N+1 Problem in Django Rest Framework

PyCon AU 2023 | Django Track

Agenda

- Speaker Introduction
- Quick brief of Django Rest Framework
- What is N+1 problem
- How N+1 can occur in DRF Views
- How DRF Serializers can contain N+1 problem
- Identifying N+1 problems in APIs
- Mitigating N+1 problems
- Closing Thoughts

whoami

- Software Engineer Contractor @ Arbisoft, Lahore, Pakistan
- Python & Django
- Working with “edX/2U” as Arbisoft contractor
- Internal products development within Arbisoft

Syed M. Dawoud Sheraz Ali

DawoudSheraz

Medium @dawoud.sheraz

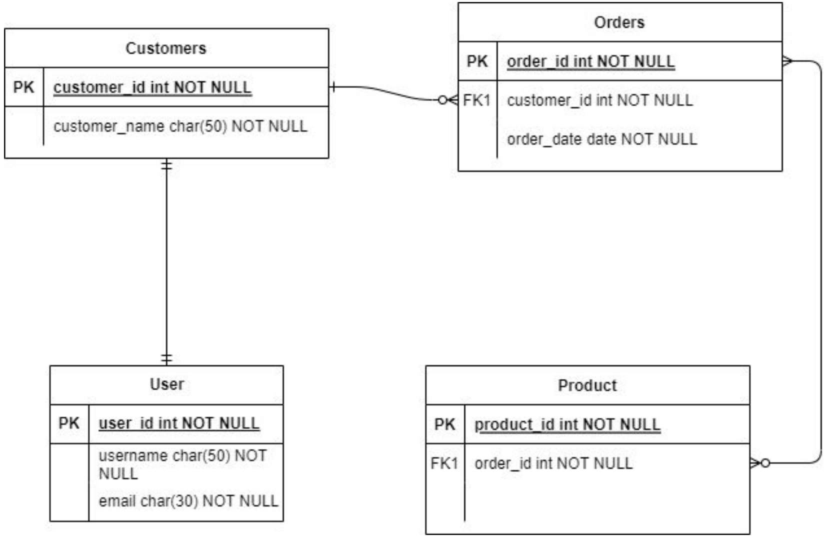
LinkedIn syed-muhammad-dawoud-sheraz-ali

Django Rest Framework (DRF)

- A powerful, flexible API framework to create RESTful APIs in Django
- Integrates with core Django features
  - Model
  - Views
  - URLs
- Serialization & Deserialization
  - Convert model/querysets into JSON & other formats
  - Convert incoming JSON and other data formats into respective model objects
- Views
  - Viewset, generics, API Views
  - Utilize serializers
- Authentication & Permissions

N+1 problem & Relational Databases

- The data models are scattered across in different tables
- Tables are connected/related to each other via relations like 1-1, 1-Many or Many-Many
- Joining tables to get complete information of an entity



Customers

PK	customer_id int NOT NULL
	customer_name char(50) NOT NULL

Orders

PK	order_id int NOT NULL
FK1	customer_id int NOT NULL
	order_date date NOT NULL

User

PK	user_id int NOT NULL
	username char(50) NOT NULL
	email char(30) NOT NULL

Product

PK	product_id int NOT NULL
FK1	order_id int NOT NULL

N+1 problem

- From previous example, the 2 ways to get complete customer order information using DB queries
- First way
    - Get required customer info from customer table
    - From customer_id, get all of orders made by that customers
    - For each order, get all of the products present in the order (M2M relationship, intermediate model is queried first)
- Second way
    - Join tables using Foreign keys
    - Get all the required data in a single query

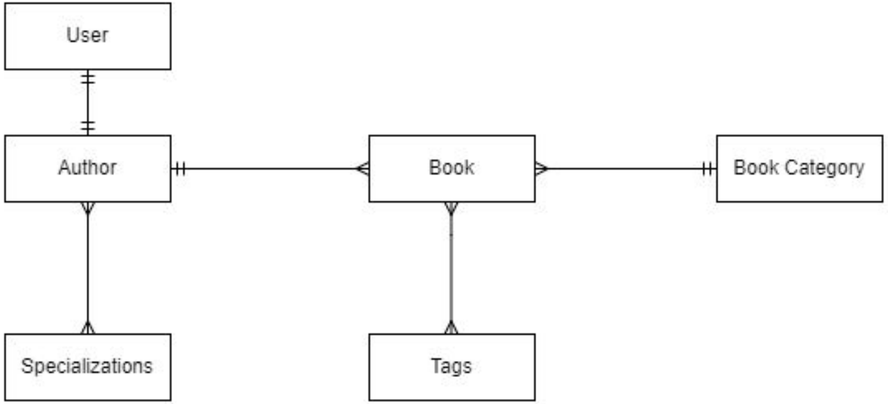
N+1 problem

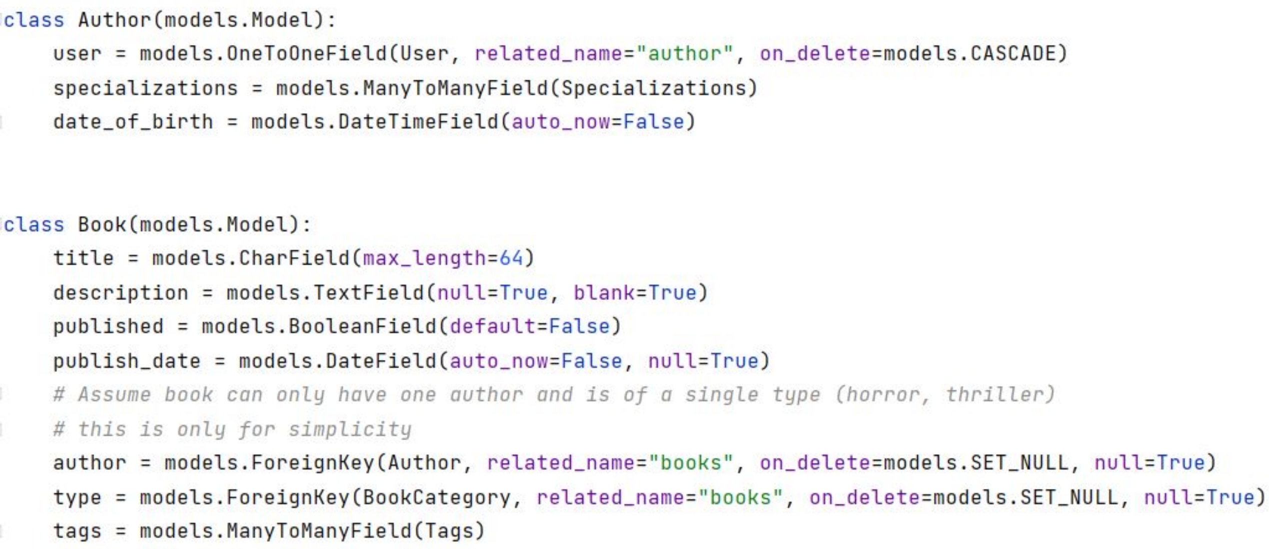
- First Method
  - Make 1 query to get the main object
  - For each related object (1-1 or FK), make a separate query
  - 1 + N queries or N+1 queries
- Second Method
  - 1 join query to get main object with related 1-1 & FK objects
- Both methods get the job done, so why is N+1 an issue?

Why is N+1 a problem?

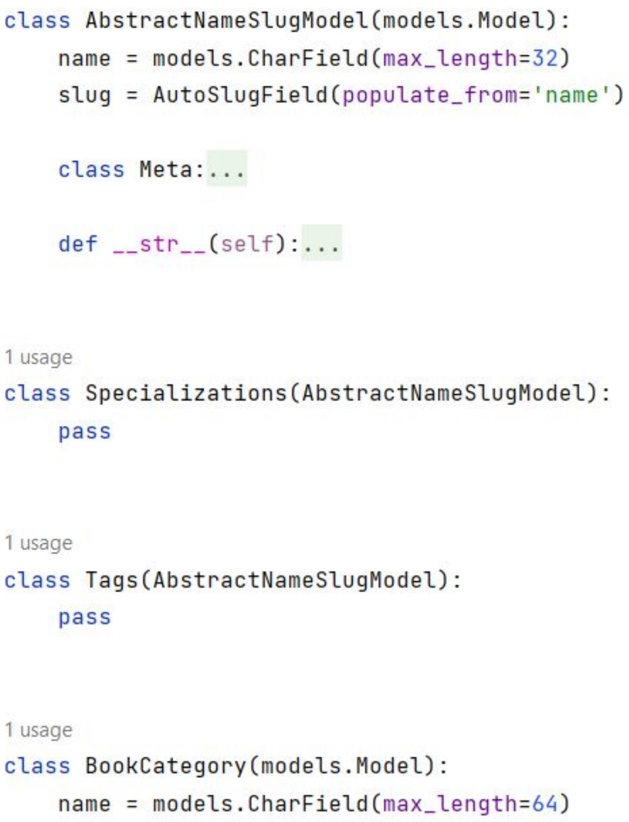
- ORM or Object Relational Mapping
- ORM need to make a DB round trip for every query
    - Individual queries might be fast
    - The accumulated time for all queries will be problematic
- The more queries are there in a ORM object fetch, the more round trips to database will be made by ORM
    - Holds true for Django, the ORM powering DRF

