1. **Introduction （400-500）**

Affected by the covid-19 , the public's demand for home entertainment has increased significantly. Playing games is also becoming indispensable for most young people under such social pressure. The happiest time for most modern young people is undoubtedly playing switch games on the sofa after work or study. This social situation has boosted Nintendo's game consoles and game software sales.

Nintendo is a Japanese company that mainly focuses on developing video game software and hardware. This company is one of the three majorities of the video game industry, and currently, it is the modern video game industry pioneer. Their most popular product is Switch, a console released in March 2017, which is a revolutionary console.

Switch's features are its design, which adopts the integrated design of the home console and handheld console. This design arguably redefines gaming. Players can play Switch games anywhere, such as at bus stops, planes, trains, or even in a doctor's waiting room due to its portability.

Five years past from 2017, the Switch has sold about 107.2 billion units in its lifetime. This impressive selling result made Nintendo Switch turn to be the seventh video game platform to sell more than 100 million units. Other platforms such as PS2, DS, Game Boy, PS4, PS1, and Wii are well-known. This indicates that Nintendo Switch is promoted to be one of the most popular products in the gaming world. A lot has changed for Nintendo since the Switch arrived in 2017; Nintendo Switch's derivatives have also kept up with trends and come into the public eye. One of the most important is Switch Games. Players can experience a variety of games with Switch, top switch games are simply stunning and profitable.

Metacritic is a website that collects reviews on games, which will integrate the scores of each reviews and make a final mark on the games. The final score is worthy as a reference by many players.

In this project, we will scrape data from Metacritic about the top hundred switch games and data from youtube. We aim to use those data to create a database, which will be convenient for further data pipeline study. >>>>我们要做什么 步骤 最后添加

Graph : Work flow create ([https://lucid.app/lucidchart/9b96a3e0-bd57-4acd-92b4-6d1e67ed96b7/edit?beaconFlowId=596B37B87BD2A753&page=0\_0&invitationId=inv\_ae3ec4e5-f8ca-41db-b267-12acbb4e7915#](https://lucid.app/lucidchart/9b96a3e0-bd57-4acd-92b4-6d1e67ed96b7/edit?beaconFlowId=596B37B87BD2A753&page=0_0&invitationId=inv_ae3ec4e5-f8ca-41db-b267-12acbb4e7915))

1. **Data Extraction**

In this project, we used two methods to grab data to ensure the information is realistic and up to date in the database. The first method is called web scraping, and we used the first web crawler and second web crawler in this project. The web crawler is a simulation way as a browser to open a webpage to extract the data we want from that page. As long as the data can be accessed through a browser, we can acquire it through the web crawler. The first web we call here stands for the first web page we reach. The second web stands for the web page we open after clicking through the link from the first page. API is the abbreviation of the Application Programming Interface as the second method we have used during this project to help in data extraction. An API is simply a programming code that will allow data transfer between software. API working processing is as shown in the following flow graph. This method is formal and under license, as most platforms provide API to the public. Both ways extract data with their unique benefits. We will show how we played with these two methods to catch data in more detail.

图形用户界面, 图示

描述已自动生成

Figure.2 API Working Flow

* 1. ***Metacritic***

As mentioned in the introduction, Metacritic's official website will be the primary data source in our project as they can provide millions of data under different filters. Our goal is to build a Switch games database, we would like to catch the data about games that can play on Switch, and we will not set a time section in this project. These selections will bring us to a web page that lists all Switch games under user scores from top to bottom. This web page will be the first web page we want to focus on under the first web crawler. We aim to collect the top 200 game names that stand as the primary key for the game table, which we will introduce in the Relational Database section. To prepare the work, we need to understand the URL of the web page. Since we want to collect the first 200 games and the web page only contains 100 games per page. Therefore we need a total of 2 web pages during the first web crawler. In this project, we found that the URL difference between the first and second of the first web page is the page number printed in the URL. We used a loop during the programming code to achieve two web pages. As we have analyzed the web page, we are ready to work.

The first stage is to retrieve data. To archive this aim, we need to send a request to the target web page through an HTTP. We used a package called *requests* from the python library and included headers in the request. After completing this step, we will get a response; The web page's content has been obtained. The second stage aims to parse the content we just archived, which could be done by using another package called *beautifulSoup*. The content we get is in HTML format; we parse with it and extract critical data such as game name, game ranking, release date, and the second weblink through *regular expressions*. The final step is to save the data. We hold the data in CSV format locally.

