Item list is a collection with the following fields.

UPC, is a string that serves as one of the ID for individual products

• item\_id, is a string that serves as one of the ID for individual products

• Status, is a numeric that tells if an item is active, suspended, or deleted

• RootID, is a string that is potentially used to tie several items to a single item

• LongDesc, is a string that provides a complete description of an item

• ShortDesc, is a string that provides a shorter description

• ClassCode, is a numeric that groups products together

• ClassDesc, is a string that describes the class

• CategoryCode is a numeric that groups classes together

• CategoryDesc is a string that describes the class

• FamilyCode is a numeric that groups categories together

• FamilyDesc is a string that describes the family

• DepartmentCode is a numeric that groups families together

• DepartmentDesc is a string that describes the department

• StoreBrand is a string that is “Y” or “N” depending on if item is a store brand

• ExtraDes is a string that gives an additional description. Could be country of origin, packaging information, or other attribute

sales\_details is a collection of the following fields:

• store\_num is a numeric ID for stores

• register is a numeric ID for a check-out counter. Only unique within each store

• trans\_num is a numeric ID for a transaction. Range between 0 and 9999 and repeat within days

• trans\_date is a string for transaction date

• trans\_time is a string of the hour, minute, and second of a customer transaction scanned

• bus\_date is a string for the business date of the transaction

• UPC is a string that is an ID for individual products

• item\_id, is a string that serves as one of the ID for individual products

• dept\_num is a numeric ID assigned to each department in a store

• item\_quant is a numeric measure of product units purchased. Integers and decimals possible

• weight\_amt is a numeric measure of the weight of an item

• sales\_amt is a numeric measure of the amount of the sale

• cost\_amt is a numeric measure of the cost of the sale

• cashier\_num is a numeric ID for a cashier

• price\_type is a string for a price promotion

• service\_type is a string that describes if a purchase was in-store or online

• tender\_type is a string describing the payment type

• loyalty\_card\_no is a numeric ID assigned to participating customers

customer\_details is collection of the following fields

• loyalty\_card\_no is a numeric ID assigned to participating customers

• household\_no is a numeric ID assigned to customers to identify their household

• member\_fav\_store is a numeric ID of the customer’s favorite store

• city is a string of the city the customer lives in

• state is a string of the state a customer lives in

• zip is a numeric of the zip code of a customer

item\_attribute is a collection of the following fields

• UPC is a string that is an ID for individual products

• item\_pos\_desc is a string that gives a more complete description of an item

• item\_attribute\_desc is a string that contains values like, “Made in Texas”, “Gluten Free”, or others

• item\_attribute\_value is a string that is “Y” or “N” in regards to the attribute

• attribute\_start\_date is a string of the start date of an attribute

• attribute\_end\_date is a string of the end date of an attribute

weather\_info is a collection of the following fields

• zip is a number storing zip address

• date\_time is a dateTime object

• date\_time\_str is a string storing datetime as a string

• tempF is a float storing temperature in Fahrenheit

• tempC is a float storing temperature in Celsius

• fl\_F is a float storing feels like in Fahrenheit

• fl\_C is a float storing feels like in Celsius

• rh is a float describing relative humidity

• windDir is a string describing the direction of wind

• windSp is a number telling us about wind speed

• windGust is a number telling us about wind gust

• dewPt is a number telling us about dew point

• visis a number telling us about visibility

• press\_in is a number telling us about pressure

• press\_mmHg is a number telling us about pressure in mm

• press\_tend is a string telling us about pressure

store\_addresses is a collection of the following fields:

• store\_num is a number identifying store

• street\_num is a number identifying street no

• street is a string storing name of the street

• unit\_num

• city is a string storing name of the city

• state is a string storing name of the state

• state\_abb is a string storing abbreviation of the state name

• zip is a number storing zip address

• store\_type is a string and it stores store type

Scraping\_store is a collection of the following fields:

• store\_name is a string storing name of the store

• store\_id is a number identifying a particular store

• location\_name is a string telling the location name

• state\_abb is a string storing abbreviation of the state name

• zip is a number storing zip address

• service\_name is a string that stores name of the service

• service\_value is a string that stores values like “True”

store\_list is a collection of the following fields:

• store\_nbr is a number identifying a particular store

• store\_name is a string storing name of the store

• actv\_rec\_ind is a string storing values like “Y”

• store\_addr\_line\_1 is a string storing the first address line

• store\_city\_nm is a string storing the name of city

• store\_state\_prv\_Cd is a string storing the state code

• store\_post\_cd is a number storing zip code

• store\_sq\_feet is a number storing the area of the store

• store\_rgn\_desc is a string storing the store region

• store\_clst\_desc is a string storing the store description

Relationships

• In Item list, UPC and Item\_id together becomes a unique key for the products. Likewise UPC is again found in the sales details table; hence both the tables are related to each other thru the UPC.

• Likewise UPC is also found in item attribute table for the common connection between the above two tables.

• Similarly loyalty card no in customer details table and sales details table are connected thru the loyalty card no key.

