Crochet Pattern/Project Database

Mary McSweeney – 106179736

2025-06-13

```
library(DBI)
library(RSQLite)
```

```
con <- dbConnect(RSQLite::SQLite(), "crochet.db")
```

Introduction

My favorite hobby is to crochet. There are millions of patterns online to help guide you in making pretty much anything you can think of. However, things can get complicated when I see so many patterns that I am interesting in starting while I also have so many projects in progress. There have also been many time when I used a pattern and wanted to make it again, but could never remember where the pattern was from. Additionally, keeping track of the yarn and hook size used for each project can sometimes be difficult, and using the wrong hook or a different yarn, even if its the same general color, can have huge consequences on the final Projects.

For these reasons, I decided to make a database to keep track of everything I need to keep track of my projects.

Tables

My most important table is Projects. This is where I keep track of projects I have started, including which pattern and hook size was used as well as the start date, end date, and a few other columns to help me keep track of it all. Here I can easily see how many projects I have in progress and find the pattern again.

To keep track of which yarn I used for the project, I created the Project_Yarn table as there is a many to many relationship. Each project can have multiple yarn brands or even the same yarn brand in multiple colors, so having a table with the project id as well as the yarn id solves this problem.

The next table is Patterns which keeps track of important information about each pattern, such as the name, type, price, and designer id. This is useful for searching because I can filter patterns by how much I am looking to spend, the hook size I want to use, and more.

Similar to projects, patterns can have suggested yarn to use and they can use more than one in the pattern. The Pattern_Yarn table fixes this many to many relationship the same way as Project_Yarn, using pattern id and yarn id.

Pattern_Yarn and Project_Yarn are both connected to the Yarn table, which keeps track of all the yarn used in either a pattern or a project. In general, since each brand can have tons of colors, it would probably be better to have color in its own table for a large database, but in this case it worked for me to just have each variation of a brand in its own row. This table is useful for searching because I can filter by color, weight, fiber content, and more to find a pattern or project.

Lastly, I have the Designers table because there are many designers who create many patterns I like. Here, I can easily find their name and website because chances are, if I like one pattern from a designer, I will want

to look at their other patterns.

The diagram below shows the relationships between tables as well as their columns. Primary keys are noted with a key, and foreign keys are noted with a chain. The primary keys for Pattern_Yarn and Project_Yarn are made up of unique combinations of project/pattern id and yarn id.

