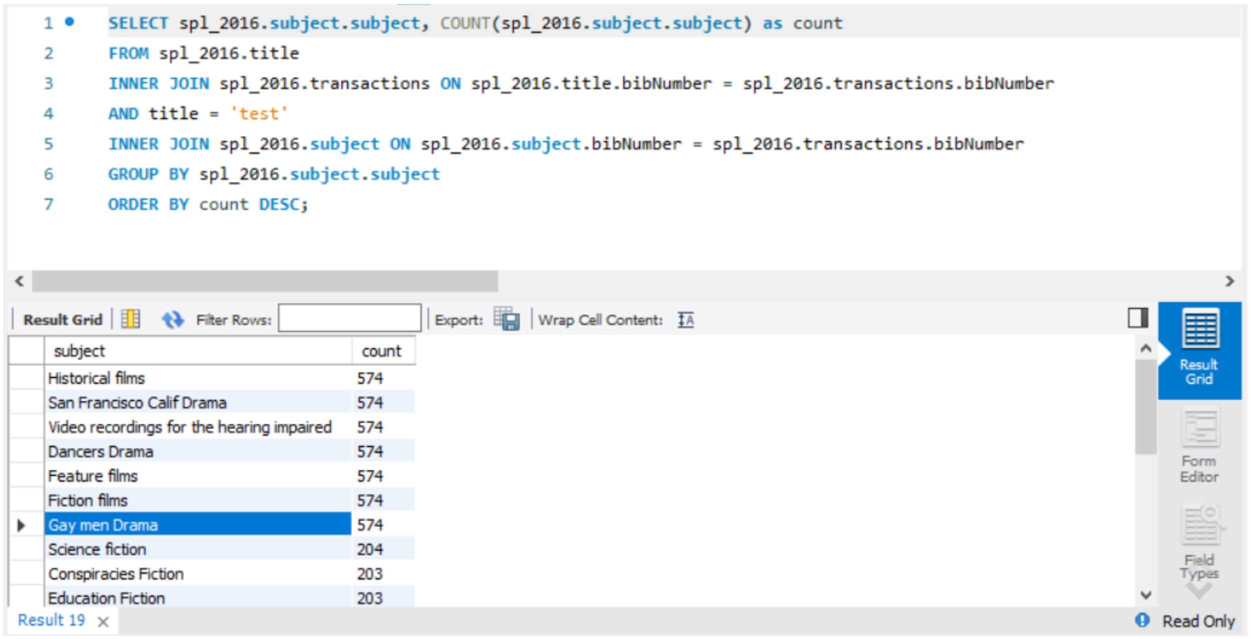
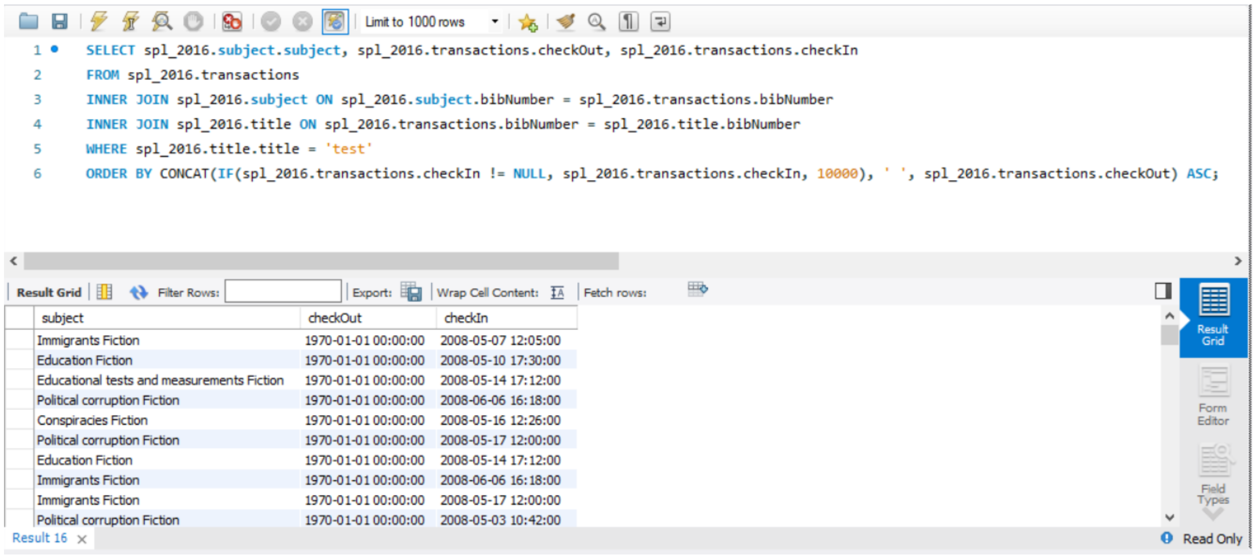
I was interested in the anomaly of repeat titles in the database of items. My first query revealed that the most commonly repeated titles with unique bibNumbers were simply “ “ and “test”.



```sql
SELECT spl_2016.subject.subject, COUNT(spl_2016.subject.subject) as count
FROM spl_2016.title
INNER JOIN spl_2016.transactions ON spl_2016.title.bibNumber = spl_2016.transactions.bibNumber
AND title = 'test'
INNER JOIN spl_2016.subject ON spl_2016.subject.bibNumber = spl_2016.transactions.bibNumber
GROUP BY spl_2016.subject.subject
ORDER BY count DESC;
```

