CSPB 3287: Lab #1

Data Normalization

This assignment will provide you with practical ("hands on") experience in system design. As a database developer you may be asked to create tables from current business activities. Often these business activities are captured in forms that are filled out by the sales department or customers. For example, a "Retail Sales Ticket" and a "Customer Record" are used to capture the information needed to handle specific work for a specific customer. Your task is to collect the list of information that is captured on those forms and iteratively change the data into a form that can be used to describe a set of relationships between tables in a database.

The information contained in the "Case Study" document will provide you with an understanding of the practices of a hypothetical business. Your role as a consultant will be to evaluate the information given and create a database design to meet the company's requirements. The first step in creating the database is to collect the set of attributes (fields) needed. Once you have collected that un-normalized data, you will proceed to normalize the data into the first three normal forms.

Although the assignment will focus on a subset of the information provided in the case study, make sure to thoroughly review Eden Landscaping's business. Pay particular attention to the forms used by this business as these will dictate the information you will need to capture in your design.

Your assignment is to create 3rd Normal Form Schemas for the business documents "Customer Record" and "Retail Sales Ticket" (shown below in Appendix A) used by Eden Landscaping. Your final 3NF column must show entities, attributes, primary keys and foreign keys for all the data appearing in these documents.

Submission

The easiest way to complete this assignment is to create a "Spreadsheet" (see the format provided in the example below) with one column for each of the normal forms. Please save your spreadsheet as a PDF and submit the PDF via the Moodle Assignment.

Submission Format

Record your results in columnar format imitating the spreadsheet template provided in the image below. Your spreadsheet should have four columns:

Unnormalized   1st Normal Form   2nd Normal Form   3rd Normal Form

Begin by listing, for each document, all data elements ("attributes") found on the business document in the "unnormalized" column.

List the document name in UPPER CASE and HIGHLIGHT it in yellow.

For example, if a customer form had the following fields, your unnormalized column would look something like:

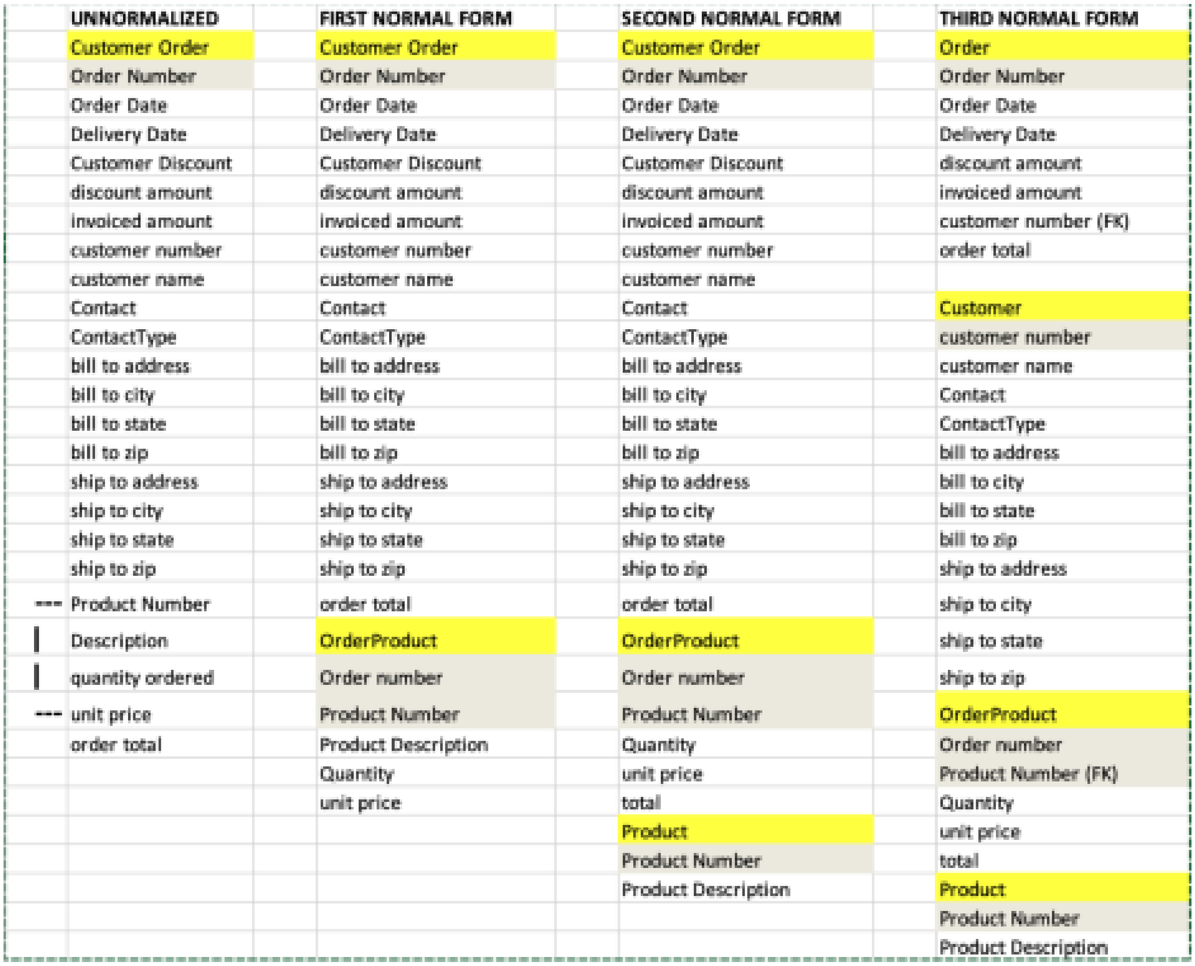
Unnormalized
CUSTOMER ORDER
Order Number
Order Date
Delivery Date
Customer Number
Customer Name
Etc.
Etc.

