DatasourceCustomProvider

BIsql BI

DatasourceCustomProvider

```
package com.finebi.api.sql;


import com.finebi.api.dialect.DialectGenerator;
import com.finebi.api.sql.database.DatabaseSqlTranslator;
import com.finebi.api.sql.temporary.CrossLockManager;
import com.finebi.api.sql.temporary.TemporaryDialect;
import com.finebi.base.tuple.Pair;
import com.fr.common.annotations.Open;
import com.fr.data.core.db.dialect.Dialect;
import com.fr.stable.fun.mark.Mutable;

import java.util.List;
import java.util.function.BiFunction;
import java.util.function.Predicate;

/**
 * @author kris
 * @version 5.1.3
 * Created by kris on 2021/5/26
 */
@Open
public interface DatasourceCustomProvider extends Mutable {

    String XML_TAG = "DatasourceCustomProvider";

    int CURRENT_LEVEL = 1;

    /**
     * Sql
     */
    BaseSqlTranslator getBaseSqlTranslator();

    /**
     *
     * Predicate<Dialect>:
     * BiFunction<Dialect, CrossLockManager, TemporaryDialect> DialectTemporaryDialectCrossLockManager
CrossLockFactory.getSingleton()
     */
    List<Pair<Predicate<Dialect>, BiFunction<Dialect, CrossLockManager, TemporaryDialect>>>
getTemporaryDialectList();


    /**
     * dbsql
     *
     * @return
     */
    DatabaseSqlTranslator getDatabaseSqlTranslator();

    /**
     *
     *
     * @return
     */
    DialectGenerator getDialectGenerator();
}
```

BaseSqlTranslator

```
@Open
public interface BaseSqlTranslator {

    /**
     * BaseCriteria SQL
     *
     * @param dialect
     * @param baseCriteria baseCriteria
     * @return sql
     */
    String translate(Dialect dialect, BaseCriteria baseCriteria);

    /**
     * baseCriteriasql
     *
     * @param dialect
     * @return
     */
    boolean supportWithBaseCriteria(Dialect dialect);

    /**
     * Table SQL
     *
     * @param dialect
     * @param table   table
     * @return sql
     */
    String translate(Dialect dialect, Table table);

    /**
     * tablesql
     *
     * @param dialect
     * @return
     */
    boolean supportWithTable(Dialect dialect);

}
```

BaseSqlTranslatorsqlBaseCriteriaBaseSqlTranslatorBaseCriteriasqlTableBaseCriteria

BaseCriteriaBaseCriteria

TemporaryDialect

```
@Open
public interface TemporaryDialect {
    /**
     *
     */
    boolean checkTableValid(CrossConnect connection, String tableName) throws SQLException;

    /**
     *
     */
    boolean checkTableExist(CrossConnect connection, String tableName) throws SQLException;

    /**
     * SQLSERVER##
     * tableName
     * tableName
     */
    String handleTemporaryTableName(String tableName) throws Exception;

    /**
     * sql -> ""
     */
    String column2SQL(String columnName);

    /**
     * sqlT_A -> `T_A`
     */
    String table2SQL(String tableName);

    /**
     * ' -> ''
     */
    String escapeStringConstant(String constant);

    /**
     *
     */
    boolean supportTransaction();

    /**
     *
     */
    void createTempTable(CrossConnect connection, String tableName, List<Field> fields, String collate) throws
SQLException;

    /**
     *
     */
    @Nullable
    String analysisCollate(CrossConnect connection) throws SQLException;

    /**
     * excel
     */
    void insertData(CrossConnect connection, String tableName, List<Field> fields, RowIterator rowIterator)
throws SQLException;

    /**
     *
     */
    default TemporaryLifeCircle lifeCircle() {
        return new TemporaryLifeCircle() {
        };
    }
}
```

TemporaryDialectExcelsql

DatabaseSqlTranslator

```
@Open
public interface DatabaseSqlTranslator {

    /**
     * dbsql
     *
     * @param connection  (connection)
     * @param schema
     * @param dbTableName
     * @return
     */
    String translate(Connection connection, String schema, String dbTableName);

    /**
     *
     *
     * @param dialect
     * @return
     */
    boolean supportWithDialect(Dialect dialect);

}
```

DatabaseSqlTranslatorschemaSQL select * from tableName

DialectGenerator

```
@Open
public interface DialectGenerator {

    /**
     * Connection
     *
     * @param connection
     * @return
     */
    Dialect getDialectFromConnection(Connection connection);

}
```

DialectGenaratorJDBCDialect

DialectDialectBIDialect

DialectDefaultDialectSchemaDialectschemaschemaDialect

DialectDialect Development Kit Dialect

```
<extra-core>
   <DatasourceCustomProvider class="" />
</extra-core>
```

AbstractExportHandleProvider

```
package com.finebi.provider.api.sql;

import com.finebi.api.sql.DatasourceCustomProvider;
import com.fr.stable.fun.mark.API;

/**
 * sql
 *
 * @author kris
 * @version 5.1.3
 * Created by kris on 2021/6/16
 */
@API(level = DatasourceCustomProvider.CURRENT_LEVEL)
public abstract class AbstractDatasourceCustomProvider implements DatasourceCustomProvider {

    @Override
    public int currentAPILevel() {
        return DatasourceCustomProvider.CURRENT_LEVEL;
    }

    @Override
    public String mark4Provider() {
        return getClass().getName();
    }
}
```