Page URL:

To see our HTML page: click here

Feedback by the peer reviewer/ Actions based on the feedback Step 1 draft:

1. Review form Zhi Liang:

   We incorporated his suggestions by merging the “Shipments” and “Orders” tables into a single table called "Orders." Additionally, we removed the "not null" constraint from other attributes, retaining only the constraints for primary keys (PK) and foreign keys (FK). However, we decided against his suggestion to create an item that could belong to multiple categories, because when an item is limited to a single category, it promotes consistency and standardization in the categorization process. It ensures that each item is accurately assigned to a specific category, reducing ambiguity and potential errors.
2. Review from Gunnar Egge:

   We took his advice and have implemented changes to improve the readability of our Entity-Relationship Diagram (ERD) for users. One specific improvement we made was altering the placement of entities within the ERD, ensuring that the relationships are more easily discernible at a glance. While the initial layout was good, we acknowledged the need to enhance the clarity of the connections by adjusting the positioning of entities and lines. This adjustment prevents line overlap, reducing the need for users to double-check and ensuring a smoother and more intuitive visual representation of the relationships. We value his valuable input and are committed to continuously refining our design to optimize the user experience.

3. Review from Charlie Chen:

   Charlie Chen suggested that we maintain greater consistency in our document by aligning our naming conventions. We took his suggestion and made the necessary changes to adopt a camel case for all our names.

4. Review from Robert Long and Samantha Egge

   The main suggestion from these two is that our idea was not specific enough and therefore not applicable to the assignment scope. We looked into this with the professor and were informed that our idea is acceptable. It was still useful to look into as it would have been very unfortunate to mess up the first assignment. Samantha and Robert both pointed out that our ERD and written description were inconsistent with one another, so we made the change to reflect this suggestion being fixed. Robert suggested that we no longer need to maintain separate tables for "Shipments" and "Orders" since we have already merged them into a single table called "Orders." Additionally, he pointed out that our table layout was difficult for readers to comprehend. Taking his advice into consideration, we made the changes to address these issues. Samantha also brought up a concern regarding the M: N relationship between orders and items. While our documents stated that orders had an M: N relationship with shipments, our ERD showed a 1:M relationship. Moreover, the placement of the reviews/item relationship in our ERD seemed to be reversed due to the consolidation of the "Shipments" and "Orders" tables. As a result, we have successfully resolved these conflicts by merging tables and changing our relationship.

Feedback by the peer reviewer/ Actions based on the feedback Step 2 draft:

1. Review form Evan Hock:
   - Description of issues:

     There is only one issue that Evan has dressed as an issue he couldn't find a correspondence between the ER diagram's Categories.numCategories attribute and anything else in the document.
   - Actions to solve the issues:
     We deleted the numCategories in the ERD because it was a typo when we make the ERD.

2. Review from Charlie Chen:
   - Description of issues:
     Charlie pointed out that our DDL.sql doesn't have a CASCADE operation defined for any foreign key constraints, which might help by doing the following ALTER TABLE Orders ADD CONSTRAINT Orders_ibfk_1 FOREIGN KEY (customerID) REFERENCES Customers (customerID) ON

DELETE CASCADE and sample data shown in the PDF is not full inserted into the database.

- Actions to solve the issues:

  We added the on delete cascade and fully implemented our sample data in the table

3. Review from Samantha Egge
   - Description of issues:
     She didn't see any CASCADE operations present and for the sample data there customerOrders doesn't have any example data, and the items table has all of the reviewID attributes set to NULL.
   - Actions to solve the issues:
     We added on delete cascade to our FK and implemented the example data in our database and also fixed reviewID to not null.

Review from Robert Long:

- Description of issues: he also pointed out that our DDL.sql file doesn't have any cascade operations
- Actions to solve the issues: already fixed it.

Grading Feedback from Instructor:

- Description of issues:

  I don't believe I was asked about your outline as page 5 indicates (but please correct me if I'm wrong). I *do* think the outline should be for a specific use-case (either a particular shop or kind of shop). This will help guide your design somewhat. Please update this for future revisions. The current review lacks motivation for including "Reviews" and "Categories". How many reviews can a user write, for instance. And can items only belong to one Category?
  ERD and Outline aren't consistent. For example, Customers does not have postalCode (in ERD) and Items has "stockAmount" instead of "numItems".Items FK to Categories should point to CategoryID, but it seems to be a varchar(50) and not an INT. The current design doesn't imply that a single Order row can be associated with multiple items. We

would need an intersection table (composite entity) between Orders and Items to achieve this.
Also, consider what your primary keys should be for each relation. Making a distinct primary key (e.g., reviewID) has a very different effect on row participation than using the FKs as the foreign keys.

