_LNAME ADMIN\_FNAME SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES CLASS\_NOTES CLASS\_ROOM

Intro to Aircraft mechanics Intro to Aircraft mechanics 3 Jenkins Mary Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules Instructor Chair broke C

SQL> SELECT \* FROM InstructorDBA.Instr\_Student\_Class;

COURSE\_NAME SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF CLASS\_ROOM STUDENT\_NUM STUDENT\_LNAME STUDENT\_FNAME STUDENT\_GRADE

Intro to Aircraft mechanics Mon-Wed 1000-1200 Oct. 14th C 2 Myers James A

Intro to Aircraft mechanics Mon-Wed 1000-1200 Oct. 14th C 3 Howard Rico C

SQL> SELECT \* FROM InstructorDBA.Instr\_All\_Class;

INSTR\_LNAME INSTR\_FNAME COURSE\_NAME COURSE\_DESC COURSE\_HOURS ADMIN\_LNAME ADMIN\_FNAME SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES CLASS\_NOTES CLASS\_ROOM

Roberts John Intro to Computer Science Intro to Computer Science 3 Evans Bob Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules Computer one broke A

Russell Brandon Intro to Art Intro to Art 3 Johnson Mike Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules Room Ready B

Lopez Mike Intro to Aircraft mechanics Intro to Aircraft mechanics 3 Jenkins Mary Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules Instructor Chair broke C

Monteor Jim Intro to Robotics Intro to Robotics 3 Russell Jim Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule A/C not working D

Hamm Patricia Intro to Databases Intro to Databases 3 Bargueno Patricia Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule Computer two broke E

Ingle Tammy Advanced English Advanced English 3 Hopkins John Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule No whiteboard F

Jordan Keith Intro to Physics Intro to Physics 3 Smith Will Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule station 3 missing keyboard G

Cooper Adam Intro to Chemistry Intro to Chemistry 3 Brown Adam Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule Room Ready H

McCoy Amber Intro to Spanish Intro to Spanish 3 Williams Shane Sat 0900-1100 Oct. 20th Weekend Schedule Room Ready I

Roberts John Intro to Spanish Intro to Spanish 3 Williams Shane Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules Room Ready Z

Fuller Jana Automotive mechanics Intro Automotive mechanics Intro 3 Robins Amber Sun 0900-1100 Oct. 21st Weekend Schedule Room Ready J

Fuller Jana Automotive mechanics Intro Automotive mechanics Intro 3 Robins Amber Sat 0900-1100 Oct. 20th Weekend Schedule Room Ready J

McCoy Amber Automotive mechanics Intro Automotive mechanics Intro 3 Robins Amber Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule Room Ready X

13 rows selected.

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

12 rows selected.

SQL> SELECT \* FROM InstructorDBA.Course\_List;

COURSE\_NUM COURSE\_NAME ADMIN\_NUM COURSE\_DESC COURSE\_HOURS SECTION\_NUM

1 Intro to Computer Science 1 Intro to Computer Science 3 1

2 Intro to Art 2 Intro to Art 3 2

3 Intro to Aircraft mechanics 3 Intro to Aircraft mechanics 3 3

4 Intro to Robotics 4 Intro to Robotics 3 4

5 Intro to Databases 5 Intro to Databases 3 5

6 Advanced English 6 Advanced English 3 6

7 Intro to Physics 7 Intro to Physics 3 7

8 Intro to Chemistry 8 Intro to Chemistry 3 8

9 Intro to Spanish 9 Intro to Spanish 3 9

10 Automotive mechanics Intro 10 Automotive mechanics Intro 3 10

11 Advanced Robotics 11 Advanced Robotics 6 4

11 rows selected.

SQL>

SQL> /\*Test student ability to view\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Instr\_Personal\_Info;

Error starting at line : 423 in command -

SELECT \* FROM InstructorDBA.Instr\_Personal\_Info

Error at Command Line : 423 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Instr\_Section\_Info;

Error starting at line : 424 in command -

SELECT \* FROM InstructorDBA.Instr\_Section\_Info

Error at Command Line : 424 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Instr\_Class\_Course\_Info;

Error starting at line : 425 in command -

SELECT \* FROM InstructorDBA.Instr\_Class\_Course\_Info

Error at Command Line : 425 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Instr\_Student\_Class;

Error starting at line : 426 in command -

SELECT \* FROM InstructorDBA.Instr\_Student\_Class

Error at Command Line : 426 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Instr\_All\_Class;

Error starting at line : 427 in command -

SELECT \* FROM InstructorDBA.Instr\_All\_Class

Error at Command Line : 427 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

Error starting at line : 428 in command -

SELECT \* FROM InstructorDBA.Class\_Sched

Error at Command Line : 428 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Course\_List;

Error starting at line : 429 in command -

SELECT \* FROM InstructorDBA.Course\_List

Error at Command Line : 429 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

