# CSC3170 Database Systems
# Term Project Report


Student ID: 116010274

Name (Chinese): 俞佳含

Name (Pinyin): Jiahan YU


Date: 12/05/2018

**A. Draw an E-R data model for the company database schema. Use the IE Crow's Foot E-R model for your E-R diagrams. Justify the cardinalities of each relationship.**

The current database schema given by the question is like below:

CUSTOMER (CustomerID, LastName, FirstName, Address, City, State, Zip, Phone, Email)

ITEM (ItemID, ItemDescription, CompanyName, PurchaseDate, ItemCost, ItemPrice)

SALE (SaleID, CustomerID, SaleDate SubTotal, Tax, Total)

SALE_ITEM (SaleID, SaleItemID, ItemID, ItemPrice)

EMPLOYEE (EmployeeID, LastName, FirstName, Phone, Email)

VENDOR (VendorID, CompanyName, ContactLName, ContactFName, Address, City, State, Zip, Phone, Fax, Email)

Based on this schema, I made several reasonable modifications and designed their relationships:

| Relationship | | | Cardinality | |
|---|---|---|---|---|
| **Parent** | **Child** | **Type** | **Max** | **Min** |
| CUSTOMER | SALE | Strong Type | 1:N | M-O |
| EMPLOYEE | SALE | Strong Type | 1:N | M-O |
| SALE | SALE_ITEM | ID Dependent Type | 1:N | M-M |
| ITEM | SALE_ITEM | Strong Type | 1:1 | M-O |
| VENDOR | ITEM | Strong Type | 1:N | M-O |

Explanation of the above chart for question A:

1. A customer may have a purchase. A customer can have many purchases.
   One sale order must have and can only have one customer.
2. A employee may be in charge of a sale order. A employee can be in charge of many sale orders.
   A sale order must have and only have one employee in charge of it.
3. A sale order must have a sale item. A sale order can have many sale items.
   A sale item must have and only have one corresponding sale order.
4. An item may be sold, meaning an item may have a sale item. An item can have at most one sale item.
   A sale item must match and can only match one recorded item.
5. A vendor may store an item. A vendor can store many items.
   One item must have and only have one vendor (at least in current ER model).

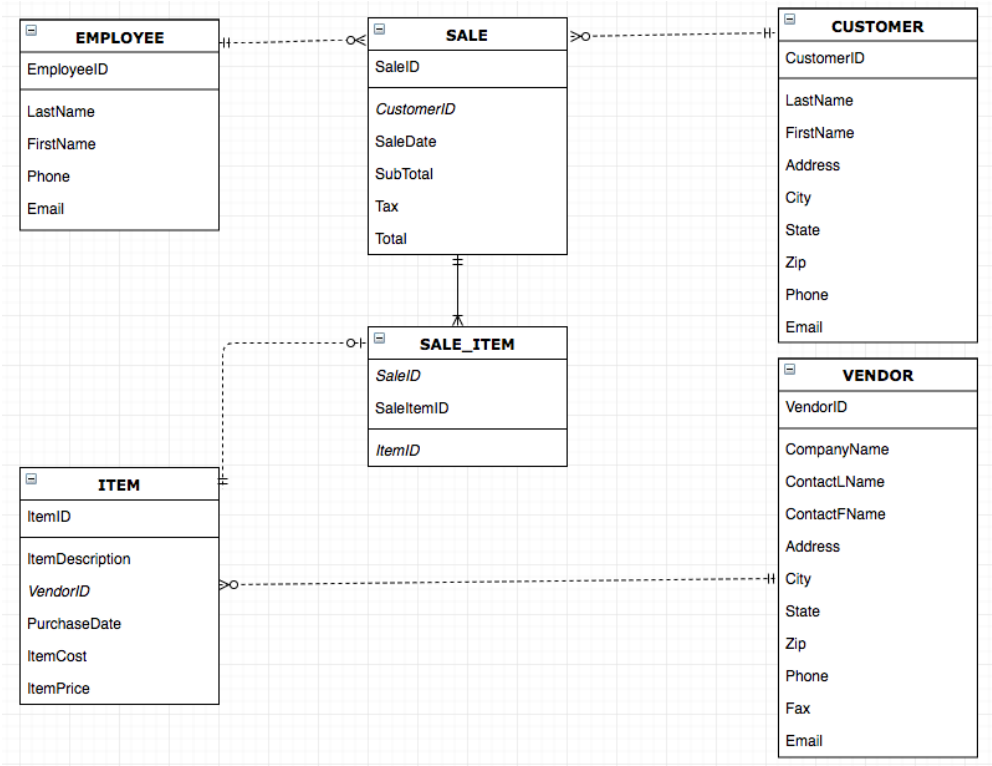



Figure 1 - E-R data model for the company database schema
(using www.draw.io tool)

