ALTER TABLE SUPPORT FOR CARBON DATA CONNECTOR

Objective

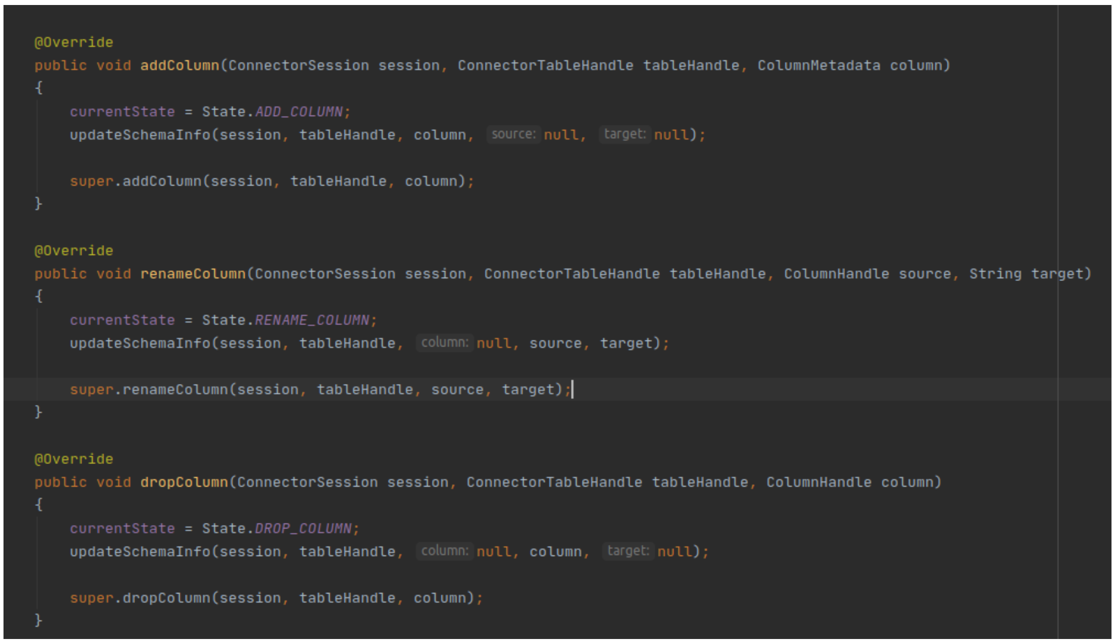
To support ALTER TABLE operations such as add column, rename column and drop column on a carbon data table.

