Захожу в hive выбираю БД student3_16 и создаю внешнюю таблицу в формате хранения parquet

```
create external table v_parquet

stored as parquet

as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_parquet
```

Устанавливаю сжатие snappy

```
SET parquet.compression=SNAPPY;
```

Создаю новую таблицу

```
create external table v_parquet_sn

stored as parquet

as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_parquet_sn
```

Устанавливаю сжатие GZIP

```
SET parquet.compression=GZIP;
```

Создаю новую таблицу

```
create external table v_parquet_gz

stored as parquet

as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_parquet_gz
```

Смотрю на служебную информацию по файлам

```
Hadoop jar /opt/parquet-tools.jar meta
hdfs://10.0.0.7/warehouse/tablespace/external/hive/student3_16.db/v_parquet
```

```
Hadoop jar /opt/parquet-tools.jar meta
hdfs://10.0.0.7/warehouse/tablespace/external/hive/student3_16.db/v_parquet_sn
```

```
Hadoop jar /opt/parquet-tools.jar meta
hdfs://10.0.0.7/warehouse/tablespace/external/hive/student3_16.db/v_parquet_gz
```

Создаю таблицу в формате хранения avro

```
create external table v_avro

stored as avro

as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_avro
```

Устанавливаю сжатие snappy

```
SET avro.output.codec=SNAPPY;
```

Создаю новую таблицу

```
create external table v_avro_sn

stored as avro

as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_avro_sn
```

Устанавливаю сжатие gzip

```
SET avro.output.codec=GZIP;
```

Создаю новую таблицу

```
create external table v_avro_sn
stored as avro
as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_avro_gz
```

Смотрю на служебную информацию по файлам

```
Hadoop jar /opt/avro-tools.jar meta hdfs://10.0.0.7/warehouse/tablespace/external/hive/student3_16.db/v_avro
Hadoop jar /opt/avro-tools.jar meta hdfs://10.0.0.7/warehouse/tablespace/external/hive/student3_16.db/v_avro_sn
Hadoop jar /opt/avro-tools.jar meta hdfs://10.0.0.7/warehouse/tablespace/external/hive/student3_16.db/v_avro_sn
```

Создаю таблицу в формате хранения orc

```
create external table v_orc
stored as orc
as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_orc
```

Создаю новую таблицу со сжатием zlib

```
create external table v_orc_zlib
stored as orc
TBLPROPERTIES ("orc.compress"="ZLIB")
as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_orc_zlib
```

Создаю новую таблицу со сжатием snappy

```
create external table v_orc_sn
stored as orc
TBLPROPERTIES ("orc.compress"="snappy")
as select * from vehicles_2 limit 1000;
```

Проверяю наличие и размер файла

```
hdfs dfs -du -h /warehouse/tablespace/external/hive/student3_16.db/ v_orc_sn
```

Смотрю служебную информацию по файлам

```
hive –orcfiledump /warehouse/tablespace/external/hive/student3_16.db/ v_orc
hive –orcfiledump /warehouse/tablespace/external/hive/student3_16.db/ v_orc_zlib
hive –orcfiledump /warehouse/tablespace/external/hive/student3_16.db/ v_orc_sn
```

