

Data Admin Concepts & Database Management

Table of Contents

Data Admin Concepts & Database Management	1
Lab 04 – Normalization	1
Overview	1
Learning Objectives.....	2
Lab Goals.....	2
What You Will Need to Begin.....	2
Part 1 – Normalizing an Existing Data Set.....	3
Setup	3
Case Study 1 – Rad Chad's Rad Bikes	3
Case Study 2 – More Bikes	11
Case Study 3 – Books Again	12
Part 2 – Normalizing Video Data for VidCast	21
Setup	21
To-do	21

Lab 04 – Normalization

Overview

This lab is the fourth of ten labs in which we will build a database using the systematic approach covered in the asynchronous material. Each successive lab will build upon the one before and can be a useful guide for building your own database projects.

In this lab, we will evaluate data sets and decompose them into at least third normal form (3NF). Part 1 of this lab explores a sample data set taken from a manufacturing system and ask you to decompose the data into normalized relations (tables). As a preview of the technical aspects of databases, you'll also implement a small portion of a database in Microsoft Access. Part 2 returns to our VidCast database and asks you to evaluate data collected from our stakeholders for normalization.

For this lab, we will continue our work in Draw.io. There are other software tools that can accomplish this task (Visio, Vertabelo, and Lucidchart to name a few). If you would prefer to use these other tools, consult with your course section lead. This lab also introduces a relational notation that can also be used in lieu of diagramming. We will also use Microsoft Access for a portion of the lab.

Read this lab document once through before beginning.

Learning Objectives

In this lab you will

- Demonstrate understanding normalization
- Demonstrate ability to normalize relations into at least third normal form (3NF)
- Identify candidate keys and functional dependencies and remove them from relations

Lab Goals

This lab consists of two sections. The first section presents a data set retrieved from a manufacturing resource planning system and walks through the process of normalizing the data. The first section also implements a small database in Microsoft. In the second section, you will apply your ability to normalize relations on our ongoing database project, VidCast.

What You Will Need to Begin

- This document
- An active Internet connection
- Visit draw.io (<https://www.draw.io/>) and ensure you can create a blank diagram. Ideally, connect draw.io to your Google Account, OneDrive, or local disk. If you are new to using online software tools, the recommendation is to connect draw.io to your Google Drive. If you do not have a drive, visit drive.google.com (<https://drive.google.com>) to create one for free before beginning the second part of this lab.
- A blank Word (or similar) document into which you can place your answers. Please include your name, the current date, and the lab number on this document. Please also number your responses, indicating which part and question of the lab to which the answer pertains. Word docx format is preferred. If using another word processing application, please convert the document to pdf before submitting your work to ensure your instructor can open the file.
- To have completed Lab 03 – Logical Modeling
- Understanding of the normal forms through 3NF and the dependencies and anomalies these forms attempt to resolve
- An installation of Microsoft Access. This lab uses the latest version of Access 365, but the tools presented are available in most prior versions. If you don't already have a copy of Access, you can obtain a license in the following ways:
 - Install Office 365 using your license available from your SEmail portal account. For more information, visit this link:
<https://answers.syr.edu/display/software/Office+365+ProPlus+for+Staff+and+Faculty+through+Microsoft+Student+Advantage>
 - Download and install the latest version of Access from the Microsoft Imagine service accessible at <https://my.ischool.syr.edu/>
 - If you would prefer to not install Access, or are not running a compatible operating system, you can use the Access installation provided on the iSchool Remote Lab. For

help connecting to the Remote Lab, visit this page (netid login required):

<https://answers.syr.edu/pages/viewpage.action?pageId=9666607>

Part 1 – Normalizing an Existing Data Set

Setup

The following exercises will cover the process for normalizing data sets through third normal form (3NF). As is often the case, we have been given a set of data and have been asked to build a database from it.

Normalization usually happens in parallel with logical modeling. In fact, several of the steps we take to convert from conceptual to logical ERD mitigate dependencies and anomalies in our final data tables. For instance, the act of moving multivalued attributes to their own tables places the new table in at least first normal form (1NF).

Also, once someone is familiar with the process of normalizing, it is not uncommon to integrate this process into the act of producing a logical model diagram, merging these two processes in an efficient design methodology.

It is also not uncommon to prototype a database in a desktop database application such as Microsoft Access. This can be helpful in identifying problem spots in our database before we commit the tables to the server-side database in SQL Server (or similar).

Case Study 1 – Rad Chad's Rad Bikes

Note: This is a walkthrough case study. There is no deliverable for case study 1.

Rad Chad's Rad Bikes, a bicycle manufacturer in Syracuse, NY, has asked us to migrate their customer order data to a new system of our design. They have provided us with a spreadsheet of the order data they currently have. This spreadsheet gives us the customer data, the order data, and the order detail data in one "convenient" spot. Take a few minutes to familiarize yourself with the data in Figure 1:

Cust Num	Name	Postal Code	State	Order Num	OrderedDate	Line	Item Number	Description	Qty Ordered	Price Each
9	Carolina Frames, Inc.	29401	SC	106	7/25/2017	1	FA-10000	Bicycle Model 30,26"	100	\$ 280.00
9	Carolina Frames, Inc.	29401	SC	106	7/25/2017	2	TA-50000	Tire 26" Tubeless	100	\$ 15.00
9	Carolina Frames, Inc.	29401	SC	106	7/25/2017	3	TA-60000	Hub, Wheel 32" Hole	200	\$ 4.00
9	Carolina Frames, Inc.	29401	SC	108	8/17/2017	1	AL-10000	Steel,Chromium	10	\$ 3.85
9	Carolina Frames, Inc.	29401	SC	108	8/17/2017	2	CG-10000	Grips,Cushioned	200	\$ 2.50
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	1	FA-20000	Bicycle,Model-50,26"	200	\$ 320.00
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	2	CP-15000	Seat,Deluxe	1,500	\$ 8.00
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	3	TA-60000	Hub, Wheel 32" Hole	400	\$ 8.00
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	4	TA-50000	Tire,26",Tubeless	1,000	\$ 30.00
12	Lake House Cycles	13903	NY	110	8/9/2017	1	FA-10000	Bicycle,Model-30,26"	250	\$ 280.00
12	Lake House Cycles	13903	NY	110	8/9/2017	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
12	Lake House Cycles	13903	NY	110	8/9/2017	3	FA-30000	Bicycle,Model-100,700mm,Eurocycle	150	\$ 450.00
12	Lake House Cycles	13903	NY	110	8/9/2017	4	CP-15000	Seat,Deluxe	100	\$ 4.00
14	Hornes Department Store	15122	PA	116	8/3/2017	1	FA-10000	Bicycle,Model-30,26"	100	\$ 290.00
14	Hornes Department Store	15122	PA	116	8/3/2017	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
14	Hornes Department Store	15122	PA	116	8/3/2017	3	CP-15000	Seat,Deluxe	50	\$ 5.00
14	Hornes Department Store	15122	PA	116	8/3/2017	4	TA-50000	Tire,26",Tubeless	25	\$ 20.00
14	Hornes Department Store	15122	PA	116	8/3/2017	5	CP-10000	Seat,Padded	50	\$ 2.30
16	Stevenson Sporting Goods	92805-4778	CA	134	9/25/2017	1	FA-30000	Bicycle,Model-100,700mm,Eurocycle	20	\$ 450.00
16	Stevenson Sporting Goods	92805-4778	CA	134	9/25/2017	2	FA-20000	Bicycle,Model-50,26"	15	\$ 350.00

