Alibaba Cloud MaxCompute

Security

Issue: 20190122

Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.
2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company, or individual in any form or by any means without the prior written consent of Alibaba Cloud.
3. The content of this document may be changed due to product version upgrades, adjustments, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and the updated versions of this document will be occasionally released through Alibaba Cloud-authorized channels. You shall pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.
4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides the document in the context that Alibaba Cloud products and services are provided on an "as is", "with all faults" and "as available" basis. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not bear any liability for any errors or financial losses incurred by any organizations, companies, or individuals arising from their download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, bear responsibility for any indirect, consequential, exemplary, incidental, special, or punitive damages, including lost profits arising from the use or trust in this document, even if Alibaba Cloud has been notified of the possibility of such a loss.
5. By law, all the content of the Alibaba Cloud website, including but not limited to works, products, images, archives, information, materials, website architecture, website graphic layout, and webpage design, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectu

al property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of the Alibaba Cloud website, product programs, or content shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates).

6. Please contact Alibaba Cloud directly if you discover any errors in this document.

Generic conventions

Table -1: Style conventions

Style	Description	Example
	This warning information indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results.	 **Danger:** Resetting will result in the loss of user configuration data.
	This warning information indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results.	 **Warning:** Restarting will cause business interruption. About 10 minutes are required to restore business.
	This indicates warning information, supplementary instructions, and other content that the user must understand.	 **Notice:** Take the necessary precautions to save exported data containing sensitive information.
	This indicates supplemental instructio ns, best practices, tips, and other content that is good to know for the user.	 **Note:** You can use **Ctrl** + **A** to select all files.
>	Multi-level menu cascade.	**Settings** > **Network** > **Set network type**
Bold	It is used for buttons, menus, page names, and other UI elements.	Click **OK**.
`Courier font`	It is used for commands.	Run the `cd /d C:/windows` command to enter the Windows system folder.
`Italics`	It is used for parameters and variables.	`bae log list --instanceid` *`Instance_ID`*
[] or [a|b]	It indicates that it is a optional value, and only one item can be selected.	`ipconfig` *`[-all|-t]`*
{} or {a|b}	It indicates that it is a required value, and only one item can be selected.	`swich` *`{stand | slave}`*

Contents

1 Security features

1.1 Target users

This article is intended for MaxCompute project owners, administrators, and users interested in the MaxCompute multi-tenant data security system.

The MaxCompute multi-tenant data security system includes:

- User authentication.
- User and authorization management of projects.
- Sharing of resources across projects.
- Data protection of projects.

1.2 Quick Start

1.2.1 Use case: Add users and grant permissions

Description:

Jack is the project administrator of a project prj1. A new team member named Alice, who already has an Alibaba Cloud account as alice@aliyun.com, applies to join the prj1project. Alice requests the following permissions: view table lists, submit jobs, and create tables.

Solution:

As a project administrator, Jack performs the following procedure to add Alice as the user and grant her permissions to view table lists, submit jobs, and create tables:

```
    use prj1;
    add user aliyun$alice@aliyun.com; --Add the user
    grant List, CreateTable, CreateInstance on project prj1 to user
 aliyun$alice@aliyun.com; --Authorize the user by using the GRANT
statement
```

1.2.2 Use case: Add users and grant permissions using ACL

This article shows you how to add a project role and authorize it through ACLs.

Description:

Jack is the project administrator of a project prj1. The three new data auditors, Alice, Bob, and Charlie, are added to the project team. They all need to apply for the following permissions: view table lists, submit jobs, and read the table userprofile.

Solution:

As a project administrator, Jack can perform authorization by using the object-based *ACL Authorization*.

Jack must perform the following procedure:

```
    use prj1;
    add user aliyun$alice@aliyun.com; --Add the user
    add user aliyun$bob@aliyun.com;
    add user aliyun$charlie@aliyun.com;
    create role tableviewer; --Create a role
    grant List, CreateInstance on project prj1 to role tableviewer; --
Grant permissions to the role
    grant Describe, Select on table userprofile to role tableviewer;
    grant tableviewer to aliyun$alice@aliyun.com; --Grant the
tableviewer role to the user
    grant tableviewer to aliyun$bob@aliyun.com;
    grant tableviewer to aliyun$charlie@aliyun.com;
```

1.2.3 Use case: Project data protection

Description:

Jack is the project administrator of a project prj1. The project involves a large volume of sensitive data including user IDs, shopping records along with the data mining algorithms with proprietary intellectual property rights. Jack wants to protect the sensitive data and algorithms and allow only project users to access the data within the project. He also wants to make sure that data flows within the project only.

Solution:

To protect the project data, Jack must perform these steps:

```
    use prj1;
    set ProjectProtection=true; --Enable the project data protection
mechanism
```

Once the project data protection is enabled, data within the project cannot be transferred out of the project. All the data flows only within the project.

If users want to export data tables out of the project, an approval of the project administrator is needed. Here, MaxCompute provides the TrustedProject configuration to support external data

export from the protected project. In this case, configure project prj2 as a trusted project of prj1 and enable data flow from prj1 to prj2 through the following command:

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

1.3 User authentication

MaxCompute supports the **Alibaba Cloud account system** and the **RAM account system**.

Note:

MaxCompute recognizes the RAM account system but cannot recognize the RAM permission system. As a user, you can add any of your RAM sub-accounts to a MaxCompute project. However, MaxCompute skips the RAM permission definitions when it verifies the permissions of the RAM sub-account.

By default, the MaxCompute project only recognizes the Alibaba Cloud account system. You can view the account system supported by this project by running `list accountproviders;`.

Typically, only Alibaba Cloud accounts are displayed. To add the RAM account system, run the `add accountprovider ram;` command. After the RAM account system is added, run `list accountproviders;` to make sure it has been successfully added to the supported account systems.

