

Unit-4



MySQL

The MySQL logo is displayed in yellow text on a black rectangular background. Below the logo is a large, solid black downward-pointing arrow.

MySQL : Features ?

- ❑ MySQL is a relational database management system (RDBMS) based on the SQL (Structured Query Language) queries, which is one of the most popular languages for accessing and managing the records in the table.
- ❑ MySQL is open-source and free software under the GNU license. Oracle Company supports it. Some of the key features of MySQL are as follows:
 - **Easy to use** : MySQL is easy to use. Anyone having basic knowledge of SQL-syntax can build and interact with MySQL by using only a few simple SQL statements.
 - **It is secure** : MySQL consists of a solid data security layer that protects sensitive data from intruders. In MySQL all passwords are stored in encrypted format.
 - **Client/ Server Architecture**: MySQL follows the working of a client/server architecture. There is a database server (MySQL) and arbitrarily many clients (application programs), which communicate with the server; that is, they can query data, save changes, etc.
 - **Free to download**: MySQL is free to use so that we can download it from MySQL official website without any cost.
 - **It is scalable**: MySQL supports multi-threading that makes it easily scalable. It can handle almost any amount of data, up to as much as 50 million rows or more. The default file size limit is about 4 GB. However, we can increase this number to a theoretical limit of 8 TB of data.
 - **Speed**: MySQL is considered one of the very fast database languages, backed by a large number of the benchmark test.
 - **High Flexibility**: MySQL supports a large number of embedded applications, which makes MySQL very flexible.

MySQL : Features ?

- **Compatible on many operating systems**: MySQL is compatible to run on many operating systems, like Novell NetWare, Windows*, Linux*, many varieties of UNIX* (such as Sun* Solaris*, AIX, and DEC* UNIX), OS/2, FreeBSD*, and others. MySQL also provides a facility that the clients can run on the same computer as the server or on another computer (communication via a local network or the Internet).
- **Allows roll-back**: MySQL allows transactions to be rolled-back, commit, and crash recovery.
- **Memory efficiency**: Its efficiency is high because it has a very low memory leakage problem.
- **High Performance**: MySQL is faster, more reliable, and cheaper because of its unique storage engine architecture. It provides very high-performance results in comparison to other databases without losing an essential functionality of the software. It has fast loading utilities because of the different cache memory.
- **High Productivity**: MySQL uses Triggers, Stored procedures, and views that allow the developer to give higher productivity.
- **Platform Independent**: It can download, install, and execute on most of the available operating systems.

MySQL : Features ?

- **High Productivity:** MySQL uses Triggers, Stored procedures, and views that allow the developer to give higher productivity.
- **Platform Independent:** It can download, install, and execute on most of the available operating systems.
- **Partitioning:** This feature improves the performance and provides fast management of the large database.
- **GUI Support:**
 - MySQL provides a unified visual database graphical user interface tool named "MySQL Workbench" to work with database architects, developers, and Database Administrators.
 - MySQL Workbench provides SQL development, data modeling, data migration, and comprehensive administration tools for server configuration, user administration, backup, and many more. MySQL has a fully GUI supports from MySQL Server version 5.6 and higher.
- **Dual Password Support:** MySQL version 8.0 provides support for dual passwords: one is the current password, and another is a secondary password, which allows us to transition to the new password.

MySQL : installation ?

❑ Already covered during Unit-1 & 2

PostgreSQL



PostgreSQL : Features ? (website → www.postgresql.org)

- ❑ PostgreSQL (also known as Postgres), is a **free** and **open-source** relational database management system emphasizing extensibility and SQL compliance. It was originally named **POSTGRES**, referring to its origins as a successor to the Ingres database developed at the University of California, Berkeley.
- ❑ PostgreSQL features transactions with **Atomicity, Consistency, Isolation, Durability** (ACID) properties, automatically updatable views, materialized views, triggers, foreign keys, and stored procedures.
- ❑ It is designed to handle a range of workloads, from single machines to **data warehouses** or **Web services with many concurrent users**. It is the default database for **macOS Server** and is also available for **Windows, Linux, FreeBSD, and OpenBSD**.

How to Download & Install Postgres for Windows?

- ❑ To use Postgre in our machine, we need to install:
 - ✓ Postgre Database Server
 - ✓ A graphical tool to administer and manage the DB
(pgAdmin is the most popular tool GUI Tool for Postgre)
- ❑ We can individually Download PostgreSQL for Windows and install these components but coupling the settings between the DB server, and a GUI tool could be a challenge.
- ❑ Consequently, It's easier to use a bundled installer which takes care of configuration complexities.

Steps for installation

Step 1) Open your browser.

Go to <https://www.postgresql.org/download> and select Windows

Step 2) Check options.

You are given two options 1) Interactive Installer by EnterpriseDB and 2) Graphical Installer by BigSQL.

{ BigSQL currently installs pgAdmin version 3 which is deprecated.
We may therefore choose EnterpriseDB which installs the latest version 4 }

Installation – Steps...

❑ Step 3) Select PostgreSQL version.

✓ We will be prompted to desired PostgreSQL version and operating system.

✓ We shall select the latest PostgreSQL version and OS as per our environment

❑ Click the Download Button

Installation – Steps...

Step 4) Open exe file.

Once you Download PostgreSQL, open the downloaded exe and Click next on the install welcome screen.

Step 5) Update location.

Change the Installation directory if required, else leave it to default

Click Next

Installation – Steps...

Step 6) Select components.

You may choose the components you want to install in your system. You may uncheck Stack Builder

[Click Next](#)

Step 7) Check data location.

You may change the data location

[Click Next](#)

Step 8) Enter password.

Enter super user password. Make a note of it

[Click Next](#)

Step 9) Check port option.

Leave the port number default

[Click Next](#)

Installation – Steps...

Step 10) Check summary.

Check the pre-installation summary:

[Click Next](#)

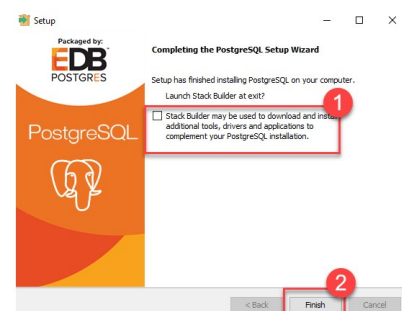
Step 11) Ready to install.

[Click the next button.](#)

Step 12) Check stack builder prompt.

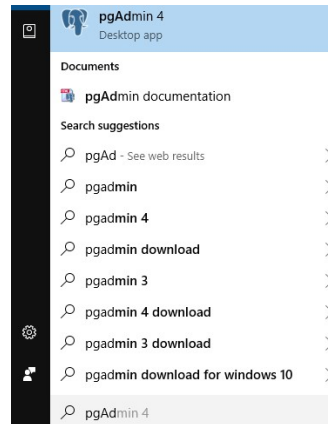
Once install is complete you will see the Stack Builder prompt

Uncheck that option.



