URL: https://web.engr.oregonstate.edu/~ambattup/CS340_Website/

Project Step 4 Draft

Authors: Philip Ambattu, Shushanna Petrosyan

Group Number: 126

Group Name: The Terrible Tennis Team

Project Title: Racket Supply

Verbatim Feedback by the TAs and Peer Reviewers

STEP 1 FEEDBACK TA + #1 - 4

TA Submission Feedback:

Great work on your Step 1 Draft.

Some of the relationships in the ERD are incorrectly formed and one relationship is possibly redundant:

- The cardinality of the 1:M relationships between Orders---OrderDetails and Products---OrderDetails should be flipped. The many side should be against the OrderDetails table for both of these 1:M relationships.
- I see that you have a many-to-many (M:N) relationship between Products and Orders. Unless this relationship is serving to record some different information (other than products that have been are ordered), it is redundant to your schema and should be removed. An M:N relationship has already been formed between Orders and Products (through the Order Details table).

Keep up the good work!

Peer Review #1:

Yes the overview describes the problem as a popularity skyrocket across Los Angeles, and that the demand has overloaded the capacity of the manual sales tracking system that Terrible Tennis Teams (T-Squad) Racket Supply uses to manage its sales.

The overview lists specific facts involving the increase in demand since the pandemic and the popularity rise in Los Angeles. The specific fact is that sales roared by 50% in 2023. The specific requirement facts are also present where the database system of Terrible Tennis Teams (T-Squad) Racket Supply is designed to "manage inventory over 2,500 unique products and process more than 10,000 orders annually."

In the entity description outline there are exactly four entities listed. Customers, Products, Orders, Order Details ; These are the four listed entities. Each entity represents a single idea and can be stored as a list, for example Customers represents a single customer, and can store multiple, same for Products, and Orders and Order Details.

Each entity described in the outline of entity details lists attribute datatypes and constraints and describes the relationships between entities. The Customers and Products entities describes the purpose of each respectively.

Customers and Orders correctly formulate 1:M relationships there is not a M:M relationship. The ERD present does present a logical view of the database. The ERD diagram expresses the mandatory to many optional relationship between customers and orders which logically represents how the Customer to Order relationship works. Each order can have multiple products and does logically represent that. The ERD diagram roughly estimates how the system will work.

There is consistency in the naming convention between overview and the entity/attributes where the entities are plural and the attributes are singular. The convention of capitalization for naming is present and used. All of the names for each entity table shares the same name between the overview and the erd and each attribute also shares the same name between the overview and the erd.

Peer Review #2:

The overview describes a business that has grown such that manual tracking is no longer practical, and a website with a DB backend could help solve the tracking of inventory and orders more effectively. The tracking of 2500 products and 10000 orders annually give a good idea of the throughput and general storage capacity necessary for the database.

The first entity described is Customers, which tracks customer details and is 1:M related to orders. Orders tracks orders made by customers, and is also 1:1 related to Order Details. Order Details also tracks order information, and is 1:M related to products. Products track individual product details. I don't know that Order Details and orders really need to exist as separate entities. Maybe you could instead track suppliers, which provide multiple different products.

The outline does not describe the purpose of each entity, but it does list datatypes of each attribute, as well as their constraints. The relationships between the four entities seem reasonable and accurate, except that the strange relationship described between products and orders, with the order details as the go between, doesn't make much sense, or indeed work as a M:M relationship.

The 1:M relationships make sense, but there is no M:M relationship as such. The ERD seems to work within the logic laid out by the outline. The naming conventions are internally consistent, as well as within the bounds of recommended conventions from the course material.

Peer Review #3:

Does the overview describe what problem is to be solved by a website with DB back end?

The overview perfectly described the problem in a way that is practical and understandable. I loved the fact that it used a real-world example to illustrate its point about the fact that slow retrieval of data can cause dissatisfaction among consumers. Thus, to mediate that specific problem, they would need a more advanced database that can handle large amounts of data and efficiently retrieve those data, so that consumers don't have to wait for a long time. To be specific, the document states how a tennis company met a large number of customers, which slowed their system. The document then stated how it will address this problem of delayed processing and inventory issues with a new data-driven database website. I love the fact that it took a practical example and used it to create a practical solution to it that depends on a website with DB back end.

Does the overview list specific facts?

Yes, the overview stated how their system will be designed to manage inventory of over 2,500 unique products and process more than 10,000 orders every year. They did this, because The Terrible Tennis Team's Rocket Supply received a 50% growth in their sales in 2023. However, their old system could not handle the massive growth, and thus needs a new website with DB back end. It also specified how problems such as inventory issues, delayed order processing, and customer satisfaction reduction can be resolved by this new system.

Are at least four entities described, and does each one represent a single idea to be stored as a list?

Yes, there are at least four entities described in the document. Each one also represents a single idea to be stored as a list. For instance, the entity Customer represents the consumers that will partake in their systems. That idea, being the consumer, is stored as a list. The Products represents the products that are available for sale, Orders, which are the orders of the products from a customer, and the Order details, which will specify the details of that order. All of these are stored as a list.

Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints, and describe relationships between entities?

Yes, the outlines are logically sound, and describe the purpose of each of the entities, their list attribute datatypes and constraints, as well as the relationships between those entities. For instance, Customers contains 4 attributes, each of those attributes have specific datatypes that are stated, and the constraints, as well as the relationships that exists between those attributes. This is also done for Products, Orders, and Order Details.

Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?

The 1:M relationships are correctly and logically formulated between each of the entities. There is also at least one M:M relationship, which is the Products and Orders. In other words, a single product can have many orders, and one order can have many products. Finally, the ERD does indeed present a logical view of the database in an organized manner. The connections that are used between the entities are also accurate to the relationships that were stated in the document itself.

Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

a) The naming between overview and entity/attributes are logically acceptable. Since they are dealing with Tennis sales, the name of their overview and entities matches that description.

b) The entities are indeed plural, and the attributes are singular.

c) There usage of capitalization for naming is perfectly sound, especially because they used CamelCase to make it easier to read.

Peer Review #4:

Does the overview describe what problem is to be solved by a website with DB back end?

Yes, the overview describes the problem accurately by describing how the company's growth has outgrown the manual sales tracking system. The overview does well to highlight the problems that a website with a DB back end would solve, such as logging customer orders, enhancing efficiency, and increasing customer satisfaction.

Does the overview list specific facts?

The overview includes facts such as "sales have increased by 50% in 2023". It also lists how big the database will need to be to accommodate the amount of products and orders. I think it would be a good idea to include the amount of orders for the previous year so it would help to understand how the size of the database was agreed upon.

Are at least four entities described, and does each one represent a single idea to be stored as a list?

Yes, there are four entities. Customers, Orders, Order Details, and Products are all single ideas to be stored as individual lists.

Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints, and describe relationships between entities? If yes, summarize. If not, based on the course material, what changes can you suggest to improve?