Background

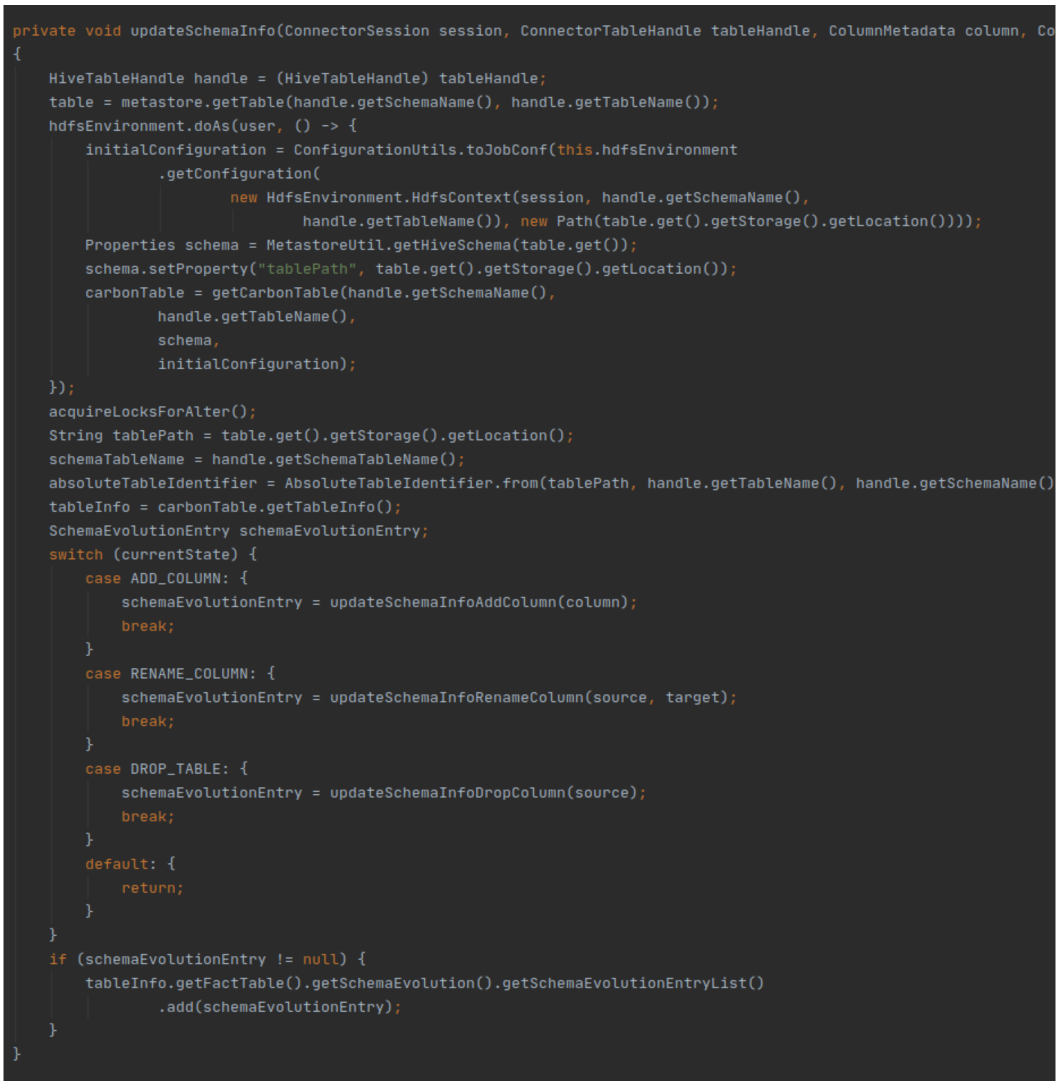
For a carbon data table, the schema of the table is stored in the schema file in the Metadata folder. Any change in the table schema needs to be also updated in the schema file.

Design Approach

For each alter table operation, the table metadata is edited in HiveMetadata.java by the hive flow. For a carbon data table, since the schema file also needs to be updated, additional changes are required in CarbondataMetadata to override the existing methods. For each alter table operation a method has been introduced which updated the schema and writes the updated schema into the schema file.