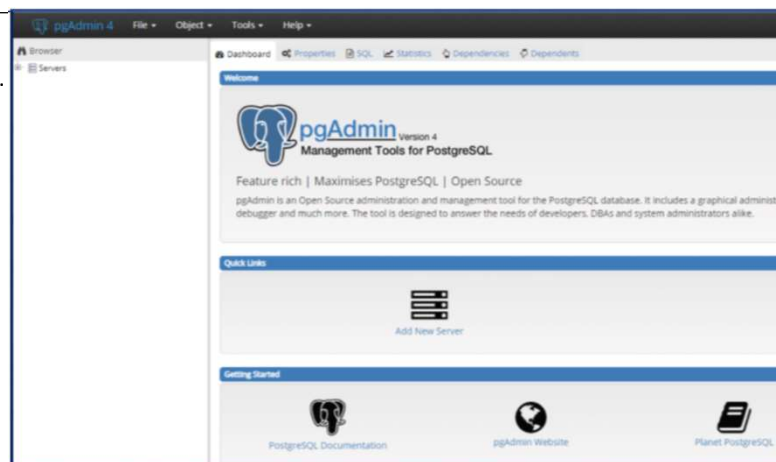
Installation – Steps...

Step 13) Launch PostgreSQL.
To launch PostgreSQL go to Start Menu and search pgAdmin 4



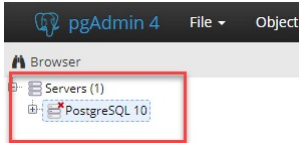
Installation – Steps...

Step 14) Check pgAdmin.
We will see pgAdmin homepage.

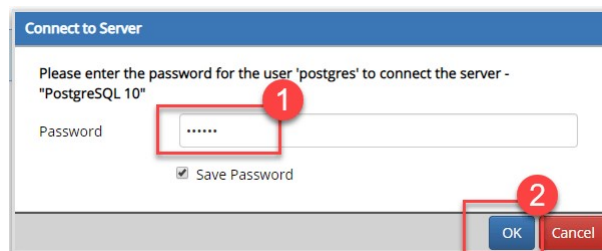


Installation – Steps...

Step 15) Find PostgreSQL 14.2
Click on Servers > PostgreSQL
14.2 in the left tree



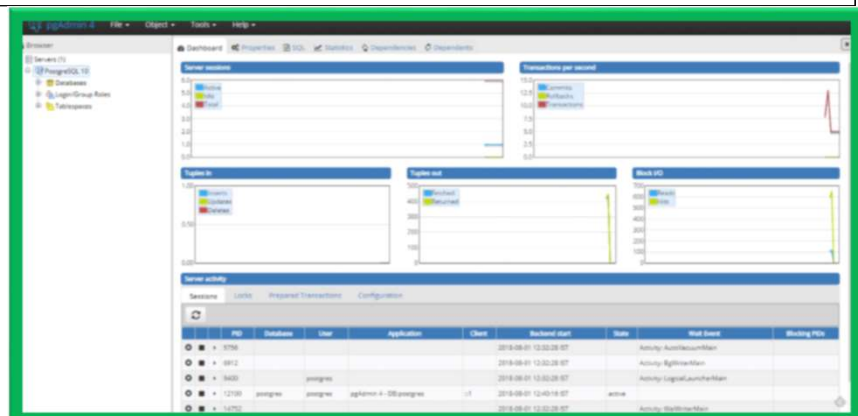
Step 16) Enter password.
Enter super user password set
during installation
Click OK



Installation – Completed

Step 17) Check
Dashboard.

We shall see the
Dashboard like ----->



SQLite



SQLite ?

- ☐ **SQLite** is an infinitely useful self-contained, portable, open source database.
- ☐ Using SQLite makes it easy to create, parse, query, modify, and transport data.
- ☐ Learning to use it interactively is a great first step toward managing it for web applications or using it through programming language libraries.

Using SQLite ?

- ❑ It's common to interact with a database through a programming language. For this reason, there are **SQLite interfaces** (or "**bindings**") for **Java**, **Python**, **PHP**, **Ruby**, **C++**, and so many others.
- ❑ However, before using these libraries, it is important to understand what's actually happening with the database engine and why our choice of a database is significant.
- ❑ To use SQLite one can invoke the **sqlite3** command so one can get familiar with the basics of how this database handles data.

Interacting with SQLite

- ❑ We can interact with SQLite using the **sqlite3** command.
- ❑ This command provides an interactive shell so that we can view and update our databases.

```
$ sqlite3
SQLite version 3.34.0 2020-12-01 16:14:00
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite>
```

- ✓ The command places us in an SQLite subshell, and so our prompt is now an SQLite prompt.
- ✓ Our usual Bash commands don't work here.
- ✓ We must use **SQLite commands**.

SQLite Commands

❑ To peruse a list of SQLite commands, we need to type **.help** from within SQLite subshell.

```
sqlite> .help
.archive ...           Manage SQL archives
.auth ON|OFF           SHOW authorizer callbacks
.backup ?DB? FILE      Backup DB (DEFAULT "main") TO FILE
.bail ON|off           Stop after hitting an error.  DEFAULT OFF
.binary ON|off         Turn BINARY output ON OR off.  DEFAULT OFF
.cd DIRECTORY          CHANGE the working directory TO DIRECTORY
[...]
```

- ✓ Some of these commands are binary, while others require unique arguments (like filenames, paths, etc.).
- ✓ These are administrative commands for your SQLite shell and are not database queries.
- ✓ Databases take queries in Structured Query Language (SQL), and many SQLite queries are the same as what we already know from the MySQL databases.
- ✓ However, data types and functions differ, so pay close attention to minor differences if you're familiar with another database.

SQLite : Features ?

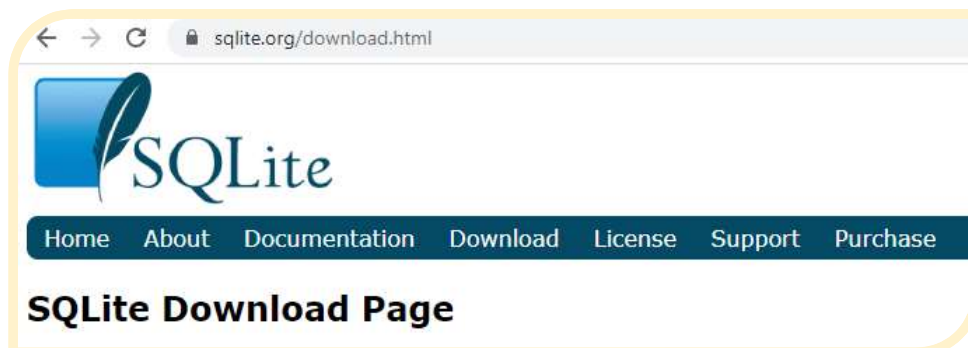
- ❑ Self-contained: no external dependencies.
- ❑ Written in ANSI-C.
- ❑ Zero-configuration - no setup or administration needed.
- ❑ A complete database is stored in a single cross-platform disk file.
- ❑ The database file format is cross-platform (Linux, Mac OS-X, Android, iOS and Windows) - you can freely copy a database between 32-bit and 64-bit systems or between big-endian and little-endian architectures.
- ❑ Great for use as an application file format.
- ❑ Available as a single ANSI-C source-code file that can be easily dropped into another project.
- ❑ Simple, easy to use API.
- ❑ Implements most of the SQL92 except some features.
- ❑ Transactions are ACID (atomic, consistent, isolated, and durable) compatible, even after system crashes and power failures.
- ❑ Comes with a standalone command-line interface (CLI) client that can be used to administer SQLite databases.