Remember to put Summary of Changes (reviews and how you acted on them) at the top

- Actions to solve the issues:
  We decided to design a database for an Onlie electronic retailer called OregonTech and rewrote the project outline, we also changed the relationship between the table Customers and Reviews, now a customer can make 0 and many reviews, and a review can be written by one and only one customer. Yes, we have three categories(Phones,iPads, Laptops) in our database and each item has one and only one category to which it belongs. We deleted postalCode from our outline and ERD. Also changed the stockAmount to numItems in the table Items. An FK(categoryID) is added to the Items table and we confirmed all ID in our ERD are int, we also added a composite table between Items and Categories. We also changed the order of the PDF.

Feedback by the peer reviewer/ Actions based on the feedback Step 3 draft:

1.) Review from Gunnar Egge:

Description of issues: Gunnar suggested that we are missing a NULLable relationship.
Actions to solve issues: Items in the orders table will be set to be able to be NULLable.

2.) Samantha Egge:

Description of issues: Samantha suggested that we add a filter at some place in our website.

Actions to solve issues: Dropdown menu on reviews page that shows certain reviews based on the review rating.

3.) Robert Long:

Description of issues: Robert suggested that we should have anonymous reviews.
Actions to solve issues: We will make customerID in the reviews table nullable.

4.) Charlie Chen:

Description of issues: Charlie suggested that we add a filter at some place in our website.
Actions to solve issues: For each page we will add a search bar at the top that will filter by a logical name for each page. Example - items will filter by item name, customers will filter by customer first and last name.

5.) Evan Hock:

Description of issues: UI is stated as being jarring in the transition from the homepage to other pages in the website.
Actions to solve issues: Will add the same header and footer in the pages for each table to add a sense of familiarity while also smoothing the transition.

Project Outline and Database Outline - Updated Version:

Team member names and the project title

Team members: Ya Zou, Maxwell Zimmer.

Project title: E-commerce

Overview:

OregonElectronics is a fast-growing electronic online shop specializing in selling iPads, phones, and laptops. With an impressive annual sales revenue of $ 12 million, the client requires an efficient database-driven website with a relational database backend to handle their high volume of sales and product inventory. The system's main focus will be on recording Sales Orders for iPads, phones, and laptops to Customers, ensuring real-time tracking of inventory levels, and providing accurate stock information to both customers and the management team. The website must support a seamless and user-friendly shopping experience, allowing customers to easily browse through a wide range of electronic products, make purchases, and track their orders. As a result, the database system must be designed to handle a substantial number of daily transactions, ensure data integrity, and offer exceptional performance to cater to the demands of OregonElectronics' thriving online retail business.

Database Outline:

1. **Customers:** Record the details of customers who do business with the online electronic shop.
   Attributes:
   - customerID: int, auto_increment, NOT NULL, PK
   - firstName: varchar(50)
   - lastName: varchar(50)
   - emailAddress: varchar(50)
   - address: varchar(50)
   - city: varchar(50)

2. **Items:** Represents the electronic products available for sale on the online shop.

   Attributes:
   - itemID: PK, auto_increment. Int, unique
   - reviewID FK2, int, NOT NULL
   - categoryID FK3,int
   - itemName varchar(50)
   - price int (12)

3. **Orders:** tracks details of customer orders placed on the online shop
   Attributes:
   - orderID: PK, int, auto-incement, NOT NULL
   - customerID: FK1 ->customer table, not NULL
   - orderDate: date
   - creditCardNumb int(12)
   - creditCardExpDate date
   - numOrderedItmes int(12)
   - pricePaid int

4. **Categories:** Divide electronic products into different categories for sale on the online shop.
   Attributes:
   - categoryID int, auto_increment, unique, not NULL, PK
   - categoryName varchar(50)
   - Relationship: M: 1 between categories and items.

5. **Reviews:** Records customer reviews and ratings for products
   Attributes:
   - reviewID: PK, int, auto-increment, not NULL.
   - customerID: int not NULL, FK.
   - overallRating: decimal(3,2), not NULL
   - feedback: varchar(200).