• Weather\_info stores weather details for locations which can be joined on date field of sales\_details

• Store\_list, store\_address contain details about the store which can be used for analysis

Process Flow

- Firstly we researched which scenarios would need embedding or reference

- We did requirement gathering from all the relevant stakeholders

- Created the requested schema

- We also created a hierarchy table, using the UPC and item\_id, which is unique for every product in a separate collection

- Helped the web scraping team to insert data into mongodb

- We researched on index , understanding the pros and cons of the same and then decided to make index on UPC key column

- We created steps to import and export the DB so that people can easily share their databases and start next step quickly

- Modified the tables with all the new updates from the stakeholders

Collections

# Item\_List Collection: # UPC is a string

{

"UPC" : "1238573728",

"item\_id" : "12345671234",

"Status" : 0, "RootID" : "str",

"LongDesc" : "str",

"ShortDesc" : "str", "ClassCode" : 213456789876,

"ClassDesc" : "PURGE", "CategoryCode" : 5345665487, "CategoryDesc" : "WHOLE HEALTH", "FamilyCode" : 923,

"FamilyDesc" : "WELLNESS", "DepartmentCode" : 11, "DepartmentDesc" : "HBA", "StoreBrand" : "N", "ExtraDes" : "DEFAULT"

}

# hierarchy\_item\_list # UPC is a string

{

"UPC": "00003701426401",

"item\_id" : "12345671234",

"DepartmentCode" : 7, "DepartmentDesc" : "deptInfo", "Family" : {"FamilyCode" : 3,

"FamilyDesc" : "FamilydesInfo", "Category" : {"CategoryCode" : 7,

"CategoryDesc" : "categoryInfo", "Class" : {"ClassCode" : 5,

"ClassDesc" : "classInfo"

}

}

}

# weather\_info collection

# Give the date either as a string or a Date object

{

"zip" : 79415,

"date\_time" : "DateObject", "date\_time\_str" : "string", "tempF" : 75.23,

"tempC" : 23.4,

"fl\_F" : 65.89,

"fl\_C" : 23.4,

"rh" : 70.67,

"windDir" : "North",

"windSp" : 45.65,

"windGust" : 41.678,

"dewPt" : 20.5,

"vis" : 5.23,

"press\_in" : 5.675,

"press\_mmHg" : 34.55, "press\_tend" : "str"

}

# sales\_details # UPC is a string

{

"store\_num" : 500,

"register" : 12,

"trans\_num" : 996789880, "trans\_date\_time\_gmt" : "gmt time", "trans\_date\_time\_local" "string", "bus\_date" : "YYYY-mm-dd",

"UPC" : "1265478954",

"item\_id" : "12345671234",

"dept\_num" : 5,

"item\_quant" : 1.000,

"weight\_amt" : .421,

"sales\_amt" : .69,

"cost\_amt" : .28,

"cashier\_num" : 69,

"price\_type" : "Manager Special", "service\_type" : "In-Store", "tender\_type" : "CASH", "loyalty\_card\_no" : 51001

}

# customer\_list

{

"loyalty\_card\_no" : 51001,

"household\_no" : 3898347,

"member\_fav\_store" : 524, "city" : "Claude",

"state" : "Texas", "zip" : 79019

}

# item\_attribute # UPC is a string

{

"UPC" : "00000000000210",

"item\_pos\_desc" : "10LB BAG POTATOES", "item\_attribute\_desc" : "Made in Texas", "item\_attribute\_value" : "N", "attribute\_start\_date" : "2016-07-22",

"attribute\_end\_date" : "2016-08-19"

}

# store\_addresses

{

"store\_num" : 525,

"street\_num" : 2806,

"street" : "South Main Street", "unit\_num" : NA,

"city" : "Perryton",

"state" : "Texas",

"state\_abb" : "TX", "zip" : 79070

"store\_type" : "United Supermarkets"

}

# Scraping\_Rupam

{

"store\_name" : "United Supermarkets", "store\_id" : 545,

"location\_name" : "abc" "state\_abb" : "TX", "zip" : 79601,

"service\_name" : "Walk-in Clinic", "service\_value" : "True"

}

# store\_list

{

"store\_nbr" : 503, "store\_name" : "LBK-N UNIV", "actv\_rec\_ind" : "Y",

"store\_addr\_line\_1" : "112 N. UNIVERSITY", "store\_city\_nm" : "LUBBOCK", "store\_state\_prv\_Cd" : "TX", "store\_post\_cd" : 79417,

"store\_sq\_feet" : 51133, "store\_rgn\_desc" : "LUBBOCK CITY", "store\_clst\_desc" : "AMIGOS"

}

# Exporting a database as json:

# Connect mongod server and ensure that mongo is working by viewing current dbs and tables # In a new Command Prompt window, type following -

mongodump --db united\_supermarket\_db --out C:\Users\rudas\Downloads\testing.json # This will export entire DB in the specified location, ready to be shared

# Importing a DB from the dump File

# Connect to mongoDB and ensure that it's running

# In a new Command Prompt window, type following -

mongorestore --db NameOfNewDB C:\Users\rudas\Downloads\testing.json\united\_supermarket\_db

# Note that we need to give path to the folder which actually contains metadata and .bson files