MITx | MicroMasters™
in Supply Chain Management

MIT Center for Transportation & Logistics

CTL.SC4x – Technology and Systems

Key Concepts Document

This document contains the Key Concepts for the SC4x course, Weeks 1 and 2.

These are meant to complement, not replace, the lesson videos and slides. They are intended to be references for you to use going forward and are based on the assumption that you have learned the concepts and completed the practice problems.

The first draft was created by Dr. Alexis Bateman in the Winter of 2017.

This is a draft of the material, so please post any suggestions, corrections, or recommendations to the Discussion Forum under the topic thread "Key Concept Documents Improvements."

Thanks,

Chris Caplice, Eva Ponce and the SC4x Teaching Community
Winter 2017 v1

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Introduction to Data Management

Summary

Supply chains are moving at ever-faster rates with technology and systems supporting this movement. Data management plays a critical role in enabling supply chains to move at the speed and precision they do today, and the need for advanced data management will only continue.

In recent years there has been an explosion of information and this is especially true in supply chains. A few examples introduced include Amazon's massive supply chains selling 480 million unique items to 244 million customers while UPS is delivering 20 million packages to 8.4 million delivery points. This information is coming from multiple sources, in addition to sensors, the "internet of things", and regulations requiring increasing amounts of information.

All of this information is commonly referred to as the "Big Data" challenge. Data is driving our modern world, but how can we be sure of it and use it most effectively. As we will review – data is messy, it requires cleaning and programming. Data is frequently trapped in siloes coming from different sources, which makes it more challenge to work with. In addition, data is big and getting even bigger daily. The tools we have all become comfortable with (spreadsheets) can no longer handle that amount of data, so we must use different tools to enable greater analysis.

To better understand the role of data and how to manage it, the following summaries cover an introduction to data management, data modeling, and data normalization – to get us started on a solid ground with handling large data sets – an absolute essential in supply chain management.

Data Management

In data management supply chain managers will be faced with immense complexity. This complexity is influences by the volume (how much), velocity (pace), variety (spread), and veracity (accuracy). Each of these components will influence how data is treated and used in the supply chain.

There are several reoccurring issues that supply chain managers must be aware of as they are working with data:

- Is the data clean?
- Is the data complete?
- What assumptions are you making about the data?
- Are the results making sense? How can I check?

Cleaning data is one of the most important; yet time consuming process in data analysis. It can greatly influence the outcome of analysis if not completed properly. Therefore – SC

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

professionals should always plan enough time for basic data checks (meaning if you get garbage in, you will get garbage out).

There are several typical checks you should always look for:

- **Invalid values** - negative, text, too small, too big, missing
- **Mismatches between related data sets** - # of rows, # of cols
- **Duplication** – unique identifiers
- **Human error** – wrong dates, invalid assumptions
- **Always explore the outliers** – they are the most interesting!

When cleaning data, you should be organized. This means we must make sure to version the documents we are working with and keep track of data changes.

Querying the Data

Once you have a clean and organized set of data, querying the data can make data extremely powerful. Querying data refers to the action of retrieving data from your database. Because a database can be so large – we only want to query for data that fits a certain criteria.

There are several basic options that can help you get some quick answer in big data sets, such as using Pivot Tables:

- There is data summarization tools found in LibreOffice, Google Sheets, and Excel
- They automatically sort, count, total or average the data stored in one table or spreadsheet, displaying the results in a second table showing the summarized data.
- Very useful in tabulating and cross-tabulating data

No more spreadsheets!

Unfortunately, as we dive deeper into the big data challenge, we find that spreadsheets can no longer service all of our needs. We have the choice of working with structured or unstructured data. A database is a structured way of storing data. You can impose rules, constraints and relationships on it. Furthermore it allows for:

- **Abstraction**: Separates data use from how and where the data is stored. This allows systems to grow and makes them easier to develop and maintain through modularity.
- **Performance**: Database may be tuned for high performance for the task that needs to be done (many reads, many writes, concurrency)

Spreadsheets are unstructured data. You have a data dump into on spreadsheet and you need to be able to do lots of different things. Spreadsheets will always be great for a limited set of analysis such as informal, causal, and one-off analysis and prototyping. Unfortunately they are no longer suited for repeatable, auditable, or high performance production. Unstructured data commonly has problems with: redundancy, clarity, consistency, security, and scalability.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Learning Objectives

- Understand the importance of data in supply chain management.
- Review the importance of high quality and clean databases.
- Recognize the power of querying data.
- Differentiate between unstructured and structured data and the need for tools beyond spreadsheets.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Data Modeling

Summary

Now that we have been introduced to data management and the issue of big data, we now deep drive into data modeling where we learn how to work with databases. Data modeling is the first step in database design and programming to create a model for how data relates to each other within a database. Data modeling is the process of transitioning a logical model into a physical schema.