After listing all data elements ("attributes") in the "unnormalized" column for the two documents, proceed to go through the list and put all data into First Normal Form.

- Replace each document name with an entity name as needed (highlighted in yellow). If an attribute appears on multiple documents (for example, CustomerName), then you should only list it once in your First Normal Form column.
- Identify the Primary Keys by highlighting them in a different color other than yellow (the keys are highlighted in gray in the example below.)
- Identify the Foreign Keys by adding a "(FK)" tag after the foreign key attribute name (as you see in the example below.)
- Do the same for the second and third normal form columns.
- Your final 3NF column should contain all the data in all the documents organized by entity, listing all attributes in each entity with primary keys and foreign keys defined.
- If an entity does not change from first to third normal form, then simply copy and paste the data in the second normal form column to show that the data is already in second normal form.

  HINT: you'll know this if the entity has an atomic key – that is, NOT concatenated/compounded. Without a compound key, the data is already in 2NF.

Spreadsheet Format with example data:



	UNNORMALIZED	FIRST NORMAL FORM	SECOND NORMAL FORM	THIRD NORMAL FORM
	Customer Order	Customer Order	Customer Order	Order
	Order Number	Order Number	Order Number	Order Number
	Order Date	Order Date	Order Date	Order Date
	Delivery Date	Delivery Date	Delivery Date	Delivery Date
	Customer Discount	Customer Discount	Customer Discount	discount amount
	discount amount	discount amount	discount amount	invoiced amount
	invoiced amount	invoiced amount	invoiced amount	customer number (FK)
	customer number	customer number	customer number	order total
	customer name	customer name	customer name	
	Contact	Contact	Contact	Customer
	ContactType	ContactType	ContactType	customer number
	bill to address	bill to address	bill to address	customer name
	bill to city	bill to city	bill to city	Contact
	bill to state	bill to state	bill to state	ContactType
	bill to zip	bill to zip	bill to zip	bill to address
	ship to address	ship to address	ship to address	bill to city
	ship to city	ship to city	ship to city	bill to state
	ship to state	ship to state	ship to state	bill to zip
	ship to zip	ship to zip	ship to zip	ship to address
---	Product Number	order total	order total	ship to city
┃	Description	OrderProduct	OrderProduct	ship to state
┃	quantity ordered	Order number	Order number	ship to zip
---	unit price	Product Number	Product Number	OrderProduct
	order total	Product Description	Quantity	Order number
		Quantity	unit price	Product Number (FK)
		unit price	total	Quantity
			Product	unit price
			Product Number	total
			Product Description	Product
				Product Number
				Product Description

Case Study

To do this assignment, you must first read the Eden Landscaping Case Study. This will help you understand Eden's organization and how their business operates.

