Project Proposal and Outline - Final Version CS 340 – Oregon State University
Kaitlyn Coyle and Chris Jacobs Winter 2022

URL: https://web.engr.oregonstate.edu/~jacobsc2/

Peer Feedback (2/17/2022)

Reviewer #1 Feedback
"Orders import a corresponding fk attributes when being created. I believe these need some way to be updated however, i.e. what products and what payment method."

Yes, updates are present for the entities, however, to reiterate, the order's would need a way to change their payment method or what products are being ordered.

Reviewer #2
"All entities have a page and each page offers a search or select function. The only issue I see is there is no method to select for some of the relationship tables such as OrderItems or ShipmentItems."

Not all Insert/Create methods have this functionality. For example in the Pay Methods page there is only one field for inserting a pay method. It is not clear for example how the foreign key address_id would be set during the create method.

"For the addresses page, one question I have is how does the update process work? If an address is entered to the field to update, how does the system then find the original address row, then make updates to that row? Or is the update address field populated by the search for address function?"

"For some of the search items I would add some additional fields to search/filter by. For example on addresses it would be nice to be able to search by zipcode or state."

Reviewer #3
"Most of the schema tables have a select function. I can see having a select function for every single table would provide superfluous information considering that performing a search in some of the entities would show the results of the intersection tables.

However, since some of the intersection tables will require user input, there should be a way to select and update the information in those tables."

No, the intersection tables and the transaction tables appear to be missing a way to insert. I think there should be a way to add to an order since it is entirely possible to miss an item and need to add it back later.

I think this is missing. I don't see a way to insert into the intersection table between orders and products. This is necessary since you have the qty as a user input for the orderitems.

There are options to delete entries from various tables but it's not clear how it's done. I would expect there to be a way to include the ID number for whichever entity you wanted to delete from.

There are some update functionalities but I'm not sure how they work. For example, the update address includes a text field but I don't see a way to designate which address you want to update. I would expect there to be a way to select the address ID and provide the new address to perform the update.

It appears that multiple tables have the option to have null values. For example, the customer information does not look required except for the customer ID #.

My recommendation would be to think of how you can designate which entries you want to delete and update. Right now it seems like there is just a standalone delete button with no information on which entry it's supposed to delete. The same goes for the updates. There is an option to update for some of the tables but it doesn't specify which entry is being updated. A good way to do this is to tie the ID of the entry you want to update to the new information. Another suggestion is to have the entities ID numbers auto increment. I noticed that some of them have ID entries but this opens the door to have duplicate IDs or having to guess until you find a unique one.

Reviewer #4

- *The Pay Methods page seems to be missing. Pay Method entity has four attributes in their outline. However, the 'create a new payment method' section has only one input box.*
- *The product price page's title changed item prices. Also, there is naming inconsistency in that page like 'Select Item Prices' and 'Set New Product Price'.*

Revisions 2/17/2022

We have made no updates to our project overview/ERD/schema.

With respect to our peer feedback and our own edits, we have made the following revisions:

1. Added three pages, customer-pay-methods.html, customer-addresses.html and order-items.html to allow for selection and insertion into those intersection tables.

2. We were going to make a revision to add a shipment-items.html page so that we could include that M:M table, but decided to consolidate shipments.html and shipment-items.html into a single page for ease of use.

3. Insert functionality in the tables mentioned above will occur once we write the appropriate JavaScript.

4. Addresses.html: has been updated to specify that an AddressID is needed for SELECT/UPDATE/DELETE functions.
5. Addresses.html has been updated to require a valid customer ID in order to create an address.

6. Orders.html: has been updated to…
   a. allow for “order by” functions when searching for an order by status/customerID (you can order by most recent, item count etc.)
   b. orders that are found now have buttons so that shipments/shipment-items can be added to them and order-items can be added/edited. A “cancel order” button has also been added into the form table.

7. Order-Items.html: allows for viewing the item that are part of an order, as well as providing functionality to add/edit/and remove items from the order.

