# Reflexión sobre el uso del ORM en Django

Actividad 5 | Módulo 7

**Autor:** *Ximena Garrido*
**Fecha:** Febrero 2026

## 1. ¿Qué ventajas encuentras en usar el ORM frente a SQL tradicional?

El **ORM de Django** permite interactuar con la base de datos utilizando objetos y métodos de Python, eliminando la necesidad de escribir SQL manual para tareas comunes.

- **Claridad y Legibilidad:** Consultas como `filter()`, `exclude()` o `annotate()` son semánticamente más claras y fáciles de mantener que sentencias SQL extensas.
- **Seguridad:** Django previene inyecciones SQL automáticamente al parametrizar las consultas de forma interna.
- **Integridad Referencial:** Las relaciones (`ForeignKey`) se gestionan de manera automática y coherente gracias a la integración con los modelos.
- **Portabilidad:** Facilita el cambio entre motores de base de datos (ej. de SQLite a PostgreSQL) sin reescribir código.

## 2. ¿En qué situaciones te parece útil ejecutar SQL directamente?

Aunque el ORM es potente, el SQL directo es indispensable en escenarios específicos:

- **Optimización Extrema:** Cuando se requieren consultas muy específicas que el ORM genera de forma ineficiente.
- **Joins Complejos:** Para tener control total sobre uniones de tablas intrincadas.
- **Reportes Avanzados:** Para agregaciones complejas o uso de funciones nativas del motor (ej. PostgreSQL).

## 3. ¿Qué dificultades encontraste con consultas avanzadas?

El proceso de aprendizaje presentó los siguientes desafíos:

1. **Anotaciones y Agregaciones:** Comprender la lógica detrás de `annotate` y `Count`.
2. **Relaciones Inversas:** El uso correcto de `related_name` para acceder a datos desde el modelo "padre".
3. **Traducción Mental:** Visualizar cómo Python convierte una línea de código en una sentencia SQL real.
4. **Duplicidad de Datos:** Manejar consultas con múltiples relaciones sin generar filas repetidas en el resultado.

> *Sin embargo, una vez asimilada la estructura, el ORM se convierte en una herramienta insustituible.*

## Conclusión

El **ORM de Django** cubre la gran mayoría de los casos de uso en el desarrollo web moderno, garantizando:

- 🔒 Seguridad
- 🛠️ Mantenibilidad
- ⏭️ Rapidez de desarrollo

No obstante, el dominio de **SQL** sigue siendo una habilidad fundamental. Entender qué ocurre "bajo el capó" es lo que permite optimizar el rendimiento y resolver problemas complejos cuando la abstracción del ORM no es suficiente.

---