Apply for an Alibaba Cloud account

If you do not have an *Alibaba Cloud account*, visit here to apply for one.

Note:

A valid email address is needed, when you apply for an Alibaba Cloud account. Because this email address is used as the account name after registration. For example, Alice can use her email address alice@aliyun.com to register an Alibaba Cloud account. Her account name will be alice@aliyun.com after Alibaba Cloud account registration.

Apply for AccessKey

Click here to create or manage your *AccessKey* list after you register an Alibaba Cloud account.

An AccessKey consists of the AccessKeyID and AccessKeySecret. The AccessKeyID is used to retrieve the AccessKey, and the AccessKeySecret is used to sign the computing messages. You

must secure your AccessKey for further use. If you need to update an AccessKey, create a new AccessKey and disable the existing one.

Log on to MaxCompute with an Alibaba Cloud account

Configure the AccessKey in the configuration file *conf/odps_config.ini* before you use odpscmd to log on. See the following example:

```
project_name=myproject
 access_id=<Input the AccessKeyID here, excluding the angle brackets>
access_key=<Input the AccessKey here, excluding the angle brackets>
end_point=http://service.odps.aliyun-inc.com/api
```

Note:

To enable or disable an AccessKey on the Alibaba Cloud website, wait for at least 15 minutes after the operation is complete.

1.4 User management

Any user, except the project owner, must be added to the MaxCompute project and granted the corresponding permissions to manage data, jobs, resources, and functions in MaxCompute. This article describes how a project owner can add, authorize, and remove other users, including RAM sub-accounts to MaxCompute.

If you are a project owner, we recommend that you read this article carefully. If you are a typical user, we recommend that you submit an application to the project owner to be added to the corresponding project. We recommend all users to read the subsequent sections.

All the operations mentioned in this article are executed on the console. For Linux, run **./bin/odpscmd** and for Windows, run **./bin/odpscmd.bat**.

Add a user

In this example, the project owner, Alice, wants to authorize another user, therefore she must add the user to the project first. **Only a user who has been added to the project can be authorized**.

The command to add a user is as follows:

```
add user
```

The <username> of an Alibaba Cloud account is a valid email address registered with Alibaba Cloud, or a RAM sub-account of an Alibaba Cloud account that runs the command. For example:

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

```
add user RAM$ram_test_user;
```

Assume that the Alibaba Cloud account of Alice is alice@aliyun.com. When Alice runs these statements, the following results are returned by running the `list users;` command:

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

This indicates that the Alibaba Cloud account odps_test_user@aliyun.com and the sub-account ram_test_user created by Alice using RAM have been added to the project.

Add a RAM sub-account

The two ways to add a RAM sub-account are as follows:

- By using DataWorks, for more information, see *Prepare a RAM account*.
- By using MaxCompute client commands as described in this document.

> **Note:**
>
> - MaxCompute only allows a primary account to add its own RAM sub-accounts to a project. RAM sub-accounts of other Alibaba Cloud accounts are not allowed. Therefore, you can skip to specify the name of the primary account before the RAM sub-accounts when `add user`. MaxCompute determines by default that the account which runs the command is the corresponding sub-account.
> - MaxCompute only recognizes the RAM account system and does not recognize the RAM permission system. Users can add any of their RAM sub-accounts to a MaxCompute project, but MaxCompute does not consider the permission limits in RAM when performing permission verification of RAM sub-accounts.

By default, MaxCompute project only recognizes Alibaba Cloud account systems. To view the supported account systems use the `list accountproviders;` command. Typically, only the ALIYUN account is visible, for example:

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

> **Note:**
>
> Only the project owner has the permission to perform operations related to `accountproviders`.

As shown in the preceding command, you can only see the `ALIYUN` account system. If you want to add RAM accounts support, run the `add accountprovider ram;` as follows:

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

The user will still not be able to operate MaxCompute successfully. This is because, the user must be granted certain permissions to operate MaxCompute within the permissive limits. For more information, see *Authorization*.

User Authorization

Once the user is added, the project owner or project administrator must authorize the user. The user can perform the operations only after obtaining the permissions.

MaxCompute provides ACL authorization, cross-project resource sharing, and project resource protection. The following are two common scenarios, for more information, see *ACL Authorization* .

Scenario 1

In the following scenario, Jack is the administrator of the project prj1. A new project team member Alice (Alibaba Cloud account: alice@aliyun.com) applies to join the project prj1, and for permission to view table lists, submit jobs, and create tables.

The admin or the project owner can run the following command on the client:

```
use prj1; --Open the project prj1
add user aliyun$alice@aliyun.com; --Add the user
grant List, CreateTable, CreateInstance on project prj1 to user aliyun
$alice@aliyun.com; --Authorize the user
```

Scenario 2

In the following scenario, assume Alibaba Cloud account user (bob@aliyun.com) has been added to a project ($user_project_name), and must be granted permission to create tables, obtain table information, and run functions.

The admin or the project owner can run the following command on the client:

```
grant CreateTable on PROJECT $user_project_name to USER ALIYUN$bob@
aliyun.com;
  --Grant CreateTable permission on project “$user_project_name” to
bob@aliyun.com
grant Describe on Table $user_table_name to USER ALIYUN$bob@aliyun.com
;
  --Grant Describe permission on table “$user_table_name” to bob@
aliyun.com
grant Execute on Function $user_function_name to USER ALIYUN$bob@
aliyun.com;
```

```
  --Grant Run permission on function "$user_function_name" to bob@
aliyun.com
```

Authorize RAM Sub-account

To check accounts support, run `list accountproviders;` command as follows:

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

In this project, RAM accounts are also supported. You can add a RAM sub-account to this project and grant `Describe` permission on the tables. For example:

```
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
```

After running these commands, *`Alice`* account, which is a RAM sub-account of *`bob@aliyun.com`* , can logon to MaxCompute with the **AccessKeyID** and **AccessKeySecret**, and run `desc` on the table *`src`*.

