This is a PySpark ETL process for Google Chrome History. It begins by extracting data on visited and linked URLs from sqlite3 on local machine, followed by the analysis of keyword searches. Additionally, the process examines successful, unsuccessful downloads, interruptions and reasons for interruptions. Moreover, it assesses the frequency of visits for the visited links during active sessions. To gain a more comprehensive understanding, the analysis extends to determining the hour of the day when users are most active. This multifaceted approach aims to provide insights into user behavior, search patterns, download issues, and overall engagement within the Google Chrome browser.

In [2]:
```
#code to extract your google browser history
#your data being extracted from sqlite3

import sqlite3
import os
import csv
from datetime import datetime, timedelta
# Determine the path to the Chrome history database
# On Windows:
history_db = os.path.join(os.environ['LOCALAPPDATA'], 'Google', 'Chrome', 'User Data', 'Default', 'History')
# On macOS:
# history_db = os.path.expanduser('~/Library/Application Support/Google/Chrome/Default/History')
# On Linux:
# history_db = os.path.expanduser('~/.config/google-chrome/Default/History')

# Connect to the database
print(history_db)
connection = sqlite3.connect(history_db)
cursor = connection.cursor()

#select all the tables in database
cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")
tables=cursor.fetchall()
print(tables)
cursor.execute("PRAGMA table_info(visits)")
columns = cursor.fetchall()
print(columns)


# # Execute a PRAGMA statement to get the column information for the "urls" table
cursor.execute("PRAGMA table_info(urls)")
columns = cursor.fetchall()
#print(columns)
c=[]
# # Print the column information
for column in columns:
    x = column[1].replace("'", "").replace("{", "").replace("}", "")
    c.append(x)


#datetime conversion function
def microseconds_to_datetime(microseconds_str):
    microseconds = int(microseconds_str)
    base_datetime = datetime(1601, 1, 1)
    visit_time = base_datetime + timedelta(microseconds=microseconds)
    return visit_time.strftime('%Y-%m-%d %H:%M:%S')

# Execute a query to select all columns from the "urls" table
cursor.execute("SELECT * FROM urls")
rows = cursor.fetchall()
#print(rows)
rows = [row for row in rows if microseconds_to_datetime(row[5]) is not None]


# Close the database connection
connection.close()


#set output directory and output filename
output_directory = "C:/Documents/jupternotebookprac/history_analysis"
csv_filename = "output.csv"

# Create the directory if it doesn't exist
if not os.path.exists(output_directory):
    os.makedirs(output_directory)

# Write the rows to the CSV file
csv_path = os.path.join(output_directory, csv_filename)
with open(csv_filename, 'w', newline='',encoding='utf-8') as csv_file:
    csv_writer = csv.writer(csv_file)
    csv_writer.writerow(c)
    for row in rows:
        #print(row)
```

```
        timestamp_microseconds = row[5]
        visit_time = microseconds_to_datetime(timestamp_microseconds)
        #print(visit_time)

        row = list(row)
        row[5] = visit_time
        #print(row)

        csv_writer.writerow(row)
print(f"Data has been written to {csv_filename}")
```

```
C:\Users\manju\AppData\Local\Google\Chrome\User Data\Default\History
[('meta',), ('urls',), ('sqlite_sequence',), ('visits',), ('visit_source',), ('keyword_search_terms',), ('downlo
ads',), ('downloads_url_chains',), ('downloads_slices',), ('segments',), ('segment_usage',), ('content_annotatio
ns',), ('context_annotations',), ('clusters',), ('clusters_and_visits',), ('cluster_keywords',), ('cluster_visit
_duplicates',), ('visited_links',), ('history_sync_metadata',)]
[(0, 'id', 'INTEGER', 0, None, 1), (1, 'url', 'INTEGER', 1, None, 0), (2, 'visit_time', 'INTEGER', 1, None, 0),
(3, 'from_visit', 'INTEGER', 0, None, 0), (4, 'external_referrer_url', 'TEXT', 0, None, 0), (5, 'transition', 'I
NTEGER', 1, '0', 0), (6, 'segment_id', 'INTEGER', 0, None, 0), (7, 'visit_duration', 'INTEGER', 1, '0', 0), (8,
'incremented_omnibox_typed_score', 'BOOLEAN', 1, 'FALSE', 0), (9, 'opener_visit', 'INTEGER', 0, None, 0), (10, '
originator_cache_guid', 'TEXT', 0, None, 0), (11, 'originator_visit_id', 'INTEGER', 0, None, 0), (12, 'originato
r_from_visit', 'INTEGER', 0, None, 0), (13, 'originator_opener_visit', 'INTEGER', 0, None, 0), (14, 'is_known_to
_sync', 'BOOLEAN', 1, 'FALSE', 0), (15, 'consider_for_ntp_most_visited', 'BOOLEAN', 1, 'FALSE', 0), (16, 'visite
d_link_id', 'INTEGER', 1, '0', 0)]
Data has been written to output.csv
```