The referential integrity constraints in ER data model are:

1. VendorID in ITEM must exist in VendorID in VENDOR
2. CustomerID in SALE must exist in CustomerID in CUSTOMER
3. ItemID in SALE_ITEM must exist in ItemID in ITEM
4. SaleID in SALE_ITEM must exist in SaleID in SALE

***Remark: I have modified some of foreign keys of entities while considering referential integrity constaints:*

1. *Alter CompanyName in ITEM to VendorID as VendorID is surrogate key and therefore better to store VendorID rather than CompanyName.*
2. *Remove ItemPrice in SALE_ITEM since it can be tracked by the foreign key ItemID in SALE_ITEM.*
3. *This is not complete version for foreign key constraint since it is still in the data model not database design.*

**B. Extend and modify the E-R data model by adding only the company's inventory system requirements. Use the IE Crow's Foot E-R model for your E-R diagrams. Create appropriate identifiers and attributes for each entity. Justify the decisions you make regarding minimum and maximum cardinalities. (4 points)**

Principles:

1. When an item is purchased from a Vendor they get an Order receipt.
2. Record the date when an Order is made and when the Order is received.
3. Record the original cost of a single item on the Order.
4. Store how many items were purchased on the Order.
5. Each order has a subtotal of the items, how much tax was applied and the total cost of each Order.
6. Eliminate redundant information from the Item model in part A.
7. Add the ability to track what is on-hand and what is on-order in the Item model.
8. Sale_Item can now have a quantity and an extended price.

| Relationship | | | Cardinality | |
|---|---|---|---|---|
| **Parent** | **Child** | **Type** | **Max** | **Min** |
| ITEM | SALE_ITEM | Strong Type | 1:N | M-O |
| ITEM | ORDER | Strong Type | 1:N | M-O |
| VENDOR | ORDER | Strong Type | 1:N | M-O |

Explanation of the above chart for question B:

1. Every kind of item may be sold. One kind of item can have one or more sale items. One particular item sold in a purchase must have and can only have one corresponding item information.
2. Every kind of item may be appeared in sale orders. One kind of item can appeared in many sale orders. One order can only have one sale item (meaning the kind, not the quantity).
3. A vendor may have an order. A vendor can have many orders. One order must have and only have one vendor who provides the sale items.

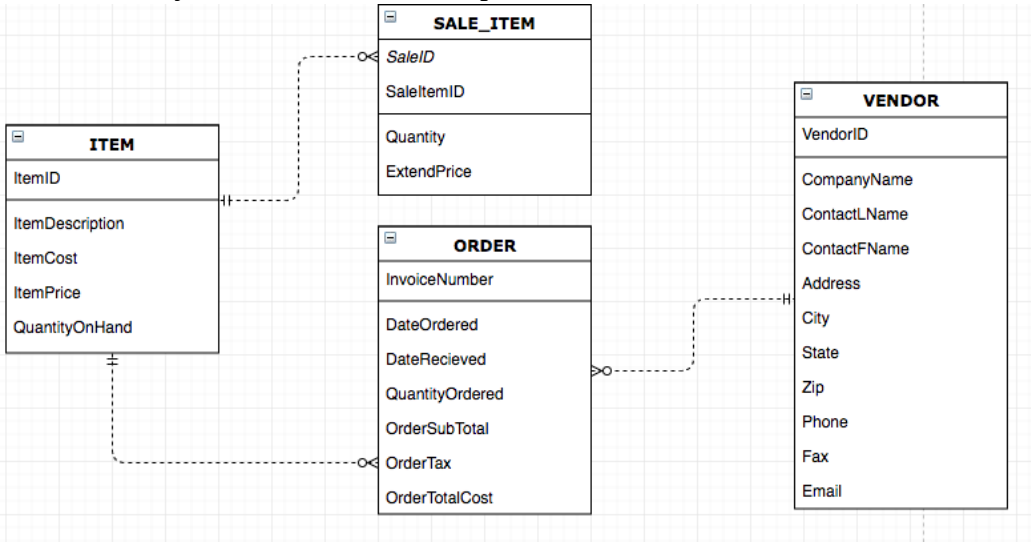



Figure 2 – Partial E-R data model for the company inventory system (using www.draw.io tool)

