# 阿里云
# MaxCompute

## 管理

文档版本：20190613

# 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:**<br>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:**<br>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:**<br>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 安全管理基础

## 1.1 安全模型

为了方便MaxCompute的Project Owner或安全管理员进行Project的日常安全运维和保障数据安全，在您开始配置安全功能之前，建议您先进行安全模型的学习。

MaxCompute有安全模型，*DataWorks*也有安全模型。当通过DataWorks使用MaxCompute，而DataWorks的安全模型不满足业务安全需求时，需要合理地将两个安全模型结合使用。

您可以通过观看*MaxCompute*安全管理解析视频进行安全管理基础的学习。

### MaxCompute安全模型

安全体系

MaxCompute多租户的数据安全体系，主要包括如下内容：

- 用户认证

  支持云账号和RAM账号两种账号体系。对于RAM账号，仅识别账号体系不识别RAM权限体系，即可将主账号自身的任意RAM子账号加入MaxCompute的某一个项目中，但MaxCompute在对该RAM子账号做权限验证时，并不会考虑RAM中的权限定义。

- 用户与授权管理

  您可以在MaxCompute Project中对用户进行添加（Add）、移除（Remove）、授权（Grant）管理，还可以通过角色（Role）管理授权。MaxCompute Project默认有Admin Role。授权方式包含ACL和Policy方式，此处只讲解ACL方式。

  ACL使用的语法类似于SQL92定义的GRANT/REVOKE语法，通过简单的授权语句来完成对已存在的项目空间对象的授权或撤销授权。授权语法举例如下：

  ```
  grant actions on object to subject;
  revoke actions on object from subjec;
  ```

- 标签安全策略

  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），它可以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。

- 跨项目空间的资源分享

  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。即可以分享Table、Resource、Function等资源给其他项目，但无需对其他项目的用户进行管理。

- 项目空间的数据保护

  主要解决类似于不允许用户将数据转移到项目空间之外的需求。

对象赋权、Role和Label关系

MaxCompute安全体系包含多种策略，而各种策略赋权是权限递增关系。我们以获取一个L4等级表权限的具体步骤来举例说明权限递增关系，具体步骤如下：

1. 如果用户未有过授权记录，且非本项目用户，首先需要给项目添加该User(用户)。这个过程中，用户还没有任何实际权限。
2. 赋权User（用户）对象的操作权限，有以下方式赋权：
   a. 可以是单独的操作权限。
   b. 将ACL赋权给Role再赋权给用户。 如果资源是没有设置Label的，则此时用户已经拥有的该资源的权限。
3. 对于拥有*Label*的资源，例如：数据表、打包了数据表的Package，则还需要赋予Label权限。有以下四种Label赋权：
   a. 针对某个数据表的字段。
   b. 针对某个数据表（暂不支持）。
   c. 针对某个Package。
   d. 给某个User（用户）一个批量的Label权限，不支持Role。

各权限赋权过程及关系图如下：

数据流出保护机制和Package关系

*Project Protection*(数据流出保护机制)是MaxCompute防止项目内的数据批量流出的安全功能。开启数据流出保护后，如果相互之间没有建立Trusted Project Group，与其它Project的数据赋权就必须通过Package方式进行，Package赋权后，对方Package可以自主赋权Package内的资源给组内用户。

部分资源(例如某些常用表、UDF等)，如想通过Package管理赋权，也可以将资源打包后赋权给其他Project。

ProjectProtection(数据流出保护机制)支持做例外处理：部分特殊的业务场景，可以针对应用的IP地址、产品云账号做Exception策略，以满足特殊的数据流出需求。

### DataWorks安全模型

DataWorks提供多人协同数据开发工作的平台，其安全模型需要考虑几方面：

1. 企业之间数据的安全隔离。
2. 数据开发，即ETL过程中的安全问题。例如生产任务如何保障不可随意变更、哪些成员可以进行代码编辑调试、哪些成员可以进行发布生产任务等。
3. 由于底层的MaxCompute有自己的安全模型，项目成员做ETL过程肯定会需要MaxCompute的各种资源（Table、Resource、Function、Instance）的相关权限。

> 说明:
>
> 针对第一点：DataWorks支持用户认证以及对接RAM。云账号可作为主账号开通并创建DataWorks项目，而项目成员必须为该主账号的RAM子账号，不能是其他云账号。
>
> 另外，同个主账号创建的项目作为一个组织，项目与项目之间的任务可以进行依赖配置；不同主账号创建的项目之间数据（各种任务）隔离。
>
> 针对第二点：DataWorks通过业务划分开发项目与生产项目进行任务开发调试和稳定生产的隔离。通过成员角色控制哪些成员可以进行任务开发调试，哪些成员可以运维生产任务等。
>
> 针对第三点：DataWorks在MaxCompute Project创建成功的同时，在MaxCompute Project里会对应DataWorks的角色创建Role，并给不同Role进行赋权。

## 1.2 MaxCompute和DataWorks权限关系

通过MaxCompute的安全模型进行权限控制，并不会影响成员在DataWorks任何界面操作。而通过DataWorks的用户角色分配，则有可能影响成员的MaxCompute资源权限。下面详细介绍这两个产品之间的权限如何交叉关联。

### 项目关系

通过MaxCompute或DataWorks官网产品页进入管理控制台创建的项目时，有两种选择：

- 简单模式的项目实际上是创建了关联绑定好的一个MaxCompute Project和一个DataWorks项目空间（工作空间）。同时在MaxCompute的Project里创建对应的几个Role，具体Role权限请参见成员角色权限关系。

- 标准模式的项目实际上是创建了关联绑定好的一个开发（dev）MaxCompute Project、一个生产（prod）MaxCompute Project来同时对应一个DataWorks项目空间（工作空间）。同时在MaxCompute的Project里创建对应的几个Role，具体Role权限请参见成员角色权限关系。

### 账号认证

云账号在DataWorks项目中只能是主账号，即项目Owner。在MaxCompute既可以为Owner也可以为普通User。当通过DataWorks项目成员管理添加成员时只能添加当前项目主账号对应的RAM子账号。而MaxCompute可以通过命令行`add user xxx;`命令添加其他云账号。

### 成员角色与权限关系

如前文所述，DataWorks为了解决项目成员在ETL过程中需要的MaxCompute相关资源权限，绑定了一些MaxCompute Role。具体是指DataWorks项目固定有几个成员角色，同时在对应MaxCompute Project上创建了对应的几个Role。此外，MaxCompute Project本身除Project Owner之外，还包含有一个admin Role。具体权限对应如下表所示。

| MaxCompute角色 | MaxCompute数据权限 | DataWorks成员角色 | DataWorks平台权限特征 |
| --- | --- | --- | --- |
| project owner | MaxCompute Project的Owner，拥有所有Project的权限。 | 无 | 无 |

| MaxCompute角色 | MaxCompute数据权限 | DataWorks成员角色 | DataWorks平台权限特征 |
| --- | --- | --- | --- |
| admin | 每一个Project在创建时，会自动创建一个admin的role，并且为该role授予了确定的权限：可以访问Project的所有对象、对user或role进行管理、对user或role进行授权。<br>与Project Owner相比：admin角色不能将 admin 权限指派给用户，不能设定项目空间的安全配置，不能修改项目空间的鉴权模型，admin角色所对应的权限不能被修改。<br>Project Owner可以将这admin role赋权给一个user，让该user代理安全管理。 | 无 | 无 |
| role_project_admin | project/table/fuction/resource/instance/job/package的所有权限 | 项目管理员 | 指项目空间的管理者。可对该项目空间的基本属性、数据源、当前项目空间计算引擎配置和项目成员等进行管理，并为项目成员赋予项目管理员、开发、运维、部署、访客角色。 |
| role_project_dev | project/fuction/resource/instance/job/package/table 的所有权限 | 开发 | 开发角色的用户能够创建工作流、脚本文件、资源和UDF，新建/删除表，同时可以创建发布包，但不能执行发布操作。 |
| role_project_pe | project/fuction/resource/instance/job的所有权限，拥有package的read权限和table 的read/describe权限。 | 运维 | 运维角色的用户由项目管理员分配运维权限，拥有发布及线上运维的操作权限，没有数据开发的操作权限。 |

| MaxCompute角色 | MaxCompute数据权限 | DataWorks成员角色 | DataWorks平台权限特征 |
| --- | --- | --- | --- |
| role_project_deploy | 默认无权限 | 部署 | 部署角色与运维角色相似，但是它没有线上运维的操作权限。 |
| role_project_guest | 默认无权限 | 访客 | 访客角色的用户只具备查看权限，没有权限进行编辑工作流和代码等操作。 |
| role_Project_security | 默认无权限 | 安全管理员 | 安全管理员仅在数据保护伞中用到，用于敏感规则配置，数据风险审计等 |

**说明:**

由上表可知，DataWorks角色对应的MaxCompute权限是固定的：一旦某个user通过DataWorks角色获取MaxCompute相关role权限后，如果此后又通过命令行方式获得了其他的MaxCompute权限，会使该user在MaxCompute的权限与在DataWorks上看到的不一致。

用户和权限关系图

一个DataWorks项目空间绑定一个MaxCompute project，此时您可以根据DataWorks项目管理 > 工作空间配置 > 访问身份这个属性设置决定DataWorks其他项目成员是否拥有MaxCompute project的权限。

访问身份分为个人账号和计算引擎指定账号（系统账号），具体用户和权限对应关系如下图所示。

对于标准模式，一个DataWorks项目空间绑定两个MaxCompute project：一个开发项目和一个生产项目。DataWorks其他项目成员根据成员角色拥有MaxCompute开发project对应的role权限，但没有MaxCompute生产project的权限。MaxCompute任务需要通过发布流程发布到生产project后，以owner账号提交到MaxCompute执行。

