Database Security & Data Ethics

Learning Outcomes

- Understand and be able to explain the concept of database security.
- Be able to explain the six database security measures.
- Understand and be able to explain SQL injection attack.
- Be able to explain how to prevent SQL injection attack.
- Understand and be able to explain the concept of data ethics.
- Be able to explain the principles of data ethics.

Database Security

- Data is a valuable resource that must be strictly controlled and managed.
- Part (or all) of the corporate data may have strategic importance and therefore needs to be kept secure and confidential.

Database Security

- Database Security: Mechanisms that protect the database against intentional or accidental threats.
- Involves measures to avoid:
    - Theft and fraud
    - Loss of confidentiality (secrecy)
    - Loss of privacy
    - Loss of integrity
    - Loss of availability

Database Security

- **Threat**: any situation or event, whether intentional or unintentional, that will adversely affect a system or consequently an organization.

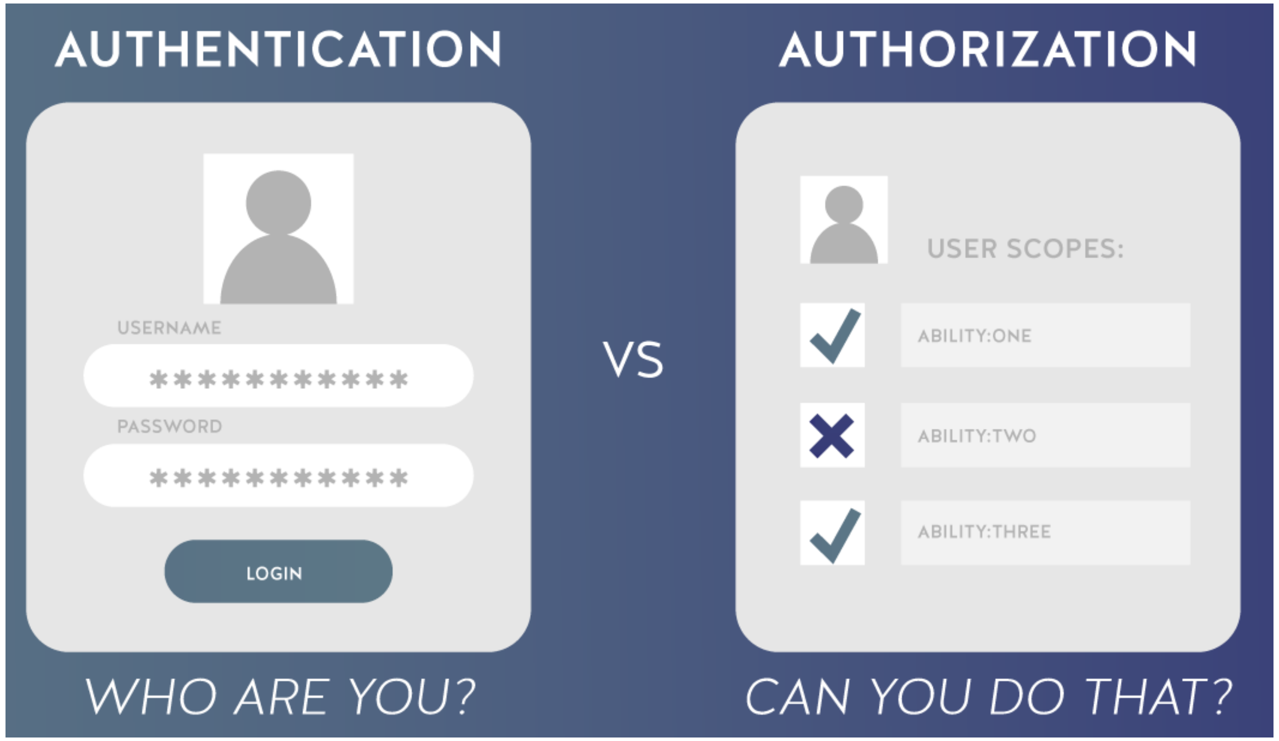
Examples of threats

THREAT	Theft and Fraud	Loss of Confidentiality	Loss of Privacy	Loss of Integrity	Loss of Availability
Illegal entry by hacker	√	√	√		
Theft of data	√	√	√		
Using another person’s means of access	√	√	√		
Data corruption owing to power loss or surge				√	√
Physical damage to equipment				√	√
Unauthorised amendment or copying of data	√			√	