Data Normal Forms

Data normalization is a process in database design and data management that organizes data to reduce redundancy and improve data integrity. It involves structuring a relational database in accordance with a series of normal forms to:

- Minimize data redundancy
- Reduce data anomalies
- Ensure data consistency
- Simplify data management

The main goals of data normalization are to:

- Organize data efficiently
- Eliminate redundant data
- Ensure data dependencies make sense

There are several levels of normalization, known as normal forms. The most common are First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF). Each form has specific rules and builds upon the previous one.

First Normal Form (1NF) is the initial level of database normalization. It sets the basic rules for an organized database.

Key concepts of 1NF:

1. Atomicity: Each column should contain atomic (indivisible) values. This means each cell should have a single value, not multiple values or lists.
2. Unique column names: Each column in a table must have a unique name.
3. Order independence: The order of rows and columns should not affect the data's meaning or functionality.
4. No repeating groups: There should be no repeating groups of columns.

To achieve 1NF:

1. Eliminate repeating groups in individual tables.
2. Create a separate table for each set of related data.
3. Identify each set of related data with a primary key.

Second Normal Form (2NF) builds upon First Normal Form (1NF) and addresses the issue of partial dependencies. A table is in 2NF if it's in 1NF and all non-key attributes are fully functionally dependent on the entire primary key.

Key concepts of 2NF:

1. The table must already be in 1NF.
2. All non-key attributes must depend on the entire primary key.

To achieve 2NF:

1. Identify any partial dependencies (where an attribute depends on only part of a composite key).
2. Remove these partial dependencies by creating separate tables.

Third Normal Form (3NF) builds upon Second Normal Form (2NF) and addresses the issue of transitive dependencies. A table is in 3NF if it's in 2NF and all non-key attributes are directly dependent on the primary key and not on other non-key attributes.

Key concepts of 3NF:

