Assignment 2 - 26 September 2020

Identify keys (superkeys, keys, and candidate keys)

Order

For “Order” table I have chosen `orderID` as my candidate key, `orderID`, is also my primary key in the “Order” table. This is because since `orderID` is primary key, it will automatically incremented as new item is added into the “Order” table. Therefore, there will be no same `orderID` number in the table. I have chosen `dateOfReceipt` and `actShipDate` as my superkeys, since the receipt date and actual shipping date should be different. This is since, the company cannot ship the product on the same day when they have receive an order. There will be some process have to be done and other shipments must be completed before. Since, `dateOfReceipt` and `actShipDate` will be different in every row, therefore these two columns will be unique and suitable for superkeys.

Part

For “Part” table I have chosen `partID` and `Name` as my candidate key. The column `partID` is the primary key for this table and `Name` for all and it can be used for primary key to identify the tuples. `Name` will be different for every tuples, therefore it is unique. For superkeys, I have chosen `Name` and `Price` since it will be unique, that there will be no repeated values in the same row. Every tuples will have different values in the same row. Since `PartID` is unique as well, therefore it is superkey as well. For **key**, `Name` can be used, since every part has different name, which it will be unique for every tuples in the `Name` column. Since every part has different name, the `Name` tuple will be unique as well as there will be no two identical tuples with value holding for that item.

Employee

For “Employee” table I have chosen `employeeID` and `ZIP` as my candidate key. `employeeID` is my primary key from this table, since `ZIP` also can be a primary key. Since `ZIP` tuple is different for every tuples in the “Employee” table and it is unique, which is suitable for candidate key. For superkey, I have chosen `ZIP`, since it will be unique, that there will be no repeated values in the same row. `ZIP` must be different since every employees are living in different house, which it must be different and unique, it can define tuple individually with `ZIP`. For **key**, `ZIP` can be used, since every employee has different ZIP code.

Therefore, it will be unique for every tuples in the `ZIP` column. Since every employee has different ZIP code, the `ZIP` tuple will be unique as well as there will be no two identical tuples with value holding for that item.

Customer

For "Customer" table I have chosen `customerID` and `ZIP` as my candidate key. `customerID` is my primary key from this table, since `ZIP` also can be a primary key. Since `ZIP` tuple is different for every tuples in the "Customer" table and it is unique, which is suitable for candidate key. For superkey, I have chosen `ZIP`, since it will be unique, that there will be no repeated values in the same row. `ZIP` must be different since every customers are living in different house, which it must be different and unique, it can define tuple individually with `ZIP`. For **key**, `ZIP` can be used, since every customer has different ZIP code, which it will be unique for every tuples in the `ZIP` column. Since every customer has different ZIP code, the `ZIP` tuple will be unique as well as there will be no two identical tuples with value holding for that item.

Describe attribute data types and domains; Are there any restricted data types?

Order

In "order" table, for `orderID`, `empID`, `custID` and `partID`, I am using `INT` data type. Since `orderID` is a unique number to identify every orders. Since `empID`, `custID` and `partID` are foreign keys which is linked "employee", "customer" and "part" tables, which they are primary key in the original table. For `dateOfReceipt`, `expShipDate` and `actShipDate`, I am using `DATE` data type. Since, they will store the dates, it must be date data type.

Part

In "part" table `partID`, `Stock` and `Quantity` I am using `INT` data type. Since `partID` is a primary key which it must be `INT`, with `Stock` and `Quantity` since they are only can be represented in integers, which they will also have `INT` data type. With `Name` I am using `VARCHAR(15)` since, very part will have different name length therefore I am using `VARCHAR` data type. The number `15` is the maximum number of characters that the `Name` column can store. With part name, I don't it is necessary to have more than length of 15 characters, therefore I have set domain to 15. Since, there may be character length of more than 15, in this case this is a restricted data type. For `Price` I am using `DECIMAL`, since price will not be a whole number, there will be decimal places of up to two. However, since MySQL Workbench does not have `MONEY` data type therefore I have set to `DECIMAL`. Since with `DECIMAL` the number can be extremely big as well as more than two decimal places, this is a restricted data type.

Employee and Customer

In "Employee" and "Customer" tables, for `employeeID` and `customerID` I am using `INT` data type. Since `employeeID` and `customerID` a primary key which it must be `INT` to be able to identify every employee and customer uniquely. With `firstName` and `lastName` I am using data type of `VARCHAR(45)`. Since considering people from many around the country, if they write their name in alphanumeric, it may become really long, such as Japanese name. Therefore, I have set the maximum characters that can be stored into be length of `45`. Also every people have different length of name, therefore I have chosen to use `VARCHAR` data type. With `ZIP` I have used `VARCHAR` data type as well but the maximum character input to be `15`. Since, every country have different ZIP code format and length therefore, I am using `VARCHAR` data type. With `firstName`, `lastName` and `ZIP`, they are all restricted data type. Since specified amount of characters may not fit in the database which some employees and customers may encounter that they cannot store their information.

Find best suited foreign keys, stating assumptions and explaining reasons

For foreign keys I am using `partID`, `employeeID` and `customerID`. They are all primary keys from every table. Since primary key is the only uniquest identification to look up each item from the table. Therefore, by using primary key as a foreign key in "Order" table, this allows us to identify which employee's job is to handle this order, from which customer order was made and which part was ordered, just by executing a simple SQL statement.

What types of constraints would you expect to check on those relations?

