

УДК 62-97/-98

Dmytro Derhachov<sup>1</sup>, Nataliia Zhukova<sup>2</sup>

<sup>1</sup>student of group CST-123 NU “Zaporizhzhia Polytechnic”

<sup>2</sup>PhD (Philology), associate professor NU “Zaporizhzhia Polytechnic”

## **ORM IN WEB DEVELOPMENT**

SQL is the standard language for relational databases. The demand for SQL Developers is rising globally, particularly in the finance and technology sectors, driven by extensive data management needs.

Key features of SQL include efficiency in handling large data volumes, compatibility with various database systems, and a user-friendly interface. Proficiency in SQL is crucial for professionals navigating relational databases in data-driven industries. However, SQL has its limitations.

Expressiveness is one of the limitations. SQL’s complexity limits its effectiveness in handling complex procedural logic and data transformations, making implementation cumbersome. Portability concerns also arise due to variations in SQL dialects among different database management systems, posing challenges for code portability and requiring modifications for compatibility.

Maintenance difficulty is another issue. Large SQL scripts are challenging to maintain and debug, especially without proper documentation, leading to potential errors. Performance issues can also occur due to inefficient queries and underutilization of indexes, particularly in complex scenarios.

One solution to these issues is using Object-Relational Mapping (ORM) tools like Django ORM, which offers several advantages. Firstly, expressiveness is improved; Django ORM allows developers to handle complex database tasks using Python, simplifying database interactions. Secondly, portability is enhanced; Django ORM ensures code compatibility across different database types, saving time and effort in adaptation. Thirdly, maintenance is streamlined; organized code structure and queriesets in Django ORM enhance maintenance and comprehension. Finally, performance is optimized; features like lazy loading and query optimization improve database performance, along with built-in support for indexes and optimizations.

Additionally, Django ORM provides various features to further enhance functionality. These include async support for facilitating asynchronous database operations, leveraging specific capabilities of different database

engines for enhanced functionality through database-specific features, an extension for geographic data handling called GeoDjango, enabling spatial queries and location-aware applications, query set expressions to perform complex database operations using Python syntax for better readability, schema migrations to automate database schema updates and simplify database management, and regular optimizations to enhance query performance and reduce database load.