Figure 1- Rad Chad's Rad Bikes Orders

It is entirely possible for us to just create one large database table and put all of this in, but this opens us up to several “anomalies”, so we want to normalize this into distinct tables. We will use the systematic approach of normalization to decompose this sheet into some number of tables (often called *relations*) to reduce duplication of data and mitigate data inconsistencies.

Step 1 – Identify the current normal form

With the data in front of us, we can use our rules of normal forms to identify which normal form we’re currently in.

- First Normal Form (1NF)
 - There are no multivalued attributes
- Second Normal Form (2NF)
 - Is in 1NF -and-
 - There are no partial functional dependencies (PFDs)
- Third Normal Form (3NF)
 - Is in 2NF -and-
 - There are no transitive functional dependencies (TFDs)

There are normal forms beyond 3NF, but we can safely stop at 3NF and have a good design.

Assessing for 1NF

To see if we’re in 1NF, we need to look for *multivalued attributes*. This means that for each line / row in the source data, there are more than one value for each distinct attribute. We usually see this when many values are packed into one cell (as in a spreadsheet) or if the data creator has allocated more than one column to a specific attribute.

In Figure 1, the data in each cell appears to be for one value and one value only. Each column in the dataset is distinct from all the others, so we assert that this data is in 1NF.

Assessing for 2NF

We must remove any partial functional dependencies (PFD).

A partial functional dependency, or PFD, is “a functional dependency in which one or more nonkey attributes are functionally dependent on part, but not all, of the primary key.” (Hoffer, p185)

We’re missing at least one part of that statement. We must identify the primary key. Because we are not yet ready to commit to any data in this sheet as a primary key, we can call it a candidate key.

Candidate Key: An attribute, or combination of attributes, that uniquely identifies a row in a relation (Hoffer, p181)

One way to spot the candidate key is to think on this question: What is the minimum amount of data someone would have to give you for you to precisely know to which row they're referring? For example, given Figure 1's data, if you were having a telephone conversation and the person on the other end asked, "What is the name of the customer who bought item number 'FA-10000'?" you would not be able to answer definitively since more than 1 customer ordered that item, as shown in Figure 2:

Cust Nu	Name	Postal Code	State	Order Nu	OrderedDa	Li	Item Number	Description	Qty Order	Price Ea
9	Carolina Frames, Inc.	29401	SC	106	7/25/2017	1	FA-10000	Bicycle Model 30,26"	100	\$ 280.00
12	Lake House Cycles	13903	NY	110	8/9/2017	1	FA-10000	Bicycle,Model-30,26"	250	\$ 280.00
14	Hornes Department Store	15122	PA	116	8/3/2017	1	FA-10000	Bicycle,Model-30,26"	100	\$ 290.00

Figure 2- Only FA-10000

You would need more information from them to get to a unique response. However, if they asked, "What item was sold on order number 110, line 3?", we would be able to answer that question definitively.

Cust Nu	Name	Postal Code	State	Order Nu	OrderedDa	Li	Item Number	Description	Qty Order	Price Ea
12	Lake House Cycles	13903	NY	110	8/9/2017	3	FA-30000	Bicycle,Model-100,700mm,Eurocycle	150	\$ 450.00

Figure 3- Only Order 110, Line 3

Notice that when the sheet is filtered on those two columns in the sheet, only 1 row remains. This tells us that the combination of Order Number and Line uniquely identifies any row in our data. This is a good candidate key.

Here is figure 1 again with the candidate key highlighted:

Cust Num	Name	Postal Code	State	Order Num	OrderedDate	Line	Item Number	Description	Qty Ordered	Price Each
9	Carolina Frames, Inc.	29401	SC	106	7/25/2017	1	FA-10000	Bicycle Model 30,26"	100	\$ 280.00
9	Carolina Frames, Inc.	29401	SC	106	7/25/2017	2	TA-50000	Tire 26" Tubeless	100	\$ 15.00
9	Carolina Frames, Inc.	29401	SC	106	7/25/2017	3	TA-60000	Hub, Wheel 32" Hole	200	\$ 4.00
9	Carolina Frames, Inc.	29401	SC	108	8/17/2017	1	AL-10000	Steel,Chromium	10	\$ 3.85
9	Carolina Frames, Inc.	29401	SC	108	8/17/2017	2	CG-10000	Grips,Cushioned	200	\$ 2.50
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	1	FA-20000	Bicycle,Model-50,26"	200	\$ 320.00
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	2	CP-15000	Seat,Deluxe	1,500	\$ 8.00
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	3	TA-60000	Hub, Wheel 32" Hole	400	\$ 8.00
8	Ian's Bicycle Products	30338	GA	109	8/5/2017	4	TA-50000	Tire,26",Tubeless	1,000	\$ 30.00
12	Lake House Cycles	13903	NY	110	8/9/2017	1	FA-10000	Bicycle,Model-30,26"	250	\$ 280.00
12	Lake House Cycles	13903	NY	110	8/9/2017	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
12	Lake House Cycles	13903	NY	110	8/9/2017	3	FA-30000	Bicycle,Model-100,700mm,Eurocycle	150	\$ 450.00
12	Lake House Cycles	13903	NY	110	8/9/2017	4	CP-15000	Seat,Deluxe	100	\$ 4.00
14	Hornes Department Store	15122	PA	116	8/3/2017	1	FA-10000	Bicycle,Model-30,26"	100	\$ 290.00
14	Hornes Department Store	15122	PA	116	8/3/2017	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
14	Hornes Department Store	15122	PA	116	8/3/2017	3	CP-15000	Seat,Deluxe	50	\$ 5.00
14	Hornes Department Store	15122	PA	116	8/3/2017	4	TA-50000	Tire,26",Tubeless	25	\$ 20.00
14	Hornes Department Store	15122	PA	116	8/3/2017	5	CP-10000	Seat,Padded	50	\$ 2.30
16	Stevenson Sporting Goods	92805-4778	CA	134	9/25/2017	1	FA-30000	Bicycle,Model-100,700mm,Eurocycle	20	\$ 450.00
16	Stevenson Sporting Goods	92805-4778	CA	134	9/25/2017	2	FA-20000	Bicycle,Model-50,26"	15	\$ 350.00

Figure 4 - Candidate Key Highlighted

Note: There are probably other possibilities. For instance, (Order Num, Item Number) would probably yield similar results, but we'll go with (Order Num, Line) for now.