## 1.3 用户与权限管理

在您进行用户与权限管理配置前，您可以首先通过本文了解MaxCompute与DataWorks的用户权限管理区别。

### 用户管理

| 操作类型 | MaxCompute用户管理 | DataWorks用户管理 |
| --- | --- | --- |
| 操作描述 | 应删除或锁定无属主、闲置以及离职人员的账号权限。通过DataWorks新增的用户，注意会可能授权于默认的role。 | 准确添加和管理用户，应删除或锁定无属主、闲置以及离职人员的账号权限，严控管理员、运维权限。 |
| 操作角色 | owner或admin role | 项目管理员 |
| 现状查看 | 查看项目下的用户：`list users;`<br>查看指定用户拥有的权限：`show grants for <username>;` | DataWorks项目管理 > 成员管理查看现有成员及角色，并确认各个成员权限的合理性。 |
| 赋权操作 | 成员仅加入 MaxCompute project，不在DataWorks项目成员中体现，且无任何权限。 需要结合对象操作、role权限、label权限才能使用，所以应关注成员是否拥有对象操作、role权限、label权限，并对相应授权进行清理。此外，可以添加云账号和RAM子账号。<br>在项目空间中添加用户：`add user <username>;` | DataWorks项目管理 > 成员管理添加成员和分配角色。<br>说明：<br>1. 只能添加该项目负责人账号下的RAM子账号为项目成员。<br>2. 添加一个成员，并分配角色，可能会在MaxCompute赋予默认的role权限。 |
| 回退操作 | 在项目空间中移除用户：`remove user <username>;` | 清理成员或对应角色权限。删除后，会自动清除对应 MaxCompute 内的user和默认role。 |

### 角色管理

| 操作类型 | MaxCompute role管理 | DataWorks角色管理 |
| --- | --- | --- |
| 操作描述 | 准确的创建role并配置role权限，及时清理离职或转岗人员的账号，清理role中不必要开放的资源和权限。<br>MaxCompute project创建成功后除了默认有admin role外，DataWorks创建了其他role。 | 准确的分配角色。成员工作性质发生改变需及时改变角色，严格控制项目管理员和运维角色的分配。 |
| 操作角色 | project owner或admin | 项目管理员 |
| 现状查看 | 查看当前project所有role：`list roles;`<br>查看role中的权限：`describe role <role_name>;`<br>查看某用户在什么role中：`show grants for <username>;`<br>目前暂时不支持查看某个role被指派给哪些user。 | DataWorks项目管理 > 成员管理可以点击每个角色查看该角色下的成员。 |

| 操作类型 | MaxCompute role管理 | DataWorks角色管理 |
| --- | --- | --- |
| 赋权操作 | MaxCompute除了默认的role，还可以自定义role，通过命令自定义role权限并将role授权给user。<br>创建role：`create role <role_name>;`<br>给角色授权：`grant actions on object to <role_name>;`<br>给用户授予角色：`GRANT <roleName> TO <full_username> ;`<br>此外，在DataWorks的项目管理 > MaxCompute高级配置- > 自定义用户角色页面，可以通过图形页面方式创建MaxCompute自定义role、对role进行授权、将role授权给成员。<br>说明:<br>通过命令行创建的role不会在这个界面显示。 | DataWorks角色是固定的,不能自定义。成员添加到DataWorks项目时勾选角色分配给成员，该成员即可拥有对应角色的权限。 |

| 操作类型 | MaxCompute role管理 | DataWorks角色管理 |
| --- | --- | --- |
| 回退操作 | 删除角色中的用户：`REVOKE <roleName> FROM <full_username>;`<br>撤销对角色的授权：`revoke <privList> on <objType> <objName> from role <rolename>;`<br>删除角色：`DROP ROLE <roleName>;`<br>如果是通过DataWorks项目管理 > MaxCompute高级配置- > 自定义用户角色页面创建的role，也通过这个页面进行回退操作。 | DataWorks的角色不能删除，只能将某个成员的角色去掉。 |

ACL（对象操作）的授权管理

| 操作类型 | 说明 |
| --- | --- |
| 操作描述 | 回收非必须必要的对象操作授权，操作权限涉及多种操作对象和类型，应逐一确认。 |
| 操作角色 | project owner或admin role |
| 现状查看 | 查看指定用户的权限： `show grants for <username>;`<br>查看当前用户的权限：`show grants;`<br>查看指定对象的授权列表：`show acl for <objectName> [on type <objectType>];`<br>查看某package赋权情况案例：`show acl for alipaydw.alipaydw_for_alisec_app on type package;` |

| 操作类型 | 说明 |
| --- | --- |
| 赋权操作 | 进行某对象的操作赋权：`grant actions on object to subject;`<br>操作、主体、客体类型的表达式如下。<br>actions类型: `action_item1, action_item2, ...`<br>object类型：`project project_name,table schema_name ,instance inst_name ,function func_name ,resource res_name`<br>subject类型：`user full_username ,role role_name`<br>有关操作、主体、客体类型的更多信息，请参见授权。 |
| 回退操作 | 回收某对象的操作权限： `revoke actions on object from subject;` |

Package授权管理

| 操作类型 | 说明 |
| --- | --- |
| 操作描述 | 开启 *ProjectProtection*的项目如果没有在同一个互信项目组（TrustedProject Group），则必须使用package方式赋权，package赋权有两种方式。请确保package合理打包和赋权，无闲置package赋权。 |
| 操作角色 | project owner |
| 现状查看 | 1. 了解本项目Package创建及赋权情况：<br>查看已创建和已安装的Package列表：`show packages;`<br>查看Package详细信息： `describe package <pkgname>;`<br>2. 本项目安装的Package，对用户的授权情况：`show acl for <project_name.package_name> on type package;` |

| 操作类型 | 说明 |
| --- | --- |
| 赋权操作 | **Package创建者：**<br>**1. 创建Package：** `create package <pkgname>;`<br>**2. 将分享的资源添加到Package：** `add project_object to package package_name [with privileges privileges ];`<br>**project_object表达式：** `table table_name ,instance inst_name ,function func_name ,resource res_name`<br>**3. 许可其他项目空间使用Package：** `allow project <prjname> to install package <pkgname> [using label<number>];`<br>**Package使用者：**<br>**1. 安装Package：** `install package <pkgname>;`<br>**2. 将package赋权给用户、角色，package赋权给具体用户时，不能指定label(project owner或admin都可操作)。**<br>`grant actions on package <pkgName> to user <username>;`<br>`grant actions on package <pkgName> to role <role_name>;`<br>**actions 类型：参见授权，通常将Package的read权限赋给对象，即可满足对象访问package中资源的需求。** |

| 操作类型 | 说明 |
| --- | --- |
| 回退操作 | 1. 撤销其他项目空间使用Package的许可：`disallow project <prjname> to install package <pkgname>;`<br>2. 删除Package：`delete package <pkgname>;`<br>3. 将分享的资源移出Package：<br>`remove project_object from package package_name;`<br>project_object表达式：<br>`table table_name ,instance inst_name ,function func_name ,resource res_name`<br>4. 撤销package的用户、角色的权限。<br>`revoke actions on package <pkgName> from user <username>;`<br>`revoke actions on package <pkgName> from role <role_name>;` |

Label授权管理

| 操作类型 | 说明 |
| --- | --- |
| 操作描述 | Maxcompute的字段、表、package分为1~4个等级，应根据用户实际需要，赋予对应的label权限。 |
| 操作角色 | project owner |

| 操作类型 | 说明 |
| --- | --- |
| 现状查看 | 1. 查看一个用户能访问哪些敏感数据集：<br>`SHOW LABEL [<level>] GRANTS [FOR USER <username>];`<br>--省略[FOR USER <username>]时，查看当前用户所能访问的敏感数据集.<br>--省略<level>时，将显示所有label等级的授权；<br>--若指定<level>，则只显示指定等级的授权.<br>2. 查看一个敏感数据表能被哪些用户访问：<br>`SHOW LABEL [<level>] GRANTS ON TABLE <tablename>;`<br>--显示指定table上的Label授权<br>3. 查看一个用户对一个数据表的所有列级别的Label 权限：<br>`SHOW LABEL [<level>] GRANTS ON TABLE <tablename> FOR USER <username>;`<br>--显示指定用户对指定table上列级别的 *Label* 授权 |

| 操作类型 | 说明 |
| --- | --- |
| 赋权操作 | 1. 给用户单个表或字段的安全许可标签。<br>GRANT LABEL <number> ON TABLE <tablename>[(column_list)] TO USER <username> [WITH EXP <days>]; --默认过期时间是 180 天<br>举例：GRANT LABEL 2 ON TABLE t1 TO USER alice WITH EXP 1; --显式授权alice访问t1表中敏感度不超过2级的数据，授权有效期为1天<br>GRANT LABEL 3 ON TABLE t1(col1, col2) TO USER alice 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<br>2. 给用户授权整个项目的安全许可标签。<br>SET LABEL <number> TO USER <username>;<br>3. 控制package安装者对package中敏感资源的许可访问级别<br>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<br>4. 将package赋权给用户、角色，package赋权给具体用户时，不能指定label。<br>grant actions on package <pkgName> to user <username>;<br>grant actions on package <pkgName> to role <role_name>; |

| 操作类型 | 说明 |
| --- | --- |
| 回退操作 | 1. 撤销用户单个表或字段的安全许可标签。<br>撤销授权：`REVOKE LABEL ON TABLE <tablename>[(column_list)] FROM USER <username>;`<br>清理过期的授权：`CLEAR EXPIRED GRANTS;`<br>举例：`REVOKE LABEL ON TABLE t1 FROM USER alice;`<br>--撤销alice对t1表的敏感数据访问<br>2. 更改用户授权整个项目的安全许可标签，默认等级为 0。<br>`SET LABEL <number> TO USER <username>;`<br>3. 更改package安装者对package中敏感资源的许可访问级别，调整为其他级别，默认为0。<br>`ALLOW PROJECT <prjName> TO INSTALL PACKAGE <pkgName> [USING LABEL <number>];`<br>4. 撤销package的用户、角色的权限。<br>`revoke actions on package <pkgName> from user <username>;`<br>`revoke actions on package <pkgName> from role <role_name>;` |

## 1.4 启用安全功能

您可以通过项目空间的安全配置、*#unique_10*、列级别访问控制启用MaxCompute的安全功能。

### 设置ProjectProtection（数据流出保护机制）

*#unique_10*主要用于满足不允许用户将数据转移到项目空间之外的需求。

| 操作类型 | 说明 |
| --- | --- |
| 操作描述 | 设置 *ProjectProtection*避免项目批量数据下载到本地电脑，出现批量数据泄露风险。 |
| 操作角色 | Project Owner |

| 操作类型 | 说明 |
| --- | --- |
| 查看现状 | 执行命令： `show SecurityConfiguration;`<br>需要检查是否`ProjectProtection=true` |
| 操作设置 | 设置ProjectProtection机制，默认为false。<br>两种设置方法：<br>1. DataWorks项目管理 > MaxCompute高级配置 > 项目空间数据保护。<br>2. 执行MaxCompute命令： `SET ProjectProtection=true [WITH EXCEPTION <policyFile>];`<br>开启后由于部分公共账号或个人用户因种种原因需要数据流出权限，根据需要可设置exception例外策略（白名单）。<br>以下情形建议配置Exception策略：<br>1. 需要数据流出权限的应用系统云账号或IP地址。<br>2. 个人账号开通白名单，应指定允许下载的表。<br>添加项目互信：<br>对于数据可互通的project可以通过项目互信的方式确保数据顺利流转。<br>查看当前project中的所有TrustedProjects： `list trustedprojects;`<br>在当前project中添加一个TrustedProject`add trustedproject <projectname>;`<br>在当前project中移除一个TrustedProject:`remove trustedproject <projectname>;`<br>未添加TrustedProject的项目，申请本项目数据，需要以*Package*方式授权。 |

| 操作类型 | 说明 |
| --- | --- |
| 回退操作 | 关闭 ProjectProtection机制: `SET ProjectProtection=false;`<br>移除TrustedProject: `remove trustedproject <projectname>;` |