To understand the process of data modeling, we review several components including relational databases, data organization, data models for designing databases, and what constitutes a good data model. A data model consists of several parts including: entities and attributes, primary keys, foreign keys, and relationships and cardinality.

Relational Models

The relational model is an approach to managing data that uses structure and language where all data is grouped into relations. A relational model provides a method for specifying data and queries. It is based on first-order predicate logic, which was described by Edgar F. Codd in 1969. This logic defines that all data is represented in terms of tuples, grouped into relations. There are several definitions to be familiar with as we reviewed previously with relational models:

- **Entity**: object, concept or event
- **Attribute** (column): a characteristic of an entity
- **Record or tuple** (row): the specific characteristics or attribute values for one example of an entity
- **Entry**: the value of an attribute for a specific record
- **Table**: a collection of records
- **Database**: a collection of tables

Tables and Attributes

Data in relational tables are organized into tables, which represent entities. Single tables within a database can be seen as similar to a spreadsheet. However, we use different words to refer to "rows" and "columns". Attributes are the characteristics of an entity.

Tables

- Tables represent entities, which are usually plural nouns
- Tables are often named as what they represent (typically plural nouns, without spaces): e.g. Companies, Customers, Vehicles, Orders, etc.

Attributes

- Characteristics of an entity (table), typically nouns
- Examples in the form of: Table (Attr1, Attr2, ... AttrN), Vehicles (VIN, Color, Make, Model, Mileage)

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Entity Types and Entity occurrence: an entity is any object in the system we want to want to model and store. An entity occurrence is a uniquely identifiable object belonging to an entity type.

Designing Data Models

There are several steps to designing a database to store and analyze data.

1. Develop a data model that describes the data in the database and how to access it
2. Data model defines tables and attributes in the database (each important concept/noun in the data is defined as a table in the database)

Data models help specify each entity in a table in a standardized way. They allow the user to impose rules, constraints and relationships on the data that is stored. It also allows users to understand business rules and process and analyze data.

Rules for a Relational Data Model

There are several rules for relational data models:

- Acts as a schematic for building the database
- Each attribute (column) has a unique name within a table
- All entries or values in the attribute are examples of that attribute
- Each record (row) is unique in a good database
- Ordering of records and attributes is unimportant

What makes a good relational data model? A good relational model should be complete with all the necessary data represented. There should be no redundancy. Business rules should be effectively enforced. Models should also be reusable for different applications. And finally, it should be flexible and be able to cope with changes to business rules or data requirements.

Relationships and Cardinality

When we begin to work with the data – we have to understand how data relates to each other and data uniqueness of the attributes. Some of this can be managed through entity types and attributes. Relationships + cardinality = business rules.

Entity and Attributes

An entity is a person, place, thing, or concept that can be described by different data. Each entity is made of a number of attributes, which make up that entity. Entity types should be described as part of the data modeling process, this will help with the documentation and determination of business rules.