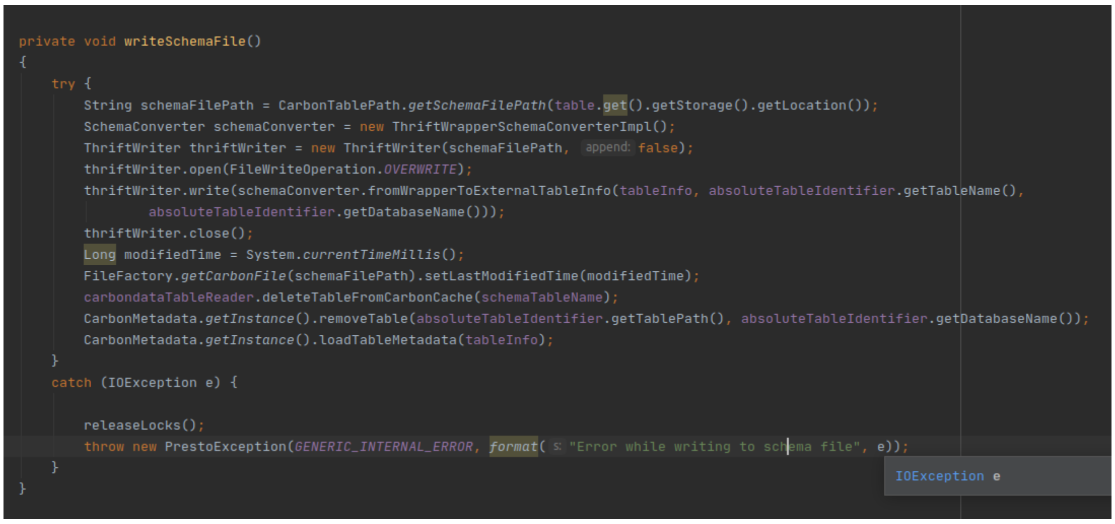


```
@Override
public void addColumn(ConnectorSession session, ConnectorTableHandle tableHandle, ColumnMetadata column)
{
    currentState = State.ADD_COLUMN;
    updateSchemaInfo(session, tableHandle, column, source: null, target: null);

    super.addColumn(session, tableHandle, column);
}

@Override
public void renameColumn(ConnectorSession session, ConnectorTableHandle tableHandle, ColumnHandle source, String target)
{
    currentState = State.RENAME_COLUMN;
    updateSchemaInfo(session, tableHandle, column: null, source, target);

    super.renameColumn(session, tableHandle, source, target);
}

@Override
public void dropColumn(ConnectorSession session, ConnectorTableHandle tableHandle, ColumnHandle column)
{
    currentState = State.DROP_COLUMN;
    updateSchemaInfo(session, tableHandle, column: null, column, target: null);

    super.dropColumn(session, tableHandle, column);
}
```



```java
private void updateSchemaInfo(ConnectorSession session, ConnectorTableHandle tableHandle, ColumnMetadata column, Co
{
    HiveTableHandle handle = (HiveTableHandle) tableHandle;
    table = metastore.getTable(handle.getSchemaName(), handle.getTableName());
    hdfsEnvironment.doAs(user, () -> {
        initialConfiguration = ConfigurationUtils.toJobConf(this.hdfsEnvironment
                .getConfiguration(
                        new HdfsEnvironment.HdfsContext(session, handle.getSchemaName(),
                                handle.getTableName()), new Path(table.get().getStorage().getLocation())));
        Properties schema = MetastoreUtil.getHiveSchema(table.get());
        schema.setProperty("tablePath", table.get().getStorage().getLocation());
        carbonTable = getCarbonTable(handle.getSchemaName(),
                handle.getTableName(),
                schema,
                initialConfiguration);
    });
    acquireLocksForAlter();
    String tablePath = table.get().getStorage().getLocation();
    schemaTableName = handle.getSchemaTableName();
    absoluteTableIdentifier = AbsoluteTableIdentifier.from(tablePath, handle.getTableName(), handle.getSchemaName()
    tableInfo = carbonTable.getTableInfo();
    SchemaEvolutionEntry schemaEvolutionEntry;
    switch (currentState) {
        case ADD_COLUMN: {
            schemaEvolutionEntry = updateSchemaInfoAddColumn(column);
            break;
        }
        case RENAME_COLUMN: {
            schemaEvolutionEntry = updateSchemaInfoRenameColumn(source, target);
            break;
        }
        case DROP_TABLE: {
            schemaEvolutionEntry = updateSchemaInfoDropColumn(source);
            break;
        }
        default: {
            return;
        }
    }
    if (schemaEvolutionEntry != null) {
        tableInfo.getFactTable().getSchemaEvolution().getSchemaEvolutionEntryList()
                .add(schemaEvolutionEntry);
    }
}
```

Metadata and compaction locks are acquired in the beginning of the alter table operation. The table schema is updated and schema evolution is added to the schema evolution entry list of the carbon table. The updated schema is then written in to the schema file on commit.



```java
private void writeSchemaFile()
{
    try {
        String schemaFilePath = CarbonTablePath.getSchemaFilePath(table.get().getStorage().getLocation());
        SchemaConverter schemaConverter = new ThriftWrapperSchemaConverterImpl();
        ThriftWriter thriftWriter = new ThriftWriter(schemaFilePath,  append: false);
        thriftWriter.open(FileWriteOperation.OVERWRITE);
        thriftWriter.write(schemaConverter.fromWrapperToExternalTableInfo(tableInfo, absoluteTableIdentifier.getTableName(),
                absoluteTableIdentifier.getDatabaseName()));
        thriftWriter.close();
        Long modifiedTime = System.currentTimeMillis();
        FileFactory.getCarbonFile(schemaFilePath).setLastModifiedTime(modifiedTime);
        carbondataTableReader.deleteTableFromCarbonCache(schemaTableName);
        CarbonMetadata.getInstance().removeTable(absoluteTableIdentifier.getTablePath(), absoluteTableIdentifier.getDatabaseName());
        CarbonMetadata.getInstance().loadTableMetadata(tableInfo);
    }
    catch (IOException e) {

        releaseLocks();
        throw new PrestoException(GENERIC_INTERNAL_ERROR, format( s: "Error while writing to schema file", e));
    }
}
```