SQLite : Installation

- ❑ To install SQLite in our system we need to download the required executable file from the SQLite website based on our OS platform like Windows, Linux, etc.
- ❑ We have different options available based on OS platforms like Windows, Linux, Mac, etc. From here we can download the suitable files from available options based on our system **OS** and **configuration** (32 bit or 64 bit).

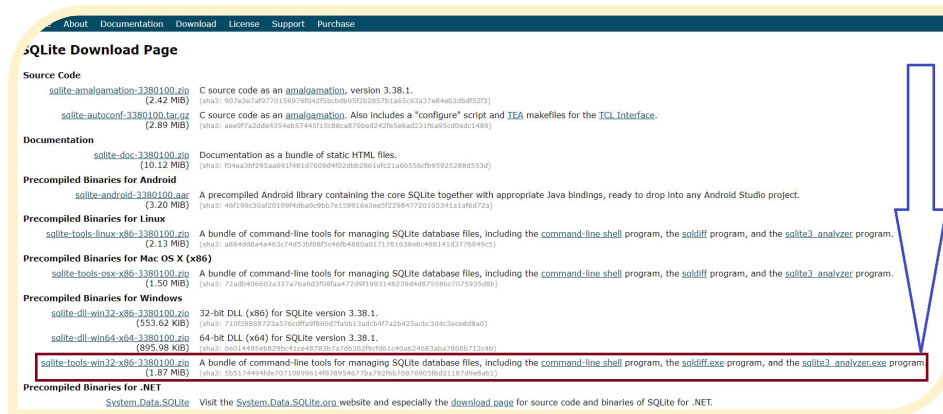
SQLite : Installation

STEP-1 → Open URL <https://www.sqlite.org/download.html>



SQLite : Installation

STEP-2 → Choose appropriate pre-compiled binaries for your intended operating system. Let us consider example for Windows:



SQLite : Installation

❑ This SQLite command-line tool will contain the following SQLite products:

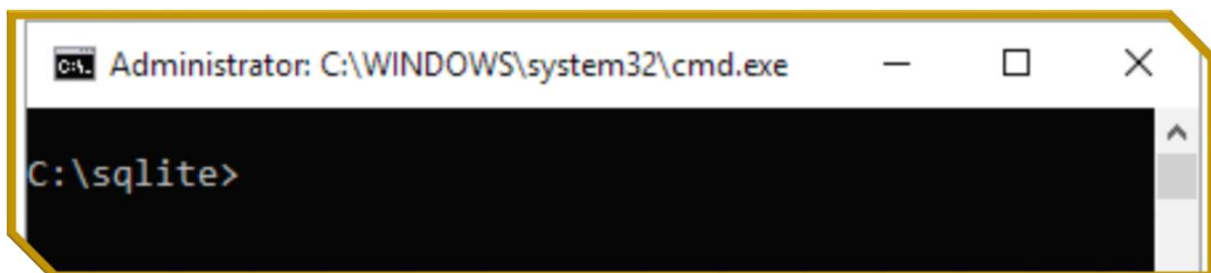
- ✓ **SQLite core:** The SQLite core contains the actual database engine and public API.
- ✓ **SQLite3 command-line tool:** The **sqlite3** application is a command-line tool that is built on top of the **SQLite core**.
- ✓ **Tcl extension:** This library is essentially a copy of the SQLite core with the Tcl bindings tacked on.
- ✓ **SQLite analyzer tool:** The **SQLite analyzer** is used to analyze database files.

- ☐ After downloading the files, we can install SQLite on the **Windows** OS system by using the following steps:
- ☐ Create a new folder “**sqlite**” in **C** or **D** or **E** drive based on your requirement like **C:\sqlite**.
- ☐ Once we create a new folder, extract the downloaded files content to a new folder (C:\sqlite), now we will be able to see **sqlite3.exe** in **C:\sqlite** folder.
- ☐ We can start using SQLite directly by opening **sqlite3.exe** from a folder or from the command line window.

Running SQLite using Command Line Window

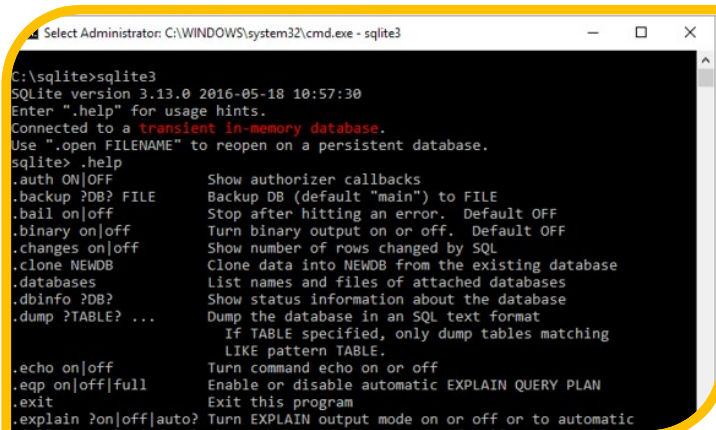
In case if we want to open from the command line window follow the below steps:

Open command line window and navigate to **C:\sqlite** folder like as shown below:



SQLite Commands Help

- ❑ We can type **.help** command from **sqlite>** prompt to see all available commands in **sqlite3** like as shown below:

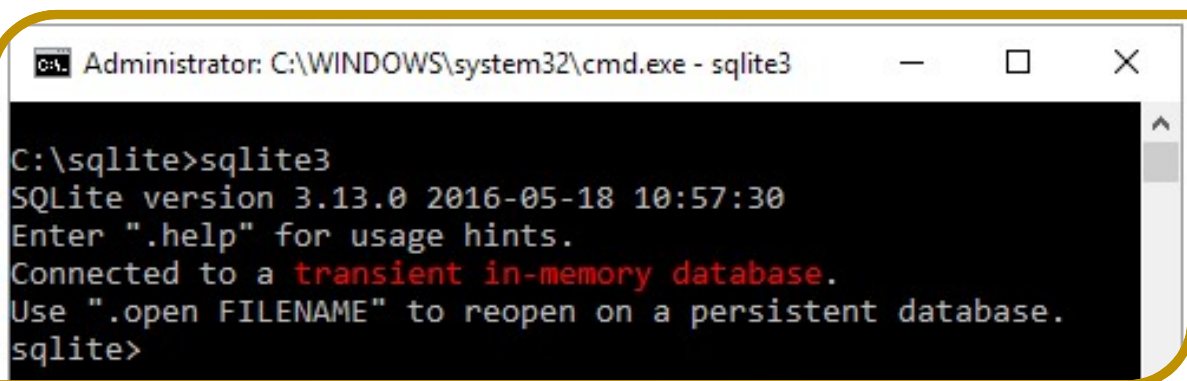


```

C:\sqlite>sqlite3
SQLite version 3.13.0 2016-05-18 10:57:30
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite>.help
.auth ON|OFF          Show authorizer callbacks
.backup ?DB? FILE     Backup DB (default "main") to FILE
.bail on|off          Stop after hitting an error. Default OFF
.binary on|off        Turn binary output on or off. Default OFF
.changes on|off        Show number of rows changed by SQL
.clone NEWDB           Clone data into NEWDB from the existing database
.databases             List names and files of attached databases
.dbinfo ?DB?          Show status information about the database
.dump ?TABLE? ...     Dump the database in an SQL text format
                      If TABLE specified, only dump tables matching
                      LIKE pattern TABLE.
.echo on|off          Turn command echo on or off
.eqp on|off|full       Enable or disable automatic EXPLAIN QUERY PLAN
.exit                Exit this program
.explain ?on|off|auto? Turn EXPLAIN output mode on or off or to automatic
  
