What is Data Independence?

Data Independence is defined as a property of DBMS that helps you to change the Database schema at one level of a database system without requiring to change the schema at the next higher level. Data independence helps you to keep data separated from all programs that make use of it.

You can use this stored data for computing and presentation. In many systems, data independence is an essential function for components of the system.

Types of Data Independence

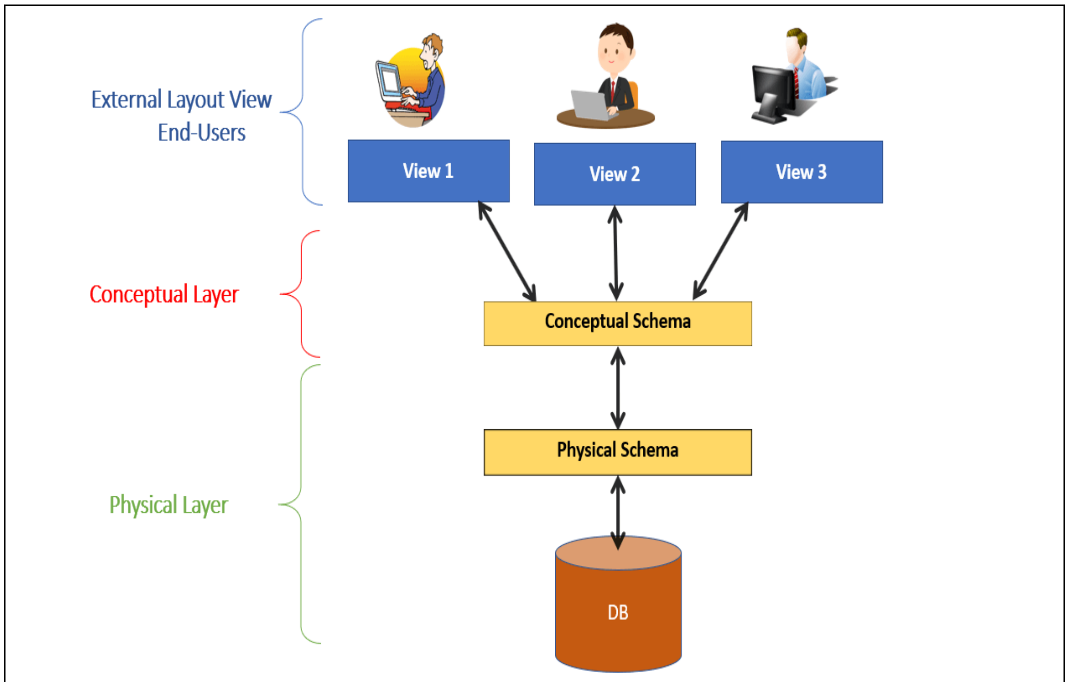
There are two types of data independence

(i) Physical data independence
(ii) Logical data independence.

Levels of Database

Before we learn Data Independence, a refresher on Database Levels is important. The database has 3 levels as shown in the diagram below

(i) Physical/Internal
(ii) Conceptual
(iii) External



Levels of Architecture Diagram

Consider an Example of a University Database. At the different levels this is how the implementation will look like:

Type of Schema	**Implementation**
External Schema	**View 1**: Course info(cid:int,cname:string) **View 2**: studeninfo(id:int. name:string)

Conceptual Shema	Students(id: int, name: string, login: string, age: integer)
	Courses(id: int, cname.string, credits:integer)
	Enrolled(id: int, grade:string)

Physical Schema	• Relations stored as unordered files. • Index on the first column of Students.

Physical Data Independence
Physical data independence helps you to separate conceptual levels from the internal/physical levels. It allows you to provide a logical description of the database without the need to specify physical structures. Compared to Logical Independence, it is easy to achieve physical data independence.
With Physical independence, you can easily change the physical storage structures or devices with an effect on the conceptual schema. Any change done would be absorbed by the mapping between the conceptual and internal levels. Physical data independence is achieved by the presence of the internal level of the database and then the transformation from the conceptual level of the database to the internal level.

Examples of changes under Physical Data Independence

Due to Physical independence, any of the below change will not affect the conceptual layer.

- Using a new storage device like Hard Drive or Magnetic Tapes
- Modifying the file organization technique in the Database
- Switching to different data structures.
- Changing the access method.
- Modifying indexes.
- Changes to compression techniques or hashing algorithms.
- Change of Location of Database from say C drive to D Drive

Logical Data Independence

Logical Data Independence is the ability to change the conceptual scheme without changing

(i) External views

(ii) External API or programs

Any change made will be absorbed by the mapping between external and conceptual levels.

When compared to Physical Data independence, it is challenging to achieve logical data independence.

Examples of changes under Logical Data Independence

Due to Logical independence, any of the below change will not affect the external layer.

(iii) Add/Modify/Delete a new attribute, entity or relationship is possible without a rewrite of existing application programs

(iv) Merging two records into one

(v) Breaking an existing record into two or more records

Difference between Physical and Logical Data Independence

Logical Data Independence	Physical Data Independence
Logical Data Independence is mainly concerned with the structure or changing the data definition.	Mainly concerned with the storage of the data.
It is difficult as the retrieving of data is mainly dependent on the logical structure of data.	It is easy to retrieve.
Compared to Logic Physical independence it is difficult to achieve logical data independence.	Compared to Logical Independence it is easy to achieve physical data independence.
You need to make changes in the Application program if new fields are added or deleted from the database.	A change in the physical level usually does not need change at the Application program level.
Modification at the logical levels is significant whenever the logical structures of the database are changed.	Modifications made at the internal levels may or may not be needed to improve the performance of the structure.
Concerned with conceptual schema	Concerned with internal schema
Example: Add/Modify/Delete a new attribute	Example: change in compression techniques, hashing algorithms, storage devices, etc

Importance of Data Independence

- Helps you to improve the quality of the data
- Database system maintenance becomes affordable
- Enforcement of standards and improvement in database security
- You don't need to alter data structure in application programs

- Permit developers to focus on the general structure of the Database rather than worrying about the internal implementation
- It allows you to improve state which is undamaged or undivided
- Database incongruity is vastly reduced.
- Easily make modifications in the physical level is needed to improve the performance of the system.

