**INFO 6210**

**34833**

**Assignment 2**

**Report**

**Group Members**

**HAN WU  
JIANYU HU**

***- Catalog***

***-Abstract***

***-Introduction***

*Social Media, particular domain selected and database design planning*

***-Conceptual Model***

*Domain*

*Entities*

*Relationships*

*Attributes and keys*

*Diagrams*

*7 questions*

***-Physical Model***

*Updated ER Diagram*

*SQL and Diagrams*

*Use cases and SQL Expressions*

***-Code and Data Sample***

***-Contribution Statement***

***-Conclusion***

***-Reference***

***Abstract***

***In this assignment, we are assumed to be working for a company called Nerd Analytics and that we are completely in charge of the database. Another group of statisticians and machine learning experts will be using the data and database to ask analyze Social Media for a particular domain. Each domain must have entities that represent consumers, producers and companies. In this assignment, we have selected gaming as our domain and scrap posted data from Twitter and YouTube. We will use MySQL to construct our physical database and write queries for specific use cases.***

1. **Introduction**

Nowadays, Gaming is one of the most popular topics which people often talk about and would like to post on their social media. It is important for consumers, producers and companies to know about games they are interested in through gaming topics in social media. In this case, we used Python to get these post information through social media’s API, then from these data set, we analyze the relationships between entities and attributes and construct conceptual database model. After that we updated our conceptual model to fit the physical database and illustrate the entire conceptual model in a form that directly maps to SQL. Moreover, we have list several use cases which are particular to our domain and we also gave our SQL queries for solution.

1. **Conceptual Model**

* **Domain**

The domain we selected is gaming, which category ID is 20 in YouTube. The posts/tweet related to this domain contains tags, tittle, descriptions, publish date, channels, video parameters and URL.

* **Entities**

The plan on tweet entities is separate tweets into three entities regarding to the consumers, producers and companies that will mentioned in the next point.

Each entity will have a same model. Attributes should include import information such as id, tweets, time, likes count, and hashtags etc.

For social media user (which most likely to be a producer such as steam platform), the entities will only contain information like their id, user name and screen name (@id for twitter). This happens due to the request limit on my twitter developer account and it is the limitation of using API.

For tweet, considered as posts, will have attributes such as user name, text about gaming domain, type and publish date in this entity.

Also if posts type is media, this entity is consisting of attributes like URL, display URL. For URL entity, we have attributes like detailed URL, expanded URL, display URL.

For entities in YouTube data set, attributes should include import information such as id, tags, publish time, and channel ID etc.

For items, considered as posts, will have attributes such as id, tags, descriptions, tittle, default language, kind, category ID and publish date in this entity.

Each item belongs to a channel which contains channel ID and channel tittle.

Also, items have thumbnails which conclude an entity named URL, in this entity, the parameter of thumbnails such as height and width is determined and it also contains the URL attribute.

Moreover, for entity item, it has localized tittle and description. As a result, we can link them with foreign keys.

* **Relationships**

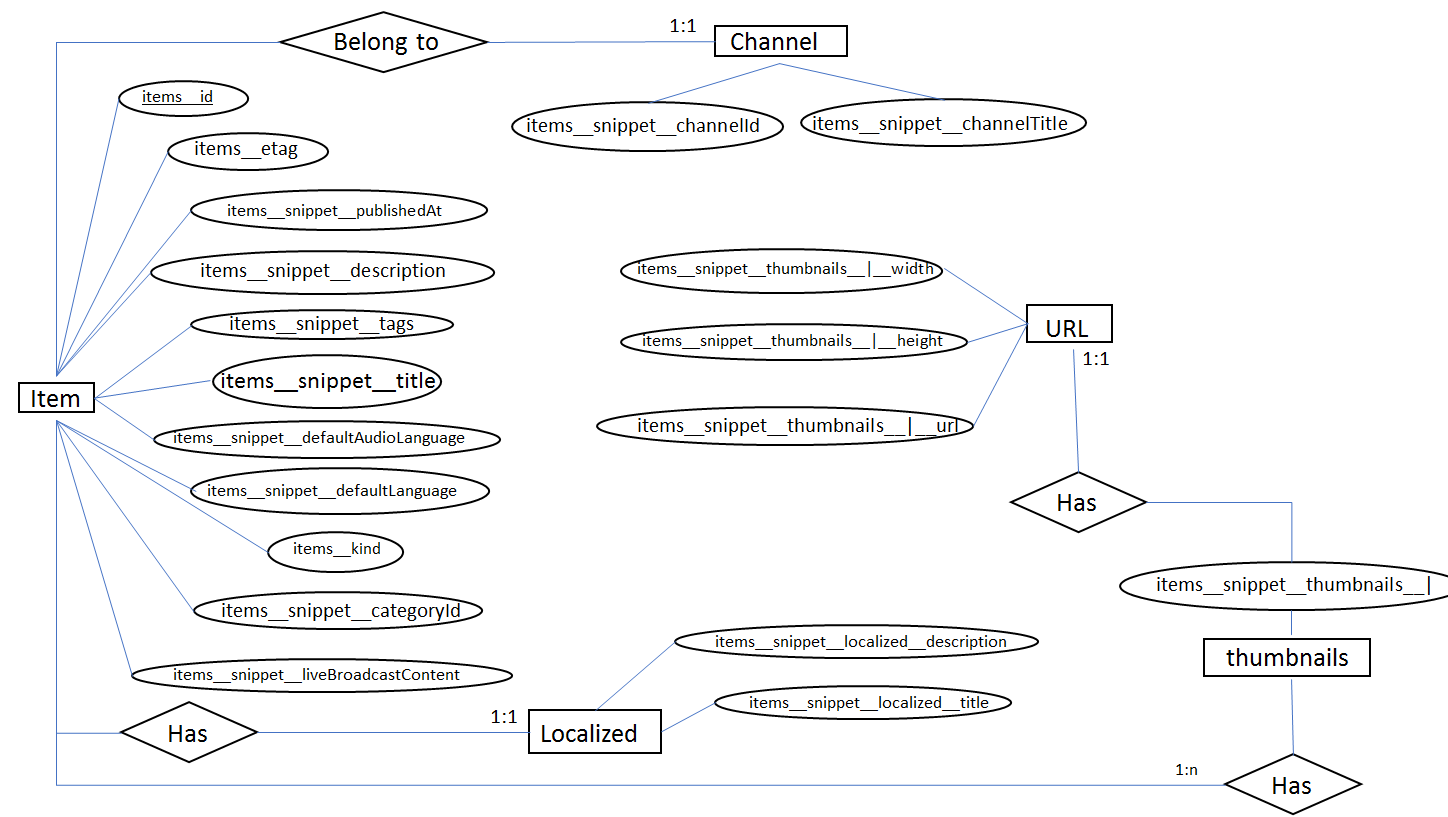
Shown as the figure.

