Chapter 1 - Introduction to Database - Part 2

1 Introduction to Database - Part 2

1.3 Data Storage Format

There are several ways to store large amounts of data, the 4 common ways are

- **Comma Separated Values (CSV)** - data stored in CSV files are the simplest and most popular format used for exchanging data. Each line stores a record in text format and fields are typically separated with a comma (`,`). The format of the data is generally tabular. However, the separator used by CSV does not always have to be commas because if the data contains commas (like `Michael Connelly, Sr`), we would have to use other characters as separators such as asterisks, tabs, etc. Do note that each record in a CSV must have the same number of fields. A good way to test if your CSV file has been properly formatted, use a spreadsheet program such as MS Excel or LibreOffice Cal to open it. It should not have any issues if the CSV file has been formatted properly. An example of a CSV file opened with Notepad can be seen from figure 9 below.

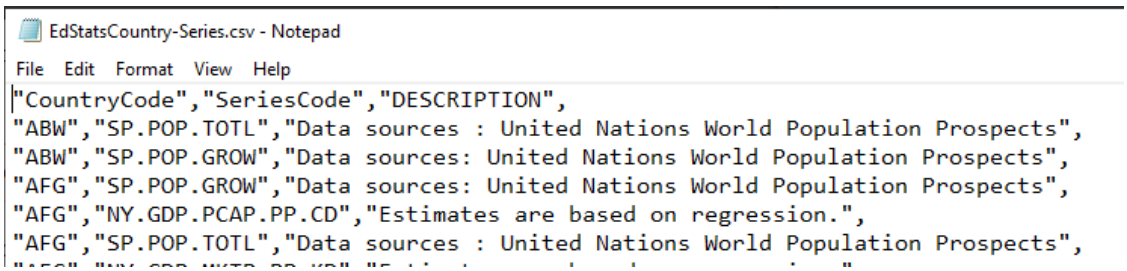

Figure 9: CSV file opened in Notepad in Windows.

- **Extensible Markup Language (XML)** - is a markup language that stores data in a structured non-tabular format. Data is wrapped with custom tags which has some similarities to HTML. However, the tags for XML generally have descriptive tag names. The use of an open-source XML parser is required to read XML files. An example of XML is seen below

```
<?xml version="1.0"?>
<contactinfo>
    <address category="office">
        <name>Olympus Inc.</name>
        <location>587 Drive, Mount Olympus, Greece</location>
        <contact>+30 281 8154 2445</contact>
    </address>
</contactinfo>
```

- **JavaScript Object Notation (JSON)** - is an open standard file and data interchange format that is used to transfer data between programs. It uses attribute-value pairs to store and transmit the text as data objects. As JSON allows storage of array data types which are a collection of elements and other serializable objects it is generally left for more complex structured objects that do not fit in tabular formats. An example of JSON is shown below

```
{
    "menu":{
        "id": "file",
        "popup": {
            "menuitem": [
                {"value": "New", "onclick": "CreateNewDoc()"},
                {"value": "Open", "onclick": "OpenDoc()"},
                {"value": "Close", "onclick": "CloseDoc()"}
            ]
        }
    }
}
```

- **Databases** - are a collection of data organized in some format. The type of data stored in databases are persistent (ie not temporary). A database management system is a software used to manage the interactions between users and other software applications with the database. the 2 mainstream types of databases are Relational databases that stores data in tabular formats and uses SQL, and NoSQL databases that uses query languages.

1.4 Data Acquisition

Data acquisition is the process of acquiring, filtering and cleaning data obtained from various sources before the data is placed into a data warehouse or some other form of storage medium. The crux of gathering data can be summed up by these questions:

- Where can I get my data from?
- Does the data require authorization to acquire? If yes, do I have the authorization or know avenues who have the authorization to get the data?
- How can I get the data (after the first 2 questions have been answered)?

Sources of data can be varied ranging from proprietary data from companies, academic data like experiments carried out for medical research, data scrapped from websites or even logging data like the browsing habits of customers from an online shop. Data gathered during this processed is generally termed "messy" data as the data could have errors and/or missing data. However, the goal of data acquisition is to obtain raw data for processing into reports that you or organizations can use to base decisions upon.

1.5 Data Preprocessing

After gathering the data, we need to clean the data before the data can be processed. This step is required to ensure that the data used for processing (in a later step) is free from errors and is valid for the processing tasks to be carried out on it. There are several things to look out for when cleaning raw data. In this section we will look at some of the ways to identify then clean the data.