The outline of entity details is straightforward and accurately describes datatypes, constraints, and relationships between entities. I think you should include the purpose of "Orders" and "Order Details" in your outline.

Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database? If yes, summarize. If not, based on the course material, what changes can you suggest to improve?

One customer can have many different orders. One order detail can have many products associated with it. One order detail can have many orders associated with it. I do see M:M relationship between products and orders, because one order can have many products, and one product can be associated with many orders.

Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming? If yes, summarize. If not, based on the course material, what changes can you suggest to improve?

Yes, the naming is consistent and uses camelCase. Attributes are singular and entities are plural, which make the tables easy to understand.

STEP 2 FEEDBACK #5 - 8

Peer Review #5:

Hi Group 126!

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly? If yes, summarize what you see. If not, what changes would you suggest be made?
The outline, ER, and schema match. There are four tables: OrderDetails, Orders, Products, and Customers.

Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming? If yes, summarize what the group uses for each of the above. If not, what changes would you suggest be made for improved consistency?
Everything seems to match up, except the attribute "status" in the Orders table. It is capitalized in the outline, but lowercase in the ER and schema. Otherwise, entities are plural and capitalized, attributes are in camelCase.

Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)? If yes, summarize what makes the schema clear and easy to read. If not, what changes would you suggest to improve the schema's readability?
The schema is very clean and easy to read. None of the relationship lines overlap.

Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)? If yes, summarize the components you see making up the properly formed intersection tables/M:N relationship. If not, what changes would you suggest to improve them?
The Order Details intersection table has two foreign keys from the Orders and Products tables and is properly formed.

Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies? If yes, what are the issues, and what can be done to improve upon them? If not, summarize how the sample data shows normalization.
It seems that everything is normalized properly. Each column holds a single value, non-key attributes depend on the primary key, and non-key columns seem to depend on the primary key as well.

Is the SQL file syntactically correct? This can be easily verified by using phpMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do

this!) If yes, summarize what you see upon importing (e.g. any status message, which tables were imported AND populated with their corresponding sample data, etc.). If not, what happened with the import/what appears to be the issue(s) with the file that are causing this?

In the SQL, are the data types appropriate considering the description of the attribute in the database outline? If yes, summarize why they seem fitting. If not, what changes would you suggest based on the attribute descriptions?
The SQL file looks syntactically correct. The tables are created correctly with the correct attributes. The sample data is inserted correctly.

In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? If yes, summarize how they are correctly defined, and where CASCADE operations are declared. If not, what changes should be made to the key definitions and/or what CASCADE operations would you suggest declaring?
The primary key of each table is correctly defined. The foreign keys, where appropriate, are correctly defined as well. The two foreign keys in the intersection table have CASCADE operations declared.

In the SQL, are relationship tables present when compared to the ERD/Schema? If yes, summarize the relationship tables present in both the SQL and the ERD/Schema. If not, which relationship tables aren't present?
The OrderDetails table is present in both SQL and the ER/Schema. The primary key orderDetailID is present, the foreign keys orderID and product ID are present, and the attributes quantity and individualPrice are present.

In the SQL, is all sample data shown in the PDF INSERTED? If yes, summarize which tables in the SQL have the sample data from the PDF inserted. If not, what sample data is missing or inconsistent?
Customers, Products, Orders, and OrderDetails tables have the same insert sample data in the SQL file as is presented in the PDF file.

Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)? If yes, summarize the structure and comments you see. If not, what changes would you suggest to improve the SQL file?
The SQL code is well structured, in my opinion. It does not have any commenting; be it hand authored or from an export service. That being said, I do not see any complicated areas in the SQL file that would warrant extensive explanation through commenting.

Peer Review #6:

Great work, Terrible Tennis Team! Thoughts and feedback below:

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly? If yes, summarize what you see. If not, what changes would you suggest be made?
Yes, so you have three tables -- Customers, Orders, and Products -- and an intersection table, OrderDetails, that connects Orders and Products.

Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?
Yes, attributes are singular and in Camel Case, entities are plural and Pascal Case.

Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)? If yes, summarize what makes the schema clear and easy to read. If not, what changes would you suggest to improve the schema's readability?
The schema's nice and easy, though I might center OrderDetails, since it's sort of the heart of everything happening in this database, as it's 1) an intersection table that connects 2 of the 3 independent entities in the database, and 2) contains more information than your average intersection table.

Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)? If yes, summarize the components you see making up the properly formed intersection tables/M:N relationship. If not, what changes would you suggest to improve them?
Yes! The intersection table OrderDetails has its own auto-incrementing PK, but then incorporates FKs from Orders and Products.

Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies? If yes, what are the issues, and what can be done to improve upon them? If not, summarize how the sample data shows normalization.
No, the sample data looks really clean -- Customers has a PK and then everything else stays there; Products works similarly. Orders incorporates CustomerID as a FK, and then OrderDetails has its own PK, FKs for Orders and Products...yeah, it looks good. There's a little redundancy in the sample data between Products.priceValue and OrderDetails.individualPrice, but 1) I understand why that's there, and 2) I think it looks more redundant because every order contains just 1x of each item right now. Might be worth having an order have, say, 2 rackets or something, to break that up.

Is the SQL file syntactically correct? This can be easily verified by using phpMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!) If yes, summarize what you see upon importing (e.g. any status message, which tables were imported AND populated with their corresponding sample data, etc.). If not, what happened with the import/what appears to be the issue(s) with the file that are causing this?
Yes! No error messages, all queries processed quickly, data was inserted into tables Customers, Orders, OrderDetails, and Products.

In the SQL, are the data types appropriate considering the description of the attribute in the database outline? If yes, summarize why they seem fitting. If not, what changes would you suggest based on the attribute descriptions?
Yes! Plenty of VARCHARs, which we love, DECIMAL for price, DATETIME for orders...yeah, looks good.

In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared? If yes, summarize how they are correctly defined, and where CASCADE operations are declared. If not, what changes should be made to the key definitions and/or what CASCADE operations would you suggest declaring?
Yes, PKs and FKs look good. CASCADE in OrderDetails for both FK OrderID (so if an Order gets deleted, its associated OrderDetails are deleted) and for FK productID (so if a product gets deleted, its associated OrderDetails are deleted). Maybe it's just preference, but I'm not sure I'd delete OrderDetails if a product's PK was deleted?

In the SQL, are relationship tables present when compared to the ERD/Schema? If yes, summarize the relationship tables present in both the SQL and the ERD/Schema. If not, which relationship tables aren't present?
Yes! So you have entity tables for Customers, Products, and Orders, and an intersection table for Products - Orders.