SQL>

Instructor Modify Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Instructor Modify Procedure - update personal info, class notes, student grades enrolled in class, remove students\*/

SQL>

SQL> /\*Update personal info\*/

SQL> GRANT UPDATE (Instr\_Address, Instr\_PhoneNum, Instr\_LName, Instr\_FName, Instr\_ZipCode) ON Instr\_Personal\_Info TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Update their class schedule\*/

SQL> CREATE OR REPLACE VIEW Instr\_Classes\_Update AS

2 SELECT Instr\_Classes.SCHED\_NUM SCHED\_NUM,

3 Instr\_Classes.COURSE\_NUM COURSE\_NUM,

4 Instr\_Classes.CLASS\_NOTES CLASS\_NOTES,

5 Instr\_Classes.CLASS\_ROOM CLASS\_ROOM FROM Instr\_Classes INNER JOIN Instr\_List USING (Instr\_Num)

6 WHERE Instr\_List.User\_Name = USER WITH CHECK OPTION;

View INSTR\_CLASSES\_UPDATE created.

SQL> GRANT SELECT, UPDATE (Class\_Notes) ON Instr\_Classes\_Update TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Update student grade enrolled in their classes. Remove students from their classes\*/

SQL> CREATE OR REPLACE VIEW Instr\_Student\_Modify AS

2 SELECT Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade FROM Instr\_List INNER JOIN Instr\_Classes USING (Instr\_Num)

3 INNER JOIN Student\_Class\_Signup USING (Instr\_Num, Sched\_Num)

4 WHERE Instr\_List.User\_Name = USER WITH CHECK OPTION;

View INSTR\_STUDENT\_MODIFY created.

SQL> GRANT SELECT, UPDATE (Student\_Grade) , DELETE ON Instr\_Student\_Modify TO instructor\_role;

Grant succeeded.

SQL>

SQL> /\*Test instructor update on personal info\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SAVEPOINT Instr\_Modify\_Personal\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Instr\_Personal\_Info;

INSTR\_NUM INSTR\_ADDRESS INSTR\_PHONENUM SECTION\_NUM INSTR\_LNAME INSTR\_FNAME INSTR\_ZIPCODE USER\_NAME

3 103 Apple Street 111-222-3436 3 Lopez Mike 31003 MLOPEZ3

SQL> UPDATE InstructorDBA.Instr\_Personal\_Info SET Instr\_Address = '867 Orange St';

1 row updated.

SQL> UPDATE InstructorDBA.Instr\_Personal\_Info SET User\_Name = 'BillyBob7';

Error starting at line : 462 in command -

UPDATE InstructorDBA.Instr\_Personal\_Info SET User\_Name = 'BillyBob7'

Error at Command Line : 462 Column : 22

Error report -

SQL Error: ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without

the necessary privileges.

\*Action: Ask your database administrator or designated security

administrator to grant you the necessary privileges

SQL> SELECT \* FROM InstructorDBA.Instr\_Personal\_Info;

INSTR\_NUM INSTR\_ADDRESS INSTR\_PHONENUM SECTION\_NUM INSTR\_LNAME INSTR\_FNAME INSTR\_ZIPCODE USER\_NAME

3 867 Orange St 111-222-3436 3 Lopez Mike 31003 MLOPEZ3

SQL> ROLLBACK TO Instr\_Modify\_Personal\_Procedure;

Rollback complete.

SQL> /\*Test student account access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> UPDATE InstructorDBA.Instr\_Personal\_Info SET Instr\_Address = '867 Orange St';

Error starting at line : 468 in command -

UPDATE InstructorDBA.Instr\_Personal\_Info SET Instr\_Address = '867 Orange St'

Error at Command Line : 468 Column : 22

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Instr\_Personal\_Info;

Error starting at line : 469 in command -

SELECT \* FROM InstructorDBA.Instr\_Personal\_Info

Error at Command Line : 469 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test instructor ability to select and update on their class schedule\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SAVEPOINT Instr\_Modify\_Class\_Procedure;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Instr\_Classes\_Update;

SCHED\_NUM COURSE\_NUM CLASS\_NOTES CLASS\_ROOM

3 3 Instructor Chair broke C

SQL> UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Notes = 'A/C Broke' WHERE Sched\_Num = 3;

1 row updated.

SQL> UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Room = 'Party\_Room' WHERE Sched\_Num = 3;

Error starting at line : 477 in command -

UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Room = 'Party\_Room' WHERE Sched\_Num = 3

Error at Command Line : 477 Column : 22

Error report -

SQL Error: ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without

the necessary privileges.

