

connect(): Connection - disconnect(ofbConnection): boolean - connect(): Connection - disconnect(ofbConnection): boolean - testConnection(): boolean - testConnection(): boolean - connect(): Connection - disconnect(dbConnection): connection(): boolean - testConnection(): boolean

+getAllRawTableNames(): List<String> + tableExists(tableName: String): boolean

+ getTableData(tableName: String): List<List<Object>> + getTableFields(tableName: String): List<String[]>

+ getPivotTableData(pvtTblSchema: PivotTableSchema): List<List<Object>>

Gliffy / *Class Diagram: Entities, v1 🔒

DataField

- name: String
 type: DataType
 alias: String
- + DataField(name: String, type: DataType, alias: String)
- + DataField(name: String, type: DataType)
- + equals(other: Object): boolean
- + hashCode(): int getName(): String getType(): DataType getAlias(): String toJson(): String

<<enumeration>> DataType

val: String

- DataType(val: String) + toString(): String
- + getDataType(val: String): DataType

DataSet

- columns: List<DataField>
- rows: List<List<Object>>
- + DataSet()
- + DataSet(columns: List<DataField>, rows: List<List<Object>>)
- + getRowAt(index: int): List<Object>
- + setColumns(fields: List<DataField>): void
- + addField(field: DataField): void
- + insertRow(row: List<Object>): void
- + getRows(): List<List<Object>>
- + getColumns(): List<DataField>
- + getSize(): int
- + toJSON(): String

PivotTable

- schema: PivotTableSchema
- data: List<List<Object>>
- + PivotTable()
- + PivotTable(schema: PivotTableSchema, data: List<List<Object>>)
- + toJSON(): String
- + getSchema(): PivotTableSchema
- + setSchema(PivotTable: schema): void
- + getData(): List<List<Object>>
- + setData(data: List<List<Object>>): voic

PivotTableSchema

- tableName: String
- columnLabels: List<String>
- rowLabels: List<String>
- pageLabel: String
 functionName: String
- valueField: String
- + PivotTableSchema(tableName: String, columnLabels: List<String>,
- rowLabels: List<String>, pageLabel: String, functionName: String, valueField: String)
- + fromJSON(json: String): PivotTableSchema
- + getTableName(): String
- + setTableName(tableName: String)
- + getColumnLabels(): List<String>
- + setColumnLabels(columnName: List<String>): void
- + getRowLabels(): List<String>
- + setRowLabels(rowName: List<String>): void
- + getPageLabel(): String
- + setPageLabel(pageLabel: String): void
- + getFunctionName(): String
- + setFunctionName(functionName: String)
- + getValueField(): String
- + setValueField(valueField: String): void

'Class Diagram: Service Layer, v1 🔒

<<interface>> DataRetrievalService

checkDataSourceConnection(): boolean getAllRawReportNAmes(): List<String> rawReportExists(reportName: String): boolean getRawReport(reportName: String): DataSet

getPivotTable(pvtTblSchema: PivotTableSchema): PivotTable

DataRetrievalServiceImpl

- dataSource: DataSourceAccess
- + DataRetrievalServiceImpl(dataSource: DataSourceAccess)
- +checkDataSourceConnection(): boolean
- + getAllRawReportNAmes(): List<String>
- + rawReportExists(reportName: String): boolean
- + getRawReport(reportName: String): DataSet
- + getPivotTable(pvtTblSchema: PivotTableSchema): PivotTable

Class Diagram: Utils, v1 🔒

ControllerFactory

- container: DependenciesContainer
- + ControllerFactory(container: DependenciesContainer)
- + getController(clazz: Class<T>): <T extends Controller> T

DependenciesContainer

- container: HashMap<String, Object>
- log: Logger
- + DependenciesContainer()
- wireDependencies(): void
- wireDependency(name: String): Object
- + get(className: String): <T> T

Application

- GET: Map<String, Class>
- POST: Map<String, Class>
- + main(): void
- defineRoutes(): void