In the SQL, is all sample data shown in the PDF INSERTED? If yes, summarize which tables in the SQL have the sample data from the PDF inserted. If not, what sample data is missing or inconsistent?
So the sample data matches the SQL data, though I'm curious about the data in OrderDetails.orderID 1003 -- that order's total price is $139.97, and it's made up of 1 product 101 (Tennis Racket Pro, $99.99) and 1 product 103 (Tennis Shoes, $79.99), which don't add up to the order's total price. Maybe you meant to do 2 of product 102 in addition to 1 of 101? That'd get $139.97 and it'd also add some variety to OrderDetails.quantity.

Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)? If yes, summarize the structure and comments you see. If not, what changes would you suggest to improve the SQL file?
Yes! SQL is very clean, though there are no comments to indicate that it was hand-authored (which is fine, just noting it here).

Peer Review #7

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?
Yes, the new schema follows the diagram which matches the the database out 'in words'. There are four tables in each including 1 intersection table.

Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

Yes this group using camel-casing for everything, there are no outliers except for variables that contain only 1 word but that's fine, since its still camel-case

Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?

Yes, the schema is clear and easy to read, the ER diagram relationship lines do no intersection, the PK and FK's are clearly labeled, everything has a consistent naming-scheme.

Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?

Yes, the intersection table Order Details facilitates the M:N relationship between products and order. Not only does it contain orderID and productID as FK, it has its own fields for quantity and individual price.

Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?

There does not appear to be issues related to the schema design or normalization. It looks like everything is pretty well compartmentalized with no full or even partial dependencies that I can see.

Is the SQL file syntactically correct? This can be easily verified by using phpMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!)

Yes, the SQL file executes properly and I can query the sample data that it generates.

In the SQL, are the data types appropriate considering the description of the attribute in the database outline?

Yes, the types seem to fit. Decimal (up to 2 place values) is used for monetary amounts, integers are used for quantities etc.

In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared? If yes, summarize how they are correctly defined, and where CASCADE operations are declared. If not, what changes should be made to the key definitions and/or what CASCADE operations would you suggest declaring?

Yes, the cascade operations are declared on tables that contain foreign keys which allows parent rows to delete themselves and rows which depend on them.

In the SQL, are relationship tables present when compared to the ERD/Schema?

Yes, both have four tables with corresponding data types.

In the SQL, is all sample data shown in the PDF INSERTED?

Yes, they have three entries for orders, 5 for orderDetails, 3 for products and 3 for customers. Additionally, they show possible values for fields like Status which can be Completed or Pending.

Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?

The SQL is well-structured and easy to read, though it doesnt have any comments. If not understanding what was done may be a problem this should definitely be updated to include comments.

Peer Review #8

Does the schema present a physical model that follows the database outline and the ER logical diagram exactly?

Yes, the schema, database outline, and ER diagram all match, to the best that I can tell. The four tables are: Customers, Orders, Order Details, and Products, where Order Details is the intersection table with many to many relationships with Orders and Products.

Is there consistency in a) naming between overview, outline, ER and schema entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

Yes, there is consistency in naming. All tables are capitalized, and all attributes use pascal case. The use of pascal case and capitalizing table names seems to be consistent across overview, outline, and ER and schema diagram.

Is the schema easy to read (e.g. diagram is clear and readable with relationship lines not crossed)?

The schema is pretty clear, but there are a few small changes I would make. Instead of labeling the ER diagram as "old" and schema as "new", just label them with their respective names (ER diagram and schema). Also, while the nature of the relationships between tables is clear in the ER diagram, I think it could be a little clearer in the schema (bigger lines/maintaining the relationship symbols on the lines)

Are intersection tables properly formed (e.g. two FKs and facilitate a M:N relationship)?

Yes, the intersection table is properly formed. There are two foreign keys originating from M:N relationships with Products and Orders. There is also a distinct, non-composite primary key.

Does the sample data suggest any non-normalized issues, e.g. partial dependencies or transitive dependencies?

The sample data doesn't have any normalization issues. One potential dependency problem I thought about was if individualPrice would be effected if priveValue is deleted from Products table. However, due to the ON DELETE CASCADE, this problem is avoided.

Is the SQL file syntactically correct? This can be easily verified by using phpMyAdmin and your CS 340 database (do not forget to take backup of your own database before you do this!)

The SQL file is syntactically correct. Upon importing the DDL file, everything runs correctly. I am able to query the tables, and selecting for all entries in each successive table returns the sample data in its correct form.

In the SQL, are the data types appropriate considering the description of the attribute in the database outline?

The data types are all appropriate and match the description of the attributes in the outline. When returning the sample data, the output is as expected.

In the SQL, are the primary and foreign keys correctly defined when compared to the Schema? Are appropriate CASCADE operations declared?

One question I have is for the primary keys in each table, the outline indicates that the keys must not be null. Shouldn't the primary key declarations also have NOT NULL? However, as previously mentioned. the cascade operation declarations are correct.

In the SQL, are relationship tables present when compared to the ERD/Schema?

Yes, the relationship tables are present and are properly connected to the other tables. OrderDetails is properly connected to Orders and Products, and Orders is properly connected to Customers.

In the SQL, is all sample data shown in the PDF INSERTED?

Yes, all data from the PDF has been successfully inserted into the database. This includes example customers, products, orders, and order details. One minor comment I would have is- would it make sense to also have a last name attribute for customers? While not strictly necessary, I think it would look more professional.

Is the SQL well-structured and commented (e.g. hand authored) or not (e.g. exported from MySQL)?

While the SQL is well structured, I think it could use much more comments describing the different table declaration, sample data, etc.

Overall, great job group 126!

STEP 3 FEEDBACK TA + #9 - 12

DRAFT TA Submission Feedback

Team 126,

Excellent work updating your report and on implementing the DML queries.

Please ensure that the website has all of the forms required for the project. Please reference the CS340 Project Guide (https://canvas.oregonstate.edu/courses/1976520/pages/cs340-project-guide?module_item_id=24719009) for more details.

- Did not see any delete forms
- Does adding a new row on the Orders page also add a new line to the OrderDetails table? If so, perhaps consider adding a note under the 'Add New Order' form.

Good work!

FINAL TA Submission Feedback For Final

From Rubric –

HTML Pages Quality: No edit/update forms implemented

Data Manipulation Queries: No JOINs used to make foreign keys user friendly

Peer Review #9

Does the UI show where it will utilize a SELECT for every table in the schema? **In other words, a data table for each table in the schema should be displayed on the UI (which are not required to be populated with sample data, but should at least have column names). If yes, which tables from the schema do you see fully represented in the UI with a SELECT? If not, which tables and/or attributes are missing? Note: it is generally not acceptable for just a single query to join all tables and display them.**

Yes, the UI does a good job of utilizing a SELECT for each table without relying on a single massive query to join them all. Each table in the database is well represented in the UI. For the Customers table, there’s a page that shows columns for Name, Email, and Phone, which matches all the attributes in the database. The Products table is covered with a page that displays columns for Product Name, Price, Category, and Quantity Left. The Orders table also has its own section, showing Order ID, Customer, Order Date, Status, and Total, which captures everything it needs from the schema. Lastly, there’s a page for Order Details that lists Product, Quantity, Price, and Total. One issue may be that the orderDetailID attribute doesn’t explicitly appear in the UI, but this ID may not really be necessary for display, so I guess it’s fine that it’s left out. Overall, each table in the database has its place in the UI where users can view or interact with the data.