So far to this step, we have done the data collection for the first web page. As we have the link to the second web page for each game, we are ready to work on the second web crawler. Similar to working as the first web crawler. Firstly, we have to understand the structure of the HTML page. The data such as user score and the number of players are the main points we want to focus on. We used *Xpath* instead of *regular expressions* as we wanted to be more specific and accurate to the resulting data. By the end, we collected data over the game name, game ranking, release date, and scores of 14 columns of relational data of the top 200 Switch games listed by the user score from Metacritic.

文本

低可信度描述已自动生成

Figure.3 columns from web scraping

* 1. ***YouTube***

YouTube is the second primary data source in our project as it is the most popular and largest video sharing platform in the world. Nintendo Switch has its official account on YouTube, where people can find the newest information on YouTube, such as the new game released and game updates. They can comment, share and favorite the videos instead of just watching. They could even upload their own clipping videos to comment, introduce, reaction to the relational games. Therefore, we could find millions of relational videos by keyword searching. YouTube is also providing the API to the public. The data from YouTube could be one way to reflect the popularity of the Switch games as it gives data such as the number of views, likes, comments and favorites.

Therefore, we will use YouTube V3 API through the second method. In this part, the major function to use is called the V3 Video function, which returns us the relational video information. However, it requires video IDs for each relational video. To obtain video IDs, we could use the function called Search. The name of each game will be the keywords as we suppose most game relational videos must contain the game name in the title to relate and increase exposure opportunities on YouTube. We have to clean up the data we scraped by web scraping before the API work starts. Such as dropping all the nulls and merging the two datasets by game name as we need to create the top 100 Switch games list from the data we got from Metacritic. We used a loop to achieve the video IDs of the top 50 relational videos from the top 100 games during the programming code. After we got the video IDs, we could use the V3 Video function to generate the relational video data, including view count, like count, favorite count and comment count of each game.

By the end of this Data Extraction stage, we gathered 3 CSV datasets related to the top 100 Switch games ranked by Metacritic. The next stage is the process of Data Storage which AWS S3 does.

1. **Data Storage**

图片包含 图形用户界面

描述已自动生成

Figure.4 Amazon S3 working flow

Amazon Web Services (AWS) is the most comprehensive and broadly adopted cloud platform globally. They owned millions of customers, helping them in their infrastructure, data storage etc from 2006 till now. Simple Storage Service (S3) is the first server provided by AWS started. S3 service can be thought of as a global storage area network (SAN) that behaves as a big size hard drive by theoretical thinking. We can store and retrieve data from S3 storage. The object is defined as the result of storing and retrieving through S3. These objects are stored in buckets. Again, we can think about the hard drive; objects are like files, and buckets are like folders or directories. We could put data such as videos and pictures, analytics data, and log files into Amazon S3 for the later advanced analytics or machine learning.

This project will focus on using AWS; the S3 service will be used as the primary data storage. We will put the 3 CSV files into S3 storage in this project. To access AWS, we first need an account created by the location EU-west-2 (London); this will also be the region name used later in the programming code. After a few simple steps to create the AWS account, we will be able to view the AWS services with hundreds of them around several different science regions. At this stage, we need the service called S3, as already mentioned above. Creating a bucket will be the first step under this service, which just sets a name, and the rest are set by default and recommended. The Aws region will follow the selection areas by the default; we just need to make sure it belongs to EU(London) EU-west-2. After creating the bucket, we will be ready to go through to programming code to upload our local CSV to AWS S3 storage. Several key points are needed in the code, such as username, access key id and secret access key, bucket name, region, etc. These are available to check from AWS, such as the access key id and secret access key by clicking the Security credentials.

**图形用户界面, 文本, 应用程序, 电子邮件

描述已自动生成**

Figure.5 screenshot of Amazon S3 under bucket de-individual-miu

As the main points and the programming codes are set correctly, we have uploaded the CSV files to the AWS bucket under the S3 storge called de-individual-miu. The following figure is a screenshot showing the Amazon S3 bucket:de-individual-miu page. Up to this stage, we are ready to connect our data from S3 to a database such as PostgreSQL. We will claim this in the next section.

1. **Database**

In this section, we will focus on dealing with the database PostgreSQL. We will need to create schemes for our data after reading data from S3 Storage by Python programming code and cleaning the data. These works help in importing data to the database in the further steps. By the end of this section, we will move to Apache Spark.

