For Lab 2, you will design, create, and load data into a database that you have created. You will design and implement a simple database schema to capture data to support the business process described in Appendix A. You will then add records to your database to capture the specific business entities and events as described in Appendix B.

**Deliverables**: Your primary deliverable is a set of well-formed tables created in a SQLite database file that you will create on your personal laptop using Valentina Studio. Having created the database structure, you then need to load an appropriate set of records into those tables to capture the events and entities described in Appendices A and B. Your tables should contain an appropriate set of columns, use a consistent naming scheme for table and column names, and use appropriate data types for each table. Your database design should use primary keys and foreign keys to correctly implement the constraints and business rules described for the database in Appendix A.

In addition to a working database, you need to submit a PDF or Valentina Studio diagram file (Diagram > Print menu item, .pdf format file) to Canvas that contains the diagram for your database that you've created in Valentina.

Optionally, you may include a written document in PDF or MS-Word format of up to one page explaining or providing guidance regarding the database design decisions and/or location and details of the design. If you have a clear and clean design that properly captures all of the requirements and stores all of the requested data, this document will not be necessary. If you feel it is important to explain your decisions or provide guidance on why the design is set up the way that it is, you may do so in a brief document that we will use to guide our testing and grading of your database.

**Groups**: This lab may be completed individually or in groups of two students. If you are working as a group, you should work with the same lab partner with whom you completed the previous labs. If, for some reason, you are unable to work with the same partner for lab 2 please contact me (Prof. Monroe) to explain the situation and what your changed working situation will be.

**Grading criteria**:  I will use the attached rubric to grade your submission.

**Appendix A: business process description and database requirements**

Congratulations, you have been hired by Nile.com to help develop a database system to manage the processing of orders for their fast-growing e-commerce business. Nile.com sells a wide variety of goods through their online store and dispatches those orders from fulfillment centers using their fleet of delivery vehicles. The fulfillment centers serve both as warehouses to store items awaiting shipment and processing centers where employees and robots gather the items purchased in each order, package them, and dispatch the orders for delivery to homes and businesses.

Your manager at Nile.com has compiled the following specification for you regarding what the database needs to be able to support. This is primarily a description of business processes and business needs. A substantial portion of your task here is to translate these non-technical requirements into a flexible and efficient relational database design. The company organizes its broader order fulfillment process with the following sub-processes:

1. Order receipt, reservation of items, and process scheduling
2. Gathering order items and moving those items to the packing station
3. Placing items in packaging, confirming the correct items have been selected and that they are intact, sealing packaging and labeling for delivery
4. Delivering packages to the customer

**Process 1: Order receipt, reservation of items, and process scheduling**:

An order is created when a customer completes the check-out process and purchases the items in her cart on the Nile.com website. At this point, an order is generated and added to Nile.com's fulfillment system, where it is assigned an order id and the order's status is set to *Submitted* [note - in these process descriptions, status values are typically specified in *italics*]. The order specifies the set of items that have been purchased, the price and quantity to be charged for each of those items, the amount of tax to be collected, the amount to be charged for delivery, the customer who made the order, the address to which it is to be delivered, the date and time on which the order was submitted, and the promised order delivery date.

Once the order has been submitted, the fulfillment system assigns the order to the fulfillment center that is geographically closest to the delivery location. The geographic location of each fulfillment center is identified by its address and the precise latitude and longitude of its loading dock. Each fulfillment center is uniquely identified with an integer ID. To make sure that purchased items are not re-sold to another customer before they can be packaged and delivered, once an order is submitted, the system marks each item in the order as *Reserved* until it can be gathered and packaged for shipment. If the fulfillment center has more than one of the purchased items in inventory it needs to keep track of how many of those items are available for purchase and how many are reserved and awaiting shipment to customers. If an item in the order is not in inventory, that item's status is set to *Backordered* and the order's status is set to *AwaitingBackorderedItems*. If all items in the order were successfully *reserved* then the order's status is set to *ReadyForPacking.*

**Process 2: Gathering order items and moving those items to the packing station**:

The Nile.com fulfillment centers are remarkably automated. Although humans do most of the placing of individual items into packages, each fulfillment center has a fleet of robots that retrieve items from storage and bring them to a human employee at a packing station. The employee retrieves the items from the robot and puts them into an appropriate package for shipment, as described in the next section.

Each robot in the Nile.com fulfillment center is assigned a unique ID, which is painted on its side. The database also records the robot's make and serial number. To keep track of the robots and keep them charged efficiently, each robot keeps a record of its current charge (as a percent of total capacity) and its current location within the fulfillment center. These values are updated approximately once per second when the robot is actively moving and once per minute when it is at rest or charging. The database does not keep a record of the history of locations and charge status, it simply records the most recently received data sent by the robot.