Does at least one form utilize a search/filter/dropdown with a dynamically populated list of properties? **If yes, which form(s) have which features incorporated? For which attribute(s)? If not, where are a couple places this could be implemented?**

Yes, the UI does have a couple of forms that use dropdowns filled with data from the database. For example, on the Orders page, there’s a dropdown for choosing a customer which likely populates all the names from the Customers table. There's also a product dropdown on the same page, showing available items with their names and prices. If we were looking to add more search or filter options, a couple of ideas come to mind. On the Products page, you can possibly add a filter by category or price range, as that will make it easier to find specific items, especially when the list gets long. Another feature you can add is a search feature on the Order Details page, which will allow users to find orders by product name or date.

***Does the UI implement an INSERT for every table in the schema?* In other words, there should be UI input fields that correspond to each table and attribute in that table. If yes, which tables from their schema do you see a complete INSERT for? If not, which tables either do not have INSERTs, or have attributes missing from their INSERT? In general, do you have any suggestions for their INSERT forms?**
The UI mostly covers INSERT forms for each table, but there are ways it can be improved. The Customers page has a form that lets you add new customers, with fields for Name, Email, and Phone, so that one's good to go. On the Products page, you can add new products with fields for Product Name, Price, Category, and Quantity—pretty much everything needed for the Products table. The Orders page, however, is a bit different. It has a form where you can select a customer, choose a product, and add the quantity, but it doesn't seem to have fields for things like orderDate, totalAmount, or status for the Orders table. Like, what if a new order from the same customer is given on a separate date? And we need to indicate that the new order is pending until it is completed, right? So those might be set automatically in the background or require a separate form to edit after the initial entry.

***Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?* In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and line_total). Or, alternatively, there should be an INSERT for INSERTing into the intersection table(s) directly. If yes, list all the table INSERTs that correctly add their corresponding FK attributes, and describe the group's implementation for INSERTing into the intersection of their M:M relationship. If not, which INSERTs need to be altered, and in what way?**
The UI does have forms that handle some of the foreign key stuff, but it's not set up to cover everything especially when it comes to managing the OrderDetails table, which links Orders and Products in a many-to-many relationship. On the Orders page form, you can pick a customer, select a product, and set the quantity, and this covers adding the orderID, customerID, and productID, but the form doesn't have fields for all the details in OrderDetails. For example, there's nowhere to enter the individualPrice of each item or calculate the line_total. To make this

many-to-many relationship easier to manage, it'd be helpful if the Orders page had extra fields to let users add or review the individualPrice and confirm the line_total directly. You can probably also have a dedicated form just for OrderDetails so everything, including the orderID and productID links, is entered correctly. Right now, the setup kind of covers the basics, but adding some fields or a separate form would make it way easier to handle all the details in OrderDetails properly.

Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship? **In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers. If yes, describe all the DELETEs you see. If not, what's missing in terms of this requirement?**
The UI doesn't seem to have any DELETE buttons at all, not for any rows in any of the tables, so you're stuck with whatever's entered. If you wanted to delete something, like an order, there's no option to do it. Ideally, deleting an order should automatically clean up the related entries in OrderDetails without affecting anything in Products or Customers, however, that option just isn't there. And it's the same story if you want to delete a product; there's no DELETE function for that either.

Is there at least one UPDATE for any one entity, with fields for the corresponding attributes for that entity? **In other words, in the case of Products, can productName, listPrice, qtyOnHand, e.g. be updated for a single ProductID record? If yes, describe all the UPDATEs you see. If not, where would you suggest adding an UPDATE, or what's missing from the UPDATE(s) you see?**
Aside from the Update Order form for OrderDetails that lets you adjust details like product and quantity for an order, It looks like there's no way to update anything else in the UI for any of the other tables. For example, on the Products page, there's no option to edit or change things like productName, priceValue, or quantityLeft for a specific product. So if you need to adjust a product's price or update the stock count, there's no way to do it here. Perhaps, you could have an option next to each product that opens up a form to update the details of its attributes. The same goes for other tables like Customers and Orders.

***Is at least one relationship NULLable?* In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus, it should be feasible to edit an Order and change the value of Employee to be empty. If yes, which NULLable relationship(s) do you see, and does it seem to make sense? If not, where would you suggest adding a NULLable relationship and why?**

It seems like most relationships between tables are required, with no outright NULLable ones. For example, each order has to be linked to a customer since customerID in the Orders table is required, which makes sense because every order typically needs a customer. The same goes for Products and Order Details since each entry in Order Details has to have a productID, so you can't have an order item that doesn't link to a product. Also, every item in Order Details must have an orderID, tying it directly to a specific order. However, even though Order Details requires both orderID and productID to connect products and orders, Products itself doesn't require an order to be stored in the database. So, a product can exist without any connection to an Order Details entry until it's actually ordered. This sort of creates a NULLable relationship on the Products side, which allows for a product to be in the system independently until they're added to an order.

Do you have any other suggestions for the team to help with their HTML UI? For example, using AS aliases to replace obscure column names such as fname with First Name.

It seems like this UI was built with customers in mind rather than for the company to manage the database directly. The forms and dropdowns are mostly focused on letting users place orders, browse products, and fill in basic info, but it doesn't go deep enough for backend details that a company might need. I would suggest the group consider making this an internal use and add more robust forms that cover everything, especially in areas like OrderDetails, where each entry links an order to a product. A company facing UI would probably include options for things like setting individualPrice and calculating line_total right in the interface, which would make it way more practical for the company to keep track of everything directly.

Peer Review #10

Does the UI show where it will utilize a SELECT for every table in the schema?
Yes, the UI includes pages for each primary table—Customers, Products, Orders, and Order Details. For example, the Customers page lists columns for Name, Email, and Phone, while the products page shows the Product Name, Price, Category, and Quantity.

Does at least one form utilize a search/filter/dropdown with a dynamically populated list of properties?
Yes, there is a search feature on the customers page and a drop-down for the products on the orders and order details page. The orders page also has a drop-down to be able to add a new order and select from one of the customers already in the system. Though I do think it is a good idea to provide this, a search option might be better in this option as there may come to be a large amount of customers and this could be overwhelming to find them in the list.

Does the UI implement an INSERT for every table in the schema?
Yes, all tables include an insert option when adding new information and updating. This is included in the Customers, Products, Orders, and Order Details pages. Customers has Name, Email, and Phone. Products has Name, Price, Category, and Quantity. Orders has Order Date, Total Amount, and Status.

Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?
Yes, the UI does manage foreign key relationships but could use some work when it comes to M:M relationships.

Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship?
Currently, DELETE functionality is missing, so entries cannot be removed. Ideally, deleting an order would remove its linked entries, to avoid discrepancies.

Is there at least one UPDATE for any one entity, with fields for the corresponding attributes for that entity?
Only OrderDetails offers an update form for product and quantity. Other tables, like Products, lack options to edit attributes (e.g., price, quantity), which would be valuable for updating inventory.

Is at least one relationship NULLable?
Most relationships are required, like customerID in Orders, making each order customer-specific. However, Products can exist independently until they're ordered, allowing for a NULLable link on the Products side.

Do you have any other suggestions for the team to help with their HTML UI?
I would suggest adding a navigation bar for the site so that it makes accessing each page more seamless. However, everything looks straightforward on your site and easy to understand.

Peer Review #11

Does the UI show where it will utilize a SELECT for every table in the schema?
The UI effectively demonstrates the use of SELECT queries for each individual table within the schema. Every primary table, such as Customers, Products, Orders, and Order Details, is represented with distinct pages that display their respective columns. For example, the Customers page includes columns for Name, Email, and Phone, while the Products page presents Product Name, Price, Category, and Quantity Left. This design avoids relying on a single large query that joins multiple tables, which can simplify data management. Though the orderDetailID column is not shown on the UI, its absence might be appropriate since it may not be critical for user interaction.

Does at least one form utilize a search/filter/dropdown with a dynamically populated list of properties?
Some forms in the UI include dropdown menus that are dynamically populated based on database data, such as the Customer selection dropdown on the Orders page. Additionally, the Products dropdown on the same page displays item names and prices. For added convenience, filter options could be incorporated on the Products page, allowing users to search by category or price range, which would be helpful as the product list grows. Similarly, a search option on the Order Details page could enable users to filter orders by product name or date.

Does the UI implement an INSERT for every table in the schema?
The UI generally includes INSERT forms for most tables. For instance, the Customers page provides fields for adding Name, Email, and Phone, while the Products page includes fields for Product Name, Price, Category, and Quantity. The Orders page, however, could be more comprehensive, as it currently lacks fields for attributes like orderDate, totalAmount, and status. These fields might be automatically managed in the background or could require a separate form for editing details post-initial entry.

Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship?

While some foreign key relationships are managed within the UI, certain aspects of many-to-many relationships, particularly in the OrderDetails table (which links Orders and Products), are less complete. The Orders form allows users to select a customer and a product while setting the quantity, covering some required fields. However, it lacks fields for details like individualPrice and line_total. Adding fields on the Orders page for individualPrice and line_total, or creating a separate form for OrderDetails, could simplify managing this relationship.

Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship?

The UI currently lacks DELETE functionality for any of the tables, meaning that users cannot remove entries once added. Ideally, deleting an order should also remove its related entries in OrderDetails without impacting Products or Customers. Adding DELETE capabilities for orders and products would help maintain data integrity and make the UI more functional.

Is there at least one UPDATE for any one entity, with fields for the corresponding attributes for that entity?

Currently, OrderDetails is the only table that offers an update form, allowing users to modify details like product and quantity for an order. Other tables, such as Products, do not have update options for attributes like productName, price, or quantity. Adding UPDATE options for these attributes would facilitate inventory management. Similar functionality could also be extended to tables like Customers and Orders for greater flexibility.

Is at least one relationship NULLable?

Most relationships within the schema are mandatory; for example, each order must have an associated customer (with customerID as a required field in the Orders table). However, products are not required to be linked to orders, allowing them to exist independently in the system until added to an order, which creates a NULLable relationship for Products. This design permits products to be in the system without needing a direct link to an order until they are actively used in one.

Do you have any other suggestions for the team to help with their HTML UI?

One suggestion might be that in order to improve navigation on your website, you can try adding a navigation bar to allow users to easily identify and access different webpages. While the UI as it stands is user-friendly and intuitive, these added functionalities could support backend management and enhance the interface's usability.

Peer Review #12

Hi, team 126.

Here is my review:

Does the UI show where it will utilize a SELECT for every table in the schema? **In other words, a data table for each table in the schema should be displayed on the UI (which are not required to be populated with sample data, but should at least have column names). If yes, which tables from the schema do you see fully represented in the UI with a SELECT? If not, which tables and/or attributes are missing? Note: it is generally not acceptable for just a single query to join all tables and display them.**

Yes, the UI implements SELECT statements for each table, including Customers, Products, Orders, and OrderDetails.

Does at least one form utilize a search/filter/dropdown with a dynamically populated list of properties? **If yes, which form(s) have which features incorporated? For which attribute(s)? If not, where are a couple places this could be implemented?**

Yes, the Products table provides a dynamically populated dropdown for categoryItem by using the DISTINCT query. This dropdown enables users to filter products by category in the UI, enhancing the user experience by allowing more specific searches.

Does the UI implement an INSERT for every table in the schema? **In other words, there should be UI input fields that correspond to each table and attribute in that table. If yes, which tables from their schema do you see a complete INSERT for? If not, which tables either do not have INSERTs, or have attributes missing from their INSERT? In general, do you have any suggestions for their INSERT forms?**

Yes, the UI includes INSERT functionality for each table. Each table’s INSERT operation has an input form for users to add records with fields corresponding to each table’s attributes.

Does each INSERT also add the corresponding FK attributes, including at least one M:M relationship? **In other words if there is a M:M relationship between Orders and Products, INSERTing a new Order (e.g. orderID, customerID, date, total), should also INSERT row(s) in the intersection table, e.g. OrderDetails (orderID, productID, qty, price and**

line_total). Or, alternatively, there should be an INSERT for INSERTing into the intersection table(s) directly. If yes, list all the table INSERTs that correctly add their corresponding FK attributes, and describe the group's implementation for INSERTing into the intersection of their M:M relationship. If not, which INSERTs need to be altered, and in what way?

Yes, the UI includes INSERT operations that handle FK attributes, particularly for the M:M relationship in OrderDetails. When adding new entries to OrderDetails, users specify both orderID and productID, which ensures the insertion process reflects FK dependencies. However, users need to manually add data to OrderDetails, as there isn't an automated join between Orders and Products.

Is there at least one DELETE, and does at least one DELETE remove things from a M:M relationship? **In other words, if an order is deleted from the Orders table, it should also delete the corresponding rows from the OrderDetails table, BUT it should not delete any Products or Customers. If yes, describe all the DELETEs you see. If not, what's missing in terms of this requirement?**

Yes, the UI includes DELETE functionality for the M:M relationship. When an order is deleted from the Orders table, related rows in OrderDetails are deleted via the ON DELETE CASCADE constraint. This ensures that only dependent entries in OrderDetails are removed, leaving Products and Customers intact.