* **Attributes and keys**

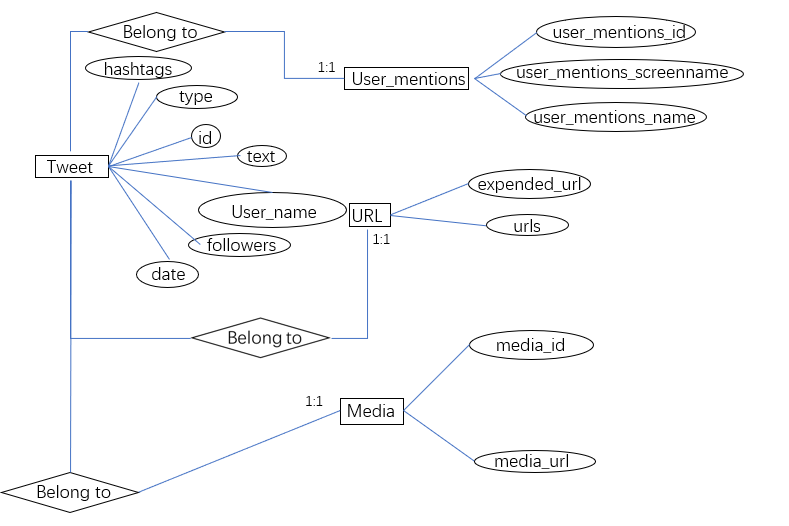
Shown as the figure.

* **Diagrams**

Shown as the figure.



*Figure-1 YouTube*



*Figure-2 Twitter*

* **7 questions**

1. What are the ranges, data types and format of all of the attributes in your entities?

The details of this question can be found in the ER diagram. All the attributes that contain string are designed as text such as screenname, tweet and description. The ranges can vary. Details of changes will be presented in the ER diagram on physical database model.

Numerical values are mostly integer such as ID. The range for user ID is mostly non-negative value and may not have a rule but it depends on the register record or date. One exception is for twitter account id, some of the number is so large that double is needed.

Date and time is used when there is date involved such as publishing date or posting date.

1. When should you use an entity versus attribute? (Example: address of a person could be modeled as either)

Entities do have a key attribute are considered as weak entities. That's when it should be split into attributes. For my case tweet accounts for each game company do not have a strong relationship since the game entity is trying to have relation with games and consumers. I decide the put them into a separate entity where twitter account information for all parties are stored.

1. When should you use an entity or relationship, and placement of attributes? (Example: a manager could be modeled as either)

When the attributes can have relation with in them where a relational entities can explain it easier. Another case is when multiple information is stored in one cell. For example, hashtags for tweets, it is very difficult to deal with. This is why I choose to build an entity for hashtags which have a one to many relationships with the entity for tweets.

1. How did you choose your keys? Which are unique?

Foreign keys for most entities are a unique ID that is pre generated when acquiring the data. These IDs are verified to be unique during the data auditing process. Moreover, tables have connections with each other by using these IDs as FKs. When it comes to primary keys, each entity has its own unique attributes. For instance, a tweet has a unique id as its own record. And for URL entity it has a unique URL which is character value type.

1. Did you model hierarchies using the “ISA” design element? Why or why not?

No. Because in our database design, it doesn’t exist any inheritance relationships for using entities.

1. Were there design alternatives? What are their trade-offs: entity vs. attribute, entity vs. relationship, binary vs. ternary relationships?

The original design is to get as much as possible twitter accounts in gaming domain, however, there exists limit by using twitter developer API, so we choose several accounts and focus on their tweets context.

Twitter account and tweets for game companies, players and platforms which considered as producer were all originally designed to be stored separately due to the structure of my raw data. However, they were combined into two entities where one stored all the twitter account and the other one stored tweets.