The fulfillment center's scheduling algorithm uses a variety of criteria to select which order to fulfill next. For our purposes, we can just assume that when there are robots available to gather items for an order, it will select the next order whose status is set to ReadyForPacking and assign robots to gather the items for that order. To do so, it chooses a packing station for the order (each packing station has a unique ID) and gets a list of the items included in the order from the database. For each item in the order, it checks the inventory tracking system to determine the rack number and bin number in which that item is stored and assigns a robot to gather that item from that rack and bin and deliver it to the assigned packing station. If the order contains more than one of the same item, the robot will retrieve the correct quantity ordered. The scheduling algorithm assigns each robot a target time for arrival at the packing station, for coordination and efficiency.

Once a robot receives this assignment, it sets its assignment status to *InProcess*, plans a route, and travels to the assigned rack and bin to retrieve the item(s). Once the items have been retrieved, it sets their status for this order to *Picked* and requests that the inventory tracking system reduce the quantity of that item in that bin by the number of items it has retrieved. For example, if it retrieves two pairs of Apple Air Pods (ItemID 2252351) from rack 552, bin 21 then it will request that the quantity of ItemID 2252351 in rack 552, bin 21 be reduced by 2. The robot then travels to the assigned packing station. Once it arrives at the packing station, it sets its assignment status to *AtPackingStation* and waits for the other robots assigned for this order to arrive with their items. When all robots have arrived and are ready to deliver their items to the packing station, they alert the scheduling algorithm and await instructions to proceed to the packing station to unload their gathered items.

**Process 3: Placing items in packaging, confirming the correct items have been selected and that they are intact, sealing packaging and labeling for delivery**:

Once the robots have gathered all of the items required to fulfill an order, they gather at a packing station and wait their turn in line. The system presents the employee working at that packing station with a screen showing the items to be packaged and shipped. It recommends the correct size and form of packaging (e.g. a specific size of cardboard box or envelope) based on the dimensions of the items. Each item's physical dimensions are stored in the database as Length x Width x Height, all in cm. When the employee has gathered the correct packaging, the system prints a shipping label that he affixes to the package to be used for shipping. The shipping label includes the delivery address and a barcode that encodes a PackageID that is linked back to the OrderID contained within.  When the label is printed, the system generates a new Package record using the PackageID encoded on the label's barcode, setting that package's status to *InProcess*. The system also associates that PackageID with the OrderID contained inside, and records the exterior dimensions of the package, based on the packaging recommended by the system.

At this point, the employee presses a button on the screen indicating that he is ready to load the items into the package. The robots carrying items for this package proceed to the packing station at which point the employee retrieves the items to be packaged from each robot and puts them into the package. As each item is put into the package the employee inspects it to make sure that it is the correct item and that it does not appear to be broken or otherwise unacceptable. When the item has been taken from the robot and added to the package, the employee taps a button on the screen indicating that the item has been added to the package. The robot that delivered that item sets the status for that package delivery to *Completed* and moves away from the packing station to go get its next assigned item. The next robot carrying an item for this order then moves forward to the packing station and the process repeats until all items have been placed into the package. Once all items have been put into the package, the package is sealed, its status is set to *PackingComplete*, and the package is put on a conveyor belt that moves the package from the packing station to its assigned loading dock. When the package arrives at its assigned loading dock, the package status is set to *AwaitingShipment* and the status of all items in the order are set to *Packed*.

**Process 4: Delivering packages to the customer**:

The final step in this process is to deliver the package from the fulfillment center to the customer. Although this is a complicated process, for the purpose of this design we are only concerned about tracking the status of the package as it travels from the packing station to the shipper and then on to the customer.  To that end, once the package has been placed on a truck, it is assigned a tracking number ~~by the~~ and a shipper (FedEx, UPS, etc.) that is recorded and the package status is marked as *InTransit*.  Likewise, the Order's status is marked as *Shipped*. Once the package has been delivered to the customer, the status for both the package and the order are marked as *Delivered*.