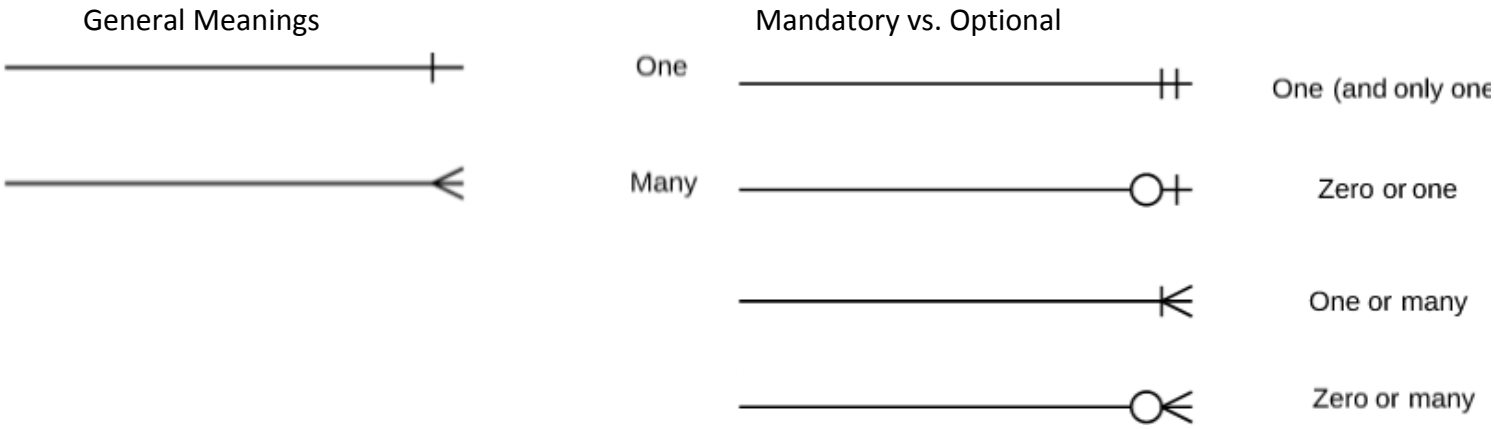
How to draw and entity-relationship diagram:

An ERD is a graphical representation of an information system that visualizes the relationship between the entities within that system.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

- ERD or entity-relationship diagram is a schematic of the database
- Entities are drawn as boxes
- Relationships between entities are indicated by lines between these entities
- Cardinality describes the expected number of related occurrences between the two entities in a relationship and is shown using crow's foot notation (see figures below)

Cardinality – crow's foot notation



Domain Validation Entities: Also known as pick lists or validation lists. Domain validation entities are used to standardize data in a database, they restrict entries to a set of specified values. They are a table with a single attribute that enforces values of attribute in related table.

Keys

Primary keys are attributes used to uniquely identify a record while foreign keys are attributes stored in a dependent entity, which show how records in the dependent entity are related to an independent entity.

Primary key: one or more attributes that uniquely identify a record. The attribute has be uniquely suited.

Foreign Key: Primary key of the independent or parent entity type is maintained as a non-key attribute in the related, dependent or child entity type, this is known as the foreign key

Composite key: is a primary key that consists of more than one attribute, ex: charter airline, every flight has a different number.

Many to Many Relationships: A many to many relationship refers to a relationship between tables in a database when a parent entity contains several child entity types in the second table. ex- Vehicle can be driven by many drivers, drivers can drive many vehicles. In this case an associative table (entity), aka junction table is appropriate where the primary key of parent is used in primary key of child.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Referential integrity

Referential integrity maintains the validity of foreign keys when the primary key in the parent table changes. Every foreign key either matches a primary key (or is null).

Cascade rules: choose among delete options

- Cascade restrict: Rows in the primary key table can't be deleted unless all corresponding rows in the foreign key tables have been deleted
- Cascade delete: When rows in the primary key table are deleted, associated rows in foreign key tables are also deleted

Learning Objectives

- The data model describes the data that is stored in the database and how to access it.
- Data models enable users to understand business rules and effectively process and analyze data.
- Understand that business rules are imposed on the database through relationships and cardinality.
- Recognize that data models may vary for a given dataset as business logic evolves.
- Remember that the data modeling process may reveal inconsistencies or errors in the data, which will have to be corrected before importing into a database.
- Selection of entities and associated attributes from a flat file is not always obvious.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Database Normalization

Summary

Database normalization, or normalization, is an important step in database management. Normalization is intrinsic to relational databases and is the process of organizing attributes into relations (or tables). This process is vital in reducing data redundancy and improving data integrity. In addition, normalization helps organize information around specific topics that can be used to digest the massive amount of information in databases into something digestible.

When SC professionals are presented with large amounts of raw data, that raw data may be in stored in a single table, containing redundant information or information about several different concepts. The data can be separated into tables and normalized to allow for better data handling and comprehension. To get to this place, updating a data model can be done collaboratively during meetings and discussions to define the business rules. During updates, normalization prevents mistakes and data inconsistencies. Normalization helps prevent redundancy, confusion, improper keys, wasted storage, and incorrect or outdated data.

