Frequently **Borrowed** Together

The Question
I wanted to create a query that would return a pairs of items that were borrowed together and the number of times the items were borrowed together. I believe this is an interesting question because it could allow us to create a simple 'recommendation' program similar to Amazon's 'Frequently bought together' feature. I also wanted to see if there are any interesting data patterns hat might give insights into user behavior.

The Approach
In order to find items that were checked out together, I assumed that items that had the same checkout and checkin timestamp were likely to be checked out by the same person. It should be noted that this assumption might lead to false positive and false negative results. It is possible that two individuals happen to check out and return a book at the exact same time; maybe friends who visit the library together. It is also possible that someone checked out multiple books and did not return them all at the same time.

The Query
In order to perform the search, I looked at the cartesian product of the inraw table with itself and selected the items that had the same checkout and checkin times. The data was then grouped to come up with a count of times items appear together. I also wanted to look at the dewey number to see if there were any items that fell into different dewey classes but were somehow related because they were frequently checked out together. I narrowed down my search by only looking at books that contained the word 'algorithms' in the title in order to reduce the processing time (Query 1). However, this query was too intensive and multiple attempts resulted in failures.

Query 1

```
SELECT
    t1.title,
    t2.title AS title2,
    COUNT(t2.title) AS freq,
    t1.deweyClass AS dewey1,
    t2.deweyClass AS dewey2
FROM
    spl2.inraw AS t1,
    spl2.inraw AS t2
WHERE
    t1.cout = t2.cout AND t1.cin = t2.cin
        AND t1.itemNumber IN (SELECT
            itemNumber
        FROM
            spl2.inraw
        WHERE
            title LIKE '%algorithms%')
        AND t2.title != ''
        AND t1.itemNumber != t2.itemNumber
GROUP BY title2
ORDER BY freq DESC
LIMIT 100;
```

Processing time: NA

I decided to only look at the most frequently checked out book containing the word ‘algorithm’. A subquery (Query 2) was used to get the item number of the most popular (most checkouts) book.

Query 2

```
SELECT
    itemNumber
FROM
    (SELECT
        itemNumber, COUNT(cout) AS checkouts
    FROM
        spl2.inraw
    WHERE
        title LIKE '%algorithms%'
    GROUP BY itemNumber) AS frequency
WHERE
    checkouts = (SELECT
            MAX(checkouts)
        FROM
            (SELECT
                itemNumber, COUNT(cout) AS checkouts
            FROM
                spl2.inraw
            WHERE
                title LIKE '%algorithms%'
            GROUP BY itemNumber) AS frequency);
```

Processing time: 107.943 seconds

This query was then embedded into the original query to return only the books that were checked out with the most popular book.

Query 3

```
SELECT
    t1.title,
    t2.title AS title2,
    COUNT(t2.title) AS freq,
    t1.deweyClass AS dewey1,
    t2.deweyClass AS dewey2
FROM
    spl2.inraw AS t1,
    spl2.inraw AS t2
WHERE
    t1.cout = t2.cout AND t1.cin = t2.cin
        AND t1.itemNumber = (SELECT
            itemNumber
        FROM
            (SELECT
                itemNumber, COUNT(cout) AS checkouts
            FROM
```

```
            spl2.inraw
        WHERE
            title LIKE '%algorithms%'
        GROUP BY itemNumber) AS frequency
    WHERE
        checkouts = (SELECT
                MAX(checkouts)
            FROM
                (SELECT
                    itemNumber, COUNT(cout) AS checkouts
                FROM
                    spl2.inraw
                WHERE
                    title LIKE '%algorithms%'
                GROUP BY itemNumber) AS frequency))
    AND t2.title != ''
    AND t1.itemNumber != t2.itemNumber
GROUP BY title2
ORDER BY freq DESC;
```

Processing time: 124.223 seconds

Interesting Findings

Although the resulting dataset was very small, it did provide some insights to help formulate interesting hypotheses about user behavior. The 'most popular' book was in the 005 (Computer programming, programs & data) dewey category. According to the data, the two books that were most frequently checked out with the 'most popular' book fell into similar dewey categories as the 'most popular' book (one belonged to category 005 and the other to 006-Special computer methods). Some other interesting patters can be indirectly observed in the dataset. We can see that there are two entries where books in the 294 (Religion of Indic origin) dewey category were checked out along with the book about algorithms. There are also two entries in the 919 (Geography and travel in Australia…) category. We might assume that these books were checked out by the same user. In fact, checking the checkout time for the books reveals that the time stamp on both entries is the same.

From this very limited experiment we can create the hypothesis that the books that are more frequently checked out together are likely to fall within closely related dewey categories. We can also hypothesize that if we see multiple books from an unrelated dewey category, they were likely checked out by the same user.

I would like to create a visualization of a graph with books as the vertices and the edges representing simultaneous checkouts. Because it is difficult to create this data structure with SQL, I would get the raw data from inraw and process it in Java to create the data structure I need. For example, the following query returns 50,000 rows in 0.0026 seconds and provides all the data I need to create the graph.

Query 4

```
SELECT
    title, cout, cin, deweyClass
FROM
    spl2.inraw;
```

Title 1	Title 2	Freq	Dewey 1	Dewey 2
Data mining concepts models methods and algorithms	Data mining practical machine learning tools and techniques	2	5.741	6.3
Data mining concepts models methods and algorithms	Absolute beginners guide to databases	2	5.741	5.74
Data mining concepts models methods and algorithms	EBay business all in one desk reference for dummies	1	5.741	381.177
Data mining concepts models methods and algorithms	Relating to a spiritual teacher building a healthy relationship	1	5.741	294.361
Data mining concepts models methods and algorithms	Luminous emptiness understanding the Tibetan book of the dead	1	5.741	294.3423
Data mining concepts models methods and algorithms	Creating Web pages for dummies	1	5.741	5.72
Data mining concepts models methods and algorithms	Beginning Visual basic SQL server 7 0	1	5.741	5.7585
Data mining concepts models methods and algorithms	Poor Richards web site geek free commonsense advice on building a low cost web site	1	5.741	5.276
Data mining concepts models methods and algorithms	Emergence from chaos to order	1	5.741	3.85
Data mining concepts models methods and algorithms	JavaScript the definitive guide	1	5.741	5.133
Data mining concepts models methods and algorithms	thrill of it all the story of Bryan Ferry Roxy Music	1	5.741	782.42166
Data mining concepts models methods and algorithms	Ten lessons to transform your marriage Americas love lab experts share their strategies for strengthening your relationship	1	5.741	306.81
Data mining concepts models methods and algorithms	Ajax for dummies	1	5.741	5.133
Data mining concepts models methods and algorithms	Movement for actors	1	5.741	792.028
Data mining concepts models methods and algorithms	Learning Python	1	5.741	5.133
Data mining concepts models methods and algorithms	Pacific crucible war at sea in the Pacific 1941 1942	1	5.741	940.5426
Data mining concepts models methods and algorithms	Cracking the GMAT	1	5.741	650.076
Data mining concepts models methods and algorithms	Kauai	1	5.741	919.69404
Data mining concepts models methods and algorithms	Frommers Kauai	1	5.741	919.6941