8. Shipments & Shipment Items: have been consolidated into a single page (shipments.html). This page allows the user to view all shipments created for an order, and allows for the viewing/adding/editing of items within individual shipments.

9. Products: CSS has been updated. “Edit Product” section has been taken out, and we now have an edit button for each selected row that will allow via javascript the

ability to edit the product. We have also added a button to discontinue a product (we do not delete products, in order to ensure report integrity)

10. Product Prices has been updated to allow for the addition of a new price. We are still not allowing editing or deletion of price histories of items, only insertions and selects.

11. Transactions has not been updated. Transactions can't be modified or deleted. They also can't be created manually, they only occur when an order is successfully processed.
12. Pay Methods has been updated to include additional fields required by our schema. It now also requires a valid customer ID. Pay Methods asks for address information. When addresses info is inserted here, a new row in Addresses is created - this returns an addressID, and the PayMethod is created using that addressID as a foreign key. PaymentMethod search has also been updated to clarify that a PaymentID is required for the search.

13. Customer Pay Methods: this page was added so that we can view the M:M relationship between Customers and Pay Methods. It allows for deletion and editing of the pay methods. The edit fields will be generated with JS when the Edit button is pushed. However, we are unsure whether we want to allow pay methods to be edited or deleted, rather we think they might just need to be archived, because a Pay Method that is associated with a Transaction may cause reporting errors.

Revisions: 2/10/2022

We have received no grader feedback (we’ve actually never received grader feedback throughout the entire course) and no peer feedback since 2/6.

Revisions we made on our own:

1. In order to achieve a little more simplicity, we removed the email and phone number tables. We updated our ERD and Schema diagrams to reflect this.

Peer Feedback (2/6/2022)

“After reading over all the details I did notice as many others had already the varchar for the item_id when it might be more work to have it that was since they'd be sorting by the characters rather than sorting it more easily by integer. If the item ids are to also be automatically produced then having it be a randomized integer will be more beneficial. all the other datatypes are clearly depicted.”

“While I don't see a huge issue with some of the plural/non-plural use of naming conventions, I would say for the sake of programming or just the outline in general you need to keep things named fairly the same otherwise it could cause difficulty for fellow coders or anybody that is to observe your code.”

“Naming is not consistent between overview and outline entities. Entities should be plural ex: (change Order to Orders in outline). All attributes are singular. The Item_id attribute for the items entity is capitalized while the others aren't.”

“For the most part, this is done well. The Item PK however is a varchar when it should be an int if planning to use auto-increment.”

“The overview is not consistent with the entities at times. For example, "Customer" in the overview does not match the "Customers" entity and "Payments" in the overview does not match the "Payment" entity. Attributes are all singular, while some entities are singular and some are plural (it's suggested to make all the entities plural). Entities are correctly capitalized and attributes use snakecase and for the most part are not capitalized, one exception being the "Item_id" attribute for the "Item" entity.”

“The entities are explained and their attributes and relationships listed. I did notice some non ideal data types for certain attributes. For example, under the "Item" entity we have the "Item_id" attribute, which is listed as "varchar", or a string of characters. I'm assuming the item IDs are numerical, so it would be better to use an "int" data type. If the item ID is actually a combination of numbers and letters, then I'm not sure the database can "auto_increment" such a value.”

Revisions (2/6/2022)

1. (Peer) We fixed type errors that were pointed out in the feedback we received.

2. (Peer) We fixed the discrepancies in the naming of Attributes and Entities (plural vs singular) and fixed various discrepancies in capitalization. We also updated our naming schema for attributes to be more succinct - attribute names no longer prefixed with table names and foreign keys are not longer prefixed with “fk_”

3. We separated customer phone numbers and customer email addresses into separate entities (CustomerPhones and CustomerEmails)

4. We got rid of the Payments entity and replaced it with an entity named Transactions. We also added a PayMethods entity, which has a M:M relationship with Customers.

5. We added an Addresses entity and associated it with Customers, PayMethods, and Shipments.

6. We added the tables CustomerAddresses and CustomerPayMethods to allow for our many-to-many relationships between Addresses/PayMethods and Customers.