Example under consideration





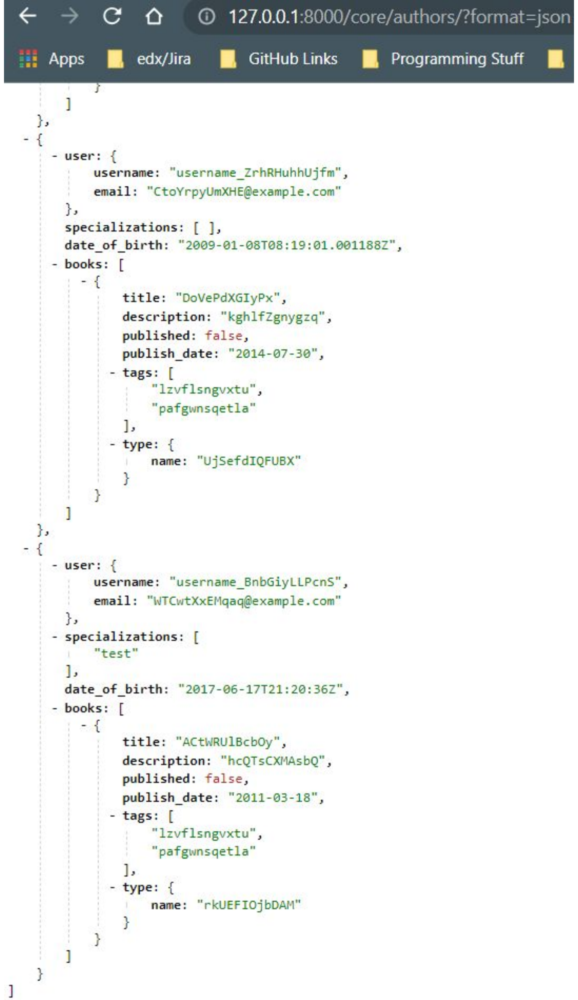
```
class Author(models.Model):
    user = models.OneToOneField(User, related_name="author", on_delete=models.CASCADE)
    specializations = models.ManyToManyField(Specializations)
    date_of_birth = models.DateTimeField(auto_now=False)


class Book(models.Model):
    title = models.CharField(max_length=64)
    description = models.TextField(null=True, blank=True)
    published = models.BooleanField(default=False)
    publish_date = models.DateField(auto_now=False, null=True)
    # Assume book can only have one author and is of a single type (horror, thriller)
    # this is only for simplicity
    author = models.ForeignKey(Author, related_name="books", on_delete=models.SET_NULL, null=True)
    type = models.ForeignKey(BookCategory, related_name="books", on_delete=models.SET_NULL, null=True)
    tags = models.ManyToManyField(Tags)
```



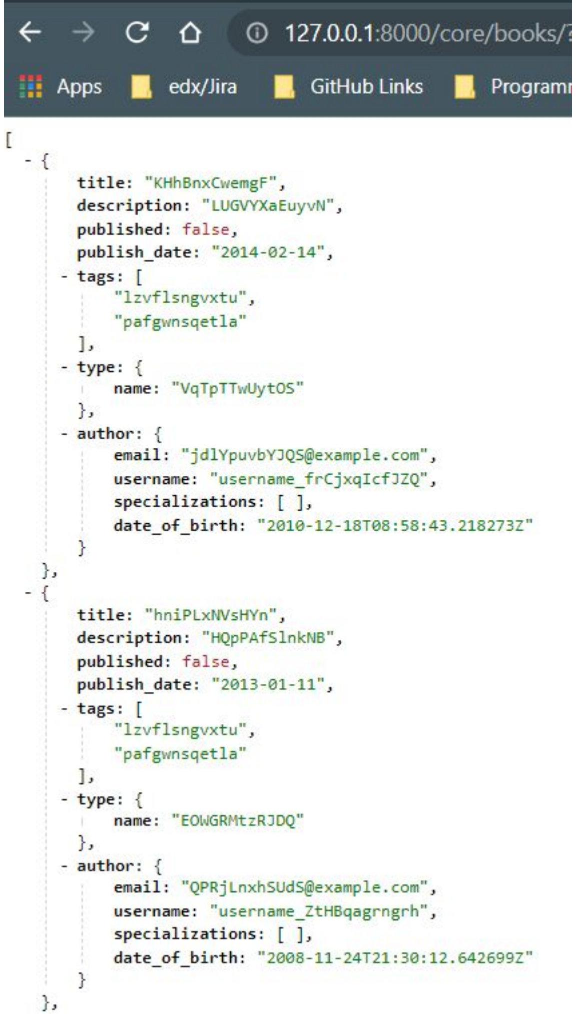
```
class AbstractNameSlugModel(models.Model):
    name = models.CharField(max_length=32)
    slug = AutoSlugField(populate_from='name')

    class Meta:...

    def __str__(self):...


1 usage
class Specializations(AbstractNameSlugModel):
    pass


1 usage
class Tags(AbstractNameSlugModel):
    pass


1 usage
class BookCategory(models.Model):
    name = models.CharField(max_length=64)
```

APIs under consideration

- List all authors with their written books
- List all books including the person who authored it



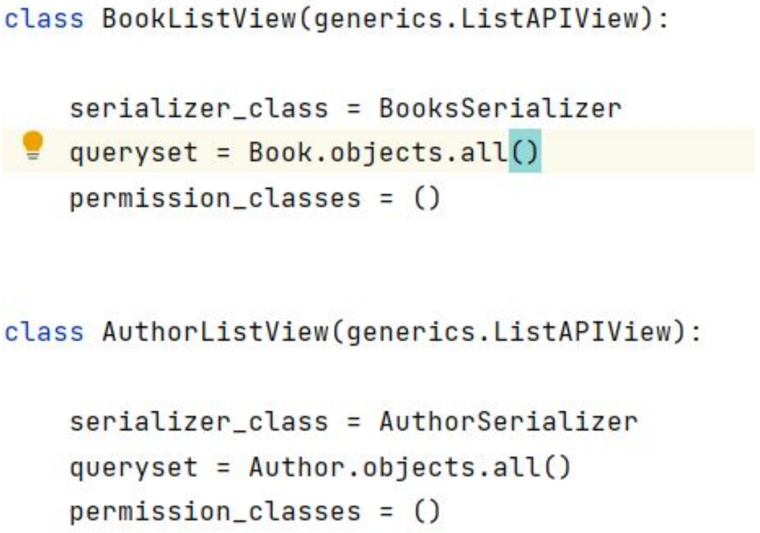
```
            }
        ]
    },
  - {
      - user: {
            username: "username_ZrhRHuhhUjfm",
            email: "CtoYrpyUmXHE@example.com"
        },
        specializations: [ ],
        date_of_birth: "2009-01-08T08:19:01.001188Z",
      - books: [
          - {
                title: "DoVePdXGIyPx",
                description: "kghlfZgnygzq",
                published: false,
                publish_date: "2014-07-30",
              - tags: [
                    "lzvflsngvxtu",
                    "pafgwnsqetla"
                ],
              - type: {
                    name: "UjSefdIQFUBX"
                }
            }
        ]
    },
  - {
      - user: {
            username: "username_BnbGiyLLPcnS",
            email: "WTCwtXxEMqaq@example.com"
        },
      - specializations: [
            "test"
        ],
        date_of_birth: "2017-06-17T21:20:36Z",
      - books: [
          - {
                title: "ACtWRUlBcbOy",
                description: "hcQTsCXMAsbQ",
                published: false,
                publish_date: "2011-03-18",
              - tags: [
                    "lzvflsngvxtu",
                    "pafgwnsqetla"
                ],
              - type: {
                    name: "rkUEFIOjbDAM"
                }
            }
        ]
    }
]
```



```
[
  - {
        title: "KHhBnxCwemgF",
        description: "LUGVYXaEuyvN",
        published: false,
        publish_date: "2014-02-14",
      - tags: [
            "lzvflsngvxtu",
            "pafgwnsqetla"
        ],
      - type: {
            name: "VqTpTTwUytOS"
        },
      - author: {
            email: "jdlYpuvbYJQS@example.com",
            username: "username_frCjxqIcfJZQ",
            specializations: [ ],
            date_of_birth: "2010-12-18T08:58:43.218273Z"
        }
    },
  - {
        title: "hniPLxNVsHYn",
        description: "HQpPAfSlnkNB",
        published: false,
        publish_date: "2013-01-11",
      - tags: [
            "lzvflsngvxtu",
            "pafgwnsqetla"
        ],
      - type: {
            name: "EOWGRMtzRJDQ"
        },
      - author: {
            email: "QPRjLnxhSUdS@example.com",
            username: "username_ZtHBqagrngrh",
            specializations: [ ],
            date_of_birth: "2008-11-24T21:30:12.642699Z"
        }
    },
```

N+1 in DRF Views

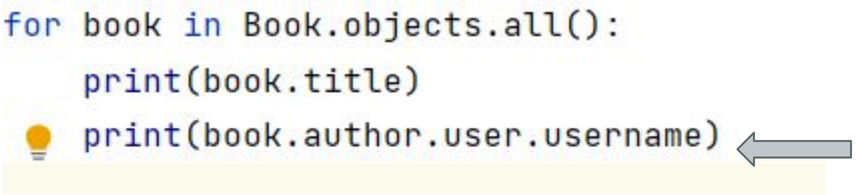
- In generic views and model viewsets, queryset is necessary to return objects
  - Either queryset attr or get_queryset
- Indicates which model will be the used to list or lookup specific object



```
class BookListView(generics.ListAPIView):

    serializer_class = BooksSerializer
    queryset = Book.objects.all()
    permission_classes = ()


class AuthorListView(generics.ListAPIView):

    serializer_class = AuthorSerializer
    queryset = Author.objects.all()
    permission_classes = ()
```