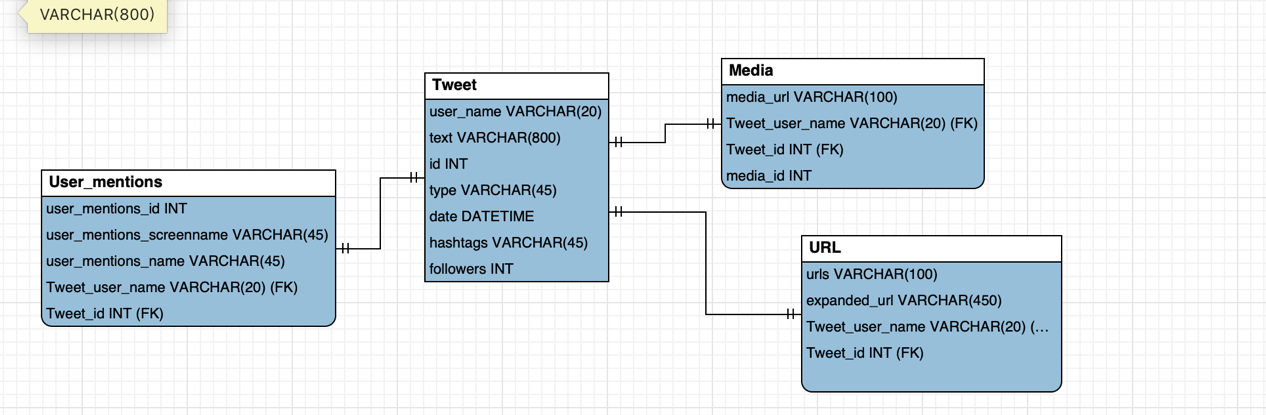
1. Where are you going to find real-world data to populate your model?

The data set of YouTube was get by using google YouTube API V3, we ran the Python code to get json file then convert it into csv files. For this data, we can get useful text information in video descriptions or tags and we can also follow up these information using URLs which contains.

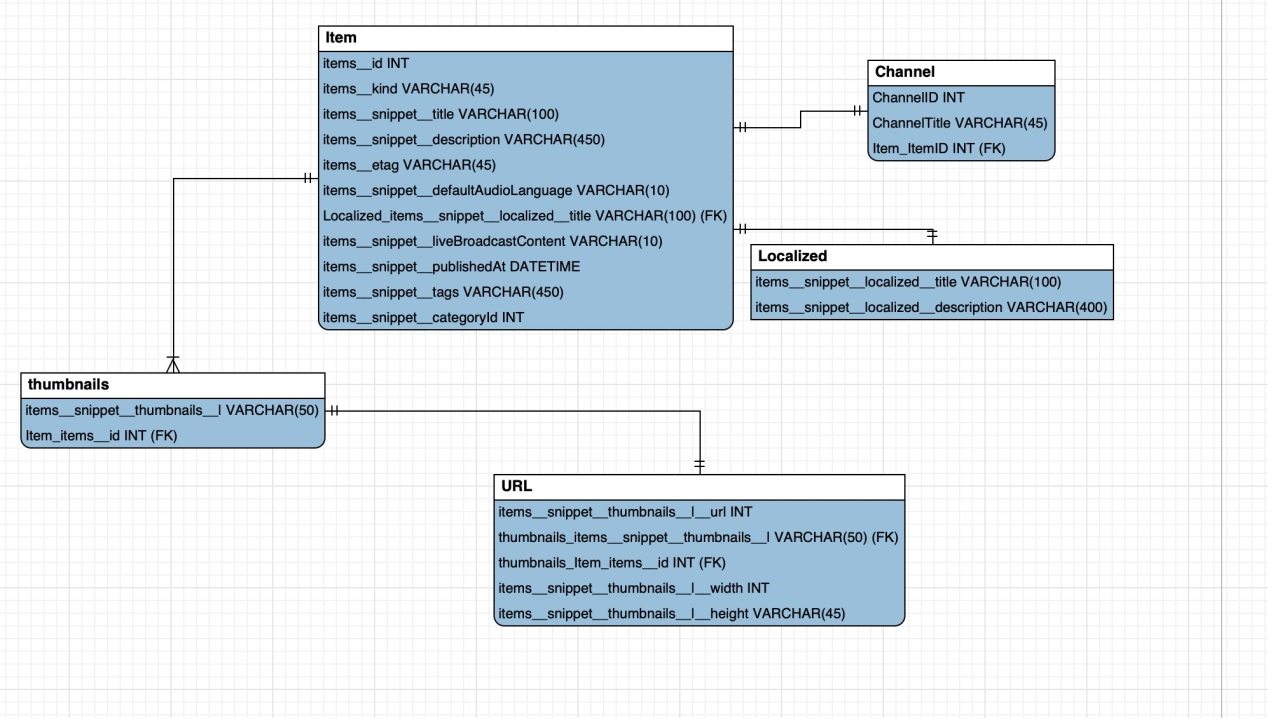
Other data is Tweets data which acquired with twitter developer API, we selected six popular game company and got their tweets information to construct our database. We can get useful information from these posts and tags too.

1. **Physical Model**

* **Updated ER Diagram:** *Shown as the figure*

****

*Firgure-1 Twitter*

****

*Firgure-2 YouTube*

* **SQL and Diagrams**

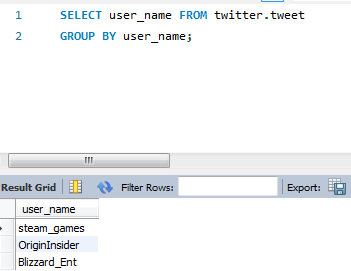
Queries you must answer about your physical model (In SQL):