**C. Extend and modify the E-R data model by adding only the company's need for more efficient storage of CUSTOMER and EMPLOYEE data. Use the IE Crow's Foot E-R model for your E-R diagrams. Create appropriate identifiers and attributes for each entity. Justify the decisions you make regarding minimum and maximum cardinalities.**

Principles:

1. A Person can be a Customer or Employee or both.
2. A Person has all of information from the Customer model Part A.
3. Customer and Employee are subtypes of a Person.
4. Use the primary key of the supertype for each subtype.
5. Store the card type, card number and expiration date of the credit card a Customer uses.
6. Keep track of when an Employee is hired and their rate of pay.

(Principle 5-6 just define some useful attributes of employees and customers, which is not required explicitly in the question.)

| Relationship | | | Cardinality | |
|---|---|---|---|---|
| **Parent** | **Child** | **Type** | **Max** | **Min** |
| PERSON | CUSTOMER | ID dependent Inclusive Type | 1:N | M-O |
| PERSON | EMPLOYEE | ID dependent Inclusive Type | 1:N | M-O |

This relationships and cardinalities can be inferred from the above principles.

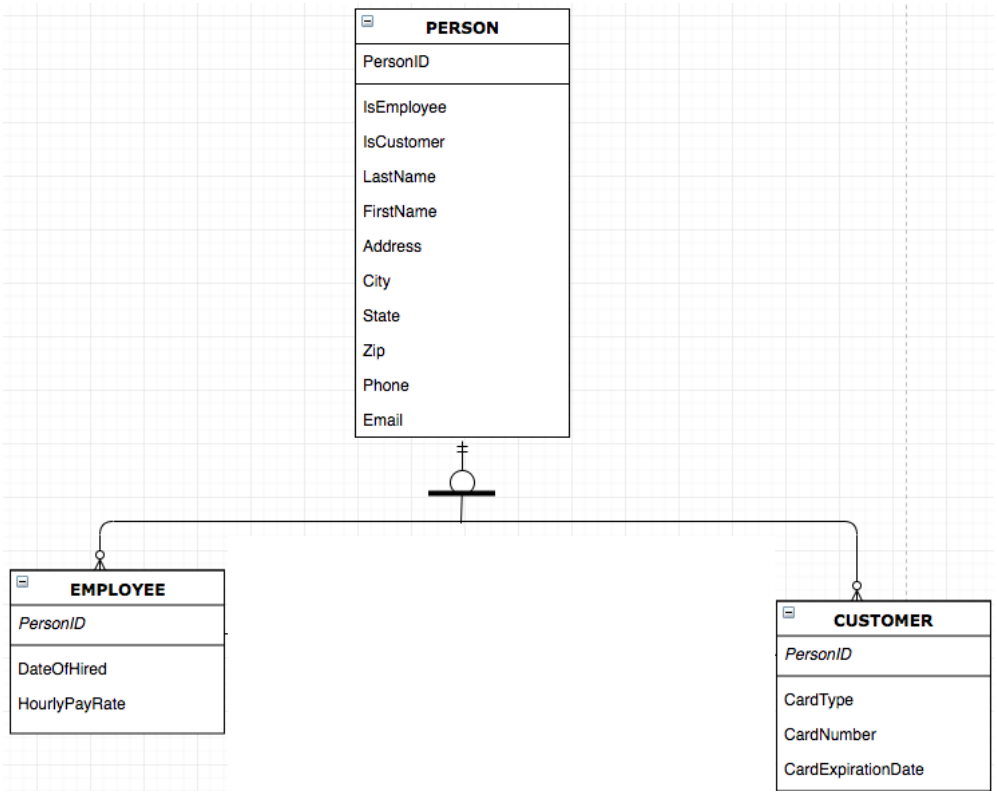



Figure 3 – Partial E-R data model using subtypes
(using www.draw.io tool)

**D. Combine the E-R data models from parts B and C to meet all the company's new requirements, making additional modifications as needed. Use the IE Crow's Foot E-R model for your E-R diagrams.**

