Postgres + analytics = ❤️

Advisory

The following presentation contains nothing sexy, revolutional or new …

… just KISS on a company scale

Context

What does Vumedi do?

Video educational platform for doctors

1. Get videos
2. Show them to doctors
3. Promote them
4. Repeat

Context

Tech stack

“Boring” technology

1. Python + Django + Celery
2. AWS
3. Postgres 13 (RDS)

What questions do we need to answer?

- What are top videos per specialty?
- How did each email video promotion do?
  - Sends, opens, clicks, unsubscribes, views
- Are we delivering our contractual obligations for video promotions?
- Active users

V1

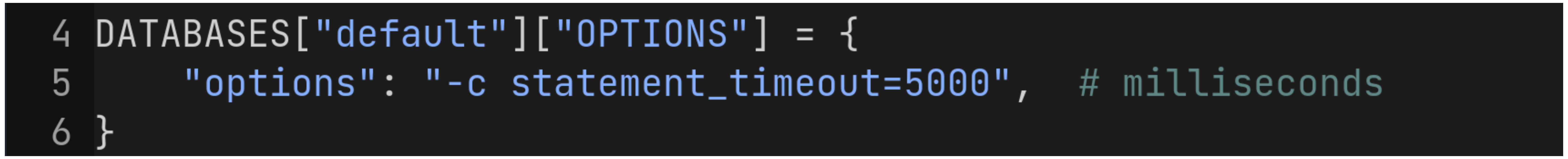
Just query the production database directly

- If the tables are small enough, no impact on site usability
- If needed, upscale instance size

V1

Just query the production database directly

- Problem: Runaway long running queries impacting user experience
- Solution: Set statement timeout
    - Server level - changing Postgres server configuration requires downtime
    - Connection level - Django-only



```
DATABASES["default"]["OPTIONS"] = {
    "options": "-c statement_timeout=5000",  # milliseconds
}
```

- Database level - ALTER TABLE

V2

Aggregations

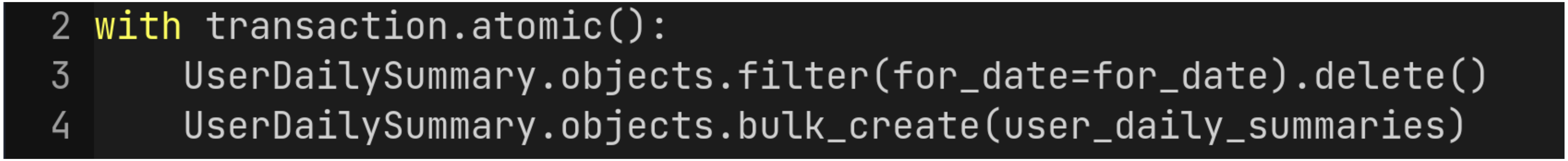
- Daily/weekly/monthly
- User/video/mail campaign

- How?
  - Periodic Celery tasks during off-hours

V2

Aggregations

- Recipe
  - Pull data into memory
  - Crunch numbers
  - Prepare model instances for saving (e.g. UserDailySummary)
  - Bulk create inside a transaction
    - No upsert in Django for now :(