Is at least one relationship NULLable? **In other words, there should be at least one optional relationship, e.g. having an Employee might be optional for any Order. Thus, it should be feasible to edit an Order and change the value of Employee to be empty. If yes, which NULLable relationship(s) do you see, and does it seem to make sense? If not, where would you suggest adding a NULLable relationship and why?**

Yes, UPDATE functionality exists for several entities, including Customers, Products, and Orders. Users can update key details, such as customer information, product quantity and price, and order status, within the UI forms.

Do you have any other suggestions for the team to help with their HTML UI? For example, using AS aliases to replace obscure column names such as fname with First Name.

Create links on the side pages to the home page. Add some borders to the table.

Actions Based on Feedback

Step 1 Edits

Based on the feedback we received, we made four changes to our project draft. TA feedback complimented the work overall and pointed out that there was a redundant M:N relationship depicted in the ERD, as well as incorrect cardinality between two 1:M relationships. Based on this, we removed the extra line between Products and Orders as the M:N relationship requirement was already accomplished by using the OrderDetails table as an intersection. In addition, we edited the relationship between both Products—OrderDetails and Orders—OrderDetails to correct the cardinality. The “many” side is now depicted correctly on OrderDetails for both relationships.

We also received four peer reviews that encouraged small but meaningful changes. Peer review #4 encouraged adding more numerical information to the overview to better explain the need for a DB solution. While the overview already included information regarding a 50% growth, the scope was not defined well. In response, we added values to show that sales had doubled to 6500 total orders in 2023 with 1400 unique products. Two peer reviews mentioned that we missed the description and purpose of the Orders and OrderDetails entities. We have added a description for each. In addition, the peer reviews all mentioned inconsistencies in our M:N relationship. This was something that we could not quite figure out until getting further help from TA feedback. The changes implemented to the ERD now show correct M:N and 1:M relationships.

Upgrades to the Draft Version

- Removed a redundant line on ERD depicting an M:N relationship between Products and Orders
- Changed the cardinality of 1:M relationship between Orders—OrderDetails and Products—OrderDetails
- Added specific numerical data to show the level of sales in prior year to better explain the need for a DB solution (E.g 6500 orders in FY2023 with 1400 unique products)
- Added the purpose of Order and OrderDetails entities to database outline

Step 2 Edits

Based on the feedback we received, we made five changes to our project draft. This step did not include any TA feedback. Regarding peer feedback, we got six pieces of tangible feedback and suggested changes to our project. For one, we chose not to implement feedback from Peer Review #6 regarding centering the OrderDetails table. The reviewer has a certain design preference for this table since it's the intersection table. However, we feel like the schema still properly displays the entities in a clear and logical manner, especially when considering how the OrderDetails table has two 1:M relationships attached to it.

Review #5 pointed out a naming discrepancy between the ERD/Schema and the Database Outline for "status." We changed the Database Outline to lowercase status to match naming conventions and the ERD/Schema. Review #6 made an excellent point that the "ON DELETE CASCADE" in the "FOREIGN KEY (productID) REFERENCES Products(productID)" would incorrectly delete all OrderDetails that had a deleted product. To fix this, we removed the "ON DELETE CASCADE" from this time. In addition, Review #6 found a mistake in the Sample Data math for Order #1003. This was easily fixed by changing the products in the order. Review #8 pointed out that naming the ERD and Schema as "New" and "Old" did not make sense. We changed the names to their proper names.

Several reviews mentioned the lack of comments. While we did not believe the code needed at comments at this point, we do agree with reviewers that adding them will help with ensuring easier maintenance and readability of the code. We decided to add them from this step forward.

Upgrades to the Draft Version

- Changed "Status" in database outline to "status" to match up with ERD and Schema naming convention
- Deleted "ON DELETE CASCADE" from 'FOREIGN KEY (productID) REFERENCES Products(productID)' to ensure that the deletion of a Product did not delete all OrderDetails with the product
- Changed the order details for Order #1003 to make the math match up correctly with the product prices

- Adding comments to the code to explain any difficult pieces of code and ensure easier readability and maintenance of the code
- Changed the naming of ERD and Schema from “New” and “Old” to their proper names

Step 3 Edits (From Draft Reviews)

Based on the feedback we received, we made eight changes to our draft version. All the reviews (TA and Peers) indicated that we did not have any delete buttons for our pages. Thus, we added a delete button for all the entities. Most of the reviews also indicated that we did not have update buttons either, which we added for all the entities.

Moving on, a majority of the edits were centered around the Order Details page. This is because we did not submit an order details page for our draft submission according to our intended vision. The original page did not have any links to individual customer orders; It only had an option to update the amount of products total that were in all the existing orders on the website. After a drastic overhaul, the Order Details page is now updated and adheres to most of the peer suggestions. For example, review 9 suggested that the Order ID should be explicitly shown in the UI. We have now shown the Order ID within the Order Details page, as we actually need it to separate orders within customers.

Review 9 also proposed that we add a filter on the products page to search for products based on price range, category, etc. Due to the large scope of our project, we agree with this suggestion, as we now have a filter to search for products on the page. When we will eventually have hundreds of products, this option will be beneficial. Reviews 9 and 10 also recommended that we add more fields to the orders page to add attributes such as ‘price’, ‘date’, and ‘status’. However, with the now revamped Order Details page, it is better now to actually have fewer attributes needed to create a new order. Now, all we need is a customer’s name and the date to add a new order. The rest (such as status and price), can be either modified directly within the row after the order is created or modified through editing the order details on the Order Details page. Review 9 also had a misunderstanding, thinking that this interface is customer-faced. This interface is actually for company use geared toward the employees of the Racket Supply shop.

Moving on, review 10 suggested adding a search option for customers on the Customers page due to the large scope. We have added this functionality as now employees can search for customers by either name or email. Reviews 10 and 11 suggested adding a navigation bar so that users can easily navigate between pages of our interface. Technically, we already had a nav bar, but it was not consistent across all the pages. Now, we have included a nav bar that has links to all the pages for every single page. Finally, review 12 suggested adding borders to all the tables.

This makes sense as it allows for a cleaner feel and provides more organization to the website. Thus, we added borders to all the tables on the website, where necessary.

Upgrades to the Draft Version

- Added delete buttons for all entities to enable easy removal of records as per feedback.
- Included update buttons for all entities, addressing feedback about missing options for modifying records.
- Displayed the Order ID explicitly on the Order Details page, improving order separation and clarity in the UI.
- Added filter options on the Products page, allowing users to search for products by price range, category and more for better searchability and usability.
- Streamlined fields on the Orders page to only require ‘customer name’ and ‘date’ when adding new orders. Additional fields (e.g., status and price) are now editable post-creation directly within the Order Details page.
- Implemented search functionality on the Customers page, allowing employees to quickly locate customers by name or email due to the scale of the dataset.
- Standardized the navigation bar across all pages, making it easier for users to navigate consistently throughout the site.
- Added borders to tables across the website, improving visual organization and to create a cleaner, more structured user interface.

