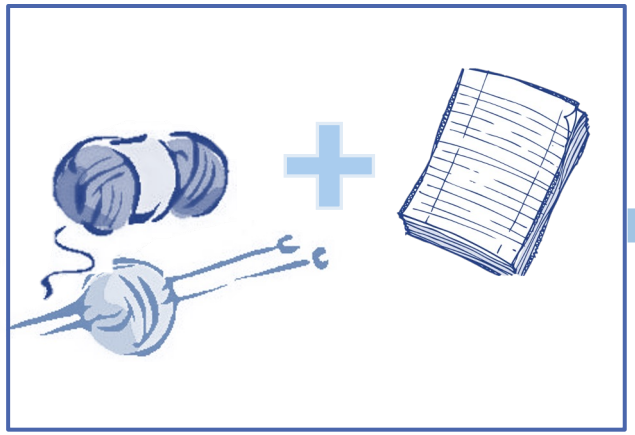
Yarn Project Schedule

WENYING WU

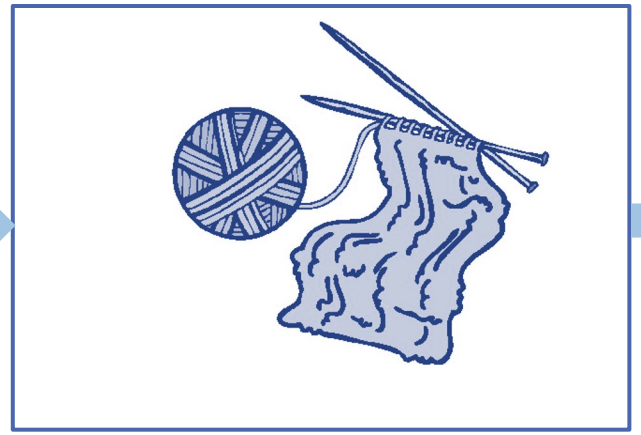
Real World Domain

https://www.woolandthegang.com/en

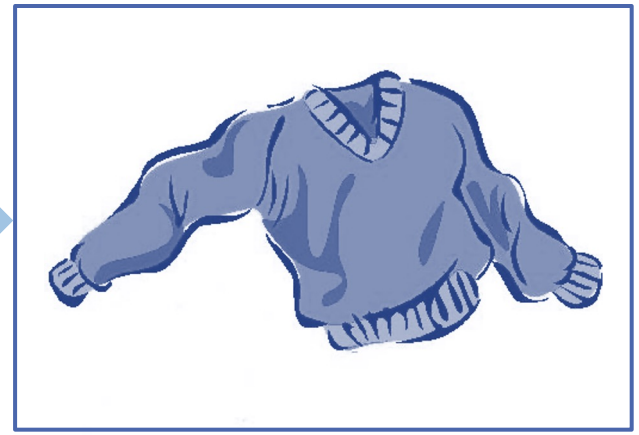
Yarn + Project



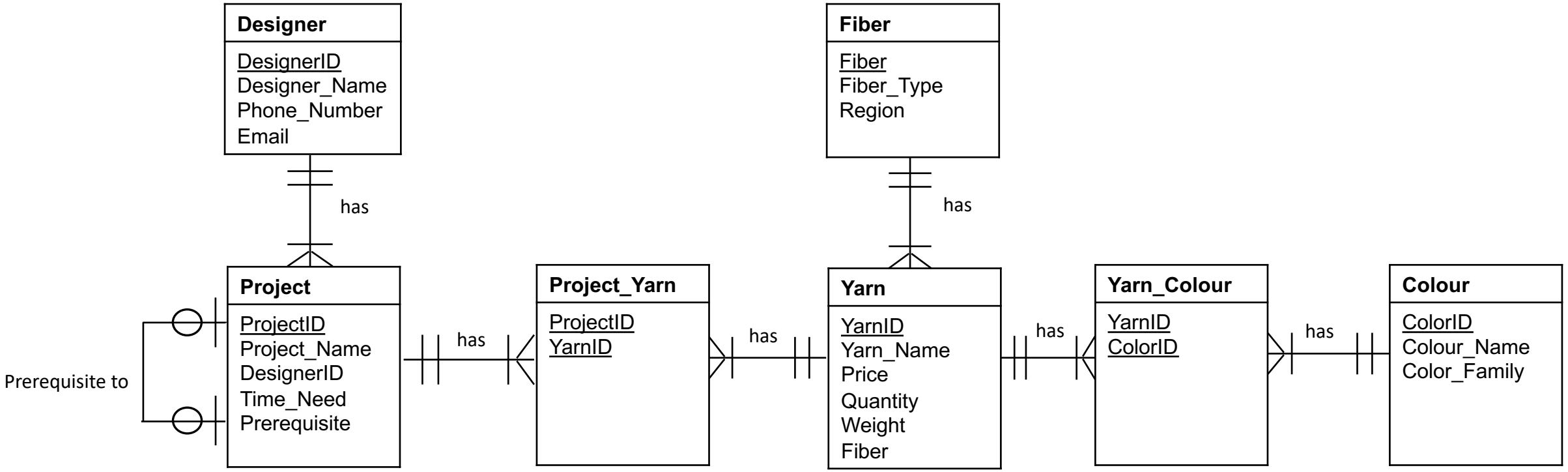
Knitting



Final Product



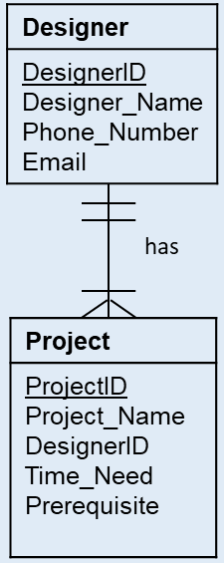
Completed ERD



One-to-many Relationship

There are two one-to-many relationship in this ERD

Example:



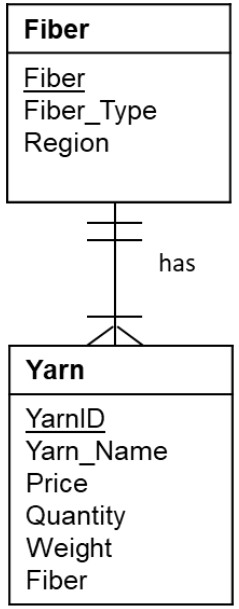
- One and only one Designer has one or many Project

```
postgres=# select * from Designer order by DesignerID;
 designerid | designer_name | phone_number |        email
------------+---------------+--------------+---------------------
        370 | Bruce         | 6128849754   | Bruce3785@gmail.com
        374 | John          | 6104337060   | John77@gmail.com
        385 | Jenny         | 6147998327   | Jenny0776@gmail.com
        399 | Ann           | 6104278859   | Ann.knit@hotmail.com
(4 rows)

postgres=# Select * from Project order by DesignerID;
 projectid |  project_name  | designerid | time_need | prerequisite
-----------+----------------+------------+-----------+--------------
         1 | Loop weave     |        370 |         5 |            0
         5 | Chullo hats    |        370 |        29 |            5
         3 | Cute sock      |        374 |         9 |            2
         4 | Blanket        |        385 |        15 |            3