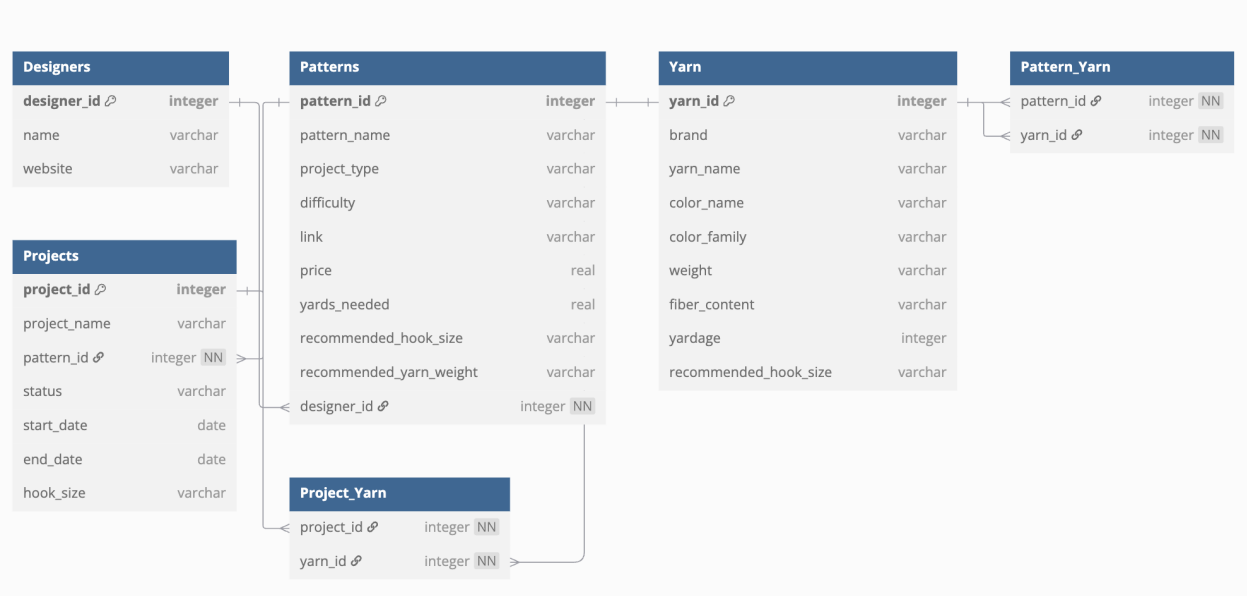

Figure 1: "Entity Relationship Diagram, Source: dbdiagram.io"

Data

I populated this data set with 50 patterns that looked interesting to me from Ravelry, a website used for pattern sharing. To fill the project table, I picked some patterns and created sample data. I create the tables and filled in the data manually, using DB Browser. In the future, I can actually use real projects I have done and keep track of them there.

Keeping track of all of this information in a relational database is a better option that a single spreadsheet because of all of the one-to-many and many-to-many relationships. This allows for easier look up and fast queries of any information I would like to find out.

Queries

Which designers have more than one pattern in Patterns?

This is an important question to answer because it shows which designers I tend to like the most patterns of. With this information, I can look at what other patterns these designers have made to see if I like any of those as well with hopefully a higher success rate.

```
sql_query_1 <- "
  SELECT Designers.designer_id, name, COUNT(pattern_id) AS pattern_count
  FROM Designers
  INNER JOIN Patterns
  ON Designers.designer_id = Patterns.designer_id
  GROUP BY Designers.designer_id, name
  HAVING COUNT(pattern_id) > 1;
```

```
"
dbGetQuery(con, sql_query_1)
```

```
##   designer_id                  name pattern_count
## 1           1      K.A.M.E. Crochet             3
## 2           2        Heather Brooke             2
## 3           4           Toni Lipsey             3
## 4          11  Crochet 365 Knit Too             2
## 5          14             Justine T.             2
## 6          18 Stephanie Jessica Lau             2
## 7          19         Sweet Softies             2
```

K.A.M.E has the most patterns I was interested in, with 3 different one while there are a few other designers with multiple patterns as well.

2. Which yarn is the most used in patterns and how is it split by color?

It can get expensive to buy new yarn every time I want to make a new project, so it would be nice to buy one type of yarn that can be used for multiple projects. Additionally, I have found that when using yarn with the same name in different colors, the project will come out the most consistently.

```
sql_query_2 <- "
  WITH YarnUsage AS (
    SELECT Yarn.yarn_name,
           COUNT(Pattern_Yarn.pattern_id) AS total_uses
    FROM Yarn
    INNER JOIN Pattern_Yarn
    ON Yarn.yarn_id = Pattern_Yarn.yarn_id
    GROUP BY Yarn.yarn_name
  ),
  MostUsedYarn AS (
    SELECT yarn_name
    FROM YarnUsage
    ORDER BY total_uses DESC
    LIMIT 1
  )
  SELECT
         Yarn.yarn_id,
         Yarn.yarn_name,
         Yarn.color_name,
         COUNT(Pattern_Yarn.pattern_id) AS uses_by_color
  FROM Yarn
  INNER JOIN Pattern_Yarn
  ON Yarn.yarn_id = Pattern_Yarn.yarn_id
  WHERE Yarn.yarn_name = (SELECT yarn_name FROM MostUsedYarn)
  GROUP BY Yarn.yarn_name, Yarn.color_name
  ORDER BY uses_by_color DESC;
"
dbGetQuery(con, sql_query_2)
```

```
##   yarn_id   yarn_name color_name uses_by_color
## 1       5 24/7 Cotton  Goldenrod             3
## 2      18 24/7 Cotton      Denim             2
```

Here I can see that a good yarn for me to buy would be 24/7 Cotton in either the color Goldenrod or Denim.