We can write this relation as (note the underline on the candidate key columns):

OrderData (Cust Num, Name, Postal Code, State, Order Num, Ordered Date, Line, Item Number, Description, Qty Ordered, Price Each)

If you prefer to see these in diagram form, this would be the equivalent ERD:

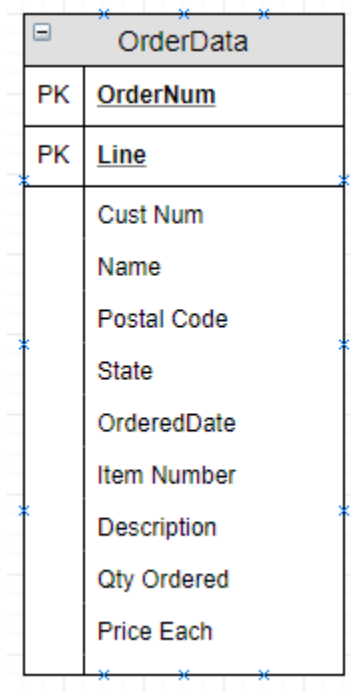


Figure 5- Order Data as an ERD

For the remainder of this lab, we will use the text based relational notation. Primary Keys will be underlined, and foreign keys will be *italicized*. Attributes that are both primary and foreign keys will be *italicized and underlined*.

Having identified the key for this relation / spreadsheet / table, we can look for partial functional dependencies. We need to look at the nonkey attributes (columns with no highlighting) and decide if they are determined by only one of our two key columns. Said another way, if we look at one of our two key columns, can we precisely determine the value for a nonkey column?

A look at Cust Num, for instance, shows us that for any given Order Num, we know its Cust Num. As shown in the following figure, the Cust Num for every Order Num of 116 is “Horne’s Department Store”. We do not need to know the Line number to make that determination (literally no pun intended, that’s why Order Num is called a Determinant!)

Cust Nu	Name	Postal Code	State	Order Nu	OrderedDa	Li	Item Numbe	Description	Qty Orderi	Price Ea
14	Hornes Department Store	15122	PA	116	8/3/2017	1	FA-10000	Bicycle,Model-30,26"	100	\$ 290.00
14	Hornes Department Store	15122	PA	116	8/3/2017	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
14	Hornes Department Store	15122	PA	116	8/3/2017	3	CP-15000	Seat,Deluxe	50	\$ 5.00
14	Hornes Department Store	15122	PA	116	8/3/2017	4	TA-50000	Tire,26",Tubeless	25	\$ 20.00
14	Hornes Department Store	15122	PA	116	8/3/2017	5	CP-10000	Seat,Padded	50	\$ 2.30

Figure 6 - Order 116, Hornes Department Store

Now that we have found a PFD, we need to remove it by creating a new relation (aka table, entity). The determinant becomes the primary key of the new relation and the dependency becomes a nonkey attribute of that new relation:

Order (Order Num, Name)

The dependency is removed completely from the original relation, and the determinant becomes a foreign key:

OrderData (Cust Num, Postal Code, State, Order Num, Ordered Date, Line, Item Number, Description, Qty Ordered, Price Each)

To help visualize the effect of this, the following data sheets have been modified to reflect the new relations:

Cust Num	Postal Code	State	Order Num	OrderedDate	Line	Item Number	Description	Qty Ordered	Price Each
9 29401	SC		106	7/25/2017	1	FA-10000	Bicycle Model 30,26"	100	\$ 280.00
9 29401	SC		106	7/25/2017	2	TA-50000	Tire 26" Tubeless	100	\$ 15.00
9 29401	SC		106	7/25/2017	3	TA-60000	Hub, Wheel 32" Hole	200	\$ 4.00
9 29401	SC		108	8/17/2017	1	AL-10000	Steel,Chromium	10	\$ 3.85
9 29401	SC		108	8/17/2017	2	CG-10000	Grips,Cushioned	200	\$ 2.50
8 30338	GA		109	8/5/2017	1	FA-20000	Bicycle,Model-50,26"	200	\$ 320.00
8 30338	GA		109	8/5/2017	2	CP-15000	Seat,Deluxe	1,500	\$ 8.00
8 30338	GA		109	8/5/2017	3	TA-60000	Hub, Wheel 32" Hole	400	\$ 8.00
8 30338	GA		109	8/5/2017	4	TA-50000	Tire,26",Tubeless	1,000	\$ 30.00
12 13903	NY		110	8/9/2017	1	FA-10000	Bicycle,Model-30,26"	250	\$ 280.00
12 13903	NY		110	8/9/2017	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
12 13903	NY		110	8/9/2017	3	FA-30000	Bicycle,Model-100,700mm,Eurocycle	150	\$ 450.00
12 13903	NY		110	8/9/2017	4	CP-15000	Seat,Deluxe	100	\$ 4.00
14 15122	PA		116	8/3/2017	1	FA-10000	Bicycle,Model-30,26"	100	\$ 290.00
14 15122	PA		116	8/3/2017	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
14 15122	PA		116	8/3/2017	3	CP-15000	Seat,Deluxe	50	\$ 5.00
14 15122	PA		116	8/3/2017	4	TA-50000	Tire,26",Tubeless	25	\$ 20.00
14 15122	PA		116	8/3/2017	5	CP-10000	Seat,Padded	50	\$ 2.30
16 92805-4778	CA		134	9/25/2017	1	FA-30000	Bicycle,Model-100,700mm,Eurocycle	20	\$ 450.00
16 92805-4778	CA		134	9/25/2017	2	FA-20000	Bicycle,Model-50,26"	15	\$ 350.00

Figure 7- Order Data with Name dependency removed

Order Num	Name
106	Carolina Frames, Inc.
108	Carolina Frames, Inc.
109	Ian's Bicycle Products
110	Lake House Cycles
116	Hornes Department Store
134	Steveson Sporting Goods

Figure 8 - Order relation

Now we repeat this process for each other nonkey attribute. Cust Num, Postal Code, State, and Ordered Date all are also determined by Order Num. Since we already have a relation in which the determinant is the primary key, rather than create a new relation for them, we can just add these dependent attributes to the Order relation.