开启Label Security（列及安全控制）

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)，它能让项目空间管理员更灵活地控制用户对列级别敏感数据的访问。

| 操作类型 | 说明 |
| --- | --- |
| 操作描述 | 打开LabelSecurity确保字段级别安全控制生效， 项目空间中的LabelSecurity安全机制默认是关闭的。 |
| 操作角色 | Project Owner |
| 查看现状 | `show SecurityConfiguration;`<br>查看是否`LabelSecurity=true` |
| 操作设置 | 开启LabelSecurity机制，默认为false。<br>`Set LabelSecurity=true;` |
| 回退操作 | 关闭LabelSecurity机制:<br>`Set LabelSecurity=false;`<br>操作前，需要确认本Project是否为其它Project赋于了本Project里表的Label权限。 |

设置字段的Label

| 操作类型 | 说明 |
| --- | --- |
| 操作描述 | MaxCompute数据的敏感性可以分为0~4级。所有数据表均可以设置安全等级，避免数据表出现不合理授权访问情形。 |
| 查看现状 | 查看MaxCompute表字段的等级，两个方式。<br>· 执行命令：`DESCRIBE <tablename>;`<br>· 在DataWorks的数据管理查看表详情中的字段信息。 |

| 操作类型 | 说明 |
| --- | --- |
| 操作设置 | 给表字段设置安全级别可以通过以下两种方式。<br>· 方式一（推荐）<br>DataWorks的数据管理里，新建表或者编辑已有表的字段信息，均可以设置字段安全级别。<br>说明:<br>只有Project的LabelSecurity=true，数据管理页面才可见字段安全级别属性。<br>· 方式二<br>执行命令：SET LABEL <number> TO TABLE tablename[(column_list)]; --number取值范围：[0, 4]。<br>举例：<br>SET LABEL 1 TO TABLE t1; --设置表 t1的label为1级<br>SET LABEL 2 TO TABLE t1(mobile, addr);<br>--将t1的mobile，addr两列的label设置为2级<br>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。<br>说明:<br>通过命令行设置自动安全级别后，在DataWorks的数据管理界面，对应表字段安全等级不同步。因此，建议使用DataWorks对表的字段进行安全级别设置。 |
| 回退操作 | 将安全等级调整回原来等级。<br>说明:<br>字段安全等级的上调，会导致原有的授权失效（涉及package授权、生产账号和个人账号）。因此，调整前必须通知到受影响用户。 |

设置访问Project的IP白名单

| 操作类型 | 说明 |
| --- | --- |
| 操作描述 | 设置IP白名单，指定白名单列表中的IP（Console或者SDK所在的出口IP）能够访问这个Project。<br><br>说明：<br>· 当前Project的所有User（包括主账号）都会受限。<br>· DataWorks的机器默认在白名单内，因此通过DataWorks提交MaxCompute任务不会受此限制。 |
| 操作角色 | Project Owner |
| 查看现状 | 通过Console执行命令： `setproject;`<br><br>查看对应的命令`odps.security.ip.whitelist=`若等号后面为空，则表示未设置白名单列表。 |
| 操作设置 | 设置前请特别注意：在白名单List加上自己当前机器IP，以免把自己屏蔽。<br><br>通过客户端执行命令：<br><br>`setproject odps.security.ip.whitelist=xxx.xxx.xxx.xxx,xxx.xxx.x.x/xx,xxx.xxx.xxx.xxx-xxx.xxx.xxx.xxx;`<br><br>白名单中IP列表的表示形式有以下三种，目前已支持IPV6格式。<br>· 单纯IP：例如101.132.236.134、FE80:0202:B3FF:FE1E:8329。<br>· 子网掩码：例如100.116.0.0/16、FE80:0101:4567:F456:0202:B3FF:1111:1111/126。<br>· 网段：例如101.132.236.134-101.132.236.144、FE80:0101:4567:F456:0202:B3FF:FE1E:8330-FE80:0101:4567:F456:0202:B3FF:FE1E:8331。<br><br>设置完IP白名单，您需要等待五分钟后才会生效。<br><br>若您需更精细化管理，也可以通过policy授权。 |

| 操作类型 | 说明 |
| --- | --- |
| 回退操作 | IP白名单清空后，MaxCompute会认为Project关闭了白名单功能。<br>`setproject odps.security.ip.whitelist=;` |

禁止DataWorks的select结果下载到本地

| 操作类型 | 说明 |
| --- | --- |
| 描述 | 开发者通过DataWorks进行的数据分析，通常会屏显在IDE上并且可以下载结果。Project设置ProjectProtection为true后，在本Project中只要有表的读取权限，依然可以通过DataWorks的数据开发界面select后进行结果下载。 |
| 操作角色 | DataWorks管理员 |
| 查看现状 | 进入DataWorks的项目管理 > 工作空间配置，查看下载select结果属性是否打开。 |
| 操作设置 | 进入DataWorks的项目管理 > 工作空间配置，关闭下载select结果开关。 |
| 回退操作 | 进入DataWorks的项目管理 > 工作空间配置，打开下载select结果开关。 |

通过其它云服务提高安全管理等级

使用MaxCompute过程中，会关联使用到其他的云服务，因此也需要考虑通过其他云服务提高MaxCompute的安全管理。通过DataWorks使用MaxCompute时，添加项目成员必须会用到RAM子账号，以下为您介绍如何在RAM子账号服务上提高安全管理等级。

MaxCompute的用户认证支持云账号和RAM账两种账号体系。对于RAM账号，仅识别账号体系不识别RAM权限体系，即可将主账号自身的任意RAM子账号加入MaxCompute 的某一个项目中。MaxCompute在对该RAM子账号做权限验证时，并不会考虑RAM中的权限定义。因此，我们只需要从RAM子账号登录验证入手进行安全控制。

子账号密码强度设置

如果您允许子用户更改登录密码，则应该要求他们创建强密码并且定期轮换。

您可以通过RAM控制台设置密码策略，如最短长度、是否需要非字母字符、必须进行轮换的频率等等。

子账号登录掩码设置

**通过设置网络掩码决定哪些IP地址会受到登录控制台的影响，子用户必须只能从指定的IP地址进行登录。**

**及时撤销用户不再需要的权限**

**当一个子账号对应员工由于工作职责变更而不再使用权限时，应该及时将对应子账号的权限撤销。**

# 2 安全功能详解

## 2.1 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

## 2.2 快速开始

### 2.2.1 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。

操作方法

由项目空间管理员输入下列命令行：

```
use prj1; --切换项目空间。
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。
```

## 2.2.2 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。他们想申请如下权限：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

由项目管理员输入下列命令。

```
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tableviewe
r。
grant tableviewer to aliyun$bob@aliyun.com;
grant tableviewer to aliyun$charlie@aliyun.com;
```

## 2.2.3 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，例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。
- Jack需要进行如下操作。

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

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TrustedProject机制来支持

受保护项目空间的数据流出。您可以通过设置TrustedProject，将prj2设置为prj1的可信项目空间，具体语句如下。设置完成后，prj1中的所有数据将被允许流出到prj2。

```
use prj1;
add trustedproject prj2;
```

# 2.3 用户及授权管理

## 2.3.1 用户认证

本文为您介绍如何申请云账号、AccessKey，并使用云账号登录MaxCompute进行用户认证。

目前，MaxCompute支持云账号和*RAM*账号两种账号体系。

**说明:**

MaxCompute仅能识别RAM的账号体系，不能识别RAM的权限体系。即您可以将自身的任意RAM子账号加入MaxCompute的某一个项目中，但MaxCompute在对该RAM子账号做权限验证时，并不会考虑RAM中的权限定义。

在默认情况下，MaxCompute项目仅能识别阿里云账号系统，您可以通过`list accountproviders;`查看该项目所支持的账号系统。

通常情况下仅会看到阿里云账号，如果您想添加对RAM账号的支持，可以执行`add accountprovider ram;`。添加成功后，您可再次通过`list accountproviders;`查看所支持的账号系统是否有所变化。

### 申请云账号

如果您还没有云账号，请访问阿里云官网，申请一个属于您的云账号。

**说明:**

申请云账号时，需要一个有效的电子邮箱地址，而且此邮箱地址将被当作云账号。比如：Alice可以使用她的alice@aliyun.com邮箱来注册一个云账号，那么她的云账号就是alice@aliyun.com。

### 申请AccessKey

拥有云账号之后，您即可登录访问*AccessKeys* 页面，以创建或管理当前云账号的AccessKey列表。

一个AccessKey由两部分组成：AccessKeyId和AccessKeySecret。AccessKeyId用于检索AccessKey，而AccessKeySecret用于计算消息签名，所以需要严格保护以防泄露。当一个AccessKey需要更新时，您可以创建一个新的AccessKey，然后禁用老的AccessKey。

使用云账号登录MaxCompute

当使用odpscmd登录时，需要在配置文件*conf/odps_config.ini*中配置AccessKey的相关信息，如下所示：

```
project_name=myproject
access_id=<这里输入Access ID，不带尖括号>
access_key=<这里输入Access Key，不带尖括号>
end_point=http://service.odps.aliyun-inc.com/api
```

说明：

在阿里云网站上禁用或解禁一个AccessKey时，目前需要15分钟后才能完全生效。

## 2.3.2 用户管理

任意非项目空间Owner用户必须被加入MaxCompute项目空间中，并被授予相对应权限，方能操作 MaxCompute中的数据、作业、资源及函数。本文将介绍项目空间Owner如何将其他用户（包括RAM子账号）加入和移出 MaxCompute，如何给用户授权。