```

Running SQLite using Command Line Window

Once we navigate to **C:\sqlite** folder enter **sqlite3** and click on enter button (in case if **sqlite3** file exists) then it will connect to the in-memory database like as shown here:



```

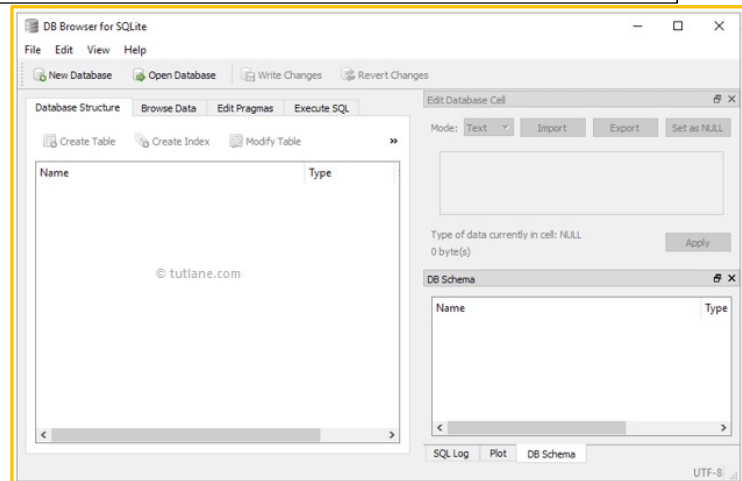
Administrator: C:\WINDOWS\system32\cmd.exe - sqlite3

C:\sqlite>sqlite3
SQLite version 3.13.0 2016-05-18 10:57:30
Enter ".help" for usage hints.
Connected to a transient in-memory database.
Use ".open FILENAME" to reopen on a persistent database.
sqlite>
  
```

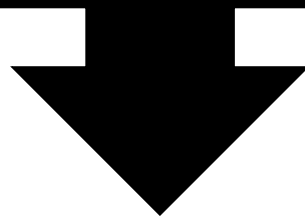
This way we can **install** and **run** SQLite directly by using the SQLite command-line tool or from the command prompt.

DB Browser for SQLite

- ✓ In case if we want a visual interface tool to work with SQLite we have a tool called [DB SQLite Browser](#).
- ✓ **DB Browser for SQLite** is a high quality, visual, open-source tool to create, design, and edit database files compatible with SQLite.



Spreadsheets



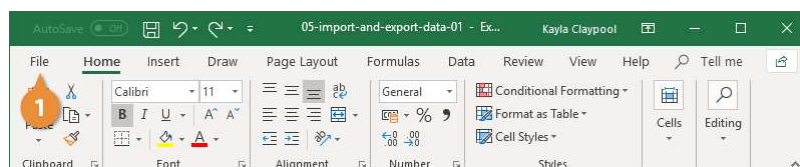
How to Import & Export Data into Excel

Excel can import and export many different file types aside from the standard .xlsx format. If your data is shared between other programs, like a database, you may need to save data as a different file type or bring in files of a different file type.

Export Data

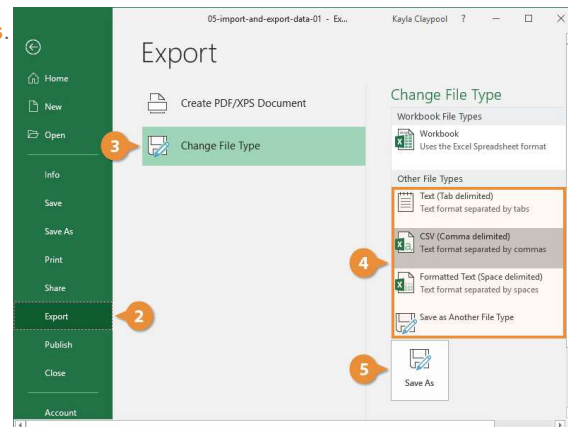
When you have data that needs to be transferred to another system, export it from Excel in a format that can be interpreted by other programs, such as a text or CSV file.

1. Click the **File** tab

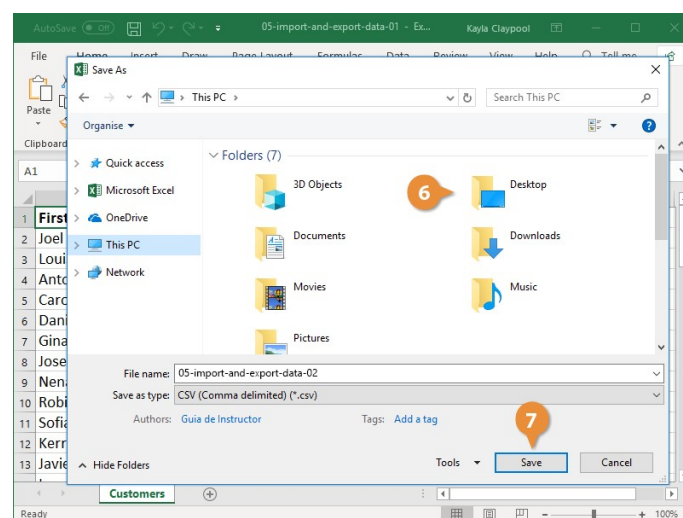


2. At the left, click **Export**.

3. Click the **Change File Type**.
 4. Under Other File Types, select a file type.
 1. **Text (Tab delimited)**: The cell data will be separated by a tab.
 2. **CSV (Comma delimited)**: The cell data will be separated by a comma.
 3. **Formatted Text (space delimited)**: The cell data will be separated by a space.
 4. **Save as Another File Type**: Select a different file type when the Save As dialog box appears.
- The file type you select will depend on what type of file is required by the program that will consume the exported data.
5. Click **Save As**.



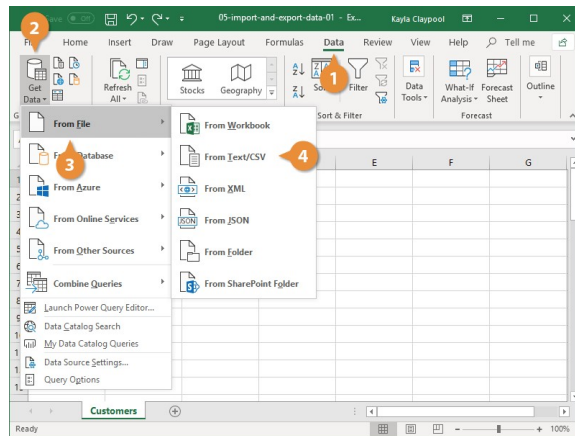
6. Specify where you want to save the file.
7. Click **Save**.



Import Data

Excel can import data from external data sources including other files, databases, or web pages.