Order (Order Num, Cust Num, Name, Postal Code, State, Ordered Date)

Remember to remove the dependent attributes from the OrderData relation:

OrderData (Order Num, Line, Item Number, Description, Qty Ordered, Price Each)

In data, it would look like this:

Order Num	Line	Item Number	Description	Qty Ordered	Price Each
106	1	FA-10000	Bicycle Model 30,26"	100	\$ 280.00
106	2	TA-50000	Tire 26" Tubeless	100	\$ 15.00
106	3	TA-60000	Hub, Wheel 32" Hole	200	\$ 4.00
108	1	AL-10000	Steel,Chromium	10	\$ 3.85
108	2	CG-10000	Grips,Cushioned	200	\$ 2.50
109	1	FA-20000	Bicycle,Model-50,26"	200	\$ 320.00
109	2	CP-15000	Seat,Deluxe	1,500	\$ 8.00
109	3	TA-60000	Hub, Wheel 32" Hole	400	\$ 8.00
109	4	TA-50000	Tire,26",Tubeless	1,000	\$ 30.00
110	1	FA-10000	Bicycle,Model-30,26"	250	\$ 280.00
110	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
110	3	FA-30000	Bicycle,Model-100,700mm,Eurocycle	150	\$ 450.00
110	4	CP-15000	Seat,Deluxe	100	\$ 4.00
116	1	FA-10000	Bicycle,Model-30,26"	100	\$ 290.00
116	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
116	3	CP-15000	Seat,Deluxe	50	\$ 5.00
116	4	TA-50000	Tire,26",Tubeless	25	\$ 20.00
116	5	CP-10000	Seat,Padded	50	\$ 2.30
134	1	FA-30000	Bicycle,Model-100,700mm,Eurocycle	20	\$ 450.00
134	2	FA-20000	Bicycle,Model-50,26"	15	\$ 350.00

Figure 9 - Order data with more dependencies removed

Order Num	Cust Num	Name	Postal Code	State	OrderedDate
106	9	Carolina Frames, Inc.	29401	SC	7/25/2017
108	9	Carolina Frames, Inc.	29401	SC	8/17/2017
109	8	Ian's Bicycle Products	30338	GA	8/5/2017
110	12	Lake House Cycles	13903	NY	8/9/2017
116	14	Hornes Department Store	15122	PA	8/3/2017
134	16	Stevenson Sporting Goods	92805-4778	CA	9/25/2017

Figure 10 - Order relation with new attributes

Now, if we need to know the customer's name for any Order Num, we can look up that value in the order relation without worrying about having duplicate Order Nums to worry about.

For the rest of the nonkey attributes in our original data, we need to know all parts of the key to know what the value is. For example, Order Num 110 has four different Item Number, Description, Qty Ordered, and Price Each values, so we also need to know the Line number.

For this reason, we know we have removed all PFDs and our relation is in 2NF. Here are the final 2NF relations:

Order (Order Num, Cust Num, Name, Postal Code, State, Ordered Date)

OrderData (Order Num, Line, Item Number, Description, Qty Ordered, Price Each)

Assessing for 3NF

At this point, it's worth noting that each relation needs to be normalized. This means we must look at each relation and assess for the normal forms independently. It is entirely possible to have a relation in 1NF and another different relation in 2NF or 3NF.

Revisiting our relations from before:

Order (Order Num, Cust Num, Name, Postal Code, State, Ordered Date)

OrderData (Order Num, Line, Item Number, Description, Qty Ordered, Price Each)

We will need to assess both for normalization steps. For the remainder of this case study, we will focus on the Order relation and will leave the 3NF for the OrderData relation to a future case study.

Rad Chad sent us some more order data to work with, so we can work from the following list (notice we will have the candidate key highlighted):

Order Num	Cust Num	Name	Postal Code	State	OrderedDate
106	9	Carolina Frames, Inc.	29401	SC	7/25/2017
107	1	Coordinated Bicycles	46514	IN	7/31/2017
108	9	Carolina Frames, Inc.	29401	SC	8/17/2017
109	8	Ian's Bicycle Products	30338	GA	8/5/2017
110	12	Lake House Cycles	13903	NY	8/9/2017
111	1	Coordinated Bicycles	46514	IN	8/15/2017
112	1	Coordinated Bicycles	46514	IN	8/3/2017
113	2	Price Brothers Dept Store	44857	OH	7/28/2017
114	2	Price Brothers Dept Store	44857	OH	8/5/2017
115	13	Brand Central Dept Store	06902	CT	7/29/2017

Figure 11 - More Order Data

Because we built this relation using the steps from the 2NF process above, we can assume two things:

- The candidate key is Order Num because that was the determinant in the dependencies found above
- Because there is only one key attribute in our candidate key, it is impossible for there to be a PFD, meaning we are in 2NF already.

With that information in our corner, we can assess this relation for 3NF by looking for any transitive functional dependencies.

A transitive functional dependency exists when a nonkey attribute is dependent on, or determined by, another nonkey attribute.

Have a look at Cust Num above. If we sort the data by Cust Num, we can see some patterns emerge. Note how Name remains the same when Cust Num remains the same, and then Name changes when Cust Num does?

Order Num	Cust Num	Name
107	1	Coordinated Bicycles
111	1	Coordinated Bicycles
112	1	Coordinated Bicycles
113	2	Price Brothers Dept Store
114	2	Price Brothers Dept Store
109	8	Ian's Bicycle Products
106	9	Carolina Frames, Inc.
108	9	Carolina Frames, Inc.
110	12	Lake House Cycles
115	13	Brand Central Dept Store

Figure 12- Visualizing TFD

Cust Num is not part of the key, but it is clear from this that Cust Num determines Name. Think of it this way: If the person on our phone call asked, “What is the name of customer number 2?” there is no ambiguity in saying “Price Brothers Dept Store”, even though there are multiple rows. This is not entirely bad, per se, but having two rows for one data point opens us up to errors in our data.