In [3]:

```
#create SparkSession and read data from previous generated output file for chrome data
from pyspark.sql import SparkSession
from pyspark.sql.types import StructType, StructField, StringType, IntegerType, DateType, DoubleType
from pyspark.sql.functions import hour, col

[(0, 'id', 'INTEGER', 0, None, 1), (1, 'url', 'LONGVARCHAR', 0, None, 0), (2, 'title', 'LONGVARCHAR', 0, None, (
 (3, 'visit_count', 'INTEGER', 1, '0', 0),
 (4, 'typed_count', 'INTEGER', 1, '0', 0), (5, 'last_visit_time', 'INTEGER', 1, None, 0), (6, 'hidden', 'INTEGE
customschema = StructType([
    StructField("id", IntegerType(), True),
    StructField("url", StringType(), True),
    StructField("title", StringType(), True),
    StructField("visit_count", IntegerType(), True),
    StructField("typed_count", IntegerType(), True),
    StructField("last_visit_time", StringType(), True),
    StructField("hidden", IntegerType(), True),
])
print(os.getcwd())
dateFormat = "MM/dd/yyyy"
spark= SparkSession.builder.getOrCreate()
print(spark)
urls=spark.read.csv("output.csv",schema=customschema,header=True,dateFormat="MM/dd/yyyy HH:mm:ss")
urls.show()
urls.select(hour(col("last_visit_time")).alias("visit_hour")).show()
urls.count()
```

```
C:\Users\ganes\Documents\Manju\history_analysis
<pyspark.sql.session.SparkSession object at 0x00000245C349F150>
+---+--------------------+--------------------+-----------+-----------+-------------------+------+
| id|                 url|               title|visit_count|typed_count|    last_visit_time|hidden|
+---+--------------------+--------------------+-----------+-----------+-------------------+------+
|  1|https://accounts....|Sign in - Google ...|          1|          0|2023-07-29 01:01:08|     0|
|  2|https://accounts....|Sign in - Google ...|          2|          0|2023-07-29 01:01:09|     0|
|  3|https://www.googl...|gmail - Google Se...|          2|          0|2023-07-29 01:09:23|     0|
|  4|https://mail.goog...|Inbox (1,170) - g...|          2|          0|2023-07-29 03:43:15|     0|
|  5|https://mail.goog...|Inbox (1,170) - g...|          3|          0|2023-07-29 03:43:15|     0|
|  6|https://mail.goog...|Inbox (1,189) - g...|         54|          2|2023-10-17 23:15:56|     0|
|  7|https://accounts....|Gmail: Private an...|          1|          0|2023-07-29 01:09:24|     0|
|  8|https://mail.goog...|Gmail: Private an...|          1|          0|2023-07-29 01:09:24|     0|
|  9|https://www.googl...|Gmail: Private an...|          1|          0|2023-07-29 01:09:24|     0|
| 10|https://www.googl...|Gmail: Private an...|          1|          0|2023-07-29 01:09:24|     0|
| 11|https://accounts....|               Gmail|          1|          0|2023-07-29 01:09:30|     0|
| 12|https://accounts....|               Gmail|          1|          0|2023-07-29 01:09:30|     0|
| 13|https://accounts....|               Gmail|          1|          0|2023-07-29 01:09:30|     0|
| 14|https://accounts....|               Gmail|          1|          0|2023-07-29 01:09:30|     0|
| 15|https://accounts....|               Gmail|          2|          0|2023-07-29 01:09:30|     0|
| 16|https://accounts....|               Gmail|          1|          0|2023-07-29 01:09:39|     0|
| 17|https://accounts....|               Gmail|          1|          0|2023-07-29 01:09:44|     0|
| 18|https://accounts....|Inbox (1,167) - g...|          1|          0|2023-07-29 01:10:02|     0|
| 19|https://mail.goog...|Inbox (1,167) - g...|          1|          0|2023-07-29 01:10:02|     0|
| 20|https://accounts....|Inbox (1,167) - g...|          1|          0|2023-07-29 01:10:02|     0|
+---+--------------------+--------------------+-----------+-----------+-------------------+------+
only showing top 20 rows

+----------+
|visit_hour|
+----------+
|         1|
|         1|
|         1|
|         3|
|         3|
|        23|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
|         1|
+----------+
only showing top 20 rows
```