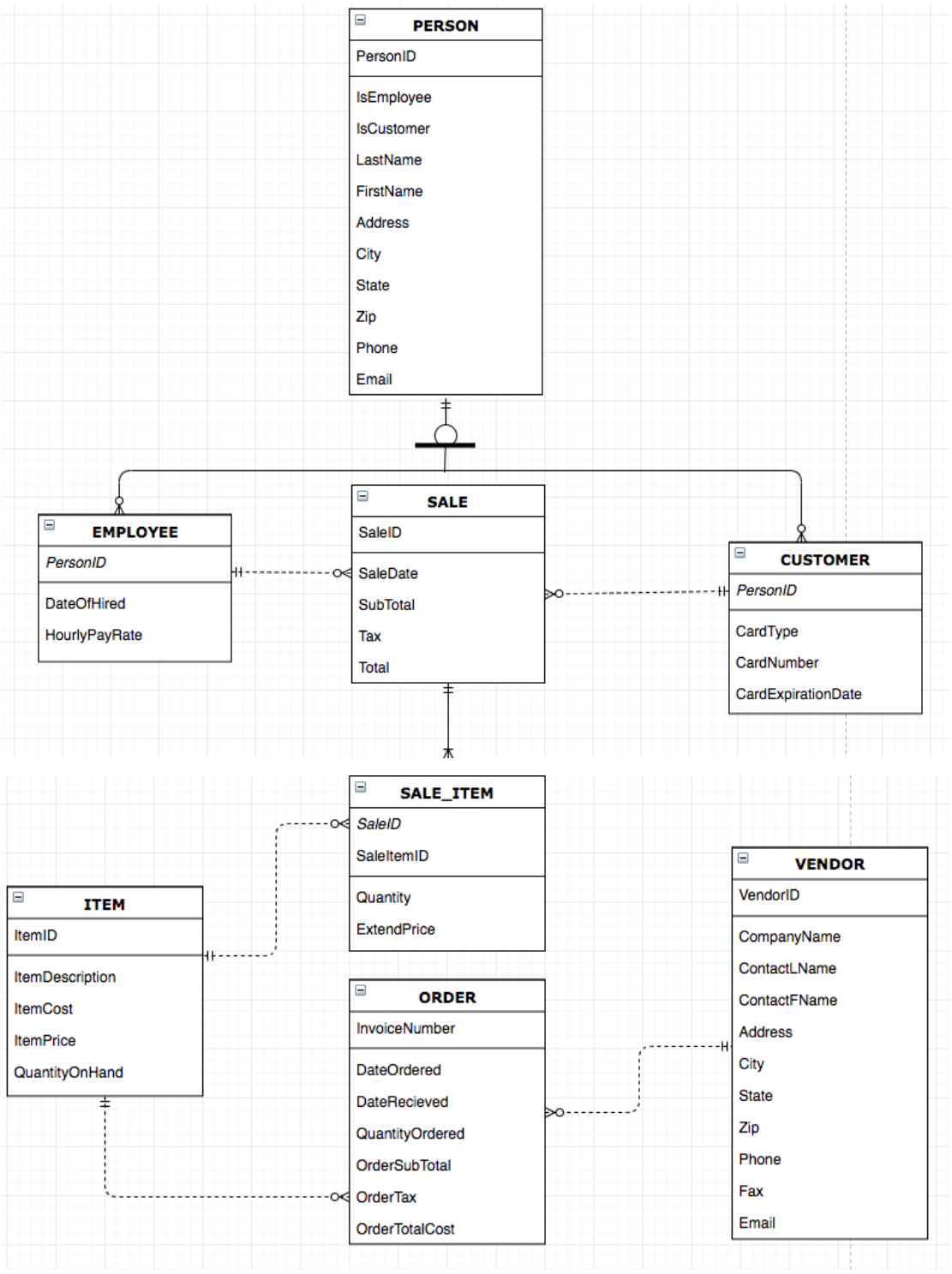



Figure 4 – Complete E-R data model
(using www.draw.io tool)

**E. Describe how you would go about validating your data model in part D.**

For inventory system:
These changes allow the sales system to handle nonunique items that can be bought and sold in quantity. When new items from vendors arrive, the office personnel unpack the items, put them in the stockroom, and run an Item Quantity Received Transaction that adds the quantity received to QuantityOnHand. At the same time, another transaction called an Item Price Adjustment Transaction is run, if necessary, to adjust ItemCost and ItemPrice. Sales may occur at any time, and when a sale occurs, the Sale Transaction is run. Every time a SALE_ITEM line is entered, the input Quantity is subtracted from QuantityOnHand in ITEM.