1. What user posted this (e.g. tweet, facebook post, IG post, etc.)?

*Twitter:*

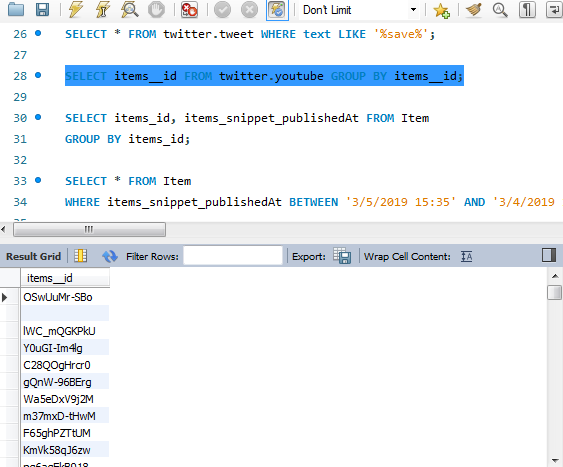
*SELECT user\_name FROM Tweet*

*GROUP BY user\_name;*

**

*YouTube:*

*SELECT items\_id FROM Item GROUP BY items\_id;*

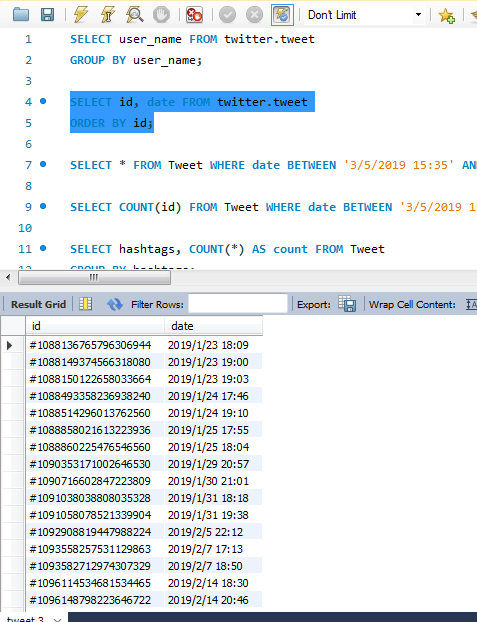
**

1. When did the user post this (e.g. tweet, facebook post, IG post, etc.)??

*Twitter:*

*SELECT id, date FROM Tweet*

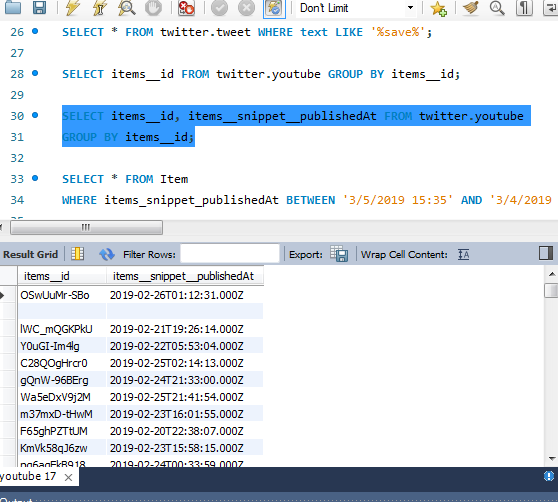
*ORDER BY id;*

**

*YouTube:*

*SELECT items\_id, items\_snippet\_publishedAt FROM Item*

*GROUP BY items\_id;*

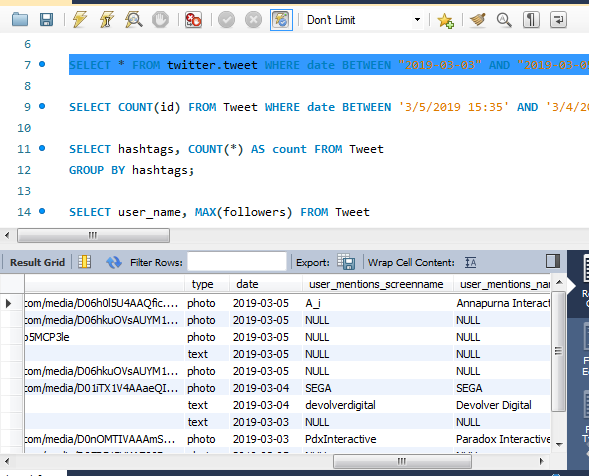
**

1. What posts has this user posted in the past 24 hours?

*Twitter:*

*SELECT \* FROM Tweet*

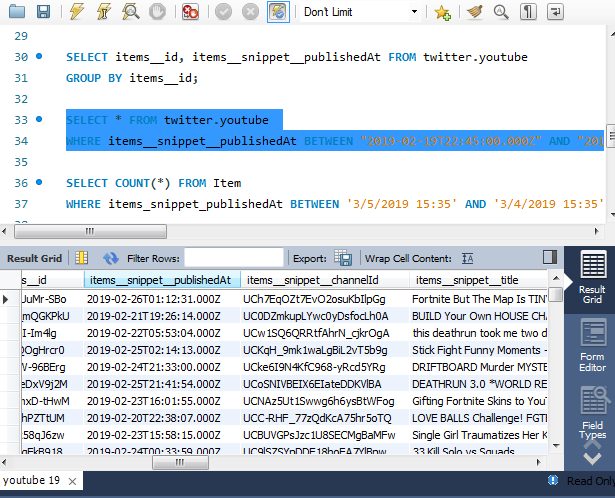
*WHERE date BETWEEN '3/5/2019 15:35' AND '3/4/2019 15:35';*

**

*YouTube:*

*SELECT \* FROM Item*

*WHERE items\_snippet\_publishedAt BETWEEN '3/5/2019 15:35' AND '3/4/2019 15:35';*

