**IS-4420 Database Fundamentals**

**Lab 4**

* SQL Data Definition Language (DDL)
* SQL Data Manipulation Language (DML)

20 points

**Summary**

Lab 4 consists of using your SQL lab environment to convert a physical ERD into a relational database and INSERTing hypothetical data into the tables.

The deliverable will be screenshots and SQL code. Copy the screenshots into a Word document, and then save it as a .pdf file for the lab submission. Submit the .sql files & the .pdf file to Canvas.

**Setup**

1. Connect to the SQL lab environment.
   1. If you have not yet connected to the SQL lab environment, follow the steps outlined in the **Connecting\_To\_IS4420\_RDS\_WebAccess.docx** file under the SQL Lab Setup module, or watch the equivalent video available at the same place.
   2. Launch SQL Server Management Studio by clicking the Windows charm and typing SSMS. Change the authentication method from Windows to SQL, and then type Swoop for the username and Swoop for the password.
   3. Please be sure to watch the **Lab 4 Introduction Video.mp4** provided with the lab to understand how to use SQL Server Management Studio (SSMS), if you have not yet done so.

**Note**

* You are encouraged to collaborate on homework assignments with your peers.
* Please refer to the PowerPoints & videos for SQL syntax.
* If you get stuck, attend tutoring office hours or send questions via email to the TAs.
* Assignments will take time, start them early.
* Class attendance will be rewarded by getting hints in class about the assignments.

**1: Initial Setup (4 points)**

1. Log into your SQL lab environment, launch SQL Server Management Studio (SSMS), and create a new database called, **Lab\_4<YourName>.**

For example, I would type: **CREATE DATABASE Lab\_4\_JacobCase;**

Highlight the command and press F5, or click on the Execute button with the green triangle.

**Take a screenshot showing the command you used to create the database and the database on the left-side menu.**

Graphical user interface, application

Description automatically generated

1. Change the database context to the empty **Lab\_4<YourName>** database you just created by typing:

For example, I would type: **USE Lab\_4\_ JacobCase**;

**Take a screenshot showing the command you used to create the database and the database on the left-side menu.**

**2: Data Definition Language (DDL) (6 points)**

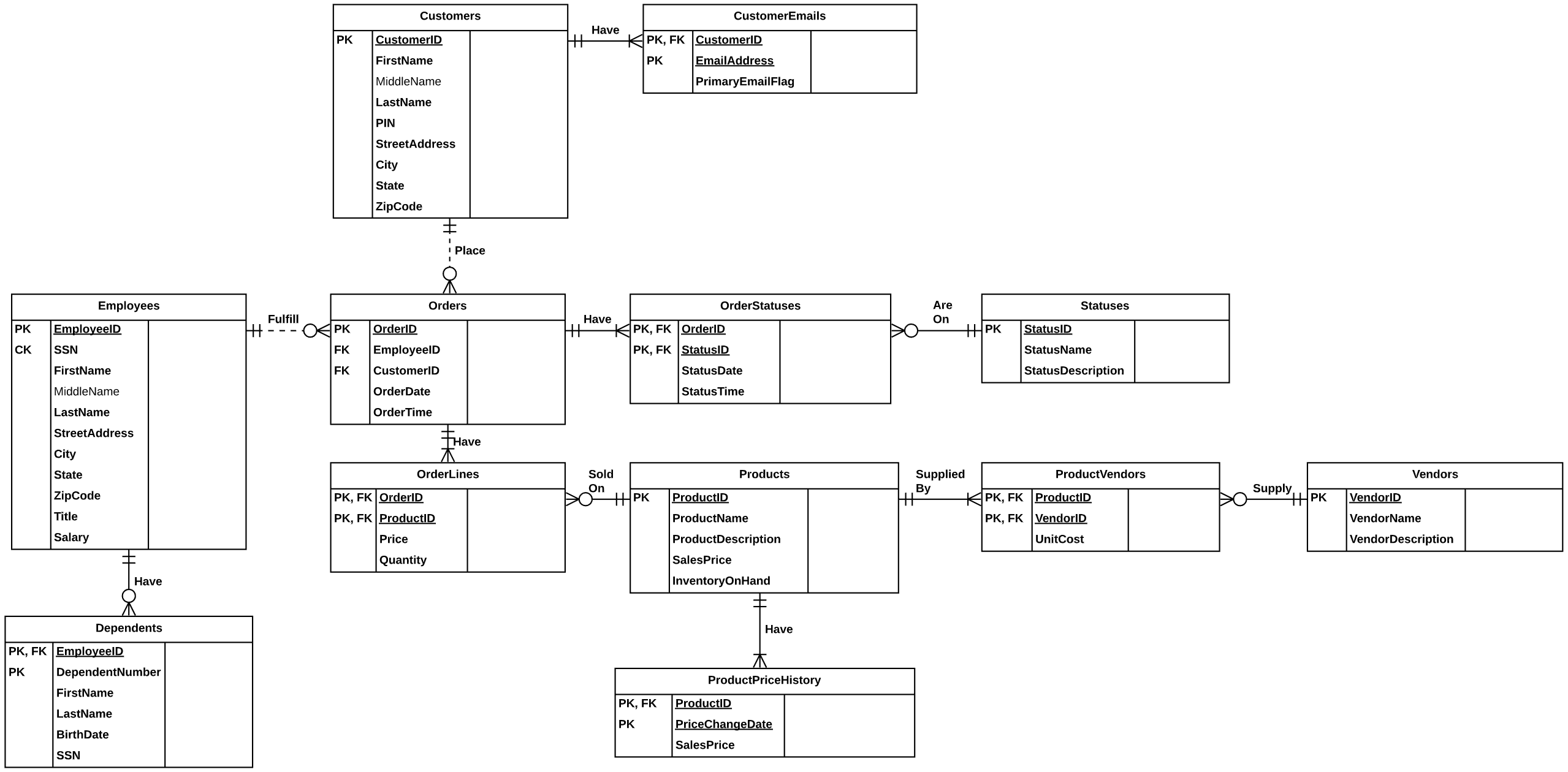
During the database development process, when the Entity-Relationship Diagram (ERD) is finished, you enter a phase of work that is very procedural.  Converting the entities in your ERD to database tables requires very little creativity and is mostly a process of following a series of steps.