Database Security measures

- Authorization
- Access controls
- Views
- Backup and recovery
- Integrity
- Encryption

Database Security measures - Authorization



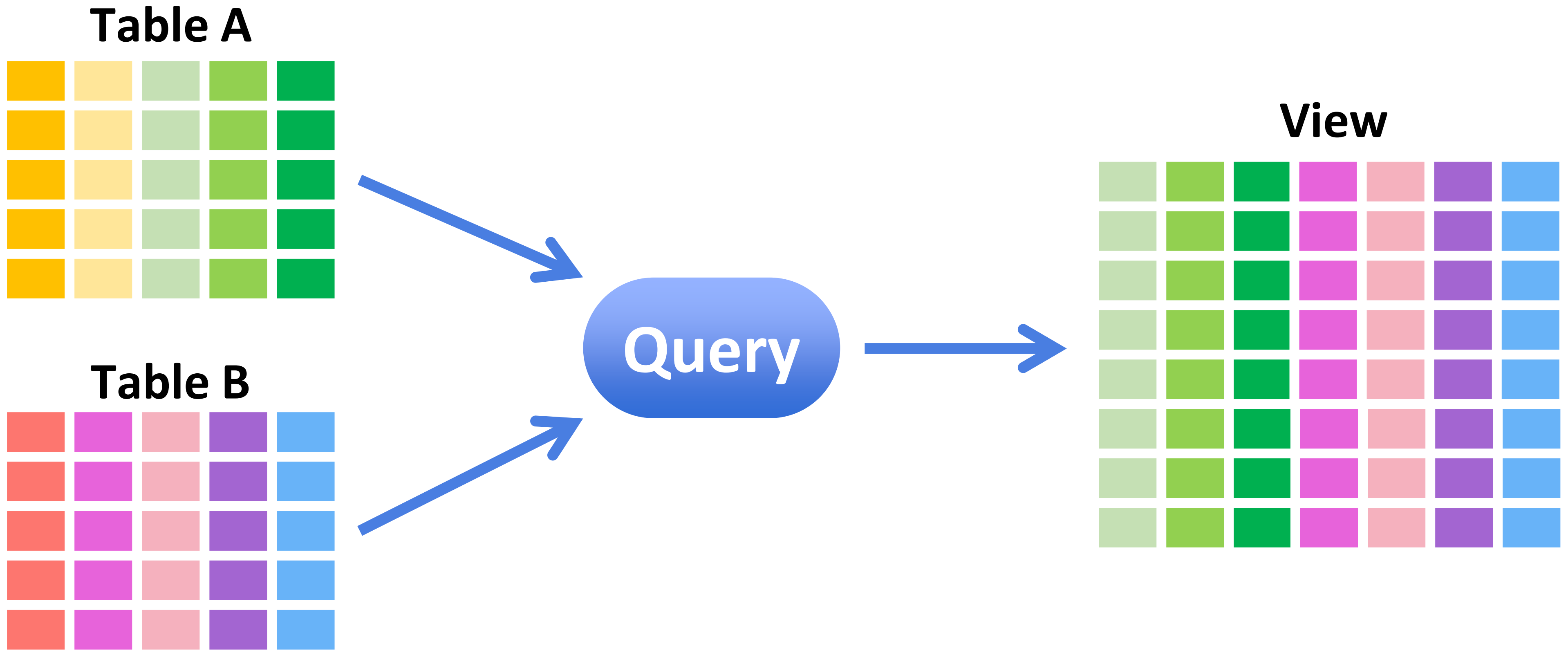
- The process of **authorization** involves authentication of subjects requesting access to objects.
  - where "subject" represents a user or program, and
  - "object" represents a database table, view, procedure, trigger, or any other object that can be created within the system.
- **Authentication** is a mechanism that determines whether a user is, who he or she claims to be.