\*Action: Ask your database administrator or designated security

administrator to grant you the necessary privileges

SQL> SELECT \* FROM InstructorDBA.Instr\_Classes\_Update;

SCHED\_NUM COURSE\_NUM CLASS\_NOTES CLASS\_ROOM

3 3 A/C Broke C

SQL> ROLLBACK TO Instr\_Modify\_Class\_Procedure;

Rollback complete.

SQL> /\*Test Select & Update with instructor trying to update another instructor's class\*/

SQL> CONNECT brussell2/TheSecPass1;

Connected.

SQL> show user;

USER is "BRUSSELL2"

SQL> SELECT \* FROM InstructorDBA.Instr\_Classes\_Update;

SCHED\_NUM COURSE\_NUM CLASS\_NOTES CLASS\_ROOM

2 2 Room Ready B

SQL> UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Notes = 'A/C Broke' WHERE Sched\_Num = 3;

0 rows updated.

SQL> UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Notes = 'A/C Broke' WHERE Instr\_Num = 3 AND Sched\_Num = 3;

Error starting at line : 485 in command -

UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Notes = 'A/C Broke' WHERE Instr\_Num = 3 AND Sched\_Num = 3

Error at Command Line : 485 Column : 79

Error report -

SQL Error: ORA-00904: "INSTR\_NUM": invalid identifier

00904. 00000 - "%s: invalid identifier"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Instr\_Classes\_Update;

SCHED\_NUM COURSE\_NUM CLASS\_NOTES CLASS\_ROOM

2 2 Room Ready B

SQL>

SQL> /\*Test student account access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Instr\_Classes\_Update;

Error starting at line : 491 in command -

SELECT \* FROM InstructorDBA.Instr\_Classes\_Update

Error at Command Line : 491 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Notes = 'A/C Broke' WHERE Sched\_Num = 3;

Error starting at line : 492 in command -

UPDATE InstructorDBA.Instr\_Classes\_Update SET Class\_Notes = 'A/C Broke' WHERE Sched\_Num = 3

Error at Command Line : 492 Column : 22

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test instructor ability to select, update, and delete students from their class\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SAVEPOINT Instr\_Modify\_Delete\_Point;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Instr\_Student\_Modify;

STUDENT\_NUM INSTR\_NUM SCHED\_NUM STUDENT\_GRADE

2 3 3 A

3 3 3 C

SQL> UPDATE InstructorDBA.Instr\_Student\_Modify SET Student\_Grade = 'F' WHERE Student\_Num = 2;

1 row updated.

SQL> UPDATE InstructorDBA.Instr\_Student\_Modify SET Student\_Grade = 'F' WHERE Student\_Num = 3;

1 row updated.

SQL> UPDATE InstructorDBA.Instr\_Student\_Modify SET Student\_Num = 4 WHERE Student\_Num = 3;

Error starting at line : 501 in command -

UPDATE InstructorDBA.Instr\_Student\_Modify SET Student\_Num = 4 WHERE Student\_Num = 3

Error at Command Line : 501 Column : 22

Error report -

SQL Error: ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without

the necessary privileges.

\*Action: Ask your database administrator or designated security

administrator to grant you the necessary privileges

SQL> SELECT \* FROM InstructorDBA.Instr\_Student\_Modify;

STUDENT\_NUM INSTR\_NUM SCHED\_NUM STUDENT\_GRADE

2 3 3 F

3 3 3 F

SQL> DELETE FROM InstructorDBA.Instr\_Student\_Modify WHERE Student\_Num = 3;

1 row deleted.

SQL> SELECT \* FROM InstructorDBA.Instr\_Student\_Modify;

STUDENT\_NUM INSTR\_NUM SCHED\_NUM STUDENT\_GRADE

2 3 3 F

SQL> ROLLBACK TO Instr\_Modify\_Delete\_Point;

Rollback complete.

SQL>

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Instr\_Student\_Modify;

Error starting at line : 510 in command -

SELECT \* FROM InstructorDBA.Instr\_Student\_Modify

Error at Command Line : 510 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> UPDATE InstructorDBA.Instr\_Student\_Modify SET Student\_Grade = 'F' WHERE Student\_Num = 2;

Error starting at line : 511 in command -

UPDATE InstructorDBA.Instr\_Student\_Modify SET Student\_Grade = 'F' WHERE Student\_Num = 2

Error at Command Line : 511 Column : 22

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> DELETE FROM InstructorDBA.Instr\_Student\_Modify WHERE Student\_Num = 2;

Error starting at line : 512 in command -

DELETE FROM InstructorDBA.Instr\_Student\_Modify WHERE Student\_Num = 2

Error at Command Line : 512 Column : 27

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL> SELECT \* FROM InstructorDBA.Instr\_Student\_Modify;

Error starting at line : 513 in command -

SELECT \* FROM InstructorDBA.Instr\_Student\_Modify

Error at Command Line : 513 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

SQL>

Student View Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Student View Procedure - personal info, instructor class schedule, classes enrolled in, grades for classes\*/

SQL> /\*Show personal info\*/

SQL> CREATE OR REPLACE VIEW Student\_Personal\_Info AS

2 SELECT \* FROM Student\_List

3 WHERE Student\_List.User\_Name = USER WITH CHECK OPTION;

View STUDENT\_PERSONAL\_INFO created.

