



# Make Django Admin Great Again

---

OPTIMIZATON FOR TABLE WITH MILLION ROWS

MINCHANG (CARSON) ZHANG

# Table of Content

---

- Intro to Django Admin
- Django Admin Optimization
  - The N + 1 queries problem in list view
  - Read only field
  - Raw id field
  - Filters
  - Paginator
- How to debug Django Admin performance

# What is Django Admin site

---

- Django Admin site is an automatic admin interface
  - Full front end interface
  - No extra front end code needed (to a certain degree)
- It reads metadata from your models to provide a quick, model-centric interface where trusted users can manage content on your website (but its not intended to be used as CMS)
- The recommended use is limited to an organization's internal management tool.

# Django Admin Optimization

---

AS YOUR APP GETS BIGGER

# The N+1 problem in list view

---

- What's N+1 problem: code that loops over a list of results from one query, and then performs another query per result. (Common for ORM)
- **list\_display** controls which fields are displayed on the change list page of the admin
- Django first counts the objects (1 query), then fetches the actual objects (N queries), then passes the data on to the template for rendering
- However, Django automatically queries db once for one row, so if there are 100 rows on a page (pagination size), there will be 101 queries!!
- How to fix this: use **list\_select\_related** to perform a join instead of fetching the names one by one

```
@admin.register(models.Product)
class ProductAdmin(admin.ModelAdmin):
    list_display = (
        'id',
        'name',
        'category',
    )

    list_select_related = (
        'category',
    )
```

# Read only fields

---

- In the detail page Django creates an editable element for each field.
- Choice fields and foreign key fields will be rendered as a <select> element (i.e. dropdown menu)
- The detail page will fetch the ENTIRE table and the option list!!
- How to fix it: use **readonly\_fields** so it will render the description of the related model

```
@admin.register(SomeModel)
def SomeModelAdmin(admin.ModelAdmin):
    readonly_fields = (
        'user',
    )
```

# Raw id field

---

- What if you have to change it?
- How to fix it: use **raw\_id\_fields**

```
class ArticleAdmin(admin.ModelAdmin):  
    raw_id_fields = ("newspaper",)
```

Newspaper:



# Filters

---

- Since most of the time admin site is used as a day-to-day support tool, most of the times we use the same filter
- Thus we can apply default filters for a given page so it doesn't have to fetch the entire database
- Sometimes we can also cache certain filter results for a quicker display



# Paginator

---

- List display page comes with paginator and a count of the total number of pages
- However, Django spends most of time (> 99%) to count the rows in the table.
- What happened was that Django counts the entire table to know how many rows first, and then determine how many pages to show on the template
- How to fix it: write custom paginator to override the default paginator
  - There are many ways to do it, but the way I used was to estimate how many rows in the table use database provided function (i.e. **reltuples** for postgres)

# Other tricks

---

- You can always override the **get\_query** function to write custom queries
- **show\_full\_result\_count** flag prevents Django from displaying the total amount of rows in the list view
- **defer** is used to do a lazy evaluation of queries (i.e. only fetch it when you need it), best suitable for large columns (i.e. json, text fields, etc)
- Set **date\_hierarchy** to the name of a **DateField** or **DateTimeField** in your model, and change the list page will include a date-based drilldown navigation by that field, but its quite slow
  - How to fix it: Similar as the filter, we can give pre-estimated date range to limit the query scope so it is much faster