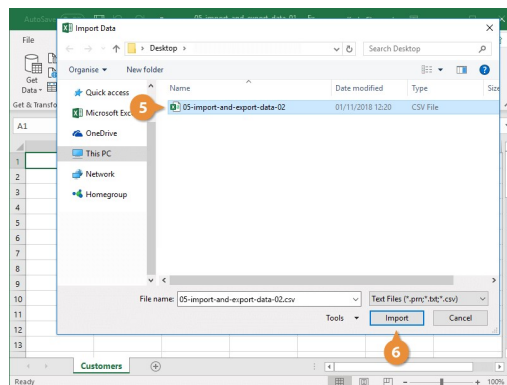
1. Click the **Data** tab on the Ribbon..
2. Click the **Get Data** button. Some data sources may require special security access, and the connection process can often be very complex. Enlist the help of your organization's technical support staff for assistance.
3. Select **From File**.
4. Select **From Text/CSV**.



If you have data to import from Access, the web, or another source, select one of those options in the Get External Data group instead.

5. Select the file you want to import.
6. Click **Import**.

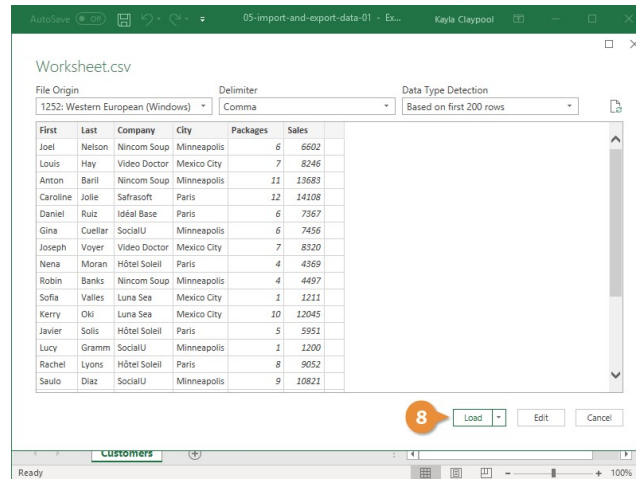
If, while importing external data, a security notice appears saying that it is connecting to an external source that may not be safe, click **OK**.



7. Verify the preview looks correct.

Because we've specified the data is separated by commas, the delimiter is already set. If you need to change it, it can be done from this menu.

8. Click **Load**.



Spreadsheet : Features ?

❑ TBD1

❑ TBD2

❑ TBD3

❑ TBD4

Spreadsheet : Installation ?

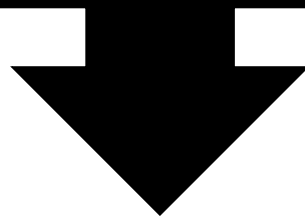
☐ TBD1

☐ TBD2

☐ TBD3

☐ TBD4

MongoDB



SQL Database Examples

- Commercial
 - IBM DB2
 - Oracle RDMS
 - Microsoft SQL Server
 - Sybase SQL Anywhere
- Open Source (with commercial options)
 - MySQL
 - Ingres

**Significant portions of the
world's economy use SQL databases!**

SQL Characteristics

- Data stored in columns and tables
- Relationships represented by data
- Data Manipulation Language
- Data Definition Language
- Transactions
- Abstraction from physical layer

NoSQL Products/Projects

<http://www.nosql-database.org/> lists 122

NoSQL Databases

- Cassandra
- CouchDB
- Hadoop & Hbase
- MongoDB
- StupidDB
- Etc.

NoSQL Database Types

Discussing NoSQL databases is complicated because there are a variety of types:

- Column Store – Each storage block contains data from only one column
- Document Store – stores documents made up of tagged elements
- Key-Value Store – Hash table of keys

NoSQL Example: Document Store

- Example: CouchDB
 - <http://couchdb.apache.org/>
 - BBC
- Example: MongoDB
 - <http://www.mongodb.org/>
 - Foursquare, Shutterfly
- JSON – JavaScript Object Notation

CouchDB JSON Example

```
{
  "_id": "guid goes here",
  "_rev": "314159",

  "type": "abstract",

  "author": "Keith W. Hare"

  "title": "SQL Standard and NoSQL Databases",

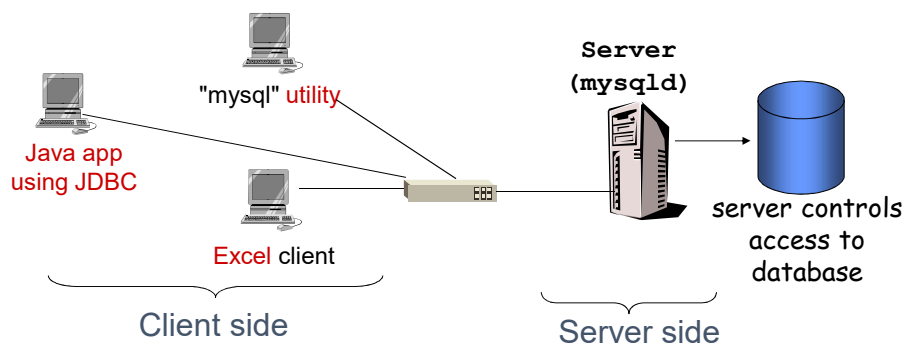
  "body": "NoSQL databases (either no-SQL or Not Only SQL)
          are currently a hot topic in some parts of
          computing.",
  "creation_timestamp": "2011/05/10 13:30:00 +0004"
}
```

NoSQL Distinguishing Characteristics

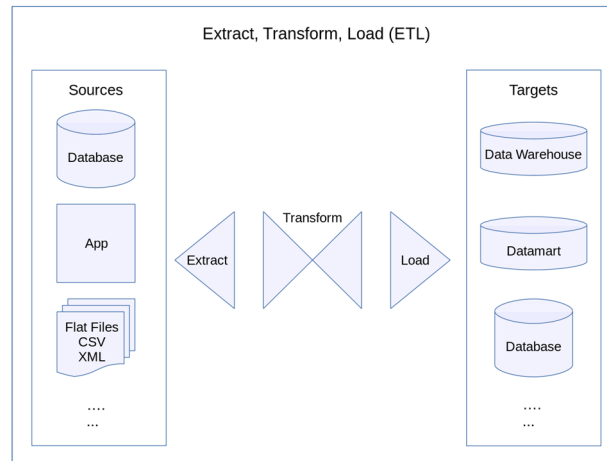
- Large data volumes
 - Google's "big data"
- Scalable replication and distribution
 - Potentially thousands of machines
 - Potentially distributed around the world
- Queries need to return answers quickly
- Mostly query, few updates
- Asynchronous Inserts & Updates
- Schema-less
- ACID transaction properties are not needed – BASE
- CAP Theorem
- Open source development

Client - Server Databases

- Database **Server** is a separate *process* running on a host.
- **Clients** can run on *any* machine.
- Many programs may be clients using a **standard API**.



- The **import and export of data** is the automated or semi-automated input and output of data sets between different software applications.
- It involves "translating" from the format used in one application into that used by another, where such translation is accomplished automatically via machine processes, such as transcoding, data transformation, and others.
- True exports of data often contain data in raw formats otherwise unreadable to end-users without the user interface that was designed to render it.