         6 | Taylor Sweater |        385 |        35 |            6
         2 | Shopping Bag   |        399 |         6 |            1
(6 rows)
```

Second one- to-many relationship:

- One and only one Fiber used in one or many Yarn

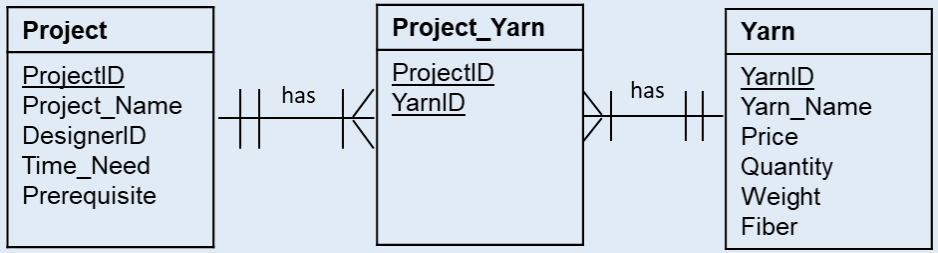


Many-to-many Relationship

There are two many-to-many relationship in this ERD

Example:

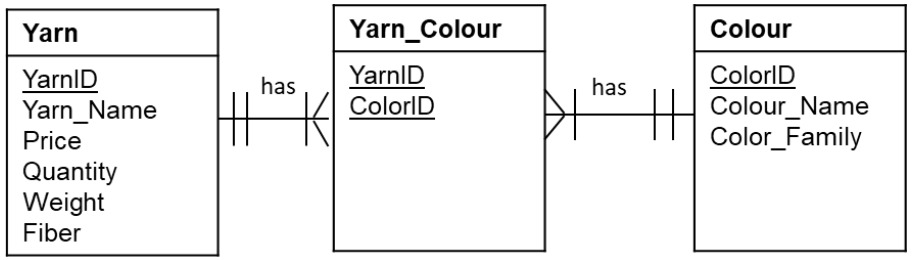
- One or many Project have one or many Yarn




```
 projectid | yarnid |  project_name   | designerid | time_need | prerequisite
-----------+--------+-----------------+------------+-----------+--------------
         1 |   1070 | Loop weave      |        370 |         5 |            0
         2 |   1171 | Shopping Bag    |        399 |         6 |            1
         3 |   1070 | Cute sock       |        374 |         9 |            2
         3 |   1171 | Cute sock       |        374 |         9 |            2
         4 |   1171 | Blanket         |        385 |        15 |            3
         4 |   1473 | Blanket         |        385 |        15 |            3
         5 |   1271 | Chullo hats     |        370 |        29 |            5
         5 |   1473 | Chullo hats     |        370 |        29 |            5
         6 |   1070 | Taylor Sweater  |        385 |        35 |            6
         6 |   1271 | Taylor Sweater  |        385 |        35 |            6
         6 |   1473 | Taylor Sweater  |        385 |        35 |            6
(11 rows)
```