7. We associated Shipments with Products instead of just with Orders. This is because we wanted the ability for an Order to have separate Shipments, which means that a Shipment will need to be associated with specific Products that are part of that Shipment. Therefore, we also added a ShipmentItems table that tracks individual items within a Shipment.

8. We changed the name of the Items entity to “Products”.

9. We added a ProductPrices entity so that product prices could be tracked over time and not just changed and lost. Without ProductPrices, if a Product price were changed and a closed Order tried to reference the original sale price, we would have a reporting discrepancy because we would only know the new price of the Product, not the price at the time of the Order.

10. We got rid of the composite keys we were using because they were being used incorrectly (composite keys are no longer used). We also fixed how we were using foreign keys in the weak entities like ShipmentItems and OrderItems.

11. We added various attributes in Shipments to track weight, volume, and cost. We added attributes in Products to track name and color and changed the way we were tracking volume (switched to inches cubed).

12. The price of the order is no longer stored in Order, it is only stored within Transaction. Likewise, sub_total is no longer stored in OrderItems and total_weight and total_volume have also been removed from OrderItems. We did this because we now think we will be able to aggregate this type of data using SELECT and JOIN and store the totals in a Transaction and a Shipment.

13. Most of these revisions were done in an attempt to achieve third normal form. We are still validating whether 3NF has been achieved and may continue updating entity relationships in order to achieve 3NF and eventually BCNF.

Overview

CookNook, a small business that sells printed cookbooks, currently sells around 100 books per month. Up until now, low sales volume has allowed CookNook to use Etsy as their online storefront. However, due to an increase in sales volume, CookNook is interested in developing a self-hosted online storefront with a database backend that can handle a minimum of 1000 orders per month. Additionally, because of the increase in sales, CookNook wants to expand their product line into kitchen hardware.

For their database, CookNook needs to track **Customer** account information and online **Orders** that the customer makes, as well as the **Payments** used by Customers. CookNook also needs to track **Items** in their inventory as well as those that are part of **Orders** (**OrderItems**). CookNook ships products themselves, so they need to track **Shipments** related to orders.

Database Outline

primary key** ***foreign key* *****foreign key and member of candidate key

<u>Customers</u>

Represents an individual customer.

Attributes

***customer_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
first_name	VARCHAR			NOT NULL
last_name	VARCHAR,			NOT NULL
date_of_birth	DATE			NOT NULL

Relationships

- **PayMethods**: Customers may have many PayMethods, PayMethods may be used by many Customers (M:M)
- **Addresses**: Customers may have many Addresses, Addresses have many Customers (M:M)

- **Orders**: Customers may have many Orders, Orders must have only one Customer (0:M)

Addresses

Represents a customer’s address that can be used for shipping and/or billing purposes.

Attributes

***address_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
first_name	VARCHAR			NOT NULL
last_name	VARCHAR			NOT NULL
address1	VARCHAR			NOT NULL
address2	VARCHAR			
city	VARCHAR			NOT NULL
state	VARCHAR			NOT NULL
zip_code	VARCHAR			NOT NULL
country	VARCHAR			NOT NULL

Relationships

- **PayMethods**: An Address may belong to a PayMethod, a PayMethod must have a single Address (0:M)

- **Shipments**: An Address may belong to a Shipment and a Shipment must have a single Address (0:M)

- **Customers**: An Address may have many Customers and a Customer may have many Addresses (M:M)

PayMethods

Represents payment methods used by customers for transactions.