Database Security measures – Access control

- Based on the granting and revoking of privileges.
- A privilege allows a user to create or access (that is read, write, or modify) some database object (such as a relation, view, and index) or to run certain DBMS utilities.
- Privileges are granted to users to accomplish the tasks required for their jobs.

Database Security measures – View

- A View is the dynamic result of one or more relational operations on the base relations.



Database Security measures – View

- A view is a virtual relation that does not actually exist in the database. It is a query that is saved in the database.
- Views are generally read-only.
- The view provides a security mechanism by hiding parts of the database from certain users. A view can be defined over several relations with a user being granted the appropriate privilege to use it, but not to use the base relations.

Database Security measures – Backup and Recovery

- Backup: process of periodically taking a copy of the database and log file (and possibly programs) to offline storage media.
- Recovery: process of restoring database to a correct state in the event of a failure. (Discussed in detail in lecture notes 3.3 Transaction management.)

Database Security measures – Backup and Recovery

- Backup: process of periodically taking a copy of the database and log file (and possibly programs) to offline storage media.
- Recovery: process of restoring database to a correct state in the event of a failure. (Discussed in detail in lecture notes 3.3 Transaction management.)

Database Security measures – Integrity

- Integrity: Prevents data from becoming invalid, and hence giving misleading or incorrect results.
- Integrity constraints (Discussed in 1.2 Relational Model):
  - Entity Integrity
  - Referential Integrity
  - General Constraints

Database Security measures – Encryption

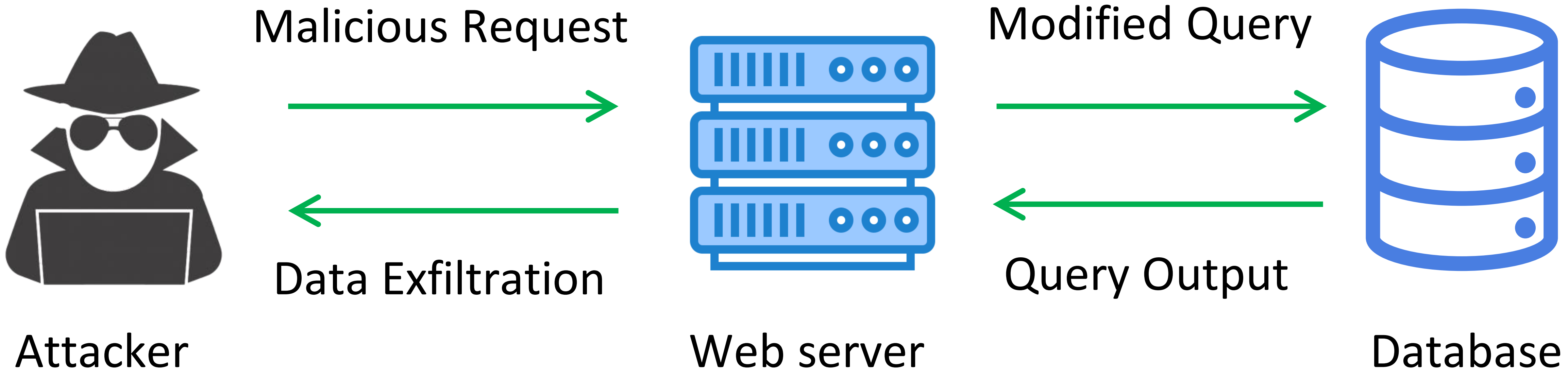
- Encryption: the encoding of the data by a special algorithm that renders the data unreadable by any program without the decryption key.
- If a database system holds particularly sensitive data (e.g. password, credit card information, etc), it may be deemed necessary to encrypt it as a precaution against possible external threats or attempts to access it.

