

Note - avoid N+1 query problems 1. Using the shell plus and the django project you created before, run these queries to:

Fetch all the objects of authors with only id, first name and last name

```
>>> from book.models import Author, Book
>>>
>>> author = list(Author.objects.values('id', 'first_name', 'last_name'))
>>> print(author)
[{'id': 0, 'first_name': 'Albin', 'last_name': 'AK'}, {'id': 1, 'first_name': 'Rahul', 'last_name': ''}, {'id': 2, 'first_name': 'Arun', 'last_name': 'Kumar'}, {'id': 3, 'first_name': 'Akshita', 'last_name': 'Jhon'}, {'id': 4, 'first_name': 'Madavi', 'last_name': 'Kuttu'}]
```

Fetch all the objects of books as a list that contains name

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE COMMENTS python + v [ ] [ ] [ ] [ ] [ ] [ ]
>>>
>>> from book.models import Book
>>>
>>> books = list(Book.objects.values_list('name', flat=True))
>>> print(books)
['Book 4', 'Book 6', 'Book 7', 'Book 8', 'Book 9']
>>>
```

Fetch all the objects of books as a tuple that contains name and count

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE COMMENTS python + v [ ] [ ] [ ] [ ] [ ] [ ]
>>>
>>> books = list(Book.objects.values_list('name', 'count'))
>>> print(books)
[('Book 4', 60), ('Book 6', 40), ('Book 7', 55), ('Book 8', 20), ('Book 9', 15)]
>>>
```

Fetch an object from books table using id and update the count.

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE COMMENTS python + v [ ] [ ] [ ] [ ] [ ] [ ]
>>> from book.models import Book
>>>
>>> book_id = 8
>>> book = Book.objects.get(id=book_id)
>>> print(f'Previous count of book {book.name} was {book.count}')
Previous count of book Book 9 was 25
>>> book.count = 30
>>> book.save()
>>> print(f'Count of book {book.name} has been updated to {book.count}')
Count of book Book 9 has been updated to 30
```

Fetch an object from the books using name and update the author of that book



Fetch an author from the table and increase the price with 100 for all the books of that author

```
... for book in author.book_set.all():
...     print(f'Current book price {book.name} is {book.price}')
...     book.price += 100
...     book.save()
...     print(f'Updated book price {book.name} is {book.price}')
...
... print(f"The price of all books by {full_name} has been increased by 100.")
...
... except Author.DoesNotExist:
...     print("Error: Author not found.")
... except Exception as e:
...     print(f"An error occurred: {e}")
...
Current book price Book 4 is 650.00
Updated book price Book 4 is 750.00
Current book price Book 7 is 800.00
Updated book price Book 7 is 900.00
Current book price Book 8 is 850.00
Updated book price Book 8 is 950.00
The price of all books by Akhila Jhon has been increased by 100.
```

Fetch all the authors from the table and show the name of authors with the list of names of the books of that author that have an average rating greater than 3 in a list of dictionaries format

```
>>> authors = Author.objects.all()
>>> authors_with_books = []
>>>
>>> for author in authors:
...     books_with_high_rating = author.book_set.filter(average_rating__gt=3)
...     books_info = [{'name': book.name, 'average_rating': book.average_rating} for book in books_with_high_rating]
...     author_dict = {
...         'author_name': f"{author.first_name} {author.last_name}",
...         'books_with_high_rating': books_info
...     }
...     if books_info:
...         authors_with_books.append(author_dict)
>>> print(authors_with_books)
[{'author_name': 'Akhila Jhon', 'books_with_high_rating': [{'name': 'Book 4', 'average_rating': 4.7}, {'name': 'Book 7', 'average_rating': 4.8}]}, {'author_name': 'Madavi Kutty', 'books_with_high_rating': [{'name': 'Book 9', 'average_rating': 4.6}]}]
```

Fetch all the authors from the table and show the name of authors with the list of names of the books of that author that have an average rating less than or equal to 3 in a list of dictionaries format

