**Difference between Zend session and Zend registry.**

**Zend\_Registry** is used to store objects/values for the current request. In short, anything that you commit to Registry in index.php can be accessed from other controllers/actions (because EVERY request is first routed to the index.php bootstrapper via the .htaccess file). Config parameters and db parameters are generally prepped for global use using the Zend\_Registry object.

**Zend\_Session\_Namespace** actually uses PHP sessions. Data stored using Zend\_Session can be accessed in different/all pages. So, if you want to create a variable named ‘UserRole’ in the /auth/login script and want it to be accessible in /auth/redirect, you would use Zend\_Session.

***For Example***

if you have a DB object stored in the registry, and when you are access a page say login page, this particular DB object will be accessible to any of the files used in that page (to construct), meaning global access (same keyword in PHP [Scope Variable](http://in2.php.net/manual/en/language.variables.scope.php#language.variables.scope.global)).

Whereas, Session scope means you can access anywhere on any page until you close your browser (until your session get’s destroyed).

**Zend\_Session** is for working with the session extension in PHP.This would be for tracking loggin in user(s), etc.

**Zend\_Registry** is used to store objects / resources / etc in at runtime. The idea is that you want to use maybe 1 config object, or 1 database throughout your entire application. So what you would do, when you create these objects, you would assign them to the registry.

**Benefits using composer**

With Composer we get the following advantages:

1. The dependencies required by the package we are pulling in are automatically taken care by Composer itself, leaving we free to focus on the programming instead of dependency management.
2. When the package we are using gets a new version, a simple composer update will do everything for us, without ever needing to do any file management manually.
3. With Composer we get a centralized autoload.php file which also be optimized with Composer. It loads everything we need and all we do is include one file. The optimized version is super fast! Simply do composer dump-autoload -o
4. We can use psr-4 namespaces to load a specific path on our application and have it be included in the autoloader file. Then we can simple use the namespace and it's available application wise!

**2nd highest salary + MySQL limit in detail**

**If else and switch case query in MySQL**

**Clone function in jQuery**

**Traversing parent child in jQuery**

**On function in jQuery**

**How indexes work in MySQL**

**How session work in PHP**

**What is abstract? What is the benefit of using abstract class?**

**Benefit of orm?**

**How Zend framework protect from MySQL injection?**

**Advantages and disadvantages of views in MySql?**

A database view is a virtual table or logical table which is defined as a [SQL SELECT query](http://www.mysqltutorial.org/mysql-select-statement-query-data.aspx) with [joins](http://www.mysqltutorial.org/mysql-inner-join.aspx). Because a database view is similar to a database table, which consists of rows and columns, so you can query data against it. Most database management systems, including MySQL, allow you to [update data](http://www.mysqltutorial.org/mysql-update-data.aspx) in the underlying tables through the database view with some prerequisites.

A database view is dynamic because it is not related to the physical schema. The database system stores database views as a [SQL SELECT](http://www.mysqltutorial.org/mysql-select-statement-query-data.aspx) statement with joins. When the data of the tables changes, the view reflects that changes as well.

## Advantages of database view

The following are advantages of using database views.

* A database view allows you to simplify complex queries: a database view is defined by an SQL statement that associates with many underlying tables. You can use database view to hide the complexity of underlying tables to the end-users and external applications. Through a database view, you only have to use simple SQL statements instead of complex ones with many joins.
* A database view helps limit data access to specific users. You may not want a subset of sensitive data can be queryable by all users. You can use a database view to expose only non-sensitive data to a specific group of users.
* A database view provides extra security layer. Security is a vital part of any relational database management system. The database view provides extra security for a database management system. The database view allows you to create the read-only view to expose read-only data to specific users. Users can only retrieve data in read-only view but cannot update it.
* A database view enables computed columns. A database table should not have calculated columns however a database view should. Suppose in the orderDetails table you have quantityOrder (the number of ordered products) and priceEach (price per product item) columns. However, the orderDetails table does not have a computed column to store total sales for each line item of the order. If it has, the database schema would not be a good design. In this case, you can create a computed column named total , which is a product of quantityOrder and priceEach to represent the computed result. When you query data from the database view, the data of the computed column is calculated on fly.
* A database view enables backward compatibility. Suppose you have a central database, which many applications are using it. One day, you decide to redesign the database to adapt with the new business requirements. You remove some tables and create new tables, and you don’t want the changes affect other applications. In this scenario, you can create database views with the same schema as the legacy tables that you will remove.

## Disadvantages of database view

Besides the advantages above, there are several disadvantages of using database views:

* Performance: querying data from a database view can be slow especially if the view is created based on other views.
* Tables dependency: you create a view based on underlying tables of the a database. Whenever you change the structure of those tables that view associated with, you have to change the view as well.

**Advantages and disadvantages of indexes in MySql?**

##### Advantages of MySQL Indexes

1- Indexes make search queries much faster.

2- Indexes like primary key index and unique index help to avoid duplicate row data.

3- Full-text indexes in MySQL, users have the opportunity to optimize searching against even large amounts of text located in any field indexed as such.

##### Disadvantages of MySQL indexes

Actually a separate file created when a new index created on the table column. that file stored only the field you’re interested in sorting on. So when we create index, it takes up disk space. but because of creating index on every column in every possible combination, the index file would grow much more quickly than the data file. In the case when a table is of large table size, the index file could reach the operating system’s maximum file size.

The index also slow down the speed of writing queries, such as INSERT, UPDATE and DELETE. AS MySQL has to internally maintain the pointers to the inserted rows in the actual data file, so there is a performance price to pay in case of above said writing queries because every time a record is changed, the indexes must be updated.

So Indexes are important to speed in large MySQL databases. it doesn’t matter how small your table, a 100000-row table scan will never be fast. So If you have a site with a 100000-row table, you should really spend time analyzing possible indexes and possibly consider rewriting queries to optimize your application.

**What are composite indexes in Mysql?**

**How to parse JSON data in console?**

**Strstr, substr and strpos in php**

**How to define constant in php?**

define("GREETING", "Welcome to W3Schools.com!");  
echo GREETING;

**How to use ‘explain’ for query optimization in mysql?**