N+1 in DRF Views

- Querysets are Lazy – DB won’t be hit until the objects are needed
- Holds true for main object, related objects & related related objects
- Django’s Queryset behavior that seeps into DRF views

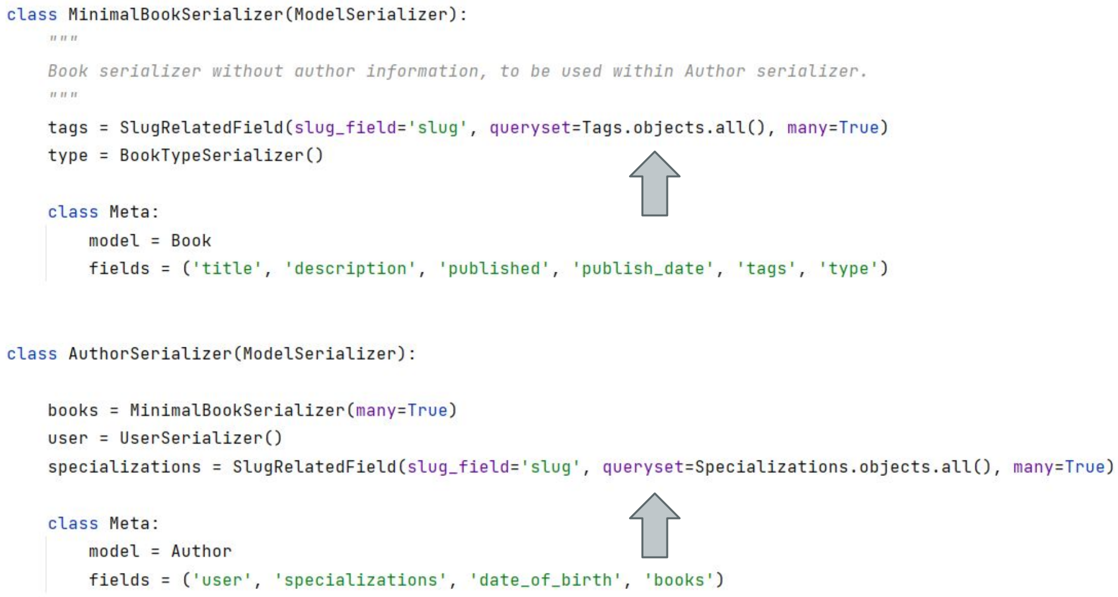


```
for book in Book.objects.all():
    print(book.title)
    print(book.author.user.username)
```

Hits database
for every book

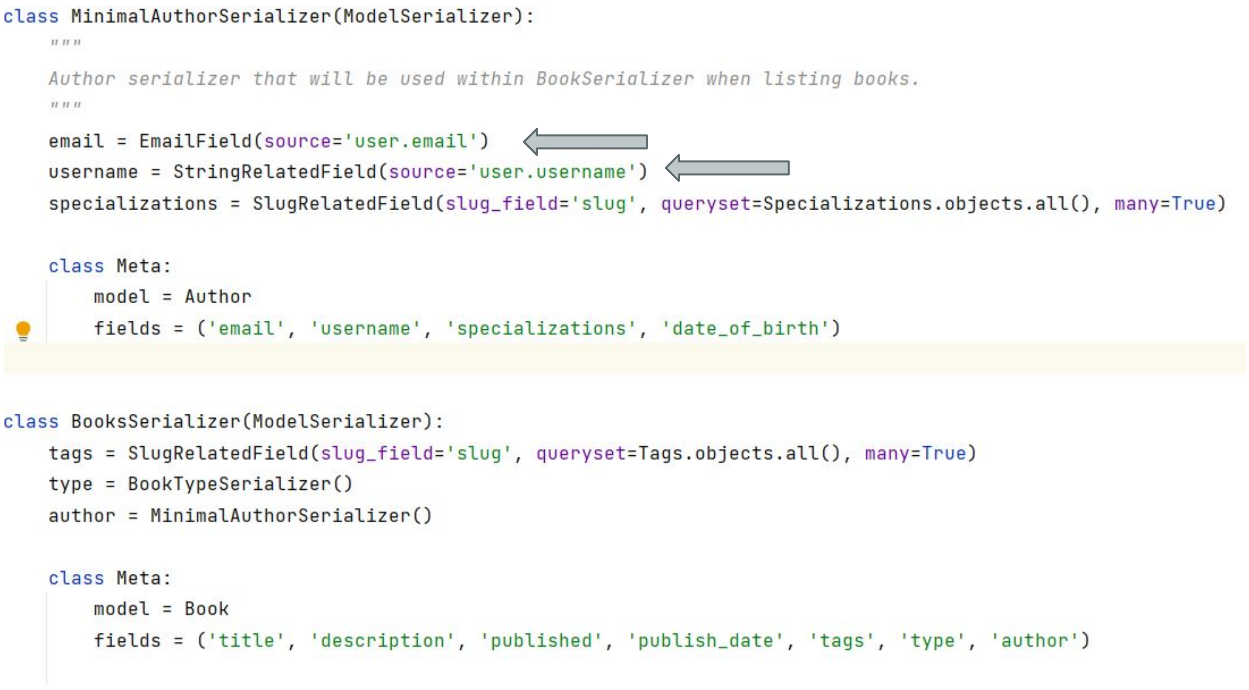
N+1 in Serializers

- Serializers define how the object and its related objects will be represented
- Complete control in providing full or part of object (both main & related)



```
class MinimalBookSerializer(ModelSerializer):
    """
    Book serializer without author information, to be used within Author serializer.
    """
    tags = SlugRelatedField(slug_field='slug', queryset=Tags.objects.all(), many=True)
    type = BookTypeSerializer()

    class Meta:
        model = Book
        fields = ('title', 'description', 'published', 'publish_date', 'tags', 'type')


class AuthorSerializer(ModelSerializer):

    books = MinimalBookSerializer(many=True)
    user = UserSerializer()
    specializations = SlugRelatedField(slug_field='slug', queryset=Specializations.objects.all(), many=True)

    class Meta:
        model = Author
        fields = ('user', 'specializations', 'date_of_birth', 'books')
```

N+1 in Serializers



```
class MinimalAuthorSerializer(ModelSerializer):
    """
    Author serializer that will be used within BookSerializer when listing books.
    """
    email = EmailField(source='user.email')
    username = StringRelatedField(source='user.username')
    specializations = SlugRelatedField(slug_field='slug', queryset=Specializations.objects.all(), many=True)

    class Meta:
        model = Author
        fields = ('email', 'username', 'specializations', 'date_of_birth')


class BooksSerializer(ModelSerializer):
    tags = SlugRelatedField(slug_field='slug', queryset=Tags.objects.all(), many=True)
    type = BookTypeSerializer()
    author = MinimalAuthorSerializer()

    class Meta:
        model = Book
        fields = ('title', 'description', 'published', 'publish_date', 'tags', 'type', 'author')
```

N+1 in Serializers

- Fields that take `source` or `queryset` attribute are prone to N+1 issues
- In MinimalAuthorSerializer EmailField, `user.email`will cause DRF to make DB call to get user object for author if not prefetched already
- Querysets, being lazy, do not fetch related objects unless accessed

