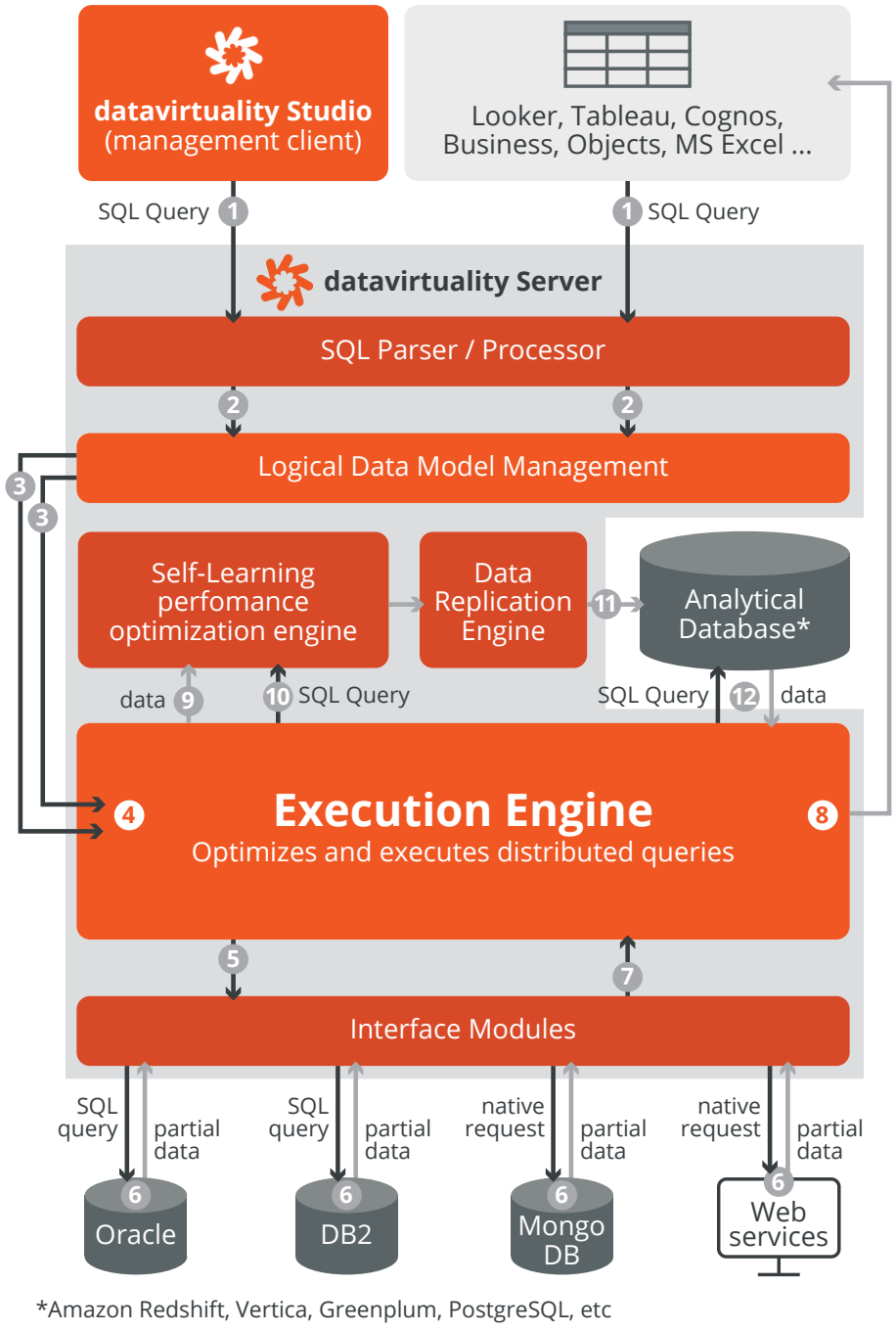
datavirtuality use cases



*Amazon Redshift, Vertica, Greenplum, PostgreSQL, etc

Logical data warehouse

Datavirtuality is a Java-based software solution that enables quick and flexible integration of heterogeneous data sources to a single logical data warehouse.