6. **ordersItems:** composition table, It keeps tracking our inventory
   Attributes:
   - orderID: int not NULL, FK,PK
   - itemID: int NOT NULL ,FK,PK
   - totalNumItems: int

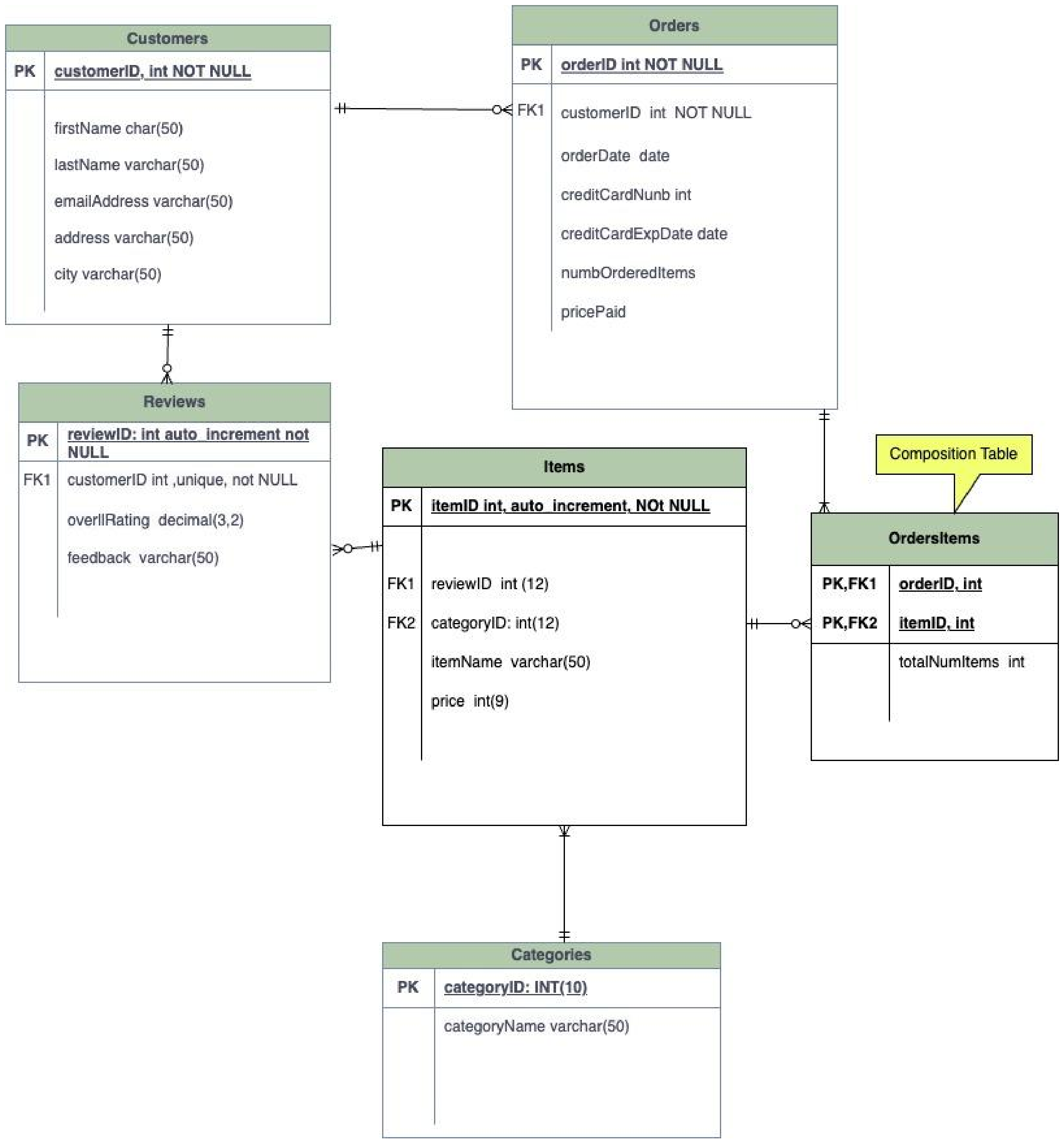
Relationship

Left Entity	**Cardinality**	**Right Entity**	**Notes**
Customers	1:M	Orders	One to zero or many
Customers	1:M	Reviews	One to zero or many
Reviews	M:1	Items	Zero or many to One
orders	M: N	Items	This is our Many to Many relations
Items	M: 1	Categories	
items	1:M	OrdersItems	Composite table (OrderItems)
OrderItems	M:1	Orders	Composite table (OrderItems)