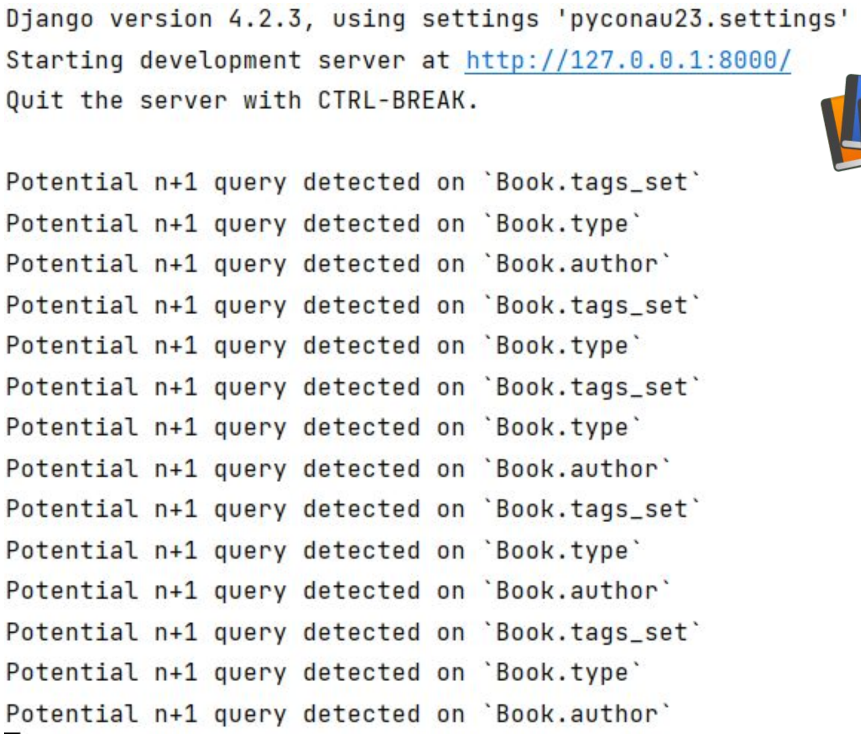
Locating N+1 problems

- nplusone
- django-debug-toolbar

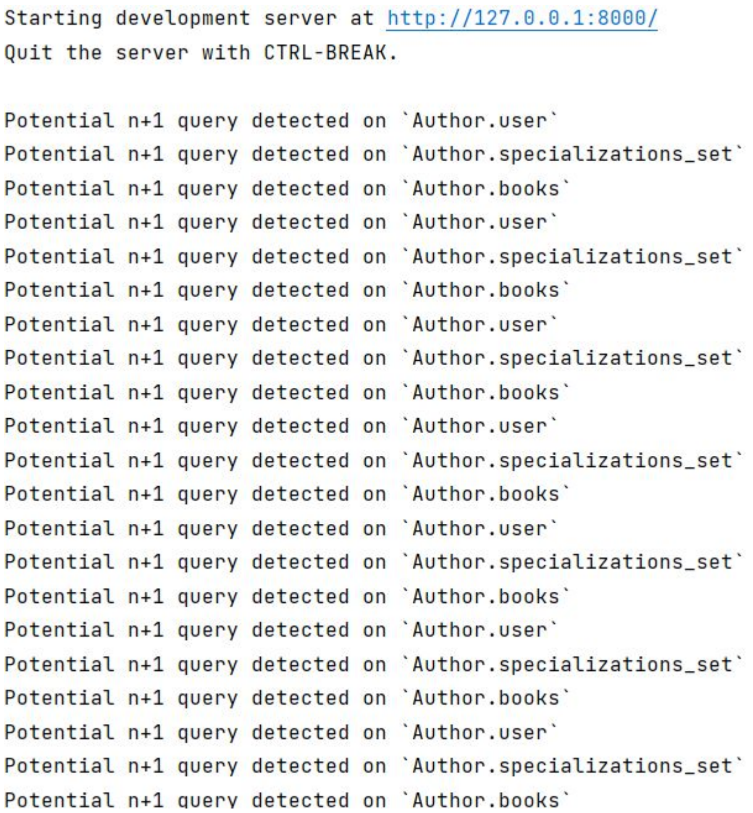
Locating N+1 problems: nplusone

- jmcarp/nplusone – package to auto-detect N+1 problems (Django, SQLAlchemy)
- Analyzes the application & logs the potential N+1 issues
  - Does not point out where the issue is originating from
- Not actively maintained anymore but still works

Locating N+1 problems: nplusone



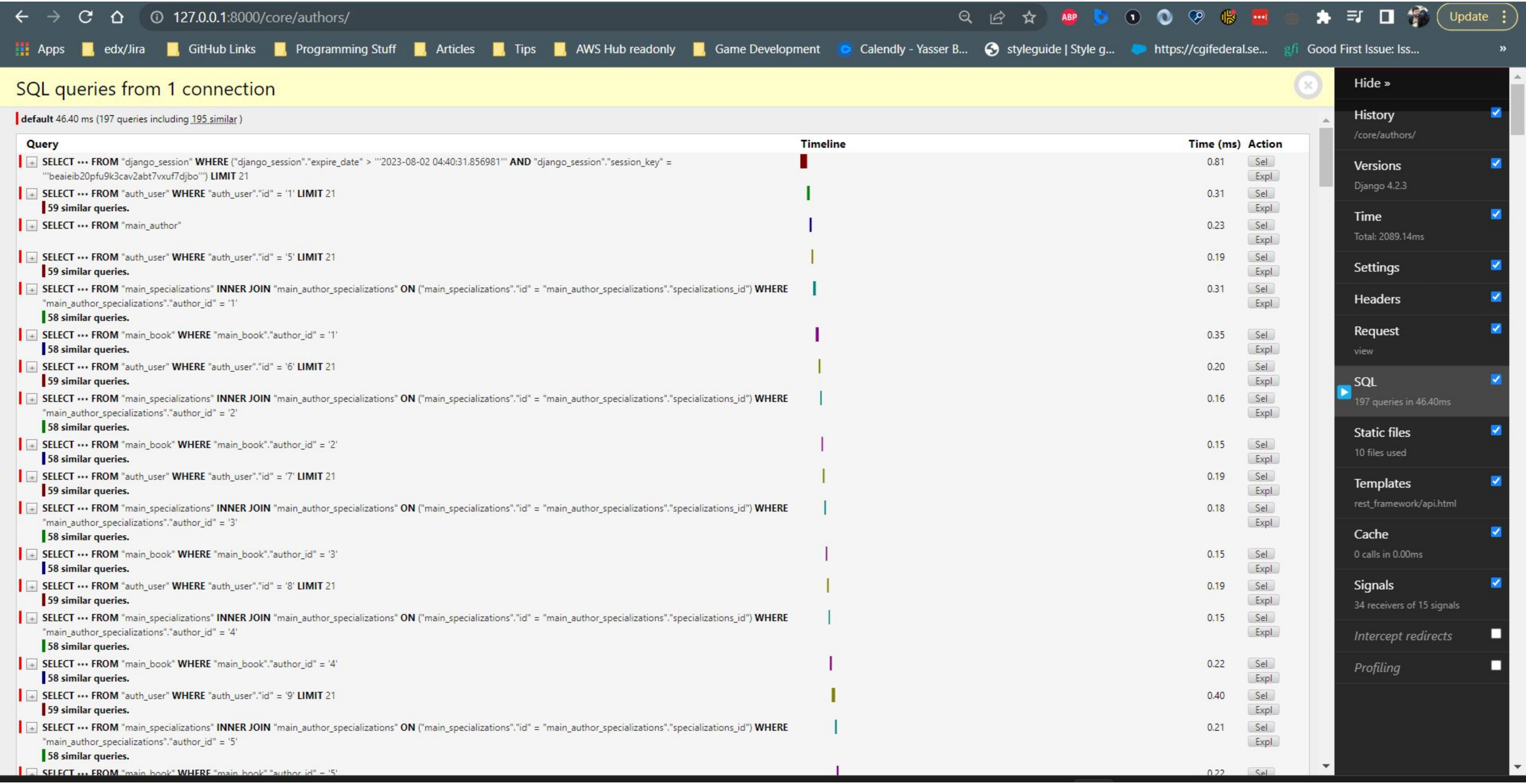
```
Django version 4.2.3, using settings 'pyconau23.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.type`
Potential n+1 query detected on `Book.author`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.type`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.type`
Potential n+1 query detected on `Book.author`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.type`
Potential n+1 query detected on `Book.author`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.type`
Potential n+1 query detected on `Book.author`
```



```
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

Potential n+1 query detected on `Author.user`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.user`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.user`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.user`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.user`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.user`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.user`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
```

Locating N+1 problems: django-debug-toolbar

- Django-debug-toolbar – a very important package in debugging Django apps
- Much more heavy weight than nplusone
- Outlines different amount of details on views such as
  - Django version
  - **SQL queries**
  - Signals
  - Templates
  - Request headers
  - … and much more
- How to Locate N+1 → Observe duplicate queries



SQL queries from 1 connection

default 46.40 ms (197 queries including 195 similar)

Query	Timeline	Time (ms)	Action
SELECT ••• FROM "django_session" WHERE ("django_session"."expire_date" > '''2023-08-02 04:40:31.856981''' AND "django_session"."session_key" = '''beaieib20pfu9k3cav2abt7vxuf7djbo''') LIMIT 21		0.81	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '1' LIMIT 21 59 similar queries.		0.31	Sel Expl
SELECT ••• FROM "main_author"		0.23	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '5' LIMIT 21 59 similar queries.		0.19	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '1' 58 similar queries.		0.31	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '1' 58 similar queries.		0.35	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '6' LIMIT 21 59 similar queries.		0.20	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '2' 58 similar queries.		0.16	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '2' 58 similar queries.		0.15	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '7' LIMIT 21 59 similar queries.		0.19	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '3' 58 similar queries.		0.18	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '3' 58 similar queries.		0.15	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '8' LIMIT 21 59 similar queries.		0.19	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '4' 58 similar queries.		0.15	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '4' 58 similar queries.		0.22	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '9' LIMIT 21 59 similar queries.		0.40	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '5' 58 similar queries.		0.21	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '5'		0.22	Sel

Hide »