Out[3]: 10356

In [4]:
```
#handle null values
#drop row if all the columns has null values
urls=urls.dropna(how='all')
urls.count()
```

Out[4]: 10356

In [5]:
```
#handle null values
#fill 0 for numeric null value fields 'visit_count','typed_count','hidden','id'
urls=urls.fillna(0,subset=['visit_count','typed_count','hidden','id'])
#fill NA for string/text null value fields
urls=urls.fillna('NA',subset=['url','title'])
```

In [6]:
```
#creating temp table
urls.createOrReplaceTempView('urls_t')
```

In [7]:
```
# Convert microseconds to seconds and then create a timestamp
spark.sql("SELECT id, url, title, visit_count, typed_count, hour(last_visit_time ) AS last_visit_time, hidden FI

#analzying the duration and no_of visits when a user is active on chrome
# Select the hour component from the 'last_visit_time' column
hour_visit_count=spark.sql('SELECT hour(last_visit_time) AS visit_hour, count(*) visits_count FROM urls_t group
hour_visit_count.show()
```

```
+----------+------------+
|visit_hour|visits_count|
+----------+------------+
|        17|         839|
|        15|         799|
|        19|         727|
|        22|         717|
|         1|         679|
|        18|         620|
|        16|         605|
|        20|         581|
|        23|         557|
|         2|         522|
|         0|         498|
|        21|         475|
|         5|         458|
|        14|         455|
|         3|         397|
|         4|         379|
|         6|         352|
|        13|         217|
|         7|         196|
|         8|         118|
+----------+------------+
only showing top 20 rows
```

In [8]:
```
#extract and perform join on url and visits for perform analysis later
connection = sqlite3.connect(history_db)
cursor = connection.cursor()

# # Execute a PRAGMA statement to get the column information for the "urls" table
cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")
tables=cursor.fetchall()
#print(tables)
cursor.execute("PRAGMA table_info(visits)")
columns = cursor.fetchall()
print(columns)
cursor.execute("SELECT * FROM visits limit 10")
rows = cursor.fetchall()
print(rows)
cursor.execute("SELECT v.url, u.title FROM urls u JOIN visits v ON u.id = v.from_visit limit 6")
rows = cursor.fetchall()
#print(rows)
cursor.close()
```

```
[(0, 'id', 'INTEGER', 0, None, 1), (1, 'url', 'INTEGER', 1, None, 0), (2, 'visit_time', 'INTEGER', 1, None, 0),
(3, 'from_visit', 'INTEGER', 0, None, 0), (4, 'transition', 'INTEGER', 1, '0', 0), (5, 'segment_id', 'INTEGER',
0, None, 0), (6, 'visit_duration', 'INTEGER', 1, '0', 0), (7, 'incremented_omnibox_typed_score', 'BOOLEAN', 1, '
FALSE', 0), (8, 'opener_visit', 'INTEGER', 0, None, 0), (9, 'originator_cache_guid', 'TEXT', 0, None, 0), (10, '
originator_visit_id', 'INTEGER', 0, None, 0), (11, 'originator_from_visit', 'INTEGER', 0, None, 0), (12, 'origin
ator_opener_visit', 'INTEGER', 0, None, 0), (13, 'is_known_to_sync', 'BOOLEAN', 1, 'FALSE', 0), (14, 'consider_f
or_ntp_most_visited', 'BOOLEAN', 1, 'FALSE', 0), (15, 'external_referrer_url', 'TEXT', 0, None, 0), (16, 'visite
d_link_id', 'INTEGER', 0, None, 0)]
[(1, 1, 13335066068378610, 0, 268435462, 0, 0, 0, 0, '', 0, 0, 0, 0, 1, None, None), (2, 2, 13335066068378610, 1
, -1610612730, 0, 1559833, 0, 0, '', 0, 0, 0, 0, 1, None, None), (3, 2, 13335066069936832, 0, 805306368, 0, 1171
676, 0, 2, '', 0, 0, 0, 0, 1, None, None), (4, 3, 13335066561529635, 0, 838860805, 0, 1647787, 0, 0, '', 0, 0, 0
, 0, 1, None, None), (5, 3, 13335066563175923, 0, 805306368, 0, 1522799, 0, 4, '', 0, 0, 0, 0, 1, None, None), (
6, 4, 13335066564696833, 5, 268435456, 0, 0, 0, 0, '', 0, 0, 0, 0, 1, None, None), (7, 5, 13335066564696833, 6,
-2147483648, 0, 0, 0, 0, '', 0, 0, 0, 0, 1, None, None), (8, 6, 13335066564696833, 7, -2147483648, 0, 0, 0, 0, '
', 0, 0, 0, 0, 1, None, None), (9, 7, 13335066564696833, 8, -2147483648, 0, 0, 0, 0, '', 0, 0, 0, 0, 1, None, No
ne), (10, 8, 13335066564696833, 9, -2147483648, 0, 0, 0, 0, '', 0, 0, 0, 0, 1, None, None)]
```

