**Vectorization**  
Vectorized query execution is a Hive feature that greatly **reduces the CPU usage** for **typical query operations like scans, filters, aggregates, and joins**.

**Vectorized query execution streamlines operations by processing a block of 1024 rows at a time.**

Vectorization allows Hive to process a batch of rows together instead of processing one row at a time. Each batch is usually an array of primitive types. Operations are performed on the entire column vector, which improves the instruction pipelines and cache usage.

**To use vectorized query execution, you must store your data in**[**ORC**](https://cwiki.apache.org/confluence/display/Hive/LanguageManual+ORC)**format**

**Enable Vectorization in Hive: hive.vectorized.execution.enabled=true ;**

When vectorization is enabled, Hive examines the query and the data to determine whether vectorization can be supported. If it cannot be supported, Hive will execute the query with vectorization turned off.

**Supported Functionality**

**The current implementation supports only single table read-only queries. DDL queries or DML queries are not supported.**

The supported operators are **selection**, **filter** and **group by**.

**Partitioned** tables are supported.

**These data types are supported:**

**Tinyint,smallint,int,bigint,date,Boolean, float,double,timestamp,string,char,varchar,binary**

**These expressions are supported:**

* Comparison: >, >=, <, <=, =, !=
* Arithmetic: plus, minus, multiply, divide, modulo
* Logical: AND, OR
* Aggregates: sum, avg, count, min, max

**Only the ORC file format** is supported in the current implementation

Seeing whether vectorization is used for a query:

>>create table vectorizedtable(state string,id int) stored as orc;

>>insert into vectorizedtable values('haryana',1);

>>set hive.vectorized.execution.enabled = true;

>>explain select count(\*) from vectorizedtable;

.

**Reference:**

[**https://streever.atlassian.net/wiki/spaces/HADOOP/pages/1933314/Hive+Performance+Tips**](https://streever.atlassian.net/wiki/spaces/HADOOP/pages/1933314/Hive+Performance+Tips)

|  |  |
| --- | --- |
| **Application** | Key |
| **Skew Merge Bucket Joins** | set hive.optimize.skewjoin = true;  set hive.skewjoin.key = skew\_key\_threshold |
| **Sort Merge Bucket Map Join** | set hive.optimize.bucketmapjoin = true;  set hive.optimize.bucketmapjoin.sortedmerge = true;  set hive.input.format=[org.apache.hadoop.hive.ql.io](http://org.apache.hadoop.hive.ql.io/).BucketizedHiveInputFormat; |
| **Bucket Map Join** | set hive.optimize.bucketmapjoin = true; |
| **Prevent MapJoins for Large Tables** | set hive.auto.convert.join=false; |
| **Controlling the CombinedHiveInputFormat Size** | |  |  |  | | --- | --- | --- | | set mapred.max.split.size=268435456; |  |  | | set mapred.min.split.size= |  |  | | set mapreduce.input.fileinputformat.split.maxsize= |  |  | | set mapreduce.input.fileinputformat.split.minsize= |  |  | |  |  |  | |
| **Dynamic Partition Creation** | set hive.exec.compress.output=true |
| **Control the Interme** | |  |  |  |  | | --- | --- | --- | --- | | set hive.exec.compress.intermediate=true |  |  |  | | set io.sort.mb=400  set hive.limit.pushdown.memory.usage=0.1f  set hive.optimize.correlation=true; |  |  |  | |
| **Vectorization** |  |
|  | set hive.vectorized.execution.enabled=true;  set hive.mapjoin.lazy.hashtable=false; |
| **Hive Security** | hive.security.authorization.enabled=[true|false]  hive.security.authorization.manager   * org.apache.hadoop.hive.ql.security.authorization.StorageBasedAuthorizationProvider * org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLStdHiveAuthorizerFactory   hive.server2.enable.doAs   * false (when hive.security.authorization.manager=org.apache.hadoop.hive.ql.security.authorization.plugin.sqlstd.SQLStdHiveAuthorizerFactory)   hive.security.authenticator.manager   * org.apache.hadoop.hive.ql.security.SessionStateUserAuthenticator |

|  |  |
| --- | --- |
| **Tez** | set hive.optimize.ppd=[true,false]  set hive.execution.engine=[tez,mr]  set hive.optimize.index.filter=true;  set hive.optimize.ppd.storage=true;  set tez.queue.name=myqueue  set tez.am.container.session.delay-allocation-millis=1000  0.13 - Note: This feature doesn't work too well yet  <property>    <name>hive.prewarm.enabled</name>   <value>false</value>   <description>Enables container prewarm for tez (hadoop 2 only)</description>  </property> |
| **Hive Stats** | **The items below will require the stats for each table.  Set the stats and 'analyze' the table/partitions as directed.**  **set hive.stats.autogather=true;**  **set hive.stats.dbclass=fs;**  **analyze table store\_sales partition(ss\_sold\_date) compute statistics partialscan;**  set hive.stats.autographer = [true, \*\*false\*\*]  set hive.compute.query.using.stats=true;  set hive.fetch.task.conversion=more;  set hive.optimize.index.filter=true; |
| **Hive CBO** | hive.compute.query.using.stats = [true, \*\*false\*\*];  hive.stats.fetch.column.stats = [true, \*\*false\*\*];  hive.stats.fetch.partition.stats = [true, \*\*false\*\*];  hive.cbo.enable = [true, \*\*false\*\*]; |
| **Hive Tuning** | hive.optimize.sort.dynamic.partition = [ \*\*true\*\*, false ] |
| **Hive Server 2** | hive.execution.engine  hive.tez.container.size  hive.tez.java.opts  hive.server2.tez.default.queues  hive.server2.tez.sessions.per.default.queue  hive.server2.tez.initialize.default.sessions  hive.server2.enable.doAs  hive.prewarm.numcontainers |
| **ORC File Tuning** | hive.exec.orc.memory.pool |
| **ORC File Versioning** | set hive.exec.orc.write.format="0.11" |