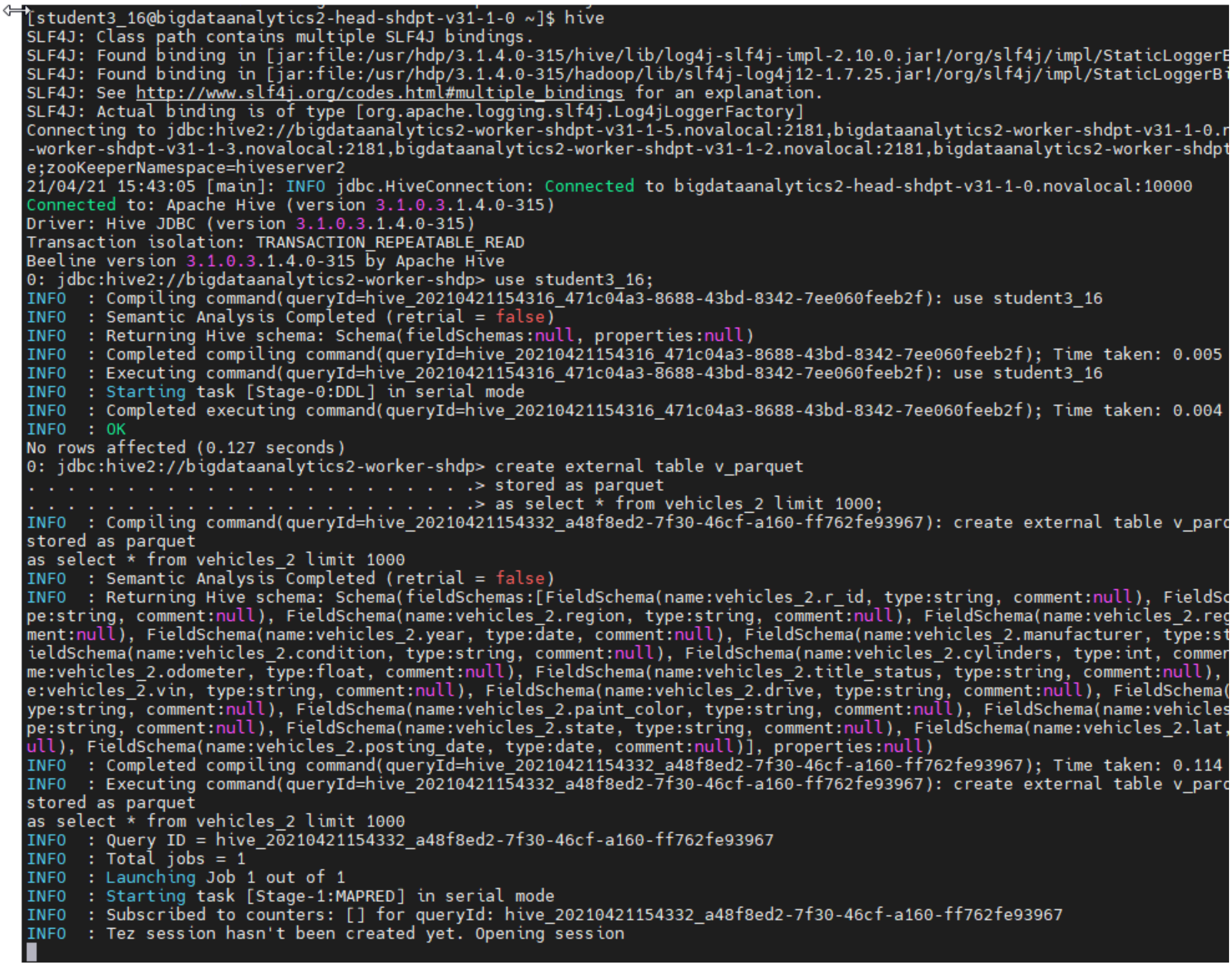
Сравниваю по времени выполнения запросов разные форматы хранения и сжатия

```
select manufacturer, sum(price) from vehicles group by manufacturer order by manufacturer;
select manufacturer, sum(price) from v_parquet group by manufacturer order by manufacturer;
select manufacturer, sum(price) from v_parquet_sn group by manufacturer order by manufacturer;
select manufacturer, sum(price) from v_parquet_gz group by manufacturer order by manufacturer;

select manufacturer, sum(price) from v_avro group by manufacturer order by manufacturer;
select manufacturer, sum(price) from v_avro_sn group by manufacturer order by manufacturer;
select manufacturer, sum(price) from v_avro_gz group by manufacturer order by manufacturer;

select manufacturer, sum(price) from v_orc group by manufacturer order by manufacturer;
select manufacturer, sum(price) from v_orc_zlib group by manufacturer order by manufacturer;
select manufacturer, sum(price) from v_orc_sn group by manufacturer order by manufacturer;
```