In our case, we can say that Cust Num determines Name. Said the other way, Name is dependent on Cust Num.

To be complete, the precise verbiage is “Name is functionally dependent on Order Num via Cust Num”, but that’s just overkill.

To remove this dependency, we take a similar approach as removing a PFD: Create a new relation with the determinant as primary key and move the dependency to that new relation. The determinant in the original relation becomes a foreign key.

Order (Order Num, Cust Num, Postal Code, State, Ordered Date)

Customer (Cust Num, Name)

Here’s our data after that change:

Order Num	Cust Num	Postal Code	State	OrderedDate
107	1	46514	IN	7/31/2017
111	1	46514	IN	8/15/2017
112	1	46514	IN	8/3/2017
113	2	44857	OH	7/28/2017
114	2	44857	OH	8/5/2017
109	8	30338	GA	8/5/2017
106	9	29401	SC	7/25/2017
108	9	29401	SC	8/17/2017
110	12	13903	NY	8/9/2017
115	13	06902	CT	7/29/2017

Figure 13- Order relation after eliminating Name TFD

Cust Num	Name
1	Coordinated Bicycles
2	Price Brothers Dept Store
8	Ian's Bicycle Products
9	Carolina Frames, Inc.
12	Lake House Cycles
13	Brand Central Dept Store

Figure 14 - New Customer relation

Looking at the remaining nonkey attributes in the Order relation, we can see that a similar dependency exists, whereby Cust Num determines Postal Code and State, so we move them to the Customer relation.

Order (Order Num, Cust Num, Ordered Date)

Customer (Cust Num, Name, Postal Code, State)

Ordered Date needs to stay put because that is fully dependent on the key, Order Num, and not the Cust Num.

Our relations now look like this:

Order (Order Num, Cust Num, Ordered Date)

Customer (Cust Num, Name, Postal Code, State)

OrderData (Order Num, Line, Item Number, Description, Qty Ordered, Price Each)

Case Study 2 – More Bikes

Your Turn: Using the steps to normalize a relation, normalize the OrderData relation to 3NF. You can either diagram the final relations or use the relational notation described in this document. You do not

need to recreate the spreadsheets for this. You need only provide the final normalized relations in your answer document.

Here is the data again:

Order Num	Line	Item Number	Description	Qty Ordered	Price Each
106	1	FA-10000	Bicycle Model 30,26"	100	\$ 280.00
106	2	TA-50000	Tire 26" Tubeless	100	\$ 15.00
106	3	TA-60000	Hub, Wheel 32" Hole	200	\$ 4.00
108	1	AL-10000	Steel,Chromium	10	\$ 3.85
108	2	CG-10000	Grips,Cushioned	200	\$ 2.50
109	1	FA-20000	Bicycle,Model-50,26"	200	\$ 320.00
109	2	CP-15000	Seat,Deluxe	1,500	\$ 8.00
109	3	TA-60000	Hub, Wheel 32" Hole	400	\$ 8.00
109	4	TA-50000	Tire,26",Tubeless	1,000	\$ 30.00
110	1	FA-10000	Bicycle,Model-30,26"	250	\$ 280.00
110	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
110	3	FA-30000	Bicycle,Model-100,700mm,Eurocycle	150	\$ 450.00
110	4	CP-15000	Seat,Deluxe	100	\$ 4.00
116	1	FA-10000	Bicycle,Model-30,26"	100	\$ 290.00
116	2	FA-20000	Bicycle,Model-50,26"	50	\$ 320.00
116	3	CP-15000	Seat,Deluxe	50	\$ 5.00
116	4	TA-50000	Tire,26",Tubeless	25	\$ 20.00
116	5	CP-10000	Seat,Padded	50	\$ 2.30
134	1	FA-30000	Bicycle,Model-100,700mm,Eurocycle	20	\$ 450.00
134	2	FA-20000	Bicycle,Model-50,26"	15	\$ 350.00

Figure 15 - The Order Data needs 3NFifying. 3NFifying? Is that a word?

What to Submit

Starting with

OrderItem(Order Num, Line, Item Number, Description, Qty Ordered, Price Each)

list the relations, normalized to 3NF, on your answer document. Feel free to diagram with Draw.io (or similar) or use the relational notation introduced above.

Case Study 3 – Books Again

Setup

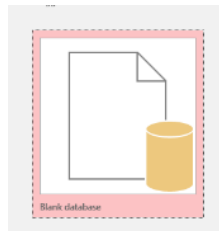
Microsoft Access is a desktop database management system that is commonly used for personal databases and databases maintained by small workgroups. While it's not the best tool for building databases at scale, we can use it to demonstrate the benefits of well-normalized relational databases. We can also use it to prototype decisions made during the design process and assess them for fit.

We will explore connecting Access to SQL Server in future sessions, but for now we will work with self-contained structures within Access.

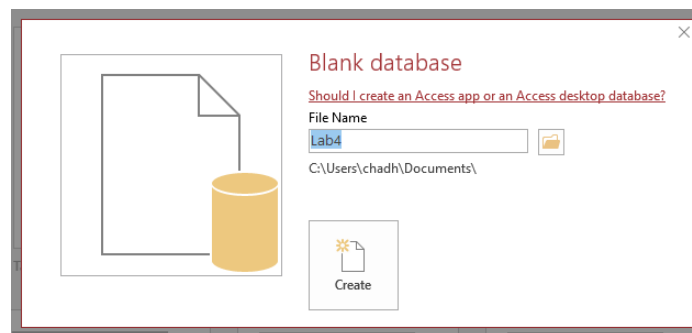
Note that all screen shots were captured for this process in Access for Office 365, version 1803. Some of the user interface elements may have changed slightly, but the objectives and the tasks should be the same.

To do


Let's make a simple database to store our books. Begin by opening Microsoft Access and creating a blank database.



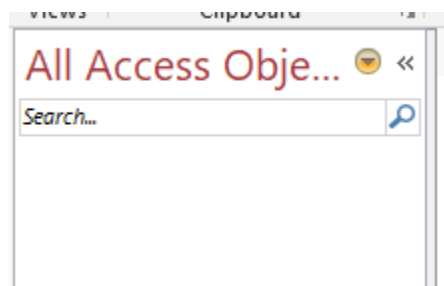
Name your database Lab4 (or similar) and click Create



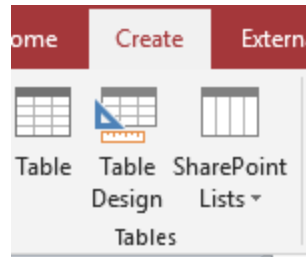
Access opens our new database and is ready for us to build the objects. It uses a tabbed layout for our open objects and has a list of all the database objects in a panel on the left-hand side of the window. There are several ribbons across the top that are context sensitive and change based on what object is active and what task we wish to perform.