☑ History
/core/authors/

☑ Versions
Django 4.2.3

☑ Time
Total: 2089.14ms

☑ Settings

☑ Headers

☑ Request
view

☑ SQL
197 queries in 46.40ms

☑ Static files
10 files used

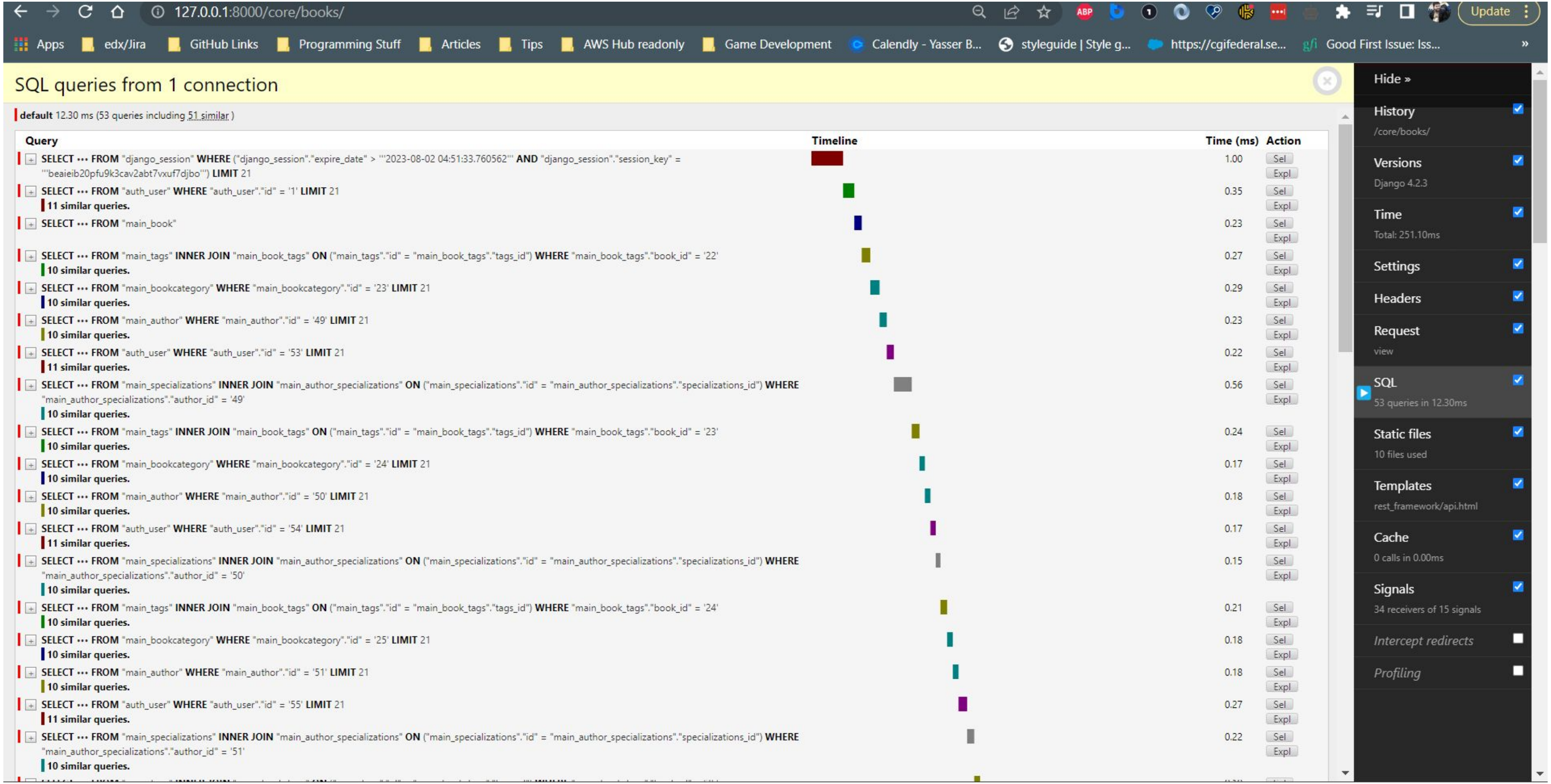
☑ Templates
rest_framework/api.html

☑ Cache
0 calls in 0.00ms

☑ Signals
34 receivers of 15 signals

□ *Intercept redirects*

□ *Profiling*



SQL queries from 1 connection

default 12.30 ms (53 queries including 51 similar)

Query	Timeline	Time (ms)	Action
SELECT ••• FROM "django_session" WHERE ("django_session"."expire_date" > '''2023-08-02 04:51:33.760562''' AND "django_session"."session_key" = '''beaieib20pfu9k3cav2abt7vxuf7djbo''') LIMIT 21		1.00	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '1' LIMIT 21 **11 similar queries.**		0.35	Sel Expl
SELECT ••• FROM "main_book"		0.23	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '22' **10 similar queries.**		0.27	Sel Expl
SELECT ••• FROM "main_bookcategory" WHERE "main_bookcategory"."id" = '23' LIMIT 21 **10 similar queries.**		0.29	Sel Expl
SELECT ••• FROM "main_author" WHERE "main_author"."id" = '49' LIMIT 21 **10 similar queries.**		0.23	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '53' LIMIT 21 **11 similar queries.**		0.22	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '49' **10 similar queries.**		0.56	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '23' **10 similar queries.**		0.24	Sel Expl
SELECT ••• FROM "main_bookcategory" WHERE "main_bookcategory"."id" = '24' LIMIT 21 **10 similar queries.**		0.17	Sel Expl
SELECT ••• FROM "main_author" WHERE "main_author"."id" = '50' LIMIT 21 **10 similar queries.**		0.18	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '54' LIMIT 21 **11 similar queries.**		0.17	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '50' **10 similar queries.**		0.15	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '24' **10 similar queries.**		0.21	Sel Expl
SELECT ••• FROM "main_bookcategory" WHERE "main_bookcategory"."id" = '25' LIMIT 21 **10 similar queries.**		0.18	Sel Expl
SELECT ••• FROM "main_author" WHERE "main_author"."id" = '51' LIMIT 21 **10 similar queries.**		0.18	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '55' LIMIT 21 **11 similar queries.**		0.27	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '51' **10 similar queries.**		0.22	Sel Expl

Hide »

History ☑
/core/books/

Versions ☑
Django 4.2.3

Time ☑
Total: 251.10ms

Settings ☑

Headers ☑

Request ☑
view

SQL ☑
53 queries in 12.30ms

Static files ☑
10 files used

Templates ☑
rest_framework/api.html

Cache ☑
0 calls in 0.00ms

Signals ☑
34 receivers of 15 signals

Intercept redirects □

Profiling □

Locating N+1 problems

- nplusone & django-debug-toolbar → Both very easy to configure
- Only meant for local usage to problem identification and not on Production
- Once identified, how are they resolved?

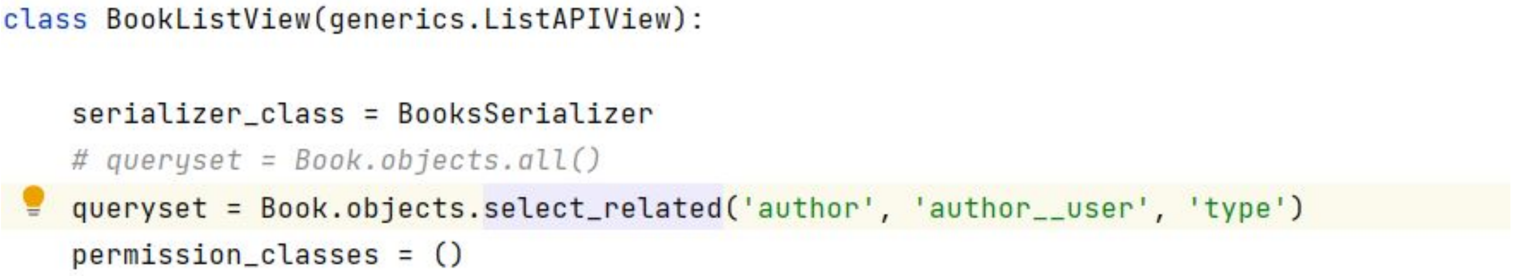
Resolving N+1 problems

- Built-in solutions in Queryset provided by Django
- select_related (*fields)
- prefetch_related (*lookups)

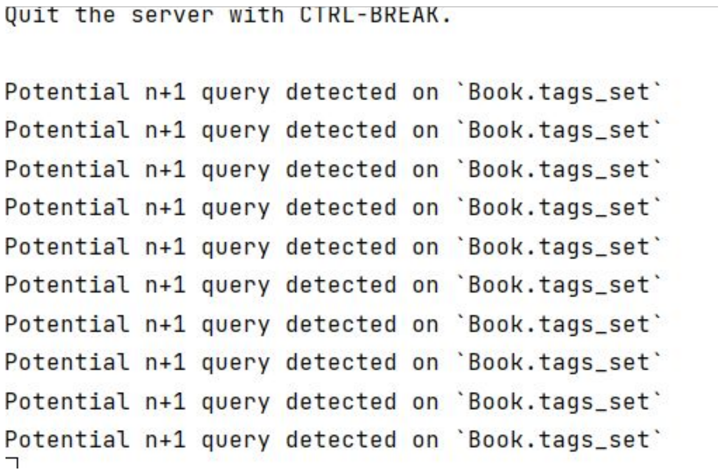
select_related

- Takes list of foreign key fields when fetching from DB
    - Follows the FK relationships
- Single complex query to fetch object and its related FK object instead of doing separate round trips for each related obj
- Prefetch both main FK of the object and FKs of the related objects as well