SQL> GRANT SELECT ON Student\_Personal\_Info TO student\_role;

Grant succeeded.

SQL>

SQL> /\*View the instructor class schedule\*/

SQL> GRANT SELECT ON Instr\_All\_Class TO student\_role;

Grant succeeded.

SQL>

SQL>

SQL> /\*See what classes they are enrolled in and their grade\*/

SQL> CREATE OR REPLACE VIEW Student\_Class\_Grade\_View AS

2 SELECT Student\_LName, Student\_FName, Course\_Name, Student\_Grade,

3 Instr\_LName, Instr\_FName, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Class\_Room

4 FROM Student\_List INNER JOIN Student\_Class\_Signup USING (Student\_Num)

5 INNER JOIN Instr\_Classes USING (Instr\_Num, Sched\_Num)

6 INNER JOIN Class\_Sched USING (Sched\_Num)

7 INNER JOIN Course\_List USING (Course\_Num)

8 INNER JOIN Instr\_List USING (Instr\_Num)

9 WHERE Student\_List.User\_Name = USER WITH CHECK OPTION;

View STUDENT\_CLASS\_GRADE\_VIEW created.

SQL> GRANT SELECT ON Student\_Class\_Grade\_View TO student\_role;

Grant succeeded.

SQL>

SQL>

SQL> /\*Test Student View personal info\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Student\_Personal\_Info;

STUDENT\_NUM STUDENT\_ADDRESS STUDENT\_PHONENUM STUDENT\_LNAME STUDENT\_FNAME STUDENT\_ZIPCODE USER\_NAME

1 Dorm 1 222-333-3434 Gilbert Sarah 31005 SGILBERT1

SQL>

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Student\_Personal\_Info;

Error starting at line : 551 in command -

SELECT \* FROM InstructorDBA.Student\_Personal\_Info

Error at Command Line : 551 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test Student View instructor class schedule\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Instr\_All\_Class;

INSTR\_LNAME INSTR\_FNAME COURSE\_NAME COURSE\_DESC COURSE\_HOURS ADMIN\_LNAME ADMIN\_FNAME SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES CLASS\_NOTES CLASS\_ROOM

Roberts John Intro to Computer Science Intro to Computer Science 3 Evans Bob Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules Computer one broke A

Russell Brandon Intro to Art Intro to Art 3 Johnson Mike Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules Room Ready B

Lopez Mike Intro to Aircraft mechanics Intro to Aircraft mechanics 3 Jenkins Mary Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules Instructor Chair broke C

Monteor Jim Intro to Robotics Intro to Robotics 3 Russell Jim Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule A/C not working D

Hamm Patricia Intro to Databases Intro to Databases 3 Bargueno Patricia Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule Computer two broke E

Ingle Tammy Advanced English Advanced English 3 Hopkins John Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule No whiteboard F

Jordan Keith Intro to Physics Intro to Physics 3 Smith Will Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule station 3 missing keyboard G

Cooper Adam Intro to Chemistry Intro to Chemistry 3 Brown Adam Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule Room Ready H

McCoy Amber Intro to Spanish Intro to Spanish 3 Williams Shane Sat 0900-1100 Oct. 20th Weekend Schedule Room Ready I

Roberts John Intro to Spanish Intro to Spanish 3 Williams Shane Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules Room Ready Z

Fuller Jana Automotive mechanics Intro Automotive mechanics Intro 3 Robins Amber Sun 0900-1100 Oct. 21st Weekend Schedule Room Ready J

Fuller Jana Automotive mechanics Intro Automotive mechanics Intro 3 Robins Amber Sat 0900-1100 Oct. 20th Weekend Schedule Room Ready J

McCoy Amber Automotive mechanics Intro Automotive mechanics Intro 3 Robins Amber Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule Room Ready X

13 rows selected.

SQL>

SQL> /\*Test Student View classes enrolled in and grade\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Student\_Class\_Grade\_View;

STUDENT\_LNAME STUDENT\_FNAME COURSE\_NAME STUDENT\_GRADE INSTR\_LNAME INSTR\_FNAME SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF CLASS\_ROOM

Gilbert Sarah Intro to Computer Science A Roberts John Mon-Tues-Wed 0900-1100 Oct. 13th A

Gilbert Sarah Intro to Art B Russell Brandon Mon-Tues-Thur 0900-1100 Oct. 13th B

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Student\_Class\_Grade\_View;

Error starting at line : 565 in command -

SELECT \* FROM InstructorDBA.Student\_Class\_Grade\_View

Error at Command Line : 565 Column : 29

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

Student Modify Procedure. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Student Modify Procedure - update personal info and enroll in class\*/

SQL>

SQL> /\*Student update personal information\*/

SQL> GRANT UPDATE (Student\_Address, Student\_PhoneNum, Student\_LName, Student\_FName, Student\_Zipcode) ON Student\_Personal\_Info TO student\_role;

Grant succeeded.