In [10]:
```
#extract the visited_links data into csv file
connection = sqlite3.connect(history_db)
cursor = connection.cursor()

# # Execute a PRAGMA statement to get the column information for the "urls" table
cursor.execute("SELECT name FROM sqlite_master WHERE type='table';")
tables=cursor.fetchall()
#print(tables)
cursor.execute("PRAGMA table_info(visited_links)")
columns = cursor.fetchall()
print(columns)
c=[]
for column in columns:
    x = column[1].replace("'", "").replace("{", "").replace("}", "")
    c.append(x)
cursor.execute("SELECT * FROM visited_links") #add if data is not present
rows = cursor.fetchall()
print(rows)

#print(rows)
cursor.close()
```

```
#write visits data to csv file
output_directory = "C:/Documents/jupternotebookprac/history_analysis"

csv_filename="visited_links.csv"
csv_path = os.path.join(output_directory, csv_filename)
with open(csv_filename, 'w', newline='',encoding='utf-8') as csv_file:
    csv_writer = csv.writer(csv_file)
    csv_writer.writerow(c)
    for row in rows:
        print(row)
        visit_time = row[2]
        visit_time = microseconds_to_datetime(visit_time)
        #print(visit_time)

        row = list(row)
        row[2] = visit_time
        #print(row)

        csv_writer.writerow(row)
print(f"Data has been written to {csv_path} , {output_directory}")
```

```
[(0, 'id', 'INTEGER', 0, None, 1), (1, 'link_url_id', 'INTEGER', 1, None, 0), (2, 'top_level_url', 'LONGVARCHAR'
, 1, None, 0), (3, 'frame_url', 'LONGVARCHAR', 1, None, 0), (4, 'visit_count', 'INTEGER', 1, '0', 0)]
[]
Data has been written to C:/Documents/jupternotebookprac/history_analysis\visited_links.csv , C:/Documents/jupte
rnotebookprac/history_analysis
```