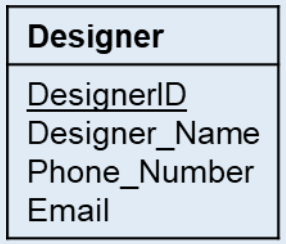
Second many- to many relationship:

- One or many Yarn have one or many Colour



Simple Query Example

Example:



- All the data in Designer table

```
postgres=# select * from Designer order by DesignerID;
 designerid | designer_name | phone_number |        email
------------+---------------+--------------+----------------------
        370 | Bruce         | 6128849754   | Bruce3785@gmail.com
        374 | John          | 6104337060   | John77@gmail.com
        385 | Jenny         | 6147998327   | Jenny0776@gmail.com
        399 | Ann           | 6104278859   | Ann.knit@hotmail.com
(4 rows)
```

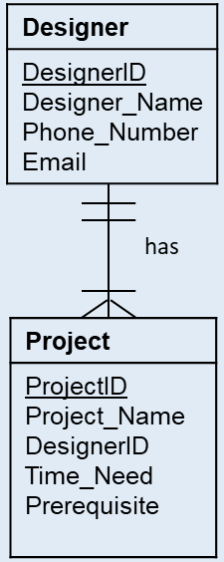
- Show the designer information about a specific designer, Bruce

```
postgres=# select * from Designer where Designer_Name = 'Bruce';
 designerid | designer_name | phone_number |        email
------------+---------------+--------------+--------------------
        370 | Bruce         | 6128849754   | Bruce3785@gmail.com
(1 row)
```

Join Example

Show the information about designer's project, including designer, project name, project ID and time need.

Example:



Natural Join

```
postgres=# select Designer_name as Designer, project_name as Project, projectID,
time_need from Designer natural join Project order by Designer;
  designer  |    project     | projectid | time_need
------------+----------------+-----------+-----------
 Ann        | Shopping Bag   |         2 |         6
 Bruce      | Loop weave     |         1 |         5
 Bruce      | Chullo hats    |         5 |        29
 Jenny      | Blanket        |         4 |        15
 Jenny      | Taylor Sweater |         6 |        35
 John       | Cute sock      |         3 |         9
(6 rows)
```

Cross Product

```
postgres=# select Designer_name as Designer, Project_Name as Project, ProjectID,
time_need from Designer, Project where Designer.DesignerID = Project.DesignerID
order by Designer;
  designer  |    project     | projectid | time_need
------------+----------------+-----------+-----------
 Ann        | Shopping Bag   |         2 |         6
 Bruce      | Loop weave     |         1 |         5
 Bruce      | Chullo hats    |         5 |        29
 Jenny      | Blanket        |         4 |        15
 Jenny      | Taylor Sweater |         6 |        35
 John       | Cute sock      |         3 |         9
(6 rows)
```

Group by & Sub query

- Previous example table

```
 designer  |    project     | projectid | time_need
-----------+----------------+-----------+-----------
 Ann       | Shopping Bag   |         2 |         6
 Bruce     | Loop weave     |         1 |         5
 Bruce     | Chullo hats    |         5 |        29
 Jenny     | Blanket        |         4 |        15
 Jenny     | Taylor Sweater |         6 |        35
 John      | Cute sock      |         3 |         9
(6 rows)
```

Group By (with HAVING)

- Show the number of project for designers who has more than one project.

```
postgres=# select Designer_name as Designer, count(projectID) as Number from Designer natural
join Project group by Designer HAVING count(*) >1 order by Designer;
  designer  | number
------------+--------
 Bruce      |      2
 Jenny      |      2
(2 rows)
```