SQL> /\*Student enroll in classes\*/

SQL> /\*Trigger for Student class enrollment - Prevent students from signing other students up for class.\*/

SQL> CREATE OR REPLACE TRIGGER Student\_Class\_Signup\_Trigger

2 BEFORE INSERT

3 ON Student\_Class\_Signup

4 FOR EACH ROW

5

6 DECLARE

7 Tmp\_Counter INTEGER;

8 Tmp\_Student\_Num INTEGER;

9 BEGIN

10 /\*Only run code for Student users\*/

11 SELECT count(\*) into Tmp\_Counter FROM Student\_List WHERE Student\_List.User\_Name = USER;

12 IF Tmp\_Counter > 0 THEN

13 SELECT Student\_Num INTO Tmp\_Student\_Num FROM Student\_List WHERE Student\_List.User\_Name = USER;

14 :new.Student\_Num := Tmp\_Student\_Num;

15 :new.Student\_Grade := '';

16 END IF;

17 END;

18 /

Trigger STUDENT\_CLASS\_SIGNUP\_TRIGGER compiled

SQL> GRANT INSERT ON Student\_Class\_Signup TO student\_role;

Grant succeeded.

SQL>

SQL> /\*Test update personal info\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SAVEPOINT Student\_Modify\_Personal\_Point;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Student\_Personal\_Info;

STUDENT\_NUM STUDENT\_ADDRESS STUDENT\_PHONENUM STUDENT\_LNAME STUDENT\_FNAME STUDENT\_ZIPCODE USER\_NAME

1 Dorm 1 222-333-3434 Gilbert Sarah 31005 SGILBERT1

SQL> UPDATE InstructorDBA.Student\_Personal\_Info SET Student\_ZipCode = '90210';

1 row updated.

SQL> UPDATE InstructorDBA.Student\_Personal\_Info SET User\_Name = 'HackerMode';

Error starting at line : 601 in command -

UPDATE InstructorDBA.Student\_Personal\_Info SET User\_Name = 'HackerMode'

Error at Command Line : 601 Column : 22

Error report -

SQL Error: ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without

the necessary privileges.

\*Action: Ask your database administrator or designated security

administrator to grant you the necessary privileges

SQL> SELECT \* FROM InstructorDBA.Student\_Personal\_Info;

STUDENT\_NUM STUDENT\_ADDRESS STUDENT\_PHONENUM STUDENT\_LNAME STUDENT\_FNAME STUDENT\_ZIPCODE USER\_NAME

1 Dorm 1 222-333-3434 Gilbert Sarah 90210 SGILBERT1

SQL> ROLLBACK TO Student\_Modify\_Personal\_Point;

Rollback complete.

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> UPDATE InstructorDBA.Student\_Personal\_Info SET Student\_ZipCode = '90210';

Error starting at line : 607 in command -

UPDATE InstructorDBA.Student\_Personal\_Info SET Student\_ZipCode = '90210'

Error at Command Line : 607 Column : 22

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL>

SQL> /\*Test enroll in class\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SAVEPOINT Student\_Class\_Signup\_Point;

Savepoint created.

SQL> SELECT \* FROM InstructorDBA.Student\_Class\_Grade\_View;

STUDENT\_LNAME STUDENT\_FNAME COURSE\_NAME STUDENT\_GRADE INSTR\_LNAME INSTR\_FNAME SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF CLASS\_ROOM

Gilbert Sarah Intro to Computer Science A Roberts John Mon-Tues-Wed 0900-1100 Oct. 13th A

Gilbert Sarah Intro to Art B Russell Brandon Mon-Tues-Thur 0900-1100 Oct. 13th B

SQL> INSERT INTO InstructorDBA.Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES ( 3,6, 6, 'A');

1 row inserted.

SQL>

SQL> INSERT INTO InstructorDBA.Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES ( 1,3, 3, 'A');

1 row inserted.

SQL>

SQL> SELECT \* FROM InstructorDBA.Student\_Class\_Grade\_View;

STUDENT\_LNAME STUDENT\_FNAME COURSE\_NAME STUDENT\_GRADE INSTR\_LNAME INSTR\_FNAME SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF CLASS\_ROOM

Gilbert Sarah Intro to Computer Science A Roberts John Mon-Tues-Wed 0900-1100 Oct. 13th A

Gilbert Sarah Intro to Art B Russell Brandon Mon-Tues-Thur 0900-1100 Oct. 13th B

Gilbert Sarah Advanced English Ingle Tammy Tues-Wed 1300-1500 Oct. 17th F

Gilbert Sarah Intro to Aircraft mechanics Lopez Mike Mon-Wed 1000-1200 Oct. 14th C

SQL> ROLLBACK TO Student\_Class\_Signup\_Point;

Rollback complete.