A well-known attack involving Database - SQL Injection

SQL injection

- SQL injection (SQLi) is a web security attack that allows an attacker to interfere with the queries that an application makes to its database.
- This can allow an attacker to view data that they are not normally able to retrieve (unauthorized access to data).
- In many cases, an attacker can modify or delete this data, causing persistent changes to the database.

SQL injection



SQL injection example 1 – Retrieving hidden data

- Imagine a shopping application that displays products in different categories. When the user clicks on the Gifts category, their browser requests the URL:

```
https://insecure-website.com/products?category=Gifts
```

- This causes the application to make a SQL query to retrieve details of the relevant products from the database:

```
SELECT * FROM products
WHERE category = 'Gifts' AND released = 1
```

SQL injection example 1 – Retrieving hidden data

```
SELECT * FROM products
WHERE category = ‘Gifts’ AND released = 1
```

- This SQL query asks the database to return:
  - all details (`*`)
  - from the `products` table
  - where the `category` is `Gifts`
  - and `released` is `1`.
- The restriction `released = 1` is being used to hide products that are not released. We could assume for unreleased products, `released = 0`.

SQL injection example 1 – Retrieving hidden data

- an attacker can construct the following attack:

```
https://insecure-website.com/products?category=Gifts’--
```

- This results in the SQL query:

```
SELECT * FROM products
WHERE category = ‘Gifts’--’ AND released = 1
```

- note that -- is a comment indicator in SQL. this means the query no longer includes `AND released = 1`.

SQL injection example 1 – Retrieving hidden data

- an attack to cause the application to display all the products in any category:

```
https://insecure-
website.com/products?category=Gifts'+OR+1=1--
```

- This results in the SQL query:

```
SELECT * FROM products
WHERE category = 'Gifts'OR 1=1--' AND released = 1
```

- The modified query returns all items where either the category is Gifts, or 1 is equal to 1. As 1=1 is always true, the query returns all items.

SQL injection example 2 – Retrieving data from other database tables

- Use the **UNION** keyword to execute an additional SELECT query and append the results to the original query.
- if an application executes the following query containing the user input Gifts:

```
SELECT name, description
FROM products WHERE category = 'Gifts'
```

- An attacker can submit the input:

```
 ' UNION SELECT username, password FROM users--
```

- This causes the application to return all usernames and passwords along with the names and descriptions of products.

SQL injection example 3 – Blind SQL injection

- **Blind SQL injection** is used when a web application is vulnerable to an SQL injection but the results of the injection are not visible to the attacker.
- A web application uses a tracking cookie for analytics. When a request containing a TrackingId cookie is processed, the application uses a SQL query to determine whether this is a known user:

```
SELECT TrackingId FROM TrackedUsers
WHERE TrackingId = 'xyz'
```

SQL injection example 3 – Blind SQL injection

- The results from the query are not returned to the user. However, if you submit a recognised TrackingId, you receive a "Welcome back" message in the response.
- An attacker can retrieve information by triggering different responses conditionally.
- Suppose that two requests are sent containing the following TrackingId cookie values in turn: `…xyz' AND '1'='1`

  `…xyz' AND '1'='2`

```
SELECT TrackingId FROM TrackedUsers
WHERE TrackingId = ‘xyz’ AND ‘1’=‘1’

SELECT TrackingId FROM TrackedUsers
WHERE TrackingId = ‘xyz’ AND ‘1’=‘2’
```

SQL injection example 3 – Blind SQL injection

- The first value causes the query to return results, because the injected `AND '1'='1` condition is true. As a result, the "Welcome back" message is displayed.
- The second value causes the query to return no results, because the injected condition is false. The "Welcome back" message is not displayed.