What is database security?

Database security refers to the collective measures used to protect and secure a database or database management software from illegitimate use and malicious cyber threats and attacks.

Database security procedures are aimed at protecting not just the data inside the database, but the database management system and all the applications that access it from intrusion, misuse of data, and damage.

It is a broad term that includes a multitude of processes, tools and methodologies that ensure security within a database environment.
Database security covers and enforces security on all aspects and components of databases. This includes:

- Data stored in database.
- Database server.
- Database management system (DBMS).
- Other database workflow applications.

Database security is generally planned, implemented and maintained by a database administrator and or other information security professional.

Some of the ways database security is analyzed and implemented include:

- Restricting unauthorized access and use by implementing strong and multifactor access and data management controls.
- Load/stress testing and capacity testing of a database to ensure it does not crash in a distributed denial of service (DDoS) attack or user overload.
- Physical security of the database server and backup equipment from theft and natural disasters. Regular data backups can be planned as part of a database security protocol, and multiple copies can be stored off-site to provide redundancy and emergency recovery.
- Reviewing the existing system for any known or unknown vulnerabilities and defining and implementing a road map/plan to mitigate them.
- Data encryption can provide an additional layer of security to protect the integrity and confidentiality of data.

Enforcing adequate database security practices is vital for any organizations for a variety of reasons. These include:

- **Ensuring business continuity:** Many enterprises cannot operate until the breach is resolved.
- **Minimizing financial damage:** Once a breach occurs, an organization must sustain significant financial costs to communicate the breach to all its customers, manage the crisis, repair or update the affected systems and hardware, pay for investigative activities, etc.
- **Loss of intellectual property:** If a database is accessed, there’s a chance that a company’s trade secrets, proprietary procedures, and other forms of intellectual property are stolen or

exposed. In some instances, this means the complete loss of any competitive edge maintained by that organization.

- **Brand reputation damage:** Once a breach is notified to the customer base, partners and customers may lose faith in the organization's ability to protect their data. The brand's reputation will suffer, and many might decide not to buy that organization's products or services anymore.
- **Penalties and fines:** Organizations must be compliant with a large number of regulations, such as those in the General Data Protection Regulation (GDPR), Payment Card Industry Data Security Standard (PCI DSS), Health Insurance Portability and Accountability Act (HIPAA), and more. If a data breach occurs because the organization failed to comply with these regulations, fines and penalties can be very severe, in some cases even exceeding several million dollars per violation.

Database security controls

These different security controls help to manage the circumventing of security protocols.

System hardening and monitoring

The underlying architecture provides additional access to the DBMS. It is vital that all systems are fixed consistently, hardened using known security configuration standards, and monitored for access, including insider threats.

DBMS configuration

It is critical that the DBMS be properly configured and hardened to take advantage of security features and limit privileged access that may cause a misconfiguration of expected security settings. Monitoring the DBMS configuration and ensuring proper change control processes helps ensure that the configuration stays consistent.

Authentication

Database security measures include authentication, the process of verifying if a user's credentials match those stored in your database, and permitting only authenticated users' access to your data, networks, and database platform.

Access

A primary outcome of database security is the effective limitation of access to your data. Access controls authenticate legitimate users and applications, limiting what they can access in your database. Access includes designing and granting appropriate user attributes and roles and limiting administrative privileges.

Database auditing

Monitoring (or auditing) actions as part of a database security protocol delivers centralized oversight of your database. Auditing helps to detect, deter, and reduce the overall impact of unauthorized access to your DBMS.

Backups

A data backup, as part of your database security protocol, makes a copy of your data and stores it on a separate system. This backup allows you to recover lost data that may result from hardware failures, data corruption, theft, hacking, or natural disasters.

Encryption

Database security can include the secure management of encryption keys, protection of the encryption system, management of a secure, off-site encryption backup, and access restriction protocols.

Application security

Database and application security framework measures can help protect against common known attacker exploits that can circumvent access controls, including SQL injection.

Why is database security important?

Protection the data of any company is utmost importance. Database security can lookout against a compromise of your database, which can lead to financial loss, reputation damage, consumer confidence disintegration, brand erosion, and non-compliance of government and industry regulation.

Database security defend against a countless of security threats and can help protect any enterprise from:

- Deployment failure
- Excessive privileges
- Privilege abuse
- Platform vulnerabilities
- Unmanaged sensitive data
- Backup data exposure
- Weak authentication
- Database injection attacks

Database Auditing

The general database auditing concept is about tracking the use of database records and authority. When you audit a database, each operation on the data can be monitored and logged to an audit trail, including information about which database object or data record was touched, what account performed the action and when the activity occurred.

Database Audit Trail

(i) "Who exactly accessed or changed data within our systems?"
(ii) "When was that data access or when was it changed?"
(iii)"How did a specific user gain access to the data?"
(iv)"Was the change to the database table approved before the change was made?"
(v) "Are the privileged users abusing their unlimited access?"

Security Issues and the Insider Threat

A lot of this is about security threats. Every day, every year, every month, internal and external forces are actively compromising company data (accidentally and deliberately). Some of the most serious threats come from current employees with authorized access. Some want to settle a score while others are opportunistically looking to make extra money selling private information. An audit trail helps uncover this type of activity.