No-SQL Databases

No-SQL Database

- ❑ A No-SQL database **has dynamic schema** for unstructured data.
- ❑ Data is stored in many ways which means it can be **document-oriented**, **column-oriented**, **graph-based** or organized as a **Key-Value Store**.
- ❑ This **flexibility** means that documents can be created **without having defined structure first**.
- ❑ Also, each document can have **its own unique structure**.
- ❑ The **syntax varies** from database to database, and we can add fields **as we go**

SQL vs No-SQL Database

SQL-Databases	No-SQL Databases
Relational Database Management System (RDBMS)	Non-relational or distributed database system.
SQL-databases have fixed or static or predefined schema	No-SQL databases have dynamic schema
These databases are not suited for hierarchical data storage.	These databases are best suited for hierarchical data storage.
These databases are best suited for complex queries	These databases are not so good for complex queries
Vertically Scalable	Horizontally Scalable

BASE Property

- ❑ NoSQL relies upon a softer model known as the BASE model.
- ❑ BASE (Basically Available, Soft state, Eventual consistency).
- ❑ **Basically Available:** Guarantees the availability of the data . There will be a response to any request (can be failure too).

Types of No SQL Database

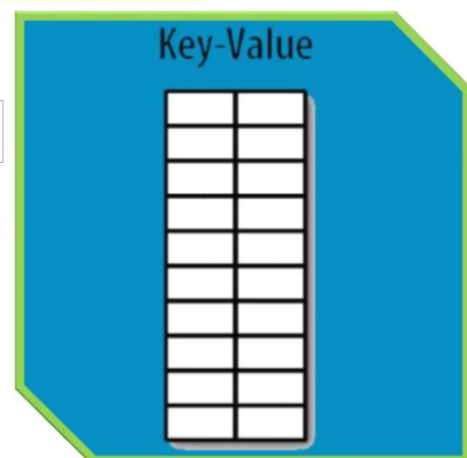
Key value Stores — Riak, Voldemort, and Redis

A key value store uses a **hash table** in which there exists a **unique key** and a **pointer** to a particular item of data.

Imagine key value stores to be like a phone directory where the names of the individual and their numbers are mapped together.

Key value stores have no default query language. You retrieve data using *get*, *put*, and *delete* commands. This is the reason it has **high performance**.

Applications: Useful for storage of Comments and Session information.
Pinterest uses Redis to store lists of users, followers, unfollowers, boards.



Types of No SQL Database

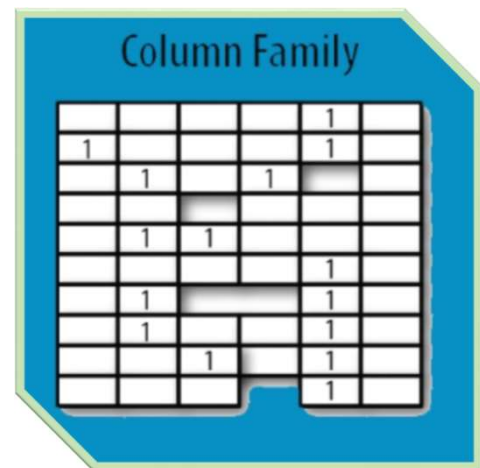
Wide Column Stores – Cassandra and HBase

In a column store database, the columns in each row are contained within that row.

Each **column family** is a container of rows in an RDBMS table. The key identifies the row consisting of multiple columns.

Rows do not need to have the **same number** of columns. Columns can be added to any row at any time without having to add it to other rows. It is a **partitioned row store**.

Example: <https://studio3t.com/wp-content/uploads/2017/12/cassandra-column-family-example.png>



Types of No SQL Database

Document databases – MongoDB

Document stores uses JSON, XML, or BSON (binary encoding of JSON) documents to store data.

A single document is to store records and its data.

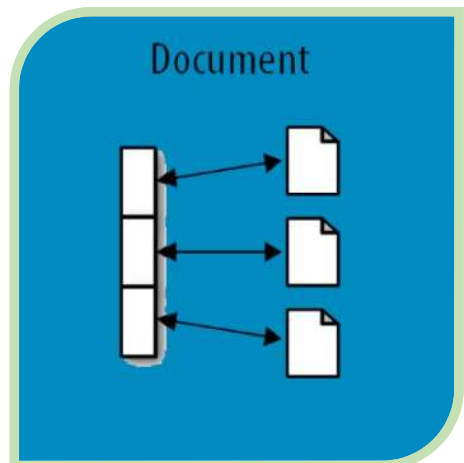
It does not support relations or joins.

Example:

https://webassets.mongodb.com/_com_assets/cms/JSON_Example_Python_MongoDB-mzqqz0keng.png

Getting Acquainted with NoSQL on Windows Azure

<https://docs.microsoft.com/en-us/archive/blogs/usisvde/getting-acquainted-with-nosql-on-windows-azure>



Types of No SQL Database

Graph databases — Neo4J and HyperGraphDB

Nodes and relationships are the essential constituents of graph databases. A **node** represents an entity. A **relationship** represents how two nodes are associated.

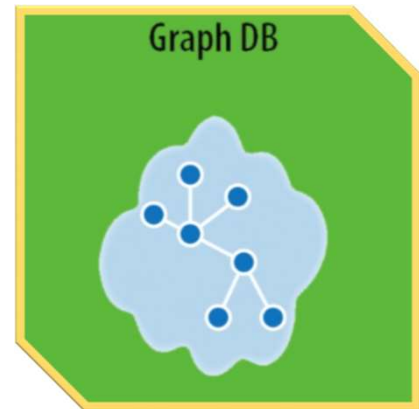
In RDBMS, adding another relation results in a lot of schema changes

Graph database requires only storing data once (nodes). The different types of relationships (edges) are specified to the stored data.

The relationships between the nodes are predetermined, that is, it is not determined at query time.

Traversing **persisted relationships** are faster.

It is difficult to change a relation between two nodes. It would result in regressive changes in the database.



Graph Traversals

EXAMPLE-1: HOW MYSQL WORKS (where it performs many operations to find a correct result for Alice)

https://s3.amazonaws.com/dev.assets.neo4j.com/wp-content/uploads/from_relational_model.png

EXAMPLE-2: HOW NO-SQL WORKS (A graph database, which predetermines relationships)

https://s3.amazonaws.com/dev.assets.neo4j.com/wp-content/uploads/relational_to_graph.png

Why use MongoDB ?

- **Document Oriented Storage** – Data is stored in the form of JSON style documents.
- Index on any attribute
- Replication and high availability
- Auto-Sharding
- Rich queries
- Fast in-place updates
- Professional support by MongoDB

Where to Use MongoDB?

- Big Data
- Content Management and Delivery
- Mobile and Social Infrastructure
- User Data Management
- Data Hub

Apache Hive



Hive : Features ?

☐ TBD1

☐ TBD2

☐ TBD3

☐ TBD4

Hive : Installation ?

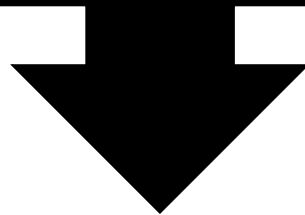
❑ TBD1

❑ TBD2

❑ TBD3

❑ TBD4

Apache HBase



Hbase : What is it ?