- This allows us to determine the answer to any single injected condition, and extract data one piece at a time.

SQL injection

- SQL injection attacks may cause severe consequences for an organization.
- An example of high-profile data breach: 2015 TalkTalk data breach.
- In 2015, British telecommunications provider TalkTalk experienced a SQL injection attack. As a result, personal and banking details of around 160,000 customers were illegally accessed.
- The direct and indirect costs of the attack for TalkTalk was estimated at £77 million. TalkTalk was fined £400,000 by the ICO for negligence on securing client data.

Preventing SQL injection

- Using prepared statements (parameterized statements)
  - Instead of concatenating user input in the statement, use a placeholder(parameter) to take user input.
  - A placeholder can only store a value of the given type; hence SQL injection would be treated as a strange (invalid) parameter value.
- Escaping all user supplied input
  - E.g. a single quote (') in an input string should be prepended with a backslash (\) so that the database understands the single quote is part of a given string, rather than its terminator.

Preventing SQL injection

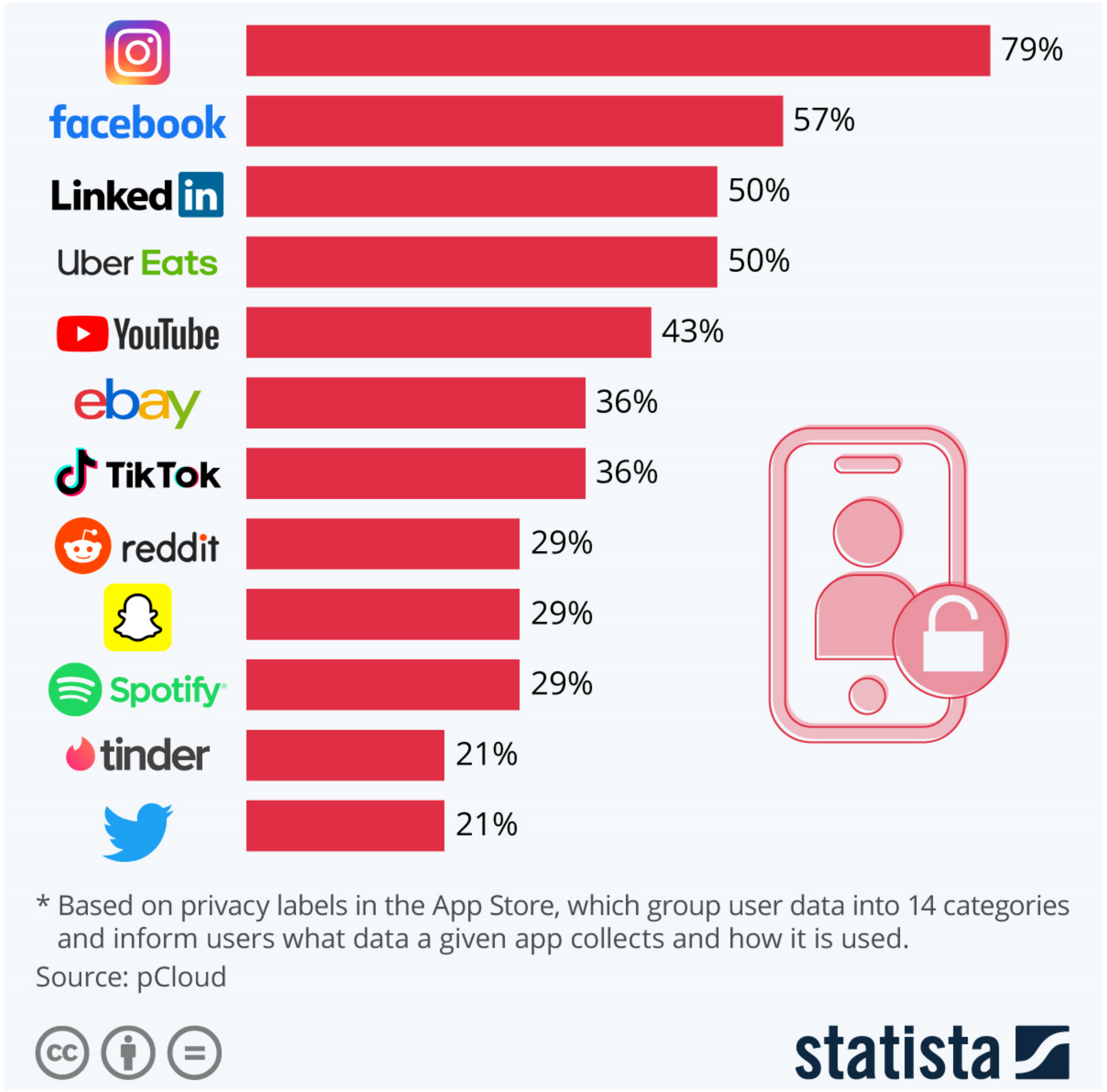
- Pattern check
  - Parameters can be checked to determine if the value is a valid representation of the given type. Strings can be check to adhere to a specific pattern (e.g. dates, phone numbers)
- Database permissions
  - Limiting the permissions on the database login used by the web application to only what is needed
- Detecting SQL injection vulnerabilities, manually or using web vulnerability scanner.