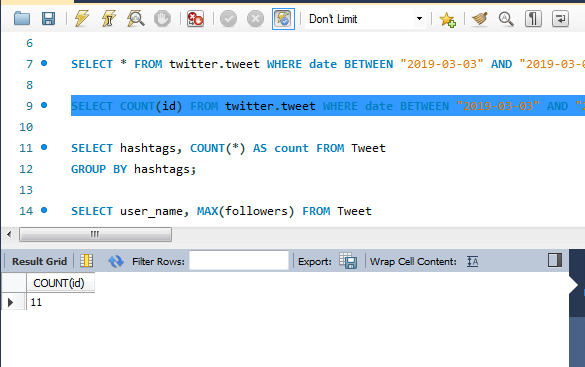
**

1. How many post has this user posted in the past 24 hours?

*Twitter:*

*SELECT COUNT(\*) FROM Tweet*

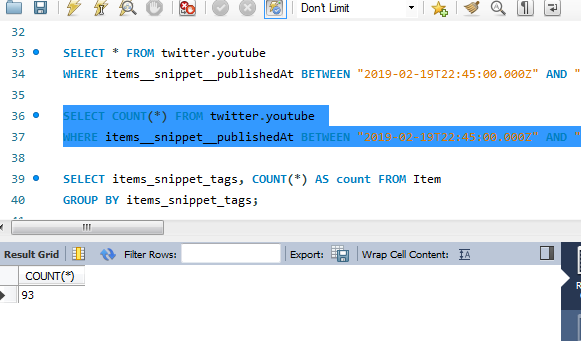
*WHERE date BETWEEN '3/5/2019 15:35' AND '3/4/2019 15:35';*

**

*YouTube:*

*SELECT COUNT(\*) FROM Item*

*WHERE items\_snippet\_publishedAt BETWEEN '3/5/2019 15:35' AND '3/4/2019 15:35';*

