一、功能说明

该文档的目的是是介绍一种可实施且通用的数据权限控制方案, 以及工具包: tools-mate的用法。

背景

- 随着后台管理系统的越来越多, 功能也越来越丰富, 对职员数据级的权限管控迫在眉睫

  > UAC应运而生, 她将数据权限的配置已经界面化管理, 但在业务系统中具体实施尚未接入..

声明

- 已接入UAC的应用可以不关注以下内容, 只需要在工具包中引入以下两个jar包

  > 该操作可以集成登陆校验与权限控制

  ```
  <!-- 权限控制SDK -->
  <dependency>
    <groupId>cn.huolala</groupId>
    <artifactId>user-auth-sdk</artifactId>
    <version>sdk-1.1.0</version>
  </dependency>
  <!-- 登陆校验SDK -->
  <dependency>
    <groupId>cn.huolala</groupId>
    <artifactId>user-sso-sdk</artifactId>
    <version>sdk-1.0.2</version>
  </dependency>
  ```

二、工具使用

- 添加依赖: pom.xml

  ```
  <!-- 读写分离依赖包 -->
  <dependency>
      <groupId>cn.hll.tools</groupId>
      <artifactId>tools-mate</artifactId>
      <version>4.4.3</version>
  </dependency>
  ```

- 启用工具

  > 在启动类上或其他配置类上扫描包: cn.hll.tools

```java
@SpringBootApplication
@ComponentScan("cn.hll.tools")
public class XxxApp {
    public static void main(String[] args) {
        SpringApplication.run(XxxApp.class, args);
    }
}
```

- 实现接口: cn.hll.tools.mate.interceptor.DataAuthServer

> 以下代码是示范代码, 不可上生产

```java
@Bean
public DataAuthServer dataAuthServer(){
    return new DataAuthServer() {
      @Override
        public String getWherein(String flag, String fields) {
            return "1";
        }

        @Override
        public boolean filter(String flag, String field) {
            return false;
        }
    };
}
```

- 添加注解: @DataAuth (建议在Mapper上加)

> flag: 表示权限树名(比如: 组织结构、行政区域、系统列表...)

> fields: 表示权限字段名, 换言之是用于控制数据返回的条件字段

> sql: 表示自定义的权限控制, 它可以和fields同时使用 (仅限行权限时)

```java
  @DataAuth(
      row = {@DataFilter(flag = "row_sys_auth", fields = "app_id", sql = "right join sys s on s.id = tab.app_id")},
      column = {@DataFilter(flag = "column-auth-user", fields = "app_id")}
  )
  @Select("select log.* from logger log left join user u on log.create_by = u.id where log.host = #{host}")
  Page<Logger> list(Page page, @Param("host") String host);
```

三、工具原理

逻辑

- 整个工具围绕着mybatis的拦截器Interceptor完成
- 其中行数据权限会在查询sql执行前拦截并根据需要改造最终执行的sql
- 其中列数据权限会在查询ResultSet(结果集)返回前拦截并置空没有权限的field
- 而具体的数据范围则抽象到实现类中可以由业务系统提供

  > 此做法可以将数据权限控制, 与数据权限配置解耦, 好处不言而喻

代码

- 拦截器类名: cn.hll.tools.mate.interceptor.DataAuthInterceptor
- 使用了Executor拦截器监听了query方法实现对行权限控制的sql改造
- 使用了ResultSetHandler拦截器监听了handleResultSets方法对查询的结果集进行过滤

```
private Object dataAuth(Invocation invocation) throws Exception {
    Method method = invocation.getMethod();
    switch (method.getName()) {
        case "query": {
            return rowDataAuth(invocation);
        }
        case "handleResultSets": {
            return columnDataAuth(invocation);
        }
    }
    return null;
}
```

数据

- 此工具没有产生额外数据

四、使用示例

- 启动类

```
@SpringBootTest
@RunWith(SpringJUnit4ClassRunner.class)
public class LoggerTest {

    @Resource
    LoggerMapper loggerMapper;

    @Test
    public void list() throws Exception {
        PageRequest pr = new PageRequest();
        Page<Logger> page = loggerMapper.list(Page.of(pr.current, pr.size),
"127.0.0.1");
        System.out.println(JsonUtil.format(new PageResponse(pr,
page.getTotal(), page.getRecords())));
```

```
    }

}
```

- 实现类

```
@Bean
public DataAuthServer dataAuthServer(){
    return new DataAuthServer() {
        @Override
        public boolean printSql() {
            return true;
        }

        @Override
        public String getWherein(String flag, String fields) {
            return "1";
        }

        @Override
        public boolean filter(String flag, String field) {
            return false;
        }
    };
}
```