For employee and customer relationship as well as redundant data:
As a person can be both employee and customer, I design the ID dependent inclusive type while using subtypes. I store the similar data under the supertype PERSON and add some more special attributes/properties to customer and exployee which I think is important information for the company to track. Meanwhile, when an employee buys something at the store, his or her data will not be reentered into the CUSTOMER table any more, meaning we store the data efficiently and give away some redundant storage.

**F. Convert the data model to a database design. Specify tables, primary keys, and foreign keys.**

Step1: specify primary keys ( ), foreign keys (FK) and alternative keys (AK).

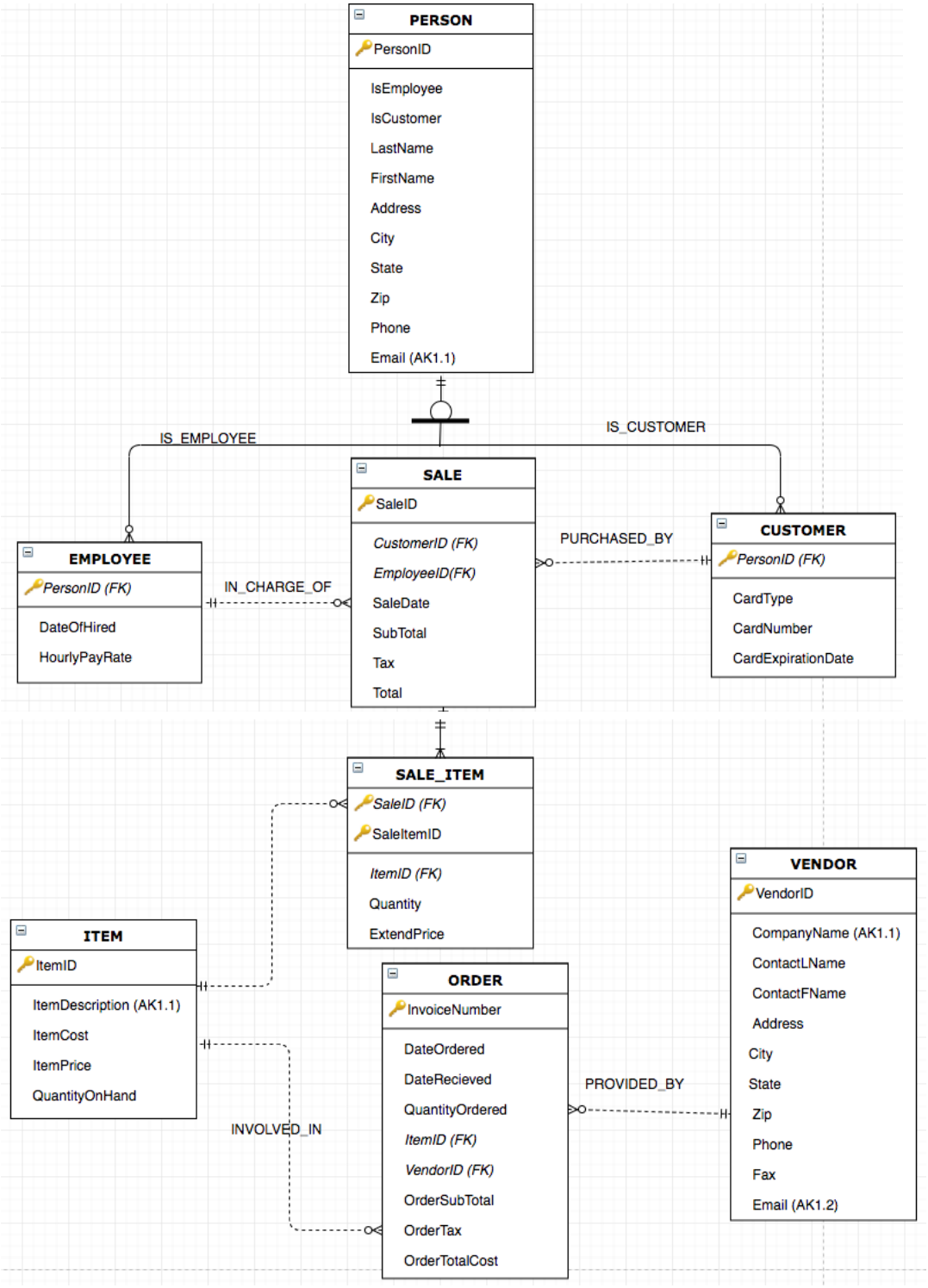



Figure 5 – database design
(using www.draw.io tool)