3. Which of my projects reused yarn from a previous completed project?

Using leftover yarn for a different project is the best case scenario because then my yarn stash doesn't get too large and I can spend a little less money.

```
sql_query_3 <- "
  WITH CompletedYarn AS (
    SELECT
      Project_Yarn.yarn_id,
      Projects.end_date
    FROM Projects
    INNER JOIN Project_Yarn
    ON Projects.project_id = Project_Yarn.project_id
    WHERE Projects.end_date IS NOT NULL
  )
  SELECT
    Projects.project_id,
    Patterns.pattern_name,
    Yarn.yarn_name
  FROM Projects
  INNER JOIN Project_Yarn
  ON Projects.project_id = Project_Yarn.project_id
  INNER JOIN CompletedYarn
  ON Project_Yarn.yarn_id = CompletedYarn.yarn_id
     AND Projects.start_date > CompletedYarn.end_date
  INNER JOIN Patterns
  ON Projects.pattern_id = Patterns.pattern_id
  INNER JOIN YARN
  ON Project_Yarn.yarn_id = Yarn.yarn_id
"
dbGetQuery(con, sql_query_3)
```

```
##   project_id           pattern_name                       yarn_name
## 1          5  Crochet Seafarers Cap         Wool Ease Thick & Quick
## 2          6 One Hour No Sew Turtle A Star is Born: Oh Baby Organic
```

This shows that I have only reused yarn for two different projects, so I will keep that in mind and try to reuse yarn more often.

4. Which designers have the highest average yarn weight in their patterns?

Yarn weight refers to the thickness of the yarn. While weight usually has a name, they can are also categorized into numeric values with 0 being as thin as a thread, and 7 being the thickest yarn available. If I want a fast project, thicker yarn builds up much faster than thinner so it would be nice to know which designers usually use higher weight yarns.

```
sql_query_4 <- "
  WITH YarnWeightNum AS (
    SELECT
      Patterns.pattern_id,
      Patterns.designer_id,
      CASE Yarn.weight
        WHEN 'Thread' THEN 0
        WHEN 'Lace' THEN 0
        WHEN 'Fingering' THEN 1
        WHEN 'Sport' THEN 2
```

```
        WHEN 'DK' THEN 3
        WHEN 'Worsted' THEN 4
        WHEN 'Aran' THEN 4
        WHEN 'Bulky' THEN 5
        WHEN 'Super Bulky' THEN 6
        WHEN 'Jumbo' THEN 7
        ELSE NULL
      END AS weight_value
    FROM Patterns
    INNER JOIN Pattern_Yarn
    ON Patterns.pattern_id = Pattern_Yarn.pattern_id
    INNER JOIN Yarn
    ON Pattern_Yarn.yarn_id = Yarn.yarn_id
    WHERE Yarn.weight IS NOT NULL
  )
  SELECT
    Designers.designer_id,
    Designers.name,
    AVG(YarnWeightNum.weight_value) AS avg_weight
  FROM YarnWeightNum
  INNER JOIN Designers
  ON YarnWeightNum.designer_id = Designers.designer_id
  GROUP BY Designers.designer_id, Designers.name
  ORDER BY avg_weight DESC
  LIMIT 5;
"
dbGetQuery(con, sql_query_4)
```

```
##   designer_id                 name avg_weight
## 1           7    Handmade by Annie          6
## 2          22    Jocelyn Elizabeth          6
## 3          24        Diane Serviss          6
## 4          31           Kali Dahle          6
## 5          37 Jade Gauthier-Boutin          6
```

By converting weight to a number, I can easily sort to see which designers prefer thicker yarn. Since this database is small and many designers have only one pattern, these results are not the most meaningful, but running this query when the database is much larger would help me find designers who use the thickest yarn.