如果您是项目空间Owner，建议您仔细阅读本文。如果您是普通用户，建议您向Owner提出申请，被加入对应的项目空间后再阅读后续章节。

本文的操作均在客户端运行，Linux系统下运行`./bin/odpscmd`，Windows下运行`./bin/odpscmd.bat`。

添加用户

当项目空间的Owner Alice决定对另一个用户授权时，Alice需要先将该用户添加到自己的项目空间中来，只有添加到项目空间中的用户才能够被授权。

添加用户的命令如下：

```
add user
```

云账号的<username>既可以是在www.aliyun.com上注册过的有效邮箱地址，也可以是执行此命令的云账号的某个RAM 子账号，示例如下：

```
add user ALIYUN$odps_test_user@aliyun.com;
add user RAM$ram_test_user;
```

假设Alice的云账号为alice@aliyun.com，那么当Alice执行上述两条语句后，通过`list users;`命令即可看到如下结果：

```
RAM$alice@aliyun.com:ram_test_user
```

```
ALIYUN$odps_test_user@aliyun.com
```

这表明云账号odps_test_user@aliyun.com以及Alice通过RAM创建的子账号ram_test_user已经被加入到了该项目空间中。

### 添加RAM子账号

添加RAM子账号有以下两种方式：

- 通过DataWorks进行操作，详情请参见准备RAM子账号。
- 通过 MaxCompute 客户端常用命令进行操作，详情如下：

  

  说明：

  - MaxCompute只允许主账号将自身的RAM子账号加入到项目空间中，不允许加入其它云账号的RAM子账号，因此在add user时，无需在RAM子账号前指定主账号名称，MaxCompute默认判定命令的执行者即是子账号对应的主账号。
  - MaxCompute只能够识别RAM的账号体系，不能识别RAM的权限体系。即用户可以将自身的任意RAM子账号加入MaxCompute的某一个项目中，但MaxCompute在对该RAM子账号做权限验证时，并不会考虑 RAM中的权限定义。

  默认情况下，MaxCompute项目只能够识别阿里云账号系统，用户可以通过list accountproviders;命令查看该项目所支持的账号系统，通常情况下仅会看到ALIYUN账号，例如下所示：

  ```
  odps@ ****>list accountproviders;
  ALIYUN
  ```

  

  说明：

  只有项目空间的Owner有权限进行accountproviders的相关操作。

  由上可见，只能看到ALIYUN账号体系，如果想添加对RAM账号的支持，可以执行add accountprovider ram;如下所示：

  ```
  odps@ odps_pd_inter>add accountprovider ram;
  OK
  ```

  添加用户成功后，此用户仍不能操作MaxCompute，需要对用户授予一定权限，用户才能在所拥有的权限范围内操作MaxCompute。关于授权的更多操作，请参见授权。

### 用户授权

添加用户后，项目空间Owner或者项目空间管理员需要给该用户进行授权，只有用户获得权限后，才能执行操作。

MaxCompute提供了ACL授权，跨项目空间数据分享及项目空间数据保护等多种策略。下面列举两个常见场景，更多详情请参见ACL授权。

场景一：

假设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，并申请查看Table列表，提交作业和创建表的权限。

项目空间的admin role或者该项目空间owner在客户端执行如下命令：

```
use prj1; --进入项目空间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
```

场景二：

假设用户云账号为bob@aliyun.com，已经被添加到某个项目空间（$user_project_name），需要给它授予建表、获取表信息和执行的权限。

项目空间的admin role或者该项目空间owner在客户端可以执行如下命令：

```
grant CreateTable on PROJECT $user_project_name to USER ALIYUN$bob@
aliyun.com;
  -- 向 bob@aliyun.com 授予名为 “$user_project_name” 的project的
CreateTable（创建表）权限
grant Describe on Table $user_table_name to USER ALIYUN$bob@aliyun.com
;
  -- 向bob@aliyun.com授予名为 “$user_table_name” 的Table的Describe（获取表
信息）权限
grant Execute on Function $user_function_name to USER ALIYUN$bob@
aliyun.com;
  -- 向bob@aliyun.com授予名为 "$user_function_name" 的Function的Execute
（执行）权限
```

### 给RAM子账号授权

通过`list accountproviders;`来查看账号支持情况，如下所示：

```
odps@ ****>list accountproviders;
ALIYUN, RAM
```

由上可见，这个项目空间已经能够支持RAM账号体系，即可以向这个项目空间添加 RAM子账号并授予某张表的`Describe`权限，如下所示：

```
odps@ ****>add user ram$bob@aliyun.com:Alice;
OK: DisplayName=RAM$bob@aliyun.com:Alice
odps@ ****>grant Describe on table src to user ram$bob@aliyun.com:
Alice;
```

```
OK
```

此时，*bob@aliyun.com*的RAM子账号*Alice*就可以通过自己的AccessKeyID及AccessKeySecret登录 MaxCompute，并对表*src*进行desc操作。

说明:

- 获取RAM子账号AccessKeyID及AccessKeySecret的相关操作请参见 *RAM* 介绍。
- 更多MaxCompute添加/删除用户的操作请参见本文相关内容。
- 更多有关授权的操作请参见 授权。

### 删除用户

当一个用户离开此项目团队时，Alice需要将该用户从项目空间中移除。用户一旦从项目空间中被移除，该用户将不再拥有任何访问项目空间资源的权限。

移除用户的命令，如下所示：

```
remove user
```

说明:

- 当一个用户被移除后，该用户不再拥有访问该项目空间资源的任何权限。
- 移除一个用户之前，如果该用户已被赋予某些角色，则需要先撤销该用户的所有角色。关于角色的介绍请参考角色管理。
- 当一个用户被移除后，与该用户有关的*ACL*授权 仍然会被保留。一旦该用户以后被再添加到该项目空间时，该用户的历史的ACL授权访问权限将被重新激活。
- MaxCompute 目前不支持在项目空间中彻底移除一个用户及其所有权限数据。

Alice 执行下述两条命令，即可移除相关用户：

```
remove user ALIYUN$odps_test_user@aliyun.com;
remove user RAM$ram_test_user;
```

查看用户是否移除，命令如下：

```
LIST USERS;
```

查看结果将不会看到这两个账号。此时，表明这两个账号已经被移出项目空间。

### 删除RAM子账号

同样，您也可以通过remove user命令删除自身的RAM子账号。示例如下：

```
odps@ ****>revoke describe on table src from user ram$bob@aliyun.com:
Alice;
```

```
OK
-- 回收子账号Alice权限
odps@ ****>remove user ram$bob@aliyun.com:Alice;
Confirm to "remove user ram$bob@aliyun.com:Alice;" (yes/no)? yes
OK
-- 删除子账号
```

如果您是项目空间的owner，也可以通过`remove accountprovider`将 RAM账号系统从当前项目中删除，如下所示：

```
odps@ ****>remove accountprovider ram;
Confirm to "remove accountprovider ram;" (yes/no)? yes
OK
odps@ ****>list accountproviders;
ALIYUN
```

## 2.3.3 角色管理

角色（Role）是一组访问权限的集合，当需要对一组用户赋予相同的权限时，可以使用角色来授权。基于角色的授权可以大大简化授权流程，降低授权管理成本。当需要对用户授权时，应当优先考虑是否应该使用角色来完成。

每一个项目空间在创建时，会自动创建一个admin的角色，并且为该角色授予了确定的权限：可以访问项目空间内的所有对象、对用户或角色进行管理、对用户或角色进行授权。与项目空间Owner相比，admin角色不能将admin权限指派给用户，不能设定项目空间的安全配置，不能修改项目空间的鉴权模型，admin角色所对应的权限不能被修改。

角色管理相关命令如下：

```
    create role <rolename> --创建角色
    drop role <rolename> --删除角色
    grant <rolename> to <username> --给用户指派某种角色
    revoke <rolename> from <username> --撤销角色指派
```

说明：

- 多个用户可以同时存在于一个角色下，一个用户也可以隶属于多个角色。
- DataWorks中成员角色类型对应的MaxCompute 角色，以及各角色的平台权限详情，请参见项目管理中的项目成员管理模块。

创建角色

创建角色的命令格式，如下所示：

```
CREATE ROLE ;
```

示例如下：

假设要创建一个 player 角色，需在客户端输入如下命令：

```
create role player;
```

说明：

您创建的角色权限可以通过权限查看查看指定的用户权限。

添加用户到角色

添加用户到角色的命令格式，如下所示：

```
GRANT <roleName> TO <full_username> ;
```

示例如下：

假设要将用户bob@aliyun.com加入player角色中，需在客户端输入如下命令：

```
grant player to bob@aliyun.com;
```

给角色授权

给角色授权的语句与给用户授权相似，更多详情请参见用户授权。

说明：

给角色授权后，该角色下的所有用户拥有相同的权限。

示例如下：

假设 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 和 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，提交作业和读取表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。

操作如下：

```
    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;
```

删除角色中的用户

删除角色中的用户的命令格式，如下所示：

```
REVOKE <roleName> FROM <full_username>;
```

示例如下：

假设将用户 bob@aliyun.com从player角色中删除，需在客户端输入如下命令：

```
revoke player from bob@aliyun.com;
```

删除角色

删除角色的命令格式，如下所示：

```
DROP ROLE <roleName>;
```

示例如下：

假设要删除 player 角色，需在客户端输入如下命令：

```
drop role player;
```

说明：

删除一个角色时，MaxCompute会检查该角色内是否还存在其他用户。若存在，则删除该角色失败。只有在该角色的所有用户都被撤销时，删除角色才会成功。

## 2.3.4 授权

授权，即授予用户对MaxCompute中的表、任务、资源等客体的某种操作权限，包括读、写、查看等。

完成用户管理后，项目空间Owner或者项目空间管理员需要给该用户进行授权，只有用户获得权限后，才能执行操作。

MaxCompute提供了ACL授权、跨项目空间数据分享、项目空间数据保护等多种策略。授权操作一般涉及到三个要素，即主体（Subject，可以是用户也可以是角色）、客体（Object）和操作（Action）。在MaxCompute中，主体是指用户或角色，客体是指项目空间中的各种类型对象。我们推荐您优先使用ACL（基于对象）授权，而非Policy（基于策略）授权。