Attributes

***payment_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
***address_id*	INT			NOT NULL
card_type	VARCHAR			NOT NULL
last_four_digits	INT			NOT NULL
expiration_date	DATE			NOT NULL

Relationships

- **Transactions**: PayMethods are used in at least one Transaction, a Transaction uses a single PayMethod (1:M)

- **Customers**: A PayMethod is used by at least one Customer, a Customer may have many PayMethods (M:M)

- **Addresses**: A PayMethod must have only one Address, an Address may be used with many PathMethod (0:M)

<u>Products</u>

Represents a product that a customer can purchase

Attributes

***product_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
name	VARCHAR			NOT NULL
color	VARCHAR			NOT NULL
weigh_lbs	DECIMAL			NOT NULL
volume_cubic_inch	DECIMAL			NOT NULL
in_stock_qty	INT			NOT NULL
reorder_at_qty	INT			
is_discontinued	TINYINT			NOT NULL

Relationships

- **Orders**: A Product may be associated with many Orders, an Order is associated with one or many Products (M:M)

- **Shipments**: A Product may be associated with many Shipments, a Shipment may be associated with many Products (M:M)

- **ProductPrices**: A Product is associated with at least one ProductPrice, a ProductPrice is associated with a single Product (1:M)

<u>ProductPrices</u>

Represents a history of prices for an individual product.

Attributes

***price_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
_**product_id_	INT			NOT NULL

price	DECIMAL			NOT NULL
date_active	DATETIME			NOT NULL
date_inactive	DATETIME			

Relationships

- **Products:** A ProductPrice must be related to a single Product, a Product can have many ProductPrices (1:M)

Orders

Represents a collection of items that are part of a customer’s order.

Attributes

***order_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
_**customer_id_	INT			NOT NULL
num_items	INT			NOT NULL
datetime_created	DATETIME			NOT NULL
datetime_processed	DATETIME			
is_closed	TINYINT			NOT NULL
when_closed	DATETIME			

Relationships

- **Customers**: An Order must have a single Customer, a Customer may have many Orders (0:M)

- **Products**: An Order must have at least one Products, a Product can be part of many Orders (M:M)

- **Transactions**: An order must have a single Transaction, a Transaction is part of a single Order (1:1)

- **Shipments**: A shipment can belong to only one Order, an Order can be split into zero or many Shipments. We say 0:M here because Shipments are created *after* an Order has been made - therefore an order can have no Shipments.

Shipments

Represents one of many shipments that belong to an order, sent to a customer.

Attributes

***shipement_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
***order_id*	INT			NOT NULL
***address_id*	INT			NOT NULL
datetime_created	DATETIME			NOT NULL
tracking_number	INT			NOT NULL
total_lbs	DECIMAL			NOT NULL
total_cubic_in	DECIMAL			NOT NULL
shipping_cost	DECIMAL			NOT NULL
datetime_ready	DATETIME			
datetime_shipped	DATETIME			
datetime_arrived	DATETIME			

Relationships

- Orders: A Shipment must belong to a single Order, an Order may have many Shipments (0:M)

- Addresses: A Shipment must have a single Address, an Address may belong to many Shipments (0:M)

- Products: A Shipment has at least one Product, a Product may be part of many Shipments (M:M)

Transactions

Represents an individual transaction i.e. a purchase of an order made by a customer.