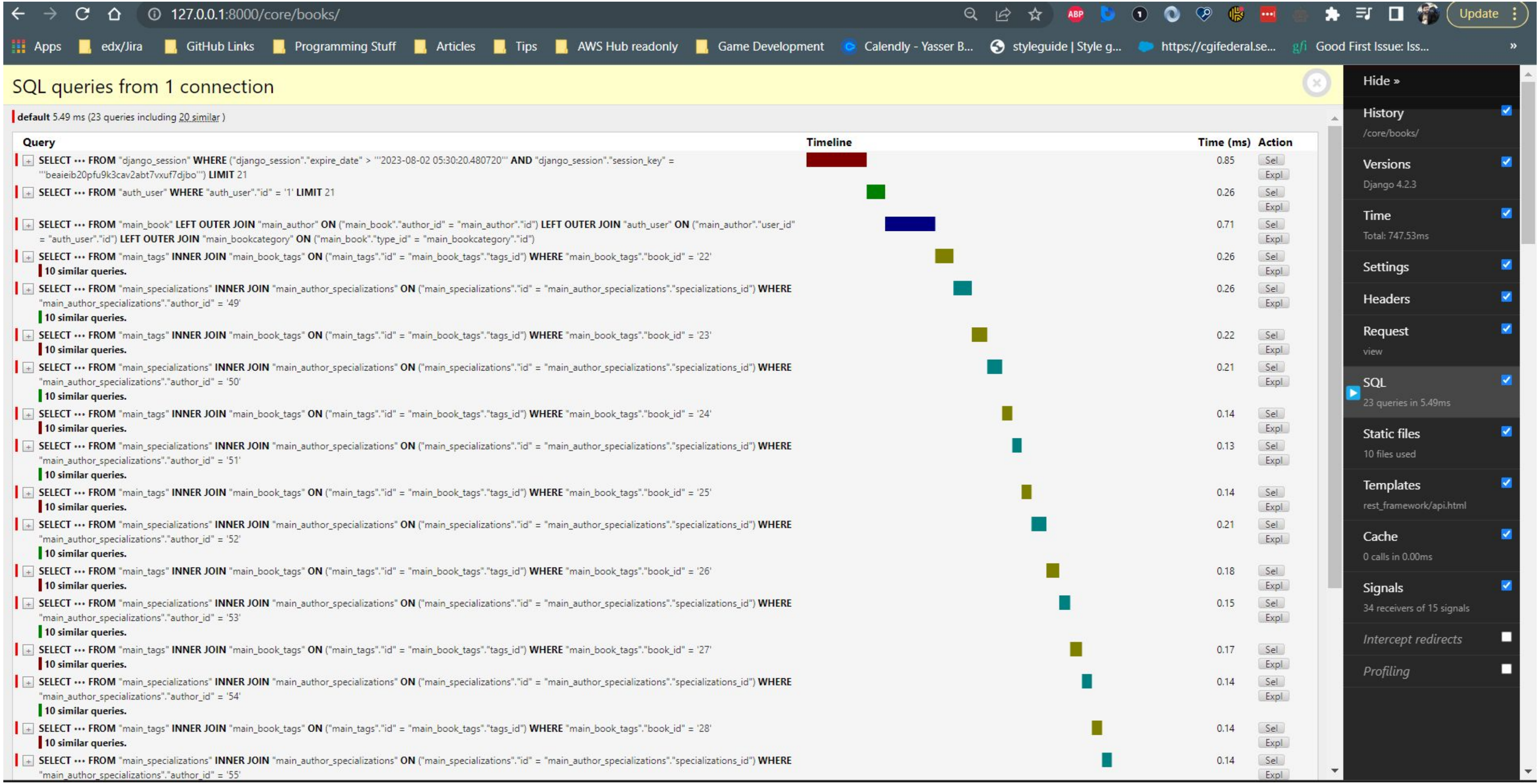
select_related: Book listing



```
class BookListView(generics.ListAPIView):

    serializer_class = BooksSerializer
    # queryset = Book.objects.all()
    queryset = Book.objects.select_related('author', 'author__user', 'type')
    permission_classes = ()
```



```
Quit the server with CTRL-BREAK.

Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
Potential n+1 query detected on `Book.tags_set`
```



SQL queries from 1 connection

default 5.49 ms (23 queries including 20 similar)

Query	Timeline	Time (ms)	Action
SELECT ••• FROM "django_session" WHERE ("django_session"."expire_date" > '"2023-08-02 05:30:20.480720"' AND "django_session"."session_key" = '"beaieib20pfu9k3cav2abt7vxuf7djbo"') LIMIT 21		0.85	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '1' LIMIT 21		0.26	Sel Expl
SELECT ••• FROM "main_book" LEFT OUTER JOIN "main_author" ON ("main_book"."author_id" = "main_author"."id") LEFT OUTER JOIN "auth_user" ON ("main_author"."user_id" = "auth_user"."id") LEFT OUTER JOIN "main_bookcategory" ON ("main_book"."type_id" = "main_bookcategory"."id")		0.71	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '22' **10 similar queries.**		0.26	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '49' **10 similar queries.**		0.26	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '23' **10 similar queries.**		0.22	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '50' **10 similar queries.**		0.21	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '24' **10 similar queries.**		0.14	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '51' **10 similar queries.**		0.13	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '25' **10 similar queries.**		0.14	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '52' **10 similar queries.**		0.21	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '26' **10 similar queries.**		0.18	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '53' **10 similar queries.**		0.15	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '27' **10 similar queries.**		0.17	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '54' **10 similar queries.**		0.14	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" = '28' **10 similar queries.**		0.14	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '55'		0.14	Sel Expl

Hide »

History ☑
/core/books/

Versions ☑
Django 4.2.3

Time ☑
Total: 747.53ms

Settings ☑

Headers ☑

Request ☑
view

SQL ☑
23 queries in 5.49ms

Static files ☑
10 files used

Templates ☑
rest_framework/api.html

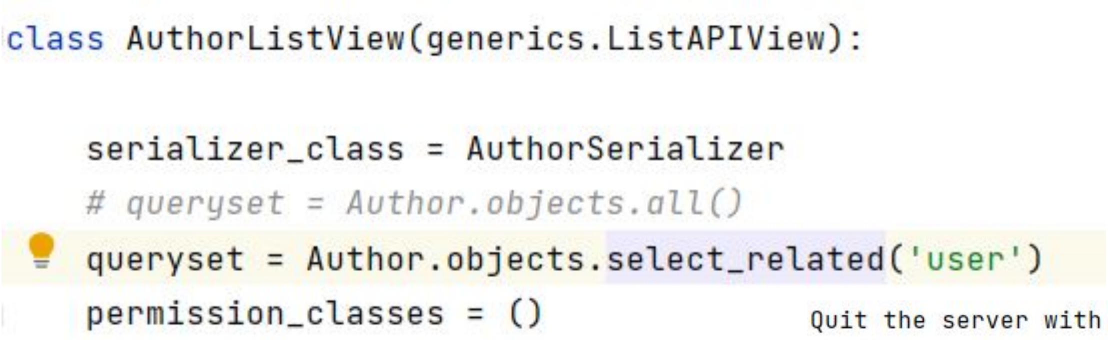
Cache ☑
0 calls in 0.00ms

Signals ☑
34 receivers of 15 signals

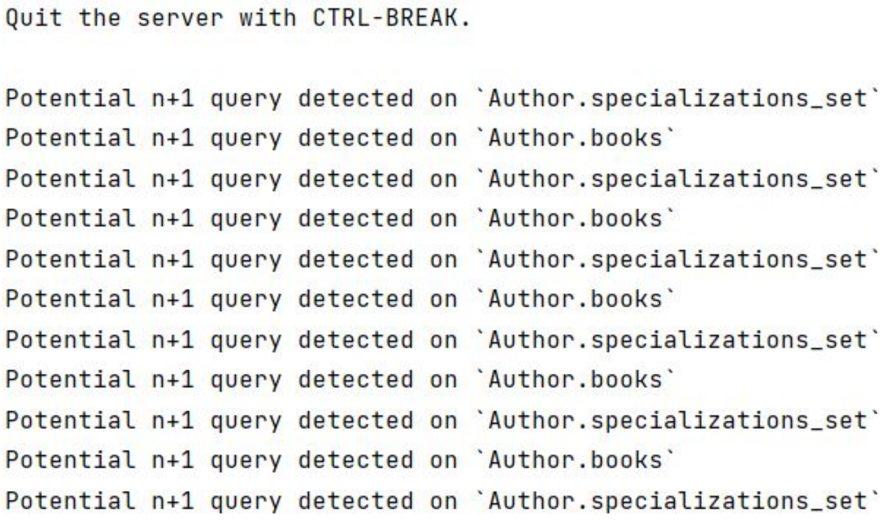
Intercept redirects □

Profiling □

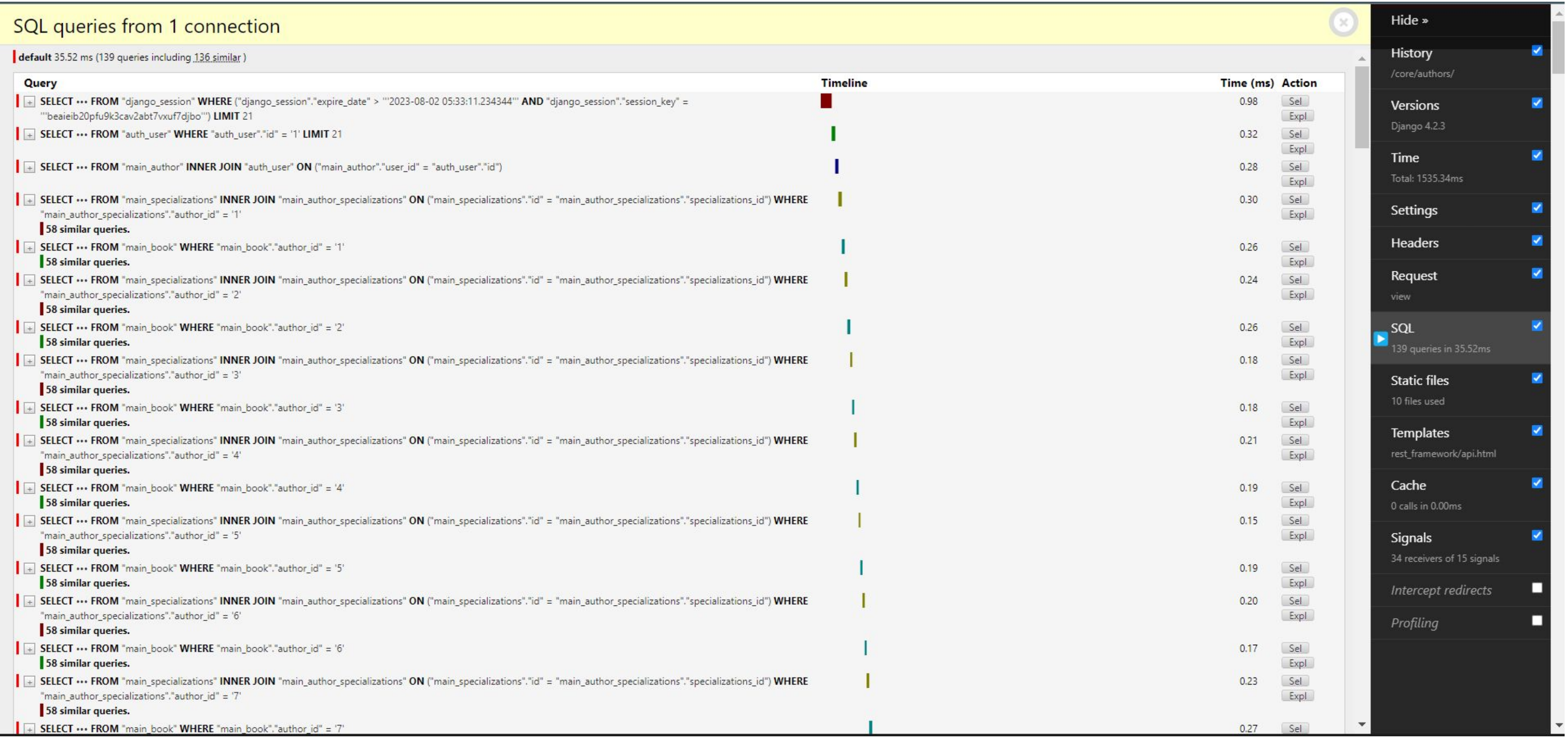
select_related: Author listing



```
class AuthorListView(generics.ListAPIView):

    serializer_class = AuthorSerializer
    # queryset = Author.objects.all()
    queryset = Author.objects.select_related('user')
    permission_classes = ()
```



```
Quit the server with CTRL-BREAK.

Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.specializations_set`
Potential n+1 query detected on `Author.books`
Potential n+1 query detected on `Author.specializations_set`
```



