ACID – Atomicity, Consistency, Isolation, Durability.

SQL vs Hive

|  |  |  |
| --- | --- | --- |
| RDBMS Operations | ANSI SQL Supports | Hive Supports |
| DDL | Create, Alter, Rename, Drop | Create, Alter, Rename, Drop |
| DML | Insert, Update, Delete | Load, Insert, Update, Delete |
| DQL | Select | Select |
| DCL | Grant, Revoke | Grant, Revoke |
| TCL | Commit, Rollback, Truncate | Not Supported |

Hive Database Objects:

Database, Schema, Table, View, Index, Partitions, Functions, Columns.

Serde:

The record parsing of a [Hive](https://www.edureka.co/blog/hive-tutorial/?utm_source=quora&utm_medium=crosspost&utm_campaign=social-media-edureka-ab) table is handled by a serializer/deserializerw or SerDe for short.

* Deserializer:

The Hive deserializer converts record (string or binary) into a java object that Hive can process (modify).

* Serializer:

Now, the Hive serializer will take this Java object, convert it into suitable format that can be stored into HDFS.

A SerDe allows Hive to read in data from a table, and write it back out to HDFS in any custom format. Anyone can write their own SerDe for their own data formats.

Built-in SerDes (Different file formats)

* [Avro](https://cwiki.apache.org/confluence/display/Hive/AvroSerDe)
* [ORC](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+ORC)
* [RegEx](https://cwiki.apache.org/confluence/display/Hive/GettingStarted#GettingStarted-ApacheWeblogData)
* [Thrift](http://thrift.apache.org/)
* [Parquet](https://cwiki.apache.org/confluence/display/Hive/Parquet)
* [CSV](https://cwiki.apache.org/confluence/display/Hive/CSV+Serde)
* [JsonSerDe](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL#LanguageManualDDL-RowFormats&SerDe)

The hive file format is chosen based on the performance and encoding required.

**Data Types:**

|  |  |  |
| --- | --- | --- |
| **Type** | **Bytes** | **Examples** |
| TINYINT | 1 Byte | 20 |
| SMALLINT | 2 Byte | 20 |
| INT | 3 Byte | 20 |
| BIGINT | 4 Byte | 20 |
| BOOLEAN | True/False | TRUE |
| FLOAT | 4 Byte | 3.14159 |
| DOUBLE | 4 Byte | 3.14159 |
| String |  |  |
| Varchar |  |  |
| Char |  |  |
| Timestamp |  |  |

**Collection DataTypes:**

Very Good:

<https://blogs.msdn.microsoft.com/bluewatersql/2014/10/01/introduction-to-hive-collections/>

**DDL Commands Syntax:**

**Create Table Syntax:**

CREATE TABLE IF NOT EXISTS mydb.employees (

name STRING COMMENT 'Employee name',

salary FLOAT COMMENT 'Employee salary',

subordinates ARRAY<STRING> COMMENT 'Names of subordinates',

deductions MAP<STRING, FLOAT>

COMMENT 'Keys are deductions names, values are percentages',

address STRUCT<street:STRING, city:STRING, state:STRING, zip:INT>

COMMENT 'Home address')

COMMENT 'Description of the table'

TBLPROPERTIES ('creator'='me', 'created\_at'='2012-01-02 10:00:00', ...)

LOCATION '/user/hive/warehouse/mydb.db/employees';

**Create Table from another table:**

create table kiran\_tb as select \*from stocks\_tb limit 10;

**Create Table Structure from other Table:**

create table kiran\_tb as select \*from stocks\_tb where 1=2;

**For Selected columns and selected rows**

CREATE TABLE ca\_employees

AS SELECT name, salary, address

FROM employees

WHERE se.state = 'CA';

**Rename:**

ALTER TABLE name RENAME TO new\_name

ALTER TABLE name ADD COLUMNS (col\_spec[, col\_spec ...])

ALTER TABLE name DROP [COLUMN] column\_name

ALTER TABLE name CHANGE column\_name new\_name new\_type

ALTER TABLE name REPLACE COLUMNS (col\_spec[, col\_spec ...])