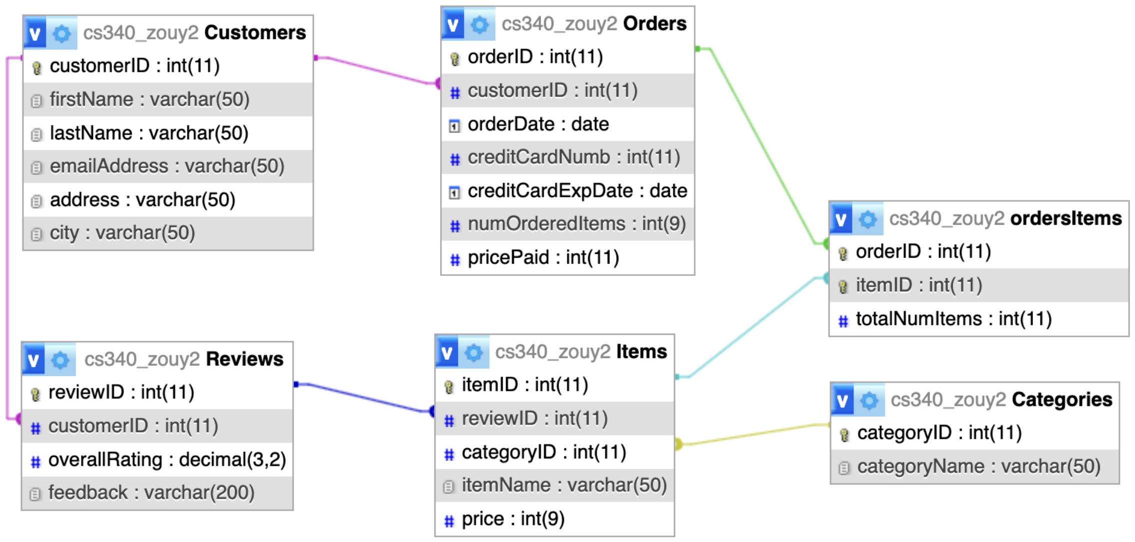
Normalization:

Our tables are normalized to improve the overall efficiency and integrity of our database. This is achieved by reducing data redundancy, ensuring atomic values in each column, establishing unique identifiers with primary keys, eliminating partial and transitive dependencies, and adhering to the principles of various normal forms, such as 1NF, 2NF, 3NF, and BCNF. Therefore, Our entities are normalized.

c) Entity-Relationship Diagram- Updated Version:



d) Schema


cs340_zouy2 Customers
customerID : int(11)
firstName : varchar(50)
lastName : varchar(50)
emailAddress : varchar(50)
address : varchar(50)
city : varchar(50)
cs340_zouy2 Orders
orderID : int(11)
customerID : int(11)
orderDate : date
creditCardNumb : int(11)
creditCardExpDate : date
numOrderedItems : int(9)
pricePaid : int(11)
cs340_zouy2 ordersItems
orderID : int(11)
itemID : int(11)
totalNumItems : int(11)
cs340_zouy2 Reviews
reviewID : int(11)
customerID : int(11)
overallRating : decimal(3,2)
feedback : varchar(200)
cs340_zouy2 Items
itemID : int(11)
reviewID : int(11)
categoryID : int(11)
itemName : varchar(50)
price : int(9)
cs340_zouy2 Categories
categoryID : int(11)
categoryName : varchar(50)

e) Example Data

Customers					
customerID(PK)	firstName	lastName	emailAddress	adress	city
auto_increment	Jack	Spring	JackSpring23@hello.com	123NW lane dr	Corvillas
auto_increment	Alice	Summer	AliceSummer@hello.com	sw In dr 1256	Salem
auto_increment	Iris	Winter	IrisWinter@hello.com	420 Database C	Portland
auto_increment = 1000					

Items				
ItemID (PK)	categoryID (FK1)	reviewID(FK2)	itemName	price
auto_increment	used select	used select	phone	729
auto_increment	used select	used select	ipad	629
auto_increment	used select	used select	laptop	999
auto_incremnt = 30				

Categories	
categoryID(PK)	categoryName
auto_increment	Iphones
auto_increment	Ipads
auto_increment	Laptops
auto_increment = 500	

Reviews		
reviewID(PK)	overallRating	feedback
auto_increment	4.8	excellent
auto_increment	1.3	bad
auto_increment =100	3.3	good

Orders						
OrderID (PK)	customerID(FK1)	orderDate	creditCardNumb	creditCardExpDa	numOrederedIte	pricePaid
401	used select to find customerID	2-2-2019	1234567	2-26-2021	2	1628
401	used select to find customerID	2-2-2019	1234567	2-26-2021	2	1628
402	used select to find customerID	10-12-2018	1234568	4-13-2023	1	629
403	used select to find customerID	2-15-2017	1234569	9-7-2021	1	729
auto_increment = 10000						

Composition_table		
ordersItems		
orderID(PK,FK)	itemID(FK,PK)	totalNumItems
used select	used slect	2
used select	used slect	2
used select	used slect	1
used select	used slect	1