Objectives of Normalization

1. To free the collection of [tables] from undesirable insertion, update and deletion dependencies.
2. To reduce the need for restructuring the collection of [tables], as new types of data are introduced, and thus increase the life span of application programs.
3. To make the relational model more informative to users.
4. To make the collection of [tables] neutral to the query statistics, where these statistics are liable to change as time goes by.

****Remember our relational model definitions**

- **Entity**: object, concept or event
- **Attribute** (column): a characteristic of an entity
- **Record or tuple** (row): the specific characteristics or attribute values for one example of an entity
- **Entry**: the value of an attribute for a specific record
- **Table**: a collection of records
- **Database**: a collection of tables

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Summary of five normal forms

1. All rows in a table must contain the same number of attributes; no sub-lists, no repeated attributes.
2. All non-key fields must be a function of the key.
3. All non-key fields must not be a function of other non-key fields.
4. A row must not contain two or more independent multi-valued facts about an entity.
5. A record cannot be reconstructed from several smaller record types.

Normal Forms

First Normal Form – the basic objective of the first normal form (defined by Codd) is to permit data to be queried and manipulated, grounded in first order logic. All rows in a table must contain the same number of attributes; no sub-lists, no repeated attributes, identify each set of related data with a primary key. *First normal form can make databases robust to change and easier to use in large organizations.*

Second Normal Forms – must first be in first normal form, all non-key fields must be a function of the primary key; only store facts directly related to the primary key in each row.

Third Normal Form - must first be in second normal form. All the attributes in a table are determined only by the candidate keys of the table and not by any non-prime attributes. Third normal form was designed to improve database processing while minimizing storage costs.

Fourth Normal Form - must first be in third normal form. A row should not contain two or more independent, multi-valued facts about an entity. Fourth normal form begins to address several issues when there is uncertainty in how to maintain the rows. When there are two unrelated facts about an entity, these should be stored in separate tables.

Fifth Normal Form - must first be in fourth normal form. A record cannot be reconstructed from several smaller record types. Size of this single table increases multiplicatively, while the normalized tables increase additively. Much easier to write the business rules from the three tables in the fifth normal form, rules are more explicit. *Supply chains tend to have fifth normal form issues.*

Normalization Implementation Details

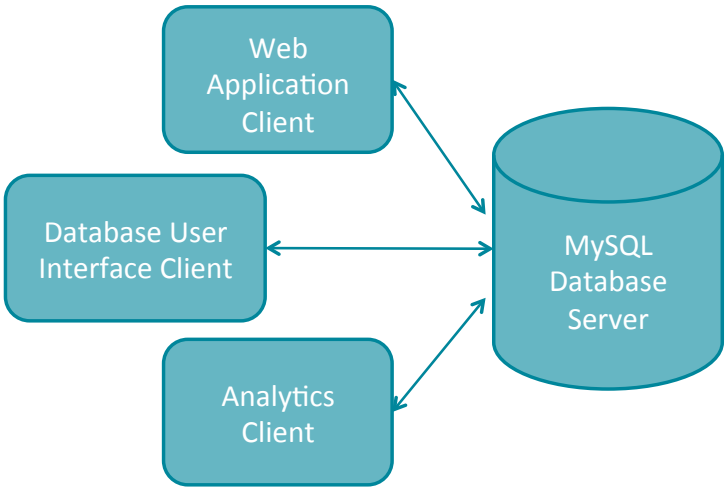
Normalization ensures that each fact is stored in one and only one place, to ensure data remains consistent. Normalizing the data model is a technical exercise. It does not change business rules! However, through the process of meetings and decisions it may help the rules be further defined through review. Care in data normalization is needed to preserve data quality. There are times when normalization is not an option – this happens when there are large, read only databases for report generation of data warehouses.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Client Server Architecture

Client Server Model

Clients can connect to servers to access a specific service using a standardized protocol, see figure below.



Database Servers

Databases are hosted on a server and not usually accessible through a file system or directory structure. The main options for hosting a databases is: on a single server, in a database cluster, or as a cloud service. All of these systems are designed to abstract the implementation details. A client has software that allows it to connect and communicate with the database server using a standardized protocol. There are many different user interfaces for many databases. Databases can be accessed remotely or on the Internet.

Learning Objectives

- Identify and understand database normalization.
- Review why we normalize our data models.
- Understand the step-by-step process of data normalization and forms.
- Learn and apply how we normalize a relational data model.
- Recognize the drawbacks of normalization.

This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.