Generally, the first few things that we would check for after we receive a dataset that was gathered either by ourselves or externally, would be the data's validity and integrity. The terms *Data validity* and *Data Integrity* are very broad and highly dependent on context and it can also be said that data validity is a prerequisite for data integrity.

With respect to databases, data integrity means to have an overall completeness, accuracy and consistency of data and data validity means that the data has undergone a strict set of rules to ensure that it is correct and useful. The rules that govern data integrity and validity share some overlaps as both are needed to ensure quality data. An example of some rules for checking the validity of data would be the use of *Regular Expressions* to check data fields such as phone number and email address. Once the data is deem valid, its integrity can be checked using various business rules.

Let's take a look at how we can identify some of the things that can go wrong during and after the data has been gathered.

- **Missing Data** - Within a database, this is normally denoted by the keyword `NULL` in the field/s of a record. Missing data generally happens during the data gathering phase when there is no data for certain fields during certain situations like when information is deliberately left out during a survey or the data was not originally available during gathering.
- **Duplicated Data** - happens when there are records that inadvertently share data with another record in the same database or another database. This can happen during any type of data movement between systems (eg: data migration). The easiest type of duplication is exact carbon copies of entire records and the most harmful and common type is the **partial** duplication where a record could be missing data from some fields. Such errors are most often caused by human error, especially when data are input by hand.
- **Errors Correction** - errors can happen even when is an absent of missing or duplicated data. For instance, names of people have many forms, the west has names comprising of first, middle and last names but Asians names only have first and last (family) names. An error would occur if a table in the database did not account for this abnormality. Correction for such errors are generally done by through a manual process of identifying before developing scripts to rectify the errors.

Once all the errors, missing or duplicated data within the data has been identified, data cleansing can begin. Data cleansing is a process whereby the detection and correcting of corrupted or inaccurate records from a record, table or database by means of replacement, modification or deleting the course or dirty data. This process is can be done manually through interactive data wrangling tools or through batch processing using scripts. An example of correct but inconsistent data can be seen in figure 10 below. A table can have a column that stores *Gender* information but the values are not consistent. A script would be developed to replace the inconsistent data based on some rules to ensure that all the values in the column are changed to reflect a more consistent set of data defined for the database.

Figure 10: Column *Gender* before and after data cleansing.

Data cleansing differs from data validation in that validation is done before data is entered to the database and data cleansing is done on data already in a database.

To ensure that the data quality is high, it needs to have certain qualities:

- **Validity** - as described earlier, data validity is property where the data has to conform to the defined business rules or constraints during the data gathering process.
- **Accurate** - accuracy of data is generally hard to achieve as it involves the comparing the data that you have with an external source that contains "True values". Example would be a table of customer's address with postal codes where you may need to compare it with an external source to make sure that the postal codes are correct.
- **Complete** - refers to how much the data meets the expectations for the task. Data can still contain missing values so long as those fields are optional. However, if the data is incomplete, it is almost impossible to fix as we will need to go back to the source to get it
- **Consistent** - refers to how consistent data are across a system or database. As mentioned earlier, data can be inconsistent like in the *Gender* column of the above example. Having 2 records of the same customer data but with different addresses on 2 different systems is also deem as inconsistent as it may contradicts each other if both are current addresses therefore we have to use various strategies to decide which is the most updated record.
- **Uniform** - refers to how the the data measures up to the defined units of measurement such as a particular currency for money data.

References

1. Comma-separated values, https://en.wikipedia.org/wiki/Comma-separated_values
2. XML, https://en.wikipedia.org/wiki/XML
3. Introducing JSON, https://www.json.org/json-en.html
4. Devins, Felin, Kauffman, Koppl, 2017, The Law and Big Data, Vol 27, Cornell Journal of Law and Public Policy, Cornell University, https://www.lawschool.cornell.edu/research/JLPP/upload/Devins-et-al-final.pdf
5. Lyko K., Nitzschke M., Ngonga Ngomo AC. (2016) Big Data Acquisition. In: Cavanillas J., Curry E., Wahlster W. (eds) New Horizons for a Data-Driven Economy. Springer, Cham
6. Skiena S.S. (2017) Data Munging. In: The Data Science Design Manual. Texts in Computer Science. Springer, Cham