```
>>> authors = Author.objects.all()
>>> authors_with_books = []
>>>
>>> for author in authors:
...     books_with_high_rating = author.book_set.filter(average_rating__lte=3)
...     books_info = [{'name': book.name, 'average_rating': book.average_rating} for book in books_with_high_rating]
...     author_dict = {
...         'author_name': f"{author.first_name} {author.last_name}",
...         'books_with_high_rating': books_info
...     }
...     if books_info:
...         authors_with_books.append(author_dict)
>>> print(authors_with_books)
[{'author_name': 'Rahul ', 'books_with_high_rating': [{'name': 'Book 6', 'average_rating': 3.0}]}, {'author_name': 'Akhila Jhon', 'books_with_high_rating': [{'name': 'Book 8', 'average_rating': 2.0}]}]
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE COMMENTS python + - [ ] ... ^ x  
  
>>> from book.models import Book  
>>>  
>>> books_with_average_rating = Book.objects.filter(average_rating=3)  
>>> books_list = list(books_with_average_rating)  
>>> print(books_list)  
[<Book: Book 6>]
```

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE COMMENTS
python + - [ ] ... ^ x

>>> from book.models import Author
>>>
>>> authors_with_few_books = Author.objects.filter(books_count__lt=10)
>>> full_names = []
>>>
>>> for author in authors_with_few_books:
...     full_name = f"{author.first_name} {author.last_name}"
...     full_names.append(full_name)
...
>>> print(full_names)
['Akhila Jhon', 'Madavi Kutty']

```

A screenshot of a Jupyter Notebook interface. The top bar shows tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (which is active), PORTS, SQL CONSOLE, and COMMENTS. On the right side of the top bar, there are icons for running code (a play button), saving (a floppy disk), and other utility icons. Below the top bar, the left sidebar contains navigation icons for Explorer, Source, Run and Debug, and Test Explorer. The main area displays a Python script being executed in the terminal. The script imports Author from book.models, gets all authors, and iterates over them to update their books\_count based on the count of books in their set. The output shows the updated counts for Albin, Rahul, Arun, Akhila, and Madavi.

```
>>> from book.models import Author
>>>
>>> authors = Author.objects.all()
>>> for author in authors:
...     books_count = author.book_set.count()
...     author.books_count = books_count
...     author.save()
...     print(f'Updated books_count for {author}: {books_count}')
```

Updated books\_count for Albin: 0  
Updated books\_count for Rahul: 1  
Updated books\_count for Arun: 0  
Updated books\_count for Akhila: 3  
Updated books\_count for Madavi: 1

```

>>> from book.models import Book
>>>
>>> books_without_count = Book.objects.all().values('id', 'name', 'price', 'average_rating', 'author_id')
>>>
>>> books_list = list(books_without_count)
>>> print(books_list)
[{'id': 3, 'name': 'Book 4', 'price': Decimal('750.00'), 'average_rating': 4.7, 'author_id': 3}, {'id': 5, 'name': 'Book 6', 'price': Decimal('650.00'), 'average_rating': 2.0, 'author_id': 1}, {'id': 6, 'name': 'Book 7', 'price': Decimal('900.00'), 'average_rating': 4.8, 'author_id': 3}, {'id': 7, 'name': 'Book 8', 'price': Decimal('950.00'), 'average_rating': 2.0, 'author_id': 3}, {'id': 8, 'name': 'Book 9', 'price': Decimal('500.00'), 'average_rating': 4.6, 'author_id': 4}]
>>>

```

```

>>> from book.models import Author
>>> authors_queryset = Author.objects.all().values('first_name', 'last_name')
>>> authors_list = list(authors_queryset)
>>> print(authors_list)
[{'first_name': 'Albin', 'last_name': 'AK'}, {'first_name': 'Rahul', 'last_name': ''}, {'first_name': 'Arun', 'last_name': 'Kumar'}, {'first_name': 'Akhila', 'last_name': 'Jhon'}, {'first_name': 'Madavi', 'last_name': 'Kutty'}]
>>>
>>>

```