Project Outline and Database Outline - Updated Version:

Overview

Following the pandemic, tennis has skyrocketed in popularity across Los Angeles, drawing a new wave of enthusiastic hobbyists eager to take up the sport. The Terrible Tennis Team's Racket Supply, the leading supplier of tennis-related goods and equipment, has experienced a dramatic increase in customer demand, with sales soaring by 50% in 2023 alone to 6500 total orders with 1400 unique products offered. Despite this overwhelming growth, the existing manual sales tracking system has become inadequate, leading to inventory issues, delayed order processing, and a dip in customer satisfaction.

In an effort to address these challenges, Racket Supply is planning to implement a new database-driven website. The system is designed to manage inventory over 2,500 unique products, and process more than 10,000 orders annually. Integrating this backend system will ensure accurate logging of all customer orders, enhance operational efficiency, and increase customer service. T-Squad's ultimate goal is to meet sales goals, increase customer satisfaction, ensure accurate inventory management and error-free order processing.

Database Outline, in Words

Customers: Provides the essential details of customers that purchase from our store

- customerID: int, not NULL, PK
- firstName: varchar(50), not NULL
- emailAddress: varchar(100), unique, not NULL
- phoneNumber: varchar(15), NULL
- Relationship: 1 to Many relationship between Customers and Orders using customerID as a FK within Orders

Products: Provides the essential details of products that are available for sale in our store

- productID: int, not NULL, PK
- productName: varchar(100), not NULL
- priceValue: decimal(10,2), not NULL
- categoryItem: varchar(50), not NULL
- quantityLeft: int, not NULL
- Relationship: 1 to Many relationship between Products and Order Details using productID as a FK within Order Details

Orders: Provides the essential details of customer orders

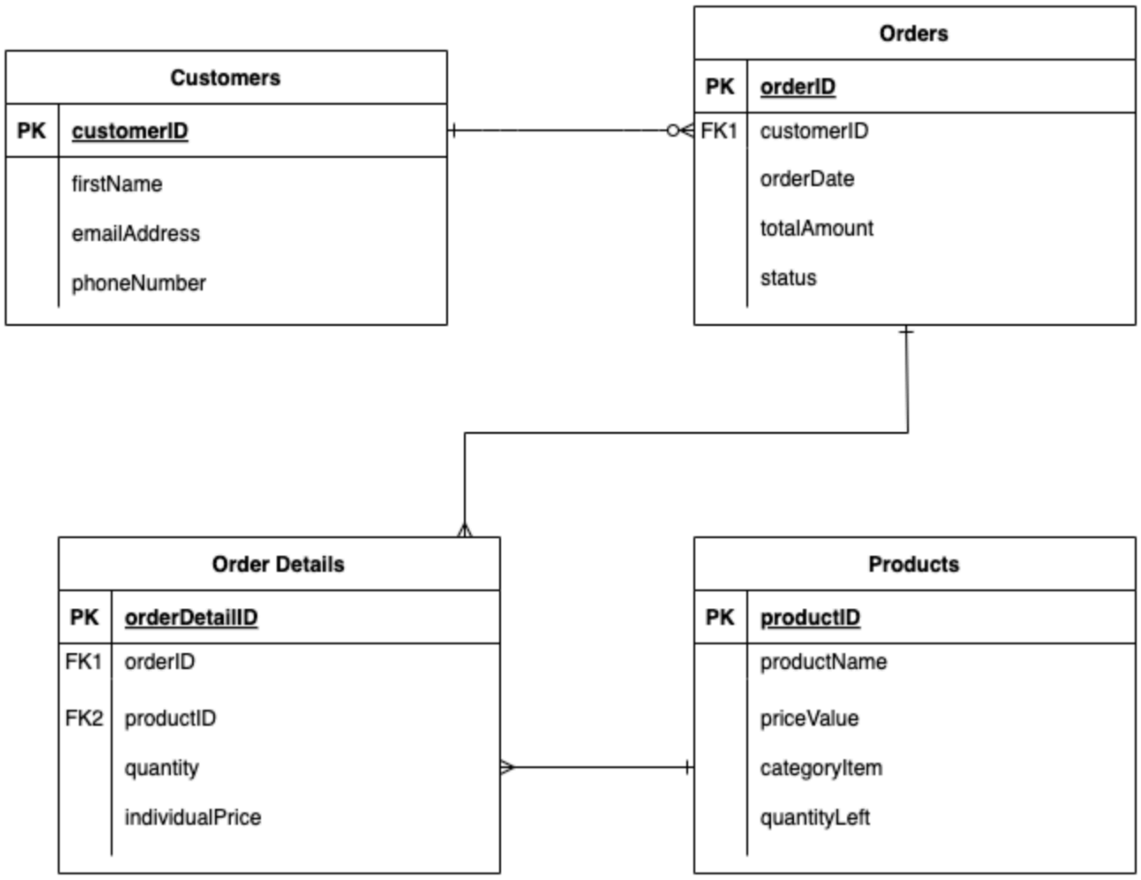
- orderID: int, unique, not NULL, PK
- customerID: int, not NULL, FK // reference Customers
- orderDate: datetime, not NULL
- totalAmount: decimal(10,2), not NULL
- status: varchar(50), not NULL // “Pending” “Completed” “Canceled”
- Relationship: 1 to Many relationship between Orders and Order Details using orderID as a FK within Order Details

Order Details: Stores the details of individual items in customer orders

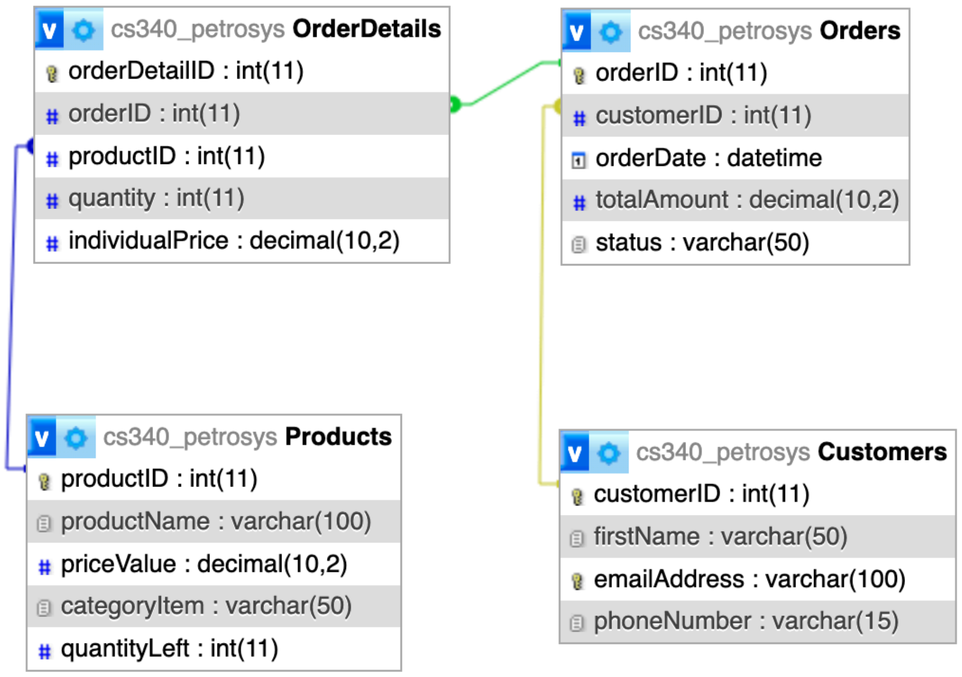
- orderDetailID: int, unique, not NULL, PK
- orderID: int, not NULL, FK // reference Orders
- productID: int, not NULL, FK // reference Products
- quantity: int, not NULL
- individualPrice: decimal(10,2) not NULL // Price of each item at sell date
- Relationship:
  - 1 to Many relationship between Orders and Order Details via orderID
  - 1 to Many relationship between Products and Order Details via productID
  - Facilitates Many to Many relationship between Products and Orders through Order Details