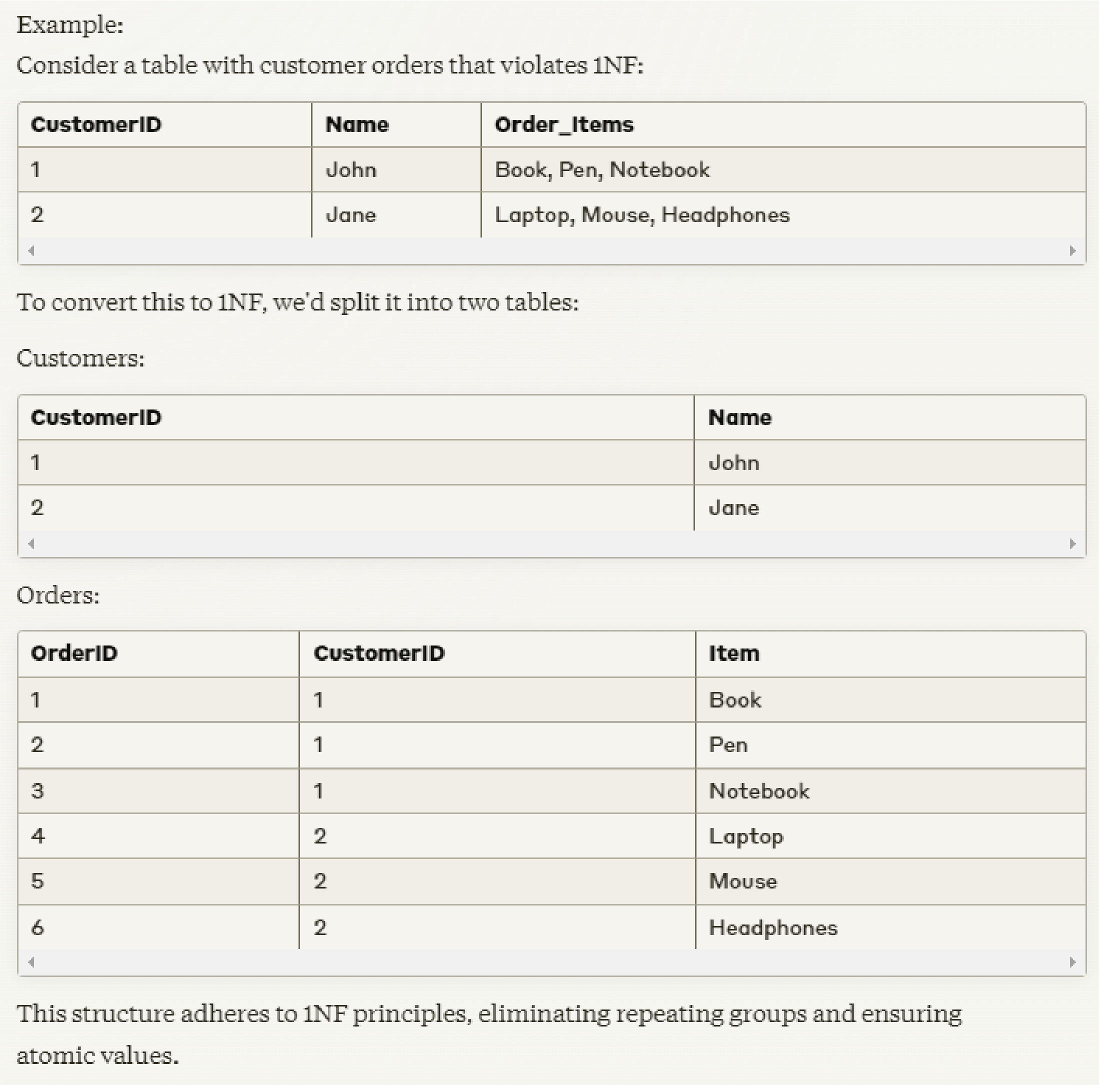
1. The table must already be in 2NF.
2. There should be no transitive dependencies for non-prime attributes.

To achieve 3NF:

1. Identify any transitive dependencies.
2. Remove these dependencies by creating separate tables.

Example

Below is an unnormalized table that contains multiple data items in a single field. This violates the rules for first normal form. To translate the unnormalized data, we can create two tables that capture the same information, but only have a single (atomic) data value in each field.



Example:
Consider a table with customer orders that violates 1NF:

CustomerID	Name	Order_Items
1	John	Book, Pen, Notebook
2	Jane	Laptop, Mouse, Headphones

To convert this to 1NF, we'd split it into two tables:

Customers:

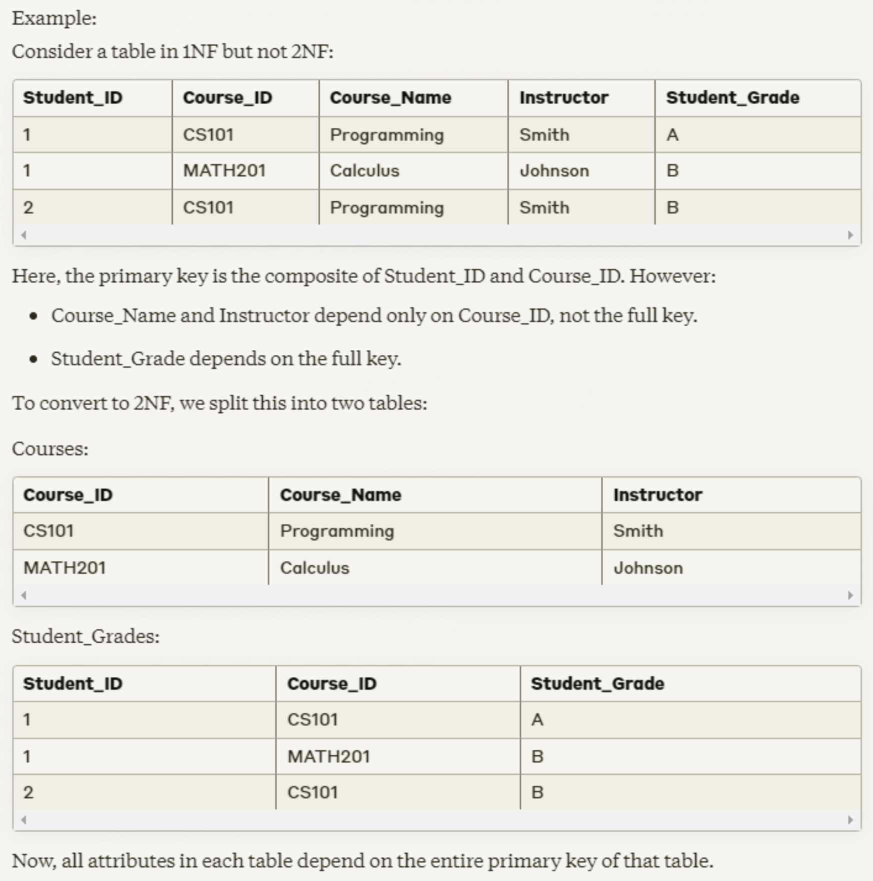
CustomerID	Name
1	John
2	Jane

Orders:

OrderID	CustomerID	Item
1	1	Book
2	1	Pen
3	1	Notebook
4	2	Laptop
5	2	Mouse
6	2	Headphones

This structure adheres to 1NF principles, eliminating repeating groups and ensuring atomic values.

Second normal form wants all data accessible by a primary key. In the example below, there is data in each record that can be obtained from one part of the primary key. Moving this dependent data to another table will keep information from being duplicated and will make each row of the data depend only on the primary key.



Example:

Consider a table in 1NF but not 2NF:

Student_ID	**Course_ID**	**Course_Name**	**Instructor**	**Student_Grade**
1	CS101	Programming	Smith	A
1	MATH201	Calculus	Johnson	B
2	CS101	Programming	Smith	B

Here, the primary key is the composite of Student_ID and Course_ID. However:

- Course_Name and Instructor depend only on Course_ID, not the full key.
- Student_Grade depends on the full key.

To convert to 2NF, we split this into two tables:

Courses:

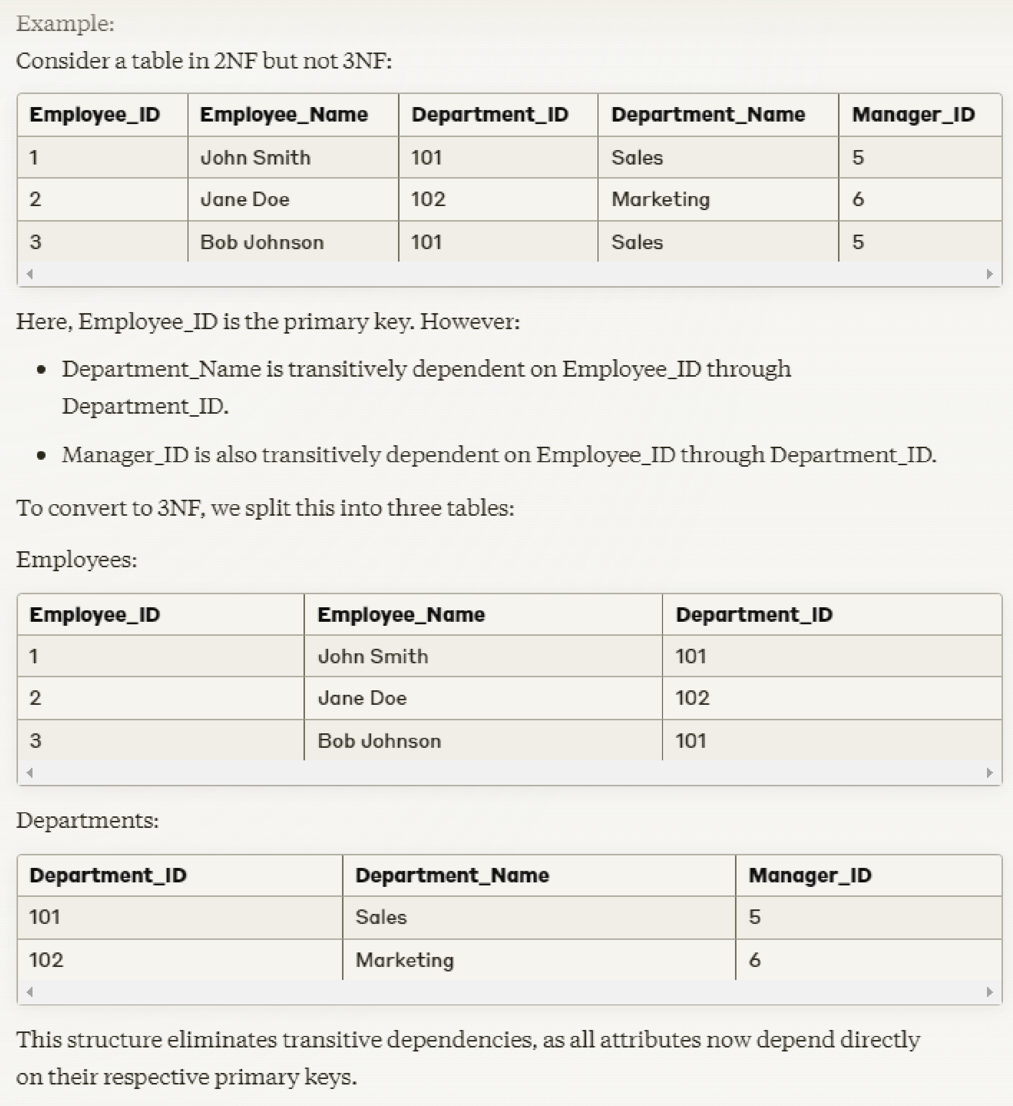
Course_ID	**Course_Name**	**Instructor**
CS101	Programming	Smith
MATH201	Calculus	Johnson