Note:

- For more information about how to create a RAM sub-account `AccessKeyID` and `AccessKeySecret`, see *RCreate a RAM user*.
- For more information about how to add or remove users on MaxCompute, see the correspond ing content of this article.
- For more information about authorizing a user, see *Authorization*.

Remove a User

When a user leaves the project team, Alice must remove the user from the project. Once removed from the project, the user no longer has any access permission to the project resources.

The command to remove a user from a project is as follows:

```
remove user
```

Note:

- A user removed from a project immediately loses an authority to access resources of the project.
- Revoke all the roles of the user, before removing a user whom the roles are assigned. For more information about roles, see *Role Management*.

- After a user is removed, all*ACL Authorization* data related to the user is retained. After a user is added to a project again, the ACL Authorization of this user is enabled again.
- MaxCompute does not support complete removal of a user and all permission data from a project.

To remove corresponding users, Alice can run the following commands:

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

To make sure the users are removed, run the following command:

```
LIST USERS;
```

If those two accounts are no longer listed after running the command, it indicates that the accounts have been removed from the project.

Remove a RAM Sub-account

Similarly, RAM sub-account can be removed by using the `remove user` command. For example:

```
odps@ ****>revoke describe on table src from user ram$bob@aliyun.com:
Alice;
OK
-- Revoke Alice sub-account permissions
odps@ ****>remove user ram$bob@aliyun.com:Alice;
Confirm to "remove user ram$bob@aliyun.com:Alice;" (yes/no)? yes
OK
-- Remove sub-account
```

If you are the project owner, you can also remove the RAM account system from the current project by `remove accountprovider` as follows:

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

1.5 Role management

A role is a defined set of access permissions. It assigns the same set of permissions to a group of users. Role-based authorization greatly simplifies the authorization process and reduces the authorization management cost. It must be used with priority.

When a project is created, an admin role is automatically created with a definite privilege authorized to the role, including access to all objects within the project, management of users

and roles, and authorization to users and roles. In comparison to a project owner, the admin role cannot assign admin permission to any user, set the project security configuration, or change the authentication model for the project. Permissions of the admin role cannot be modified.

Role management related commands include the following:

```
create role <rolename> --Create a role
drop role <rolename> --Delete a role
grant <rolename> to <username> --Grant a role to a user
revoke <rolename> from <username> --Revoke a role from a user
```

Note:

- One role can be assigned to multiple users at the same time, and one user can be assigned multiple roles.
- For more information about the mapping between the roles in DataWorks and in MaxCompute, and the platform permissions of these roles, see the project member management module in *Project Management*.

Create a role

To create a role, use the following command :

```
CREATE ROLE;
```

Example:

To create a role player, enter the following command on the client:

```
create role player;
```

Note:

The role permissions you create can view the specified user permissions through *Permission check*.

Add a user to the role

To add a user to the role, use the following command:

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

Example:

To assign user bob@aliyun.com the player role, enter the following command on the console:

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

Authorize role

The authorization statement for the role is similar to the authorization for the user. For more information, see *User authorization*.

> **Note:**
> After role authorization is complete, all users under this role have the same permissions.

Example:

Jack is the administrator of project prj1. Three new data auditors, Alice, Bob, and Charlie, are added to the project team. They must apply for the following permissions: view the table lists, submit the jobs, and read the table userprofile.

In this scenario, the project administrator can perform authorization by using the object-based *ACL Authorization*.

The commands are as follows:

```
    use prj1;
    add user aliyun$alice@aliyun.com; --Add the user
    add user aliyun$alice@aliyun.com; --Add the user
    add user aliyun$charlie@aliyun.com;
    create role tableviewer; --Create a role
    grant List, CreateInstance on project prj1 to role tableviewer; --
Grant permissions to the role
    grant Describe, Select on table userprofile to role tableviewer;
    grant tableviewer to aliyun$alice@aliyun.com; --Grant the
tableviewer role to the user
    grant tableviewer to aliyun$bob@aliyun.com;
    grant tableviewer to aliyun$charlie@aliyun.com;
```

Revoke the role from the user

To revoke the role from the user, use the following command:

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

Example:

To remove the user bob@aliyun.com from the player role, use the following command on the client:

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

Delete a Role

To delete a role, use the following command:

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

Example:

To delete the role of the player, use the following command:

```
drop role player;
```

Note:
When a role is deleted a role, MaxCompute checks whether other users are in this role. If yes, this role cannot be deleted. The role can be successfully deleted only when all users in the role are revoked from this role.

1.6 Authorization

Authorization allows a user to perform operations including read, write, and view on tables, tasks, resources, and other objects of the MaxCompute. After the *user* is added, the project owner or the project administrator must authorize the user. The user can perform operations only after obtaining the permission.

MaxCompute provides Access Control List (ACL) authorization, cross-project resource sharing, and project resource protection. Authorization typically includes three elements: subject, object, and action. In MaxCompute, the subject refers to a user or a role and the object refers to various types of objects in a project.

ACL authorization includes following MaxCompute objects: *Project*,*Table* ,*Function* , *Resource*, and *Instance*. Operations are related to specific object types, therefore different types of objects support different types of actions.

MaxCompute projects support the following object types and actions:

Object	Action	Description
Project	Read	View project information (excluding any project objects), such as the creation time.