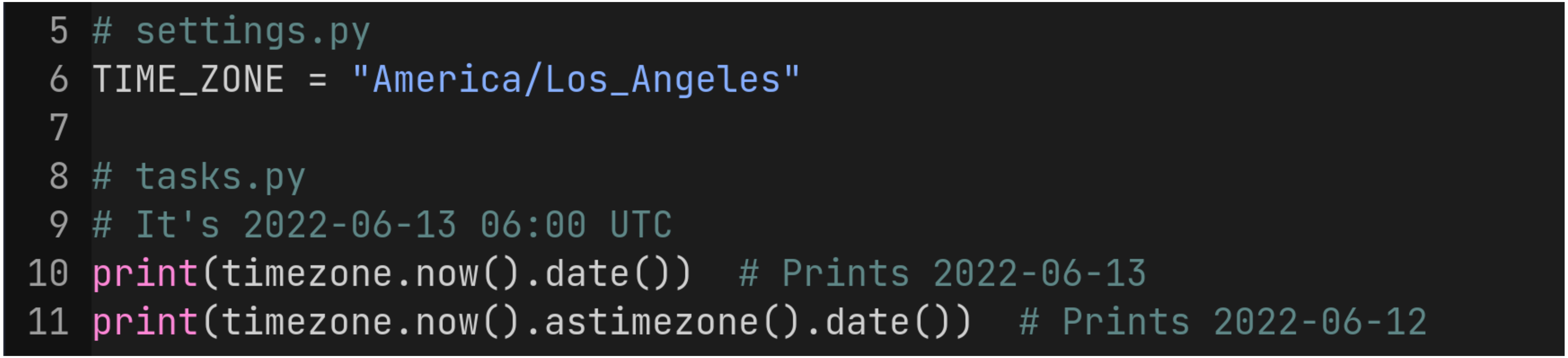


```
with transaction.atomic():
    UserDailySummary.objects.filter(for_date=for_date).delete()
    UserDailySummary.objects.bulk_create(user_daily_summaries)
```

V2

Aggregations

- What is “today”?



```
# settings.py
TIME_ZONE = "America/Los_Angeles"

# tasks.py
# It's 2022-06-13 06:00 UTC
print(timezone.now().date())  # Prints 2022-06-13
print(timezone.now().astimezone().date())  # Prints 2022-06-12
```

“There are two hard problems in computer science: Unicode and time zones.”

- T.M.

V3

Denormalize + Materialized Views

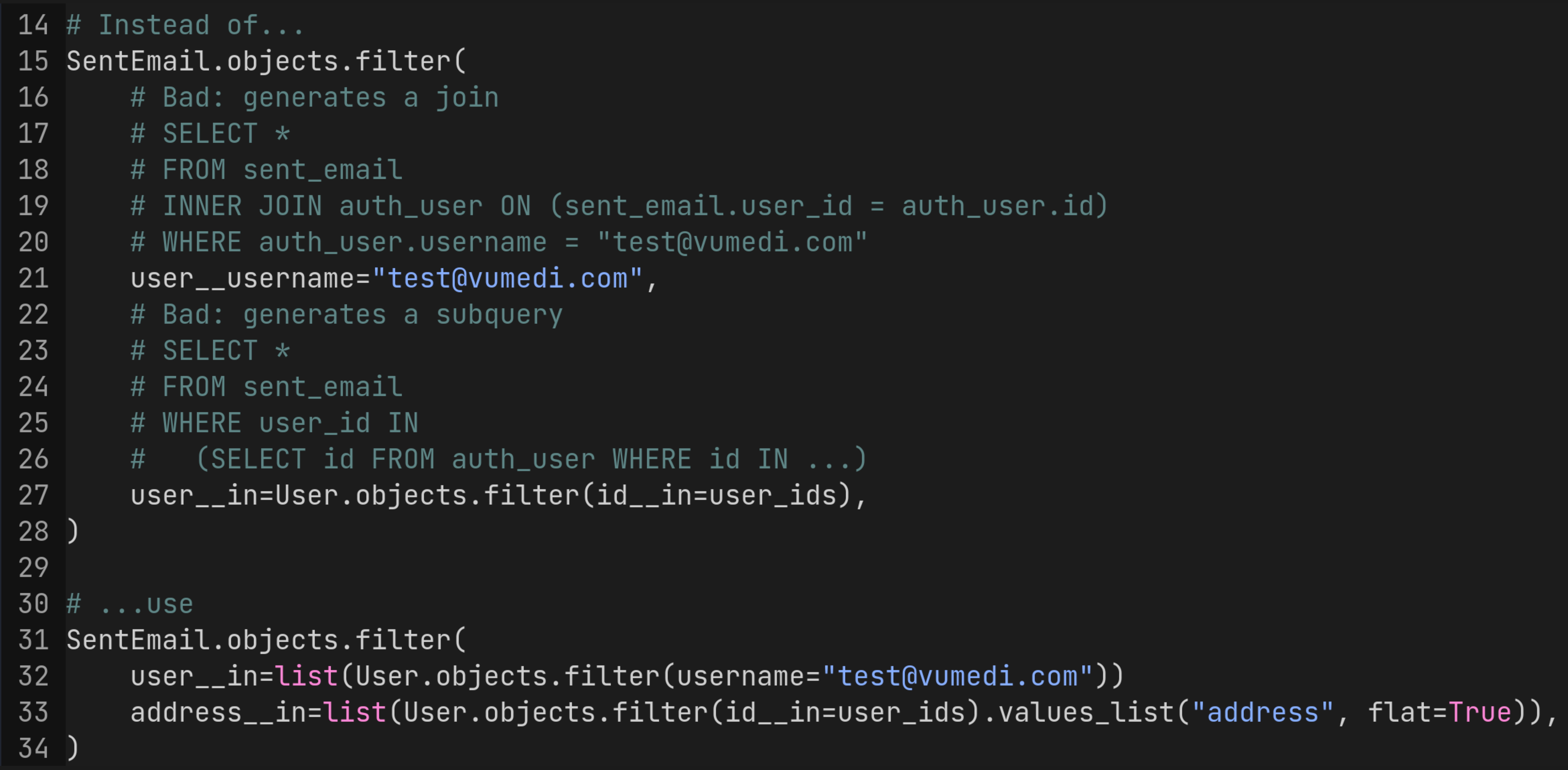
- Problem: Tableau continuously joins the same set of tables when building an extract
  - If tables are large, it consumes a lot of DB resources
- Solution: Build denormalized tables and refresh materialized views during off-hours
  - Have Tableau read those instead