SQL queries from 1 connection

default 35.52 ms (139 queries including 136 similar)

Query	Timeline	Time (ms)	Action
SELECT ••• FROM "django_session" WHERE ("django_session"."expire_date" > '''2023-08-02 05:33:11.234344''' AND "django_session"."session_key" = '''beaieib20pfu9k3cav2abt7vxuf7djbo''') LIMIT 21		0.98	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '1' LIMIT 21		0.32	Sel Expl
SELECT ••• FROM "main_author" INNER JOIN "auth_user" ON ("main_author"."user_id" = "auth_user"."id")		0.28	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '1' 58 similar queries.		0.30	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '1' 58 similar queries.		0.26	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '2' 58 similar queries.		0.24	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '2' 58 similar queries.		0.26	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '3' 58 similar queries.		0.18	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '3' 58 similar queries.		0.18	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '4' 58 similar queries.		0.21	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '4' 58 similar queries.		0.19	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '5' 58 similar queries.		0.15	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '5' 58 similar queries.		0.19	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '6' 58 similar queries.		0.20	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '6' 58 similar queries.		0.17	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" = '7' 58 similar queries.		0.23	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" = '7'		0.27	Sel

Hide »

History ☑
/core/authors/

Versions ☑
Django 4.2.3

Time ☑
Total: 1535.34ms

Settings ☑

Headers ☑

Request ☑
view

SQL ☑
139 queries in 35.52ms

Static files ☑
10 files used

Templates ☑
rest_framework/api.html

Cache ☑
0 calls in 0.00ms

Signals ☑
34 receivers of 15 signals

Intercept redirects □

Profiling □

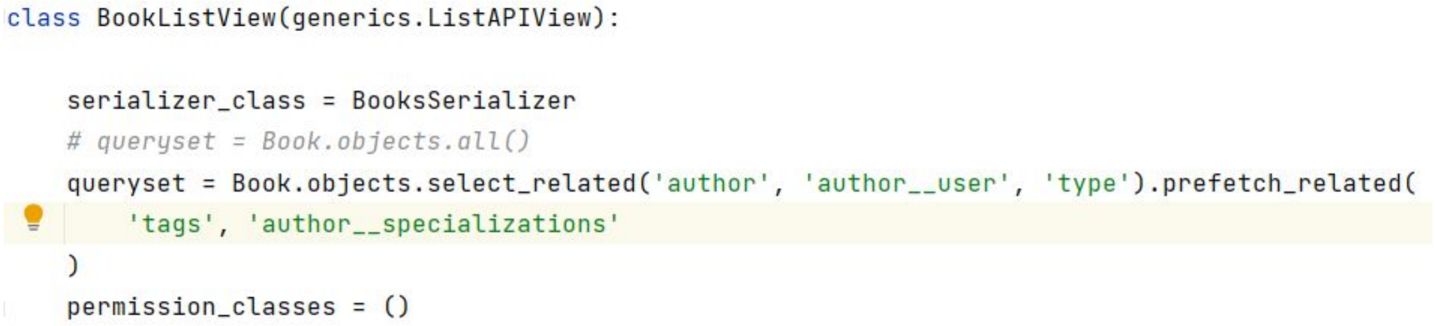
select_related

View	SQL Queries Count (Before)	SQL Queries Count (After)
Book Listing	53	23
Author Listing	197	139

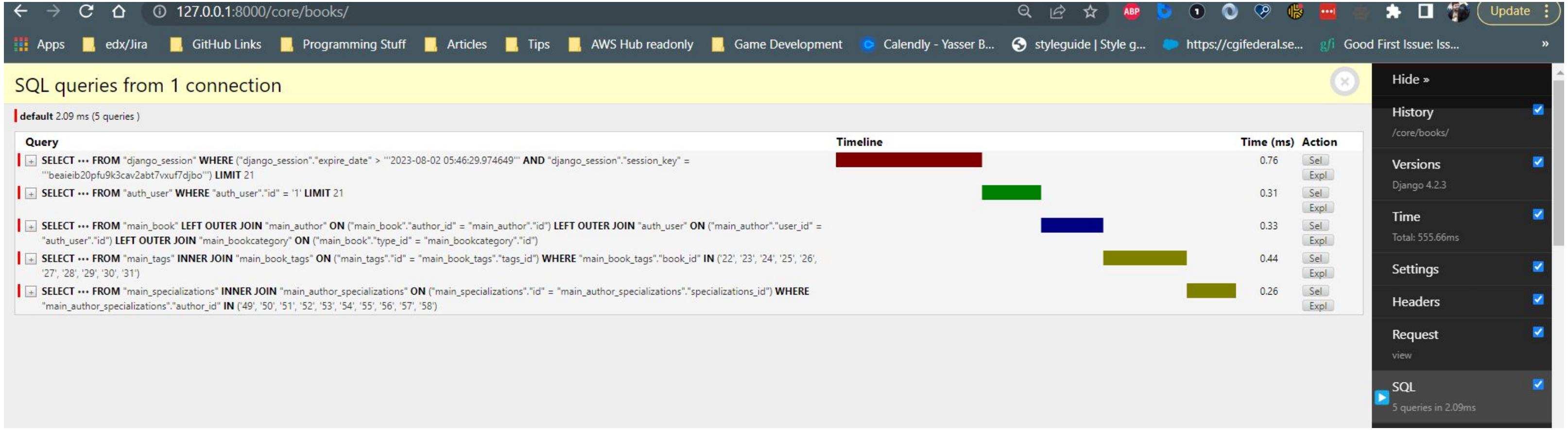
prefetch_related

- Reverse FK or Many-1 and Many-Many relationship
    - Can be applied on 1-Many & 1-1 relationships as well but not preferred
- Prefetch the objects via SQL and performs the join operation in Python
    - To avoid the large table resulting from joining large number of M2M
- Like select_related, Prefetch both main relationships of the object and relationships of the related objects as well

Prefetch_related: Book listing



```
class BookListView(generics.ListAPIView):

    serializer_class = BooksSerializer
    # queryset = Book.objects.all()
    queryset = Book.objects.select_related('author', 'author__user', 'type').prefetch_related(
        'tags', 'author__specializations'
    )
    permission_classes = ()
```



SQL queries from 1 connection

default 2.09 ms (5 queries)

Query	Timeline	Time (ms)	Action
SELECT ••• FROM "django_session" **WHERE** ("django_session"."expire_date" > '''2023-08-02 05:46:29.974649''' **AND** "django_session"."session_key" = '''beaieib20pfu9k3cav2abt7vxuf7djbo''') **LIMIT** 21	▨	0.76	Sel Expl
SELECT ••• FROM "auth_user" **WHERE** "auth_user"."id" = '1' **LIMIT** 21	▨	0.31	Sel Expl
SELECT ••• FROM "main_book" **LEFT OUTER JOIN** "main_author" **ON** ("main_book"."author_id" = "main_author"."id") **LEFT OUTER JOIN** "auth_user" **ON** ("main_author"."user_id" = "auth_user"."id") **LEFT OUTER JOIN** "main_bookcategory" **ON** ("main_book"."type_id" = "main_bookcategory"."id")	▨	0.33	Sel Expl
SELECT ••• FROM "main_tags" **INNER JOIN** "main_book_tags" **ON** ("main_tags"."id" = "main_book_tags"."tags_id") **WHERE** "main_book_tags"."book_id" **IN** ('22', '23', '24', '25', '26', '27', '28', '29', '30', '31')	▨	0.44	Sel Expl
SELECT ••• FROM "main_specializations" **INNER JOIN** "main_author_specializations" **ON** ("main_specializations"."id" = "main_author_specializations"."specializations_id") **WHERE** "main_author_specializations"."author_id" **IN** ('49', '50', '51', '52', '53', '54', '55', '56', '57', '58')	▨	0.26	Sel Expl

