A database view is nothing more than SQL statement that is stored in the database with an associated name. A view is actually a composition of a table in the form of a predefined SQL query. It is essentially a virtual table defined inside the database whose contents are defined by the query.

To the end user, the view appears just like a real table, with a set of named columns and rows of data. SQL creates the illusion of the view by giving the view a name like a table name and storing the definition of the view in the database.

Advantages of database views

Security

Each user can be given permission to access the database only through a small set of views that contain the specific data the user is authorized to see, thus restricting the user's access to stored data.

Views allow the database administrator (DBA) to pull fields of interest from tables of interest and return a coherent data set useful to some specific user or application

Aggregated data

Views can act as aggregated tables, where the database engine aggregates data (sum, average etc.) and presents the calculated results as part of the data. This is useful for all kinds of reporting functionality.

Simplicity

If you have a complex select with lots of joins, you can implement it in a view and simply call the view without need to consider all these joins

Materialized views

Sometimes when table has huge amount of data and we need to perform aggregation by joining multiple of these tables then performance becomes an issue, in this case materialized views come to the rescue. Materialized views are also the logical view of our data-driven by the select query but the result of the query will get stored in the table or disk, also the definition of the query will also store in the database.  
  
When we see the performance of Materialized view it is better than normal View because the data of materialized view will be stored in table and table may be [indexed](http://java67.blogspot.sg/2012/10/difference-between-clustered-vs-nonclustered-index-sql-database.html) so faster for joining also joining is done at the time of materialized views refresh time so no need to every time fire join statement as in case of view.

Difference between View and materialized view is that

In Views query result is not stored in the disk or database but Materialized view allow to store the query result in disk or table.

In case of View we always get latest data but in case of Materialized view we need to refresh the view for getting latest data

In case of Materialized view we need an extra trigger or some automatic method so that we can keep Materialized Views refreshed, this is not required for views in the database

Databases supporting materialized views

Oracle – Natively supported, were first implemented in this RDBMS.

Postgres - In newer version of [PostgreSQL](https://en.wikipedia.org/wiki/PostgreSQL), natively support materialized views, it is not auto-refreshed, and is populated only at time of creation. It may be refreshed later manually using SQL command.

SQL Server – In implements via concept of indexed views.

MySQL – Does not natively support, alternate ways to achieve the same.

[IBM DB2](https://en.wikipedia.org/wiki/IBM_DB2), they are called "materialized query tables"