ACL授权中，MaxCompute的客体包括项目空间、表、函数、资源、任务实例。操作与特定对象类型有关，不同类型的对象所支持的操作也不相同。

MaxCompute项目空间支持的对象类型及操作

| 客体（Object） | 操作（Action） | 说明 |
| --- | --- | --- |
| Project | Read | 查看项目空间自身（不包括项目空间的任何对象）的信息，例如CreateTime等。 |
| Project | Write | 更新项目空间自身（不包括项目空间的任何对象）的信息，例如Comments。 |
| Project | List | 查看项目空间所有类型的对象列表。 |
| Project | CreateTable | 在项目空间中创建Table。 |
| Project | CreateInstance | 在项目空间中创建Instance。 |
| Project | CreateFunction | 在项目空间中创建Function。 |
| Project | CreateResource | 在项目空间中创建Resource。 |
| Project | All | 具备上述所有权限。 |
| Table | Describe | 读取Table的元信息。 |
| Table | Select | 读取Table的数据。 |
| Table | Alter | 修改Table的元信息，添加删除分区。 |
| Table | Update | 覆盖或添加Table的数据。 |
| Table | Drop | 删除Table。 |
| Table | All | 具备上述所有权限。 |
| Function | Read | 读取及执行权限。 |
| Function | Write | 更新。 |
| Function | Delete | 删除。 |
| Function | Execute | 执行。 |
| Function | All | 具备上述所有权限。 |
| Resource | Read | 读取。 |
| Resource | Write | 更新。 |
| Resource | Delete | 删除。 |
| Resource | All | 具备上述所有权限。 |
| Instance | Read | 读取。 |
| Instance | Write | 更新。 |
| Instance | All | 具备上述所有权限。 |

**说明:**

- 上述权限描述中Project类型对象的CreateTable操作，Table类型的SELECT、ALTER、UPDATE、DROP操作需要与Project对象的CreateInstance操作权限配合使用。
- 单独使用上述几种权限而没有指派CreateInstance权限是无法完成对应操作的，这与MaxCompute的内部实现相关。同样，Table的SELECT权限也要与CreateInstance权限配合使用，当跨项目操作，例如在项目A里SELECT项目B的Table，则需要有项目A的CreateInstance和项目B的Table SELECT权限 。
- 在添加用户或创建角色之后，需要对用户或角色进行授权 。MaxCompute的ACL授权是一种基于对象的授权。通过授权的权限数据（即访问控制列表，Access Control List）将被看作是该对象的一种子资源。只有当对象已经存在时，才能进行授权操作。当对象被删除时，通过授权的权限数据会被自动删除。

### MaxCompute支持的授权方法

MaxCompute支持的授权方法类似于SQL92定义的GRANT/REVOKE语法，它通过简单的授权语句完成对已存在的项目空间对象的授权或撤销授权。授权语法如下所示。

```
grant actions on object to subject
revoke actions on object from subject
actions ::= action_item1, action_item2, ...
object ::= project project_name | table schema_name |
           instance inst_name | function func_name |
           resource res_name
subject ::= user full_username | role role_name
```

MaxCompute的ACL授权语法并不支持`[WITH GRANT OPTION]`授权参数。当用户A授权用户B访问某个对象时，用户B无法将权限进一步授权给用户C。所有的授权操作都必须由以下角色的用户来完成：

- 项目空间Owner。
- 项目空间中拥有Admin角色的用户。
- 项目空间中对象创建者。

### 使用ACL授权的应用示例

假设云账号用户alice@aliyun.com是新加入到项目空间test_project_a的成员，Allen是加入到bob@aliyun.com中的RAM子账号。在test_project_a中，他们需要提交作业、创建数据表、查看项目空间已存在的对象。管理员执行的授权操作如下所示。

```
use test_project_a; --打开项目空间。
add user aliyun$alice@aliyun.com; --bob添加用户alice到项目。
add user ram$bob@aliyun.com:Allen; --bob给添加RAM子账号到Allen项目。
create role worker; --创建角色。
grant worker TO aliyun$alice@aliyun.com; --bob指派角色给alice。
```

```
grant worker TO ram$bob@aliyun.com:Allen;  --bob指派角色给自己的子账号
Allen。
grant CreateInstance, CreateResource, CreateFunction, CreateTable,
List ON PROJECT test_project_a TO ROLE worker; --对角色授权。
```

**说明:**

主账号不能给其他主账号的子账号授权。

跨项目空间Table、Resource、Function分享示例

接前一个示例，aliyun$alice@aliyun.com和ram $bob@aliyun.com:Allen在test_project_a拥有了一定的权限后，这两个用户还需查询test_project_b中的table prj_b_test_table，且需要用到test_project_b中的UDF prj_b_test_udf。test_project_b的管理员执行的授权操作如下所示。

```
use test_project_b; --打开项目空间。
add user aliyun$alice@aliyun.com; --添加用户。
add user ram$bob@aliyun.com:Allen; --添加RAM子账号。
create role prj_a_worker; --创建角色。
grant prj_a_worker TO aliyun$alice@aliyun.com; --角色指派。
grant prj_a_worker TO ram$bob@aliyun.com:Alice;  --角色指派。
grant Describe , Select  ON TABLE prj_b_test_table TO ROLE prj_a_work
er; --对角色授予table权限。
grant Read  ON Function prj_b_test_udf TO ROLE prj_a_worker; --对角色授
予udf权限。
grant Read  ON Resource prj_b_test_udf_resource TO ROLE prj_a_worker;
 --对角色授予实现udf的Resource的权限。
--授权后，这两个用户在test_project_a中查询表和使用udf的方式如下。
use test_project_a;
select test_project_b:prj_b_test_udf(arg0, arg1) as res from
test_project_b.prj_b_test_table;
```

**说明:**

如果是在test_project_a中创建UDF，则授权时只需进行Resource授权，使用写法如下。

```
create function function_name as 'com.aliyun.odps.compiler.udf.
PlaybackJsonShrinkUdf' using 'test_project_b/resources/odps-compiler-
playback.jar' -f;
```

## 2.3.5 权限查看

MaxCompute支持从多种维度查看权限，具体包括查看指定用户的权限、查看指定角色的权限、以及查看指定对象的授权列表。

在展现用户权限或角色权限时，MaxCompute使用如下标记字符：

- A：表示Allow，即允许访问。
- D：表示Deny，即拒绝访问。
- C：表示With Condition，即为带条件的授权，只出现在Policy授权体系中。

- G：表示With Grant Option，即可以对Object进行授权。

展现权限的示例如下。

```
odps@test_project> show grants for aliyun$odpstest1@aliyun.com;
[roles]
dev
Authorization Type: ACL
[role/dev]
A       projects/test_project/tables/t1: Select
[user/odpstest1@aliyun.com]
A       projects/test_project: CreateTable | CreateInstance |
CreateFunction | List
A       projects/test_project/tables/t1: Describe | Select
Authorization Type: Policy
[role/dev]
AC      projects/test_project/tables/test_*: Describe
DC      projects/test_project/tables/alifinance_*: Select
[user/odpstest1@aliyun.com]
A       projects/test_project: Create* | List
AC      projects/test_project/tables/alipay_*: Describe | Select
Authorization Type: ObjectCreator
AG      projects/test_project/tables/t6: All
AG      projects/test_project/tables/t7: All
```

查看指定用户的权限

```
show grants; --查看当前用户自己的访问权限。
show grants for <username>; --查看指定用户的访问权限，只有Project Owner和
Admin才有执行权限。
show grants for RAM$主帐号:子帐号; --查看RAM子帐号权限。
```

示例

- 查看指定用户云账号bob@aliyun.com在当前项目空间的权限

```
show grants for ALIYUN$bob@aliyun.com;
```

- 查看RAM子帐号RAM$bob@aliyun.com:Alice在当前项目空间的权限

```
show grants for RAM$bob@aliyun.com:Alice;
```

查看指定角色的权限

```
describe role ; --查看指定角色的访问权限角色指派。
```

**说明:**

公共云环境下，`describe role`目前只显示Project和Table的对象授权类型ACL信息，其它对象（例如Function、Resource、Instance、Job）的授权类型ACL不显示。

查看指定对象的授权列表

```
show acl for <objectName> [on type <objectType>]; --查看指定对象上的用户和角色授权列表。
```

说明:

当省略`[on type <objectType>]`时，默认的type为Table。

## 2.4 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）。它的引入使得项目空间管理员可以更加灵活地控制用户对列级别敏感数据的访问。

### 理解MaxCompute中MAC与DAC的差异

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，还不可以酒后驾驶。类比MaxCompute，这个禁止酒驾就相当于禁止读取敏感度高的数据。这些属于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，通常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，通常还支持更多复杂的安全策略，例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。
> - 为了控制数据在不同项目空间之间的流动，您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，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。使用SET LABEL设置标签时，必须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用户对一张表种某些敏感列的读访

  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机制已经授权了所有用户对该表的SELECT操作。ProjectOwner希望除了Admin之外的其他用户都不被允许访问那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：**

执行以上操作后，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，则需要获得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。当上述授权成功后，Alice将有权限访问这三列数据，但此处存在轻微地过度授权。管理员通过合理设置数据列的敏感度，避免过度授权。

- 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，即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。
```

## 2.5 跨项目空间的资源分享

### 2.5.1 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。

# 2.5.2 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。它在安装了资源提供方发布的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：
>
> - 只有Project的Owner才有权限进行该操作。
> - 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在其它Project中创对象。
> - 添加资源时，对象名称不能加项目名前缀。例如当前Project为prj1，需要添加表table_test到某个Package中，则进行Add操作时，表名不能写成prj1.table_test，应该直接写表名table_test。
> - 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。如果有需要，则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时，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>;
```

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

说明:
只有Project的Owner才有权限进行该操作。

- 卸载Package

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

卸载Package时，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（即其它项目空间分享给您的资源），您必须拥有该Package的Read权限。如果请求者没有Read权限，则需要向Project Owner或Admin申请。Project Owner或Admin可以通过ACL授权机制完成此授权。

  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。

  示例

  - ACL授权允许云账号用户aliyun$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;
```

  - ACL授权允许角色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（例如*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，则管理员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; --添加表到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John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. John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。
```

## 2.6 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。在您进入项目空间后，可执行下列安全配置命令：

```
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。

# 2.7 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(如金融机构等)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，您会遇到不允许用户将数据转移到项目空间之外类似的安全需求 。

比如，当项目空间prj1的Owner遇到可以访问prj2，而prj2又不受控制的情形时，可能会担心用户Alice将她能访问的数据转移到prj2中去。

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。

则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，您可以将MaxCompute上述导致数据流出的操作统统禁止。

## 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。

默认情况下，ProjectProtection不会被设置，默认值为false。

同时在多个项目空间中拥有访问权限的用户，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Exception Policy

  Project Owner在设置ProjectProtection时可以附带一个exception策略，命令如下。