<https://www.ibm.com/topics/hbase>

Apache Hbase?

- ❑ **HBase is a column-oriented non-relational database management system that runs on top of [Hadoop Distributed File System \(HDFS\)](#).**
- ❑ **HBase provides a fault-tolerant way of storing sparse data sets, which are common in many big data use cases.**
- ❑ **It is well suited for real-time data processing or random read/write access to large volumes of data.**

Apache Hbase (contd..)

- ❑ Unlike [relational database systems](#), HBase does not support a structured query language like SQL; in fact, HBase isn't a relational data store at all.
- ❑ HBase applications are written in Java™ much like a typical [Apache MapReduce](#) application.
- ❑ HBase does support writing applications in [Apache Avro](#), REST and Thrift.

Apache Hbase?

- ❑ Apache HBase™ is the Hadoop database, a distributed, scalable, big data store.
- ❑ We use Apache HBase™ when we need random, realtime read/write access to our Big Data.
- ❑ Apache HBase is an open-source, distributed, versioned, non-relational database modeled after Google's Bigtable: A Distributed Storage System for Structured Data by Chang et al.
- ❑ Just as Bigtable leverages the distributed data storage provided by the Google File System, Apache HBase provides Bigtable-like capabilities on top of Hadoop and HDFS.

Apache Hbase (contd..)

- An HBase system is designed to scale linearly.
- It comprises a set of standard tables with rows and columns, much like a traditional database.
- Each table must have an element defined as a primary key, and all access attempts to HBase tables must use this primary key.
- Avro, as a component, supports a rich set of primitive data types including: numeric, binary data and strings; and a number of complex types including arrays, maps, enumerations and records.
- A sort order can also be defined for the data.

Apache Hbase (contd..)

- HBase relies on ZooKeeper for high-performance coordination. ZooKeeper is built into HBase, but if you're running a production cluster, it's suggested that you have a dedicated ZooKeeper cluster that's integrated with your HBase cluster.
- HBase works well with Hive, a query engine for batch processing of big data, to enable fault-tolerant big data applications.

Hbase Example(s)

- An HBase column represents an attribute of an **object**; if the table is storing diagnostic logs from servers in our environment, each **row** might be a **log record**, and a typical column could be the timestamp of when the log record was written, or the server name where the record originated.
- HBase allows for many attributes to be grouped together into column families, such that the elements of a column family are all stored together. This is **different** from a **row-oriented relational database**, where all the columns of a given row are stored together.
- With HBase we **must** **predefine the table schema and specify the column families**. However, new columns can be added to families at any time, making the schema flexible and able to adapt to changing application requirements.

Hbase Example(s)

- Just as **HDFS** has a **NameNode** and slave nodes, and **MapReduce** has **JobTracker** and **TaskTracker** slaves, HBase is built on similar concepts.
- In **HBase** a **master node** manages the cluster and **region servers** store portions of the tables and perform the work on the data. In the same way HDFS has some enterprise concerns due to the availability of the NameNode
- HBase is also sensitive to the loss of its master node.

Hbase : Features?

- Linear and modular scalability.
- Strictly consistent reads and writes.
- Automatic and configurable sharding of tables
- Automatic failover support between RegionServers.
- Convenient base classes for backing Hadoop MapReduce jobs with Apache HBase tables.
- Easy to use Java API for client access.
- Block cache and Bloom Filters for real-time queries.
- Query predicate push down via server side Filters
- Thrift gateway and a REST-ful Web service that supports XML, Protobuf, and binary data encoding options
- Extensible jruby-based (JIRB) shell
- Support for exporting metrics via the Hadoop metrics subsystem to files or Ganglia; or via JMX

Apache Foundation ?

<https://www.apache.org/>



<https://hbase.apache.org/>

The Hive logo, consisting of the word "Hive" in a bold, yellow, sans-serif font, is centered within a black rectangular box. A large, black, downward-pointing arrow is positioned directly below the box, pointing towards the text "Hive – Genesis?" in the section below.

Hive – Genesis?

- Hive was originally developed by Facebook and is now maintained as Apache Hive by APACHE SOFTWARE FOUNDATION.
- HADOOP ecosystem is scalable & cost-effective for processing large volumes of data.
- It is a relatively new framework that packs a lot of punch.
- Organizations with traditional data warehouses are based on SQL – with users and developers that rely on SQL-queries for extracting data

Hive – Genesis (contd..)

- Consequently, getting used to Hadoop system is a challenging task.
- Hive was developed to address this challenge by leveraging SQL intellect, so that users can write SQL-like queries called HQL (or HiveQL) to extract the data from Hadoop.

What is Hive ?

- Apache Hive is a distributed, fault-tolerant data warehouse system that enables analytics at a massive scale.
- A data warehouse provides a central store of information that can easily be analyzed to make informed, data driven decisions.
- Hive allows users to read, write, and manage petabytes of data using SQL.

What is Hive (contd..)

- Hive is built on top of Apache Hadoop, which is an open-source framework used to efficiently store and process large datasets.
- Hive is closely integrated with Hadoop, and is designed to work quickly on petabytes of data.
- What makes Hive unique is the ability to query large datasets, leveraging Apache Tez or MapReduce, with a SQL-like interface.

Why do we need to learn Hive?

- Hive was created to allow non-programmers familiar with SQL to work with petabytes of data, using a SQL-like interface called HiveQL.
- Traditional relational databases are designed for interactive queries on small to medium datasets and do not process huge datasets well. Hive instead uses batch processing so that it works quickly across a very large distributed database.
- Hive transforms HiveQL queries into MapReduce or Tez jobs that run on Apache Hadoop's distributed job scheduling framework, Yet Another Resource Negotiator (YARN). It queries data stored in a distributed storage solution, like the Hadoop Distributed File System (HDFS) or Amazon S3.
- Hive stores its database and table metadata in a metastore, which is a database or file backed store that enables easy data abstraction and discovery.

Need for Learning Hive (contd..)

- Hive includes **HCatalog**, which is a table and storage management layer that reads data from the Hive metastore to facilitate seamless integration between Hive, Apache Pig, and MapReduce.
- By using the metastore, HCatalog allows Pig and MapReduce to use the same data structures as Hive, so that the metadata doesn't have to be redefined for each engine.
- Custom applications or third party integrations can use WebHCat, which is a RESTful API for HCatalog to access and reuse Hive metadata.

Benefits of Hive

- **FAST** → Hive is designed to quickly handle petabytes of data using batch processing.
- **FAMILIAR** → Hive provides a familiar, SQL-like interface that is accessible to non-programmers.
- **SCALABLE** → Hive is easy to distribute and scale based on your needs.

Comparing Hbase & Hive?