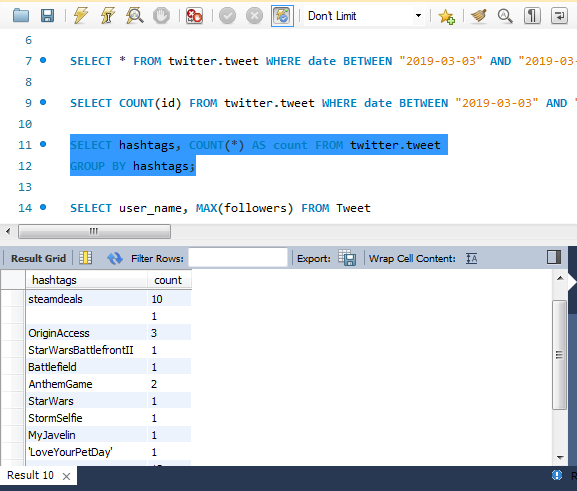
**

1. What keywords/ hashtags are popular?

*Twitter:*

*SELECT hashtags, COUNT (\*) AS count FROM Tweet*

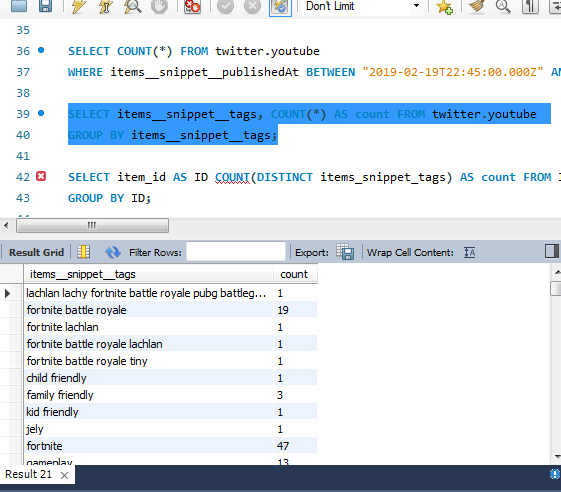
*GROUP BY hashtags;*

**

*YouTube:*

*SELECT items\_snippet\_tags, COUNT(\*) AS count FROM Item*

*GROUP BY items\_snippet\_tags;*

**

1. What posts are popular?

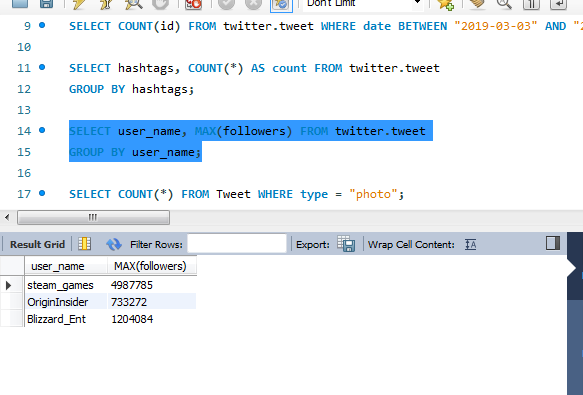
*Twitter:*

*SELECT user\_name, MAX(followers) FROM Tweet*

*GROUP BY user\_name;*

*SELECT \* FROM Tweet*

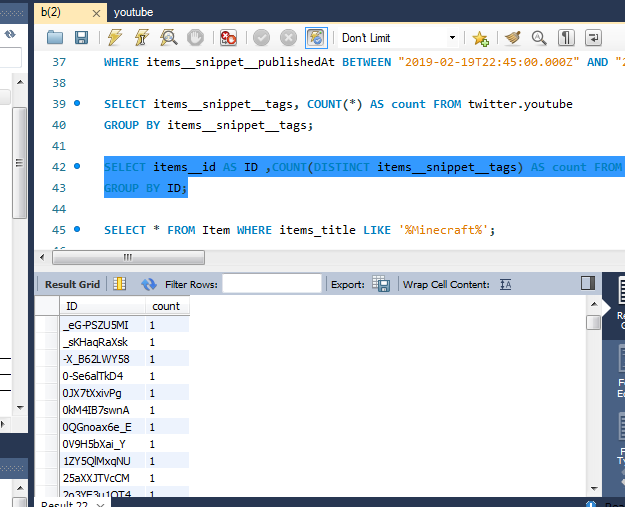
*GROUP BY text;*

**

*YouTube:*

*SELECT item\_id AS ID COUNT (DISTINCT items\_snippet\_tags) AS count FROM Item*

*GROUP BY ID;*

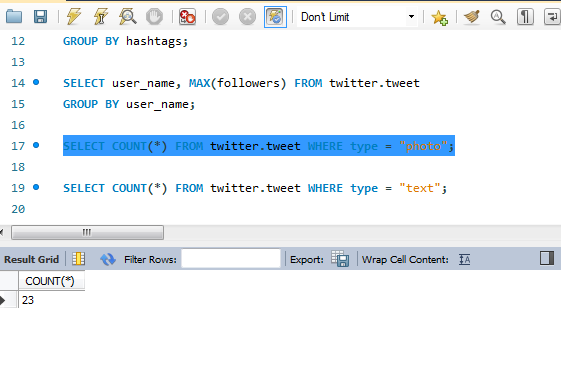


* **Use cases and SQL Expressions**

*Twitter:*

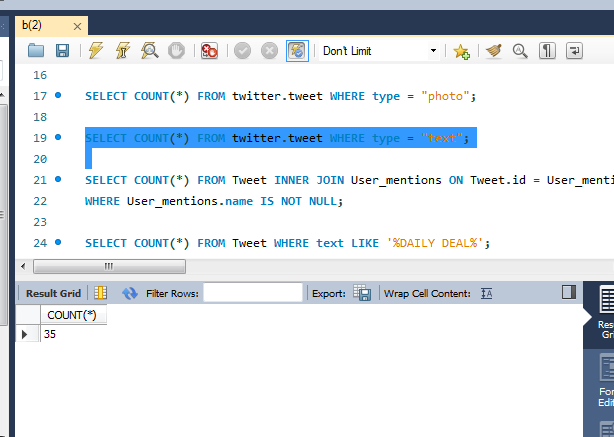
1. How many Tweets has the photo in his posts?

*SELECT COUNT(\*) FROM Tweet WHERE type = "photo";*

**

1. How many Tweets has the photo in his posts?

*SELECT COUNT(\*) FROM Tweet WHERE type = "text";*

**

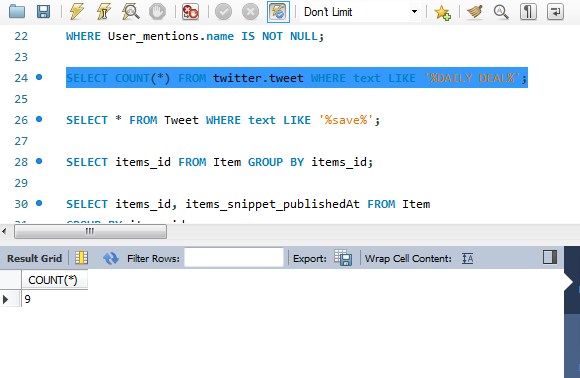
1. How many posts mentions other users?

*SELECT COUNT(\*) FROM Tweet INNER JOIN User\_mentions ON Tweet.id = User\_mentions.id*

*WHERE User\_mentions.name IS NOT NULL;*

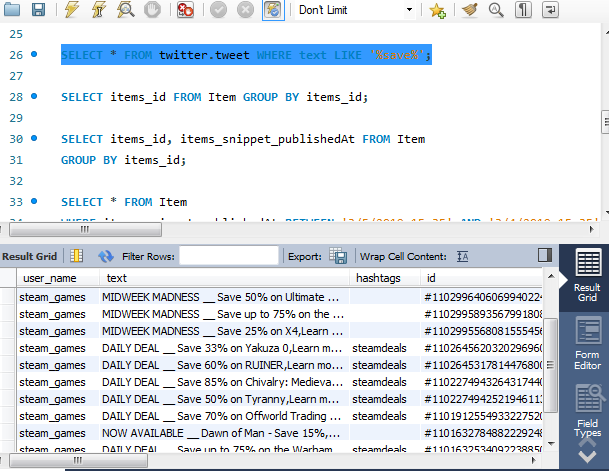
1. How many posts are related to daily deal discount?

*SELECT COUNT(\*) FROM Tweet WHERE text LIKE '%DAILY DEAL%';*

**

1. What posts are related to game discount?

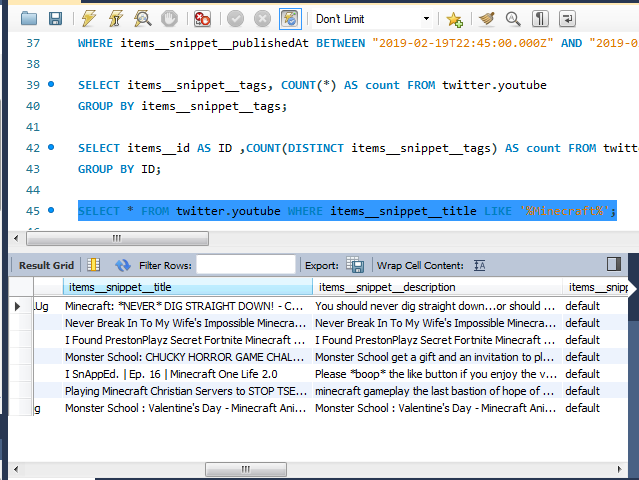
*SELECT \* FROM Tweet WHERE text LIKE '%save%'*



YouTube:

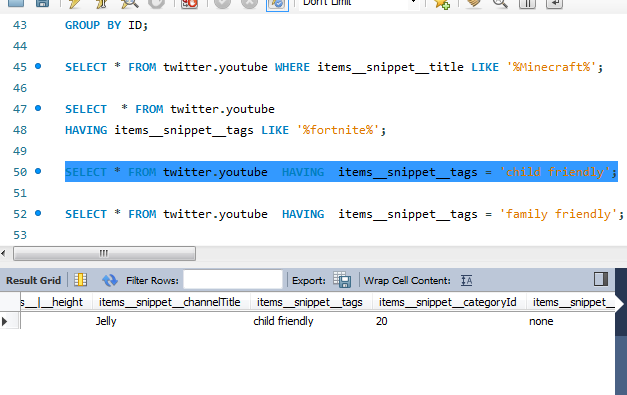
1. What post videos are related to minecraft?