- 执行效果 -> 加了注解: @DataAuth

```
2021-12-10 18:35:10.760  [main            ] DEBUG
cn.hll.tools.mate.assist.SqlResetAssist           :33   - 数据权限拦截前:
select log.* from logger log left join user u on log.create_by = u.id where
log.host = ?
2021-12-10 18:35:10.766  [main            ] DEBUG
cn.hll.tools.mate.assist.SqlResetAssist           :43   - 数据权限拦截前:
select tab.* from (select tab.* from (select log.* from logger log left join
user u on log.create_by = u.id where log.host = ?) tab right join sys s on
s.id = tab.app_id) tab where  tab.app_id in (1)
2021-12-10 18:35:10.811  [main            ] INFO
cn.hll.tools.ds.dynamic.DynamicDataSourceHolder   :36   - 数据源: master
2021-12-10 18:35:10.950  [main            ] INFO
com.alibaba.druid.pool.DruidDataSource            :985  - {dataSource-1}
inited
2021-12-10 18:35:11.098  [main            ] DEBUG
c.h.t.mate.dao.mapper.LoggerMapper.list_mpCount   :137  - ==>  Preparing:
SELECT COUNT(*) FROM (SELECT tab.* FROM (SELECT log.* FROM logger log LEFT
JOIN user u ON log.create_by = u.id WHERE log.host = ?) tab RIGHT JOIN sys s
ON s.id = tab.app_id) tab WHERE tab.app_id IN (1)
2021-12-10 18:35:11.122  [main            ] DEBUG
c.h.t.mate.dao.mapper.LoggerMapper.list_mpCount   :137  - ==> Parameters:
127.0.0.1(String)
2021-12-10 18:35:11.144  [main            ] DEBUG
c.h.t.mate.dao.mapper.LoggerMapper.list_mpCount   :137  - <==      Total: 1
```

```
2021-12-10 18:35:11.151  [main           ] DEBUG
cn.hll.tools.mate.dao.mapper.LoggerMapper.list    :137  - ==>  Preparing:
select tab.* from (select tab.* from (select log.* from logger log left join
user u on log.create_by = u.id where log.host = ?) tab right join sys s on
s.id = tab.app_id) tab where tab.app_id in (1) LIMIT ?
2021-12-10 18:35:11.151  [main           ] DEBUG
cn.hll.tools.mate.dao.mapper.LoggerMapper.list    :137  - ==> Parameters:
127.0.0.1(String), 5(Long)
2021-12-10 18:35:11.158  [main           ] DEBUG
cn.hll.tools.mate.dao.mapper.LoggerMapper.list    :137  - <==      Total: 2
{
    "current":1,
    "size":5,
    "total":2,
    "list":[
        {
            "id":2,
            "appId":1,
            "appCode":"2",
            "appName":"2",
            "host":"127.0.0.1",
            "port":"2",
            "env":"2",
            "content":"2",
            "createBy":2,
            "updateBy":2,
            "deleteTag":2,
            "updatedAt":1639115163000,
            "createdAt":1638956868000
        },
        {
            "id":3,
            "appId":1,
            "appCode":"3",
            "appName":"3",
            "host":"127.0.0.1",
            "port":"3",
            "env":"3",
            "content":"3",
            "createBy":3,
            "updateBy":3,
            "deleteTag":3,
            "updatedAt":1639115163000,
            "createdAt":1639115011000
        }
    ]
}
```

- 执行效果 -> 没加注解: @DataAuth

```
2021-12-10 18:37:37.595  [main           ] INFO
cn.hll.tools.ds.dynamic.DynamicDataSourceHolder   :36   - 数据源: master
2021-12-10 18:37:37.715  [main           ] INFO
com.alibaba.druid.pool.DruidDataSource            :985  - {dataSource-1}
inited
```

```
2021-12-10 18:37:37.868  [main           ] DEBUG
c.h.t.mate.dao.mapper.LoggerMapper.list_mpCount   :137  - ==>  Preparing:
SELECT COUNT(*) FROM logger log LEFT JOIN user u ON log.create_by = u.id
WHERE log.host = ?
2021-12-10 18:37:37.889  [main           ] DEBUG
c.h.t.mate.dao.mapper.LoggerMapper.list_mpCount   :137  - ==> Parameters:
127.0.0.1(String)
2021-12-10 18:37:37.909  [main           ] DEBUG
c.h.t.mate.dao.mapper.LoggerMapper.list_mpCount   :137  - <==      Total: 1
2021-12-10 18:37:37.915  [main           ] DEBUG
cn.hll.tools.mate.dao.mapper.LoggerMapper.list    :137  - ==>  Preparing:
select log.* from logger log left join user u on log.create_by = u.id where
log.host = ? LIMIT ?
2021-12-10 18:37:37.915  [main           ] DEBUG
cn.hll.tools.mate.dao.mapper.LoggerMapper.list    :137  - ==> Parameters:
127.0.0.1(String), 5(Long)
2021-12-10 18:37:37.922  [main           ] DEBUG
cn.hll.tools.mate.dao.mapper.LoggerMapper.list    :137  - <==      Total: 3
{
    "current":1,
    "size":5,
    "total":3,
    "list":[
        {
            "id":1,
            "appId":1,
            "appCode":"1",
            "appName":"1",
            "host":"127.0.0.1",
            "port":"1",
            "env":"1",
            "content":"1",
            "createBy":1,
            "updateBy":1,
            "deleteTag":1,
            "updatedAt":1639132647000,
            "createdAt":1638951441000
        },
        {
            "id":2,
            "appId":1,
            "appCode":"2",
            "appName":"2",
            "host":"127.0.0.1",
            "port":"2",
            "env":"2",
            "content":"2",
            "createBy":2,
            "updateBy":2,
            "deleteTag":2,
            "updatedAt":1639115163000,
            "createdAt":1638956868000
        },
        {
            "id":3,
            "appId":1,
            "appCode":"3",
            "appName":"3",
                "host":"127.0.0.1",
                "port":"3",
                "env":"3",
                "content":"3",
                "createBy":3,
                "updateBy":3,
                "deleteTag":3,
                "updatedAt":1639115163000,
                "createdAt":1639115011000
            }
        ]
}
```

数据库的数据

```
id, app_id, app_code, app_name, host, port, env, content, create_by, update_by, delete_tag, updated_at, created_at
1   1   1   1   127.0.0.1   1   1   1   1   1   1   2021-12-10 18:37:27 2021-12-08 16:17:21
2   1   2   2   127.0.0.1   2   2   2   2   2   2   2021-12-10 13:46:03 2021-12-08 17:47:48
3   1   3   3   127.0.0.1   3   3   3   3   3   3   2021-12-10 13:46:03 2021-12-10 13:43:31
```