subject	count
Historical films	574
San Francisco Calif Drama	574
Video recordings for the hearing impaired	574
Dancers Drama	574
Feature films	574
Fiction films	574
Gay men Drama	574
Science fiction	204
Conspiracies Fiction	203
Education Fiction	203

I became curious about the nature of these ‘test’ entries. My next query was into all subjects of entries titled “test”. It appeared to me that subject lines seemed only partially random. They appeared arbitrary and telling.

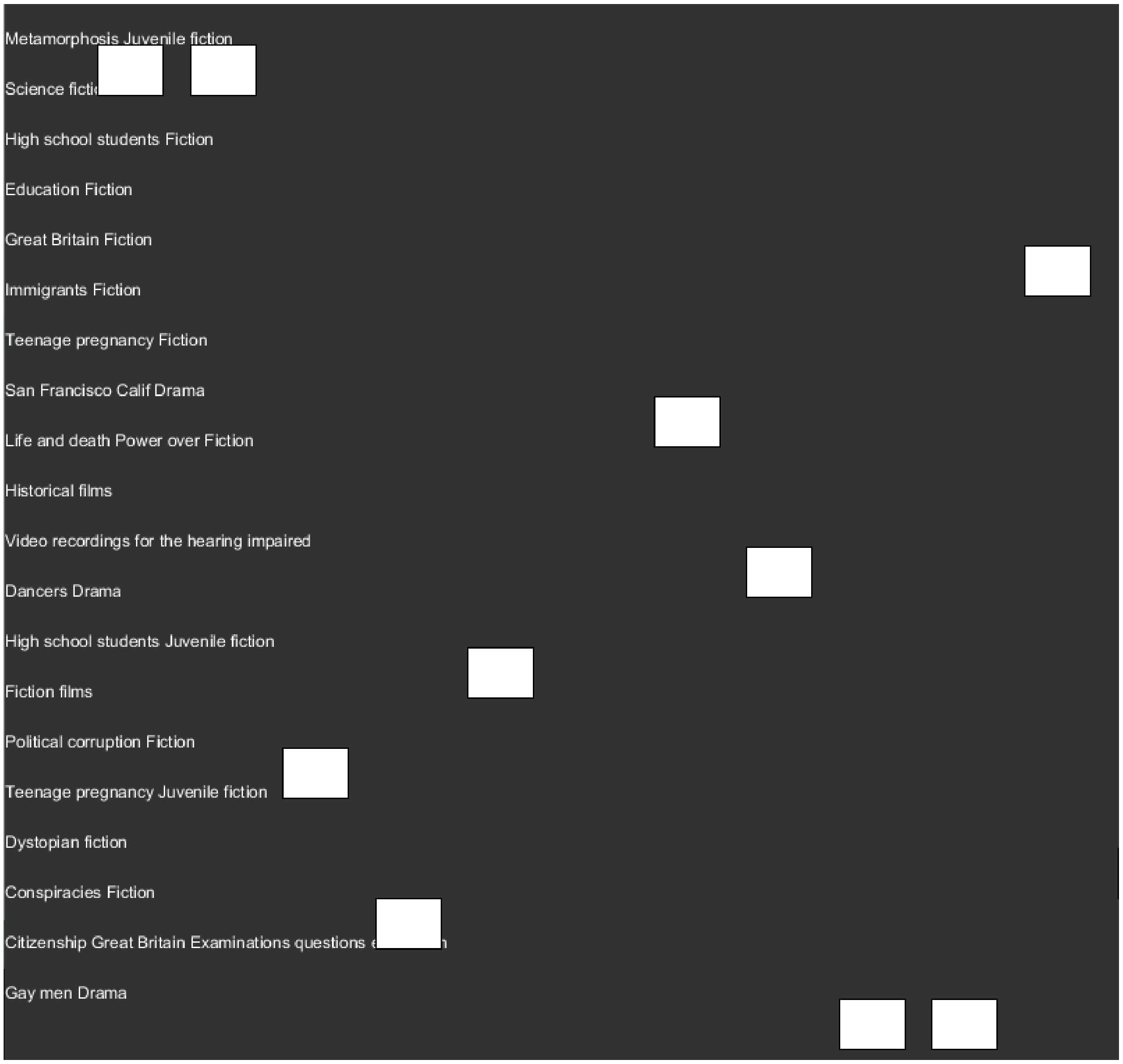


```sql
SELECT spl_2016.subject.subject, spl_2016.transactions.checkOut, spl_2016.transactions.checkIn
FROM spl_2016.transactions
INNER JOIN spl_2016.subject ON spl_2016.subject.bibNumber = spl_2016.transactions.bibNumber
INNER JOIN spl_2016.title ON spl_2016.transactions.bibNumber = spl_2016.title.bibNumber
WHERE spl_2016.title.title = 'test'
ORDER BY CONCAT(IF(spl_2016.transactions.checkIn != NULL, spl_2016.transactions.checkIn, 10000), ' ', spl_2016.transactions.checkOut) ASC;
```

subject	checkOut	checkIn
Immigrants Fiction	1970-01-01 00:00:00	2008-05-07 12:05:00
Education Fiction	1970-01-01 00:00:00	2008-05-10 17:30:00
Educational tests and measurements Fiction	1970-01-01 00:00:00	2008-05-14 17:12:00
Political corruption Fiction	1970-01-01 00:00:00	2008-06-06 16:18:00
Conspiracies Fiction	1970-01-01 00:00:00	2008-05-16 12:26:00
Political corruption Fiction	1970-01-01 00:00:00	2008-05-17 12:00:00
Education Fiction	1970-01-01 00:00:00	2008-05-14 17:12:00
Immigrants Fiction	1970-01-01 00:00:00	2008-06-06 16:18:00
Immigrants Fiction	1970-01-01 00:00:00	2008-05-17 12:00:00
Political corruption Fiction	1970-01-01 00:00:00	2008-05-03 10:42:00

My question became: “Why would someone put a particular subject line for test titles”. So my next step was to query the timestamps of check ins and check outs associated with each subject line. I realized that there were only a few subjects, although there were nearly 5000 entries. Someone had put in tests with arbitrarily chosen subject lines, and they were isolated to a few points in time. Who were these people? Was there some pattern in time that I could track to discover when they did this?