*SELECT \* FROM Item WHERE items\_title LIKE '%Minecraft%';*

**

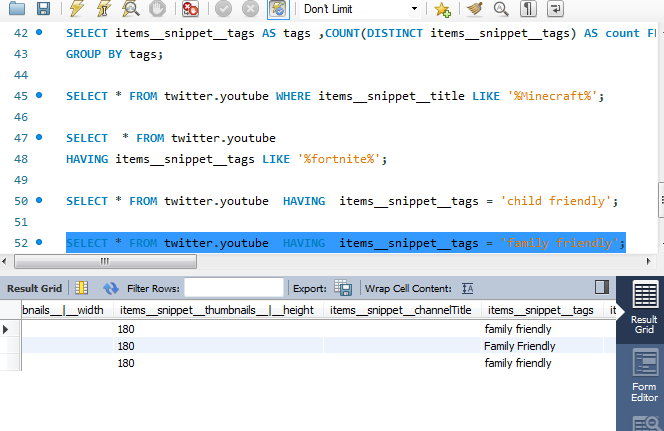
1. What posts are child friendly?

*SELECT \* FROM Item HAVING items\_snippet\_tags = 'child friendly';*

**

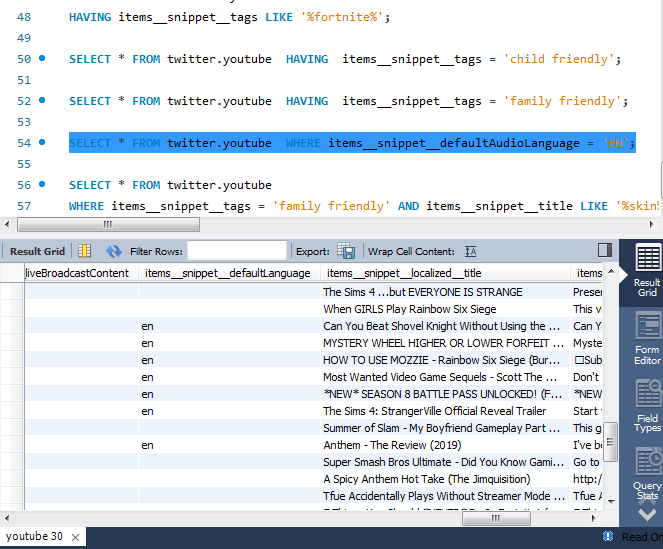
1. What posts are suitable for family?

*SELECT \* FROM Item HAVING items\_snippet\_tags = 'family friendly';*

**

1. What posts are posted through English?

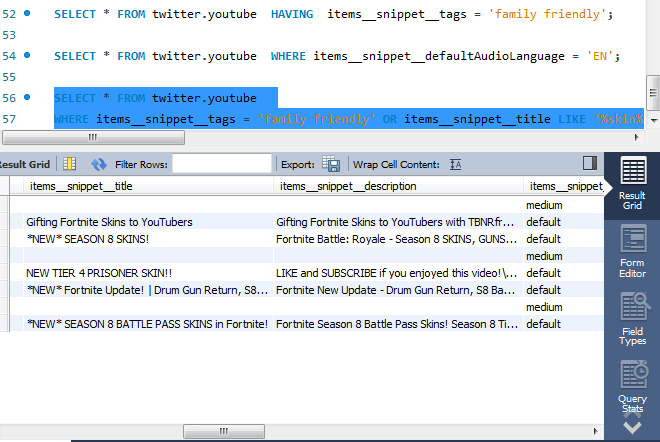
*SELECT \* FROM Item WHERE items\_snippet\_defaultAudioLanguage = 'EN';*

**

1. What posts are related to fotnite skin?

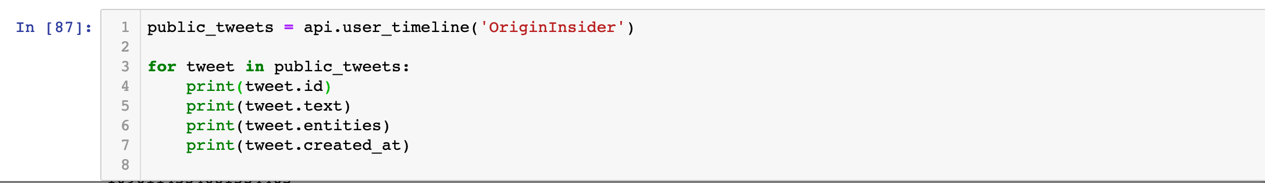
*SELECT \* FROM Item*

*WHERE items\_snippet\_tags = 'family friendly' AND items\_snippet\_title LIKE '%skin%';*