V4

Partition the tables

- Problem: Need to calculate an aggregation over a small part of a large table
- Solution: Partition the table
  - You usually don't need all of the data since beginning of time - last couple of weeks are enough

- Cons: have to be careful with filters and joins which cause entire table scans



```python
# Instead of...
SentEmail.objects.filter(
    # Bad: generates a join
    # SELECT *
    # FROM sent_email
    # INNER JOIN auth_user ON (sent_email.user_id = auth_user.id)
    # WHERE auth_user.username = "test@vumedi.com"
    user__username="test@vumedi.com",
    # Bad: generates a subquery
    # SELECT *
    # FROM sent_email
    # WHERE user_id IN
    #   (SELECT id FROM auth_user WHERE id IN ...)
    user__in=User.objects.filter(id__in=user_ids),
)

# ...use
SentEmail.objects.filter(
    user__in=list(User.objects.filter(username="test@vumedi.com"))
    address__in=list(User.objects.filter(id__in=user_ids).values_list("address", flat=True)),
)
```

V5

Read Replica

- Slight replication lag is acceptable
- Now your analytics queries do not impact the user experience!

- Cons: double the price :(

V6

Truncate Unnecessary Data

- Do we really care about what happened 5 years ago?
- Backup to S3 and truncate old data

V7?

Columnar Data Warehouse

- How the big boys do it:
    - Redshift / BigQuery / Vertica / Databricks / Snowflake
    - Segment / Snowplow
- "Horse before the cart" situation

V7?

Columnar Data Warehouse

- Business doesn't care about your data store
- GA / Mixpanel / Amplitude can provide them with necessary data
    - Moves dashboard ownership to business teams
- Keep Postgres as source of truth, push events to 3rd party
    - As long as it's cheaper than data warehouse + data engineer + data analyst

Why not an OLAP database?

- Another moving part in the system
- Not “boring” technology
- Not a silver bullet
  - You still need to think about structuring the schema
  - But, now you have two databases to think about

Conclusion

- We’re not Google / Facebook / Youtube
  - If we were, we would have a team who would take care of this
- Postgres can get you very far

Thank you!

Q & A