**The following principles are based on the**[**Agile Manifesto**](https://www.agilealliance.org/agile101/the-agile-manifesto/)**.**

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity--the art of maximizing the amount of work not done--is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

# single sign-on (SSO)

Single sign-on (SSO) is a [session](http://searchsoa.techtarget.com/definition/session) and user [authentication](http://searchsecurity.techtarget.com/definition/authentication) service that permits a user to use one set of [login](http://searchsecurity.techtarget.com/definition/logon) credentials (e.g., name and password) to access multiple applications. The service authenticates the end user for all the applications the user has been given rights to and eliminates further prompts when the user switches applications during the same session. On the back end, SSO is helpful for logging user activities as well as monitoring user accounts.

In a basic web SSO service, an [agent](http://whatis.techtarget.com/definition/agent-intelligent-agent) module on the application server retrieves the specific authentication credentials for an individual user from a dedicated SSO [policy server](http://searchsecurity.techtarget.com/definition/policy-server), while authenticating the user against a user [repository](http://searchoracle.techtarget.com/definition/repository) such as a lightweight directory access protocol ([LDAP](http://searchmobilecomputing.techtarget.com/definition/LDAP)) directory.

Some SSO services use protocols such as [Kerberos](http://searchsecurity.techtarget.com/definition/Kerberos) and the security assertion markup language (SAML). [SAML](http://searchfinancialsecurity.techtarget.com/definition/SAML) is an [XML](http://searchsoa.techtarget.com/definition/XML) standard that facilitates the exchange of user authentication and [authorization](http://searchsoftwarequality.techtarget.com/definition/authorization) data across secure [domains](http://searchsoa.techtarget.com/definition/domain). SAML-based SSO services involve communications between the user, an identity provider that maintains a user directory, and a service provider. When a user attempts to access an application from the service provider, the service provider will send a request to the identity provider for authentication. The service provider will then verify the authentication and log the user in. The user will not have to log in again for the rest of his session. In a Kerberos-based setup, once the user credentials are provided, a ticket-granting ticket ([TGT](http://searchenterprisedesktop.techtarget.com/definition/authentication-ticket)) is issued. The TGT fetches service tickets for other applications the user wishes to access, without asking the user to re-enter credentials.

Although single sign-on is a convenience to users, it present risks to enterprise security. An attacker who gains control over a user's SSO credentials will be granted access to every application the user has rights to, increasing the amount of potential damage. In order to avoid malicious access, it's essential that every aspect of SSO implementation be coupled with [identity governance](http://searchsecurity.techtarget.com/definition/identity-governance). Organizations can also use two factor authentication ([2FA](http://searchsecurity.techtarget.com/definition/two-factor-authentication)) or multifactor authentication ([MFA](http://searchsecurity.techtarget.com/definition/multifactor-authentication-MFA)) with SSO to improve security.

* **DevOps value is derived in the core principles that started the movement:**
* **Iterative.**
* **Incremental.**
* **Continuous.**
* **Automated.**
* **Self-service.**
* **Collaborative.**
* **Holistic**

Define Normalization and De- Normalization.

**Normalization**  
- It is the process of organizing data into related table.  
- To normalize database, we divide database into tables and establish relationships between the tables.  
- It reduces redundancy. It is done to improve performance of query.  
- Steps of normalization:  
  
**First Normal form**  
-Entities of the table must have unique identifier or entity key.   
**Second Normal Form**  
- All the attributes of the table must depend on the entity key for that entity.   
**Third Normal Form**  
-All attributes that are not part of the key must not depend on any other non-key attributes.  
  
**De-normalization**  
The process of adding redundant data to get rid of complex join, in order to optimize database performance. This is done to speed up database access by moving from higher to lower form of normalization.

Define Normalization and De- Normalization.

**Normalization** is the process of reducing data redundancy and maintains data integrity. This is performed by creating relationships among tables through primary and foreign keys. Normalization procedure includes 1NF, 2NF, 3NF, BCNF, and then the data is normalized.  
  
**Denomalization** on the contrary is the process of adding redundant data to speed up complex queries involving multiple tables JOINS. One might just go to a lower form of Normalization to achieve Denormalization and better performance. Data is included in one table from another in order to eliminate the second table which reduces the number of JOINS in a query and thus achieves performance.

