IST 769 Final Project Document

**Prompt:** What is HBase?

**Answer:** HBase is an open-source, NoSQL, distributed big data store that provides random, real-time access to petabytes of data. It is a column-oriented, non-relational database management system that is designed to provide quick random access to huge amounts of structured data. HBase is a wide-column db. HBase is based on Google's Big Table and is well-suited for handling large, sparse datasets. It is horizontally scalable, meaning that you can add any number of columns anytime, and it provides automatic failover to ensure high availability. HBase integrates seamlessly with the Hadoop ecosystem and runs on top of the Hadoop Distributed File System (HDFS)

**Prompt:** What is the storage model of HBase

**Answer:** The storage model of HBase is column-oriented, and the tables in it are sorted by row. The table schema defines only column families, which are the key-value pairs, and a table can have multiple column families, each of which can have any number of columns. Subsequent column values are stored contiguously on the disk, and the data is compressed to save disk space.

**Prompt:** What is the logical model of HBase

The logical model of HBase consists of several components, including tables, rows, column families, cells, columns, and versions. Tables in HBase are collections of rows and column families, and rows are a collection of column families. Each table must have an element defined as a primary key, and the row key acts as the primary key in HBase. A column in the HBase data model table represents attributes to the objects. HBase tables contain column families and rows with elements defined as primary keys. The tables in HBase are more logical than physical, and an HBase table can be described as a collection of rows. The HBase data model is similar to Google's big table and is designed to provide quick random access to huge amounts of structured data.

**Prompt:** How does HBase scale?

**Answer:** HBase is horizontally scalable, meaning that it can handle a large amount of data by adding more nodes to the cluster

**Prompt:** HBase OLTP

**Answer:** HBase is designed to be an OLTP (Online Transaction Processing) engine, allowing high-volume transactional operations. HBase is well-suited for real-time queries on large datasets and is optimized for OLTP. HBase is used for well over 100 different business use cases primarily OLTP workloads.

**Prompt:** Hbase OLAP

**Answer:** HBase is not designed for OLAP (Online Analytical Processing) workloads, which require complex queries and aggregations over large datasets. For OLAP workloads, Apache Hive is a better solution as it is mainly used for batch processing.

**Prompt:** How do you design for HBase?

**Answer**: Designing for HBase involves understanding the data model and the various concepts at play in designing the schema. HBase is a distributed column-oriented database built on top of the Hadoop file system that is horizontally scalable and provides quick random access to huge amounts of structured data. HBase is well-suited for handling large, sparse datasets and provides APIs enabling development in practically any programming language. HBase tables are collections of rows and column families, and rows are a collection of column families. Each table must have an element defined as a primary key, and the row key acts as the primary key in HBase. A column in the HBase data model table represents attributes to the objects. HBase tables contain column families and rows with elements defined as primary keys. HBase is good for semi-structured as well as structured data and is well-suited for handling large, sparse datasets.

**Prompt:** How do you connect to HBase via the Command Line?

**Answer**: To connect to HBase via the command line, you need to navigate to the HBase home folder and start the HBase interactive shell using the "hbase shell" command. The shell becomes the remote shell that connects to the HBase master via a private connection. Once you enter the HBase shell, you can execute all shell commands for table operations, table management, and data modeling.

**Prompt:** How do you connect to HBase via PySpark?

**Answer:** To connect to HBase via PySpark, you can use the HappyBase package, which allows connecting to HBase from Python by using HBase's Thrift API. First, you need to install HappyBase using the pip command. Then, you can launch the HBase shell and create a table with a column family. After that, you can launch PySpark and test that you can connect to HBase using the HappyBase package. There are also other third-party connectors available to connect to HBase in Spark, such as the Hortonworks connector. Spark does not include built-in HBase connectors. Examples are available on how to connect to HBase from Spark with Python to perform read and write operations.

**Sources:**

[**https://aws.amazon.com/big-data/what-is-hbase/**](https://aws.amazon.com/big-data/what-is-hbase/)

[**https://hbase.apache.org/**](https://hbase.apache.org/)

[**https://www.ibm.com/topics/hbase**](https://www.ibm.com/topics/hbase)

[**https://www.tutorialspoint.com/hbase/hbase\_overview.htm**](https://www.tutorialspoint.com/hbase/hbase_overview.htm)

[**https://www.guru99.com/hbase-architecture-data-flow-usecases.html**](https://www.guru99.com/hbase-architecture-data-flow-usecases.html)

[**https://techguysfdc.medium.com/logical-components-in-hbase-data-model-e445a88f086b**](https://techguysfdc.medium.com/logical-components-in-hbase-data-model-e445a88f086b)

[**https://www.projectpro.io/article/overview-of-hbase-architecture-and-its-components/295**](https://www.projectpro.io/article/overview-of-hbase-architecture-and-its-components/295)

[**https://subscription.packtpub.com/book/big-data-&-business-intelligence/9781784396688/5/ch05lvl1sec44/the-hbase-data-model**](https://subscription.packtpub.com/book/big-data-&-business-intelligence/9781784396688/5/ch05lvl1sec44/the-hbase-data-model)

[**https://www.ericsson.com/en/blog/2013/11/horizontal-scalability-with-hbase**](https://www.ericsson.com/en/blog/2013/11/horizontal-scalability-with-hbase)

[**https://blog.cloudera.com/big-data-processing-engines-which-one-do-i-use-part-1/**](https://blog.cloudera.com/big-data-processing-engines-which-one-do-i-use-part-1/)

[**https://www.geeksforgeeks.org/difference-between-hive-and-hbase/**](https://www.geeksforgeeks.org/difference-between-hive-and-hbase/)

[**https://blog.flipkart.tech/hbase-multi-tenancy-part-i-37cad340c0fa?gi=5fc9dcbf9e50**](https://blog.flipkart.tech/hbase-multi-tenancy-part-i-37cad340c0fa?gi=5fc9dcbf9e50)

[**https://www.projectpro.io/article/hive-vs-hbase-different-technologies-that-work-better-together/322**](https://www.projectpro.io/article/hive-vs-hbase-different-technologies-that-work-better-together/322)

[**https://www.tutorialspoint.com/hbase/hbase\_shell.htm**](https://www.tutorialspoint.com/hbase/hbase_shell.htm)

[**https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781783983063/10/ch10lvl1sec52/connecting-to-an-hbase-cluster-using-the-command-line**](https://subscription.packtpub.com/book/big-data-and-business-intelligence/9781783983063/10/ch10lvl1sec52/connecting-to-an-hbase-cluster-using-the-command-line)

[**https://sparkbyexamples.com/hbase/spark-hbase-connectors-which-one-to-use/**](https://sparkbyexamples.com/hbase/spark-hbase-connectors-which-one-to-use/)

[**https://kontext.tech/article/628/spark-connect-to-hbase**](https://kontext.tech/article/628/spark-connect-to-hbase)

[**https://diogoalexandrefranco.github.io/interacting-with-hbase-from-pyspark/**](https://diogoalexandrefranco.github.io/interacting-with-hbase-from-pyspark/)