By default, Access opens a blank table named Table1. Close this table using the  on the right-hand side of the screen. This will be along the same horizontal line as the tab labeled Table1.

Your All Access Objects panel should now be empty.



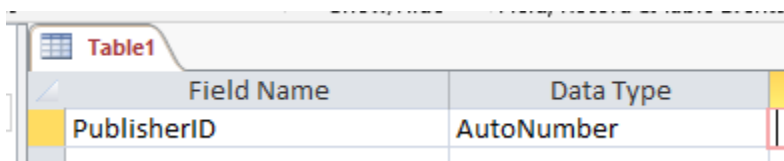
In the Create ribbon at the top of the screen, click Table Design. Table Design is available in the Tables group of that ribbon.



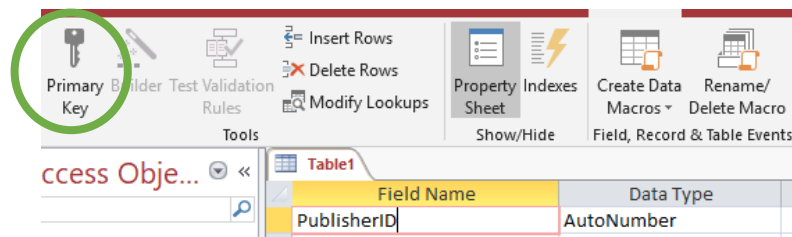
Access opens a new blank table named Table1 in Design mode. We will fill in our field names and data types before entering data.

Note: What we have been calling **attributes** and **columns** Access calls **Fields**.

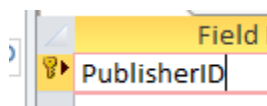
In the first field name, type PublisherID and pick AutoNumber from the Data Type drop down.



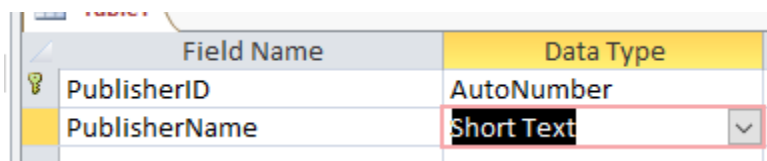
Ensure the cursor is on any part of the PublisherID line and click Primary Key in the Design ribbon at the top of the screen.



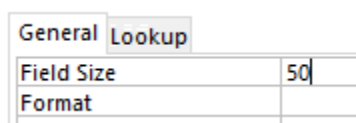
A small, yellow key will appear next to the PublisherID field name.



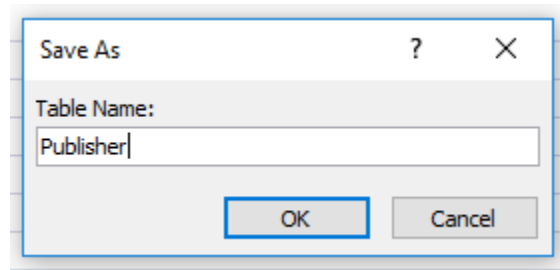
On the next line, type PublisherName in the field list and pick Short Text from the data type drop down.



With the cursor anywhere on the line with PublisherName, notice the properties at the bottom of the screen. Change the Field Size property to 50. Leave all other properties unchanged.

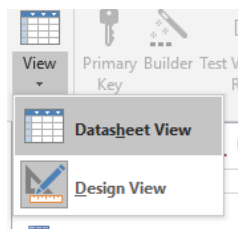


Click the save icon  in the upper left and name your table Publisher.



Click OK and the tab name changes from Table1 to Publisher. The Publisher table should also appear in the All Access Objects panel.

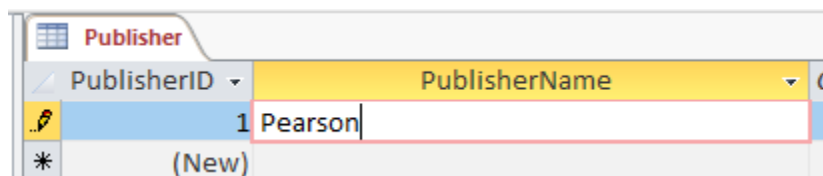
On the Design Ribbon, click the down arrow below the View icon (far left of the ribbon) and click Datasheet View



Now we can enter data! Because we specified PublisherID would be an AutoNumber field, we do not provide any values for this. Instead, the system will generate a number for us. This is called a surrogate primary key.

A surrogate primary key is “a serial number or other system-assigned primary key for a relation” (Hoffer, pg168)

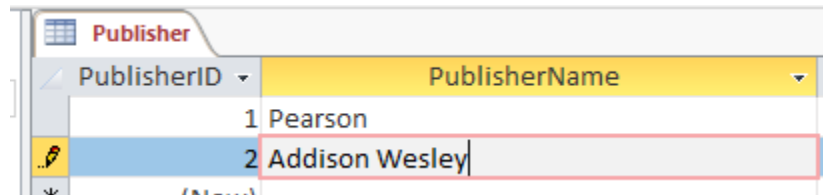
Add **Pearson** to the Publisher table by typing it in the Publisher Name column. You can resize the column widths by clicking on the divider between column headings and dragging to the right or left.



Take note of a couple things as you type. First, the PublisherID is automatically assigned by the system as promised. Also, there is a small writing-pencil icon in the record selector. This indicates that the record is “dirty”. That means it is currently being edited, but the changes have not been saved.

If you click in the PublisherName of the next box, the pencil goes away. This means that Access has saved the changes. This is important because we now know that there is an explicit save that happens whenever we modify data (for better or for worse!).

Add **Addison Wesley** to the Publishers table.

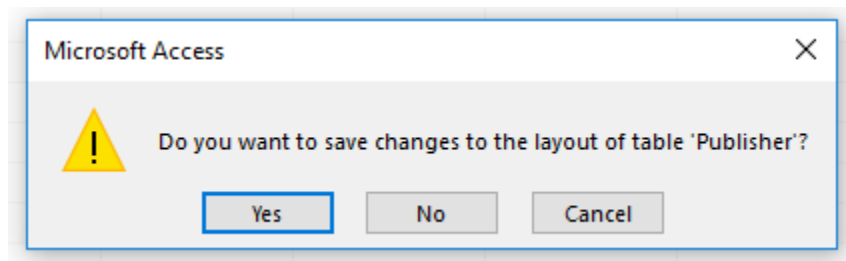


PublisherID	PublisherName
1	Pearson
2	Addison Wesley

Again, the PublisherID automatically fills in with the next available number.