Student_Grades:

Student_ID	**Course_ID**	**Student_Grade**
1	CS101	A
1	MATH201	B
2	CS101	B

Now, all attributes in each table depend on the entire primary key of that table.

The last normalization needed before the information can be used to create database tables, is to make it into third normal form. Notice that the information in the first table has a field that depends on another field within the row.



Example:

Consider a table in 2NF but not 3NF:

Employee_ID	Employee_Name	Department_ID	Department_Name	Manager_ID
1	John Smith	101	Sales	5
2	Jane Doe	102	Marketing	6
3	Bob Johnson	101	Sales	5

Here, Employee_ID is the primary key. However:

- Department_Name is transitively dependent on Employee_ID through Department_ID.
- Manager_ID is also transitively dependent on Employee_ID through Department_ID.

To convert to 3NF, we split this into three tables:

Employees:

Employee_ID	Employee_Name	Department_ID
1	John Smith	101
2	Jane Doe	102
3	Bob Johnson	101

Departments:

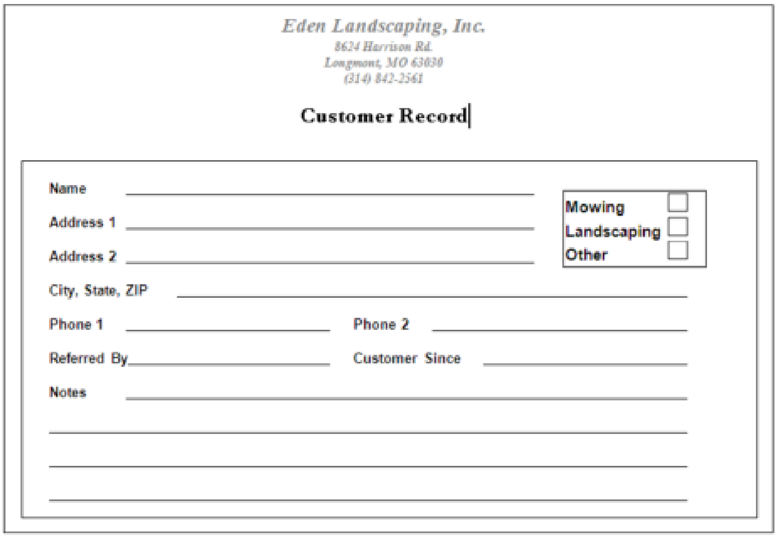
Department_ID	Department_Name	Manager_ID
101	Sales	5
102	Marketing	6

This structure eliminates transitive dependencies, as all attributes now depend directly on their respective primary keys.