Django administration

Home › Sales › Sale model

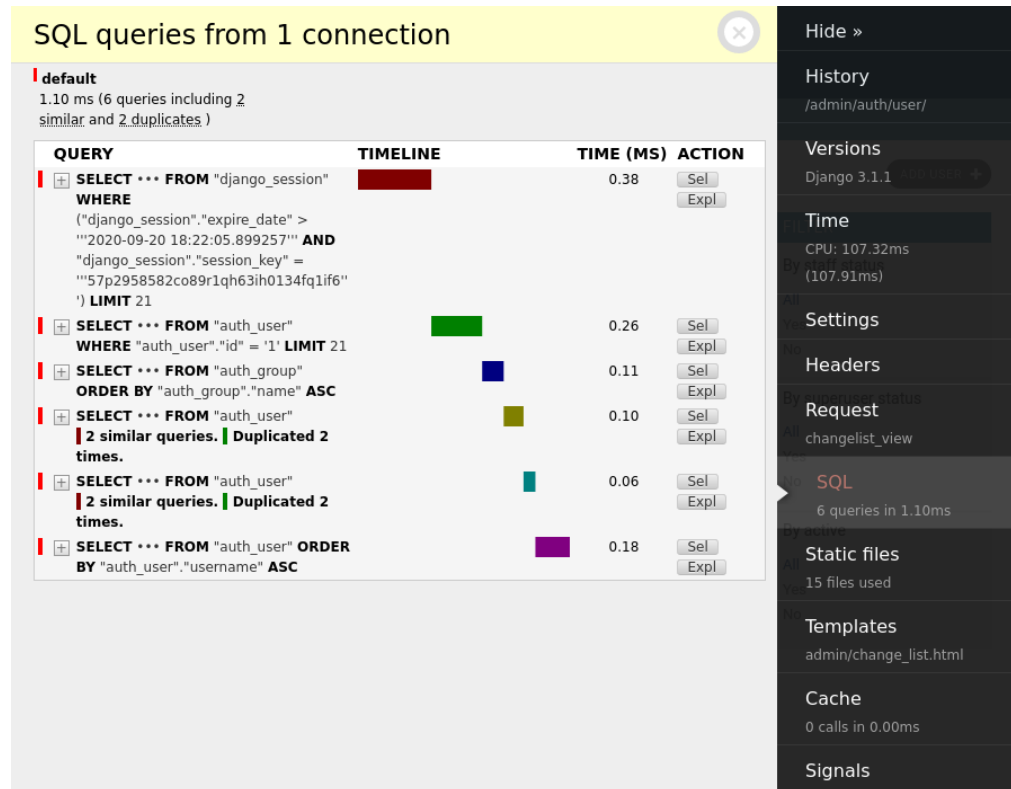
Select Sale model to change

◀ All dates January 2017 February 2017 March 2017

- Generic query optimization tips such as **select\_related**, **prefetch\_related** and **Subquery**
- The rule of thumb is try to target related data, and try NOT to fetch the entire table

# How to debug Django admin performance

## Use Django debug toolbar



The screenshot displays the Django Debug Toolbar (DDT) interface. At the top, a yellow header reads "SQL queries from 1 connection". Below this, a summary bar indicates "default" and "1.10 ms (6 queries including 2 similar and 2 duplicates)". The main panel shows a list of SQL queries with their execution times and actions. A sidebar on the right contains various tabs like History, Versions, Time, Settings, Headers, Request, SQL (selected), Static files, Templates, Cache, and Signals.

| QUERY   | TIMELINE     | TIME (MS) | ACTION      |
|---|--------------|-----------|-------------|
| <b>SELECT *** FROM "django_session" WHERE ("django_session"."expire_date" &gt; "'2020-09-20 18:22:05.899257'" AND "django_session"."session_key" = "'57p2958582co89r1qh63ih0134fq1if6"') LIMIT 21</b> | [Red bar]    | 0.38      | Sel<br>Expl |
| <b>SELECT *** FROM "auth_user" WHERE "auth_user"."id" = '1' LIMIT 21</b>  | [Green bar]  | 0.26      | Sel<br>Expl |
| <b>SELECT *** FROM "auth_group" ORDER BY "auth_group"."name" ASC</b>  | [Blue bar]   | 0.11      | Sel<br>Expl |
| <b>SELECT *** FROM "auth_user" 2 similar queries. Duplicated 2 times.</b>   | [Yellow bar] | 0.10      | Sel<br>Expl |
| <b>SELECT *** FROM "auth_user" 2 similar queries. Duplicated 2 times.</b>   | [Teal bar]   | 0.06      | Sel<br>Expl |
| <b>SELECT *** FROM "auth_user" ORDER BY "auth_user"."username" ASC</b>  | [Purple bar] | 0.18      | Sel<br>Expl |

## Setup Django logging

```
LOGGING = {  
    ...  
    'loggers': {  
        'django.db.backends': {  
            'level': 'DEBUG',  
            ...  
        },  
        ...  
    }  
}
```

# Reference

---

- [Things You Must Know About Django Admin As Your App Gets Bigger](#)
- [Optimizing Django Admin Paginator](#)
- [Scaling Django Admin Date Hierarchy](#)
- [Django and the N+1 Queries Problem](#)

*Questions?*

THANKS FOR VIEWING