Object	Action	Description
Project	Write	Update project information (excluding any project objects), such as comments.
Project	List	View the list of all types of objects in the project.
Project	CreateTable	Create a table in the project.
Project	CreateInstance	Create an instance in the project.
Project	CreateFunction	Create a function in the project.
Project	CreateResource	Create a resource in the project.
Project	All	Grant all of the preceding permissions.
Table	Describe	Read the metadata of the table.
Table	Select	Read the table data.
Table	Alter	Change the metadata of the table and add or delete a partition.
Table	Update	Overwrite or add table data.
Table	Drop	Delete a table.
Table	All	Grant all the preceding permissions.
Function	Read	Read and run permissions.
Function	Write	Update.
Function	Delete	Delete.
Function	Run	Run.
Function	All	Grant all the preceding permissions.
Resource	Read	Read.
Resource	Write	Update.
Resource	Delete	Delete.
Resource	All	Grant all the preceding permissions.
Instance	Read	Read.
Instance	Write	Update.
Instance	All	Grant all the preceding permissions.

 **Note:**

- The CreateTable action for the objects of Project type must work with the CreateInstance permission for the Project object. The Select, Alter, Update, and Drop actions for the objects of Table type must work with the CreateInstance permission for the Project object.
- If the CreateInstance permission is not granted, the corresponding operations cannot be performed even though the mentioned permissions are granted. This is related to the internal implementation of MaxCompute. The Select permission for Table type objects must work with the CreateInstance permission. While performing cross-project operation, such as selecting the table of project B in the project A, you must have the project A CreateInstance and the project B Table select permissions.
- After a user or role is added, you must grant permissions to the user or role. MaxCompute authorization is an object-based authorization method. The permission data authorized by ACL is considered as a type of sub-resource of the object. Authorization can be performed only if the object exists. When the object is deleted, the authorized permission data is automatically deleted.

- **SQL92 Authorization**

  MaxCompute supports authorization using the syntax similar to the GRANT and REVOKE commands defined by SQL92. It grants or revokes permissions to/from the existing project object through simple authorization statements. The authorization syntax is as follows:

```
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
```

  Users familiar with GRANT and REVOKE commands defined by SQL92 or with Oracle database security management can identify that the ACL authorization syntax of MaxCompute does not support [WITH GRANT OPTION] authorization parameters. For example, when User A authorizes User B to access an object, User B cannot grant the permission to User C. In this scenario, all permissions can be granted by one of the following three roles:

  - Project owner
  - Project administrator
  - Object creator

- **Use example of ACL authorization**

  In the following scenario, the Alibaba Cloud account user alice@aliyun.com is a newly added member to the project test_project_a, and Allen is a RAM-sub account added to bob@aliyun.

com. In test_project_a, they both must submit jobs, create tables, and view existing objects in the project.

The project administrator bob performs the following authorization operations:

```
    use test_project_a;
    add user aliyun$alice@aliyun.com;
    add user ram$bob@aliyun.com:Allen;
    create role worker;
    grant worker TO aliyun$alice@aliyun.com;
    grant worker TO ram$bob@aliyun.com:Allen;
    grant CreateInstance, CreateResource, CreateFunction, CreateTabl
e, List ON PROJECT test_project_a TO ROLE worker;
```

- **Cross-project Table/Resource/Function sharing**

Following the preceding example, aliyun$alice@aliyun.com and ram$bob@aliyun.com:Allen have certain permissions in test_project_a. These two users must query table prj_b_test_table in test_project_b, and use test_project_b. UDF prj_b_test_udf.

The project administrator performs the following authorization operations for test_project_b:

```
    use test_project_b; --Open the project
    add user aliyun$alice@aliyun.com; --Add the user
    add user ram$bob@aliyun.com:Allen; --Add th RAM sub-account
    create role prj_a_worker; --Create a role
    grant prj_a_worker TO aliyun$alice@aliyun.com; --Grant the role
    grant prj_a_worker TO ram$bob@aliyun.com:Alice; --Grant the role
    grant Describe , Select ON TABLE prj_b_test_table TO ROLE
prj_a_worker; --Authorize the role
    grant Read ON Function prj_b_test_udf TO ROLE prj_a_worker; --
Authorize the role
    grant Read ON Resource prj_b_test_udf_resource TO ROLE
prj_a_worker; --Authorize the role
--After authorization, the two users query table and use udf in
test_project_a as follows:
use test_project_a;
select test_project_b:prj_b_test_udf(arg0, arg1) as res from
test_project_b.prj_b_test_table;
```

Note:

If UDF is created in test_project_a, then only Resource authorization is required. Use the following code:

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

1.7 Permission check

MaxCompute provides the ability to view multiple permissions, including the permissions of certain users or roles, and authorization lists of specified objects.

MaxCompute uses the markup characters A, C, D, and G when showing the permissions of users or roles. The meanings of these markup characters are as follows:

- A: Access allowed.
- D: Access denied.
- C: Access granted with conditions. It appears only in a policy authorization system.
- G: Access granted with conditions. Permission can be granted to objects.

An example of viewing permissions is as follows:

```
    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 | CreateFunc
tion | 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
```

Note:

Currently, desc role only displays ACL information of project and table authorization types, while ACL of other objects (function, resource, instance, job) does not support display.

View permissions of a specified user

```
    show grants; --View permissions of the current user.
    show grants for <username>; --View access permissions of a
specified user. The operation can be executed by project owners and
administrators.
```

Example:

To view the user Alibaba Cloud account bob@aliyun.com permissions in the current project, run the following command on the client:

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

To view RAM sub-account permissions:

```
show grants for RAM$account:sub-account;
```

Example:

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

View permissions of a specified role:

```
describe role --View access permissions granted to a specified role
```

Note:
In the public cloud environment, description role currently only displays ACL information of the object authorization type of project and table, while ACL information of other objects (such as function, resource, instance, job) is not displayed.

View the authorization list of a specified object:

```
show acl for [on type ];--View the user and role authorization list of
 a specified object
```

Note:
When *[on type <objectType>]* is excluded, the default type is Table.