**Background on managing customer and address data**:

Each customer has a unique CustomerID that is consistent across Nile.com's systems. Each customer record needs to contain that customer's name, userid, email, and password. Each customer may have zero or more delivery addresses saved in the system. The ability to store multiple addresses per customer account allows the customer to send packages to multiple locations, such as a home address and a work address, by simply selecting from a list of known addresses at checkout. Each address needs to have a street address (optionally including an apartment number), a city, a state or province, a zip code, and a country. Customers can save a descriptive name for each address to help keep track of them (home, work, mom's house, etc.). To help cut down on fraud, Nile.com records whether each address in its system is a verified address **for that customer**. To do so, when a customer adds an address to the system, it is considered an unverified address. Once a package has been successfully ordered and delivered to that address, the address is marked as verified for that customer. Note that because an address is verified for customer A, that does not mean that the address is also verified for customer B.

Just as a customer is able to save more than one address, the system needs to be able to associate more than one customer with each address. That supports scenarios where multiple people who are Nile.com customers can keep their Nile.com accounts separate. You and your roommate, for example.

**Requirements updates and clarifications:**

Updates and/or clarifications to the requirements will be noted here. Changes to the original spec are included in the sections above in this green font.

**Appendix B: Sample data you must load into your database**

The operations team at Nile.com has provided us with the following sample data to test our database designs. To do so, you need to load data into your SQLite database to capture all of the information described below. If you are unable to find an appropriate place for all of the data to go, while satisfying primary key and foreign key constraints, then you probably have a problem with the design of your database. Your next step should be to figure out how to adapt the design to accommodate and store it all. If you find that it all fits well (a place for everything, and everything in its place), that's likely a good sign (but not a guarantee) that your design is solid.

You may find that you do not have enough data to fill in all of the fields for every row. That should be ok. The missing fields should all be allowed to be null in your database design so that if you don't have the correct data for those fields as you enter the other data, the database should still accept the data that you do have available.

All of this sample data comes from Nile.com fulfillment center #36, located at 4015 Sycamore Street, Cairo Illinois, 62914 (Latitude 37.01884502780466, Longitude -89.18412981427214).

**Robots**:  
Two of the robots currently in operation at fulfillment center #36 are Delos 5526 models (ID's # 623 and 624). As of their last update, robot #623 had a 63% charge and robot #624 had an 83% charge.  Robot #624 is at packing station 22 and robot #623 is currently located on aisle #31 at rack #15.

**Customers**:

The following three people are stored as customers in the database:

* Ken Jennings, who lives at 1103 East Orange Grove Blvd, Pasadena, CA, 91104
* Bernie McDonald, who lives at 5415 Plainfield Street, Pittsburgh PA 15217. Bernie's work address is also stored in the database - 5000 Forbes Ave., Pittsburgh PA, 15213
* Karen McDonald, who also lives at 5415 Plainfield Street, Pittsburgh PA, 15217

**Orders**:  
Both robots are currently assigned to fulfill Order #55153. This assignment was sent to the robots with instructions to meet at packing station 22 at 07:15:45am on Wednesday, October 6, 2021.  Order #55153 was placed by Ken Jennings at 11:07am, Pacific Standard Time, on October 4, 2021 with promised delivery of October 8, 2021. He took advantage of a free delivery offered so he will not be charged for delivery but he will need to have California's 10.25% sales tax added to the total price for his order. Ken has specified that the order should be delivered to his house.

* Robot #624 has already retrieved (status: Picked) item #21321, one "LEGO Ideas International Space Station 21321 Building Kit", dimensions [38.2cm x 26cm x 7cm], that was purchased on this order for $69.89. It has completed the trip to packing station #22 and is currently waiting for robot #623 to join it so that they can move to the packing station to hand their items to the employee who will pack them.
* Robot #623 is currently on its way to rack #55 to retrieve one of item #10295, stored in bin 11. Item #10295 is a "LEGO Porsche 911 (10295) Model Building Kit", with the description "Engaging Building Project for Adults; Build and Display The Iconic Porsche 911; New 2021 (1,458 Pieces)". It's dimensions are [48cm x 28cm x 12cm]. This item was purchased for $149.95. There are currently 4 of these kits in stock in this bin, of which one has been reserved for this order.

Bernie McDonald has also placed an order with Nile.com (Order #55178). He placed the order at 2:15pm, Eastern Standard Time on October 4. His order contained one item (Item #21321, Lego Ideas International Space Station 21321 Building Kit) and he requested that the item be delivered to his home address. He was charged $10.00 for shipping and 7% of the purchase price for sales tax. His order was promised for delivery on October 7, 2021.  It was packaged on October 5 at 11:17am in a package with dimensions [55cm x 30cm x 10cm). The package was shipped via UPS and assigned tracking #55262351345.

**Data updates and clarifications:**

Updates to the data requirements and/or clarifications will be flagged and listed here. Changes to the original spec and sample data are flagged in the main description by this green font.