阿里云
MaxCompute

安全指南

文档版本：20190125

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的”现状“、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含”阿里云”、Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告:** 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明:** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 确定。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
*## *	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *[-all|-t]*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *{stand | slave}*

目录

1 安全功能详解

1.1 目标用户

本章节文档主要面向 MaxCompute 项目空间所有者（Owner）、管理员以及对 MaxCompute 多租户数据安全体系感兴趣的用户。

MaxCompute 多租户的数据安全体系，主要包括如下内容：

- 用户认证。
- 项目空间的用户与授权管理。
- 跨项目空间的资源分享。
- 项目空间的数据保护。

使用限制

- 列级别访问控制使用限制
- 项目空间的数据保护限制

1.2 快速开始

1.2.1 添加用户并授权

本文向您介绍如何添加项目成员并通过ACL授权。

- 场景描述：

  Jack是项目空间prj1的管理员，一个新加入的项目组成员Alice（已拥有云账号: alice@aliyun.com）申请加入项目空间prj1。需要申请如下权限：查看Table列表，提交作业，创建表。

- 操作步骤：（由项目空间管理员来操作）

```
use prj1;
add user aliyun$alice@aliyun.com;  --添加用户
grant List, CreateTable, CreateInstance on project prj1 to user
aliyun$alice@aliyun.com; --使用grant语句对用户授权
```

1.2.2 添加角色并通过ACL授权

本文向您介绍如何添加项目角色并通过ACL授权。

- 场景描述：Jack是项目空间prj1的管理员，有三个新加入的项目组成员：Alice, Bob, Charlie，他们的角色是数据审查员。他们要申请如下权限：查看Table列表，提交作业，读取

表userprofile。对于这个场景的授权，项目空间管理员可以使用基于对象的ACL授权 机制来完成。

- 操作方法：

```
use prj1;
    add user aliyun$alice@aliyun.com; --添加用户
    add user aliyun$bob@aliyun.com;
    add user aliyun$charlie@aliyun.com;
    create role tableviewer; --创建角色
    grant List, CreateInstance on project prj1 to role tableviewer;
 --对角色赋权
    grant Describe, Select on table userprofile to role tableviewer;
    grant tableviewer to aliyun$alice@aliyun.com; --对用户赋予角色
tableviewer
    grant tableviewer to aliyun$bob@aliyun.com;
    grant tableviewer to aliyun$charlie@aliyun.com;
```

1.2.3 设置项目保护模式

本文向您介绍如何设置项目保护模式。

- 场景描述：Jack是项目空间prj1的管理员。该项目空间有很多敏感数据，比如用户身份号码和购物记录。而且还有很多具有自主知识产权的数据挖掘算法。Jack希望能将项目空间中的这些敏感数据和算法保护好，项目中用户只能在项目空间中访问，数据只能在项目空间内流动，不允许流出到项目空间之外。
- 操作方法：Jack需要如下操作

```
use prj1;
set ProjectProtection=true; --开启项目空间的数据保护机制
```

一旦当项目空间开启项目空间的数据保护机制后，无法将项目空间中的数据转移到项目空间之外，所有的数据只能在项目空间内部流动。

但是在某些情况下，可能由于业务需要，用户Alice需要将某些数据表导出到项目空间之外，并且也经过空间管理员的审核通过。针对这类情况，MaxCompute提供了一种机制来支持受保护项目空间的数据流出，即设置TrustedProject。将prj2设置为prj1的可信项目空间，设置后将允许prj1中的所有数据流出到prj2。

```
use prj1;
```

```
add trustedproject prj2;
```

1.3 列级别访问控制

基于标签的安全（LabelSecurity）是项目空间级别的一种强制访问控制策略（Mandatory Access Control, MAC），它的引入是为了让项目空间管理员能更加灵活地控制用户对列级别敏感数据的访问。

理解MaxCompute中的MAC与DAC差异

在MaxCompute中，强制访问控制机制(MAC)独立于自主访问控制机制(DAC)。为了便于理解，MAC与DAC的关系可以用下面的例子来做个类比。

对于一个国家来说（类比MaxCompute的一个项目空间），这个国家公民要想开车(类比读数据操作)，必须先申请获得驾照(类比申请SELECT权限)。这些就属于DAC考虑的范畴。

但由于这个国家交通事故率一直居高不下，于是该国新增了一条法律：禁止酒驾。此后，所有想开车的人除了持有驾照之外，还必须不能喝酒。类比MaxCompute，这个禁止酒驾就相当于禁止读取敏感度高的数据。这就属于MAC考虑的范畴。

数据的敏感等级分类

LabelSecurity需要将数据和访问数据的人进行安全等级划分。在政府和金融机构，一般将数据的敏感度标记分为四类：0级 (不保密, Unclassified), 1级 (秘密, Confidential), 2级 (机密, Sensitive), 3级 (高度机密, Highly Sensitive)。MaxCompute也遵循这一分类方法。ProjectOwner需要定义明确的数据敏感等级和访问许可等级划分标准，默认时所有用户的访问许可等级为0级，数据安全级别默认为0级。

LabelSecurity对敏感数据的粒度可以支持列级别，管理员可以对表的任何列设置敏感度标记(Label)，一张表可以由不同敏感等级的数据列构成。

对于view，也支持和表同样的设置，即管理员可以对view设置label等级，view的等级和它对应的基表的label等级是独立的，在view创建时，默认的等级也是0。

LabelSecurity默认安全策略

在对数据和user分别设置安全等级标记之后，LabelSecurity的默认安全策略如下：

- (No-ReadUp) 不允许user读取敏感等级高于用户等级的数据，除非有显式授权。
- (Trusted-User) 允许user写任意等级的数据，新创建的数据默认为0级（不保密）。

说明:

- 在一些传统的强制访问控制系统中，为了防止数据在项目空间内部的任意分发，一般还支持更多复杂的安全策略，比如不允许用户写敏感等级不高于用户等级的数据(No-WriteDown)。但在MaxCompute平台中，考虑到项目空间管理员对数据敏感等级的管理成本，默认安全策略并不支持No-WriteDown. 如果项目空间管理员有类似需求，可以通过修改项目空间安全配置(`Set ObjectCreatorHasGrantPermission=false`)以达到控制目的。
- 如果是为了控制数据在不同项目空间之间的流动，则可以将项目空间设置为受保护状态(ProjectProtection)。设置之后，只允许用户在项目空间内访问数据，这样可以有效防止数据流出道项目空间之外。

项目空间中的LabelSecurity安全机制默认是关闭的，ProjectOwner可以自行开启。

LabelSecurity安全机制一旦开启，上述的默认安全策略将被强制执行。当用户访问数据表时，除了必须拥有Select权限外，还必须获得读取敏感数据的相应许可等级。或者说，通过LabelSecurity只是通过CheckPermission的必要而非充分条件。

LabelSecurity操作

- LabelSecurity机制的开/关

```
Set LabelSecurity=true|false;
  --开启或关闭LabelSecurity机制，默认为false。
  --此操作必须由ProjectOwner完成，其他操作则可以由Admin角色完成。
```

- 给user设置安全许可标签

```
SET LABEL <number> TO USER <username>; -- number取值范围:[0, 9]，该操作只能由ProjectOwner或Admin角色完成
--举例:
ADD USER aliyun$yunma@aliyun.com; --添加用户，默认的安全许可标签为0级
ADD USER ram$yunma@aliyun.com:Allen;--添加yunma@aliyun.com的RAM子账号用户Allen
SET LABEL 3 TO USER aliyun$yunma@aliyun.com;
  --设置yunma的安全许可标签为3级，他能访问敏感等级不超过3级的数据
SET LABEL 1 TO USER ram$yunma@aliyun.com:Allen;
  --设置yunma的子账号Allen的安全许可标签为1级，他能访问敏感等级不超过1级的数据
```

- 给数据设置敏感等级标签

```
SET LABEL <number> TO TABLE tablename(column_list); -- number取值范围:[0, 9]，该操作只能由ProjectOwner或Admin角色完成
--举例:
SET LABEL 1 TO TABLE t1; --设置表t1的label为1级
SET LABEL 2 TO TABLE t1(mobile, addr); --将t1的mobile,addr两列的label设置为2级
SET LABEL 3 TO TABLE t1; --设置表t1的label为3级. 注意此时mobile,addr两列的label仍为2级
```

说明:

从上面例子可以看出，显式地对列设置的标签会覆盖对表设置的标签，而不会考虑标签设置的顺序以及敏感等级的高低。

当前MaxCompute SET LABEL必须要设置具体字段才生效。

- 显式授权低级别用户访问特定的高敏感级数据表

```
 --授权:
 GRANT LABEL <number> ON TABLE <tablename>[(column_list)] TO USER <
username> [WITH EXP <days>]; --默认过期时间是180天
 --撤销授权:
 REVOKE LABEL ON TABLE <tablename>[(column_list)] FROM USER <
username>;
 --清理过期的授权:
 CLEAR EXPIRED GRANTS;
 --举例:
 GRANT LABEL 2 ON TABLE t1 TO USER ram$yunma@aliyun.com:Allen WITH
EXP 1; --显式授权Allen访问t1表中敏感度不超过2级的数据，授权有效期为1天
 GRANT LABEL 3 ON TABLE t1(col1, col2) TO USER ram$yunma@aliyun.com
:Allen WITH EXP 1; --显式授权alice访问t1(col1, col2)中敏感度不超过3级的数据，授权有效期为1天
 REVOKE LABEL ON TABLE t1 FROM USER ram$yunma@aliyun.com:Allen; --撤销alice对t1表的敏感数据访问
```

 说明:

目前取消用户对table的label权限，会同时取消该用户对table字段的label的权限。

- 查看一个用户能访问哪些敏感数据集

```
 SHOW LABEL [<level>] GRANTS [FOR USER <username>];
   --省略 [FOR USER <username>]时，查看当前用户所能访问的敏感数据集
   --省略<level>时，将显示所有label等级的授权；若指定<level>，则只显示指定等级的授权
```

- 查看一个敏感数据表能被哪些用户访问

```
 SHOW LABEL [<level>] GRANTS ON TABLE <tablename>;
   --显示指定table上的Label授权
```

- 查看一个用户对一个数据表的所有列级别的Label权限

```
 SHOW LABEL [<level>] GRANTS ON TABLE <tablename> FOR USER <username
>;
   --显示指定用户对指定table上列级别的Label授权
```

- 查看一个表中所有列的敏感等级

```
DESCRIBE <tablename>;
```

- 控制package安装者对package中敏感资源的许可访问级别

```
 ALLOW PROJECT <prjName> TO INSTALL PACKAGE <pkgName> [USING LABEL <
number>];
```

```
  --由package创建者授权，授予package安装者对package中敏感资源的许可访问级别
```

说明:

- 省略[USING LABEL <number>]时，默认为0级，即只可以访问非敏感数据。
- 跨项目空间访问敏感数据时，package安装者的项目空间中的所有用户都将使用此命令中许可的访问级别

LabelSecurity应用场景示例

- 限制项目空间中所有非Admin用户对一张表的某些敏感的列的读访

  场景说明：user_profile是某项目空间中的一张含有敏感数据的表，它包含有100列，其中有5列包含敏感数据：id_card, credit_card, mobile, user_addr, birthday. 当前的DAC机制中已经授权了所有用户对该表的Select操作。ProjectOwner希望除了Admin之外，所有用户都不允许访问那5列敏感数据。

  ProjectOwner操作步骤如下：

```
set LabelSecurity=true;
  --开启LabelSecurity机制
set label 2 to table user_profile(mobile, user_addr, birthday);
  --将指定列的敏感等级设置为2
set label 3 to table user_profile(id_card, credit_card);
  --将指定列的敏感等级设置为3
```

说明:

以上操作执行之后，所有非Admin用户都将无法访问那5列数据。如果有人因业务需要确实要访问这些敏感数据，那么需要ProjectOwner或Admin的授权。

Alice是项目空间中的一员，由于业务需要，她要申请访问user_profile的mobile列的数据，需要访问1周时间。项目空间管理员操作步骤如下：

```
GRANT LABEL 2 ON TABLE user_profile TO USER ALIYUN$alice@aliyun.com
WITH EXP 7;
```

说明:

敏感等级为2的数据一共有三列：mobile, user_addr, birthday。那么，当上述授权成功后，Alice将能访问这三列数据。这里存在轻微的过渡授权。如果管理员合理设置数据列的敏感度，那么这种过渡授权是可以避免的。

- 限制项目空间中已获得敏感数据访问许可的用户在项目空间内对敏感数据的复制与肆意传播

  场景说明：在上一个场景中，Alice由于业务需要而获得了等级为2的敏感数据的访问权限。但管理员仍然担心Alice可能会将user_profile表中的敏感等级为2的那些数据复制到她自己新建的

另一张表user_profile_copy中，从而进一步将user_profile_copy表自主授权给Bob访问。如何限制Alice的这种行为?

考虑到安全易用性和管理成本，LabelSecurity的默认安全策略允许WriteDown，即允许用户向敏感等级不高于用户等级的数据列写入数据，因此MaxCompute还无法从根本上解决此问题。但这里可以限制用户的自主授权行为：允许对象创建者只能访问自己创建的数据，而不允许自主授权该对象给其他用户。操作如下：

```
SET ObjectCreatorHasAccessPermission=true;
  --允许对象创建者操作对象
SET ObjectCreatorHasGrantPermission=false;
  --不允许对象创建者授权对象给其他用户
```

1.4 跨项目空间的资源分享

1.4.1 基于Package的跨项目空间的资源分享

假设您是项目空间的Owner或管理员（admin角色），某个主账号下多个项目空间，其中项目空间prj1里有一批资源（包括tables、Resources、自定义functions）可以分享给其他项目空间使用，但是若把其他项目空间的user都add到这个prj1项目空间并逐个进行授权操作，需要操作非常多的步骤，对于prj1项目空间来说引入很多跟本项目业务无关（假设存在）的user非常不方便管理。那么您可以使用本节介绍的跨项目空间的资源分享功能。

如果资源需要精细控制单人使用，且申请人是本业务项目团队成员，那么建议您使用项目空间的用户与授权管理功能：项目空间的用户与授权管理

Package是一种跨项目空间共享数据及资源的机制，主要用于解决跨项目空间的用户授权问题。

如果不使用Package，对于下面的场景我们无法有效的解决：

Alifinance项目空间的成员若要访问Alipay项目空间的数据，则需要Alipay项目空间管理员执行繁琐的授权操作：首先需要将Alifiance项目空间中的用户添加到Alipay项目空间中，再分别对这些新加入的用户进行普通授权。

实际上，Alipay项目空间管理员并不期望对Alifiance项目空间中的每个用户都进行授权管理，而更期望有一种机制能使得Alifiance项目空间管理员能对许可的对象进行自主授权控制。

使用Package之后，Alipay项目空间管理员可以对Alifinance需要使用的对象进行打包授权(也就是创建一个Package)，然后许可Alifinance项目空间可以安装这个Package。在Alifinance项目空间管理员安装Package之后，就可以自行管理Package是否需要进一步授权给自己Project下的用户。

1.4.2 Package的使用方法

本文为您介绍项目空间Package创建者和Package使用者所涉及的操作。

Package的使用方法

Package的使用涉及到两个主体：Package创建者和Package使用者。

- Package创建者项目空间是资源提供方，它将需要分享的资源及其访问权限进行打包，然后许可Package使用方来安装使用。
- Package使用者项目空间是资源使用方，它在安装资源提供方发布的Package之后，便可以直接跨项目空间访问资源。

下面分别介绍Package创建者和Package使用者所涉及的操作。

Package创建者

- 创建Package

```
create package <pkgname>;
```

说明：

- 仅project的owner有权限进行该操作。
- 目前创建的package名称不能超过128个字符。

- 将分享的资源添加到Package

```
    add project_object to package package_name [with privileges
privileges];--将对象添加到Package
    remove project_object from package package_name;--将对象从Package
移除
    project_object ::= table table_name |
                       instance inst_name |
                       function func_name |
                       resource res_name
    privileges ::= action_item1, action_item2, ...
```

说明：

- 目前支持的对象类型不包括Project类型，也就是不允许通过Package在其他Project中创对象。
- 添加到Package中的不仅仅是对象本身，还包括相应的操作权限。当没有通过[with privileges privileges]来指定操作权限时，默认为只读权限，即Read/Describe/Select。对象及其权限被看作一个整体，添加后不可被更新。若有需要，只能删除和重新添加。

- 对象添加到package时，并非快照打包，因此后续对象数据有变更，通过package授权访问对象也是访问该对象的当前数据。

- 许可其他项目空间使用Package

```
allow project <prjname> to install package <pkgname> [using label <
number>];
```

- 撤销其他项目空间使用Package的许可

```
disallow project <prjname> to install package <pkgname>;
```

- 删除Package

```
delete package <pkgname>;
```

- 查看已创建和已安装的Package列表

```
show packages;
```

- 查看Package详细信息

```
describe package <pkgname>;
```

Package使用者

- 安装Package

```
install package <pkgname>;
```

对于安装Package来说，要求pkgName的格式为：`<projectName>.<packageName>`。

说明:

仅project的Owner有权限进行该操作。

- 卸载Package

```
uninstall package <pkgname>;
```

对于卸载Package来说，要求pkgName的格式为：`<projectName>.<packageName>`

- 查看Package

```
show packages;
    --查看已创建和已安装的package列表
    describe package <pkgname>;
    --查看package详细信息
```

- 使用方项目给本项目其他成员或role授权访问Package

被安装的Package是一种独立的MaxCompute对象类型。若要访问Package里的资源(即其他项目空间分享给您的资源)，您必须拥有对该Package的Read权限。

如果请求者没有Read权限，则需要向ProjectOwner或Admin申请。ProjectOwner或Admin可以通过ACL授权机制来完成。

将package授权给user或role：

```
grant actions on package <pkgName> to user <username>;
grant actions on package <pkgName> to role <role_name>;
```

说明：

授权后，user仅在此project中有权限访问该package中的对象。

比如，如下的ACL授权允许云账号用户LIYUN$odps_test@aliyun.com访问Package里的资源：

```
use prj2;
    install package prj1.testpkg;
    grant read on package prj1.testpackage to user aliyun$odps_test@
aliyun.com;
```

或者允许角色 role_dev的所有成员访问package里的资源：

```
use prj2;
    install package prj1.testpkg;
    grant read on package prj1.testpackage to role role_dev;
```

场景示例

场景描述：Jack是项目空间prj1的管理员。John是项目空间prj2的管理员。由于业务需要， Jack希望将其项目空间prj1中的某些资源(比如datamining.jar及sampletable表)分享给John的项目空间prj2。 如果项目空间prj2的用户Bob需要访问这些资源，那么管理员John可以通过ACL给Bob自主授权，无需Jack参与。

操作步骤：

1. 项目空间prj1的管理员Jack在项目空间prj1中创建资源包(package)。

```
use prj1;
    create package datamining; --创建一个package
    add resource datamining.jar to package datamining; --添加资源到
package
    add table sampletable to package datamining; --添加table到package
    allow project prj2 to install package datamining; --将package分享
给项目空间prj2
```

2. 项目空间prj2管理员Jhon在项目空间prj2中安装package。

```
use prj2;
    install package prj1.datamining; --安装一个package
```

```
    describe package prj1.datamining; --查看package中的资源列表
```

3. Jhon对package给Bob进行自主授权。

```
use prj2;
    grant Read on package prj1.datamining to user aliyun$bob@aliyun.
com; --通过ACL授权Bob使用package
```

1.5 项目空间的安全配置

MaxCompute是一个支持多租户的数据处理平台，不同的租户对数据安全需求不尽相同。为了满足不同租户对数据安全的灵活需求，MaxCompute支持项目空间级别的安全配置，ProjectOwner可以定制适合自己的外部账号支持和鉴权模型。

MaxCompute支持多种正交的授权机制，如ACL授权、隐式授权（如对象创建者自动被赋予访问对象的权限）。但是并非所有用户都需要使用这些安全机制，您可以根据自己的业务安全需求或使用习惯，合理设置本项目空间的鉴权模型。

```
show SecurityConfiguration
--查看项目空间的安全配置
set CheckPermissionUsingACL=true/false
--激活/冻结ACL授权机制，默认为true
set ObjectCreatorHasAccessPermission=true/false
--允许/禁止对象创建者默认拥有访问权限，默认为true
set ObjectCreatorHasGrantPermission=true/false
--允许/禁止对象创建者默认拥有授权权限，默认为true
set ProjectProtection=true/false
--开启/关闭项目空间的数据保护机制，禁止/允许数据流出项目空间
```

说明:

您也可以通过DataWorks进行可视化操作，完成项目空间的相关安全配置，详情请参见*MaxCompute*高级配置

1.6 项目空间的数据保护

本文为您介绍项目空间的数据保护机制和开启数据保护机制后的数据流出方法。

背景和动机

在现实中，有一些公司(比如金融机构、军工企业)对数据安全非常敏感，比如不允许员工将工作带回家，而只允许在公司内部进行操作。而且公司的所有电脑上的USB存储接口也都是禁用的。这样做的目的是禁止员工将敏感数据泄漏出去。

作为MaxCompute项目空间管理员，您是否也有类似的安全需求呢？ —— “不允许用户将数据转移到项目空间之外”。

比如，当项目空间prj1的Owner遇到下图所示的这种情形时，是否会担心用户Alice将她能访问的数据转移到prj2中去呢？因为她可以访问prj2，而prj2又不受控制。

更具体地说，假设Alice已经被您授予了访问myprj.table1的Select权限，同时Alice也被prj2的管理员授予了CreateTable的权限。

那么Alice将可以通过如下的任何一种方法将数据转移到prj2中去：

- 提交SQL：

```
create table prj2.table2 as select * from myprj.table1;
```

- 编写MapReduce将myprj.table1读出，并写入prj2.table2。

如果您项目空间中的数据非常敏感，绝对不允许流出到其他项目空间中去，您希望MaxCompute能将上述导致数据流出的操作统统禁止，可以吗？没问题。

数据保护机制

MaxCompute提供的项目空间保护机制正好可以满足上述需要。您只需要在您的项目空间中做如下设置：

```
set projectProtection=true
  --设置ProjectProtection规则：数据只能流入，不能流出
```

设置ProjectProtection后，您的项目空间中的数据流向就会得到控制 —— “数据只能流入，不能流出”。即上述的两种操作将失效，因为它们都违背了ProjectProtection规则。

默认时，ProjectProtection不会被设置，值为false。

同时在多个项目空间中拥有访问权限的用户就可以自由地使用任意支持跨Project的数据访问操作来转移项目空间的数据。如果对项目空间中的数据高度敏感，则需要管理员自行设置ProjectProtection保护机制。

开启数据保护机制后的数据流出方法

在您的项目空间被设置了ProjectProtection之后，您可能很快就会遇到这样的需求：Alice向您提出申请，她的确需要将某张表的数据导出您的项目空间。

而且经过您的审查之后，那张表也的确没有泄漏您关心的敏感数据。为了不影响Alice的正常业务需要，MaxCompute为您提供了在ProjectProtection被设置之后的两种数据导出途径。

- 设置TrustedProject

  若当前项目空间处于受保护状态，如果将数据流出的目标空间设置为当前空间的TrustedProject,

那么向目标项目空间的数据流向将不会被视为触犯ProjectProtection规则。如果多个项目空间之间两两互相设置为TrustedProject,

那么这些项目空间就形成了一个TrustedProject Group，数据可以在这个Project Group内流动，但禁止流出到Project Group之外。

管理TrustedProject的命令如下：

```
list trustedprojects;
  --查看当前project中的所有TrustedProjects
add trustedproject <projectname>;
  --在当前project中添加一个TrustedProject
remove trustedproject <projectname>;
  --在当前project中移除一个TrustedProject
```

- 资源分享与数据保护

  在MaxCompute中，基于*Package*的跨项目空间的资源分享机制与ProjectProtection数据保护机制是正交的，但在功能上却是相互制约的。

  MaxCompute规定：资源分享优先于数据保护。换句话说，如果一个数据对象是通过资源分享方式授予其他项目空间用户访问，那么该数据对象将不受ProjectProtection规则的限制。

关于最佳实践

如果要防止数据从项目空间的流出，在设置`ProjectProtection=true`之后，还需检查如下配置：

- 确保没有添加trustedproject。如果有设置，则需要评估可能的风险；
- 确保没有使用package数据分享。如果有设置，则需要确保package中没有敏感数据。

1.7 安全相关语句汇总

1.7.1 项目空间的安全配置

本文为您介绍一些项目空间安全配置中的鉴权配置概念和数据保护概念。

鉴权配置

语句	说明
show SecurityConfiguration	查看项目空间的安全配置
set CheckPermissionUsingACL=true/false	激活/冻结ACL授权机制
set CheckPermissionUsingPolicy=true/false	激活/冻结Policy授权机制

语句	说明
set ObjectCreatorHasAccessPermission=true/false	允许/禁止对象创建者默认拥有访问权限
set ObjectCreatorHasGrantPermission=true/false	允许/禁止对象创建者默认拥有授权权限

数据保护

语句	说明
set ProjectProtection=false	关闭数据保护机制
list TrustedProjects	查看可信项目空间列表
add TrustedProject `<projectName>`	添加可信项目空间
remove TrustedProject `<projectName>`	移除可信项目空间

1.7.2 项目空间的权限管理

本文为您介绍项目空间权限管理中的用户管理、角色管理、ACL授权、权限审查等相关概念。

用户管理

语句	说明
list users	查看所有已添加进来的用户
add user `<username>`	添加一个用户
remove user `<username>`	移除一个用户

角色管理

语句	说明
list roles	查看所有已创建的角色
create role `<rolename>`	创建一个角色
drop role `<rolename>`	删除一个角色
grant `<rolelist>` to `<username>`	对用户指派一个或多个角色
revoke `<rolelist>` from `<username>`	撤销对用户的角色指派

ACL授权

语句	说明
grant <privList> on <objType> <objName> to user <username>	对用户授权
grant <privList> on <objType> <objName> to role <rolename>	对角色授权
revoke <privList> on <objType> <objName> from user <username>	撤销对用户的授权
revoke <privList> on <objType> <objName> from role <rolename>	撤销对角色的授权

权限审查

语句	说明
whoami	查看当前用户信息
show grants [for <username>] [on type <objectType>]	查看用户权限和角色
show acl for <objectName> [on type <objectType>]	查看具体对象的授权信息
describe role <roleName>	查看角色的授权信息和角色指派

1.7.3 基于Package的资源分享

本文为您介绍基于Package的资源分享语句说明。

分享资源

语句	说明
create package <pkgName>	创建一个Package
delete package <pkgName>	删除一个Package
add <objType> <objName> to package <pkgName> [with privileges privs]	向Package中添加需要分享的资源
remove <objType> <objName> from package <pkgName>	从Package中删除已分享的资源
allow prOject <prjName> to install package <pkgName> [using label <num>]	许可某个项目空间使用您的Package

语句	说明
disallow project <prjName> to install package <pkgName>	禁止某个项目空间使用您的Package

使用资源

语句	说明
install package <pkgName>	安装Package
uninstall package <pkgName>	卸载Package

查看Package

语句	说明
show packages	列出所有创建和安装的Packages
describe package <pkgName>	查看package的详细信息

说明:

如果在DataWorks执行生产project授权相关的请求时:

1. project owner通过临时查询方式执行，不能提交到生产环境执行。因为生产环境是由生产账号执行，而这个账号没有授权的权限。
2. 查询语句前加use <生产project>;语句，并与命令一起提交。因为DataWorks数据开发默认当前的project是_dev结尾的开发project。通过命令行执行授权命令时，请project owner先执行:

```
use project_name;--进入对应的project
```