Appendix A – Eden Landscaping Documents

Below are copies from the Case Study document of the two forms from which you must capture the data.

Customer Record



Eden Landscaping, Inc.
8624 Harrison Rd.
Longmont, MO 63030
(314) 842-2561

Customer Record

Name ____________________

Address 1 ____________________

Address 2 ____________________

Mowing □
Landscaping □
Other □

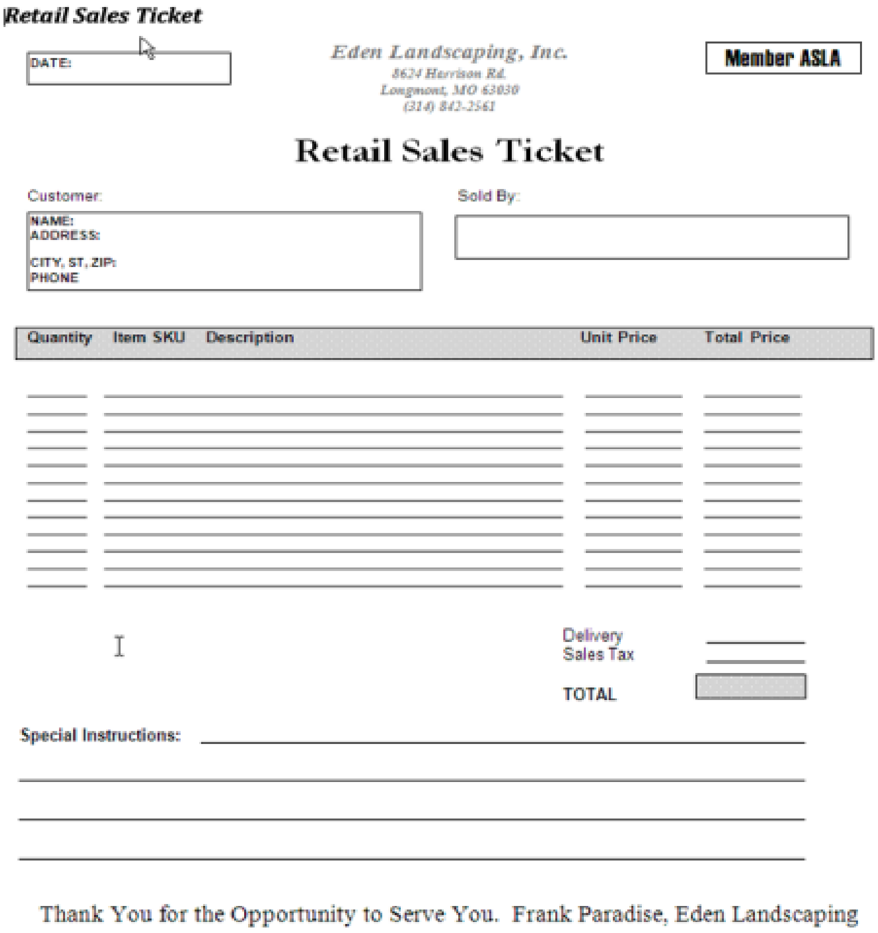
City, State, ZIP ____________________

Phone 1 __________ Phone 2 __________

Referred By__________ Customer Since __________

Notes ____________________

Retail Sales Ticket



Retail Sales Ticket

DATE:

Eden Landscaping, Inc.
8624 Harrison Rd.
Longmont, MO 63030
(314) 842-2561

Member ASLA

Retail Sales Ticket

Customer:
NAME:
ADDRESS:
CITY, ST, ZIP:
PHONE

Sold By:

Quantity	Item SKU	Description	Unit Price	Total Price

Delivery ________
Sales Tax ________
TOTAL

Special Instructions: ____________________

Thank You for the Opportunity to Serve You. Frank Paradise, Eden Landscaping