Attributes

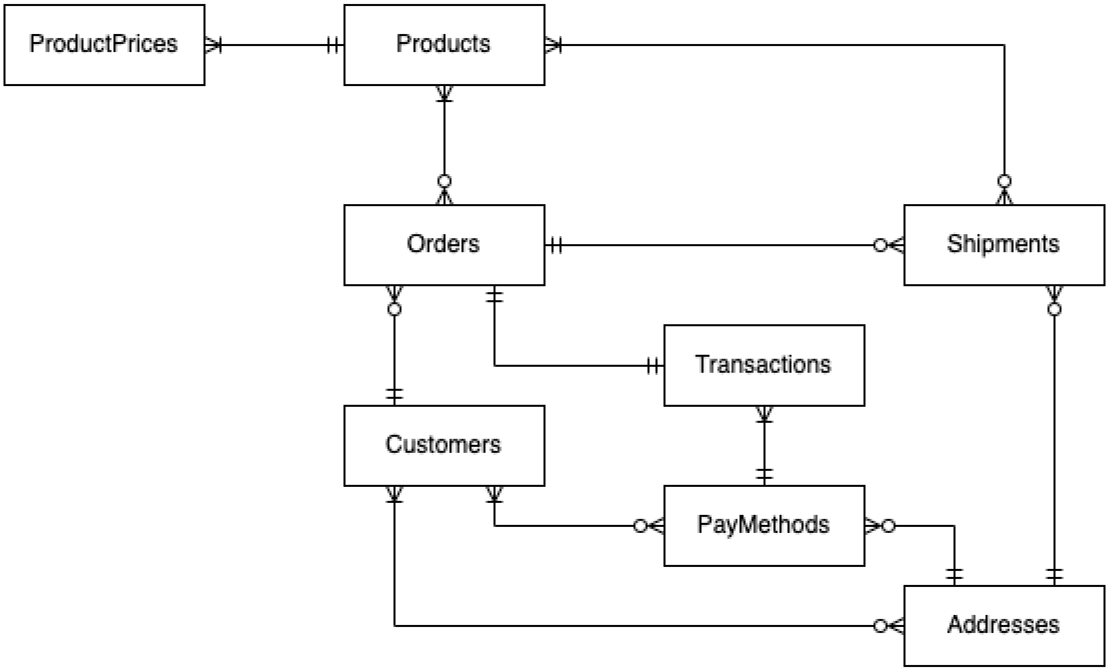
***transaction_id**	**INT**	**UNIQUE**	**AUTO_INC**	**NOT NULL**
****order_id**	**INT**	**UNIQUE**		**NOT NULL**
***payment_id*	INT			NOT NULL
transaction_amount	DECIMAL			NOT NULL
datetime_charged	DATETIME			NOT NULL
confirm_code	VARCHAR			NOT NULL
checksum	VARCHAR			NOT NULL
datetime_success	DATETIME			

Relationships

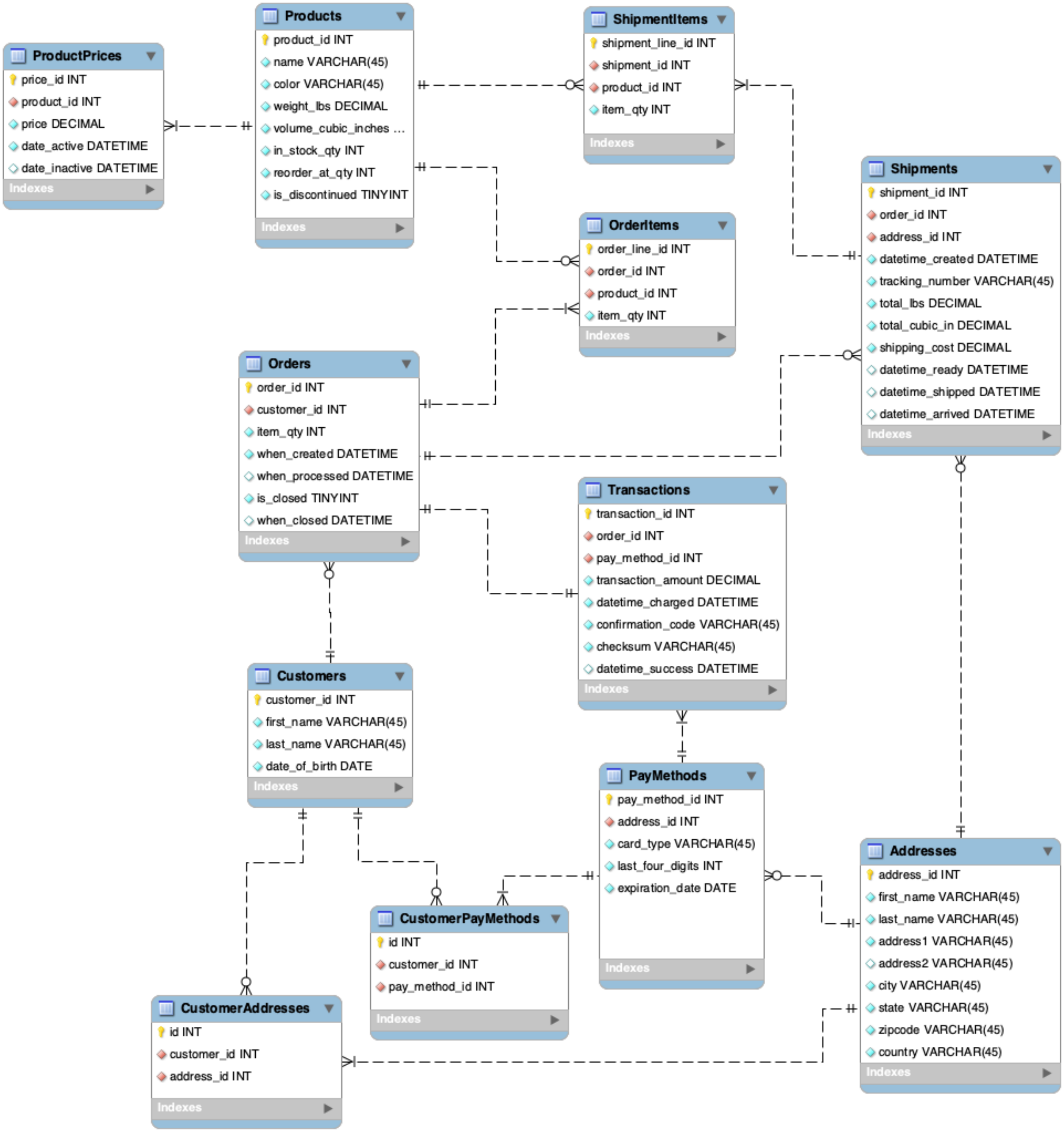
- **Orders**: A Transaction must be part of a single Order, an Order must have a single Transaction (1:1)