I would make sure there is **referential integrity constraint**. Since “Order” table is linked between “Part”, “Employee” and “Customer” tables and foreign key is the primary key from every table, the value must be same between each table. Otherwise, all the data will become mess and will not be able to proceed right order for the customer. Secondly, I need to make sure **entity integrity constraint**, which make sure that no primary key value can be NULL. Since primary key is provided to be able to identify each item from the table and provided for use of foreign key in “Order” table. If the primary key is NULL, it will not be possible to identify every item uniquely from the table and will not be able to link between tables. Thirdly, I need to take into account “semantic integrity constraints”, which reflects real world situation. For example, with `firstName` and `lastName` I need to make sure what is the realistic length considering about people’s name from other nationalities. Since if non-English names such as Japanese, Korean and Chinese names, if it is converted into alphanumeric it will become really long, therefore taking this into account I need to consider what is the realistic and suitable maximum characters to be stored in every tuples.

Which of those constraints are key, entity integrity, and referential integrity constraints, and which are not?

Order

From “Order” table, `orderID`, `empID`, `custID` and `partID` are **entity integrity constraints**, since `orderID` is a primary key of the table therefore cannot be NULL. With other three keys, they are foreign key which they are primary key in the original table, therefore they cannot be NULL as well. I would make sure there is **referential integrity constraint**. Since “Order” table is linked between “Part”, “Employee” and “Customer” tables and foreign key is the primary key from every table that is linked, the value must be same between each table. Otherwise, all the data will become mess and will not be able to proceed right order for the customer. Since `dateOfReceipt`, `expShipDate` and `actShipDate` are just dates, which they cannot be used for as a key.

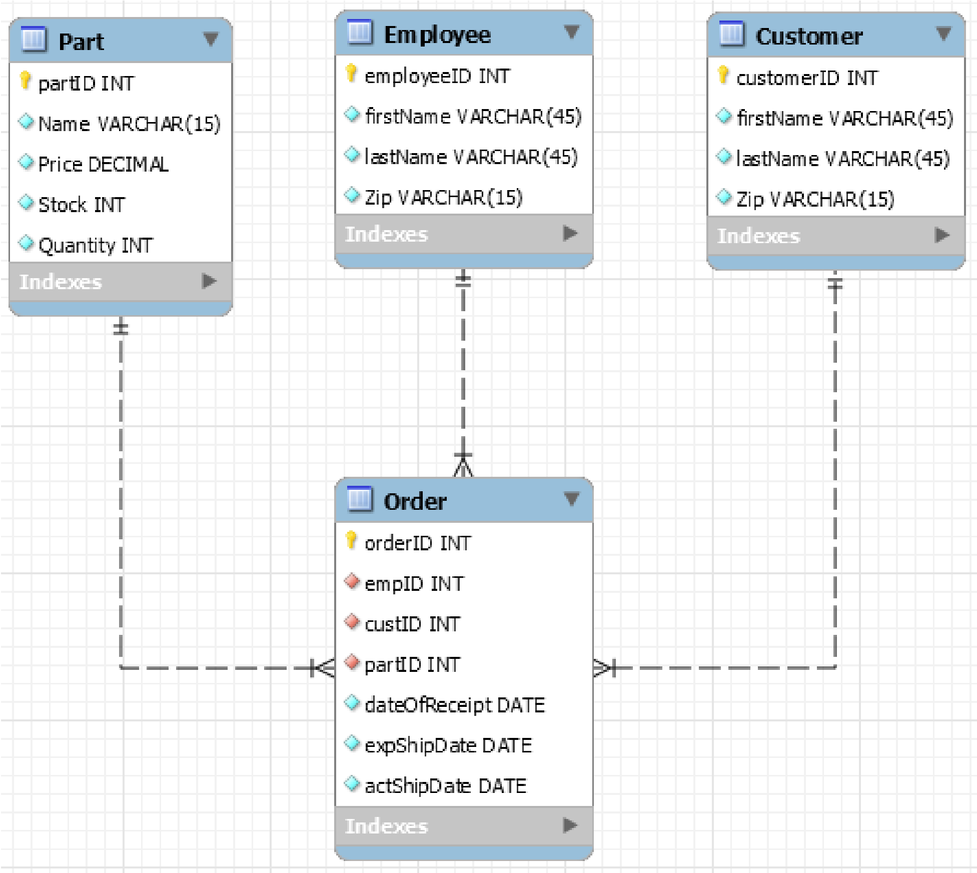
Part

From “Part” table, `PartID` is **entity integrity constraint**, since it is a primary key to identify the part from the table, which it cannot be NULL. Moreover, since `partID` is a foreign key in “Order” table, it is **referential integrity constraint**, in order to maintain the consistency between two tables. `Name` can be used as a **key**, since every part have different name, therefore it will be able to uniquely identify each part from the table. Since `Price`, `Stock` and `Quantity` can have same values repeating for that item, will not be unique since there might be other part having same values and these values can change.

Employee and Customer

From “Employee” and “Customer” tables, `employeeID` and `customerID` are **entity integrity constraints**, since they primary keys to identify each employee and customer from the tables, which it cannot be NULL. Moreover, since `employeeID` and `customerID` are foreign keys in “Order” table, they are **referential integrity constraints**, in order to maintain the consistency between tables. `ZIP` can be used for as a **key** as well, since they are unique and and has not redundancy, which no two tuples with identical values for that certain item.

Choose a primary key for each relation and mark it on the diagram



In the image above, the vales with a light bulb next to it are the primary keys. The values with orange diamond next to it are the foreign keys.