Entity-Relationship Diagram + Schema

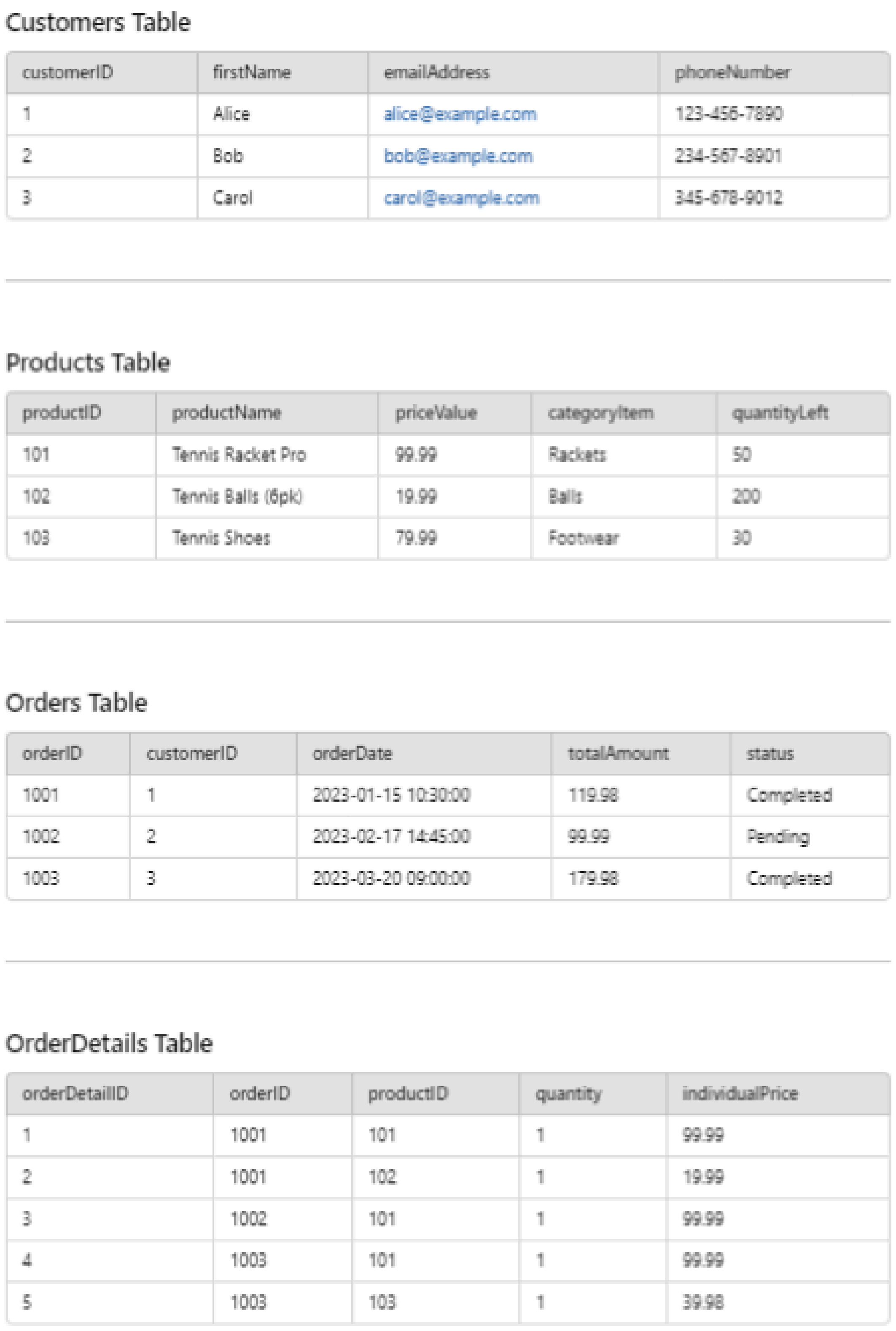
ERD



Schema


cs340_petrosys **OrderDetails**
orderDetailID : int(11)
orderID : int(11)
productID : int(11)
quantity : int(11)
individualPrice : decimal(10,2)
cs340_petrosys **Orders**
orderID : int(11)
customerID : int(11)
orderDate : datetime
totalAmount : decimal(10,2)
status : varchar(50)
cs340_petrosys **Products**
productID : int(11)
productName : varchar(100)
priceValue : decimal(10,2)
categoryItem : varchar(50)
quantityLeft : int(11)
cs340_petrosys **Customers**
customerID : int(11)
firstName : varchar(50)
emailAddress : varchar(100)
phoneNumber : varchar(15)

Sample Data



Customers Table

customerID	firstName	emailAddress	phoneNumber
1	Alice	alice@example.com	123-456-7890
2	Bob	bob@example.com	234-567-8901
3	Carol	carol@example.com	345-678-9012

Products Table

productID	productName	priceValue	categoryItem	quantityLeft
101	Tennis Racket Pro	99.99	Rackets	50
102	Tennis Balls (6pk)	19.99	Balls	200
103	Tennis Shoes	79.99	Footwear	30

Orders Table

orderID	customerID	orderDate	totalAmount	status
1001	1	2023-01-15 10:30:00	119.98	Completed
1002	2	2023-02-17 14:45:00	99.99	Pending
1003	3	2023-03-20 09:00:00	179.98	Completed

OrderDetails Table

orderDetailID	orderID	productID	quantity	individualPrice
1	1001	101	1	99.99
2	1001	102	1	19.99
3	1002	101	1	99.99
4	1003	101	1	99.99
5	1003	103	1	39.98