Surrogate keys are needed for: Surrogate Key IDENTITY(10000000,1)
PersonID, SaleID, SaleItemID, ItemID, VendorID, InvoiceNumber

Step2: Specify Column Properties (Not mention PK, AK here, they are shown in the above figure)

**Table - PERSON**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| PersonID | Int | NOT NULL | Surrogate key | |
| IsEmployee | Boolean | NOT NULL | 0 | Only be {0,1} |
| IsCustomer | Boolean | NOT NULL | 1 | Only be {0,1} |
| LastName | Char(25) | NOT NULL | "Unknown L" | |
| FirstName | Char(25) | NOT NULL | "Unknown F" | |
| Address | Char(50) | NULL | | |
| City | Char(35) | NULL | | |
| State | Char(2) | NULL | | |
| Zip | Char(9) | NULL | | |
| Phone | Char(11) | NULL | | |
| Email | Varchar(100) | NULL | | |

**Table - EMPLOYEE**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| PersonID | Int | NOT NULL | Surrogate key | Interrelation |
| DateOfHired | Date | NULL | | |
| HourlyPayRate | Int | NULL | | |

**Table - CUSTOMER**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| PersonID | Int | NOT NULL | Surrogate key | Interrelation |
| CardType | Char(25) | NULL | | Intrarelation |
| CardNumber | Char(15) | NULL | | |
| CardExpiration Date | Date | NULL | | Intrarelation |

**Table – SALE**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| SaleID | Int | NOT NULL | Surrogate key | |
| CustomerID | Int | NOT NULL | Surrogate key | Interrelation |
| EmployeeID | Int | NOT NULL | Surrogate key | Interrelation |
| SaleDate | Date | NULL | | |
| SubTotal | Numeric(15) | NULL | | |
| Tax | Numeric(15) | NULL | | |
| Total | Numeric(15) | NULL | | |

**Table – SALE_ITEM**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| SaleID | Int | NOT NULL | Surrogate key | Interrelation |
| SaleItemID | Int | NOT NULL | Surrogate key | |
| ItemID | Int | NOT NULL | Surrogate key | Interrelation |
| Quantity | Int | NULL | | |
| ExtendPrice | Numeric(15) | NULL | | |

**Table – ITEM**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| ItemID | Int | NOT NULL | Surrogate key | |
| ItemDescription | Char(50) | NOT NULL | "Unknown" | |
| ItemCost | Numeric(4) | NULL | | |
| ItemPrice | Numeric(4) | NULL | | |
| QuantityOnHand | Int | NULL | | |

**Table – ORDER**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| InvoiceNumber | Int | NOT NULL | Surrogate key | |
| DateOrdered | Date | NOT NULL | | |
| DateReceived | Date | NOT NULL | | |
| QuantityOrdered | Int | NULL | | |
| ItemID | Int | NOT NULL | Surrogate key | Interrelation |
| VendorID | Int | NOT NULL | Surrogate key | Interrelation |
| OrderSubTotal | Numeric(15) | NULL | | |
| OrderTax | Numeric(15) | NULL | | |
| OrderTotalCost | Numeric(15) | NULL | | |

**Table – VENDOR**

| Column Name | Data Type | NULL Status | Default Value | Data Constraint |
|---|---|---|---|---|
| VendorID | Int | NOT NULL | Surrogate key | |
| CompanyName | Date | NOT NULL | "Unknown" | |
| ContactLName | Char(25) | NOT NULL | "Unknown L" | |
| ContactFName | Char(25) | NOT NULL | "Unknown F" | |
| Address | Char(50) | NULL | | |
| City | Char(35) | NULL | | |
| State | Char(2) | NULL | | |
| Zip | Char(9) | NULL | | |
| Phone | Char(11) | NULL | | |
| Fax | Char(11) | NULL | | |
| Email | Varchar(100) | NULL | | |

The referential constraints owing to foreign keys are (which I have demonstrated in those above charts by key word “interrelation” in “Data Constraint” column):