SQL>

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> INSERT INTO InstructorDBA.Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES ( 3,6, 6, 'A');

Error starting at line : 627 in command -

INSERT INTO InstructorDBA.Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

VALUES ( 3,6, 6, 'A')

Error at Command Line : 627 Column : 27

Error report -

SQL Error: ORA-00942: table or view does not exist

00942. 00000 - "table or view does not exist"

\*Cause:

\*Action:

SQL>

SQL> /\*Test admin access\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SAVEPOINT Admin\_Student\_Class\_Signup\_Point;

Savepoint created.

SQL> INSERT INTO InstructorDBA.Student\_Class\_Signup (Student\_Num, Instr\_Num, Sched\_Num, Student\_Grade)

2 VALUES ( 3,6, 6, 'A');

1 row inserted.

SQL> SELECT \* FROM InstructorDBA.Student\_Class\_Signup;

STUDENT\_NUM INSTR\_NUM SCHED\_NUM STUDENT\_GRADE

1 1 1 A

1 2 2 B

2 3 3 A

3 3 3 C

4 4 4 C

5 5 5 B

6 6 6 D

7 7 7 F

8 8 8

9 9 9 A

10 10 10 A

3 6 6 A

12 rows selected.

SQL> ROLLBACK TO Admin\_Student\_Class\_Signup\_Point;

Rollback complete.

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL>

Class Tentative Schedule Policy. The following shows the SQL implementation procedure of the corresponding policy:

SQL> /\*Class Tentative Schedule Procedure - InstructorDBA and admins can view/create tentative schedules, instructors can view

SQL>tentative schedules, and students can view approved schedules.\*/

SQL>

SQL> connect sys/brr1wik7 as sysdba;

Connected.

SQL> show user;

USER is "SYS"

SQL> GRANT SELECT ON InstructorDBA.Student\_List TO lbacsys;

Grant succeeded.

SQL> GRANT SELECT ON InstructorDBA.Instr\_List TO lbacsys;

Grant succeeded.

SQL> GRANT SELECT ON InstructorDBA.Admin\_List TO lbacsys;

Grant succeeded.

SQL>

SQL> CONNECT lbacsys/brr1wik7;

Connected.

SQL> show user;

USER is "LBACSYS"

SQL>

SQL> BEGIN

2 SA\_SYSDBA.CREATE\_POLICY ('Sched\_OLS\_POL', 'ols\_col', 'READ\_CONTROL');

3

4 SA\_COMPONENTS.CREATE\_LEVEL ('Sched\_OLS\_POL',40,'HS','HIGHLY\_SENSITIVE');

5 SA\_COMPONENTS.CREATE\_LEVEL ('Sched\_OLS\_POL',30,'S','SENSITIVE');

6

7 SA\_LABEL\_ADMIN.CREATE\_LABEL ('Sched\_OLS\_POL',40,'HS');

8 SA\_LABEL\_ADMIN.CREATE\_LABEL ('Sched\_OLS\_POL',30,'S');

9

10 SA\_USER\_ADMIN.SET\_LEVELS ('Sched\_OLS\_POL','InstructorDBA', 'HS','S');

11

12 /\*Student account update classification level\*/

13 FOR x IN (SELECT User\_Name FROM InstructorDBA.Student\_List)

14 LOOP

15 SA\_USER\_ADMIN.SET\_LEVELS ('Sched\_OLS\_POL',''||x.User\_Name||'', 'S','S');

16 END LOOP;

17

18 /\*Instructor account update classification level\*/

19 FOR x IN (SELECT User\_Name FROM InstructorDBA.Instr\_List)

20 LOOP

21 SA\_USER\_ADMIN.SET\_LEVELS ('Sched\_OLS\_POL',''||x.User\_Name||'', 'HS','S');

22 END LOOP;

23

24 /\*Admin account update classification level\*/

25 FOR x IN (SELECT User\_Name FROM InstructorDBA.Admin\_List)

26 LOOP

27 SA\_USER\_ADMIN.SET\_LEVELS ('Sched\_OLS\_POL',''||x.User\_Name||'', 'HS','S');

28 END LOOP;

29

30 SA\_POLICY\_ADMIN.APPLY\_TABLE\_POLICY ('Sched\_OLS\_POL','InstructorDBA', 'Class\_Sched');

31 END;

32 /

PL/SQL procedure successfully completed.

SQL>

SQL> /\*Update classification column and test access\*/

SQL> connect sys/brr1wik7 as sysdba;

Connected.