5. Which patterns have been made into projects using yarn heavier than recommended?

Something nice about crochet is that if a pattern specifies a small hook size and low weight yarn, you can actually follow the same pattern with a bigger hook and thicker yarn and it will still come out proportional, just much bigger. I would like to see how often I have followed a pattern but used thicker yarn, resulting in a larger result.

```
sql_query_5 <- "
  WITH WeightScale AS (
    SELECT
      Yarn.yarn_id,
      CASE Yarn.weight
        WHEN 'Thread' THEN 0
        WHEN 'Lace' THEN 0
        WHEN 'Fingering' THEN 1
```

```
        WHEN 'Sport' THEN 2
        WHEN 'DK' THEN 3
        WHEN 'Worsted' THEN 4
        WHEN 'Aran' THEN 4
        WHEN 'Bulky' THEN 5
        WHEN 'Super Bulky' THEN 6
        WHEN 'Jumbo' THEN 7
        ELSE NULL
      END AS project_weight_value
    FROM Yarn
  ),
  PatternWeights AS (
    SELECT
      Patterns.pattern_id,
      CASE Patterns.recommended_yarn_weight
        WHEN 'Thread' THEN 0
        WHEN 'Lace' THEN 0
        WHEN 'Fingering' THEN 1
        WHEN 'Sport' THEN 2
        WHEN 'DK' THEN 3
        WHEN 'Worsted' THEN 4
        WHEN 'Aran' THEN 4
        WHEN 'Bulky' THEN 5
        WHEN 'Super Bulky' THEN 6
        WHEN 'Jumbo' THEN 7
        ELSE NULL
      END AS recommended_weight_value
    FROM Patterns
  )
  SELECT
    Projects.project_id,
    Patterns.recommended_yarn_weight,
    Yarn.weight AS Project_Yarn_weight
  FROM Projects
  INNER JOIN Patterns
  ON Projects.pattern_id = Patterns.pattern_id
  INNER JOIN Project_Yarn
  ON Projects.project_id = Project_Yarn.project_id
  INNER JOIN Yarn
  ON Project_Yarn.yarn_id = Yarn.yarn_id
  INNER JOIN WeightScale
  ON Yarn.yarn_id = WeightScale.yarn_id
  INNER JOIN PatternWeights
  ON Patterns.pattern_id = PatternWeights.pattern_id
  WHERE WeightScale.project_weight_value > PatternWeights.recommended_weight_value;
"
dbGetQuery(con, sql_query_5)
```

```
##   project_id recommended_yarn_weight Project_Yarn_weight
## 1          2                Worsted               Bulky
## 2          5                Worsted         Super Bulky
## 3          8                Worsted               Bulky
## 4          9                     DK             Worsted
## 5         11                  Sport                  DK
```

Here I can see that I often opted for bulky or super bulky yarn rather than worsted which results in a pretty dramatic size different. Using worsted rather than DK or DK rather than sport also will affect the size, but not as dramatically.

6. What is the longest I have spent on a project for each weight?

This would be interesting to see because in general, using a thinner yarn takes much longer to make something, but thicker yarn is often used for something like a blanket which would also take a while.

```
sql_query_6 <- "
  WITH CompletedProjects AS (
    SELECT
      Projects.project_id,
      pattern_name,
      weight,
      start_date,
      end_date,
      julianday(end_date) - julianday(start_date) AS duration_days
    FROM Projects
    INNER JOIN Patterns ON Projects.pattern_id = Patterns.pattern_id
    INNER JOIN Project_Yarn ON Projects.project_id = Project_Yarn.project_id
    INNER JOIN Yarn ON Project_Yarn.yarn_id = Yarn.yarn_id
    WHERE Projects.end_date IS NOT NULL
  ),
  RankedProjects AS (
    SELECT *,
           RANK() OVER (PARTITION BY weight ORDER BY duration_days DESC) AS rank
    FROM CompletedProjects
  )
  SELECT
    project_id,
    pattern_name,
    weight,
    duration_days,
    start_date,
    end_date
  FROM RankedProjects
  WHERE rank = 1
  ORDER BY duration_days DESC;
"
dbGetQuery(con, sql_query_6)
```

```
##   project_id                            pattern_name      weight duration_days
## 1          1                      French Market Bag       Sport             9
## 2          7                    Sholach Mosaic Afghan        Aran             7
## 3         11 Beginner Crochet Granny Triangle Bandana          DK             2
## 4          5                    Crochet Seafarers Cap Super Bulky             2
##   start_date   end_date
## 1 2024-02-08 2024-02-17
## 2 2024-04-08 2024-04-15
## 3 2024-05-03 2024-05-05
## 4 2024-02-13 2024-02-15
```