1. CustomerID in SALE must exist in PersonID in CUSTOMER
2. EmployeeID in SALE must exist in PersonID in EMPLOYEE
3. CUSTOMER and EMPLOYEE are subtypes of PERSON (supertype) – PersonID in PERSON must exist in PersonID in CUSTOMER or EMPLOYEE or both
4. SaleID in SALE_ITEM must exist in SaleID in SALE
5. ItemID in SALE_ITEM must exist in ItemID in ITEM
6. ItemID in ORDER must exist in ItemID in ITEM
7. VentorID in ORDER must exist in VentorID in VENTOR

Step3: Enforcing minimum/maximum cardinalities

3.1 Relationships and Cardinalities between tables

| **Relationship** | | | **Cardinality** | |
|---|---|---|---|---|
| **Parent** | **Child** | **Type** | **Max** | **Min** |
| PERSON | CUSTOMER | ID-dependent Inclusive Type | 1:N | M-O |
| PERSON | EMPLOYEE | ID-dependent Inclusive Type | 1:N | M-O |
| CUSTOMER | SALE | Strong Type | 1:N | M-O |
| EMPLOYEE | SALE | Strong Type | 1:N | M-O |
| SALE | SALE_ITEM | ID-dependent Type | 1:N | M-M |
| ITEM | SALE_ITEM | Strong Type | 1:N | M-O |
| ITEM | ORDER | Strong Type | 1:N | M-O |
| VENDOR | ORDER | Strong Type | 1:N | M-O |

3.2 Enforcing minimum/maximum cardinalities

3.2.1 Database M-O Relationships

**CUSTOMER-to-SALE**

| **CUSTOMER Is Required Parent** | **Action on CUSTOMER (Parent)** | **Action on SALE (Child)** |
|---|---|---|
| **Insert** | None | Get a parent |
| **Modify key or foreign key** | Prohibit – CUSTOMER uses a surrogate key | Prohibit – CUSTOMER uses a surrogate key |
| **Delete** | Prohibit if SALE exists – data about a sale and its related transaction is never deleted (business rule). Allow if no SALE exist (business rule). | None |

**EMPLOYEE-to-SALE**

| EMPLOYEE Is Required Parent | Action on EMPLOYEE (Parent) | Action on SALE (Child) |
|---|---|---|
| **Insert** | None | Get a parent |
| **Modify key or foreign key** | Prohibit –EMPLOYEE uses a surrogate key | Prohibit – EMPLOYEE uses a surrogate key |
| **Delete** | Prohibit if SALE exists – data about a sale and its related transaction is never deleted (business rule). Allow if no SALE exist (business rule). | None |

**ITEM-to-SALE_ITEM**

| ITEM Is Required Parent | Action on ITEM (Parent) | Action on SALE_ITEM (Child) |
|---|---|---|
| **Insert** | None | Get a parent |
| **Modify key or foreign key** | Prohibit –ITEM uses a surrogate key | Prohibit –ITEM uses a surrogate key |
| **Delete** | Prohibit if SALE_ITEM exists – data about an sale item and its related transaction is never deleted (business rule). Allow if no SALE_ITEM exist (business rule). | None |

**ITEM-to-ORDER**

| ITEM Is Required Parent | Action on ITEM (Parent) | Action on ORDER (Child) |
|---|---|---|
| **Insert** | None | Get a parent |
| **Modify key or foreign key** | Prohibit –ITEM uses a surrogate key | Prohibit –ITEM uses a surrogate key |
| **Delete** | Prohibit if ORDER exists – data about an order and its related transaction is never deleted (business rule). Allow if no ORDER exist (business rule). | None |

**VENDOR-to-ORDER**

| VENDOR Is Required Parent | Action on VENDOR (Parent) | Action on ORDER (Child) |
|---|---|---|
| **Insert** | None | Get a parent |
| **Modify key or foreign key** | Prohibit –VENDOR uses a surrogate key | Prohibit –VENDOR uses a surrogate key |
| **Delete** | Prohibit if ORDER exists – data about an order and its related transaction is never deleted (business rule). Allow if no ORDER exist (business rule). | None |