**

1. **Code and Data Sample**

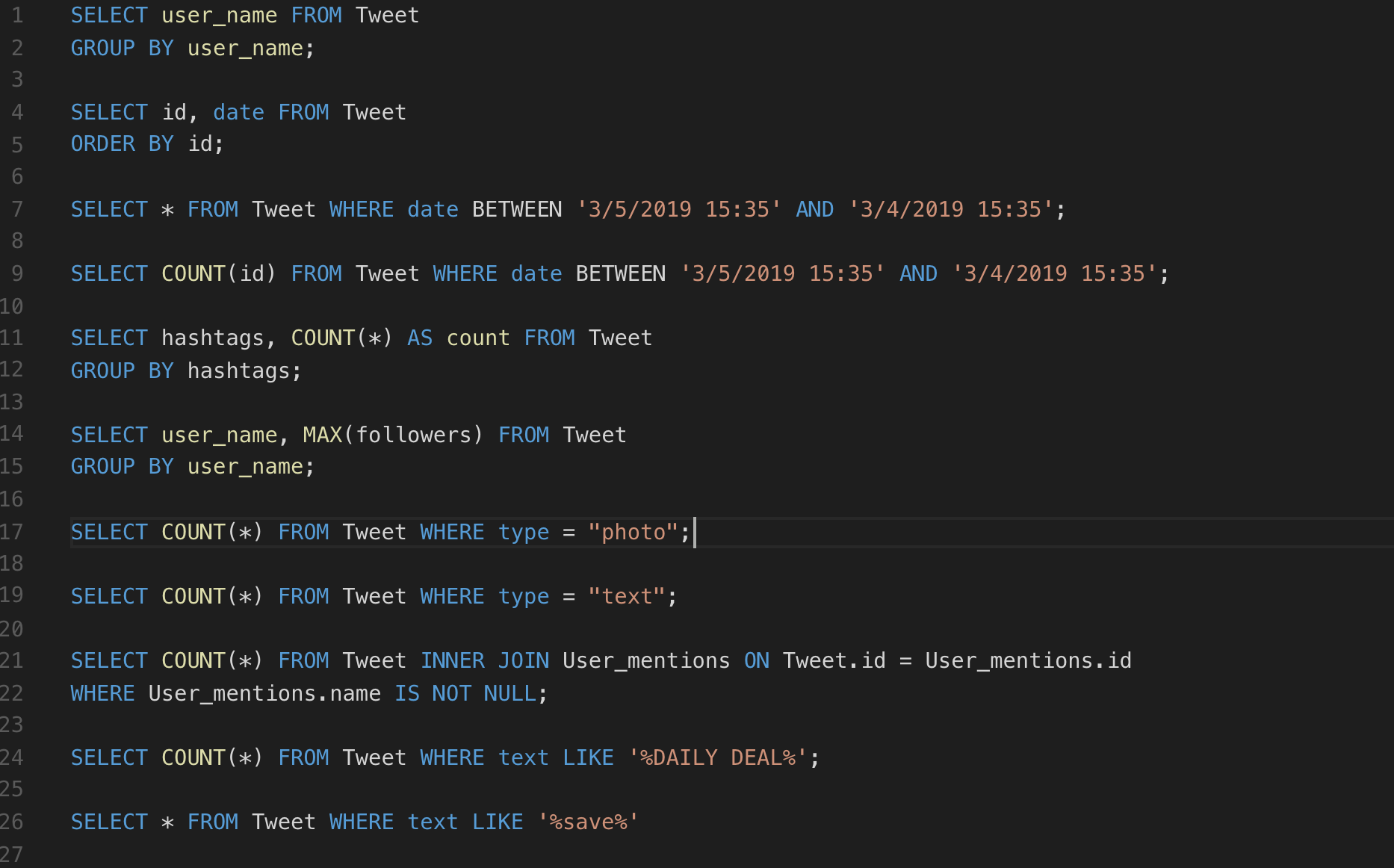
****

****

*Figure 1&2: Python Code Crawling data from Twitter Through API*

**

*Figure 3: Data Sample*

**

*Figure 4: SQL Sample*

1. **Contribution Statement**

HAN WU: Twitter Data Crawling; Data reformat; Queries coding; User cases coding; Database Designing and Building.

JIANYU HU: YouTube Data Crawling; Data reformat; Database building; Writing Report; Database Testing and Running.

1. **Conclusion**

What we did during this assignment is first we used Python to get these post information through social media’s API, then from these data set. The second step is built a conceptual model for a tweet/post, a Social Media user, a person, and a company. Then we analyze the relationships between entities and attributes and complete conceptual database model while appropriating attributes and keys. In addition, we draw an ER diagrams that illustrate the entire conceptual model. After that, we updated our conceptual model to fit the physical database and illustrate the entire conceptual model in a form that directly maps to SQL. By using MySQL workbench, we create tables for each entity, set value types and linked tables with keys. Moreover, we have list several use cases which are particular to our domain and we also gave our SQL queries for solution.

1. **Reference**

*For code:*

1. [*https://github.com/youtube/api-samples/blob/master/python/search.py*](https://github.com/youtube/api-samples/blob/master/python/search.py)
2. [*https://github.com/nikbearbrown/INFO\_6210/tree/master/Week1*](https://github.com/nikbearbrown/INFO_6210/tree/master/Week1)
3. [*https://github.com/tweepy/tweepy/blob/master/tweepy/api.py*](https://github.com/tweepy/tweepy/blob/master/tweepy/api.py)

*For data:*

1. [*https://twitter.com/Blizzard\_Ent*](https://twitter.com/Blizzard_Ent)
2. [*https://twitter.com/OriginInsider*](https://twitter.com/OriginInsider)
3. [*https://twitter.com/steam\_games*](https://twitter.com/steam_games)