In [11]:

```
#dervie the the source_link and just one_level above link details of a link
#read data from visits.csv and visited_links.csv
import pandas as pd
print(os.getcwd())
visits=spark.read.csv("visits.csv",header=True,dateFormat="MM/dd/yyyy HH:mm:ss")
#visits.show()

visited_links=spark.read.csv("visited_links.csv",header=True,dateFormat="MM/dd/yyyy HH:mm:ss")
#visited_links.show()



url_id=  urls[['id','title']]
url_id = url_id.withColumnRenamed("id", "url_id")
#drop if url_id is in visits
visits = visits.drop('url_id')
#print(visits.columns)

#print(url_id.columns)
#derive and add url name for id's  in visit using url table
visits = visits.join(url_id, (visits.url == url_id.url_id), how='inner')
visits= visits.withColumnRenamed("title", "url_title")

#derive and add visited url name based on visits table and urls table -1st level above
visits = visits.drop('url_id')
visits = visits.join(url_id, (visits.from_visit == url_id.url_id), how='inner')
visits = visits.withColumnRenamed("title", "prev_level_url_title")
visits.filter(visits['id']==12)
#visits.show()


# derive and add the url from the source level
visited_links_top_level_url=visited_links[['id','top_level_url','link_url_id']]
visited_links_top_level_url=visited_links_top_level_url.join(url_id, (visited_links_top_level_url.link_url_id ==
visited_links_top_level_url = visited_links_top_level_url.drop('url_id')

#print(visited_links_top_level_url.columns)
visited_links_top_level_url = visited_links_top_level_url.withColumnRenamed("title", "top_source_level_url_titl

#print(visits.columns)
#print(visited_links_top_level_url.columns)
#perform join

visits=visits.join(visited_links_top_level_url,(visits.id==visited_links_top_level_url.source_visit_link_id),hov

#print(visits.columns)
visits.filter(visits['id']==12).show()
linklevel_details=visits.select('id','url','url_title','visit_time','from_visit','prev_level_url_title','source_
print("*****************")
linklevel_details.show(truncate=False)
```

```
C:\Users\ganes\Documents\Manju\history_analysis
+---+---+----------+----------+----------+----------+--------------+------------------------------+------------
+--------------------+--------------------+--------------------+-----------------------+---------------+------
----------------------+----------------------+---------------+---------+------+--------------------+------------
--------+-------------+-----------+--------------------------+
| id|url|visit_time|from_visit|transition|segment_id|visit_duration|incremented_omnibox_typed_score|opener_visit
|originator_cache_guid|originator_visit_id|originator_from_visit|originator_opener_visit|is_known_to_sync|consid
er_for_ntp_most_visited|external_referrer_url|visited_link_id|url_title|url_id|prev_level_url_title|source_visit
_link_id|top_level_url|link_url_id|top_source_level_url_title|
+---+---+----------+----------+----------+----------+--------------+------------------------------+------------
+--------------------+--------------------+--------------------+-----------------------+---------------+------
----------------------+----------------------+---------------+---------+------+--------------------+------------
--------+-------------+-----------+--------------------------+
+---+---+----------+----------+----------+----------+--------------+------------------------------+------------
+--------------------+--------------------+--------------------+-----------------------+---------------+------
----------------------+----------------------+---------------+---------+------+--------------------+------------
--------+-------------+-----------+--------------------------+

****************
+---+---+---------+----------+----------+--------------------+--------------------+--------------------------+
|id |url|url_title|visit_time|from_visit|prev_level_url_title|source_visit_link_id|top_source_level_url_title|
+---+---+---------+----------+----------+--------------------+--------------------+--------------------------+
+---+---+---------+----------+----------+--------------------+--------------------+--------------------------+
```

In [12]:
```
#print the one level above link for url_id 5
five=urls.filter(urls['id']==5)
five.select("title").show(truncate=False)
```

```
+--------------------------------------------------+
|title                                             |
+--------------------------------------------------+
|Inbox (1,170) - ganeshvadlamuri15@gmail.com - Gmail|
+--------------------------------------------------+
```

In [13]:
```
#visits.write.mode("overwrite").json("C:/Documents/jupternotebookprac/history_analysis/visits")
```

In [15]:
```
#read keyword_search_terms into dataframe
keyword_search_terms=spark.read.csv("keyword_search_terms.csv",header=True,dateFormat="MM/dd/yyyy HH:mm:ss")
keyword_search_terms.createOrReplaceTempView("keyword_search_terms_t")
#spark.sql('select url_id,term, count(*) from keyword_search_terms_t group by url_id, term').show()

#derive number of times a search term appeared
spark.sql('select term, count(*) from keyword_search_terms_t group by term order by count(*) desc').show(trunca
```

```
+----------------------------------------+--------+
|term                                    |count(1)|
+----------------------------------------+--------+
|professionalcomputer science resumes    |87      |
|handshake texas state                   |31      |
|self test kit covid                     |16      |
|walmart customer service                |16      |
|resume worded software engineer         |15      |
|second hand iphone stores san antonio texas|12      |
|india time now                          |12      |
|indian restaurants in dallas            |10      |
|gas station near me                     |10      |
|virat kohli                             |9       |
|sbs texas state                         |8       |
|chocolates shops in usa                 |8       |
|honors college texas state              |8       |
|EVO Entertainment Springtown San Marcos |7       |
|quatar airways                          |7       |
|birthday decoration room ideas          |7       |
|trending hairstyles 2023 for men        |7       |
|gantt chart                             |7       |
|people do ignore the truths             |6       |
|computer security projects              |6       |
+----------------------------------------+--------+
only showing top 20 rows
```