3.2.2 Database M-M relationship

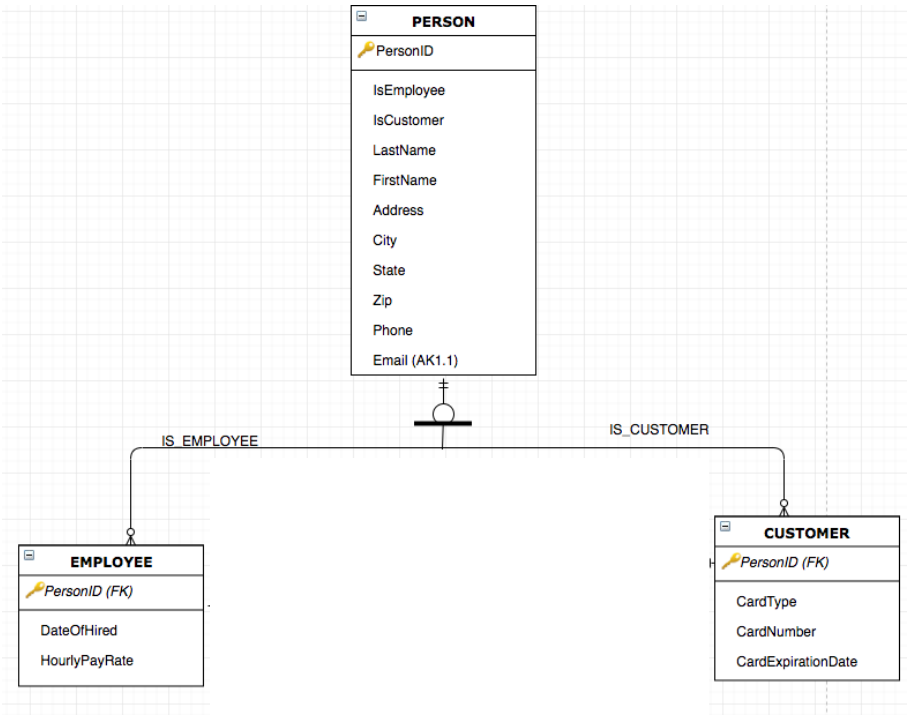
**SALE-to-SALE_ITEM**

| SALE Is Required Parent | Action on SALE (Parent) | Action on SALE_ITEM (Child) |
|---|---|---|
| **Insert** | None | Get a parent |
| **Modify key or foreign key** | Prohibit – SALE uses a surrogate key | Prohibit – SALE uses a surrogate key |
| **Delete** | Prohibit – data about a sale item and its related transaction is never deleted (business rule). | None |

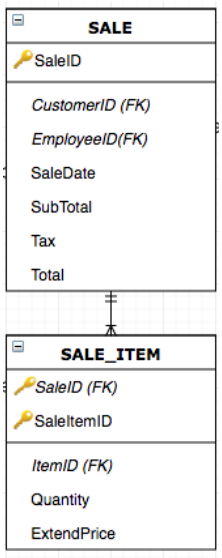
| SALE_ITEM Is Required Child | Action on SALE (Parent) | Action on SALE_ITEM (Child) |
|---|---|---|
| **Insert** | INSERT trigger on SALE to create row in SALE_ITEM. SALE_ITEM will be given data for SaleID. Disallow SALE insert if SALE_ITEM data are not available. | Will be created by INSERT trigger on SALE. |
| **Modify key or foreign key** | Prohibit – surrogate key | Prohibit – SALE_ITEM must always refer to the SALE associated with it. |
| **Delete** | Prohibit – data about a sale item and its related transaction is never deleted (business rule). | Prohibit – data related to a transaction is never deleted (business rule). |

**G. Describe how you have represented weak entities, if any exist.**

1. CUSTOMER is a weak entity, because its existence depends on PERSON. PersonID in CUSTOMER is a foreign key referencing PersonID in PERSON. CUSTOMER is ID-dependent on PERSON and therefore PersonID is the primary key of CUSTOMER.
2. EMPLOYEE is a weak entity, because its existence also depends on PERSON. PersonID in EMPLOYEE is a foreign key referencing PersonID in PERSON. EMPLOYEE is ID-dependent on PERSON and therefore PersonID is the primary key of EMPLOYEE.




3. SALE_ITEM is a weak Entity, because its existence depends on SALE. SaleID in SALE_ITEM is a foreign key referencing SaleID in SALE. SALE_ITEM is ID-dependent on SALE and therefore SaleID is also included in the identifiers of SALE_ITEM.



**H. Describe how you have represented supertype and subtype entities, if any exist.**

Supertype: PERSON

Subtype (inclusive): CUSTOMER & EMPLOYEE

The identifier of the supertype becomes the primary key and foreign key of each subtype. Here the primary key of the supertype PERSON (i.e. PersonID) is both the primary key and foreign key in the subtype CUSTOMER or EMPLOYEE.