SQL> show user;

USER is "SYS"

SQL> UPDATE InstructorDBA.Class\_Sched SET ols\_col = CHAR\_TO\_LABEL('Sched\_OLS\_POL','S');

12 rows updated.

SQL> UPDATE InstructorDBA.Class\_Sched SET ols\_col = CHAR\_TO\_LABEL('Sched\_OLS\_POL','HS') WHERE Sched\_Notes = 'Beta';

2 rows updated.

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

12 rows selected.

SQL> GRANT SELECT ON InstructorDBA.Class\_Sched TO student\_role;

Grant succeeded.

SQL>

SQL> /\*Test InstructorDBA access\*/

SQL> CONNECT InstructorDBA/brr1wik7;

Connected.

SQL> show user;

USER is "INSTRUCTORDBA"

SQL> SELECT \* FROM Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

12 rows selected.

SQL> SAVEPOINT InstructorDBA\_Student\_Class\_Sched\_Insert;

Savepoint created.

SQL> INSERT INTO Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes, ols\_col)

2 VALUES (12345,'All', '24 Hours','7 days a week', 'Alpha', CHAR\_TO\_LABEL('Sched\_OLS\_POL','HS'));

1 row inserted.

SQL> SELECT \* FROM Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

12345 All 24 Hours 7 days a week Alpha 40

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

13 rows selected.

SQL> ROLLBACK TO InstructorDBA\_Student\_Class\_Sched\_Insert;

Rollback complete.

SQL>

SQL>

SQL> /\*Test admin access\*/

SQL> CONNECT bevans1/TheSecPass0;

Connected.

SQL> show user;

USER is "BEVANS1"

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

12 rows selected.

SQL> SAVEPOINT Admin\_Student\_Class\_Sched\_Insert;

Savepoint created.

SQL> INSERT INTO InstructorDBA.Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes, ols\_col)

2 VALUES (12345,'All', '24 Hours','7 days a week', 'Alpha', CHAR\_TO\_LABEL('Sched\_OLS\_POL','HS'));

1 row inserted.

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

12345 All 24 Hours 7 days a week Alpha 40

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

13 rows selected.

SQL> ROLLBACK TO Admin\_Student\_Class\_Sched\_Insert;

Rollback complete.

SQL>

SQL>

SQL> /\*Test instructor access\*/

SQL> CONNECT mlopez3/TheSecPass2;

Connected.

SQL> show user;

USER is "MLOPEZ3"

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

12 rows selected.

SQL> SAVEPOINT Instructor\_Student\_Class\_Sched\_Insert;

Savepoint created.

SQL> INSERT INTO InstructorDBA.Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes, ols\_col)

2 VALUES (12345,'All', '24 Hours','7 days a week', 'Alpha', CHAR\_TO\_LABEL('Sched\_OLS\_POL','HS'));

Error starting at line : 721 in command -

INSERT INTO InstructorDBA.Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes, ols\_col)

VALUES (12345,'All', '24 Hours','7 days a week', 'Alpha', CHAR\_TO\_LABEL('Sched\_OLS\_POL','HS'))

Error at Command Line : 721 Column : 27

Error report -

SQL Error: ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without

the necessary privileges.

\*Action: Ask your database administrator or designated security

administrator to grant you the necessary privileges

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

11 Mon 0900-0930 6 days a week Beta 40

12 Tues 1100-1115 6 days a week Beta 40

12 rows selected.

SQL> ROLLBACK TO Instructor\_Student\_Class\_Sched\_Insert;

Rollback complete.

SQL>

SQL> /\*Test student access\*/

SQL> CONNECT sgilbert1/TheSecPass0;

Connected.

SQL> show user;

USER is "SGILBERT1"

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

10 rows selected.

SQL> SAVEPOINT Student\_Student\_Class\_Sched\_Insert;

Savepoint created.

SQL> INSERT INTO InstructorDBA.Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes, ols\_col)

2 VALUES (12345,'All', '24 Hours','7 days a week', 'Alpha', CHAR\_TO\_LABEL('Sched\_OLS\_POL','HS'));

Error starting at line : 731 in command -

INSERT INTO InstructorDBA.Class\_Sched (Sched\_Num, Sched\_Day, Sched\_Time, Sched\_Day\_Off, Sched\_Notes, ols\_col)

VALUES (12345,'All', '24 Hours','7 days a week', 'Alpha', CHAR\_TO\_LABEL('Sched\_OLS\_POL','HS'))

Error at Command Line : 731 Column : 27

Error report -

SQL Error: ORA-01031: insufficient privileges

01031. 00000 - "insufficient privileges"

\*Cause: An attempt was made to perform a database operation without

the necessary privileges.