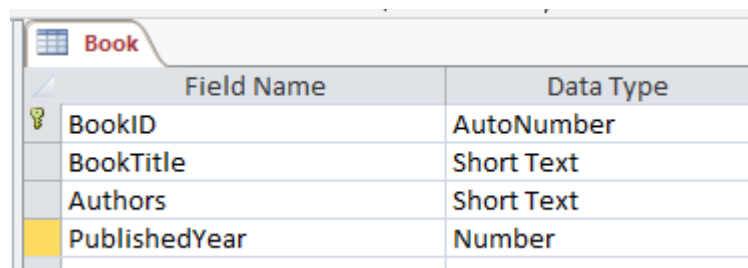
Close the Publisher table.

If you adjusted the column sizes, you will be asked:



It's okay to say yes to this. Access will preserve your column widths for you when you save.

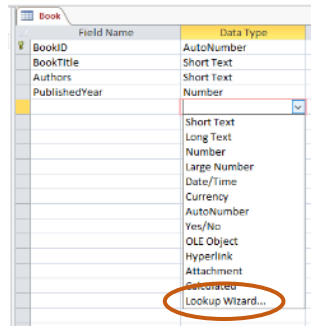
Create a new table in design mode called Book. Give it these fields and data types (feel free to leave the Field Size properties as is or customize them to whatever you see fit):



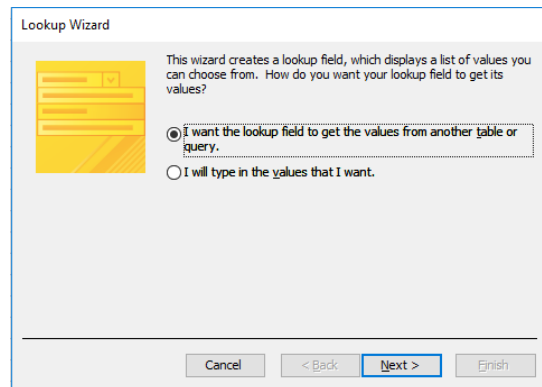
Field Name	Data Type
BookID	AutoNumber
BookTitle	Short Text
Authors	Short Text
PublishedYear	Number

Note that BookID is the Primary Key. Also, don't go to data sheet view just yet. We need to add a look up to the Publisher table.

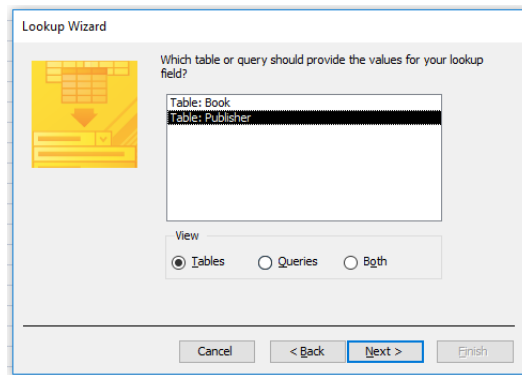
In the first empty line, click the drop down in the data type column and pick Lookup Wizard (last on the list)




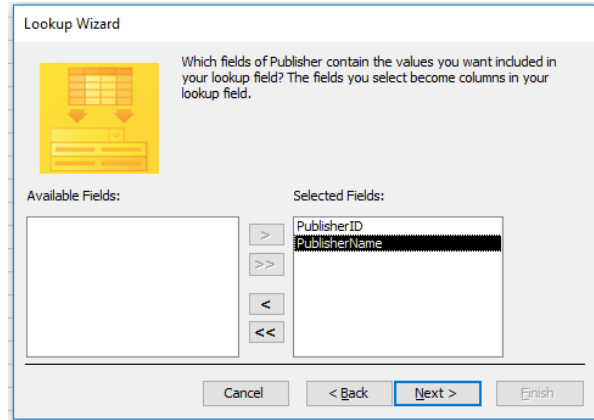
We want to look up values from another table, so leave that option selected and click Next.



Pick Table: Publisher from the list and click Next

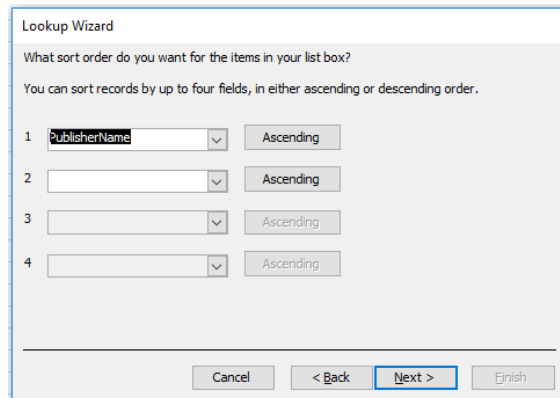


Add both PublisherID and PublisherName to the right-hand side by clicking the double chevron icon . We will need to store the PublisherID on disk, as it is the Primary Key of the Publisher table and will become a Foreign Key on the Book table, but we want the human-readable PublisherName to be what appears on the screen. The surrogate primary key is less useful for reading and finding purposes.

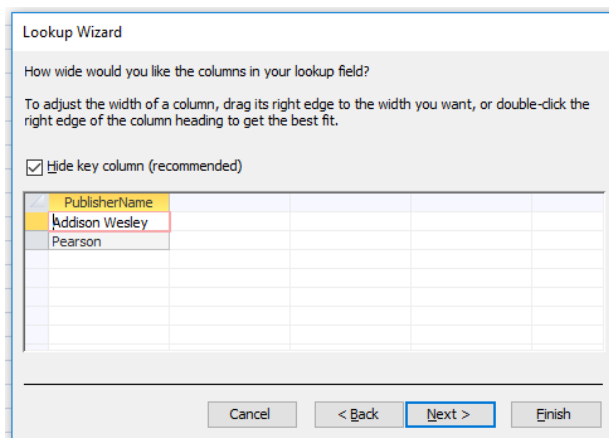


Click Next.