In [42]:
```
#select no of successful downloads for every link
from pyspark.sql.functions import when
downloads_state=spark.sql('select state, count(*) as count from downloads_t group by state order by state desc'
downloads_state = downloads_state.withColumn(
    'status',
    when(downloads_state['state'] == 1, 'Successful')
    .when(downloads_state['state'] == 2, 'Interrupted')
    .when(downloads_state['state'] == 4, 'Canceled')
)
downloads_state.show()
downloads_state.createOrReplaceTempView('downloads_state_t')
```

```
#include the status column to dowloads table
downloads_state = downloads_state.withColumnRenamed("state", "state_id")
downloads_state=downloads_state[['state_id','status']]
#print(downloads_state.columns)
#print(downloads.columns)
#drop status from downloads if exists
downloads=downloads.drop('status')
downloads=downloads.join(downloads_state, (downloads.state == downloads_state.state_id), how='inner')
downloads=downloads.drop('state_id')
#ensure there are no duplicate columns
#print(downloads.columns)
#downloads.show()
downloads.createOrReplaceTempView('downloads_t')
```

```
+-----+-----+-----------+
|state|count|     status|
+-----+-----+-----------+
|    4|    1|   Canceled|
|    2|   26|Interrupted|
|    1|  132| Successful|
+-----+-----+-----------+

['id', 'guid', 'current_path', 'target_path', 'start_time', 'received_bytes', 'total_bytes', 'state', 'danger_ty
pe', 'interrupt_reason', 'hash', 'end_time', 'opened', 'last_access_time', 'transient', 'referrer', 'site_url',
'embedded_data', 'tab_url', 'tab_referrer_url', 'http_method', 'by_ext_id', 'by_ext_name', 'by_web_app_id', 'eta
g', 'last_modified', 'mime_type', 'original_mime_type', 'state_id', 'state_id', 'state_id', 'state_id', 'state_i
d', 'state_id']
['id', 'guid', 'current_path', 'target_path', 'start_time', 'received_bytes', 'total_bytes', 'state', 'danger_ty
pe', 'interrupt_reason', 'hash', 'end_time', 'opened', 'last_access_time', 'transient', 'referrer', 'site_url',
'embedded_data', 'tab_url', 'tab_referrer_url', 'http_method', 'by_ext_id', 'by_ext_name', 'by_web_app_id', 'eta
g', 'last_modified', 'mime_type', 'original_mime_type', 'status']
```

In [59]:

```
#get the stats,reasons sbout why downloads sucessfull,interrupted, cancelled

#read interrupt_reason_description from csv file
interrupt_reasons_csv=spark.read.csv('interrupted_reason_csv.csv',header=True)
interrupt_reasons_csv.show()
#join the downloads with interrupt_reasons_csv to add interrupt reason description
downloads=downloads.drop('interrupt_id','interrupt_reason_description')
#print(downloads.columns)
downloads=downloads.join(interrupt_reasons_csv,(interrupt_reasons_csv.interrupt_id==downloads.interrupt_reason)
downloads=downloads.drop('interrupt_id')
print(downloads.columns)
downloads.createOrReplaceTempView('downloads_t')

#derive stats
interrupted_downloads = spark.sql('select d.status,d.interrupt_reason_description, count(*) from downloads_t d \
group by  d.status,d.interrupt_reason,d.interrupt_reason_description \
order by d.status,interrupt_reason,count(*) desc')
interrupted_downloads.show(truncate=False)
interrupted_downloads.createOrReplaceTempView('interrupted_downloads_t')
```

```
+------------+--------------------------+
|interrupt_id|interrupt_reason_description|
+------------+--------------------------+
|           0|        No Interrupt Success|
|           1|                File Error|
|           2|              Access Denied|
|           3|                  Disk Full|
|           5|              Path Too Long|
|           6|             File Too Large|
|           7|                     Virus|
|          10|           Temporary Problem|
|          11|                   Blocked|
|          12|        Security Check Fa...|
|          13|               Resume Error|
|          20|              Network Error|
|          30|               Server Error|
|          40|        User Input Interr...|
|          50|                     Crash|
+------------+--------------------------+
```

```
['id', 'guid', 'current_path', 'target_path', 'start_time', 'received_bytes', 'total_bytes', 'state', 'danger_ty
pe', 'interrupt_reason', 'hash', 'end_time', 'opened', 'last_access_time', 'transient', 'referrer', 'site_url',
'embedded_data', 'tab_url', 'tab_referrer_url', 'http_method', 'by_ext_id', 'by_ext_name', 'by_web_app_id', 'eta
g', 'last_modified', 'mime_type', 'original_mime_type', 'status', 'interrupt_reason_description']
+-----------+------------------------------+--------+
|status     |interrupt_reason_description  |count(1)|
+-----------+------------------------------+--------+
|Canceled   |Virus                         |1       |
|Interrupted|User Input Interrupted Download|25      |
|Successful |No Interrupt Success          |130     |
|Successful |Network Error                 |2       |
+-----------+------------------------------+--------+
```

In [ ]: