数据审计与数据权限

> 本章节主要描述了在第三代技术架构中，如何统一处理数据变动的审计以及如何设计数据权限的内容。

数据审计

开发手册

增加配置项。添加AuditEntityScan注解，值为实体所在的包路径。

```
@Configuration
@MapperScan("com.acca.opra.sys.dao")
@AuditEntityScan("com.acca.opra.sys.domainmodel")
public class MybatisConfiguration extends SimpleMybatisplusConfiguration {

}
```

在需要记录数据修改变更的实体上添加@AuditEntity注解，如下表中样例代码：

```
@Data
@EqualsAndHashCode(callSuper = true)
@Accessors(chain = true)
@AuditEntity
public class MasCurrency extends AbstractAuditableEntity {

    private static final long serialVersionUID = 1L;

    /**
     *
     */
    @TableField("decimal_units")
    private Integer decimalUnits;

    /**
```

```
     *
     */
    @TableField("description")
    private String description;

    /**
     * Currency Code Alpha
     */
    @TableField("curr_code")
    private String currCode;

    /**
     * Currency Code Numeric
     */
    @TableField("curr_numeric")
    private String currNumeric;

    /**
     *
     */
    @TableField("local_curr_fare")
    private BigDecimal localCurrFare;

    /**
     *
     */
    @TableField("other_charge")
    private BigDecimal otherCharge;

    @TableId(value = "curr_id", type = IdType.ASSIGN_ID)
    private String currId;

    /**
     *
     */
    @TableField("curr_chn_name")
    private String currChnName;

    /**
     *
     */
    @TableField("curr_eng_name")
    private String currEngName;


}
```

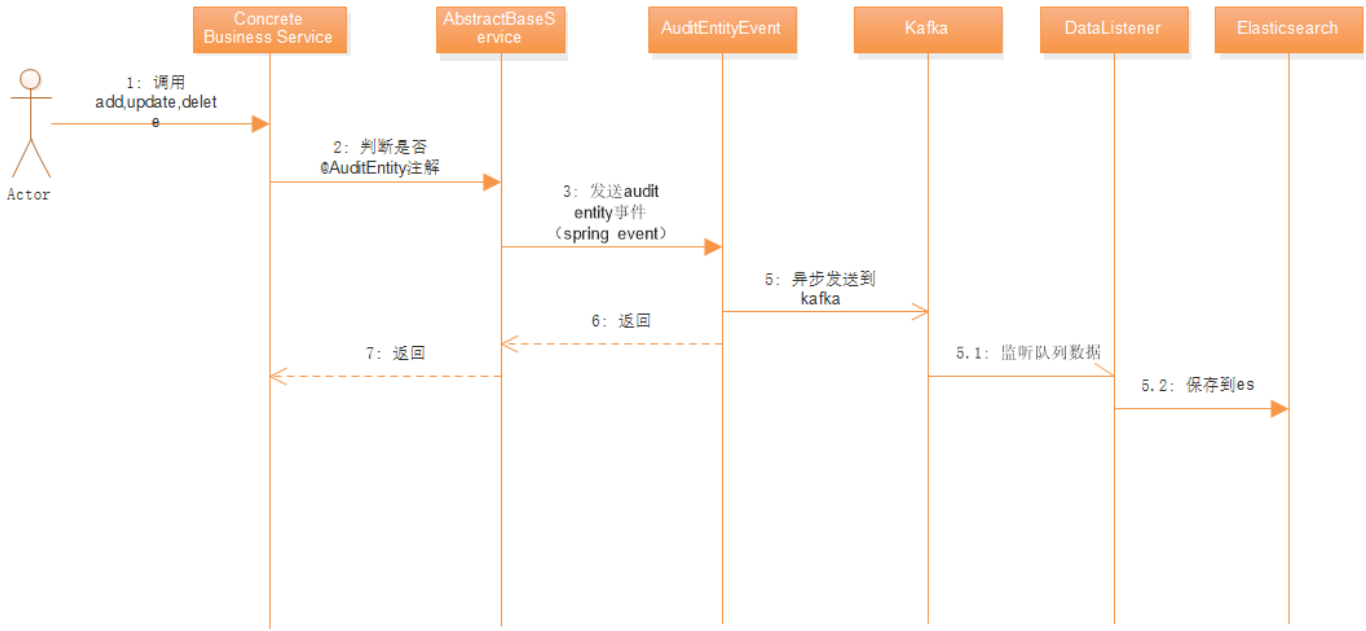
数据修改的记录项

```
    @Data
@AllArgsConstructor
@NoArgsConstructor
public static class AuditEntityWrapper {

    private String entity; // json

    private String before; // json

    private ModifyType modifyType;

    @Id
    private String id;

    private Date modifyDate;

    private String userId;

    private String tableName;
}
```

查看修改记录

系统管理模块会提供查看数据修改记录的api。根据上面的字段，任意查询。（开发中）

详细设计



数据权限

概述

> 采用拦截sql，并增加相应数据权限的查询条件的方式。
>
> 开发人员无需写死权限code，数据权限的匹配完全通过api中定义的uri来指定。
>
> 为了保证sql被修改后，能够正常运行，增加了表与字段的判断。项目启动时，会记录所有实体与字段的关联，判断数据权限时，根据sql中的表名，判断数据权限中的字段是否是表中的字段，保证sql能顺利执行。

开发手册

环境配置

增加配置项。添加AuditEntityScan注解，值为实体所在的包路径。

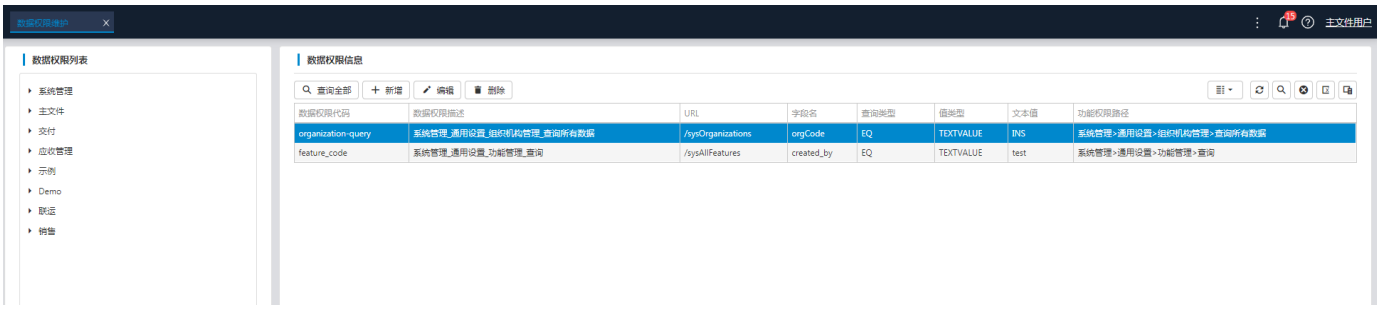
```
@Configuration
@MapperScan("com.acca.opra.sys.dao")
@AuditEntityScan("com.acca.opra.sys.domainmodel")
public class MybatisConfiguration extends SimpleMybatisplusConfiguration {

}
```

自定义数据权限接口

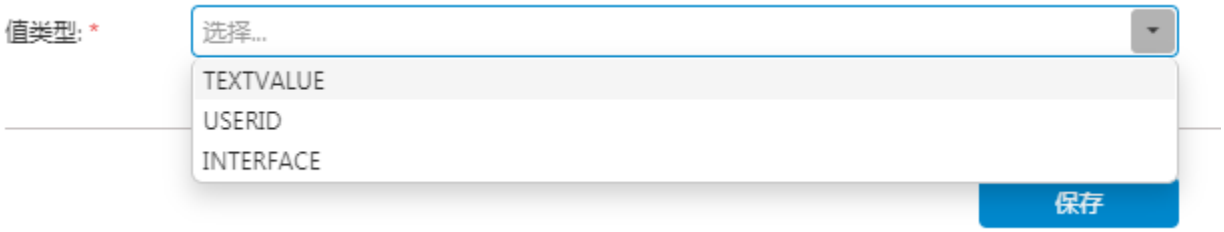
自定义数据权限的逻辑，需要实现数据权限接口

```
        /**
 *
 *
 * @version OPRA v1.0
 * @author Yu Tao, 2020317
 */
public interface DataPermissionSelector {

    /**
     * id
     * @param userId String
     * @return object
     */
    Object getDataPermissionValue(String userId);
}
```

页面维护数据权限



选中左侧功能树中，需要控制数据权限的节点。点击右侧添加按钮。



上面三种类型对应为:

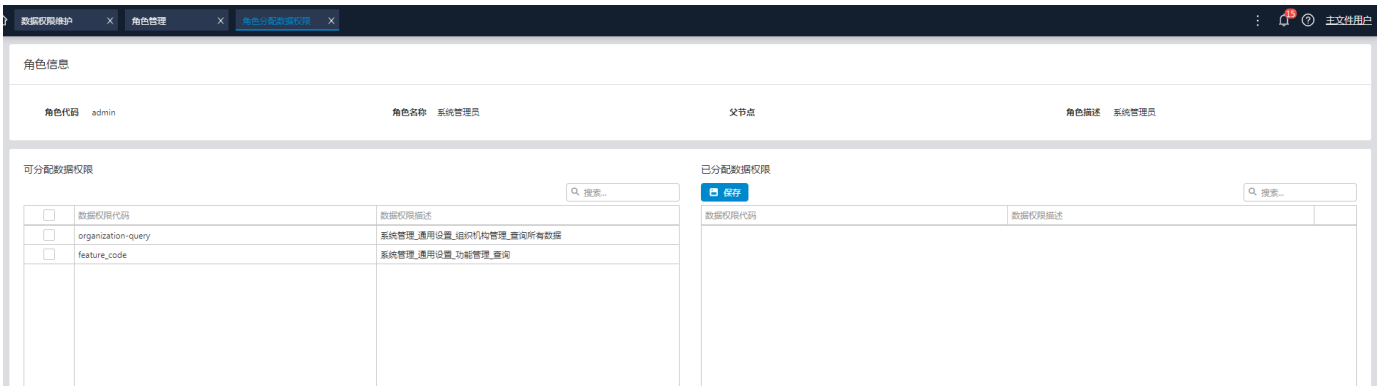
TEXTVALUE: 自己书写的固定值

USERID：系统自动获取当前用户id

INTERFACE： 上面自定义的接口实现。返回值类型为List的话，需要选择In NotIn的操作符

```
/**
 * .
 *
 * null
 *  empty list  column_name in ('')
 *
 * @version OPRA v1.0
 * @author Yu Tao, 2020416
 */
@Component
public class ManageAreaDataPermissionSelector implements
DataPermissionSelector {

    @Autowired
    private SysUserApi sysUserApi;

    @Override
    public Object getDataPermissionValue(String userId) {

        SysUserVO user = sysUserApi.findOne(userId);
        if(!ArafStringUtils.isEmpty(user.getDescription())) {
            return Arrays.asList(ArafStringUtils.split(user.
getDescription(), ","));
        }
        return Collections.emptyList();
    }

}
```

角色添加数据权限



角色页面点击分配数据权限按钮，为当前角色分配数据权限

> 当同一api中对应了多个数据权限，则采用，相同字段用or，不同字段用and的方式，拼接sql