Denormalization is a strategy used on a previously-normalized database to increase performance. The idea behind it is to add redundant data where we think it will help us the most. We can use extra attributes in an existing table, add new tables, or even create instances of existing tables. The usual goal is to decrease the running time of select queries by making data more accessible to the queries or by generating summarized reports in separate tables. This process can bring some new problems, and we’ll discuss them later.

A normalized database is the starting point for the denormalization process. It’s important to differentiate from the database that has not been normalized and the database that was normalized first and then denormalized later. The second one is okay; the first is often the result of bad database design or a lack of knowledge.

**Database vs Schema**

A system intended for easily organizing, storing and retrieving large amounts of data, is called a database. In other words, a database holds a bundle of organized data (typically in digital form) for one or more users. Databases, often abbreviated DB, are classified according to their content, such as document-text, bibliographic and statistical. On the other hand, database schema is the formal description of the organization and the structure of data in the database. This description includes the definitions of tables, columns, data types, indexes and much more.

**Database**

A database may contain different levels of abstraction in its architecture. Typically, the three levels: external, conceptual and internal make up the database architecture. External level defines how the users view the data. A single database can have multiple views. The internal level defines how the data is physically stored. The conceptual level is the communication medium between internal and external levels. It provides a unique view of the database regardless of how it is stored or viewed. There are several types of databases such as Analytical database, Data warehouses and Distributed databases. Databases (more correctly, relational databases) are made up of tables and they contain rows and columns, much like spreadsheets in Excel. Each column corresponds to an attribute, while each row represents a single record. For example, in a database, which stores employee information of a company, the columns could contain employee name, employee Id and salary, while a single row represents a single employee. A DBMS (Database Management System) is used to manage all the databases in a database system. Typically, the structure of a database is too complex to handle without a DBMS. Popular DBMS products are Microsoft SQL Server, MySQL, DB2, Oracle, and Microsoft Access.

**Schema**

A database schema of a database system describes the structure and the organization of data. A formal language supported by the Database Management System is used to define the database schema. Schema describes how the database will be constructed using its tables. Formally, schema is defined as the set of formula that imposes integrity constraints on the tables. Furthermore, the database schema will describe all tables, column names and types, indexes, etc. There are three types of schema called the conceptual schema, logical schema and physical schema. Conceptual schema describes how concepts and relationships are mapped. Logical schema defines how entities, attributes and relations are mapped. Physical schema is a specific implementation of the aforementioned logical schema.

**What is the difference between Database and Schema?**

As a summery, a database is a collection of organized data, while database schema describes the structure and organization of data in a database system. The database holds the records, fields and cells of data. The database schema describes how these fields and cells are structured and organized and what types of relationships are mapped between these entities. Understandably, the schema of a database keeps constant once created, while the actual data in the database tables may change all the time.

**Denormalization**

In a [relational database](http://searchsqlserver.techtarget.com/definition/relational-database), denormalization is an approach to speeding up read performance (data retrieval) in which the administrator selectively adds back specific instances of redundant data after the data structure has been [normalized](http://searchsqlserver.techtarget.com/definition/normalization). A denormalized database should not be confused with a [database](http://searchsqlserver.techtarget.com/definition/database) that has never been normalized.

During normalization, the database designer stores different but related types of data in separate logical tables called relations. When a [query](http://searchsqlserver.techtarget.com/definition/query) combines data from multiple tables into a single result table, it is called a join. Multiple joins in the same query can have a negative impact on performance. **Introducing denormalization and adding back a small number of redundancies can be a useful for cutting down on the number of joins.**

After data has been duplicated, the database designer must take into account how multiple instances of the data will be maintained. One way to denormalize a database is to allow the database management system ([DBMS](http://searchsqlserver.techtarget.com/definition/database-management-system)) to store redundant information on disk. This has the added benefit of ensuring the consistency of redundant copies. Another approach is to denormalize the actual logical data design, but this can quickly lead to inconsistent data. Rules called constraints can be used to specify how redundant copies of information are synchronized, but they increase the complexity of the database design and also run the risk of impacting write performance.