  ```
  SET ProjectProtection=true WITH EXCEPTION <policyFile>
  ```

  这种policy不同于Policy授权(尽管这种policy与Policy授权语法完全一样)，它只是对项目空间保护机制的例外情况的一种描述，即所有符合policy中所描述的访问情形都可以打破ProjectProtection规则。

  ProjectProtection是对数据流向的控制，而不是访问控制。只有在用户能访问数据的前提下，控制数据流向才是有意义的。

  示例

  允许云账号Alice@aliyun.com可以通过SQL任务对表alipay.table_test执行SELECT操作时将数据流出到alipay项目空间之外。

  ```
  {
  "Version": "1",
  "Statement":
  [{
      "Effect":"Allow",
      "Principal":"ALIYUN$Alice@aliyun.com",
      "Action":["odps:Select"],
      "Resource":"acs:odps:*:projects/alipay/tables/table_test",
      "Condition":{
          "StringEquals": {
              "odps:TaskType":["DT", "SQL"]
          }
      }
  }]
  }
  ```

   说明:

Exception policy不是一种普通的授权方式。如果云账号Alice没有表alipay.table_test的SELECT操作权限，即使设置了上述exception policy，Alice仍然无法导出数据。

TOC2TOU (Time-of-Check to Time-of-Use) 数据泄露问题（即Race Condition问题）的描述如下：

- [TOC阶段] 用户A向Project Owner申请将t1导出，Project Owner对t1的数据敏感程度进行评估，PASS后通过exception policy授权A可以导出t1。
- 恶意用户修改了t1的内容，将敏感数据写入到t1。
- [TOU阶段] 用户A将t1的内容导出。但是，此时导出的t1并不是Project Owner审查的t1。

关于防止出现TOC2TOU问题的建议：对于用户申请导出的表，Project Owner需要确保没有任何其他用户（含admin）能对该表进行更新(Update)操作或重建同名表操作(Drop + CreateTable)。在上述示例中，为防止出现TOC2TOU问题，建议Project Owner在第一步中以Project Owner身份创建t1的一个snapshot，设置exception policy时使用这个snapshot；并且不要授予admin角色给任何用户。

- 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。

**实践建议**

**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。**

# 2.8 安全相关语句汇总

## 2.8.1 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 |
| 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 |

## 2.8.2 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 |

| 语句 | 说明 |
| --- | --- |
| 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 |

## 2.8.3 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 |
| 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
```

# 3 安全管理案例

## 3.1 创建项目

通过安全管理基础篇了解到MaxCompute和DataWorks的相关安全模型，以及学习了各种安全功能配置后，您就可以开始动手实践安全管理案例了。本章节我们通过列举两个常见的基础业务需求来介绍项目创建和管理。

### 创建基本ETL开发业务项目

#### 场景描述

多人协同开发，成员责任划分明确，需遵循正常的开发、调试、发布流程，生产数据查看须严格控制。

#### 需求分析

- 多人协同开发，DataWorks项目本身就满足这一点。
- 成员责任划分明确，DataWorks的基础成员角色（项目管理、开发、运维、部署、访客）基本可以满足需求。
- 遵循正常开发、调试、发布流程，生产数据需严格控制。通过在DataWorks上创建并区分开发生产项目，可以实现控制。

#### 操作步骤

1. 创建项目

   a. 创建项目主要配置如图所示。

   - 项目模式选择标准模式（开发跟生产隔离）。该模式创建的结果是一个DataWorks项目空间绑定关联两个MaxCompute project（开发和生产project）。开发环境上进行开发调试，生产环境任务由开发环境通过发布流程发布过来进行稳定运行。
   - 开发环境MaxCompute访问身份为个人账号。项目成员在开发环境做任务开发调试时用个人账号进行操作。这样做的目的在于：每个成员根据业务需求在开发调试过程中需要用到的MaxCompute各种生产资源（table、Resource、function）不同。为防止权限过大，每个成员自己申请自己所需的权限（主要是生产表权限），同时还可以更好的做后续的安全审计。
   - 生产环境MaxCompute访问身份为项目负责人账号即project的owner。生产环境要保障稳定和安全生产，正常情况下：不允许成员可以随意提交job，不允许个人账号拥有生

产表的删除、修改权限以免通过命令行工具进行操作无法受到更好的控制。个人账号需要读生产表的权限示，必须进行申请以便更好的管理数据安全。

2. 添加项目成员

DataWorks上添加RAM子账号为项目成员，按需分配角色。同时，对应的开发环境project会将对应的role授权给子账号。

- 项目管理员：除拥有开发角色和运维角色全部权限外，还可以进行添加/移出项目成员并授予角色创建自定义资源组等项目级别的操作。同时拥有MaxCompute开发project的role_project_admin这个role。
- 开发：负责数据开发页面设计和维护工作流。同时拥有MaxCompute开发project的role_project_dev这个role。
- 运维：负责在运维中心页面管理全部任务的运行情况并做相应处理。同时拥有MaxCompute开发project的role_project_pe这个role。
- 部署：仅在多项目模式时审核任务代码并决定是否提交运维。同时拥有MaxCompute开发project的role_project_deploy这个role。
- 访客：仅有只读权限，可查看数据开发页面的工作流设计和代码内容。同时拥有MaxCompute开发project的role_project_guest这个role。
- 安全管理员：仅有数据保护伞模块的操作权限，无其他模块权限。同时拥有MaxCompute开发project的role_project_security这个role。

3. 任务开发调试

开发角色成员在DataWorks的数据开发模块（对应MaxCompute开发project）进行任务开发调试，其间用到的生产project表，可以到DataWorks的数据管理模块进行申请。

4. 任务发布到生产环境

开发角色成员调试好任务后，进行打包。运维角色成员可以进行代码review（开发角色成员需要线下通知运维角色成员这个流程）后执行发布包将任务发布到生产环境。 这个过程保障任务不能随意发布到生产环境执行。

5. 开发成员生产任务测试

任务发布到生产环境后，建议开发成员还需要到运维中心对生产环境任务测试执行一次，以确保生产任务的可正常执行。若任务执行返回成功状态，还是需要先查看日志判断执行是否正常，进一步验证就需要查询结果表是否有正常的产出。此时，通常您需要在开发界面进行表查询，而个人对生产环境产出的表默认无权限，可以到DataWorks的数据管理模块进行申请。

说明:

- DataWorks的数据开发模块是多人协同开发，所有本项目的成员都可以查看任务代码，且有编辑权限的成员都可以进行修改编辑。因此，无法很好地保密一些核心的敏感度高的代码。有类似高保密性的任务及数据，目前可以由单独项目的固定成员进行开发。
- 生产环境都是通过project owner访问MaxCompute，因此创建的table、function、resource的owner显示的是project owner的账号。这样会出现创建的表的owner不是创建者自己、创建表的人没有权限查看自己建的表的情况。
- 由于开发和生产项目owner都是同一个账号，请谨防通过发布任务到生产项目将生产项目表读写到开发项目，再通过开发项目获取生产数据。

单项目且每个成员只能操作自己创建的表

场景描述

业务单一，成员角色基本一致，后续业务不会扩展。如专供取数，即不做数据开发，只需要查询下载业务数据（例如运营角色需要获取一些数据进行分析）。

需求分析

- 本项目不做数据开发，则需要分析的数据必定是在其他项目中。为了避免不同主账号资源隔离，本项目的owner（主账号）必须与数据开发生产项目的owner同一账号。
- 本项目目的主要为完成数据查询下载，所以需要每个成员用自己的权限进行数据查询下载。因此这个项目的MaxCompute设置MaxCompute访问身份属性为个人账号。
- 当设置MaxCompute访问身份属性为个人账号后，DataWorks中每个项目成员将会被授予对应MaxCompute的role权限。由于需求是每个成员只能操作自己创建的表，因此您需要处理好这个默认的role权限。

操作步骤

1. 创建项目

   a. 主账号必须是需要分析的数据所在的项目主账号，创建项目主要配置如图所示。

2. 创建MaxCompute自定义role并授权

   主账号通过客户端进行操作。

```
create role custom_dev;--创建自定义role
```

```
grant List, CreateInstance,CreateTable,CreateFunction,CreateResource
 on project prj_name to role custom_dev;--给自定义role赋权
```

3. MaxCompute的project设置允许对象创建者默认拥有访问权限

   主账号通过客户端进行操作。

```
set ObjectCreatorHasAccessPermission=true;--实际上这个flag默认已经为
true，可以通过如下命令查看。
show SecurityConfiguration;
```

   也可以在DataWorks的项目管理 > MaxCompute高级配置中进行配置。

4. 添加项目成员

   DataWorks上添加子账号为新成员。例如添加成员时角色为开发，则添加成功后，在对应MaxCompute的project里该成员对应的role是role_project_dev。主账号可以通过`show grants for ram$主账号:子账号;`命令行进行查看。

5. 修改新成员的MaxCompute权限

   主账号通过客户端进行操作。

```
revoke role_project_dev from ram$主账号:子账号;--将新成员从默认授予的role
中移除。
grant custom_dev to ram$主账号:子账号;--给新成员授予自定义角色。
```

> **说明:**
>
> - 该项目的成员若重新操作添加如上描述中的开发角色，则成员又会重新被授予role_project_dev的role。
> - 该项目经过上述配置后，只能做到每个成员可以查看自己创建的表（对象），但是做不到每个成员只能看到自己创建的任务。
> - 该项目成员需要查询的表的权限必须由自己通过正常的权限申请流程（可在DataWorks的数据管理中申请），或者通过package授权方式，把其他生产项目的表加到package中，再将package安装到该项目并授权给成员。详情可参见用户与权限管理。

## 3.2 Package赋权

业务分析人员需要查看生产表，但是不允许查看生产任务代码。这种情况下，我们可以通Package赋权，将多个生产项目的部分表开放给业务分析人员。

### 场景分析

业务分析人员需要查看生产表但是又不能查看生产任务，我们可以通过单独创建一个分析项目来实现。首先，在多个生产项目创建package，把需要开放的表加到package中。然后，在分析项目中

安装package并授权给分析人员。这样可以减少成员管理成本，无需在所有生产项目中增加分析人员，同时保证这些分析人员只能在分析项目中查看package中的表。

操作步骤

1. 生产项目中创建Package

```
CREATE PACKAGE [pkgname]
如:
CREATE PACKAGE prj_prod2bi;
```