Users are able to immediately see all connected data sources as part of datavirtuality and can then query them all using a single SQL query. That query is normally generated by a business intelligence (BI) tool like Tableau, Cognos, or Looker and sent to the datavirtuality server (step 1). Once users finished rapid prototyping, they can define the global virtual logical model over all connected data sources, in which they can define KPIs, cleanse data and transform source data to a business language in a centralized manner.

Regardless if the query is directed to the logical layer or addresses directly the data sources, the datavirtuality server parses the query using a built-in SQL processor and is able to identify which logical data model parts are involved (2,3).

In the next step, the query plan is built and optimized for effective distributed execution across the connected data sources (4). In the actual execution, the query is split across the interface modules (5) that are responsible for getting partial data from the appropriate data sources (6) and transforming it to the relational format. The partial results coming from the interface modules (7) are then further processed (joined, aggregated, projected, etc.) by the execution engine (4) and returned (8) to the requesting BI tool.

In addition, the self-learning optimization engine analyses data and the data sources' (9) statistics as well as the relevance of the query (10) and recognizes performance bottlenecks.

Furthermore, it eliminates bottlenecks by automatically creating and managing the physical data structures in the analytic storage (an external source like PostgreSQL, Oracle, SAP Hana, etc.) by utilizing the data replication component (11). Data can be kept up-to-date by scheduling the replicating on a regular basis, and the amount of replication data can be reduced by running incremental replications, capturing change data from databases, for example.

Once data is physically available in the analytic database, all BI queries are automatically redirected (12) to that database without rewriting the reports.

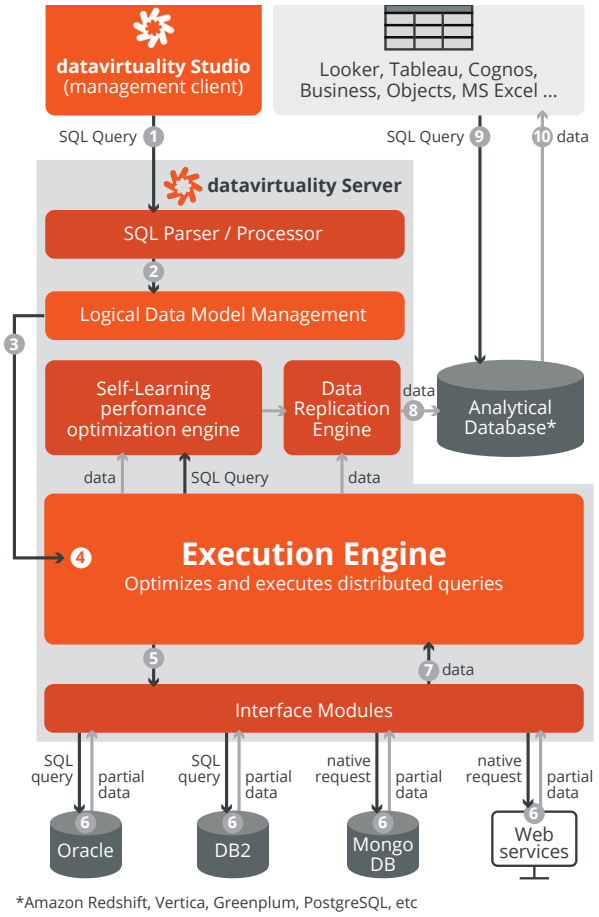
data virtuality
any data. any system. any time.

US OFFICE
1355 Market Street
Suite #488
San Francisco, CA 94103
Telephone: +1 650 285 9270

GERMAN OFFICE
Katharinenstrasse 15
04109 Leipzig | Germany
Telefon: +49 341 26437217
Fax: +49 322 239 44703

*To learn more about our solutions, visit **datavirtuality.com** or email **info@datavirtuality.com***

datavirtuality use cases



*Amazon Redshift, Vertica, Greenplum, PostgreSQL, etc

SQL-based data ingestion

In addition to the logical data warehouse scenario, datavirtuality can be used as an analytical data hub to ingest data into analytical storage. For example, the solution can move data from different (internal and cloud) sources into a cloud-based analytical database such as Amazon Redshift, Oracle, Vertical, Greenplum, and others.

In this use case, front end processors such as Looker or Tableau are connected directly to the analytical database (step 9), rather than the datavirtuality server.

To set up the data transfer from the particular data sources to the analytical store, the datavirtuality administrator connects to datavirtuality and issues appropriate SQL queries to create data transfer, cleansing, or transformation jobs as needed and to schedule them.