К сожалению, не получилось все протестировать на сервере из-за очереди Zeppelin



```
[student3_16@bigdataanalytics2-head-shdpt-v31-1-0 ~]$ hive
SLF4J: Class path contains multiple SLF4J bindings.
SLF4J: Found binding in [jar:file:/usr/hdp/3.1.4.0-315/hive/lib/log4j-slf4j-impl-2.10.0.jar!/org/slf4j/impl/StaticLoggerE
SLF4J: Found binding in [jar:file:/usr/hdp/3.1.4.0-315/hadoop/lib/slf4j-log4j12-1.7.25.jar!/org/slf4j/impl/StaticLoggerBi
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.
SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]
Connecting to jdbc:hive2://bigdataanalytics2-worker-shdpt-v31-1-5.novalocal:2181,bigdataanalytics2-worker-shdpt-v31-1-0.r
-worker-shdpt-v31-1-3.novalocal:2181,bigdataanalytics2-worker-shdpt-v31-1-2.novalocal:2181,bigdataanalytics2-worker-shdpt
e;zooKeeperNamespace=hiveserver2
21/04/21 15:43:05 [main]: INFO jdbc.HiveConnection: Connected to bigdataanalytics2-head-shdpt-v31-1-0.novalocal:10000
Connected to: Apache Hive (version 3.1.0.3.1.4.0-315)
Driver: Hive JDBC (version 3.1.0.3.1.4.0-315)
Transaction isolation: TRANSACTION_REPEATABLE_READ
Beeline version 3.1.0.3.1.4.0-315 by Apache Hive
0: jdbc:hive2://bigdataanalytics2-worker-shdp> use student3_16;
INFO  : Compiling command(queryId=hive_20210421154316_471c04a3-8688-43bd-8342-7ee060feeb2f): use student3_16
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:null, properties:null)
INFO  : Completed compiling command(queryId=hive_20210421154316_471c04a3-8688-43bd-8342-7ee060feeb2f); Time taken: 0.005
INFO  : Executing command(queryId=hive_20210421154316_471c04a3-8688-43bd-8342-7ee060feeb2f): use student3_16
INFO  : Starting task [Stage-0:DDL] in serial mode
INFO  : Completed executing command(queryId=hive_20210421154316_471c04a3-8688-43bd-8342-7ee060feeb2f); Time taken: 0.004
INFO  : OK
No rows affected (0.127 seconds)
0: jdbc:hive2://bigdataanalytics2-worker-shdp> create external table v_parquet
. . . . . . . . . . . . . . . . . . . . . . .> stored as parquet
. . . . . . . . . . . . . . . . . . . . . . .> as select * from vehicles_2 limit 1000;
INFO  : Compiling command(queryId=hive_20210421154332_a48f8ed2-7f30-46cf-a160-ff762fe93967): create external table v_parc
stored as parquet
as select * from vehicles_2 limit 1000
INFO  : Semantic Analysis Completed (retrial = false)
INFO  : Returning Hive schema: Schema(fieldSchemas:[FieldSchema(name:vehicles_2.r_id, type:string, comment:null), FieldSc
pe:string, comment:null), FieldSchema(name:vehicles_2.region, type:string, comment:null), FieldSchema(name:vehicles_2.reg
ment:null), FieldSchema(name:vehicles_2.year, type:date, comment:null), FieldSchema(name:vehicles_2.manufacturer, type:st
ieldSchema(name:vehicles_2.condition, type:string, comment:null), FieldSchema(name:vehicles_2.cylinders, type:int, commer
me:vehicles_2.odometer, type:float, comment:null), FieldSchema(name:vehicles_2.title_status, type:string, comment:null),
e:vehicles_2.vin, type:string, comment:null), FieldSchema(name:vehicles_2.drive, type:string, comment:null), FieldSchema(
ype:string, comment:null), FieldSchema(name:vehicles_2.paint_color, type:string, comment:null), FieldSchema(name:vehicles
pe:string, comment:null), FieldSchema(name:vehicles_2.state, type:string, comment:null), FieldSchema(name:vehicles_2.lat,
ull), FieldSchema(name:vehicles_2.posting_date, type:date, comment:null)], properties:null)
INFO  : Completed compiling command(queryId=hive_20210421154332_a48f8ed2-7f30-46cf-a160-ff762fe93967); Time taken: 0.114
INFO  : Executing command(queryId=hive_20210421154332_a48f8ed2-7f30-46cf-a160-ff762fe93967): create external table v_parc
stored as parquet
as select * from vehicles_2 limit 1000
INFO  : Query ID = hive_20210421154332_a48f8ed2-7f30-46cf-a160-ff762fe93967
INFO  : Total jobs = 1
INFO  : Launching Job 1 out of 1
INFO  : Starting task [Stage-1:MAPRED] in serial mode
INFO  : Subscribed to counters: [] for queryId: hive_20210421154332_a48f8ed2-7f30-46cf-a160-ff762fe93967
INFO  : Tez session hasn't been created yet. Opening session
```

Выводы.

В целом можно сказать, что если необходимо быстрое чтение данных и при этом не нужно заливать новые записи, то можно выбрать колоночные форматы хранения Parquet или ORC.

Строчный бинарный формат Avro используется для большой нагрузки по записи и целом нужен для быстрой записи данных.

По методам сжатия данных.

GZIP экономит много ресурсов HDD, но при этом тратит много ресурсов на обработку.

Snappy экономит среднее количество ресурсов HDD и тратит среднее количество ресурсов на обработку.