The results show that my longest project was a bag with sport yarn and my fastest projects were a bandana, which is a pretty small project, and a hat with thick yarn which would work up very fast.

7. For each yarn weight, what's the most commonly recommended hook size across all patterns?

Yarn weight and hook size are related to each other because generally, you want to use a hook size that fits the yarn well. However, some patterns may call for a drastic change if they want results with smaller gaps or larger gaps.

```
sql_query_7 <- "
  WITH HookCounts AS (
    SELECT
      Yarn.weight,
      Patterns.recommended_hook_size,
      COUNT(*) AS count,
      RANK() OVER (
        PARTITION BY Yarn.weight
        ORDER BY COUNT(*) DESC
      ) AS hook_rank
    FROM Patterns
    INNER JOIN Pattern_Yarn
    ON Patterns.pattern_id = Pattern_Yarn.pattern_id
    INNER JOIN Yarn
    ON Pattern_Yarn.yarn_id = Yarn.yarn_id
    GROUP BY Yarn.weight, Patterns.recommended_hook_size
  )
  SELECT
    weight,
    recommended_hook_size AS most_common_hook_size,
    count AS times_used
  FROM HookCounts
  WHERE hook_rank = 1
  ORDER BY most_common_hook_size;
"
dbGetQuery(con, sql_query_7)
```

```
##        weight most_common_hook_size times_used
## 1      Thread               1.75 mm          1
## 2       Sport                3.5 mm          2
## 3        Aran                4.0 mm          4
## 4          DK                4.0 mm          6
## 5 Super Bulky                4.5 mm          3
## 6     Worsted                5.0 mm          9
## 7       Bulky                8.0 mm          2
## 8 Super Bulky                9.0 mm          3
```

Looking at these results, they almost all show an increase in hook size as the weight goes up. The exception is super bulky which had a time for a 4.5mm hook and a 9mm hook. This actually also makes sense because 9mm would work for most project with super bulky yarn, but when making a stuffed animal, many people opt to use super bulky yarn with a smaller hook to work up fast and with minimal gaps between stitches.

8. Which pattern that I used in a project is the most expensive?

While there are many free patterns online, some patterns have a price. I would like to see how much the most expensive pattern I have actually bought costed.

```
sql_query_8 <- "
  SELECT
```

```
    Patterns.pattern_name,
    Patterns.price
  FROM Projects
  INNER JOIN Patterns
  ON Projects.pattern_id = Patterns.pattern_id
  WHERE Patterns.price IS NOT NULL
  ORDER BY Patterns.price DESC
  LIMIT 1;
"
dbGetQuery(con, sql_query_8)
```

```
##            pattern_name price
## 1 Sholach Mosaic Afghan 10.65
```

```
dbDisconnect(con)
```

The most expensive pattern was the Sholach Mosaic Afghan at $10.65.

Conclusion

This database solves a lot of the problems I encounter when looking for patterns and working on projects. Using these queries, I am able to gather information that will help me either find more patterns or projects, or learn information about my own habits to help me make decisions. There are many, many more questions I can think of that can be answered using this database. There are a number of columns I didn't use in any of these queries but still included as I can see myself coming back to this database and those being useful.