2. 生产项目中向Package中添加需要分享的资源

```
ADD table [table_name] TO PACKAGE [包名称];
如:
ADD table adl_test_table TO PACKAGE prj_prod2bi;
```

3. 生产项目许可分析项目空间使用Package

```
ALLOW PROJECT [允许安装的 project] TO INSTALLPACKAGE [包名称];
如:
ALLOW PROJECT PRJ_BI TO INSTALLPACKAGE prj_prod2bi;
```

4. 分析项目安装Package

```
INSTALL PACKAGE [应用名].[包名称];
如:
INSTALL PACKAGE prj_prod.prj_prod2bi;
```

5. Package赋权给使用者

```
赋权给user:
GRANT read on package prj_prod2bi TOUSER [云账号];
赋权给role:
GRANT read on package prj_prod2bi TOROLE [rolename];
```

## 3.3 数据安全自查

本文主要举例介绍在进行数据安全自查后应该重点调整的方向。为您提供数据安全的调整思路。

场景分析

在项目初期，为了加快进度，一些用户和权限管理相对宽松。当项目工作进入了一个相对稳定发展阶段后，数据安全将成为管理方面越来越重要的抓手。此时，您需要对数据安全进行自查分析，生成一个方案并落地。

自查思路

1. 账号数量统计。统计DataWorks项目成员和MaxCompute项目成员，确认每个成员拥有且只拥有一个工作账号，便于责任追责和管理。

2. 废弃账号及权限统计：对于已在MaxCompute或DateWorks项目中拥有角色的RAM子账号，请在删除子账号之前解除子账号在项目的角色，并在项目空间中删除子账号。否则，子账号会在项目空间中残留，显示为p4_xxxxxxxxxxxxxxxxxxxx且无法在项目空间中移除（不影响项目空间正常功能使用）。 但若是因职位发生变化的账号及遗留权限，需要回收。建议不被长期使用的成员账户，或是申请了但是长期不用的账户，可以在通过通知、调研后，清理这部分账户，待需要使用权限时再申请。
3. 个人账号调查分析（可以工单申请推送元仓数据进行分析统计）。通过调查个人账号最近3个月在开发阶段提交的查询（提交的数据检索、计算任务，主要是以SQL任务），统计topN用户，并选取代表性账号分析其日常任务。

   例如，账号对应成员主要工作的项目空间为算法开发项目，日常工作主要执行的任务是SQL任务，执行的SQL任务主要为开发环境的查询和写表操作。算法任务、MR任务相对SQL任务数量较少，但是都有分布。这也符合数据开发的实际情况，能用SQL处理，一般优先使用SQL处理数据。

   又如，某账号提交的任务非常多，经了解，其将自己的AK通过SDK的方式配置了一个查询软件，并提供多人进行查询。请谨慎开放权限，避免多人共用同一个账号。
4. 数据下载统计（可以工单申请推送元仓数据进行分析统计）。统计各个项目的数据下载请求任务，分析规划可下载项目。

调整要点

- 账号以及全新合理分配

  以每个工作成员都使用自己的个人账户为调整原则。

  针对不同人员所在的的不同业务开发小组和角色给出不同的数据访问权限，禁止相互借用他人的账户使用。避免因为用户权限过大导致的数据安全风险。 例如，按数据开发过程的业务分组进行账号分配。分组如管理组、数据集成组、数据模型组、算法组、分析组、运维组、安全组等。

- 数据流动控制

  限制部分项目的数据导出，控制部分人员的权限。数据随意在各个项目之间流动，不但会导致云平台数据架构混乱，同样也会导致数据泄露的风险。所以，针对大部分项目需要作出数据流动的限制。

  例如，通过MaxCompute层面限制数据只能流动到指定的项目或者指定的位置，从而规避未知数据流动带来的风险。

- 数据导出限制

  数据一旦从MaxCompute落地为文件，就意味着数据不可控。所以，必须要尽可能的减少数据落地带来的风险。通过用户角色的详细划分，限制只有一些业务组可以有数据导出的权限，并且也不会影响日常开发工作。

## 3.4 行级别权限控制

本文以案例分析的形式为您介绍如何实现行级别权限控制。

### 场景分析

Project A的table_order是所有商家的订单交易信息表。该表可以开放给商家查看，但每个商家只能查看自己家店铺的订单交易信息。

### 方案设计

table_order表中有商家id，可以根据商家id进行过滤，将各个商家限制于只读自己的数据，因此需要行级别的权限控制。MaxCompute不支持行级别的权限控制，但您可以通过如下方案实现行级别权限控制的需求：

- 方案一：在table_order表下游单独为每个商家创建独立的表（table），将表赋权给对应的商家。这种方式可以满足行级别权限控制的需求，但会导致数据重复存储。一旦table_order表数据有更新，下游的表也需要同步更新才能保持数据一致。
- 方案二：在table_order表下游单独给每个商家创建独立的视图（view），将视图赋权给对应的商家。这种方式可以满足行级别权限控制的需求，同时避免了方案一的弊端。

评估两个方案之后，建议您采用方案二，通过创建视图方式实现行级别权限控制。具体操作如下：

1. 在Project A中创建视图。

```
CREATE VIEW <viewname> as select * from table_order WHERE sellerid='
xxxx';
```

2. 在Project A中创建package，通过package资源共享方式将view授权给商家。

```
create package <packagename>;
add table <viewname> to package <packagename>;
allow project <Projectname_seller> to install package <packagename>;
```

3. 商家使用视图。

```
--命令均在商家的Project中操作
install package <ProjectA>.<packagename>;
```

```
grant read on package <ProjectA>.<packagename> to user <username
>;--商家项目中具体需要查询view的账号
```

**说明:**

**本案例演示的是通过package方式授权视图权限，您也可以直接ACL给user赋予视图的select和describe权限。具体的使用方式取决于现实业务需求。**

```
grant select,describe on table <viewname> to user <username>;
```

# 4 数据质量管理

## 4.1 数据质量保障原则

不同行业有不同的评估数据质量的标准。数据质量可以从完整性、准确性、一致性和及时性共四个角度进行评估。

- 完整性

  完整性是指数据的记录和信息是否完整，是否存在数据缺失情况。数据缺失主要包括记录的缺失和记录中某个字段信息的缺失，两者都会造成统计结果不准确。完整性是数据质量最基础的保障。如果某个相对稳定的业务数据量每天的都有100万条记录，某天突然下降了1万条，则可能是出现了记录缺失。如果某科高考成绩表中，每个考卷分数都对应一个准考证号，当准考证号字段的空值数大于0时，则可能是出现了信息缺失。

- 准确性

  准确性是指数据中记录的信息和数据是否准确、是否存在异常或者错误的信息。例如成绩单中分数出现负数或订单中出现错误的买家信息等，这些数据都是问题数据。确保记录的准确性也是保证数据质量的一个必不可少的原则。

- 一致性

  一致性一般体现在跨度很大的数据仓库中。 例如公司中有很多业务数仓分支，对于同一份数据必须保证一致性。例如从在线业务库加工到数据仓库，再到各个数据应用节点，用户ID必须保持同一种类型，且长度也要保持一致。因此，在数仓建设规范中会有“公共层”加工，以确保数据的一致性，详情请参见#unique_74。

- 及时性

  保障数据的及时产出，体现数据的价值。例如决策分析师通常希望当天就可以看到前一天的数据。若等待时间过长，则数据失去了及时性的价值，数据分析工作将变得毫无意义。

## 4.2 数据质量管理流程

本文为您介绍数据质量定义、数据资产的等级定义及落地方法。

数据质量定义

通过划分数据资产等级和分析元数据的应用链路，您可以确定哪些数据需要做质量管理。根据应用的影响程度，确定数据资产等级；根据数据链路血缘，将数据资产等级上推至各数据生产加工的各

个环节，确定链路上所涉及的数据的资产等级和在各个加工环节上根据资产等级的不同所采取的不同处理方式。

### 数据资产等级定义

对于数据的资产等级，在质量管理方面，可以依据数据质量“不满足四个原则”对业务的影响性质进行划分。例如划分为5个等级的性质：毁灭性质、全局性质、局部性质、一般性质、未知性质。这些性质的重要性顺序降低，具体定义如下：

- 毁灭性质，即数据一旦出错，将会引起重大资产损失，面临重大收益损失等。
- 全局性质，即数据直接或间接用于企业级业务和效果评估、重要决策等。
- 局部性质，即数据直接或间接用于某些业务线的运营、报告等，若出现问题会给业务线造成一定的影响或造成工作效率降低。
- 一般性质，即数据主要用于日常数据分析，出现问题带来的影响极小。
- 未知性质，即无法明确数据的应用场景。

资产等级可以用Asset进行标记：毁灭性质为A1，全局性质为A2，局部性质为A3，一般性质为A4，未知性质为Ax。重要程度为：A1>A2>A3>A4>Ax。若一份数据出现在多个应用场景汇总，则根据其最高重要程度进行标记。

### 数据资产等级落地方法

定义并划分好数据资产等级后，需要考虑如何落地，对数仓中庞大的数据量进行资产等级打标。您可以从使用数据流转链路开始进行数据资产等级打标。

MaxCompute进行数据加工基本流程：数据从业务系统上产生，通过同步工具（DataWorks的数据集成或阿里云DTS）进入数据数仓系统（MaxCompute），数据在数仓中进行清洗、加工、整合、算法、模型等一系列运算后，再通过同步工具输出到数据产品中进行消费。整个流程数据都是以存放在表的形式体现，流转链路大致如下图所示。

从数据流转链路上，整理哪些表是被哪些应用业务产品消费，通过给这些应用业务产品划分数据资产等级，再结合数据的上下游血缘，将整个链路打上某一类资产等级的标签。例如，一个A2等级的的数据应用产品，对应导入这个数据产品的table即数仓（MaxCompute）的导出表Table1、Table2、Table3，几个表都打上A2-xxx数据产品标记，根据血缘往上追溯，将这几个表的上都打上A2的标记，一直标记到源数据业务系统，如图所示。

通过以上方式完成数据资产等级的确认，给不同的数据定义不同的重要程度。

# 4.3 数据加工过程卡点校验

在线系统数据加工过程卡点校验，主要是指在业务系统的数据生成过程中进行的卡点校验。

在线系统卡点校验

在线业务系统产生的数据是数据仓库的数据来源之一。在线业务系统复杂多变，每次变更都会产生数据的变化，因此，数仓需要适应多变的业务发展，及时保障数据的准确性。此外，如何能将在线业务的变更高效地通知给基于MaxCompute的离线数仓，也是需要考虑的问题。

建议您同时注重工具和人员管理：既要在工具上自动捕捉每一次业务的变化，也要求开发人员下意识地进行业务变更通知。

- 注重发布平台

  在业务进行重大变更时，订阅这个发布过程，通知离线开发人员，使其知晓此次变更内容。当业务系统足够繁杂，日常发布变更频繁的情况下，若每次变更都通知离线业务，会造成不必要的时间浪费，也影响业务迭代效率。

  此时，可以通过使用数据资产等级的标识对业务进行打标。针对高等级的数据资产，整理出何种类型的变更会影响数据的加工。例如对于相关财务报表而言，如果业务系统的改造影响财务报表的计算，导致约定好的计算口径被业务系统发布变更修改，那么这种情况必须告知离线业务，离线开发人员也必须主动关注这类发布变更通知。

  > 说明:
  > 发布平台不是指阿里云提供发布平台，只是一种统称，是各个企业自己在线业务的相关发布平台。

- 注重数据库的变化感知

  随着业务的发展，业务数据库（MaxCompute数仓的数据源）会出现数据库扩容或者DDL变更，这些变更都要主动通知到离线开发人员。基于MaxCompute的数据仓库在进行离线数据抽取时，通过DataWorks的数据集成工具，可能会限制某个业务数据库表。如果该数据库表发生扩容或者迁移等，数据集成工具感知不到，可能导致数据抽取错漏，而一旦错漏，会影响下游所有依赖该表的应用，因此建议业务数据库也需要有库表变更通知。

- 注重操作工具的人员

  操作工具只是一种辅助手段，操作工具的人员才是核心。数据资产等级的上下游打通的过程需要通知给在线开发人员，使其知晓哪些是重要的核心数据资产，提高在线开发人员的数据风险意识。通过培训等方式将离线数据的诉求、离线数据的加工过程、数据产品的应用方式告诉在线业务开发人员，让其了解数据的重要性，了解数据的价值，同时也告知出错的后果。确保在线开发人员在完成业务目标时，也要考虑数据的目标，做到业务端和数据端一致。

离线系统卡点校验

MaxCompute进行数据加工的基本流程为：数据从业务系统上产生，通过同步工具（DataWorks的数据集成或阿里云DTS）进入数仓系统（MaxCompute），数据在数仓中进行清洗、加工、整合、算法、模型等一系列运算后，再通过同步工具输出到数据产品中进行消费。整个流程中，有了数据加工，才有了数据仓库模型和数据仓库代码的建设，如何保障数据加工过程中的质量是离线数据仓库保障数据质量的一个重要环节。

MaxCompute进行数据加工，可以通过DataWorks、MaxCompute studio、MaxCompute SDK提交各种任务进行加工。无论用什么工具，都会经历代码开发到测试、发布到运维、变更的过程。您可以对这个过程中的每个环节进行卡点校验。

- 代码提交的卡点校验

  即在SQL提交前进行相关规则校验。目前公共云没有直接可用的工具辅助校验，有能力的用户可以自己开发相关的工具。规则分类如下：

  - 代码规范类规则，如表命名规范、生命周期设置、表注释等。
  - 代码质量类规则，如分母为0提醒、NULL值参与计算影响结果提醒、插入字段顺序错误等。
  - 代码性能类规则，如分区裁剪失效、扫描大表提醒、重复计算检测等。
- 任务发布上线时的卡点校验

  为保障线上数据的准确性，每次变更都需要经过测试再发布到线上生产环境，且生产环境测试通过后才算发布成功。
- 任务变更或数据重跑

  在进行更新操作前，需要通知下游变更原因、变更逻辑、变更时间等信息，下游对此次变更没有异议后再按照约定时间执行发布变更，这样可以将变更对下游的影响降到最低。

## 4.4 数据风险点监控

本文为您介绍数据在线数据风险点监控和离线数据风险点监控。

在线数据风险点监控

在线业务系统的数据生成过程中必须确保数据质量，要根据业务规则对数据进行监控。MaxCompute本身没有配套的工具，需用户自己实现。

您可以针对数据库表的记录进行规则校验，制定监控规则。在业务系统中，当每个业务过程进行数据落库时，对数据进行校验。监控规则如交易系统中，订单拍下时间、订单完结时间、订单支付金额、订单状态流转都配置校验规则。订单拍下时间不会大于当天时间，也不会小于业务系统上线时

间，一旦出现异常校验则报错。当业务繁杂、规则繁多、规则配置等运行成本高时，同样根据数据资产等级进行监控。

离线数据风险点监控

- 数据准确性

  数据准确性是数据质量的关键，因此数据准确成为数据直连的重要因素，也是所有离线系统加工时的第一保障要素。本文主要介绍使用DataWorks的数据质量工具——DQC保障MaxCompute离线数据的准确性。

  > 说明:
  > 若要使用DQC，则必须使用DataWorks进行任务调度执行。

  DQC以数据集（DataSet）为监控对象，当离线MaxCompute数据发生变化时，DQC会对数据进行校验，并阻塞生产链路，以避免问题数据污染扩散。DQC还提供了历史校验结果的管理，方便数据质量的分析和定级。

  DQC主要通过配置数据质量校验规则，自动在数据处理过程中进行数据质量监控。DQC可以监控数据质量并报警，但它不对数据产出进行处理，需要报警接收人判断并决定如何处理。

  DQC数据监控规则有强规则和弱规则之分。强规则，一旦触发报警就会阻断任务的执行（将任务置为失败状态，使下游任务不会被触发执行）；弱规则，只报警但不阻断任务的执行。DQC提供了一些常用的规则模板，包括表行数较N天前波动率、表空间大小较N天前波动率、字段最大/最小/平均值相比N天前波动率、字段空值/唯一个数等等，更多内容请参见数据质量概述。

  DQC的工作流程如下图所示。

  DQC的检查是通过运行SQL任务实现，只是这个任务是嵌套在主任务中。检查次数太多会影响整体的任务执行性能，因此哪些数据需要配置DQC规则、应该配置什么规则，也要根据数据资产等级来确定。例如A1、A2类数据监控率要达到90%以上，规则类需要3种以上，而不重要的数据资产不做强要求。

  类似的规则都是由离线开发人员进行配置来确保数据准确性。不同的业务会有不同的业务规则的约束，这些规则来源于数据产品或消费的业务需求，配置消费节点，然后上推到离线系统的起点进行监控，做到规则影响最小化。

- 数据的及时性

  在确保数据准确性的前提下，要进一步让数据能够及时的提供服务，否则数据的价值将大幅降低，甚至变得无价值，所以确保数据及时性也是保障数据质量的重要一环。

  基于MaxCompute的离线任务，如常见的以天作为时间间隔，对于天任务，一些重要的业务会对数据产出有时间要求。比如一些决策报表要求早上9点或更早的时间点之前必须产出。为确保数据完整性，天任务一般都是0点开始执行计算前一天的数据。这些任务大多在夜里运行，要确保数据按时产出，需要考虑任务的优先执行（当Project里任务很多而资源有限时）和任务执行失败或时间过长时的报警问题。

  - 任务优先级

    MaxCompute平台上任务优先级都一样，无法配置。因此要对MaxCompute的任务实现“优先级”功能，只能从调度平台入手，优先调度下发重要的任务。

    对于DataWorks平台的调度任务，当对应的Project是使用预付费资源（预购固定的计算资源仅供当前项目使用）时，可以通过“智能监控”工具进行优先级设置。DataWorks的调度是一个树形结构，当配置了叶子节点的优先级，这个优先级会传递到所有的上游节点，而叶子节点往往就是服务业务的消费节点。因此在优先级的设置上，要先确定业务的资产等级，等级越高的业务对应的消费节点优先级配置越高，优先调度从而优先占用计算资源，确保高等级业务准时产出。

    > 说明:
    > 当DataWorks的节点任务所属的Project使用的是MaxCompute的后付费资源（计算按量付费，无固定资源使用），智能监控配置的优先级无效。因此，需要评估是否要购买预付费资源，同时对任务进行优化，减少不必要的资源浪费，力争在有限的资源下更高效的完成计算。

  - 任务报警

    任务报警和优先级类似，通过DataWorks的“智能监控”工具进行配置，只需要配置叶子节点即可向上游传递。任务执行过程中可能出错或者出现延迟，为了保障最重要数据（资产等级高）产出，则需要“出错”立即处理、“可能”延迟必须知晓并介入。

  - DataWorks智能监控

    对于MaxCompute的离线任务，通过DataWorks进行离线任务调度时，DataWorks提供智能监控工具，对调度任务进行监控告警。智能监控是DataWorks任务运行的监控及分析系

统。根据监控规则和任务运行情况，智能监控决策是否报警、何时报警、如何报警以及给谁报警。智能监控会自动选择最合理的报警时间，报警方式以及报警对象。

智能监控旨在：

- 降低您的配置成本
- 杜绝无效报警
- 自动覆盖所有重要任务

## 4.5 数据质量衡量

在了解保障基于MaxCompute的数据仓库数据质量的方案后，您还需要进一步学习如何制定一套标准度量方案是否合适以及如何改进。例如，针对每一个数据质量事件发生，必须分析原因，处理过程，制定后续同类事件预防方案。将严重的数据质量事件升级为故障，并对故障进行定义、等级划分、处理、回顾。

相关工具链接

DataWorks——数据质量管理工具，请参见数据质量概述。

DataWorks——智能监控工具，请参见智能监控概述

# 5 MaxCompute管家

MaxCompute管家系统状态、资源组分配、任务监控等功能，可帮助您解决预付费计算资源的监控管理问题。

您通过预付费方式购买MaxCompute计算资源后（例如，您已购买了150CU资源），发现使用预付费计算资源的项目中，仍有很多任务需在队列中长时间等待。这种情况下，您的管理或运维人员希望能够查看具体是哪些任务占用了计算资源，从而对任务进行合理的管控。例如，根据任务对应的业务优先级调整调度时间，将重要和次要的任务错开调度等。

通过MaxCompute管家，您可以快速查看系统状态、监控任务执行详情，同时调配资源组对任务进行合理的管控，解决上述问题。具体使用说明请参考*MaxCompute*预付费资源监控工具-*CU*管家。

**说明:**

您必须购买了MaxCompute CU预付费资源才可以使用MaxCompute管家。CU资源过小时可能无法发挥计算资源和MaxCompute管家的优势。使用MaxCompute管家，建议您至少已购买60或以上CU资源。