**The steps include:**

1. Determining data types for each attribute (column).
   1. The goal when choosing data types is to choose one that won’t use too much space while being large enough to store the values it requires.
2. Referring to the ERD to write the CREATE TABLE commands.
3. Running the CREATE TABLE commands in the correct order.
   1. Tables with Referential Integrity must find the parent table, or the CREATE TABLE command will fail.
4. Adding any additional constraints to the tables that were not specified during the creation of the table.

**Deliverables for Part 2:**

Refer to this ERD for this section



1. Connect to LucidChart, open a new blank document, and pin the ERD library.
2. Re-create the ERD shown in the image above. Be sure to use the 3-column entities.
3. In the column on the right of each entity, add data types. Be sure to review the PowerPoints & videos to understand how to choose data types.
   1. Please note that choosing data types is not a science. There are many judgement calls. Use your best discretion.
4. **Take a screenshot of the ERD with the data types added. Make sure they are visible for grading.**

**Table

Description automatically generated**

1. Write CREATE TABLE commands for each of the tables.  They can all be part of the same file.  Don’t forget the semicolon at the end of each command.  Please review the class PowerPoints & videos for how to write CREATE TABLE commands.
   1. To help you get started, I have included a SQL script called, **Lab\_4\_CREATE\_First\_5\_Tables.sql**. This script creates the first 5 tables and can be referred to for the syntax you will need to CREATE the rest of the tables.
   2. Run the commands to create each table in the correct order.
2. ALTER the Customers table to have a new CHAR(1) column called EmailOptIn with a default value of  ‘Y’. Please review the class PowerPoints & videos for how to write ALTER TABLE commands.
3. **Save all CREATE TABLE & ALTER TABLE commands in a .sql file called Lab\_4\_Tables\_<StudentName>.sql**
   1. **Example: my file would be Lab\_4\_Tables\_JacobCase.sql**

**3: Data Manipulation Language (DML) (6 points)**

Now that the tables have been created, we will practice INSERTing hypothetical data into them.  For each of the tables that you created, build INSERT commands to populate the tables with ***at least 3 rows of data*** that you come up with on your own.

**Please keep the following in mind:**

* By their nature, Surrogate Keys are handled by the database. You don’t specify values for them.  Therefore, you leave them out of the INSERT command.
* Foreign Keys must find a matching row in the Primary Key column of the parent table they refer to.  When INSERTing into tables with Foreign Keys, it’s necessary to look at the parent table to get relevant values.
* While you are testing your INSERT commands, you may want to start over with a fresh database. Simply DROP the database you’ve been building, CREATE it again, and rerun your CREATE TABLE commands.

**Deliverables for Part 3:**

* **Save all your INSERT commands in a .sql file called Lab\_4\_Inserts\_<StudentName>.sql**
  + **Example, my file would be Lab\_4\_Inserts\_JacobCase.sql**
* **Please make sure the INSERT commands INSERT data into the tables in the correct order to ensure that Foreign Keys find the values they need in the Primary Key columns.**

**5: Review (4 points)**

**Answer the following questions:**

1. Describe why data types are necessary and what is important to keep in mind when determining the data types for each column in your tables.

**Data types are necessary because it helps us get the right type of values. An employee ID can not be averaged so there is no point of using a function that allows that. It is also to help with data space because if you let the database store massive amounts of data about one column it will get really expensive.**

1. Describe why it is so important to create each table in the correct order, and why it is equally important to insert the rows in the correct order

You need to create the table in order because some of the parent tables literally give their child tables values. So when it comes to SQL if you code the child table first it wont pull the values we are looking for.

1. What is the difference between DDL vs DML?

**DDL is the framework of the code and DML is the specifics of what you want it to do.**

1. In class, we learned about the ANSI-SQL Standard that all major relational databases adhere to. Describe the benefits & reasons you would want to write SQL code that is compliant with the ANSI-SQL standard.

**It gives you a standardized way of using SQL so you can use mostly the same language over many different SQL platforms.**

**Submission**

* A .pdf file containing all the required screenshots called **Lab\_4\_Screenshots\_StudentName.pdf**
* A .sql file with the CREATE TABLE commands called **Lab\_4\_Tables\_StudentName.sql**
* A .sql file with the INSERT commands called **Lab\_4\_Inserts\_StudentName.sql**