RBAC (Role-Based Access Control)

- **Definition**: A security model that controls database/system access based on user roles within an organization, rather than assigning permissions directly to individual users.
- **Components**:
    - **Roles**: Defined job functions (e.g., admin, developer, read_only_user) with associated permissions.
    - **Permissions**: Specific rights granted to roles (e.g., SELECT, INSERT, UPDATE, DELETE on certain tables).
    - **Users**: Individuals assigned to one or more roles.
- **Advantages for SWEs**:
    - Simplifies permission management, especially in larger teams/systems.
    - Enhances security by adhering to the principle of least privilege (users only get access needed for their role).
    - Promotes consistency in access control policies.

Encryption

- **Definition**: The process of converting data into a secure, unreadable format (ciphertext) that requires a specific key to decrypt back into readable format (plaintext).
- **Purpose**: Protects sensitive data from unauthorized access, both when stored and when transmitted.
- **Common Types Relevant to SWEs**:
    - **Encryption At-Rest**: Protects data stored physically on disk (in database files, backups, etc.). Even if storage media is accessed directly, the data is unreadable without the key.
        - *Example*: Encrypting columns containing PII (Personally Identifiable Information) like Social Security Numbers or sensitive financial data within the database tables.
    - **Encryption In-Transit**: Protects data as it travels over networks (e.g., between the application server and the database server, or between the

user's browser and the web server). Typically achieved using protocols like TLS/SSL (HTTPS).

        - *Example*: Ensuring database connection strings specify SSL/TLS encryption; using HTTPS for all web traffic.

    - **Column-Level Encryption**: Encrypting specific sensitive columns within a table, while other columns remain unencrypted. Useful for targeted protection of fields like passwords or credit card numbers.

- **Key Considerations**:
    - **Performance**: Encryption/decryption adds computational overhead, which can impact database performance.
    - **Key Management**: Securely managing encryption keys is critical. Lost keys can mean lost data.

Data Masking

- **Definition**: Techniques used to obscure or replace sensitive data with realistic but non-sensitive substitutes.
- **Purpose**: Allows developers, testers, or analysts to work with a dataset that resembles production data (for testing, development, analytics) without exposing actual sensitive information. Primarily used in non-production environments.
- **Common Techniques**:
    - **Substitution**: Replacing original data with plausible but fake data (e.g., replacing real names with randomly generated names).
    - **Nulling Out / Redaction**: Replacing sensitive data with NULL values or fixed placeholders (e.g., 'XXXX-XXXX-XXXX-1234' for credit cards).
    - **Tokenization**: Replacing sensitive data elements with unique, non-sensitive identifiers ("tokens"). The original data is stored securely elsewhere, mapped to the tokens. Often used for credit card processing.
    - **Shuffling**: Randomly rearranging values within a column across different records.
- **Goal**: To create functionally useful but anonymized data for non-production use cases, protecting privacy and complying with regulations.