1.8 Security configurations

MaxCompute is a multi-tenant data processing platform. Distinct tenants have distinct data security requirements. Therefore, MaxCompute provides project-level security configurations to comply with the unique requirements of individual tenants. Project owners can customize their external account support and authentication models.

MaxCompute provides multiple methods of orthogonal authorization, including Access Control List (ACL) authorization and implicit authorization. An object creator is automatically granted the object access permission. Not all users need these security features. Users can properly configure the project authentication model based on their service security requirements and usage patterns.

```
    show SecurityConfiguration
      --View the project security configuration.
    set CheckPermissionUsingACL=true/false
      --Enable/Disable the ACL authorization mechanism. The default
value is true.
    set ObjectCreatorHasAccessPermission=true/false
      --Enable/Disable automatic access permission granting to object
creators. The default value is true.
    set ObjectCreatorHasGrantPermission=true/false-* +
      --Enable/Disable automatic authorization permission granting to
object creators. The default value is true.
    set ProjectProtection=true/false
        --Enable/Disable project data protection to enable/disable
data transfer from the project.
```

Note:

You can also complete the security configuration of a project in a visualized technique using DataWorks. For more information, see *Project Management*.

1.9 Data protection of projects

Background and motivation

Some companies (including financial institutions, military enterprises and so on) are extremely sensitive to data security. Hence, to secure the data, additional security measures are taken, that include not allowing employees to carry USB storage devices or personal hard disks to work; or most of the times the USB ports are disabled. Employees are not allowed to work from home. All these measures are taken to secure the sensitive data.

As a MaxCompute Project Space Administrator, do you have similar security requirements, where users are not allowed to move data out of the project space?

For example, when the owner of Project Space prj1 encounters this situation as shown in the following figure, are you worried that user Alice will transfer the data that she can access to prm9, only because she has access to prj2. prj2. and prj2?

More specifically, assume that Alice has been granted access to myprj, which is the Select permission for Table1, and then she is also granted create table permission by the administrator of prj2.

By these permissions, Alice is able to transfer the data to prm9 in any of the following ways:

- Submit SQL:

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

- Write MapReduce to read myprj. Table1 and write to the scanner.

If the data in your project space is sensitive, you will be restricted to share data out of your project . MaxCompute can resolve issues pertaining to data protection and the aforementioned operations as well.

Data protection feature

MaxCompute provides a project space protection feature that helps to resolve issues mentioned earlier. As a user, set the project as follows:

```
    set projectProtection=true
      -- Set project protection rule: data can only flow and cannot
flow out
```

When project protection is set up, the data flow in your project space is controlled , "Data can only flow and cannot flow out ". That is, both of these actions will fail because they are against the project protection rule.

By default, ProjectProtection cannot be set and its value is false.

Also, users authorized to access multiple projects can freely use cross-project data access operations to share or transfer project data. If users are highly sensitive to project data security, the administrator must define a ProjectProtection feature likewise.

Data outflow method after enabling data protection

After setting ProjectProtection in the user's project, the user may soon make requests such as Alice applies to the user for exporting the data of a table out of the user’s project.

Moreover, user review confirms that this table does not contain sensitive data. In order not to affect Alice's normal business requirements, MaxCompute provides two data export methods to the user after setting ProjectProtection.

- **Set TrustedProject**

In case, the current project space is protected, and if you set the target space for the data inflows to the trustedproject for the current space. Then, the data flow to the target project space will not be considered a violation of the project protection rule. If multiple project spaces are set to trustedproject between two and one another, so these project spaces form a trustedproject.

Group; the data can flow within the project group, but restricted to be shared out of the project group.

Use the following command to manage the TrustedProject:

```
list trustedprojects;
  -- View All trustedprojects in the current project
add trustedproject <projectname>;
  -- Add a trustdproject to the current project
remove trustedproject <projectname>;
  -- Remove a trustdproject from the current project
```

- **Resource sharing and data protection**

  In MaxCompute, the *package-based resource sharing* feature and the project protection data protection feature are orthogonal, but they are similar to each other in terms of functions.

  MaxCompute rules **give priority to resource sharing over data protection**. Therefore, if a data object allows access by users from other projects through resource sharing, the ProjectProtection rules will not apply to this data object.

Best practices

To prevent data outflow from the project, after setting `ProjectProtection=true`, check the following settings:

- Make sure the trustedproject is not added. If set, you must assess possible risks;
- Make sure that package data is not used for sharing. If set, make sure that no sensitive data exists in the package.

1.10 Column-level access control

Label-based security (LabelSecurity) is a required MaxCompute Access Control (MAC) policy at the project space level. It allows project administrators to control the user access to column-level sensitive data with improved flexibility.

Difference between MAC and DAC in MaxCompute

In MaxCompute, MAC is independent of Discretionary Access Control (DAC). Two examples are provided to illustrate the differences between MAC and DAC.

To drive a vehicle, you must first have to apply and acquire a valid driver's license, similarly, a user who wants to read data in a MaxCompute project must first apply for the SELECT permission. The permission application is within the scope of DAC.

Because the country with a high accident rate, drunk driving is strictly restricted. To curb this , all drivers are required to have a driver’s license and must not drink and drive. Likewise, in MaxCompute, reading highly sensitive data is analogous to the law against drunk driving. The read prohibition is within the scope of MAC.

Data sensitivity classification

LabelSecurity assigns security levels to data and the users who access the data. In the government and financial sectors, data sensitivity is usually classified into four levels: 0 (Unclassifi ed), 1 (Confidential), 2 (Sensitive), and 3 (Highly Sensitive). MaxCompute adopts such classifica tion. Project owners must define standards for data sensitivity classification and access level classification. The default access level of all users is 0, and the default sensitivity level of data is 0.