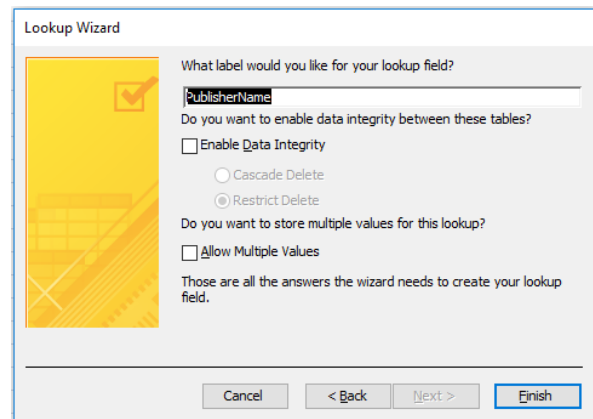
We want the ensuing drop down to sort by publisher name to make things easier to find, so select PublisherName from the first drop down and click Next



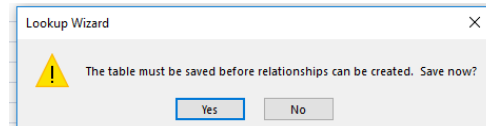
Ensure Hide Key Column is checked and click Next



Click Finish on the next screen.



Access is going to create a relationship for us between the Author and Publisher tables, but it needs to save the current table to do so. Say yes to this message.



Access has added a Field Name called PublisherName, but its data type is Number. This is because even though we will see the PublisherName on the screen, Access will be storing a Number (the PublisherID) on the disk.

Book	
Field Name	Data Type
BookID	AutoNumber
BookTitle	Short Text
Authors	Short Text
PublishedYear	Number
PublisherName	Number

Switch to Datasheet View and let's enter a couple books.

In the BookTitle column, enter **Languages and Machines**. Under Authors, enter **Sudkamp**. For PublishedYear, enter **1997**.

Book				
BookID	BookTitle	Authors	PublishedYe	PublisherNa
1	Languages and Machines	Sudkamp	1997	
(New)			0	

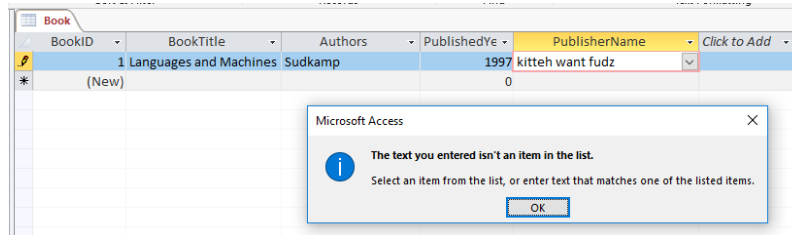
Notice that when you click in the PublisherName column, it is a drop-down box.

Book				
BookID	BookTitle	Authors	PublishedYe	PublisherName
1	Languages and Machines	Sudkamp	1997	
(New)			0	<div> Addison Wesley Pearson </div>

Pick **Addison Wesley** from the list. You can also start typing in the box and it will attempt to auto-fill the box with the first entry that matches what you have typed so far.

So, what?

By building our tables this way, we have made it easier to enter our data and have protected against bad data entering our system. If we try to enter a publisher that isn't in the list, Access prevents the change from happening. So, if a cat walks across the keyboard when you're entering data, that value doesn't accidentally save to the database for that book.







This also allows us to have publishers in our database even though we may not have books from that publisher. In this way, publisher is a distinct entity that we can manage in isolation.

Add more rows to your Book table, so that your table looks like this:

BookID	BookTitle	Authors	PublishedYe	PublisherName
1	Languages and Machines	Sudkamp	1997	Addison Wesley
2	Modern Database Management	Hoffer, Kamesh, Topi	2015	Pearson
3	Data Structures and Algorithm Analysis in C++	Weiss	2014	Pearson

Close the Book table.

Open the Publisher table by double clicking it in the All Access Objects table. You should now see a  next to each Publisher in the table. Click the  next to both publishers to show all your Books.

PublisherID	PublisherName	Click to Add
1	Pearson	
2	Addison Wesley	

BookID	BookTitle	Authors	PublishedYe	Click to Add
2	Modern Database Management	Hoffer, Kamesh, Topi	2015	
3	Data Structures and Algorithm Analysis in C++	Weiss	2014	
(New)			0	

BookID	BookTitle	Authors	PublishedYe	Click to Add
1	Languages and Machines	Sudkamp	1997	
(New)			0	

What to Submit

1. With each Publisher expanded in this way, take a screenshot of your Publisher table and include it in your answer doc.
2. Have another look at the Book table. In what normal form is this? If not in 3NF, what would need to be done to make it 3NF? Describe this in your answer doc.

Part 2 – Normalizing Video Data for VidCast

Setup

Because we did such a great job modeling our VidCast database, most of our tables are already relations in 3NF. After a few meetings, we have added to our requirements for tracking video data and need to make some adjustments to the tables to accommodate.

The following is a spreadsheet of data built by the design team.

Video ID	Video Title	User Name	User Tier	Min Tier Followers	Stream Start	Video Duration (mins)	Content Rating	Rating Description
1	Using SQL	ChadOnData	Platinum	1,000	5/9/18 21:40	48	E	Everyone
2	Just How Great is Jim Boeheim	CuckLazerbeam	Gold	500	1/9/18 10:50	111	E	Everyone
3	Top Hats are Cool	TheSaulHudson	Silver	200	5/30/18 23:10	131	M	Mature audiences
4	How Many Years for the Man in Black	TyroneRugen	Bronze	0	1/13/18 14:50	104	M	Mature audiences
5	Digital Underground: Are they the best?	CuckLazerbeam	Gold	500	2/20/18 23:30	20	C	Viewer caution recommended
6	Check out these new glasses	RegDwight	Platinum	1,000	11/2/18 22:32	17	E	Everyone
7	PolyDactyl and You	TyroneRugen	Bronze	0	9/28/18 10:16	180	C	Viewer caution recommended
8	Meeting Vizini at the Beginning	TheSpaniard	Platinum	1,000	11/26/18 8:55	123	E	Everyone
9	Me and My Les Paul	TheSaulHudson	Silver	200	3/15/18 5:17	64	M	Mature audiences
10	Careful Jogging of Memories: A Retrospective	RhymingGiant	Bronze	0	10/9/18 16:05	114	E	Everyone

To-do

For the above data set, decompose the relation into 3NF relations. You can either diagram this with Draw.io (or similar) or use the relational notation described in previous exercises.

Note: You do not need to incorporate prior lab diagrams into this.

What to Submit

Add your 3NF relations or diagram to your answer doc.

After completing Part 2, save and submit your answer doc.