Data Ethics

Data Ethics

- Data ethics is a branch of ethics that evaluates data practices that have the potential to adversely impact people and society.
- It involves the moral obligations of gathering, protecting, and using data, especially personally identifiable information (PII) and how it affects individuals.

- Data is a valuable asset that can help innovation to improve the lives of people. However, it is also a resource that many organizations are not protecting or using ethically.



* Based on privacy labels in the App Store, which group user data into 14 categories and inform users what data a given app collects and how it is used.

Source: pCloud

Percentage of personal data categories shared with third parties by select iOS apps*

Reference: Statista 2021

Principles of Data Ethics

- Ownership
- Transparency
- Data Privacy
- Accountability

Principles of Data Ethics - Ownership

- The first principle of data ethics is that an individual has ownership over their personal information.
- An individual should be able to make meaningful choices about personal data.
- Consent should be obtained before you collect someone’s personal data.

Principles of Data Ethics - Transparency

- Transparency refers to clear communication of what data will be gathered, whether (and how) it will be stored, how the data may be used, and who it might be shared with.
- While terms and conditions or policies documents can be long, and almost require legal expertise, users should understand how the company will use their information.

Principles of Data Ethics – Data Privacy

- Data privacy is about ensuring data subjects' privacy.
- Personally identifiable information (PII) refers to any information linked to a person's identity, including:
  - Full name, date of birth, address, phone number, credit card information, bank account number, passport number, etc
- To protect individuals' privacy, database security mechanisms are essential, so personal data doesn't end up in the wrong hands.

Principles of Data Ethics – Accountability

- Accountability refers to an organization taking responsibility for what happens to an individual’s data after collecting it.
- From data leaks to sales to a third party, companies need to accept accountability for the damage their processes can cause users.

Relevant legislation for Data Ethics

- GDPR (General Data Protection Regulation)
  - The GDPR is a European Union regulation on information privacy in the European Union (EU) and the European Economic Area (EEA).
  - Become effective on 25 May 2018.
  - It also governs the transfer of personal data outside the EU and EEA.
  - The GDPR's goals are to enhance individuals' control and rights over their personal information and to simplify the regulations for international business.

For more information, check https://gdpr.eu/what-is-gdpr/

What have we learned?

- Database security
- Threats related to database
- Database security measures:
  - Authorization, Access controls, Views, Backup and recovery, Integrity, Encryption
- SQL injection attack
- Preventing SQL injection
- Data Ethics: concept and importance
- Principles of Data Ethics
- Relevant legislation for Data Ethics