LabelSecurity supports data sensitivity classification at the column level. Administrators can set sensitivity labels for all the columns of a table. A table may have columns of different sensitivity levels.

Administrators can also set sensitivity labels for views. A view and its base table have independen t sensitivity labels. The default sensitivity level of a new view is 0.

Default security policies of LabelSecurity

LabelSecurity applies the following default security policies to the data and users assigned with sensitivity or security labels:

- No-ReadUp: A user is not allowed to read data with a sensitivity level higher than the user level unless the user is explicitly authorized.
- Trusted-User: A user is allowed to write data of all sensitivity levels. The default sensitivity level of new data is 0 (unclassified).

Note:

- In some traditional MAC systems, other complex security policies are applied to prohibit unauthorized data distribution in a project. For example, the No-WriteDown policy prohibits users from writing data with a sensitivity level not higher than the user level. By default,

MaxCompute does not support No-WriteDown, considering the costs involved in managing the data sensitivity levels of project administrators. The effect of No-WriteDown can be attained by modifying the project security settings (`Set ObjectCreatorHasGrantPermission=false`).

- To prohibit data flowing among different projects, you can set the projects to the protected state (ProjectProtection). With the setting, users can only access the data within their projects. This prevents data transfer or data sharing outside the project.

By default, projects disable LabelSecurity. The project owners can enable it as required.

After LabelSecurity is enabled, the default security policies are executed. When a user accesses a data table, the user must have the SELECT permission and the access level required for sensitive data reading. Compliance with LabelSecurity is a required but not the sufficient condition for passing CheckPermission.

LabelSecurity operations

- **Enable or disable LabelSecurity**

```
 Set LabelSecurity=true|false;
   -- Enables or disables LabelSecurity. The default value is false.
   -- LabelSecurity can be enabled or disabled only by the project
owner. Other operations can be performed by the project administra
tor.
```

- **Set security labels for users**

```
 SET LABEL <number> TO USER <username>;-- Value range of "number": [
0, 9]. This operation can be performed only by the project owner or
administrator.
 -Example:
 ADD USER aliyun$yunma@aliyun.com; --Adds a user with the default
security label 0.
 ADD USER ram$yunma@aliyun.com:Allen; --Adds user Allen, which is a
RAM subaccount of yunma@aliyun.com.
 SET LABEL 3 TO USER aliyun$yunma@aliyun.com;
   -- Sets the security label of yunma to 3 to allow this user to
access only the data with a sensitivity level not higher than 3.
 SET LABEL 1 TO USER ram$yunma@aliyun.com:Allen;
   -- Sets the security label of subaccount Allen to 1 to allow this
 user to access only the data with a sensitivity level not higher
than 1.
```

- **Set sensitivity labels for data**

```
 SET LABEL <number> TO TABLE tablename(column_list); -- Value range
 of "number": [0, 9]. This operation can be performed only by the
project owner or administrator.
 -Example:
 SET LABEL 1 TO TABLE t1;  --Sets the sensitivity label of table t1
to 1.
 SET LABEL 2 TO TABLE t1(mobile, addr); --Sets the sensitivity
labels of the "mobile" and "addr" columns of table t1 to 2.
```

```
 SET LABEL 3 TO TABLE t1;  --Sets the sensitivity label of table t1
to 3.  The sensitivity labels of the "mobile" and "addr" columns are
 still 2.
```

Note:

The sensitivity labels explicitly set for the columns overwrite the sensitivity label set for the table, without considering the label setting order and the sensitivity level.

- **Explicitly authorize lower-level users to access specific data tables with a high sensitivity level**

```
 --Grant permissions:
 GRANT LABEL <number> ON TABLE <tablename>[(column_list)] TO USER <
username> [WITH EXP <days>]; --The default validity period is 180
days.
 -- Revoke the permissions:
 REVOKE LABEL ON TABLE <tablename>[(column_list)] FROM USER <
username>;
 -- Clear the expired permissions:
 CLEAR EXPIRED GRANTS;
 -Example:
 GRANT LABEL 2 ON TABLE t1 TO USER ram$yunma@aliyun.com:Allen WITH
EXP 1; --Explicitly authorizes Allen to access the data of table t1
with a sensitivity level not higher than 2 for a period of 1 day.
 GRANT LABEL 3 ON TABLE t1(col1, col2) TO USER ram$yunma@aliyun.com
:Allen WITH EXP 1; --Explicitly authorizes Allen to access the data
 in col1 and col2 of table t1 with a sensitivity level not higher
than 3 for a period of 1 day.
 REVOKE LABEL ON TABLE t1 FROM USER ram$yunma@aliyun.com:Allen; --
Revokes the permission of Allen to access the sensitive data in
table t1.
```

Note:

Once the label-authorized permission of a user to access a table is revoked, the permission to access the table fields of the same user is also revoked.

- **List the sensitive data sets that a user can access**

```
 SHOW LABEL [<level>] GRANTS [FOR USER <username>];
    --When [FOR USER <username>] is unspecified, the system lists
the sensitive data sets that the current user can access.
   --When <level> is unspecified, the system lists the permissions
 granted by all label levels.When <level> is specified, the system
lists only the permissions granted by a specific label level.
```

- **List the users who can access a specific table containing sensitive data**

```
 SHOW LABEL [<level>] GRANTS ON TABLE <tablename>;
   --Displays the label-authorized permissions on the specified
table.
```

- **List the label-authorized permissions of a user at all levels to access a data table**

```
 SHOW LABEL [<level>] GRANTS ON TABLE <tablename> FOR USER <username
>;
```

```
   --Displays the label-authorized permissions of the specified user
 to access the columns of a specific table.
```

- **List the sensitivity levels of all the columns of a table**

```
DESCRIBE <tablename>;
```

- **Control the access level of a package installer regarding the sensitive resources of the package**

```
 ALLOW PROJECT <prjName> TO INSTALL PACKAGE <pkgName> [USING LABEL <
number>];
   --The package creator grants an access level to the package
installer regarding the sensitive resources of the package.
```

Note:

- When `[USING LABEL <number>]` is unspecified, the default access level is 0. The package installer can only access non-sensitive data.
- When accessing to sensitive data across projects, the access level defined by this command applies to all the users in the project of the package installer.