- **PayMethods**: A Transaction must have a single PayMethod, a PayMethod is used for at least one Transaction (1:M)

Revised ERD (2/10/2022)



Revised Schema (2/10/2022)


ProductPrices
price_id INT
product_id INT
price DECIMAL
date_active DATETIME
date_inactive DATETIME
Indexes
Products
product_id INT
name VARCHAR(45)
color VARCHAR(45)
weight_lbs DECIMAL
volume_cubic_inches …
in_stock_qty INT
reorder_at_qty INT
is_discontinued TINYINT
Indexes
ShipmentItems
shipment_line_id INT
shipment_id INT
product_id INT
item_qty INT
Indexes
Shipments
shipment_id INT
order_id INT
address_id INT
datetime_created DATETIME
tracking_number VARCHAR(45)
total_lbs DECIMAL
total_cubic_in DECIMAL
shipping_cost DECIMAL
datetime_ready DATETIME
datetime_shipped DATETIME
datetime_arrived DATETIME
Indexes
OrderItems
order_line_id INT
order_id INT
product_id INT
item_qty INT
Indexes
Orders
order_id INT
customer_id INT
item_qty INT
when_created DATETIME
when_processed DATETIME
is_closed TINYINT
when_closed DATETIME
Indexes
Transactions
transaction_id INT
order_id INT
pay_method_id INT
transaction_amount DECIMAL
datetime_charged DATETIME
confirmation_code VARCHAR(45)
checksum VARCHAR(45)
datetime_success DATETIME
Indexes
Customers
customer_id INT
first_name VARCHAR(45)
last_name VARCHAR(45)
date_of_birth DATE
Indexes
PayMethods
pay_method_id INT
address_id INT
card_type VARCHAR(45)
last_four_digits INT
expiration_date DATE
Indexes
CustomerPayMethods
id INT
customer_id INT
pay_method_id INT
Indexes
CustomerAddresses
id INT
customer_id INT
address_id INT
Indexes
Addresses
address_id INT
first_name VARCHAR(45)
last_name VARCHAR(45)
address1 VARCHAR(45)
address2 VARCHAR(45)
city VARCHAR(45)
state VARCHAR(45)
zipcode VARCHAR(45)
country VARCHAR(45)
Indexes