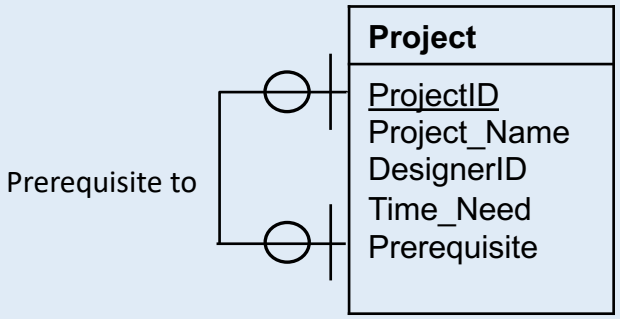
Sub query

- Show the project which used more than average time of all project, including designer, project name, project ID and time need.

```
postgres=# select Designer_name as Designer, Project_name as Project, ProjectID, time_need from
Designer natural join Project where time_need > (select AVG(time_need) from Designer natural
join Project);
  designer  |    project     | projectid | time_need
------------+----------------+-----------+-----------
 Bruce      | Chullo hats    |         5 |        29
 Jenny      | Taylor Sweater |         6 |        35
(2 rows)
```

Self Join

Example table:



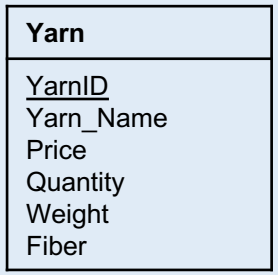
- Show all the projects which have a prerequisite, including project and project ID

```
postgres=# select ass1.ProjectID as ProjectID, ass1.Project_Name as Project,
ass2.ProjectID as required_ProjectID, ass2.Project_Name as required_ProjectID from
project ass1, Project ass2 where ass1. Prerequisite = ass2.ProjectID;
 projectid |    project     | required_projectid | required_projectid
-----------+----------------+--------------------+--------------------
         2 | Shopping Bag   |                  1 | Loop weave
         3 | Cute sock      |                  2 | Shopping Bag
         4 | Blanket        |                  3 | Cute sock
         5 | Chullo hats    |                  4 | Blanket
         6 | Taylor Sweater |                  5 | Chullo hats
(5 rows)
```

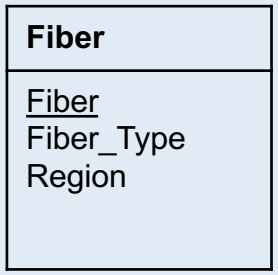
Statement

CHECK statement example:

Example 1:



Example 2:

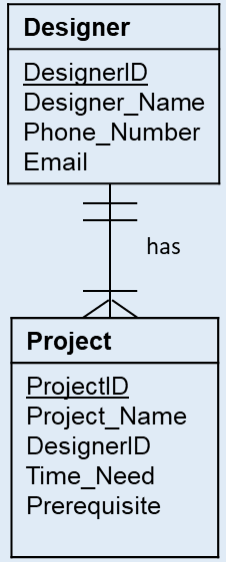



```
CONSTRAINT Yarn_YarnID CHECK ((YarnID >= 1000) AND (YarnID <= 1500)),
CONSTRAINT Yarn_Price CHECK ((Price > 0) AND (Price <= 30)),
CONSTRAINT Yarn_Quantity CHECK ((Quantity >= 50) AND (Quantity <= 200))
```

```
CONSTRAINT Fiber_Fiber CHECK (Fiber IN('Alpaca', 'Raffia', 'Wool',
'Acrylic')),
CONSTRAINT Fiber_Fiber_Type CHECK (Fiber_Type IN('Natural',
'Artificial')),
CONSTRAINT Fiber_Region CHECK (Region IN('New Zealand', 'Japan',
'Australia', 'China'))
```

Action statement example:

Example:

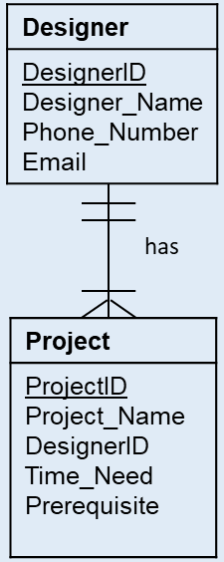



```
CONSTRAINT ProjectFK FOREIGN KEY (DesignerID) REFERENCES Designer
      ON DELETE RESTRICT
      ON UPDATE CASCADE,
```

View

- List designer and their project, including Designer ID, Designer's name and the name of the project

Example:



Code

```
CREATE VIEW Designer_Project AS
    select DesignerID, Designer_Name as Designer, Project_Name as Project
    from  Designer natural join Project
    order by DesignerID;
```

Result

```
postgres=# select * from Designer_Project;
 designerid |  designer  |    project
------------+------------+---------------
        370 | Bruce      | Loop weave
        370 | Bruce      | Chullo hats
        374 | John       | Cute sock
        385 | Jenny      | Blanket
        385 | Jenny      | Taylor Sweater
        399 | Ann        | Shopping Bag
(6 rows)
```