Parameter	Apache - Hbase	Apache - Hive
Function	Low-latency distributed key-value store with custom query capabilities. Data is stored in a column-oriented format.	SQL-like query engine designed for high volume data stores. Multiple file-formats are supported.
Processing Type	Real-time processing.	Batch processing using Apache Tez or MapReduce compute frameworks.
Latency	Low, but it can be inconsistent. Structural limitations of the HBase architecture can result in latency spikes under intense write loads.	Medium to high, depending on the responsiveness of the compute engine. The distributed execution model provides superior performance compared to monolithic query systems, like RDBMS, for the same data volumes.
Hadoop Integration	Runs on top of HDFS or Amazon S3	Runs on top of Hadoop, with Apache Tez or MapReduce for processing and HDFS or Amazon S3 for storage.
SQL-Support	No SQL support on its own. You can use Apache Phoenix for SQL capabilities.	Provides SQL-like querying capabilities with HiveQL.
Schema	Schema-free.	Defined schema for all tables.
Data Types	Supports unstructured data only. The user defines mappings of data fields to Java-supported data types.	Supports structured and unstructured data. Provides native support for common SQL data types, like INT, FLOAT, and VARCHAR

Comparative Statement

Parameter	Apache - Hbase	Apache - Hive
Function	Low-latency distributed key-value store with custom query capabilities. Data is stored in a column-oriented format.	SQL-like query engine designed for high volume data stores. Multiple file-formats are supported.
Processing Type	Real-time processing.	Batch processing using Apache Tez or MapReduce compute frameworks.
Latency	Low, but it can be inconsistent. Structural limitations of the HBase architecture can result in latency spikes under intense write loads.	Medium to high, depending on the responsiveness of the compute engine. The distributed execution model provides superior performance compared to monolithic query systems, like RDBMS, for the same data volumes.
Hadoop Integration	Runs on top of HDFS or Amazon S3	Runs on top of Hadoop, with Apache Tez or MapReduce for processing and HDFS or Amazon S3 for storage.
SQL-Support	No SQL support on its own. You can use Apache Phoenix for SQL capabilities.	Provides SQL-like querying capabilities with HiveQL.
Schema	Schema-free.	Defined schema for all tables.
Data Types	Supports unstructured data only. The user defines mappings of data fields to Java-supported data types.	Supports structured and unstructured data. Provides native support for common SQL data types, like INT, FLOAT, and VARCHAR

Installing R-interfaces of Popular Databases

R packages/libraries

Data input/output

What is an R package?

- R package – a collection of R functions, data, and compiled code in a well-defined format
- R library – directory where the packages are stored
- R has a standard set of packages
 - Free to download/install
 - `library()` # see all packages installed on your computer
 - `search()` # packages currently loaded

Install an R package

- Complete list of available packages:
<http://cran.r-project.org/web/packages/>
- Follow the steps:
 - 1. Open R, click on **Packages** menu and then **Install Packages**
 - 2. Choose a **CRAN mirror** (USA – TX1 ?)
 - 3. Select the package you want to add (e.g., survival)
 - 4. Load it using **library**(whatever you selected)
- One step: `install.packages("your choice")`

R package - help

- `library(help="pack.name")` # information/description of the package
- `unload("package:pack.name, unload=TRUE)` # unload the package
- `?? function.name` # fuzzy matching

Some add-on packages

- base # base R functions
- datasets # base R datasets
- graphics # some functions: `plot()`, `par()`
- splines # regression spline functions and classes
- stats # some functions: `lm()`, `glm()`
- tools # tools for package development and admin.
- utils # some functions: `load ()`, `data()`

Some popular R packages

- `ggplot2` # best graphs
- `zoo` # time series analysis
- `Hmisc` # sample size computation, imputing missing data
- `rms` # regression analyses and more (Frank Harrell)
- `nlme` and `lme4` # mixed models
- `ff` # working with large arrays
- `fields` # spatial statistics

Data Types

- Scalars and Vectors (numerical, character, logical, complex)
- Matrices/arrays
- Lists/data frames
- Identify the data structure
 - `typeof()`
 - `mode()`
 - `class()`
- Other useful functions
 - `str()` # see structure
 - `describe()`

Data Types (con't)

- **Vectors** – must consist of values of the same data type
- **Factors** - encode categorical data
- **Matrices/Arrays**
 - Matrix: 2-dimensional array
 - Arrays**: stored as 1 dimensional structure; can stack several matrices

Data Types (con't)

- **Lists** – ordered collections of objects of any type or dimension
- **Data frames** – store data of various formats with related entries in each row, different attributes in each column

Let's check some of these types by entering data from the keyboard!

Data Input

- Entering data within R
 - `save(object, file="path//mydata.RData")` # store R data
 - `load("mydata.Rdata")`
- Reading (importing) data into R
 - `read.table("path//file.dat")` # read table-format data
 - `read.csv("path//file.csv")` # read comma separated values (spreadsheet)
 - `read.delim()` # read tab-delimited data
 - `read.fwf()` # read fixed-width format data (.txt)
 - `scan()` # read data directly into a vector or a list

Data Input – using packages

- **From Excel** – the first row of the file needs to contain the variable names


```
library(RODBC)
excel.data <- odbcConnectExcel("path//myexcel.xls")
mydata <- sqlFetch(excel.data, "mysheet")    # specify the worksheet
odbcClose(excel.data)
```
- **From SAS** – save SAS dataset in .xpt format, e.g. "mydata.xpt"


```
library(Hmisc)
mydata <- sasxport.get("path//mydata.xpt")
```
- **From Stata**

```
library(foreign)    # can also read data from Minitab, SAS, SPSS
mydata <- read.dta("path//mydata.dta")
```

Data Input – using packages

- **From SPSS** - save SPSS dataset in .por format

```
library(Hmisc)
mydata <- spss.get("path/mydata.por", use.value.labels=TRUE)
```

- Other useful packages

```
library(gdata)          # read.xls() for importing Excel data
```

Data Output

- Export data

- Tab delimited text file:

```
write.table(mydata, "path//mydata.txt", sep="\t", rownames=F)
```

- Comma separated file:

```
write.csv(mydata, "path//mydata.csv", rownames=F, na=" ")
```

Will not preserve the special attributes, e.g. column type as character

- Save data in R binary format (more compact)

```
save("data", file="mydata.Rdata")
```

```
load("mydata.Rdata")
```

Data Output (con't)

- R functions for data output
 - `print()` # keep special characters
 - `paste()` # concatenate vectors after converting to character
 - `format()` # format an R object for nice printing
 - `cat()` # output objects, concatenating the representations
- Sweave - create dynamics reports with the R code mixed with analysis output: tables, graphs, etc. into a nice PDF

Data Output – using packages

- Export to an Excel spreadsheet


```
library(xlsReadWrite)
write.xls(mydata, "path//mydata.xls")
```
- Export to SAS


```
library(foreign)
write.foreign(mydata, "path//mydata.txt", package="SAS")
```
- Export to Stata


```
library(foreign)
write.dta(mydata, "path//mydata.dta")
```
- Package `xtable` exports tables to LaTeX or HTML

Some guidelines

- Use *read.table()* to read into a data frame (data needs to be rectangular)
- Use *write.table()* to write a data frame or matrix. It can also append data into a file.
- Use package RODB for a universal way of importing all kinds of databases
- For large amounts of Excel data, convert to CSV and then read with *read.csv()*. Export with *write.csv()*.
- Great resource:
<http://cran.r-project.org/doc/manuals/R-data.html#Introduction>

References & Acknowledgements

- <https://www.techonthenet.com/postgresql/index.php>
- <https://opensource.com/>
- <https://www.techonthenet.com/sqlite/index.php>
- <https://www.w3resource.com/sqlite/>
- <https://aws.amazon.com>