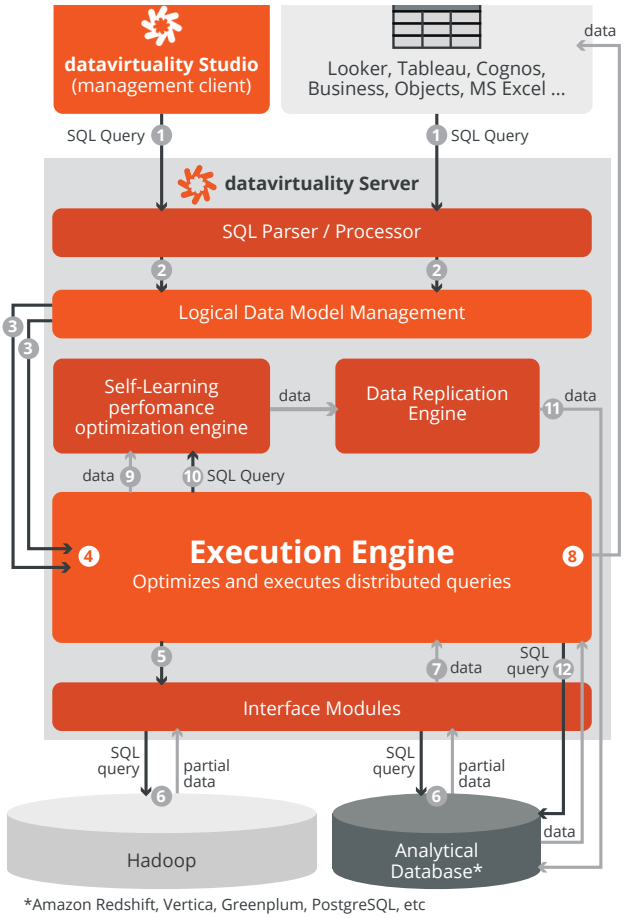
(step 1). The job queries are then processed through steps 2-7 as in the logical data warehouse scenario.

The results of the query (7) are then directly ingested into analytical storage by the execution engine with the help of the data replication component.

As soon as the data has been ingested into analytical storage, it is immediately available for the clients (9,10).

The data ingestion jobs can be set up to run on a periodic basis. In addition, datavirtuality supports a great many methods for data ingestion, including full and incremental replication, change data capture (CDC), historization (slowly changing dimensions), and batch loading.

The benefit of using datavirtuality for data ingestion is the ability to generate and manage the injestion and transformation pipelines using the familiar SQL language based on the global metadata.



*Amazon Redshift, Vertica, Greenplum, PostgreSQL, etc

Integrated analytics

As a third use case, datavirtuality enables businesses to benefit from an integrated analytics architecture. In this scenario, data originating from Hadoop-based data sources can be integrated with any analytical storage data warehouse (for example, Redshift, Exasol, or Greenplum).

As this architecture is similar to the logical data warehouse scenario, the user can instantly send federated queries to combine Hadoop and analytical data. For example, a user can run a BI report that asks for data by issuing a SQL query to the datavirtuality server (step 1). The query is then processed by the datavirtuality SQL processor (2) which recognizes the logical model parts (3). Then, the execution engine builds an execution plan for the query and optimizes it for distributed processing (4). During the query processing, the original query is split across Hadoop's and the analytical sources' interfaces (5). Partial queries are sent to the source (6) and return (7) their results to the execution engine (4) for further processing (joining, aggregating, projecting, etc.). Finally, the execution engine returns the result to the requester (8).

Analytical storage not only delivers source data but can also be used by the datavirutuality server to boost query performance. To achieve this, the self-learning performance optimization engine analyzes the data and the data sources' (9) statistics as well as the relevance of each query (10) and recognizes any performance bottlenecks. The solution automatically creates and manages the physical data structures in the connected analytical storage and eliminates any bottlenecks by utilizing the data replication component (11). Data can be kept up to date by scheduling the replicating on a regular basis. The amount of replication data can be reduced by running incremental replications or capturing change data from databases, for example. Once data is physically available in analytic storage, all original BI queries are automatically redirected (12) to it without rewriting the reports.

data virtuality
any data. any system. any time.