Hide »

History ☑
/core/books/

Versions ☑
Django 4.2.3

Time ☑
Total: 555.66ms

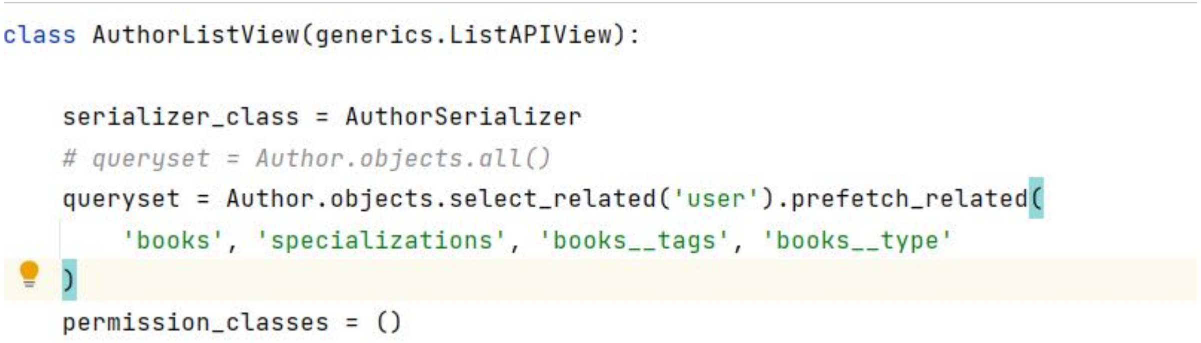
Settings ☑

Headers ☑

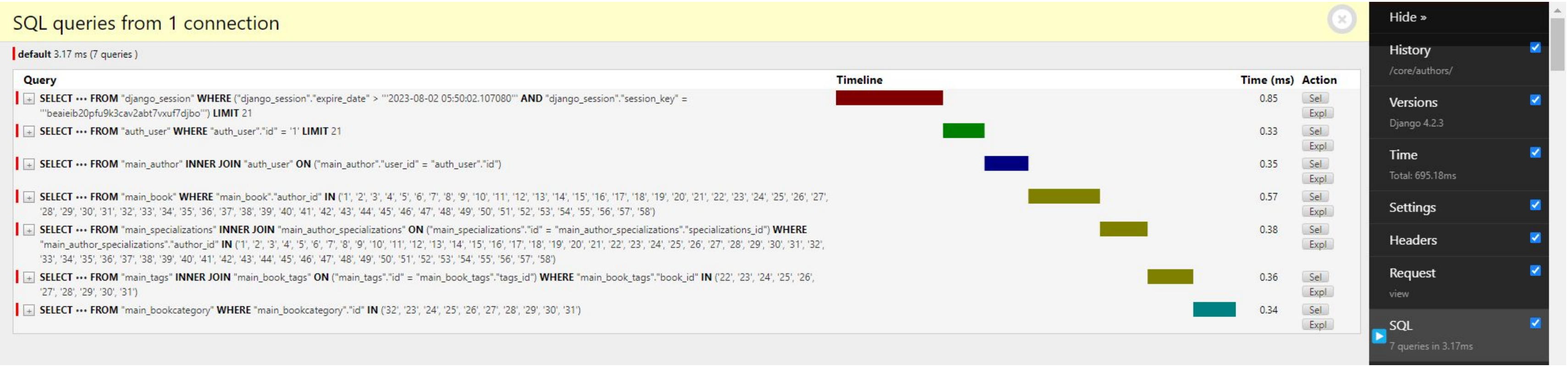
Request ☑
view

SQL ☑
5 queries in 2.09ms

Prefetch_related: Author Listing



```
class AuthorListView(generics.ListAPIView):

    serializer_class = AuthorSerializer
    # queryset = Author.objects.all()
    queryset = Author.objects.select_related('user').prefetch_related(
        'books', 'specializations', 'books__tags', 'books__type'
    )
    permission_classes = ()
```



SQL queries from 1 connection

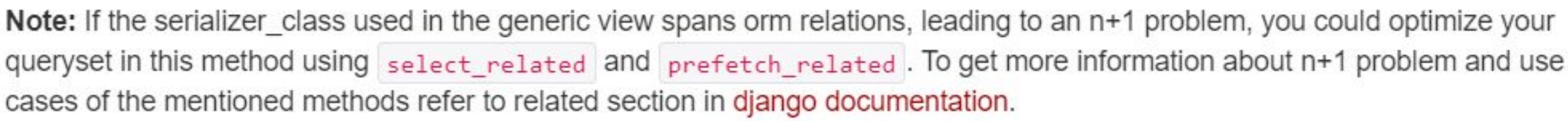
default 3.17 ms (7 queries)

Query	Timeline	Time (ms)	Action
SELECT ••• FROM "django_session" WHERE ("django_session"."expire_date" > '''2023-08-02 05:50:02.107080''' AND "django_session"."session_key" = '''beaieib20pfu9k3cav2abt7vxuf7djbo''') LIMIT 21		0.85	Sel Expl
SELECT ••• FROM "auth_user" WHERE "auth_user"."id" = '1' LIMIT 21		0.33	Sel Expl
SELECT ••• FROM "main_author" INNER JOIN "auth_user" ON ("main_author"."user_id" = "auth_user"."id")		0.35	Sel Expl
SELECT ••• FROM "main_book" WHERE "main_book"."author_id" IN ('1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '38', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '50', '51', '52', '53', '54', '55', '56', '57', '58')		0.57	Sel Expl
SELECT ••• FROM "main_specializations" INNER JOIN "main_author_specializations" ON ("main_specializations"."id" = "main_author_specializations"."specializations_id") WHERE "main_author_specializations"."author_id" IN ('1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33', '34', '35', '36', '37', '38', '39', '40', '41', '42', '43', '44', '45', '46', '47', '48', '49', '50', '51', '52', '53', '54', '55', '56', '57', '58')		0.38	Sel Expl
SELECT ••• FROM "main_tags" INNER JOIN "main_book_tags" ON ("main_tags"."id" = "main_book_tags"."tags_id") WHERE "main_book_tags"."book_id" IN ('22', '23', '24', '25', '26', '27', '28', '29', '30', '31')		0.36	Sel Expl
SELECT ••• FROM "main_bookcategory" WHERE "main_bookcategory"."id" IN ('32', '23', '24', '25', '26', '27', '28', '29', '30', '31')		0.34	Sel Expl

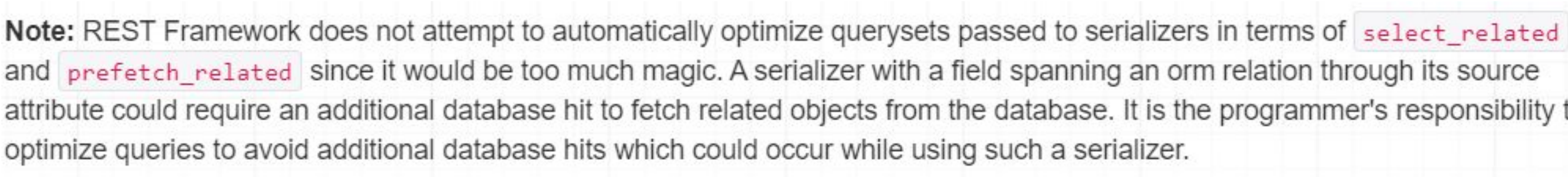
prefetch_related

View	SQL Queries Count (Initial)	SQL Queries Count (Final)
Book Listing	53	5
Author Listing	197	7

DRF recommendation



Note: If the serializer_class used in the generic view spans orm relations, leading to an n+1 problem, you could optimize your queryset in this method using `select_related` and `prefetch_related`. To get more information about n+1 problem and use cases of the mentioned methods refer to related section in django documentation.



Note: REST Framework does not attempt to automatically optimize querysets passed to serializers in terms of `select_related` and `prefetch_related` since it would be too much magic. A serializer with a field spanning an orm relation through its source attribute could require an additional database hit to fetch related objects from the database. It is the programmer's responsibility t optimize queries to avoid additional database hits which could occur while using such a serializer.

Closing Thoughts

- N+1 is pretty common in core Django views & DRF APIs
- The performance impact of N+1 issues won’t be obvious until the underlying data hit a certain limit
    - This also makes identifying N+1 on local challenging if the data is limited
- N+1 fixups result in complex joins which can impact the performance if an object has too many relationship
    - Try combination of select & prefetch related to understand which yield best results for your application

Closing Thoughts

- Inspired from similar talk at Python Web Conf "**Finding and fixing an N+1 problem in the Django docs**"
- https://www.youtube.com/watch?v=FzmOnzNov-E
- Slides are accessible at https://github.com/DawoudSheraz/conference-talks
- Code https://github.com/DawoudSheraz/pycon-au-23-demo-app

References

1. https://www.django-rest-framework.org/
2. https://testdriven.io/blog/drf-basics/
3. https://docs.djangoproject.com/en/4.2/ref/models/querysets/#select-related
4. https://docs.djangoproject.com/en/4.2/ref/models/querysets/#prefetch-related
5. https://github.com/jmcarp/nplusone
6. https://stackoverflow.com/questions/97197/what-is-the-n1-selects-problem-in-orm-object-relational-mapping