LabelSecurity use cases

- **Prohibit all the users in a project except the project administrator from reading some sensitive columns of a table**

  Description:

  user_profile is a table with sensitive data in a project. It has 100 columns, five of which contain sensitive data: id_card, credit_card, mobile, user_addr, and birthday. DAC grants all users the SELECT permission on this table. The project owner wants to prohibit all the project users except the project administrator from reading the sensitive columns of the table.

  To achieve this purpose, the project owner can perform the following operations:

```
set LabelSecurity=true;
  --Enables LabelSecurity.
set label 2 to table user_profile(mobile, user_addr, birthday);
  --Sets the sensitivity level of the specified columns to 2.
set label 3 to table user_profile(id_card, credit_card);
  --Sets the sensitivity level of the specified columns to 3.
```

Note:

After the preceding operations, non-administrator users cannot access the data in the five columns. To access the sensitive data for business purposes, the user must be authorized by the project owner or administrator.

Solution:

Alice is a member of the project. For official purposes, she wants to apply for access to the data in the mobile column of table user_profile for a period of one week. To authorize Alice, the project administrator can perform the following operation:

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

Note:

Mobile, user_addr, and birthday column contain data with a sensitivity level of 2. Birthday. After authorization, Alice can access the data in these three columns. The authorization causes the issue of excessive permission grants. This issue can be avoided if the project administrator sets the sensitive columns properly.

- **Prohibit the project users with access to sensitive data from copying and distributing the sensitive data within the project without authorization**

  Description:

  In the preceding use case, Alice is granted the access permission on the data with a sensitivity level of 2 for official purposes. The project administrator worries that Alice may copy that data from table user_profile to table user_profile_copy created by her and grants Bob the access permission on user_profile_copy. The project administrator needs a method to restrict Alice's actions.

  Solution:

  Considering security usability and management costs, LabelSecurity adopts the default security policy that allows for WriteDown. Users can write data to the columns with a sensitivity level not higher than the user level. MaxCompute cannot address the preceding requirement of the project administrator. However, the project administrator can restrict the discretionary authorization behavior of Alice by allowing her to only access the data she created, but disallowing her to grant the data access permission to other users. The procedure is as follows:

```
 SET ObjectCreatorHasAccessPermission=true;
   --Allows the object creator to operate objects.
 SET ObjectCreatorHasGrantPermission=false;
   --Prohibits the object creator from granting the object access
permission to other users.
```

1.11 Resource share across project space

1.11.1 Resource sharing across projects based on package

Assume that you are the project owner or administrator (admin role) of a few projects. One of your primary accounts has multiple projects, wherein the project prj1 has some resources (including tables, resources, and custom functions) that can be shared with other projects. However, adding users of other projects to prj1 and granting permissions to them one by one is complicated, and adding the users who are irrelevant but are added to the prj1 project (if they exist) complicates the project management.This section describes cross-project resource sharing.

If resources must be controlled by the user in a fine-grained manner, and the user who applies for the control permission is a member of the business project team, we recommend using the *Project user and authorization management* feature.

Package is used for sharing data and resources across projects. It solves the problem of cross-project user authorization.

Use package to solve the following problems effectively:

If members of the Alifinance project want to access data in the Alipay project, the administrator of the Alipay project must perform tedious authentication operations: First, add users in the Alifinance project to the Alipay project, and then perform general authentications on the newly added users, respectively.

Actually, the administrator of the Alipay project does not want to authenticate and manage all users in the Alifiance project. Instead, the administrator expects more efficient feature for autonomous authentication controls over permissive objects.

After Package is used, the administrator of the Alipay project can perform packaging authorization on the objects to be used by the Alifinance project (that is, create a Package), and then permit the Alifinance project to install the Package. After the Alifinance project's administrator installs the Package, the administrator can determine whether to grant permissions of the Package to the users of the Alifinance project as required.

1.11.2 Package usage method

This article introduces you to the operations involved in the project space Package creator and Package consumer.

Package usage method

The use of package involves two subjects: the package creator and the package user.

- The package creator provides the resources to be shared and the permissions to access it. It also allows the package user to install and use it.

- The package user uses the package. After the package is published, the user can directly access the resource across projects.

The following is a description of the operations involved with the package creator and package user.

Package creator

- Create package

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

Note:

- Only the project owner has the permission to create a package.
- The name of the package cannot exceed 128 characters.

- Add a resource to be shared to the package

```
    Add project_object to package package_name [with privileges] --
add objects to package
    Remove project_object from package package_name; -- remove
object from package
    project_object ::= table table_name |
                       instance inst_name |
                       function func_name |
                       resource res_name
    privileges ::= action_item1, action_item2, ...
```

Note:

- Currently, supported types of objects exclude projects. Therefore, you cannot use a package to create objects in other projects.
- The objects themselves and the permission to perform operations on them are added to the package at the same time. When not passed (with privileges) even specifying an action permission, the default is read-only, that is, read/describe/select. The object and its permissions are treated as a whole and cannot be updated once added. If necessary, you can only delete and re-add.
- When an object is added to a package, it is not packaged as a snapshot, so subsequent object data changes, and access to the object through package authorization is also the current data of the object.

