Project 2 Report

Option 2: Inventory Database

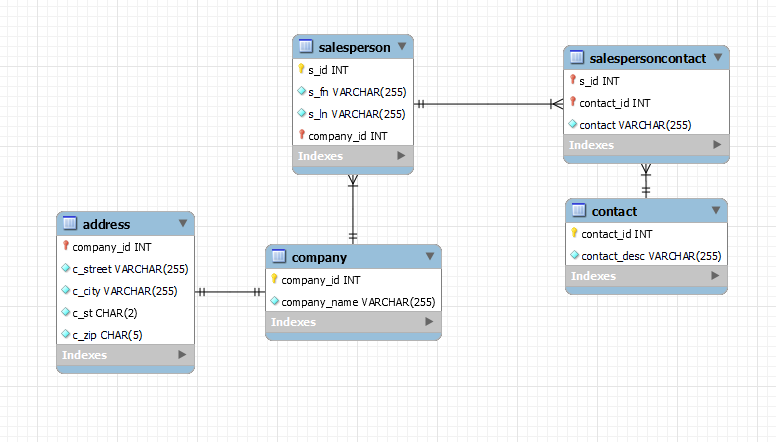
By: Harrison Blair

# Vendors Database

## Design

The Vendors Database is used to hold all of the data associated with the different vendors, as well as their contacts, and contact information. The “company” table is the parent table, holding only a company ID as well as the name of the company. The company table shares a one-to-one identifying relationship with the address table, due to the restriction that each company only has one address. The remainder of the address table is used to store the street, city, state, and zip code of the respective company. The company also shares a one-to-many identifying relationship with the salesperson table. This table holds the first and last name of the salesperson, as well as a unique salesperson ID number. The next table is the contact table, which holds a contact ID, as well as what the ID represents. This table is used in order to differentiate the different contacts that a salesperson could have, being a phone number or an email address. Since a salesperson can have multiple different types of contacts, the salespersoncontact table has been created in order to resolve the many-to-many relationship between the salesperson table and the contact table. The two primary keys of the salesperson table, s\_id, and the contact table, contact\_id, are used as foreign keys in the salespersoncontact table. The salespersoncontact table also holds the actual contact information, being a phone number or email address.

## Vendors ERD

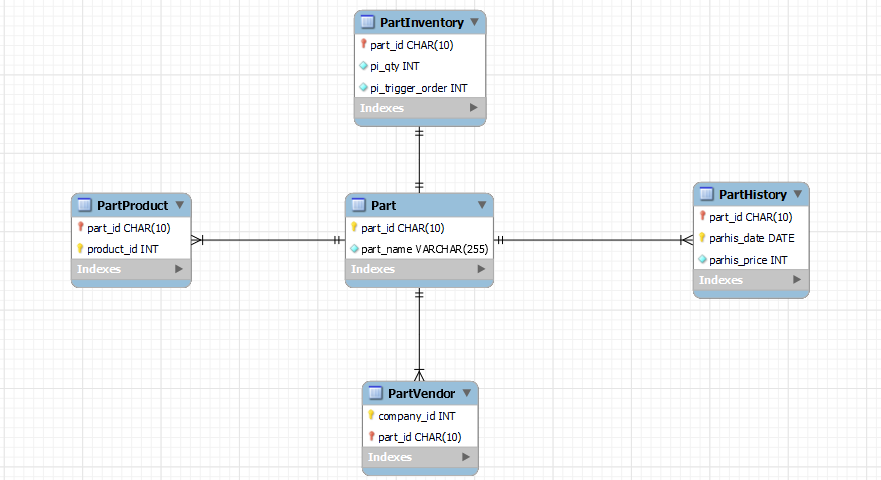


# Parts Database

## Design

The parts database is tasked with holding all the information related to parts at the bike store, being their inventory level, what vendor they are from, price and price history, as well as what products the parts are used in. The first table, the parent table, is the part table. This table holds the part ID, as well as the name of the part. The part\_id is used as a foreign key in every other table in the database, due to the part\_id being a unique identifier for each part. Multiple vendors are able to sell the same part, which is why the part\_name field would not act as a good foreign or primary key. In the PartInventory table, the part\_id is associated with a part’s current inventory level, as well as the specified quantity of a given part when a reorder should be made. This table shares a one to one relationship with the parent part table, since there is no tracking available for the part inventory levels. The next table is the part history table, which shares a one to many relationship with the part table because there are multiple histories of prices available for each part\_id. This table contains the date of the price change, as well as the price on that given date. The next table is the PartVendor table, which handles relating the parts with the different vendors in the vendor database. This table holds the part\_id as a foreign key, as well as the part’s associated company id. This table also shares a one-to-many relationship with the parent part table, due to the specification that multiple vendors may be able to supply the same part. The final table is the part product table, which holds data related to what products that parts are compatible with. This table has the part\_id acting as a foreign key, and the product\_id acting as a primary key. This table has a one-to-many relationship with the part table, because it may be possible in the future that a part is able to be used on multiple different products, and having an inability to look up exactly what products it is compatible with would be annoying.

## Parts ERD



# Products Database ERD

## Design

The Products database is used to store all of the product inventories, prices, and information. The first table, and parent table, is the product table. This table holds three variables, a product\_id, a product\_name, and a product\_desc. The product ID consists of an integer identification number, which is unique to all products, the product ID also acts as the primary key in this table. The product\_name variable is also unique, due to our assumptions telling us that every product must have a unique name. The product\_desc contains a short description of the product. The next table is the productpricehistory table, which shares a one-to-many relationship with the product table. This is due to as stated above we are dealing with price histories, so there will be multiple entries per product\_id. The product\_id acts as a foreign key in this table, which also has its own primary key in the form of a DATE variable named pph\_date. This variable helps to coincide which date change for which product is associated with which price. The price is the third variable in the table, being an integer. The last table in the products database is the productinventory table, which holds the history of a product’s inventory. This table has a one-to-many relationship with the products table with the product\_id as a foreign key, and similarly to productpricehisory has its own primary key consisting of a DATE variable, this time named pi\_date. This table also holds the product quantity in pi\_qty, as well as the trigger order amount held in pi\_trigger order.

## Products ERD