* 1. ***Connect to S3 storage***

The first step in this stage is to read the data from S3 storage. This could be done by using a defined function. The main points in the programming code, such as the bucket name, access key id and the secret access key are the same as when we uploaded the CSV file to S3. Since the defined function and the main points are correct, we read CSV as data frames successfully in Python. Now we are ready to process to the next step.

* 1. ***Clean & prepare data***

The first CSV we have read contains 200 entries and 5 columns which is the data we gained from the Metacritic first web. The game\_name, ranking, and release\_date will be the columns we are interested in. The second CSV is the data we obtained from the second web from Metacritic. It contains 200 entries as well, but ten columns in total. Taking a general look at these data, we found some columns are pending as the same results and some show unclear results. We will drop these columns after we merge CSV1 and CSV2. To be noticed, The column called game\_genre is a column that shows multiple kinds of genres, which separates by commas. We used the pandas in Python named *split* to separate them and take the first and second into account of the whole data named *game\_genre1* and *game\_genre2*. Another column, *release\_date*, is a time string which contains the *date*, *month* and *year* in one column. We used the same idea to separate them. After dealing with these issues, we need to generate three Data Frames based on the current data, corresponding to the three tables in the Schema we designed in section4.3. These three Data Frames have named a game, developer, uer\_review\_web.

**手机屏幕截图

描述已自动生成**

Figure.6 Information on CSV1

**手机屏幕截图

描述已自动生成**

Figure.7 Information on CSV2

The last CSV is the YouTube data which contains 100 entries with six columns. It is the most straightforward data to deal with; it directly corresponds to the fourth table in the Schema. All we need to do on these data is append a new column named YouTube id. The id column is also required in the other three tables, such as *game\_id*, *developer\_id*, and *uer\_review\_web\_id*, which will help us connect tables in further studies, such as SQL queries. By the end of this step, all the data we need has been cleaned. We will introduce the Schema we designed for the next step.

**文本

描述已自动生成**

Figure.7 Information on CSV3

* 1. ***Schema***

Database server contains multiple databases, and these databases could contain serval schemas with various tables. Schemas are like serval of small boxes put into the database if we thought the database be like a vast storehouse.

NOT DONE

* 1. **Import Data to PostgreSQL**

In this project, The database server that we will use is PostgreSQL, a relational database management system based on AWS and managed by Terraform. Conversely, as we want to connect to the database, we need to create a database first. This could be done similar to creating a bucket in S3 storage, which will not be difficult. This time the service we choose from AWS is called Amazon Relational Database Service (RDS). According to the official description from Amazon, RDS is a service built on a relational database, which allows users to manage relationships quickly and conveniently. Users could spend more time developing other stuff. From my understanding, RDS is a relational database with multiple functions. Using this database saves us much time; it could help us with database installation of backup/restore etc. RDS supports MySQL, Amazon Aurora, PostgreSQL, Oracle, Microsoft SQL Server and MariaDB engines. Obviously, PostgreSQL will be our target. For the database creation, we can follow the way that Week 9 lecture slides are straightforward. We used the programming code to connect to the database. The keys to joining the correct database are database name, user name, password, host, and port. These keys could be found under the database information. Before inserting the prepared data into the database, we must convert the four tables ( *game, developer, uer\_review\_web* and *youtube\_table*) from Data Frames into tuples. We also need to generate three connect tables which are *game\_developer,game\_user\_review\_web* and *game\_youtube*. These tables include two columns, game\_id with *developer\_id,user\_review\_web\_id* and *youtube\_id*, from 0 to 100, which correspond to the three connect tables described in the Schema. We also need to transform these three Data Frames into tuples before inserting them into the database. As everything was ready, we successfully used *excute* function in Python to insert the actual data into the specific Schema in our database. Now we are ready to make the query on this database at any time.

* 1. **Connect to Spark**

1. **Data processing and Cleaning**

**-** FIRST WEB

* SECOND WEB
* YOUTUBE

1. **Exploratory Data Analysis (SQL+ DV+ model)**
2. **Database Value / pipline**
3. **Conclusion**
4. **Appendix**
5. **References**

<https://www.pocket-lint.com/games/reviews/nintendo/140007-nintendo-switch-review-console-specs-price-and-features>

<https://www.altexsoft.com/blog/engineering/what-is-api-definition-types-specifications-documentation/>