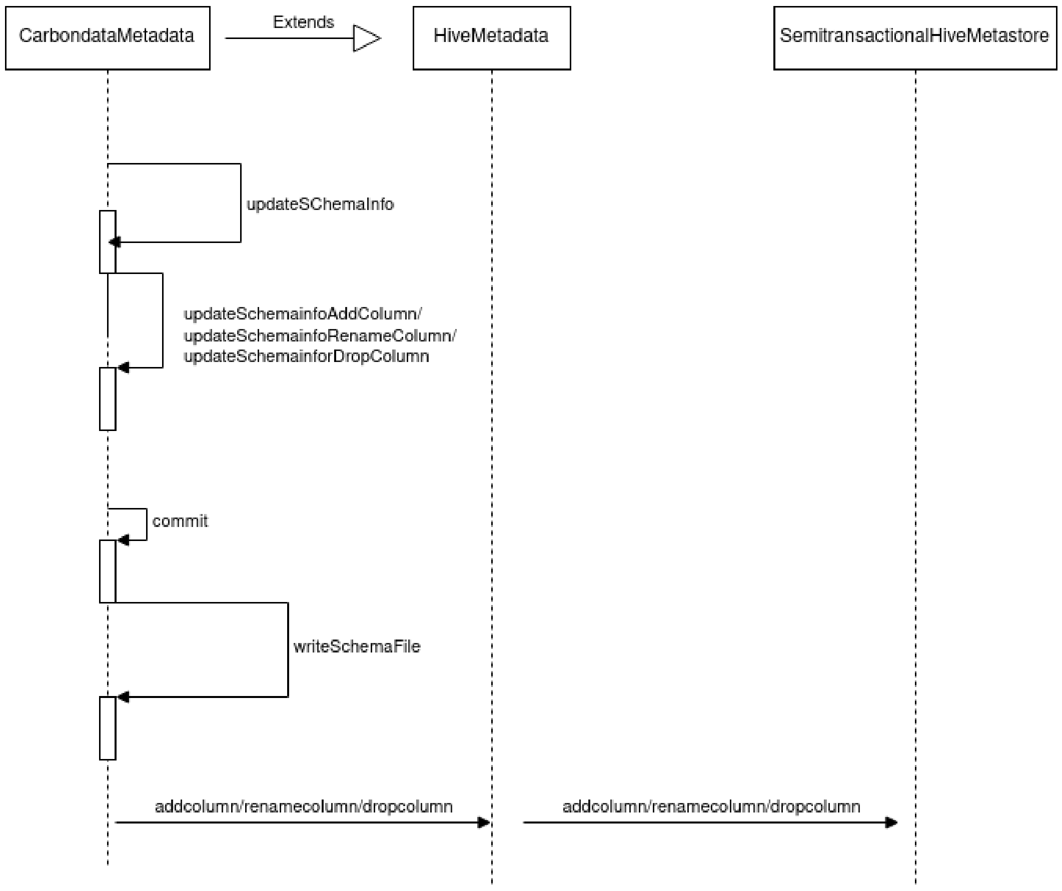

CarbondataMetadata
Extends
HiveMetadata
SemitransactionalHiveMetastore
updateSChemaInfo
updateSchemainfoAddColumn/
updateSchemainfoRenameColumn/
updateSchemainforDropColumn
commit
writeSchemaFile
addcolumn/renamecolumn/dropcolumn
addcolumn/renamecolumn/dropcolumn

Figure 1: Flow diagram for Alter table