\*Action: Ask your database administrator or designated security

administrator to grant you the necessary privileges

SQL> SELECT \* FROM InstructorDBA.Class\_Sched;

SCHED\_NUM SCHED\_DAY SCHED\_TIME SCHED\_DAY\_OFF SCHED\_NOTES OLS\_COL

1 Mon-Tues-Wed 0900-1100 Oct. 13th This is one of the main schedules 30

2 Mon-Tues-Thur 0900-1100 Oct. 13th This is one of the main schedules 30

3 Mon-Wed 1000-1200 Oct. 14th This is one of the main schedules 30

4 Mon-Wed-Fri 1000-1200 Oct. 15th This is an alternative schedule 30

5 Mon-Tues 0800-0900 Oct. 16th This is an alternative schedule 30

6 Tues-Wed 1300-1500 Oct. 17th This is an alternative schedule 30

7 Thur-Fri 1300-1500 Oct. 18th This is an alternative schedule 30

8 Wed-Fri 1600-1800 Oct. 19th This is an alternative schedule 30

9 Sat 0900-1100 Oct. 20th Weekend Schedule 30

10 Sun 0900-1100 Oct. 21st Weekend Schedule 30

10 rows selected.

SQL> ROLLBACK TO Student\_Student\_Class\_Sched\_Insert;

Rollback complete.

Connection created by CONNECT script command disconnected

# Conclusion

The importance of security cannot be understated enough in today’s highly connected, and often highly vulnerable, technology infrastructure. Counting lines of code is a risky and controversial metric for anyone to use (Harding, 2018), though I would at least point out this project took nearly double the number of lines of code to create and test security than it did to create the database itself. At the beginning of this project, we stated our overall problem, a production database that lacked any security measures. We then specified our security plan which would guide us through our security solution. We detailed the technical specifications of the project, created a Gantt chart detailing a timeline for the project to adhere to, and spelled out the conceptual, logical, and physical model to follow. We then reviewed the DDLs and DMLs that were used to implement the original database. Fifteen separate policies were created, using Views, VPD, and OLS, to ensure our database was secure and sensitive information would not be accessible by unauthorized personnel.

## Lesson Learned

This was a challenging project, and I gained a tremendous amount of knowledge through the creation and testing of the different types of security available in Oracle. For example, until this project I always used the “SYS” account as my default account for other projects. As I was creating other named accounts and implementing security policies, I began to realize why Oracle recommends creating a separate DBA account, vice using the “SYS” or “SYSTEM” accounts. I also found new ways of optimizing my SQL scripts, for example using for loops when creating accounts or assigning roles in mass. The biggest lesson learned, the level of scrutiny required in creating security policies and procedures to ensure no security gaps or leaks exists. On a personal note, I would say the key to a successful security implementation is proper planning, comprehensive documentation, and extensive testing.

## Final Thoughts

For decades, cybersecurity has been an afterthought. Now, it could arguably be the most important part of software and database development. We can no longer create products without a security focus baked right into the development process. Without security, it becomes a matter of “when” not “if” your data will be hacked, stolen, held for ransom, and/or destroyed. Every online multiuser database created today requires a complete security solution: a plan, policies, and procedures.

References

*1 Introduction to Oracle Label Security*. (2017). Retrieved from Oracle Help Center Administrator's Guide: https://docs.oracle.com/en/database/oracle/oracle-database/12.2/olsag/introduction-to-oracle-label-security.html#GUID-C20C62AE-2A30-45F9-AEEA-52A0D3286FBA

*7 Using Oracle Virtual Private Database to Control Data Access*. (2018). Retrieved from Oracle Help Center Database Security Guide: https://docs.oracle.com/cd/B28359\_01/network.111/b28531/vpd.htm#DBSEG80081

*7.3.1 SYS and SYSTEM Users*. (2017). Retrieved from Oracle Help Center Database 2 Day DBA: https://docs.oracle.com/database/121/ADMQS/GUID-CF1CD853-AF15-41EC-BC80-61918C73FDB5.htm#ADMQS12003

Grassi, P. A., Fenton, J. L., Newton, E. M., Perlner, R. A., Regenscheid, A. R., Burr, W. E., . . . Theofanos, M. F. (2017). *NIST Special Publication 800-63B.* Gaithersburg: National Institute of Standards and Technology. doi:https://doi.org/10.6028/NIST.SP.800-63b

Harding, B. (2018, March 22). *"Lines of code" is a garbage metric, so let's go dumpster diving*. Retrieved from Static Object: https://www.staticobject.com/lines\_of\_code\_garbage\_metric

Lesov, P. (2010). Database Security: A Historical Perspective. *arXiv preprint arXiv:1004.4022*.

Theriault, M., & Heney, W. (1998). *Oracle Security.* O'Reilly Media, Inc.

Wilson, J. M. (2003). Gantt charts: A centenary appreciation. *European Journal of Operational Research, 149*, 430-437.