So my next was to take that data over to Processing and plot it with time on the x axis and subject categories on the y. That produced this result:



The Following is my Processing Code:

```
import java.util.*;

Table table;
```

```
HashMap<String, PVector> subjectText;
boolean[][] dataGrid;
float cellHeight;
float cellWidth;

void setup() {
  // initialize window size
  size(800,800);

  // construct the display structures
  subjectText = new HashMap<String, PVector>();
  dataGrid = new boolean[width][height];

  // Prepare the intermediate structures
  HashMap<Float, ArrayList<String>> dates;
  dates = new HashMap<Float, ArrayList<String>>();
  table = loadTable("CSVs/SubjectsOverTimeForThingsTitledTest.csv", "header");

  // Prepare a variable for keeping track of the max date
  float maxDate = 0;

  // reformat the data into subjectText and dates
  for (TableRow row : table.rows()) {
    String subject = row.getString("subject");

    // skip empty subject lines
    if (subject.isEmpty()){
      continue;
    }

    // if this particular subject isn't in the hashmap, then add it
    if (!subjectText.containsKey(subject)) {
      subjectText.put(subject, new PVector());
    }

    // collect all the dates
    String checkInDate = row.getString("checkIn").split(" ", 0)[0];
    if (!checkInDate.equals("NULL")) {
      // convert the date string into a floating point number
      int year = int(checkInDate.split("-", 0)[0]);
      int month = int(checkInDate.split("-", 0)[1]);
      int day = int(checkInDate.split("-", 0)[2]);
      float date = year + month / 12 + day / 365;
      // put the date into the dates hashmap if it isn't already
      if (!dates.containsKey(date)) {
        dates.put(date, new ArrayList<String>());
      }
      // add that this particular subject was included on that date
      dates.get(date).add(subject);
      // update the max date variable with the most max date to date
      if (date > maxDate) {
        maxDate = date;
      }
    }
    String checkOutDate = row.getString("checkOut").split(" ", 0)[0];
    if (!checkInDate.equals("NULL")) {
      int year = int(checkOutDate.split("-", 0)[0]);
      int month = int(checkOutDate.split("-", 0)[1]);
      int day = int(checkOutDate.split("-", 0)[2]);
      float date = year + month / 12 + day / 365;
      if (!dates.containsKey(checkOutDate)) {
        dates.put(date, new ArrayList<String>());
      }
      dates.get(date).add(subject);
      if (date > maxDate) {
        maxDate = date;
      }
    }
  }

  // Calculate the cell heights and widths
  cellHeight = height / subjectText.keySet().size();
  cellWidth = width / dates.keySet().size();

  // store the proper y axis value for each subject text line
  int i = 0;
  for (String subject : subjectText.keySet()) {
    // Figure out where the text goes
    float subjectHeight = i * cellHeight;
    subjectText.get(subject).set(new PVector(0, subjectHeight));
    i++;
  }

  // set the cells in the dataGrid that fall where the subjects and dates intersect to 'true'
  for (Float date : dates.keySet()) {
    float normDate = ((date-2008) / (maxDate-2008));
    int xPos = floor((width - 1) * normDate);

    for (String subject : dates.get(date)){
      int yPos = floor(subjectText.get(subject).y);
      if (xPos > 0 && yPos > 0){
        dataGrid[xPos][yPos] = true;
      }
    }
  }

}

void draw() {
  // clear the screen
  background(50);
  // draw the subjectText on the screen
  for (String subject : subjectText.keySet()) {
    text(subject, subjectText.get(subject).x, subjectText.get(subject).y);
  }
  // draw rectangles at the positions where the subjects and dates intersect (which is where there
are 'true' in the dataGrid)
  for (int i = 0; i < width; i++){
    for (int j = 0; j < height; j++){
      if(dataGrid[i][j]){
        rect(i,j,cellWidth,cellHeight);
      }
    }
  }


}
```