Use the following commands to perform various operations on the package:

- Allow other projects to use a package

```
allow project <prjName> to install package <pkgName> [using label <
num>]
```

- Revoke other projects' permission to use a package

```
disallow project <prjName> to install package <pkgName>
```

- Drop a package

```
Delete package <pkgname>;
```

- View the list of packages already created and installed

```
Show packages;
```

- View package details

```
Describe package <pkgname>;
```

Package users

- Install package

```
Install package <pkgname>;
```

For package installation, the pkgName format is: <projectName>.<packageName>.

Note:

Only the project owner has permissions to perform this operation.

- Uninstalling package

```
Uninstall package <pkgname>;
```

For package installation, the pkgName format is: <projectName>.<packageName>.<projectName>.<packageName>

- View a package

```
    Show packages;
    View the list of packages already created and installed
    Describe package <pkgname>;
    View details of package
```

- Client project grants access to package to other members or role of this project

The installed package is an independent type of MaxCompute object. To access resources in a package (resources shared with you by other projects), you must have the permission to read package.

If you do not have the Read permission, you must apply to the project owner or admin for the permission. The project owner or admin can grant permissions through ACL authorization or policy authorization.

Authorize package to user or role:

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

Note:

After authorization, user has access to the object in that package only in this project.

For example, the following ACL authorization allows the cloud account user odps_test@aliyun.com to access resources in the package:

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

]

Or allow all members of role role_dev to access resources in package:

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

Example

Jack is the administrator of prj1. John is the administrator of prj2. To address some business needs, Jack wants to share some resources of prj1 (such as datamining.jar and sampletable) to John's prj2. If prj2 user Bob must access these resources, the prj2 administrator can self-authorize Bob through ACL administrator or policy authorization without Jack’s involvement.

Procedure:

1. Prj1 administrator Jack creates resources package in prj1.

```
    Use prj1;
    Create package datamicing; -- creating a package
    Add Resource dating. jar to package dating;-add resource to
package
    Add Table sampletable to package dating; -- adding table to
package
```

```
    Allow project prm9 to install package dating; -- sharing package
 to Project Space prm9
```

2. Prj2 administrator Bob installs a package in prj2.

```
    use prj2;
    install package prj1.datamining; -- installs a package
    describe package prj1.datamining; -- view a list of resources in
 the package
```

3. Bob self-authorizes the package.

```
    use prj2;
    grant Read on package prj1.datamining to user aliyun$bob@aliyun.
com; -- authorization of Bob to use package via ACL
```

1.12 Security command list

1.12.1 Security configuration of a project

This article introduces you to the concept of authentication configuration and data protection in some project space security configurations.

Authentication configuration

Statement	Description
show SecurityConfiguration	View the security configuration of the project.
set CheckPermissionUsingACL=true/false	Enable/Disable the ACL-based authorization.
set CheckPermissionUsingPolicy=true/false	Enable/Disable the policy authorization.
set ObjectCreatorHasAccessPermission=true/false	Grant/Revoke default access permissions to/from object creators.
set ObjectCreatorHasGrantPermission=true/false	Grant/Revoke default authorization permissions to/from object creators.

Data protection

Statement	Description
set ProjectProtection=false	Disable data protection.
list TrustedProjects	View the list of trusted projects.
add TrustedProject <projectName> `<projectName>`	Add a trusted project.
remove TrustedProject `<projectName>`	Remove a trusted project.

1.12.2 Manage permissions

This article introduces you to the related concepts of user management, role management, ACL authorization, and permission review in project space rights management.

Manage users

Statement	Description
list users	View all users added to the project.
add user <username> `<username>`	Add a user.
remove user <username> `<username>`	Remove the user.

Manage roles

Statement	Description
list roles	View all created roles.
create role <rolename> `<rolename>`	Create a role.
drop role <rolename> `<rolename>`	Delete a role.
grant `<rolelist>` to `<username>`	Assign one or multiple roles to the user.
revoke `<rolelist>` from `<username>`	Revoke a role from the user.

ACL Authorization

Statement	Description
grant `<privList>` on `<objType>` `<objName>` to user `<username>`	Authorize a user.
grant `<privList>` on `<objType>` `<objName>` to role `<rolename>`	Authorize a role.
revoke `<privList>` on `<objType>` `<objName>` from user `<username>`	Revoke user authorization.
revoke `<privList>` on `<objType>` `<objName>` from role `<rolename>`	Revoke role authorization.

Permission review

Statement	Description
whoami	View current user information.
show grants [for `<username>`] [on type `<objectType>`]	View user role and permissions.

Statement	Description
show acl for <objectName> [on type <objectType>]	View specific object authorization information.
describe role <roleName>	View role authorization information and role assignments.

1.12.3 Package-based resource sharing

This article gives you a description of resource sharing statements based on Package.

Share resources

Statement	Description
Create package <pkgname> <pkgName>	Create a package.
Delete package <pkgname> <pkgName>	Delete a package.
add <objType> <objName> to package <pkgName> [with privileges privs]	Add resources to be shared to a package.
remove <objType> <objName> from package <pkgName>	Remove shared resources from a package.
allow prOject <prjName> to install package <pkgName> [using label <num>]	Allow a project to use a user package.
disallow project <prjName> to install package <pkgName>	Disallow a project from using a user package.

Use Resources

Statement	Description:
Install package <pkgname> <pkgName>	Install a package.
uninstall package <pkgName>	Uninstall a package.

View a package

Statement	Description:
show packages	List all created and installed packages.
describe package <pkgName>	View details of a package.

Note:

If you execute a production plan authorization related request in DataWorks：

1. Project owner is executed by temporary query and cannot be submitted to the production environment for execution.Because the production environment is executed by the production account, which has no authorized authority.
2. Add the **use<production project>;** statement before the query and submit it with the command. Because DataWorks data development defaults the current project is the development project ending in _dev. When executing authorization commands from the command line, ask project owner to execute them first:

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